

GNSS-SDR

0.0.14

Generated by Doxygen 1.8.14



# Contents

<b>1</b>	<b>Main Page</b>	<b>1</b>
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
1.7	About the software license	12
1.8	Publications and Credits	13
1.9	Ok, now what?	14

<b>2</b>	<b>Reference Documents</b>	<b>15</b>
2.1	Interface Control Documents . . . . .	15
2.1.1	GPS . . . . .	15
2.1.2	GLONASS . . . . .	15
2.1.3	Galileo . . . . .	16
2.1.4	BeiDou . . . . .	16
2.1.5	Satellite Based Augmentation Systems (SBAS) . . . . .	17
2.2	Other Standards . . . . .	17
2.2.1	RINEX . . . . .	17
2.2.2	NMEA . . . . .	18
2.2.3	KML . . . . .	18
2.2.4	C++ Standards . . . . .	18
2.2.5	Positioning protocols in wireless communication networks . . . . .	18
<b>3</b>	<b>Signal model</b>	<b>21</b>
3.1	GNSS signal model . . . . .	21
3.1.1	Global Positioning System (GPS) signal in space . . . . .	22
3.1.2	GLONASS signal in space . . . . .	23
3.1.3	Galileo signal in space . . . . .	24
3.1.4	Reference . . . . .	27
<b>4</b>	<b>Todo List</b>	<b>29</b>
<b>5</b>	<b>Module Index</b>	<b>31</b>
5.1	Modules . . . . .	31
<b>6</b>	<b>Hierarchical Index</b>	<b>33</b>
6.1	Class Hierarchy . . . . .	33
<b>7</b>	<b>Class Index</b>	<b>41</b>
7.1	Class List . . . . .	41
<b>8</b>	<b>File Index</b>	<b>55</b>
8.1	File List . . . . .	55

<b>9</b>	<b>Module Documentation</b>	<b>71</b>
9.1	Acquisition . . . . .	71
9.1.1	Detailed Description . . . . .	71
9.2	acquisition_adapters . . . . .	72
9.2.1	Detailed Description . . . . .	73
9.3	acquisition_gr_blocks . . . . .	74
9.3.1	Detailed Description . . . . .	75
9.4	acquisition_libs . . . . .	76
9.4.1	Detailed Description . . . . .	76
9.5	Channel . . . . .	77
9.5.1	Detailed Description . . . . .	77
9.6	channel_adapters . . . . .	78
9.6.1	Detailed Description . . . . .	78
9.7	channel_libs . . . . .	79
9.7.1	Detailed Description . . . . .	79
9.8	Signal Conditioner . . . . .	80
9.8.1	Detailed Description . . . . .	80
9.9	conditioner_adapters . . . . .	81
9.9.1	Detailed Description . . . . .	81
9.10	Data Type Adapters . . . . .	82
9.10.1	Detailed Description . . . . .	82
9.11	data_type_adapters . . . . .	83
9.11.1	Detailed Description . . . . .	83
9.12	data_type_gr_blocks . . . . .	84
9.12.1	Detailed Description . . . . .	84
9.13	Input Filter . . . . .	85
9.13.1	Detailed Description . . . . .	85
9.14	input_filter_adapters . . . . .	86
9.14.1	Detailed Description . . . . .	86
9.15	input_filter_gr_blocks . . . . .	87

9.15.1 Detailed Description	87
9.16 Algorithms Common Library	88
9.16.1 Detailed Description	88
9.17 algorithms_libs	89
9.17.1 Detailed Description	93
9.17.2 Function Documentation	93
9.17.2.1 at()	94
9.17.2.2 back()	94
9.17.2.3 beidou_b1i_code_gen_complex()	94
9.17.2.4 beidou_b1i_code_gen_complex_sampled()	94
9.17.2.5 beidou_b1i_code_gen_float()	95
9.17.2.6 beidou_b1i_code_gen_int()	95
9.17.2.7 beidou_b3i_code_gen_complex()	95
9.17.2.8 beidou_b3i_code_gen_complex_sampled()	95
9.17.2.9 beidou_b3i_code_gen_float()	95
9.17.2.10 beidou_b3i_code_gen_int()	96
9.17.2.11 cart2geo()	96
9.17.2.12 cart2utm()	96
9.17.2.13 clear()	96
9.17.2.14 clksin()	97
9.17.2.15 clsin()	97
9.17.2.16 complex_exp_gen()	97
9.17.2.17 complex_exp_gen_conj()	97
9.17.2.18 findUtmZone()	98
9.17.2.19 front()	98
9.17.2.20 galileo_e1_code_gen_complex_sampled() [1/2]	98
9.17.2.21 galileo_e1_code_gen_complex_sampled() [2/2]	98
9.17.2.22 galileo_e1_code_gen_float_sampled() [1/2]	99
9.17.2.23 galileo_e1_code_gen_float_sampled() [2/2]	99
9.17.2.24 galileo_e1_code_gen_sinboc11_float()	99

9.17.2.25 galileo_e5_a_code_gen_complex_primary()	99
9.17.2.26 galileo_e5_a_code_gen_complex_sampled()	100
9.17.2.27 galileo_e5_b_code_gen_complex_primary()	100
9.17.2.28 galileo_e5_b_code_gen_complex_sampled()	100
9.17.2.29 galileo_e6_b_code_gen_complex_primary()	100
9.17.2.30 galileo_e6_b_code_gen_complex_sampled()	101
9.17.2.31 galileo_e6_b_code_gen_float_primary()	101
9.17.2.32 galileo_e6_c_code_gen_complex_primary()	101
9.17.2.33 galileo_e6_c_code_gen_complex_sampled()	101
9.17.2.34 galileo_e6_c_code_gen_float_primary()	101
9.17.2.35 galileo_e6_c_secondary_code()	102
9.17.2.36 galileo_e6_c_secondary_code_gen_complex()	102
9.17.2.37 galileo_e6_c_secondary_code_gen_float()	102
9.17.2.38 Geo_to_ECEF()	102
9.17.2.39 get()	103
9.17.2.40 glonass_l1_ca_code_gen_complex()	103
9.17.2.41 glonass_l1_ca_code_gen_complex_sampled()	103
9.17.2.42 glonass_l2_ca_code_gen_complex()	103
9.17.2.43 glonass_l2_ca_code_gen_complex_sampled()	103
9.17.2.44 Gnss_circular_deque() [1/2]	104
9.17.2.45 Gnss_circular_deque() [2/2]	104
9.17.2.46 gps_l1_ca_code_gen_complex()	104
9.17.2.47 gps_l1_ca_code_gen_complex_sampled()	104
9.17.2.48 gps_l1_ca_code_gen_float()	105
9.17.2.49 gps_l1_ca_code_gen_int()	105
9.17.2.50 gps_l2c_m_code_gen_complex()	105
9.17.2.51 gps_l2c_m_code_gen_complex_sampled()	105
9.17.2.52 gps_l2c_m_code_gen_float()	105
9.17.2.53 gps_l5i_code_gen_complex()	106
9.17.2.54 gps_l5i_code_gen_complex_sampled()	106

9.17.2.55 <code>gps_l5i_code_gen_float()</code> . . . . .	106
9.17.2.56 <code>gps_l5q_code_gen_complex()</code> . . . . .	106
9.17.2.57 <code>gps_l5q_code_gen_complex_sampled()</code> . . . . .	106
9.17.2.58 <code>gps_l5q_code_gen_float()</code> . . . . .	107
9.17.2.59 <code>Gravity_ECEF()</code> . . . . .	107
9.17.2.60 <code>great_circle_distance()</code> . . . . .	107
9.17.2.61 <code>hex_to_binary_converter()</code> . . . . .	107
9.17.2.62 <code>hex_to_binary_string()</code> . . . . .	107
9.17.2.63 <code>item_type_is_complex()</code> . . . . .	108
9.17.2.64 <code>item_type_size()</code> . . . . .	108
9.17.2.65 <code>item_type_valid()</code> . . . . .	108
9.17.2.66 <code>make_vector_converter()</code> . . . . .	108
9.17.2.67 <code>pop_front()</code> . . . . .	109
9.17.2.68 <code>push_back()</code> . . . . .	109
9.17.2.69 <code>pv_Geo_to_ECEF()</code> . . . . .	109
9.17.2.70 <code>resampler()</code> [1/2] . . . . .	110
9.17.2.71 <code>resampler()</code> [2/2] . . . . .	110
9.17.2.72 <code>reset()</code> [1/2] . . . . .	110
9.17.2.73 <code>reset()</code> [2/2] . . . . .	110
9.17.2.74 <code>size()</code> . . . . .	111
9.17.2.75 <code>Skew_symmetric()</code> . . . . .	111
9.17.2.76 <code>togeod()</code> . . . . .	111
9.17.2.77 <code>topocent()</code> . . . . .	112
9.18 <code>gnss_sdr_flags</code> . . . . .	113
9.18.1 Detailed Description . . . . .	113
9.18.2 Function Documentation . . . . .	114
9.18.2.1 <code>DECLARE_bool()</code> . . . . .	114
9.18.2.2 <code>DECLARE_double()</code> [1/3] . . . . .	114
9.18.2.3 <code>DECLARE_double()</code> [2/3] . . . . .	114
9.18.2.4 <code>DECLARE_double()</code> [3/3] . . . . .	114

9.18.2.5	<a href="#">DECLARE_int32()</a> [1/7]	114
9.18.2.6	<a href="#">DECLARE_int32()</a> [2/7]	115
9.18.2.7	<a href="#">DECLARE_int32()</a> [3/7]	115
9.18.2.8	<a href="#">DECLARE_int32()</a> [4/7]	115
9.18.2.9	<a href="#">DECLARE_int32()</a> [5/7]	115
9.18.2.10	<a href="#">DECLARE_int32()</a> [6/7]	115
9.18.2.11	<a href="#">DECLARE_int32()</a> [7/7]	115
9.18.2.12	<a href="#">DECLARE_string()</a> [1/7]	116
9.18.2.13	<a href="#">DECLARE_string()</a> [2/7]	116
9.18.2.14	<a href="#">DECLARE_string()</a> [3/7]	116
9.18.2.15	<a href="#">DECLARE_string()</a> [4/7]	116
9.18.2.16	<a href="#">DECLARE_string()</a> [5/7]	116
9.18.2.17	<a href="#">DECLARE_string()</a> [6/7]	116
9.18.2.18	<a href="#">DECLARE_string()</a> [7/7]	116
9.19	<a href="#">PVT</a>	117
9.19.1	<a href="#">Detailed Description</a>	117
9.20	<a href="#">algorithms_libs_rtklib</a>	118
9.20.1	<a href="#">Detailed Description</a>	129
9.20.2	<a href="#">Typedef Documentation</a>	129
9.20.2.1	<a href="#">fatalfunc_t</a>	129
9.20.3	<a href="#">Function Documentation</a>	129
9.20.3.1	<a href="#">eph_to_rtklib()</a>	129
9.20.4	<a href="#">Variable Documentation</a>	129
9.20.4.1	<a href="#">ARMODE_CONT</a>	129
9.20.4.2	<a href="#">ARMODE_FIXHOLD</a>	130
9.20.4.3	<a href="#">ARMODE_INST</a>	130
9.20.4.4	<a href="#">ARMODE_OFF</a>	130
9.20.4.5	<a href="#">ARMODE_PPPAR</a>	130
9.20.4.6	<a href="#">ARMODE_PPPAR_ILS</a>	130
9.20.4.7	<a href="#">CHISQR</a>	131

9.20.4.8 DTTOL . . . . .	131
9.20.4.9 EFACT_BDS . . . . .	131
9.20.4.10 EFACT_GAL . . . . .	131
9.20.4.11 EFACT_GLO . . . . .	132
9.20.4.12 EFACT_GPS . . . . .	132
9.20.4.13 EFACT_IRN . . . . .	132
9.20.4.14 EFACT_QZS . . . . .	132
9.20.4.15 EFACT_SBS . . . . .	132
9.20.4.16 EPHOPT_BRDC . . . . .	133
9.20.4.17 EPHOPT_LEX . . . . .	133
9.20.4.18 EPHOPT_PREC . . . . .	133
9.20.4.19 EPHOPT_SBAS . . . . .	133
9.20.4.20 EPHOPT_SSRAPC . . . . .	133
9.20.4.21 EPHOPT_SSRCOM . . . . .	134
9.20.4.22 ERR_BRDCI . . . . .	134
9.20.4.23 ERR_CBIAS . . . . .	134
9.20.4.24 ERR_SAAS . . . . .	134
9.20.4.25 FE_WGS84 . . . . .	134
9.20.4.26 FTP_TIMEOUT . . . . .	135
9.20.4.27 GAP_RESION . . . . .	135
9.20.4.28 HION . . . . .	135
9.20.4.29 INT_SWAP_STAT . . . . .	135
9.20.4.30 INT_SWAP_TRAC . . . . .	135
9.20.4.31 IONOOPT_BRDC . . . . .	136
9.20.4.32 IONOOPT_EST . . . . .	136
9.20.4.33 IONOOPT_IFLC . . . . .	136
9.20.4.34 IONOOPT_LEX . . . . .	136
9.20.4.35 IONOOPT_OFF . . . . .	136
9.20.4.36 IONOOPT_QZS . . . . .	137
9.20.4.37 IONOOPT_SBAS . . . . .	137

9.20.4.38 IONOOPT_STEC . . . . .	137
9.20.4.39 IONOOPT_TEC . . . . .	137
9.20.4.40 LAM_CARR . . . . .	137
9.20.4.41 MAXANT . . . . .	138
9.20.4.42 MAXBAND . . . . .	138
9.20.4.43 MAXCLI . . . . .	138
9.20.4.44 MAXDTE . . . . .	138
9.20.4.45 MAXDTE_BDS . . . . .	138
9.20.4.46 MAXDTE_GAL . . . . .	139
9.20.4.47 MAXDTE_GLO . . . . .	139
9.20.4.48 MAXDTE_QZS . . . . .	139
9.20.4.49 MAXDTE_S . . . . .	139
9.20.4.50 MAXDTE_SBS . . . . .	139
9.20.4.51 MAXERRMSG . . . . .	140
9.20.4.52 MAXEXFILE . . . . .	140
9.20.4.53 MAXFREQ . . . . .	140
9.20.4.54 MAXGDOP . . . . .	140
9.20.4.55 MAXLEAPS . . . . .	140
9.20.4.56 MAXNGEO . . . . .	141
9.20.4.57 MAXNIGP . . . . .	141
9.20.4.58 MAXOBS . . . . .	141
9.20.4.59 MAXOBSBUF . . . . .	141
9.20.4.60 MAXOBSTYPE . . . . .	141
9.20.4.61 MAXPRNBDS . . . . .	142
9.20.4.62 MAXPRNGAL . . . . .	142
9.20.4.63 MAXPRNGLO . . . . .	142
9.20.4.64 MAXPRNGPS . . . . .	142
9.20.4.65 MAXPRNSBS . . . . .	142
9.20.4.66 MAXRAWLEN . . . . .	143
9.20.4.67 MAXRCV . . . . .	143

9.20.4.68 MAXSBSAGEF . . . . .	143
9.20.4.69 MAXSBSAGEL . . . . .	143
9.20.4.70 MAXSBSMSG . . . . .	143
9.20.4.71 MAXSBSURA . . . . .	144
9.20.4.72 MAXSOLBUF . . . . .	144
9.20.4.73 MAXSOLMSG . . . . .	144
9.20.4.74 MAXSOLQ . . . . .	144
9.20.4.75 MAXSTATMSG . . . . .	144
9.20.4.76 MAXSTRMSG . . . . .	145
9.20.4.77 MAXSTRPATH . . . . .	145
9.20.4.78 MINPRNBDS . . . . .	145
9.20.4.79 MINPRNGAL . . . . .	145
9.20.4.80 MINPRNGLO . . . . .	145
9.20.4.81 MINPRNGPS . . . . .	146
9.20.4.82 MINPRNSBS . . . . .	146
9.20.4.83 NEXOBS . . . . .	146
9.20.4.84 NFREQ . . . . .	146
9.20.4.85 NFREQGLO . . . . .	146
9.20.4.86 NSATBDS . . . . .	147
9.20.4.87 NSATGAL . . . . .	147
9.20.4.88 NSATGLO . . . . .	147
9.20.4.89 NSATGPS . . . . .	147
9.20.4.90 NSATSBS . . . . .	147
9.20.4.91 NSYS . . . . .	148
9.20.4.92 PMODE_DGPS . . . . .	148
9.20.4.93 PMODE_FIXED . . . . .	148
9.20.4.94 PMODE_KINEMA . . . . .	148
9.20.4.95 PMODE_MOVEB . . . . .	148
9.20.4.96 PMODE_PPP_FIXED . . . . .	149
9.20.4.97 PMODE_PPP_KINEMA . . . . .	149

9.20.4.98 PMODE_PPP_STATIC . . . . .	149
9.20.4.99 PMODE_SINGLE . . . . .	149
9.20.4.100 PMODE_STATIC . . . . .	149
9.20.4.101 POLYCRC24Q . . . . .	150
9.20.4.102 POLYCRC32 . . . . .	150
9.20.4.103 POSOPT_RINEX . . . . .	150
9.20.4.104 PRN_HWBIAS . . . . .	150
9.20.4.105 RE_WGS84 . . . . .	150
9.20.4.106 REL_HUMI . . . . .	151
9.20.4.107 SERIBUFFSIZE . . . . .	151
9.20.4.108 SOLF_ENU . . . . .	151
9.20.4.109 SOLF_GSIF . . . . .	151
9.20.4.110 SOLF_LLH . . . . .	151
9.20.4.111 SOLF_NMEA . . . . .	152
9.20.4.112 SOLF_STAT . . . . .	152
9.20.4.113 SOLF_XYZ . . . . .	152
9.20.4.114 SOLQ_DGPS . . . . .	152
9.20.4.115 SOLQ_DR . . . . .	152
9.20.4.116 SOLQ_FIX . . . . .	153
9.20.4.117 SOLQ_FLOAT . . . . .	153
9.20.4.118 SOLQ_NONE . . . . .	153
9.20.4.119 SOLQ_PPP . . . . .	153
9.20.4.120 SOLQ_SBAS . . . . .	153
9.20.4.121 SOLQ_SINGLE . . . . .	154
9.20.4.122 SYS_ALL . . . . .	154
9.20.4.123 SYS_BDS . . . . .	154
9.20.4.124 SYS_GAL . . . . .	154
9.20.4.125 SYS_GLO . . . . .	154
9.20.4.126 SYS_GPS . . . . .	155
9.20.4.127 SYS_IRN . . . . .	155

9.20.4.128	SYS_LEO	155
9.20.4.129	SYS_NONE	155
9.20.4.130	SYS_QZS	155
9.20.4.131	SYS_SBS	156
9.20.4.132	TIMES_GPST	156
9.20.4.133	TIMES_JST	156
9.20.4.134	TIMES_UTC	156
9.20.4.135	TIMETAGH_LEN	156
9.20.4.136	TINTACT	157
9.20.4.137	TROPOPT_COR	157
9.20.4.138	TROPOPT_CORG	157
9.20.4.139	TROPOPT_EST	157
9.20.4.140	TROPOPT_ESTG	157
9.20.4.141	TROPOPT_OFF	158
9.20.4.142	TROPOPT_SAAS	158
9.20.4.143	TROPOPT_SBAS	158
9.21	Observables	159
9.21.1	Detailed Description	159
9.22	obs_adapters	160
9.22.1	Detailed Description	160
9.23	obs_gr_blocks	161
9.23.1	Detailed Description	161
9.24	observables_libs	162
9.24.1	Detailed Description	162
9.25	pvt_adapters	163
9.25.1	Detailed Description	163
9.26	pvt_gr_blocks	164
9.26.1	Detailed Description	164
9.27	pvt_libs	165
9.27.1	Detailed Description	165

9.28 Resampler . . . . .	166
9.28.1 Detailed Description . . . . .	166
9.29 resampler_adapters . . . . .	167
9.29.1 Detailed Description . . . . .	167
9.30 resampler_gr_blocks . . . . .	168
9.30.1 Detailed Description . . . . .	168
9.31 Signal Source . . . . .	169
9.31.1 Detailed Description . . . . .	169
9.32 signal_source_adapters . . . . .	170
9.32.1 Detailed Description . . . . .	170
9.33 signal_source_gr_blocks . . . . .	171
9.33.1 Detailed Description . . . . .	171
9.34 signal_source_libs . . . . .	172
9.34.1 Detailed Description . . . . .	173
9.34.2 Enumeration Type Documentation . . . . .	173
9.34.2.1 RTL_TCP_COMMAND . . . . .	173
9.34.3 Function Documentation . . . . .	173
9.34.3.1 rtl_tcp_command() . . . . .	173
9.35 Telemetry Decoder . . . . .	174
9.35.1 Detailed Description . . . . .	174
9.36 telemetry_decoder_adapters . . . . .	175
9.36.1 Detailed Description . . . . .	175
9.37 telemetry_decoder_gr_blocks . . . . .	176
9.37.1 Detailed Description . . . . .	177
9.38 telemetry_decoder_libs . . . . .	178
9.38.1 Detailed Description . . . . .	178
9.38.2 Function Documentation . . . . .	178
9.38.2.1 Gamma() . . . . .	178
9.38.2.2 nsc_enc_bit() . . . . .	179
9.38.2.3 nsc_transit() . . . . .	179

9.38.2.4	parity_counter()	180
9.38.2.5	Viterbi()	180
9.39	telemetry_decoder_libswiftcnv	182
9.39.1	Detailed Description	182
9.39.2	Macro Definition Documentation	182
9.39.2.1	GPS_L2_V27_HISTORY_LENGTH_BITS	183
9.39.2.2	GPS_L2C_V27_DECODE_BITS	183
9.39.2.3	GPS_L2C_V27_DELAY_BITS	183
9.39.2.4	GPS_L2C_V27_INIT_BITS	183
9.40	Tracking	184
9.40.1	Detailed Description	184
9.41	tracking_adapters	185
9.41.1	Detailed Description	185
9.42	tracking_gr_blocks	186
9.42.1	Detailed Description	187
9.43	tracking_libs	188
9.43.1	Detailed Description	189
9.43.2	Function Documentation	189
9.43.2.1	carrier_lock_detector()	190
9.43.2.2	cn0_m2m4_estimator()	190
9.43.2.3	cn0_svn_estimator()	191
9.43.2.4	dll_nc_e_minus_l_normalized()	191
9.43.2.5	dll_nc_vemlp_normalized()	191
9.43.2.6	fill_four_quadrant_atan()	192
9.43.2.7	phase_unwrap()	192
9.43.2.8	pll_cloop_two_quadrant_atan()	192
9.43.2.9	pll_four_quadrant_atan()	192
9.44	Core GNSS Receiver	193
9.44.1	Detailed Description	193
9.45	GNSS block interfaces	194

9.45.1 Detailed Description . . . . .	194
9.46 core_libs . . . . .	195
9.46.1 Detailed Description . . . . .	196
9.46.2 Function Documentation . . . . .	196
9.46.2.1 find_uio_dev_file_name() . . . . .	196
9.46.2.2 ini_parse() . . . . .	196
9.47 core_monitor . . . . .	197
9.47.1 Detailed Description . . . . .	197
9.48 core_receiver . . . . .	198
9.48.1 Detailed Description . . . . .	198
9.49 core_system_parameters . . . . .	199
9.49.1 Detailed Description . . . . .	230
9.49.2 Macro Definition Documentation . . . . .	231
9.49.2.1 GLONASS_GNAV_PREAMBLE . . . . .	231
9.49.3 Function Documentation . . . . .	231
9.49.3.1 ALPHA_0() . . . . .	231
9.49.3.2 T_OA() . . . . .	231
9.49.4 Variable Documentation . . . . .	231
9.49.4.1 AS2R . . . . .	232
9.49.4.2 AU . . . . .	232
9.49.4.3 BEIDOU_B1I_CODE_LENGTH_CHIPS . . . . .	232
9.49.4.4 BEIDOU_B1I_CODE_PERIOD_MS . . . . .	232
9.49.4.5 BEIDOU_B1I_CODE_PERIOD_S . . . . .	232
9.49.4.6 BEIDOU_B1I_CODE_RATE_CPS . . . . .	233
9.49.4.7 BEIDOU_B1I_FREQ_HZ . . . . .	233
9.49.4.8 BEIDOU_B3I_CODE_LENGTH_CHIPS . . . . .	233
9.49.4.9 BEIDOU_B3I_CODE_PERIOD_MS . . . . .	233
9.49.4.10 BEIDOU_B3I_CODE_PERIOD_S . . . . .	233
9.49.4.11 BEIDOU_B3I_CODE_RATE_CPS . . . . .	234
9.49.4.12 BEIDOU_B3I_FREQ_HZ . . . . .	234

9.49.4.13 BEIDOU_B3I_TELEMETRY_RATE_BITS_SECOND . . . . .	234
9.49.4.14 BEIDOU_DNAV_SUBFRAME_DATA_BITS . . . . .	234
9.49.4.15 BEIDOU_F . . . . .	234
9.49.4.16 BEIDOU_GM . . . . .	235
9.49.4.17 BEIDOU_OMEGA_EARTH_DOT . . . . .	235
9.49.4.18 CODE_L1A . . . . .	235
9.49.4.19 CODE_L1B . . . . .	235
9.49.4.20 CODE_L1C . . . . .	235
9.49.4.21 CODE_L1E . . . . .	236
9.49.4.22 CODE_L1I . . . . .	236
9.49.4.23 CODE_L1L . . . . .	236
9.49.4.24 CODE_L1M . . . . .	236
9.49.4.25 CODE_L1N . . . . .	236
9.49.4.26 CODE_L1P . . . . .	237
9.49.4.27 CODE_L1Q . . . . .	237
9.49.4.28 CODE_L1S . . . . .	237
9.49.4.29 CODE_L1W . . . . .	237
9.49.4.30 CODE_L1X . . . . .	237
9.49.4.31 CODE_L1Y . . . . .	238
9.49.4.32 CODE_L1Z . . . . .	238
9.49.4.33 CODE_L2C . . . . .	238
9.49.4.34 CODE_L2D . . . . .	238
9.49.4.35 CODE_L2I . . . . .	238
9.49.4.36 CODE_L2L . . . . .	239
9.49.4.37 CODE_L2M . . . . .	239
9.49.4.38 CODE_L2N . . . . .	239
9.49.4.39 CODE_L2P . . . . .	239
9.49.4.40 CODE_L2Q . . . . .	239
9.49.4.41 CODE_L2S . . . . .	240
9.49.4.42 CODE_L2W . . . . .	240

9.49.4.43 CODE_L2X . . . . .	240
9.49.4.44 CODE_L2Y . . . . .	240
9.49.4.45 CODE_L3I . . . . .	240
9.49.4.46 CODE_L3Q . . . . .	241
9.49.4.47 CODE_L3X . . . . .	241
9.49.4.48 CODE_L5A . . . . .	241
9.49.4.49 CODE_L5B . . . . .	241
9.49.4.50 CODE_L5C . . . . .	241
9.49.4.51 CODE_L5I . . . . .	242
9.49.4.52 CODE_L5Q . . . . .	242
9.49.4.53 CODE_L5X . . . . .	242
9.49.4.54 CODE_L6A . . . . .	242
9.49.4.55 CODE_L6B . . . . .	242
9.49.4.56 CODE_L6C . . . . .	243
9.49.4.57 CODE_L6I . . . . .	243
9.49.4.58 CODE_L6L . . . . .	243
9.49.4.59 CODE_L6Q . . . . .	243
9.49.4.60 CODE_L6S . . . . .	243
9.49.4.61 CODE_L6X . . . . .	244
9.49.4.62 CODE_L6Z . . . . .	244
9.49.4.63 CODE_L7I . . . . .	244
9.49.4.64 CODE_L7Q . . . . .	244
9.49.4.65 CODE_L7X . . . . .	244
9.49.4.66 CODE_L8I . . . . .	245
9.49.4.67 CODE_L8Q . . . . .	245
9.49.4.68 CODE_L8X . . . . .	245
9.49.4.69 CODE_L9A . . . . .	245
9.49.4.70 CODE_L9B . . . . .	245
9.49.4.71 CODE_L9C . . . . .	246
9.49.4.72 CODE_L9X . . . . .	246

9.49.4.73 CODE_NONE . . . . .	246
9.49.4.74 D2R . . . . .	246
9.49.4.75 DFRQ1_GLO . . . . .	246
9.49.4.76 DFRQ2_GLO . . . . .	247
9.49.4.77 FREQ1 . . . . .	247
9.49.4.78 FREQ1_BDS . . . . .	247
9.49.4.79 FREQ1_GLO . . . . .	247
9.49.4.80 FREQ2 . . . . .	247
9.49.4.81 FREQ2_BDS . . . . .	248
9.49.4.82 FREQ2_GLO . . . . .	248
9.49.4.83 FREQ3_BDS . . . . .	248
9.49.4.84 FREQ3_GLO . . . . .	248
9.49.4.85 FREQ5 . . . . .	248
9.49.4.86 FREQ6 . . . . .	249
9.49.4.87 FREQ7 . . . . .	249
9.49.4.88 FREQ8 . . . . .	249
9.49.4.89 FREQ9 . . . . .	249
9.49.4.90 GALILEO_CNAV_PAGE_MS . . . . .	249
9.49.4.91 GALILEO_CNAV_SYMBOLS_PER_PAGE . . . . .	250
9.49.4.92 GALILEO_E1_B_CODE_LENGTH_CHIPS . . . . .	250
9.49.4.93 GALILEO_E1_B_SAMPLES_PER_SYMBOL . . . . .	250
9.49.4.94 GALILEO_E1_B_SYMBOL_RATE_BPS . . . . .	250
9.49.4.95 GALILEO_E1_C_SECONDARY_CODE_LENGTH . . . . .	250
9.49.4.96 GALILEO_E1_CODE_CHIP_RATE_CPS . . . . .	251
9.49.4.97 GALILEO_E1_CODE_PERIOD_MS . . . . .	251
9.49.4.98 GALILEO_E1_CODE_PERIOD_S . . . . .	251
9.49.4.99 GALILEO_E1_FREQ_HZ . . . . .	251
9.49.4.100 GALILEO_E1_HISTORY_DEEP . . . . .	251
9.49.4.101 GALILEO_E1_OPT_ACQ_FS_SPS . . . . .	252
9.49.4.102 GALILEO_E1_SUB_CARRIER_A_RATE_HZ . . . . .	252

9.49.4.103	GALILEO_E1_SUB_CARRIER_B_RATE_HZ . . . . .	252
9.49.4.104	GALILEO_E5A_CODE_CHIP_RATE_CPS . . . . .	252
9.49.4.105	GALILEO_E5A_CODE_LENGTH_CHIPS . . . . .	252
9.49.4.106	GALILEO_E5A_CODE_PERIOD_MS . . . . .	253
9.49.4.107	GALILEO_E5A_CODE_PERIOD_S . . . . .	253
9.49.4.108	GALILEO_E5A_FREQ_HZ . . . . .	253
9.49.4.109	GALILEO_E5A_I_SECONDARY_CODE_LENGTH . . . . .	253
9.49.4.110	GALILEO_E5A_I_TIERED_CODE_PERIOD_S . . . . .	253
9.49.4.111	GALILEO_E5A_OPT_ACQ_FS_SPS . . . . .	254
9.49.4.112	GALILEO_E5A_Q_SECONDARY_CODE_LENGTH . . . . .	254
9.49.4.113	GALILEO_E5A_Q_TIERED_CODE_PERIOD_S . . . . .	254
9.49.4.114	GALILEO_E5A_SYMBOL_RATE_BPS . . . . .	254
9.49.4.115	GALILEO_E5B_CODE_CHIP_RATE_CPS . . . . .	254
9.49.4.116	GALILEO_E5B_CODE_LENGTH_CHIPS . . . . .	255
9.49.4.117	GALILEO_E5B_CODE_PERIOD_MS . . . . .	255
9.49.4.118	GALILEO_E5B_CODE_PERIOD_S . . . . .	255
9.49.4.119	GALILEO_E5B_FREQ_HZ . . . . .	255
9.49.4.120	GALILEO_E5B_I_SECONDARY_CODE_LENGTH . . . . .	255
9.49.4.121	GALILEO_E5B_I_TIERED_CODE_PERIOD_S . . . . .	256
9.49.4.122	GALILEO_E5B_OPT_ACQ_FS_SPS . . . . .	256
9.49.4.123	GALILEO_E5B_Q_SECONDARY_CODE_LENGTH . . . . .	256
9.49.4.124	GALILEO_E5B_Q_TIERED_CODE_PERIOD_S . . . . .	256
9.49.4.125	GALILEO_E5B_SYMBOL_RATE_BPS . . . . .	256
9.49.4.126	GALILEO_E6_B_CODE_CHIP_RATE_CPS . . . . .	257
9.49.4.127	GALILEO_E6_B_CODE_LENGTH_CHIPS . . . . .	257
9.49.4.128	GALILEO_E6_C_CODE_CHIP_RATE_CPS . . . . .	257
9.49.4.129	GALILEO_E6_C_CODE_LENGTH_CHIPS . . . . .	257
9.49.4.130	GALILEO_E6_C_SECONDARY_CODE_LENGTH_CHIPS . . . . .	257
9.49.4.131	GALILEO_E6_CODE_PERIOD_MS . . . . .	258
9.49.4.132	GALILEO_E6_CODE_PERIOD_S . . . . .	258

9.49.4.133	GALILEO_E6_FREQ_HZ	258
9.49.4.134	GALILEO_F	258
9.49.4.135	GALILEO_GM	258
9.49.4.136	GALILEO_INAV_PAGE_PART_SYMBOLS	259
9.49.4.137	GALILEO_INAV_PAGE_PART_WITH_PREABLE_SECONDS	259
9.49.4.138	GALILEO_INAV_PAGE_SYMBOLS	259
9.49.4.139	GLONASS_C20	259
9.49.4.140	GLONASS_EARTH_INCLINATION	259
9.49.4.141	GLONASS_EARTH_RADIUS	260
9.49.4.142	GLONASS_F_M_A	260
9.49.4.143	GLONASS_FLATTENING	260
9.49.4.144	GLONASS_GM	260
9.49.4.145	GLONASS_GNAV_HAMMING_CODE_BITS	260
9.49.4.146	GLONASS_GNAV_STRING_BITS	261
9.49.4.147	GLONASS_GNAV_STRING_SYMBOLS	261
9.49.4.148	GLONASS_GNAV_TELEMETRY_RATE_BITS_SECOND	261
9.49.4.149	GLONASS_GNAV_TELEMETRY_RATE_SYMBOLS_SECOND	261
9.49.4.150	GLONASS_GRAVITY	261
9.49.4.151	GLONASS_GRAVITY_CORRECTION	262
9.49.4.152	GLONASS_J2	262
9.49.4.153	GLONASS_J4	262
9.49.4.154	GLONASS_J6	262
9.49.4.155	GLONASS_J8	262
9.49.4.156	GLONASS_L1_CA_CHIP_PERIOD_S	263
9.49.4.157	GLONASS_L1_CA_CODE_LENGTH_CHIPS	263
9.49.4.158	GLONASS_L1_CA_CODE_PERIOD_S	263
9.49.4.159	GLONASS_L1_CA_CODE_RATE_CPS	263
9.49.4.160	GLONASS_L1_CA_DFREQ_HZ	263
9.49.4.161	GLONASS_L1_CA_FREQ_HZ	264
9.49.4.162	GLONASS_L2_CA_CHIP_PERIOD_S	264

9.49.4.163	GLONASS_L2_CA_CODE_LENGTH_CHIPS . . . . .	264
9.49.4.164	GLONASS_L2_CA_CODE_PERIOD_S . . . . .	264
9.49.4.165	GLONASS_L2_CA_CODE_RATE_CPS . . . . .	264
9.49.4.166	GLONASS_L2_CA_DFREQ_HZ . . . . .	265
9.49.4.167	GLONASS_L2_CA_FREQ_HZ . . . . .	265
9.49.4.168	GLONASS_LEAP_SECONDS . . . . .	265
9.49.4.169	GLONASS_MOON_ECCENTRICITY . . . . .	266
9.49.4.170	GLONASS_MOON_GM . . . . .	266
9.49.4.171	GLONASS_MOON_INCLINATION . . . . .	266
9.49.4.172	GLONASS_MOON_OMEGA_0 . . . . .	266
9.49.4.173	GLONASS_MOON_OMEGA_1 . . . . .	266
9.49.4.174	GLONASS_MOON_Q0 . . . . .	267
9.49.4.175	GLONASS_MOON_Q1 . . . . .	267
9.49.4.176	GLONASS_MOON_SEMI_MAJOR_AXIS . . . . .	267
9.49.4.177	GLONASS_OMEGA_EARTH_DOT . . . . .	267
9.49.4.178	GLONASS_SEMI_MAJOR_AXIS . . . . .	267
9.49.4.179	GLONASS_SUN_ECCENTRICITY . . . . .	268
9.49.4.180	GLONASS_SUN_GM . . . . .	268
9.49.4.181	GLONASS_SUN_OMEGA . . . . .	268
9.49.4.182	GLONASS_SUN_Q0 . . . . .	268
9.49.4.183	GLONASS_SUN_Q1 . . . . .	268
9.49.4.184	GLONASS_SUN_SEMI_MAJOR_AXIS . . . . .	269
9.49.4.185	GLONASS_TAU_0 . . . . .	269
9.49.4.186	GLONASS_TAU_1 . . . . .	269
9.49.4.187	GLONASS_U0 . . . . .	269
9.49.4.188	GNSS_OMEGA_EARTH_DOT . . . . .	269
9.49.4.189	GNSS_PI . . . . .	270
9.49.4.190	GPS_CA_TELEMETRY_RATE_BITS_SECOND . . . . .	270
9.49.4.191	GPS_CA_TELEMETRY_RATE_SYMBOLS_SECOND . . . . .	270
9.49.4.192	GPS_F . . . . .	270

9.49.4.193	GPS_GM	270
9.49.4.194	GPS_L1_CA_BIT_PERIOD_MS	271
9.49.4.195	GPS_L1_CA_CHIP_PERIOD_S	271
9.49.4.196	GPS_L1_CA_CODE_LENGTH_CHIPS	271
9.49.4.197	GPS_L1_CA_CODE_PERIOD_MS	271
9.49.4.198	GPS_L1_CA_CODE_PERIOD_S	271
9.49.4.199	GPS_L1_CA_CODE_RATE_CPS	272
9.49.4.200	GPS_L1_CA_OPT_ACQ_FS_SPS	272
9.49.4.201	GPS_L1_FREQ_HZ	272
9.49.4.202	GPS_L2_CNAV_DATA_PAGE_BITS	272
9.49.4.203	GPS_L2_FREQ_HZ	272
9.49.4.204	GPS_L2_L_CODE_LENGTH_CHIPS	273
9.49.4.205	GPS_L2_L_CODE_RATE_CPS	273
9.49.4.206	GPS_L2_L_PERIOD_S	273
9.49.4.207	GPS_L2_M_CODE_LENGTH_CHIPS	273
9.49.4.208	GPS_L2_M_CODE_RATE_CPS	273
9.49.4.209	GPS_L2_M_PERIOD_S	274
9.49.4.210	GPS_L2C_M_INIT_REG	274
9.49.4.211	GPS_L2C_OPT_ACQ_FS_SPS	274
9.49.4.212	GPS_L5_CNAV_DATA_PAGE_BITS	275
9.49.4.213	GPS_L5_FREQ_HZ	275
9.49.4.214	GPS_L5_OPT_ACQ_FS_SPS	275
9.49.4.215	GPS_L5I_CODE_LENGTH_CHIPS	275
9.49.4.216	GPS_L5I_CODE_RATE_CPS	275
9.49.4.217	GPS_L5I_PERIOD_MS	276
9.49.4.218	GPS_L5I_PERIOD_S	276
9.49.4.219	GPS_L5I_SYMBOL_PERIOD_MS	276
9.49.4.220	GPS_L5I_SYMBOL_PERIOD_S	276
9.49.4.221	GPS_L5Q_CODE_LENGTH_CHIPS	276
9.49.4.222	GPS_L5Q_CODE_RATE_CPS	277

9.49.4.223GPS_L5Q_PERIOD_S . . . . .	277
9.49.4.224GPS_SUBFRAME_BITS . . . . .	277
9.49.4.225GPS_SUBFRAME_LENGTH . . . . .	277
9.49.4.226GPS_SUBFRAME_MS . . . . .	277
9.49.4.227GPS_SUBFRAME_SECONDS . . . . .	278
9.49.4.228GPS_WORD_BITS . . . . .	278
9.49.4.229GPS_WORD_LENGTH . . . . .	278
9.49.4.230HALF_PI . . . . .	278
9.49.4.231MAX_TOA_DELAY_MS . . . . .	278
9.49.4.232MAXCODE . . . . .	279
9.49.4.233PI_TWO_N19 . . . . .	279
9.49.4.234PI_TWO_N23 . . . . .	279
9.49.4.235PI_TWO_N31 . . . . .	279
9.49.4.236PI_TWO_N38 . . . . .	279
9.49.4.237PI_TWO_N43 . . . . .	280
9.49.4.238R2D . . . . .	280
9.49.4.239SC2RAD . . . . .	280
9.49.4.240SPEED_OF_LIGHT_M_MS . . . . .	280
9.49.4.241SPEED_OF_LIGHT_M_S . . . . .	280
9.49.4.242TWO_N10 . . . . .	281
9.49.4.243TWO_N11 . . . . .	281
9.49.4.244TWO_N14 . . . . .	281
9.49.4.245TWO_N15 . . . . .	281
9.49.4.246TWO_N16 . . . . .	281
9.49.4.247TWO_N17 . . . . .	282
9.49.4.248TWO_N18 . . . . .	282
9.49.4.249TWO_N19 . . . . .	282
9.49.4.250TWO_N2 . . . . .	282
9.49.4.251TWO_N20 . . . . .	282
9.49.4.252TWO_N21 . . . . .	283

9.49.4.253TWO_N23 . . . . .	283
9.49.4.254TWO_N24 . . . . .	283
9.49.4.255TWO_N25 . . . . .	283
9.49.4.256TWO_N27 . . . . .	283
9.49.4.257TWO_N29 . . . . .	284
9.49.4.258TWO_N30 . . . . .	284
9.49.4.259TWO_N31 . . . . .	284
9.49.4.260TWO_N32 . . . . .	284
9.49.4.261TWO_N33 . . . . .	284
9.49.4.262TWO_N34 . . . . .	285
9.49.4.263TWO_N35 . . . . .	285
9.49.4.264TWO_N38 . . . . .	285
9.49.4.265TWO_N39 . . . . .	285
9.49.4.266TWO_N40 . . . . .	285
9.49.4.267TWO_N43 . . . . .	286
9.49.4.268TWO_N44 . . . . .	286
9.49.4.269TWO_N46 . . . . .	286
9.49.4.270TWO_N48 . . . . .	286
9.49.4.271TWO_N5 . . . . .	286
9.49.4.272TWO_N50 . . . . .	287
9.49.4.273TWO_N51 . . . . .	287
9.49.4.274TWO_N55 . . . . .	287
9.49.4.275TWO_N57 . . . . .	287
9.49.4.276TWO_N59 . . . . .	287
9.49.4.277TWO_N6 . . . . .	288
9.49.4.278TWO_N60 . . . . .	288
9.49.4.279TWO_N66 . . . . .	288
9.49.4.280TWO_N68 . . . . .	288
9.49.4.281TWO_N8 . . . . .	288
9.49.4.282TWO_N9 . . . . .	289
9.49.4.283TWO_P11 . . . . .	289
9.49.4.284TWO_P12 . . . . .	289
9.49.4.285TWO_P14 . . . . .	289
9.49.4.286TWO_P16 . . . . .	289
9.49.4.287TWO_P19 . . . . .	290
9.49.4.288TWO_P3 . . . . .	290
9.49.4.289TWO_P31 . . . . .	290
9.49.4.290TWO_P32 . . . . .	290
9.49.4.291TWO_P4 . . . . .	290
9.49.4.292TWO_P56 . . . . .	291
9.49.4.293TWO_P57 . . . . .	291
9.49.4.294TWO_PI . . . . .	291

<b>10 Class Documentation</b>	<b>293</b>
10.1 Acq_Conf Class Reference	293
10.1.1 Detailed Description	294
10.2 Acquisition_Dump_Reader Class Reference	294
10.2.1 Detailed Description	294
10.2.2 Constructor & Destructor Documentation	294
10.2.2.1 Acquisition_Dump_Reader() [1/2]	295
10.2.2.2 Acquisition_Dump_Reader() [2/2]	295
10.2.3 Member Function Documentation	295
10.2.3.1 operator=() [1/2]	295
10.2.3.2 operator=() [2/2]	295
10.3 Acquisition_msg_rx Class Reference	295
10.3.1 Detailed Description	296
10.3.2 Constructor & Destructor Documentation	296
10.3.2.1 ~Acquisition_msg_rx()	296
10.4 AcquisitionInterface Class Reference	297
10.4.1 Detailed Description	298
10.5 Ad9361FpgaSignalSource Class Reference	298
10.5.1 Detailed Description	298
10.5.2 Member Function Documentation	298
10.5.2.1 implementation()	299
10.5.2.2 start()	299
10.6 Agnss_Ref_Location Class Reference	299
10.6.1 Detailed Description	299
10.6.2 Constructor & Destructor Documentation	300
10.6.2.1 Agnss_Ref_Location()	300
10.6.3 Member Function Documentation	300
10.6.3.1 serialize()	300
10.7 Agnss_Ref_Time Class Reference	300
10.7.1 Detailed Description	301

10.7.2	Constructor & Destructor Documentation	301
10.7.2.1	Agnss_Ref_Time()	301
10.7.3	Member Function Documentation	301
10.7.3.1	serialize()	301
10.8	alm_t Struct Reference	302
10.8.1	Detailed Description	302
10.9	ambc_t Struct Reference	302
10.9.1	Detailed Description	302
10.10	ArraySignalConditioner Class Reference	303
10.10.1	Detailed Description	303
10.10.2	Constructor & Destructor Documentation	303
10.10.2.1	ArraySignalConditioner()	304
10.10.2.2	~ArraySignalConditioner()	304
10.10.3	Member Function Documentation	304
10.10.3.1	implementation()	304
10.11	Bayesian_estimator Class Reference	304
10.11.1	Detailed Description	305
10.12	beamformer Class Reference	305
10.12.1	Detailed Description	306
10.13	BeamformerFilter Class Reference	306
10.13.1	Detailed Description	306
10.13.2	Member Function Documentation	306
10.13.2.1	implementation()	307
10.14	beidou_b1i_telemetry_decoder_gs Class Reference	307
10.14.1	Detailed Description	308
10.14.2	Constructor & Destructor Documentation	308
10.14.2.1	~beidou_b1i_telemetry_decoder_gs()	308
10.14.3	Member Function Documentation	308
10.14.3.1	general_work()	308
10.14.3.2	set_channel()	308

10.14.3.3 <code>set_satellite()</code> . . . . .	309
10.15 <code>beidou_b3i_telemetry_decoder_gs</code> Class Reference . . . . .	309
10.15.1 Detailed Description . . . . .	309
10.15.2 Constructor & Destructor Documentation . . . . .	310
10.15.2.1 <code>~beidou_b3i_telemetry_decoder_gs()</code> . . . . .	310
10.15.3 Member Function Documentation . . . . .	310
10.15.3.1 <code>general_work()</code> . . . . .	310
10.15.3.2 <code>set_channel()</code> . . . . .	310
10.15.3.3 <code>set_satellite()</code> . . . . .	310
10.16 <code>Beidou_Dnav_Almanac</code> Class Reference . . . . .	311
10.16.1 Detailed Description . . . . .	311
10.16.2 Constructor & Destructor Documentation . . . . .	311
10.16.2.1 <code>Beidou_Dnav_Almanac()</code> . . . . .	312
10.16.3 Member Data Documentation . . . . .	312
10.16.3.1 <code>d_A_f0</code> . . . . .	312
10.16.3.2 <code>d_A_f1</code> . . . . .	312
10.16.3.3 <code>d_e_eccentricity</code> . . . . .	312
10.16.3.4 <code>d_M_0</code> . . . . .	312
10.16.3.5 <code>d_OMEGA</code> . . . . .	313
10.16.3.6 <code>d_OMEGA0</code> . . . . .	313
10.16.3.7 <code>d_OMEGA_DOT</code> . . . . .	313
10.16.3.8 <code>d_sqrt_A</code> . . . . .	313
10.16.3.9 <code>d_Toa</code> . . . . .	313
10.16.3.10 <code>id_satellite_PRN</code> . . . . .	314
10.16.3.11 <code>id_SV_health</code> . . . . .	314
10.17 <code>Beidou_Dnav_Ephemeris</code> Class Reference . . . . .	314
10.17.1 Detailed Description . . . . .	317
10.17.2 Constructor & Destructor Documentation . . . . .	317
10.17.2.1 <code>Beidou_Dnav_Ephemeris()</code> . . . . .	317
10.17.3 Member Function Documentation . . . . .	317

10.17.3.1 satellitePosition()	317
10.17.3.2 serialize()	318
10.17.3.3 sv_clock_drift()	319
10.17.3.4 sv_clock_relativistic_term()	319
10.17.4 Member Data Documentation	319
10.17.4.1 b_alert_flag	319
10.17.4.2 b_antispoofing_flag	320
10.17.4.3 b_fit_interval_flag	320
10.17.4.4 b_integrity_status_flag	320
10.17.4.5 d_A_f0	320
10.17.4.6 d_A_f1	321
10.17.4.7 d_A_f2	321
10.17.4.8 d_AODC	321
10.17.4.9 d_AODE	321
10.17.4.10d_Cic	322
10.17.4.11d_Cis	322
10.17.4.12d_Crc	322
10.17.4.13d_Crs	322
10.17.4.14d_Cuc	323
10.17.4.15d_Cus	323
10.17.4.16d_Delta_n	323
10.17.4.17d_dtr	323
10.17.4.18d_eccentricity	324
10.17.4.19d_i_0	324
10.17.4.20d_IDOT	324
10.17.4.21d_M_0	324
10.17.4.22d_OMEGA	325
10.17.4.23d_OMEGA0	325
10.17.4.24d_OMEGA_DOT	325
10.17.4.25d_satClkDrift	325

10.17.4.26d_satpos_X . . . . .	326
10.17.4.27d_satpos_Y . . . . .	326
10.17.4.28d_satpos_Z . . . . .	326
10.17.4.29d_satvel_X . . . . .	326
10.17.4.30d_satvel_Y . . . . .	326
10.17.4.31d_satvel_Z . . . . .	327
10.17.4.32d_sqrt_A . . . . .	327
10.17.4.33d_TGD1 . . . . .	327
10.17.4.34d_TGD2 . . . . .	327
10.17.4.35d_Toc . . . . .	327
10.17.4.36d_Toe . . . . .	328
10.17.4.37d_TOW . . . . .	328
10.17.4.38_AODO . . . . .	328
10.17.4.39_BEIDOU_week . . . . .	328
10.17.4.40_nav_type . . . . .	329
10.17.4.41i_satellite_PRN . . . . .	329
10.17.4.42_sig_type . . . . .	329
10.17.4.43_SV_accuracy . . . . .	329
10.17.4.44satelliteBlock . . . . .	329
10.18Beidou_Dnav_Iono Class Reference . . . . .	330
10.18.1 Detailed Description . . . . .	330
10.18.2 Constructor & Destructor Documentation . . . . .	330
10.18.2.1 Beidou_Dnav_Iono() . . . . .	331
10.18.3 Member Function Documentation . . . . .	331
10.18.3.1 serialize() . . . . .	331
10.18.4 Member Data Documentation . . . . .	331
10.18.4.1 d_alpha0 . . . . .	331
10.18.4.2 d_alpha1 . . . . .	331
10.18.4.3 d_alpha2 . . . . .	332
10.18.4.4 d_alpha3 . . . . .	332

10.18.4.5 d_beta0 . . . . .	332
10.18.4.6 d_beta1 . . . . .	332
10.18.4.7 d_beta2 . . . . .	333
10.18.4.8 d_beta3 . . . . .	333
10.18.4.9 valid . . . . .	333
10.19 Beidou_Dnav_Navigation_Message Class Reference . . . . .	333
10.19.1 Detailed Description . . . . .	334
10.19.2 Constructor & Destructor Documentation . . . . .	334
10.19.2.1 Beidou_Dnav_Navigation_Message() . . . . .	334
10.19.3 Member Function Documentation . . . . .	335
10.19.3.1 d1_subframe_decoder() . . . . .	335
10.19.3.2 d2_subframe_decoder() . . . . .	335
10.19.3.3 get_ephemeris() . . . . .	335
10.19.3.4 get_iono() . . . . .	335
10.19.3.5 get_utc_model() . . . . .	335
10.19.3.6 have_new_almanac() . . . . .	336
10.19.3.7 have_new_ephemeris() . . . . .	336
10.19.3.8 have_new_iono() . . . . .	336
10.19.3.9 have_new_utc_model() . . . . .	336
10.19.3.10 satellitePosition() . . . . .	336
10.19.3.11 set_satellite_PRN() . . . . .	336
10.19.3.12 sv_clock_correction() . . . . .	337
10.19.3.13 utc_time() . . . . .	337
10.20 Beidou_Dnav_Utc_Model Class Reference . . . . .	337
10.20.1 Detailed Description . . . . .	338
10.20.2 Member Data Documentation . . . . .	338
10.20.2.1 d_A0_GAL . . . . .	338
10.20.2.2 d_A0_GLO . . . . .	338
10.20.2.3 d_A0_GPS . . . . .	338
10.20.2.4 d_A0_UTC . . . . .	338

10.20.2.5 d_A1_GAL . . . . .	339
10.20.2.6 d_A1_GLO . . . . .	339
10.20.2.7 d_A1_GPS . . . . .	339
10.20.2.8 d_A1_UTC . . . . .	339
10.20.2.9 d_DeltaT_LSF . . . . .	339
10.20.2.10 d_DeltaT_LS . . . . .	340
10.20.2.11 i_DN . . . . .	340
10.20.2.12 a_WN_LSF . . . . .	340
10.21 BeidouB1iDIIPIITracking Class Reference . . . . .	340
10.21.1 Detailed Description . . . . .	341
10.21.2 Member Function Documentation . . . . .	341
10.21.2.1 set_channel() . . . . .	341
10.21.2.2 set_gnss_synchro() . . . . .	341
10.21.2.3 stop_tracking() . . . . .	342
10.22 BeidouB1iPcpsAcquisition Class Reference . . . . .	342
10.22.1 Detailed Description . . . . .	343
10.22.2 Member Function Documentation . . . . .	343
10.22.2.1 implementation() . . . . .	343
10.22.2.2 init() . . . . .	343
10.22.2.3 mag() . . . . .	344
10.22.2.4 reset() . . . . .	344
10.22.2.5 set_channel() . . . . .	344
10.22.2.6 set_channel_fsm() . . . . .	344
10.22.2.7 set_doppler_max() . . . . .	344
10.22.2.8 set_doppler_step() . . . . .	345
10.22.2.9 set_gnss_synchro() . . . . .	345
10.22.2.10 set_local_code() . . . . .	345
10.22.2.11 set_resampler_latency() . . . . .	345
10.22.2.12 set_state() . . . . .	345
10.22.2.13 set_threshold() . . . . .	346

10.22.2.14	<code>stop_acquisition()</code>	346
10.23	BeidouB1iTelemetryDecoder Class Reference	346
10.23.1	Detailed Description	347
10.23.2	Member Function Documentation	347
10.23.2.1	<code>implementation()</code>	347
10.24	BeidouB3iDIIPIITracking Class Reference	347
10.24.1	Detailed Description	348
10.24.2	Member Function Documentation	348
10.24.2.1	<code>set_channel()</code>	348
10.24.2.2	<code>set_gnss_synchro()</code>	348
10.24.2.3	<code>stop_tracking()</code>	349
10.25	BeidouB3iPcpsAcquisition Class Reference	349
10.25.1	Detailed Description	350
10.25.2	Member Function Documentation	350
10.25.2.1	<code>implementation()</code>	350
10.25.2.2	<code>init()</code>	350
10.25.2.3	<code>mag()</code>	351
10.25.2.4	<code>reset()</code>	351
10.25.2.5	<code>set_channel()</code>	351
10.25.2.6	<code>set_channel_fsm()</code>	351
10.25.2.7	<code>set_doppler_max()</code>	352
10.25.2.8	<code>set_doppler_step()</code>	352
10.25.2.9	<code>set_gnss_synchro()</code>	352
10.25.2.10	<code>set_local_code()</code>	352
10.25.2.11	<code>set_resampler_latency()</code>	352
10.25.2.12	<code>set_state()</code>	353
10.25.2.13	<code>set_threshold()</code>	353
10.25.2.14	<code>stop_acquisition()</code>	353
10.26	BeidouB3iTelemetryDecoder Class Reference	353
10.26.1	Detailed Description	354

10.26.2 Member Function Documentation . . . . .	354
10.26.2.1 implementation() . . . . .	354
10.27 byte_x2_to_complex_byte Class Reference . . . . .	354
10.27.1 Detailed Description . . . . .	355
10.28 ByteToShort Class Reference . . . . .	355
10.28.1 Detailed Description . . . . .	356
10.28.2 Member Function Documentation . . . . .	356
10.28.2.1 implementation() . . . . .	356
10.29 Channel Class Reference . . . . .	356
10.29.1 Detailed Description . . . . .	357
10.29.2 Constructor & Destructor Documentation . . . . .	357
10.29.2.1 Channel() . . . . .	358
10.29.2.2 ~Channel() . . . . .	358
10.29.3 Member Function Documentation . . . . .	358
10.29.3.1 connect() . . . . .	358
10.29.3.2 get_left_block_acq() . . . . .	358
10.29.3.3 get_left_block_trk() . . . . .	359
10.29.3.4 get_right_block() . . . . .	359
10.29.3.5 get_right_block_acq() . . . . .	359
10.29.3.6 get_right_block_trk() . . . . .	359
10.29.3.7 implementation() . . . . .	359
10.29.3.8 set_signal() . . . . .	360
10.29.3.9 start_acquisition() . . . . .	360
10.29.3.10 stop_channel() . . . . .	360
10.30 Channel_Event Class Reference . . . . .	360
10.30.1 Detailed Description . . . . .	361
10.31 channel_msg_receiver_cc Class Reference . . . . .	361
10.31.1 Detailed Description . . . . .	361
10.31.2 Constructor & Destructor Documentation . . . . .	361
10.31.2.1 ~channel_msg_receiver_cc() . . . . .	362

10.32channel_status_msg_receiver Class Reference . . . . .	362
10.32.1 Detailed Description . . . . .	362
10.32.2 Constructor & Destructor Documentation . . . . .	362
10.32.2.1 ~channel_status_msg_receiver() . . . . .	363
10.32.3 Member Function Documentation . . . . .	363
10.32.3.1 get_current_status_map() . . . . .	363
10.32.3.2 get_current_status_pvt() . . . . .	363
10.33ChannelFsm Class Reference . . . . .	363
10.33.1 Detailed Description . . . . .	364
10.34ChannelInterface Class Reference . . . . .	364
10.34.1 Detailed Description . . . . .	364
10.35cl_fft_plan Struct Reference . . . . .	365
10.35.1 Detailed Description . . . . .	365
10.36clFFT_Complex Struct Reference . . . . .	365
10.36.1 Detailed Description . . . . .	365
10.37clFFT_Dim3 Struct Reference . . . . .	366
10.37.1 Detailed Description . . . . .	366
10.38clFFT_SplitComplex Struct Reference . . . . .	366
10.38.1 Detailed Description . . . . .	366
10.39cnav_msg_decoder_t Struct Reference . . . . .	366
10.39.1 Detailed Description . . . . .	367
10.39.2 Member Data Documentation . . . . .	367
10.39.2.1 part1 . . . . .	367
10.39.2.2 part2 . . . . .	367
10.40cnav_msg_t Struct Reference . . . . .	367
10.40.1 Detailed Description . . . . .	368
10.40.2 Member Data Documentation . . . . .	368
10.40.2.1 alert . . . . .	368
10.40.2.2 msg_id . . . . .	368
10.40.2.3 prn . . . . .	368

10.40.2.4 raw_msg . . . . .	369
10.40.2.5 tow . . . . .	369
10.41cnav_v27_part_t Struct Reference . . . . .	369
10.41.1 Detailed Description . . . . .	369
10.41.2 Member Data Documentation . . . . .	370
10.41.2.1 crc_ok . . . . .	370
10.41.2.2 dec . . . . .	370
10.41.2.3 decisions . . . . .	370
10.41.2.4 decoded . . . . .	370
10.41.2.5 init . . . . .	370
10.41.2.6 invert . . . . .	371
10.41.2.7 message_lock . . . . .	371
10.41.2.8 n_crc_fail . . . . .	371
10.41.2.9 n_decoded . . . . .	371
10.41.2.10n_symbols . . . . .	371
10.41.2.11preamble_seen . . . . .	372
10.41.2.12symbols . . . . .	372
10.42Command_Event Class Reference . . . . .	372
10.42.1 Detailed Description . . . . .	372
10.43complex_byte_to_float_x2 Class Reference . . . . .	373
10.43.1 Detailed Description . . . . .	373
10.44complex_float_to_complex_byte Class Reference . . . . .	373
10.44.1 Detailed Description . . . . .	374
10.45Concurrent_Map< Data > Class Template Reference . . . . .	374
10.45.1 Detailed Description . . . . .	374
10.46Concurrent_Queue< Data > Class Template Reference . . . . .	375
10.46.1 Detailed Description . . . . .	375
10.47ConfigurationInterface Class Reference . . . . .	375
10.47.1 Detailed Description . . . . .	376
10.48conjugate_cc Class Reference . . . . .	376

10.48.1 Detailed Description . . . . .	377
10.49conjugate_ic Class Reference . . . . .	377
10.49.1 Detailed Description . . . . .	377
10.50conjugate_sc Class Reference . . . . .	378
10.50.1 Detailed Description . . . . .	378
10.51ControlThread Class Reference . . . . .	378
10.51.1 Detailed Description . . . . .	379
10.51.2 Constructor & Destructor Documentation . . . . .	379
10.51.2.1 ControlThread() [1/2] . . . . .	379
10.51.2.2 ControlThread() [2/2] . . . . .	379
10.51.2.3 ~ControlThread() . . . . .	380
10.51.3 Member Function Documentation . . . . .	380
10.51.3.1 flowgraph() . . . . .	380
10.51.3.2 run() . . . . .	380
10.51.3.3 set_control_queue() . . . . .	380
10.52Cpu_Multicorrelator Class Reference . . . . .	381
10.52.1 Detailed Description . . . . .	381
10.53Cpu_Multicorrelator_16sc Class Reference . . . . .	381
10.53.1 Detailed Description . . . . .	382
10.54Cpu_Multicorrelator_Real_Codes Class Reference . . . . .	382
10.54.1 Detailed Description . . . . .	382
10.55cshort_to_float_x2 Class Reference . . . . .	383
10.55.1 Detailed Description . . . . .	383
10.56CubatureFilter Class Reference . . . . .	383
10.56.1 Detailed Description . . . . .	384
10.57cuda_multicorrelator Class Reference . . . . .	384
10.57.1 Detailed Description . . . . .	384
10.58CustomUDPSignalSource Class Reference . . . . .	384
10.58.1 Detailed Description . . . . .	385
10.58.2 Member Function Documentation . . . . .	385

10.58.2.1 implementation()	385
10.59dgps_t Struct Reference	385
10.59.1 Detailed Description	386
10.60direct_resampler_conditioner_cb Class Reference	386
10.60.1 Detailed Description	386
10.61direct_resampler_conditioner_cc Class Reference	387
10.61.1 Detailed Description	387
10.62direct_resampler_conditioner_cs Class Reference	387
10.62.1 Detailed Description	388
10.63DirectResamplerConditioner Class Reference	388
10.63.1 Detailed Description	389
10.63.2 Member Function Documentation	389
10.63.2.1 implementation()	389
10.64Dll_Pll_Conf Class Reference	389
10.64.1 Detailed Description	390
10.65Dll_Pll_Conf_Fpga Class Reference	390
10.65.1 Detailed Description	392
10.66dll_pll_veml_tracking Class Reference	392
10.66.1 Detailed Description	392
10.67dll_pll_veml_tracking_fpga Class Reference	393
10.67.1 Detailed Description	393
10.67.2 Constructor & Destructor Documentation	393
10.67.2.1 ~dll_pll_veml_tracking_fpga()	394
10.67.3 Member Function Documentation	394
10.67.3.1 general_work()	394
10.67.3.2 reset()	394
10.67.3.3 set_channel()	394
10.67.3.4 set_gnss_synchro()	394
10.67.3.5 start_tracking()	395
10.67.3.6 stop_tracking()	395

10.68eph_t Struct Reference . . . . .	395
10.68.1 Detailed Description . . . . .	396
10.69erp_t Struct Reference . . . . .	396
10.69.1 Detailed Description . . . . .	396
10.70erpd_t Struct Reference . . . . .	396
10.70.1 Detailed Description . . . . .	396
10.71Exponential_Smoother Class Reference . . . . .	397
10.71.1 Detailed Description . . . . .	397
10.71.2 Constructor & Destructor Documentation . . . . .	397
10.71.2.1 Exponential_Smoother() [1/2] . . . . .	397
10.71.2.2 ~Exponential_Smoother() . . . . .	398
10.71.2.3 Exponential_Smoother() [2/2] . . . . .	398
10.71.3 Member Function Documentation . . . . .	398
10.71.3.1 operator=() . . . . .	398
10.71.3.2 set_alpha() . . . . .	398
10.71.3.3 set_samples_for_initialization() . . . . .	398
10.72exterr_t Struct Reference . . . . .	399
10.72.1 Detailed Description . . . . .	399
10.73fcbd_t Struct Reference . . . . .	399
10.73.1 Detailed Description . . . . .	399
10.74file_t Struct Reference . . . . .	400
10.74.1 Detailed Description . . . . .	400
10.75FileConfiguration Class Reference . . . . .	400
10.75.1 Detailed Description . . . . .	401
10.76FileSignalSource Class Reference . . . . .	401
10.76.1 Detailed Description . . . . .	402
10.76.2 Member Function Documentation . . . . .	402
10.76.2.1 implementation() . . . . .	402
10.77FirFilter Class Reference . . . . .	402
10.77.1 Detailed Description . . . . .	403

10.77.2 Constructor & Destructor Documentation . . . . .	403
10.77.2.1 FirFilter() . . . . .	403
10.77.2.2 ~FirFilter() . . . . .	403
10.77.3 Member Function Documentation . . . . .	404
10.77.3.1 implementation() . . . . .	404
10.78FlexibandSignalSource Class Reference . . . . .	404
10.78.1 Detailed Description . . . . .	405
10.78.2 Member Function Documentation . . . . .	405
10.78.2.1 implementation() . . . . .	405
10.79Fmcomms2SignalSource Class Reference . . . . .	405
10.79.1 Detailed Description . . . . .	406
10.79.2 Member Function Documentation . . . . .	406
10.79.2.1 implementation() . . . . .	406
10.80Fpga_Acquisition Class Reference . . . . .	406
10.80.1 Detailed Description . . . . .	407
10.80.2 Constructor & Destructor Documentation . . . . .	407
10.80.2.1 Fpga_Acquisition() . . . . .	407
10.80.2.2 ~Fpga_Acquisition() . . . . .	407
10.80.3 Member Function Documentation . . . . .	408
10.80.3.1 close_device() . . . . .	408
10.80.3.2 configure_acquisition() . . . . .	408
10.80.3.3 open_device() . . . . .	408
10.80.3.4 read_acquisition_results() . . . . .	408
10.80.3.5 read_fpga_total_scale_factor() . . . . .	408
10.80.3.6 reset_acquisition() . . . . .	409
10.80.3.7 run_acquisition() . . . . .	409
10.80.3.8 set_block_exp() . . . . .	409
10.80.3.9 set_doppler_max() . . . . .	409
10.80.3.10set_doppler_step() . . . . .	409
10.80.3.11set_doppler_sweep() . . . . .	410

10.80.3.12	<a href="#">set_local_code()</a>	410
10.80.3.13	<a href="#">write_local_code()</a>	410
10.81	<a href="#">Fpga_dynamic_bit_selection Class Reference</a>	410
10.81.1	<a href="#">Detailed Description</a>	411
10.81.2	<a href="#">Constructor &amp; Destructor Documentation</a>	411
10.81.2.1	<a href="#">Fpga_dynamic_bit_selection()</a>	411
10.81.2.2	<a href="#">~Fpga_dynamic_bit_selection()</a>	411
10.81.3	<a href="#">Member Function Documentation</a>	411
10.81.3.1	<a href="#">bit_selection()</a>	411
10.82	<a href="#">Fpga_Multicorrelator_8sc Class Reference</a>	412
10.82.1	<a href="#">Detailed Description</a>	413
10.82.2	<a href="#">Constructor &amp; Destructor Documentation</a>	413
10.82.2.1	<a href="#">Fpga_Multicorrelator_8sc()</a>	413
10.82.2.2	<a href="#">~Fpga_Multicorrelator_8sc()</a>	413
10.82.3	<a href="#">Member Function Documentation</a>	413
10.82.3.1	<a href="#">Carrier_wipeoff_multicorrelator_resampler()</a>	413
10.82.3.2	<a href="#">disable_secondary_codes()</a>	414
10.82.3.3	<a href="#">enable_secondary_codes()</a>	414
10.82.3.4	<a href="#">free()</a>	414
10.82.3.5	<a href="#">initialize_secondary_code()</a>	414
10.82.3.6	<a href="#">lock_channel()</a>	414
10.82.3.7	<a href="#">open_channel()</a>	414
10.82.3.8	<a href="#">read_sample_counter()</a>	415
10.82.3.9	<a href="#">set_initial_sample()</a>	415
10.82.3.10	<a href="#">set_local_code_and_taps()</a>	415
10.82.3.11	<a href="#">set_output_vectors()</a>	415
10.82.3.12	<a href="#">set_secondary_code_lengths()</a>	415
10.82.3.13	<a href="#">unlock_channel()</a>	416
10.82.3.14	<a href="#">update_local_code()</a>	416
10.82.3.15	<a href="#">update_prn_code_length()</a>	416

10.83Fpga_Switch Class Reference . . . . .	416
10.83.1 Detailed Description . . . . .	417
10.83.2 Constructor & Destructor Documentation . . . . .	417
10.83.2.1 Fpga_Switch() . . . . .	417
10.83.2.2 ~Fpga_Switch() . . . . .	417
10.83.3 Member Function Documentation . . . . .	417
10.83.3.1 set_switch_position() . . . . .	417
10.84FreqXlatingFirFilter Class Reference . . . . .	418
10.84.1 Detailed Description . . . . .	418
10.84.2 Member Function Documentation . . . . .	418
10.84.2.1 implementation() . . . . .	419
10.85FrontEndCal Class Reference . . . . .	419
10.85.1 Detailed Description . . . . .	419
10.85.2 Member Function Documentation . . . . .	419
10.85.2.1 estimate_doppler_from_eph() . . . . .	420
10.85.2.2 get_ephemeris() . . . . .	420
10.85.2.3 GPS_L1_front_end_model_E4000() . . . . .	420
10.85.2.4 set_configuration() . . . . .	420
10.86ftp_t Struct Reference . . . . .	421
10.86.1 Detailed Description . . . . .	421
10.87Galileo_Almanac Class Reference . . . . .	421
10.87.1 Detailed Description . . . . .	422
10.87.2 Constructor & Destructor Documentation . . . . .	422
10.87.2.1 Galileo_Almanac() . . . . .	422
10.87.3 Member Data Documentation . . . . .	422
10.87.3.1 d_A_f0 . . . . .	423
10.87.3.2 d_A_f1 . . . . .	423
10.87.3.3 d_Delta_i . . . . .	423
10.87.3.4 d_Delta_sqrt_A . . . . .	423
10.87.3.5 d_e_eccentricity . . . . .	423

10.87.3.6 d_M_0 . . . . .	424
10.87.3.7 d_OMEGA . . . . .	424
10.87.3.8 d_OMEGA0 . . . . .	424
10.87.3.9 d_OMEGA_DOT . . . . .	424
10.87.3.10 satellite_PRN . . . . .	424
10.88 Galileo_Almanac_Helper Class Reference . . . . .	425
10.88.1 Detailed Description . . . . .	426
10.88.2 Constructor & Destructor Documentation . . . . .	426
10.88.2.1 Galileo_Almanac_Helper() . . . . .	426
10.89 Galileo_Cnav_Message Class Reference . . . . .	426
10.89.1 Detailed Description . . . . .	427
10.90 Galileo_E1_Tcp_Connector_Tracking_cc Class Reference . . . . .	427
10.90.1 Detailed Description . . . . .	427
10.91 galileo_e5a_noncoherentIQ_acquisition_caf_cc Class Reference . . . . .	428
10.91.1 Detailed Description . . . . .	429
10.91.2 Constructor & Destructor Documentation . . . . .	429
10.91.2.1 ~galileo_e5a_noncoherentIQ_acquisition_caf_cc() . . . . .	429
10.91.3 Member Function Documentation . . . . .	429
10.91.3.1 general_work() . . . . .	429
10.91.3.2 init() . . . . .	429
10.91.3.3 mag() . . . . .	429
10.91.3.4 set_active() . . . . .	429
10.91.3.5 set_channel() . . . . .	430
10.91.3.6 set_channel_fsm() . . . . .	430
10.91.3.7 set_doppler_max() . . . . .	430
10.91.3.8 set_doppler_step() . . . . .	431
10.91.3.9 set_gnss_synchro() . . . . .	431
10.91.3.10 set_local_code() . . . . .	431
10.91.3.11 set_state() . . . . .	431
10.91.3.12 set_threshold() . . . . .	432

10.92Galileo_Ephemeris Class Reference . . . . .	432
10.92.1 Detailed Description . . . . .	434
10.92.2 Member Function Documentation . . . . .	434
10.92.2.1 Galileo_System_Time() . . . . .	435
10.92.2.2 satellitePosition() . . . . .	435
10.92.2.3 serialize() . . . . .	435
10.92.2.4 sv_clock_drift() . . . . .	435
10.92.2.5 sv_clock_relativistic_term() . . . . .	435
10.92.3 Member Data Documentation . . . . .	436
10.92.3.1 A_1 . . . . .	436
10.92.3.2 af0_4 . . . . .	436
10.92.3.3 af1_4 . . . . .	436
10.92.3.4 af2_4 . . . . .	436
10.92.3.5 BGD_E1E5a_5 . . . . .	437
10.92.3.6 BGD_E1E5b_5 . . . . .	437
10.92.3.7 C_ic_4 . . . . .	437
10.92.3.8 C_is_4 . . . . .	437
10.92.3.9 C_rc_3 . . . . .	438
10.92.3.10C_rs_3 . . . . .	438
10.92.3.11C_uc_3 . . . . .	438
10.92.3.12C_us_3 . . . . .	438
10.92.3.13d_satpos_X . . . . .	439
10.92.3.14d_satpos_Y . . . . .	439
10.92.3.15d_satpos_Z . . . . .	439
10.92.3.16d_satvel_X . . . . .	439
10.92.3.17d_satvel_Y . . . . .	439
10.92.3.18d_satvel_Z . . . . .	440
10.92.3.19delta_n_3 . . . . .	440
10.92.3.20E1B_DVS_5 . . . . .	440
10.92.3.21E1B_HS_5 . . . . .	440

10.92.3.22E5a_DVS . . . . .	441
10.92.3.23E5a_HS . . . . .	441
10.92.3.24E5b_DVS_5 . . . . .	441
10.92.3.25E5b_HS_5 . . . . .	441
10.92.3.26e_1 . . . . .	442
10.92.3.27Galileo_dtr . . . . .	442
10.92.3.28_0_2 . . . . .	442
10.92.3.29_satellite_PRN . . . . .	442
10.92.3.30Dot_2 . . . . .	443
10.92.3.31M0_1 . . . . .	443
10.92.3.32OMEGA_0_2 . . . . .	443
10.92.3.33omega_2 . . . . .	443
10.92.3.34OMEGA_dot_3 . . . . .	444
10.92.3.350c_4 . . . . .	444
10.92.3.360e_1 . . . . .	444
10.92.3.37TOW_5 . . . . .	444
10.92.3.38WN_5 . . . . .	445
10.93Galileo_Fnav_Message Class Reference . . . . .	445
10.93.1 Detailed Description . . . . .	446
10.94Galileo_HAS_data Class Reference . . . . .	446
10.94.1 Detailed Description . . . . .	447
10.95Galileo_Inav_Message Class Reference . . . . .	447
10.95.1 Detailed Description . . . . .	448
10.96Galileo_Iono Class Reference . . . . .	448
10.96.1 Detailed Description . . . . .	449
10.96.2 Constructor & Destructor Documentation . . . . .	449
10.96.2.1 Galileo_Iono() . . . . .	449
10.96.3 Member Function Documentation . . . . .	449
10.96.3.1 serialize() . . . . .	449
10.96.4 Member Data Documentation . . . . .	449

10.96.4.1 ai0_5 . . . . .	450
10.96.4.2 ai1_5 . . . . .	450
10.96.4.3 ai2_5 . . . . .	450
10.96.4.4 Region1_flag_5 . . . . .	450
10.96.4.5 Region2_flag_5 . . . . .	451
10.96.4.6 Region3_flag_5 . . . . .	451
10.96.4.7 Region4_flag_5 . . . . .	451
10.96.4.8 Region5_flag_5 . . . . .	451
10.96.4.9 TOW_5 . . . . .	452
10.96.4.10WN_5 . . . . .	452
10.97galileo_pcps_8ms_acquisition_cc Class Reference . . . . .	452
10.97.1 Detailed Description . . . . .	453
10.97.2 Constructor & Destructor Documentation . . . . .	453
10.97.2.1 ~galileo_pcps_8ms_acquisition_cc() . . . . .	454
10.97.3 Member Function Documentation . . . . .	454
10.97.3.1 general_work() . . . . .	454
10.97.3.2 init() . . . . .	454
10.97.3.3 mag() . . . . .	454
10.97.3.4 set_active() . . . . .	454
10.97.3.5 set_channel() . . . . .	455
10.97.3.6 set_channel_fsm() . . . . .	455
10.97.3.7 set_doppler_max() . . . . .	455
10.97.3.8 set_doppler_step() . . . . .	456
10.97.3.9 set_gnss_synchro() . . . . .	456
10.97.3.10set_local_code() . . . . .	456
10.97.3.11set_state() . . . . .	456
10.97.3.12set_threshold() . . . . .	458
10.98galileo_telemetry_decoder_gs Class Reference . . . . .	458
10.98.1 Detailed Description . . . . .	459
10.98.2 Member Function Documentation . . . . .	459

10.98.2.1	<code>general_work()</code>	459
10.98.2.2	<code>set_channel()</code>	459
10.98.2.3	<code>set_satellite()</code>	459
10.99	<code>Galileo_Utc_Model</code> Class Reference	460
10.99.1	Detailed Description	460
10.99.2	Constructor & Destructor Documentation	460
10.99.2.1	<code>Galileo_Utc_Model()</code>	461
10.99.3	Member Function Documentation	461
10.99.3.1	<code>GST_to_UTC_time()</code>	461
10.99.3.2	<code>serialize()</code>	461
10.99.4	Member Data Documentation	461
10.99.4.1	<code>t0t_6</code>	461
10.99.4.2	<code>WNot_6</code>	462
10.100	<code>GalileoE1BTelemetryDecoder</code> Class Reference	462
10.100.1	Detailed Description	462
10.100.2	Member Function Documentation	463
10.100.2.1	<code>implementation()</code>	463
10.101	<code>GalileoE1DIIPIVemlTracking</code> Class Reference	463
10.101.1	Detailed Description	464
10.101.2	Member Function Documentation	464
10.101.2.1	<code>implementation()</code>	464
10.101.2.2	<code>set_channel()</code>	464
10.101.2.3	<code>set_gnss_synchro()</code>	464
10.101.2.4	<code>stop_tracking()</code>	465
10.102	<code>GalileoE1DIIPIVemlTrackingFpga</code> Class Reference	465
10.102.1	Detailed Description	466
10.102.2	Constructor & Destructor Documentation	466
10.102.2.1	<code>GalileoE1DIIPIVemlTrackingFpga()</code>	466
10.102.2.2	<code>~GalileoE1DIIPIVemlTrackingFpga()</code>	466
10.102.3	Member Function Documentation	466

10.102.3.1	connect()	467
10.102.3.2	disconnect()	467
10.102.3.3	get_left_block()	467
10.102.3.4	get_right_block()	467
10.102.3.5	implementation()	467
10.102.3.6	item_size()	468
10.102.3.7	role()	468
10.102.3.8	set_channel()	468
10.102.3.9	set_gnss_synchro()	468
10.102.3.10	start_tracking()	469
10.102.3.11	stop_tracking()	469
10.103	GalileoE1Pcps8msAmbiguousAcquisition Class Reference	469
10.103.1	Detailed Description	470
10.103.2	Member Function Documentation	470
10.103.2.1	implementation()	471
10.103.2.2	init()	471
10.103.2.3	mag()	471
10.103.2.4	reset()	471
10.103.2.5	set_channel()	471
10.103.2.6	set_channel_fsm()	472
10.103.2.7	set_doppler_max()	472
10.103.2.8	set_doppler_step()	472
10.103.2.9	set_gnss_synchro()	472
10.103.2.10	set_local_code()	473
10.103.2.11	set_threshold()	473
10.103.2.12	stop_acquisition()	473
10.104	GalileoE1PcpsAmbiguousAcquisition Class Reference	473
10.104.1	Detailed Description	474
10.104.2	Member Function Documentation	474
10.104.2.1	implementation()	475

10.104.2.2	<code>nit()</code>	475
10.104.2.3	<code>mag()</code>	475
10.104.2.4	<code>reset()</code>	475
10.104.2.5	<code>set_channel()</code>	475
10.104.2.6	<code>set_channel_fsm()</code>	476
10.104.2.7	<code>set_doppler_center()</code>	476
10.104.2.8	<code>set_doppler_max()</code>	476
10.104.2.9	<code>set_doppler_step()</code>	476
10.104.2.10	<code>set_gnss_synchro()</code>	476
10.104.2.11	<code>set_local_code()</code>	477
10.104.2.12	<code>set_resampler_latency()</code>	477
10.104.2.13	<code>set_state()</code>	477
10.104.2.14	<code>set_threshold()</code>	477
10.104.2.15	<code>stop_acquisition()</code>	477
10.105	<b>GalileoE1PcpsAmbiguousAcquisitionFpga Class Reference</b>	478
10.105.1	<b>Detailed Description</b>	479
10.105.2	<b>Constructor &amp; Destructor Documentation</b>	479
10.105.2.1	<code>GalileoE1PcpsAmbiguousAcquisitionFpga()</code>	479
10.105.2.2	<code>~GalileoE1PcpsAmbiguousAcquisitionFpga()</code>	479
10.105.3	<b>Member Function Documentation</b>	479
10.105.3.1	<code>connect()</code>	480
10.105.3.2	<code>disconnect()</code>	480
10.105.3.3	<code>get_left_block()</code>	480
10.105.3.4	<code>get_right_block()</code>	480
10.105.3.5	<code>implementation()</code>	480
10.105.3.6	<code>nit()</code>	481
10.105.3.7	<code>item_size()</code>	481
10.105.3.8	<code>mag()</code>	481
10.105.3.9	<code>reset()</code>	481
10.105.3.10	<code>le()</code>	481

10.105.3.1	<a href="#">set_channel()</a>	482
10.105.3.1	<a href="#">set_channel_fsm()</a>	482
10.105.3.1	<a href="#">set_doppler_center()</a>	482
10.105.3.1	<a href="#">set_doppler_max()</a>	482
10.105.3.1	<a href="#">set_doppler_step()</a>	482
10.105.3.1	<a href="#">set_gnss_synchro()</a>	483
10.105.3.1	<a href="#">set_local_code()</a>	483
10.105.3.1	<a href="#">set_resampler_latency()</a>	483
10.105.3.1	<a href="#">set_state()</a>	483
10.105.3.2	<a href="#">set_threshold()</a>	483
10.105.3.2	<a href="#">stop_acquisition()</a>	484
10.106	<a href="#">GalileoE1PcpsCccwsrAmbiguousAcquisition Class Reference</a>	484
10.106.1	<a href="#">Detailed Description</a>	485
10.106.2	<a href="#">Member Function Documentation</a>	485
10.106.2.1	<a href="#">implementation()</a>	485
10.106.2.2	<a href="#">nit()</a>	485
10.106.2.3	<a href="#">mag()</a>	486
10.106.2.4	<a href="#">reset()</a>	486
10.106.2.5	<a href="#">set_channel()</a>	486
10.106.2.6	<a href="#">set_channel_fsm()</a>	486
10.106.2.7	<a href="#">set_doppler_max()</a>	487
10.106.2.8	<a href="#">set_doppler_step()</a>	487
10.106.2.9	<a href="#">set_gnss_synchro()</a>	487
10.106.2.10	<a href="#">set_state()</a>	487
10.106.2.11	<a href="#">set_threshold()</a>	487
10.106.2.12	<a href="#">stop_acquisition()</a>	488
10.107	<a href="#">GalileoE1PcpsQuickSyncAmbiguousAcquisition Class Reference</a>	488
10.107.1	<a href="#">Detailed Description</a>	489
10.107.2	<a href="#">Member Function Documentation</a>	489
10.107.2.1	<a href="#">implementation()</a>	489

10.107.2.2	<a href="#">nit()</a>	489
10.107.2.3	<a href="#">mag()</a>	490
10.107.2.4	<a href="#">reset()</a>	490
10.107.2.5	<a href="#">set_channel()</a>	490
10.107.2.6	<a href="#">set_channel_fsm()</a>	490
10.107.2.7	<a href="#">set_doppler_max()</a>	491
10.107.2.8	<a href="#">set_doppler_step()</a>	491
10.107.2.9	<a href="#">set_gnss_synchro()</a>	491
10.107.2.10	<a href="#">set_local_code()</a>	491
10.107.2.11	<a href="#">set_state()</a>	491
10.107.2.12	<a href="#">set_threshold()</a>	492
10.107.2.13	<a href="#">stop_acquisition()</a>	492
10.108	<a href="#">GalileoE1PcpsTongAmbiguousAcquisition Class Reference</a>	492
10.108.1	<a href="#">Detailed Description</a>	493
10.108.2	<a href="#">Member Function Documentation</a>	493
10.108.2.1	<a href="#">implementation()</a>	494
10.108.2.2	<a href="#">nit()</a>	494
10.108.2.3	<a href="#">mag()</a>	494
10.108.2.4	<a href="#">reset()</a>	494
10.108.2.5	<a href="#">set_channel()</a>	494
10.108.2.6	<a href="#">set_channel_fsm()</a>	495
10.108.2.7	<a href="#">set_doppler_max()</a>	495
10.108.2.8	<a href="#">set_doppler_step()</a>	495
10.108.2.9	<a href="#">set_gnss_synchro()</a>	495
10.108.2.10	<a href="#">set_local_code()</a>	496
10.108.2.11	<a href="#">set_state()</a>	496
10.108.2.12	<a href="#">set_threshold()</a>	496
10.108.2.13	<a href="#">stop_acquisition()</a>	496
10.109	<a href="#">GalileoE1TcpConnectorTracking Class Reference</a>	497
10.109.1	<a href="#">Detailed Description</a>	497

10.109.2	Member Function Documentation	497
10.109.2.1	implementation()	498
10.109.2.2	set_channel()	498
10.109.2.3	set_gnss_synchro()	498
10.109.2.4	stop_tracking()	498
10.110	GalileoE5aDIIPIITracking Class Reference	499
10.110.1	Detailed Description	499
10.110.2	Member Function Documentation	499
10.110.2.1	implementation()	500
10.110.2.2	set_channel()	500
10.110.2.3	set_gnss_synchro()	500
10.110.2.4	stop_tracking()	500
10.111	GalileoE5aDIIPIITrackingFpga Class Reference	501
10.111.1	Detailed Description	501
10.111.2	Constructor & Destructor Documentation	502
10.111.2.1	GalileoE5aDIIPIITrackingFpga()	502
10.111.2.2	~GalileoE5aDIIPIITrackingFpga()	502
10.111.3	Member Function Documentation	502
10.111.3.1	connect()	502
10.111.3.2	disconnect()	502
10.111.3.3	get_left_block()	503
10.111.3.4	get_right_block()	503
10.111.3.5	implementation()	503
10.111.3.6	item_size()	503
10.111.3.7	role()	503
10.111.3.8	set_channel()	504
10.111.3.9	set_gnss_synchro()	504
10.111.3.10	start_tracking()	504
10.111.3.11	stop_tracking()	504
10.112	GalileoE5aNoncoherentIQAcquisitionCaf Class Reference	505

10.112.1	Detailed Description	506
10.112.2	Member Function Documentation	506
10.112.2.1	implementation()	506
10.112.2.2	init()	506
10.112.2.3	mag()	506
10.112.2.4	reset()	506
10.112.2.5	set_channel()	507
10.112.2.6	set_channel_fsm()	507
10.112.2.7	set_doppler_max()	507
10.112.2.8	set_doppler_step()	507
10.112.2.9	set_gnss_synchro()	508
10.112.2.10	set_local_code()	508
10.112.2.11	set_state()	508
10.112.2.12	set_threshold()	508
10.112.2.13	stop_acquisition()	509
10.113	GalileoE5aPcpsAcquisition Class Reference	509
10.113.1	Detailed Description	510
10.113.2	Member Function Documentation	510
10.113.2.1	init()	510
10.113.2.2	mag()	510
10.113.2.3	reset()	511
10.113.2.4	set_channel()	511
10.113.2.5	set_channel_fsm()	511
10.113.2.6	set_doppler_center()	511
10.113.2.7	set_doppler_max()	511
10.113.2.8	set_doppler_step()	512
10.113.2.9	set_gnss_synchro()	512
10.113.2.10	set_local_code()	512
10.113.2.11	set_resampler_latency()	512
10.113.2.12	set_state()	512

10.113.2.1	<a href="#">set_threshold()</a> . . . . .	513
10.113.2.1	<a href="#">stop_acquisition()</a> . . . . .	513
10.114	<a href="#">GalileoE5aPcpsAcquisitionFpga Class Reference</a> . . . . .	513
10.114.1	<a href="#">Detailed Description</a> . . . . .	515
10.114.2	<a href="#">Constructor &amp; Destructor Documentation</a> . . . . .	515
10.114.2.1	<a href="#">GalileoE5aPcpsAcquisitionFpga()</a> . . . . .	515
10.114.2.2	<a href="#">~GalileoE5aPcpsAcquisitionFpga()</a> . . . . .	515
10.114.3	<a href="#">Member Function Documentation</a> . . . . .	515
10.114.3.1	<a href="#">connect()</a> . . . . .	515
10.114.3.2	<a href="#">disconnect()</a> . . . . .	516
10.114.3.3	<a href="#">get_left_block()</a> . . . . .	516
10.114.3.4	<a href="#">get_right_block()</a> . . . . .	516
10.114.3.5	<a href="#">implementation()</a> . . . . .	516
10.114.3.6	<a href="#">init()</a> . . . . .	516
10.114.3.7	<a href="#">item_size()</a> . . . . .	517
10.114.3.8	<a href="#">mag()</a> . . . . .	517
10.114.3.9	<a href="#">reset()</a> . . . . .	517
10.114.3.10	<a href="#">role()</a> . . . . .	517
10.114.3.11	<a href="#">set_channel()</a> . . . . .	517
10.114.3.12	<a href="#">set_channel_fsm()</a> . . . . .	518
10.114.3.13	<a href="#">set_doppler_center()</a> . . . . .	518
10.114.3.14	<a href="#">set_doppler_max()</a> . . . . .	518
10.114.3.15	<a href="#">set_doppler_step()</a> . . . . .	518
10.114.3.16	<a href="#">set_gnss_synchro()</a> . . . . .	518
10.114.3.17	<a href="#">set_local_code()</a> . . . . .	519
10.114.3.18	<a href="#">set_resampler_latency()</a> . . . . .	519
10.114.3.19	<a href="#">set_single_doppler_flag()</a> . . . . .	519
10.114.3.20	<a href="#">set_state()</a> . . . . .	519
10.114.3.21	<a href="#">set_threshold()</a> . . . . .	519
10.114.3.22	<a href="#">stop_acquisition()</a> . . . . .	520

10.115	<a href="#">GalileoE5aTelemetryDecoder Class Reference</a>	520
10.115.1	<a href="#">Detailed Description</a>	521
10.115.2	<a href="#">Member Function Documentation</a>	521
10.115.2.1	<a href="#">implementation()</a>	521
10.116	<a href="#">GalileoE5bDIPIITracking Class Reference</a>	521
10.116.1	<a href="#">Detailed Description</a>	522
10.116.2	<a href="#">Member Function Documentation</a>	522
10.116.2.1	<a href="#">connect()</a>	522
10.116.2.2	<a href="#">disconnect()</a>	523
10.116.2.3	<a href="#">get_left_block()</a>	523
10.116.2.4	<a href="#">get_right_block()</a>	523
10.116.2.5	<a href="#">implementation()</a>	523
10.116.2.6	<a href="#">set_channel()</a>	523
10.116.2.7	<a href="#">set_gnss_synchro()</a>	524
10.116.2.8	<a href="#">stop_tracking()</a>	524
10.117	<a href="#">GalileoE5bPcpsAcquisition Class Reference</a>	524
10.117.1	<a href="#">Detailed Description</a>	525
10.117.2	<a href="#">Constructor &amp; Destructor Documentation</a>	525
10.117.2.1	<a href="#">GalileoE5bPcpsAcquisition()</a>	526
10.117.2.2	<a href="#">~GalileoE5bPcpsAcquisition()</a>	526
10.117.3	<a href="#">Member Function Documentation</a>	526
10.117.3.1	<a href="#">connect()</a>	526
10.117.3.2	<a href="#">disconnect()</a>	526
10.117.3.3	<a href="#">get_left_block()</a>	526
10.117.3.4	<a href="#">get_right_block()</a>	527
10.117.3.5	<a href="#">implementation()</a>	527
10.117.3.6	<a href="#">init()</a>	527
10.117.3.7	<a href="#">item_size()</a>	527
10.117.3.8	<a href="#">mag()</a>	527
10.117.3.9	<a href="#">reset()</a>	528

10.117.3.10	le()	528
10.117.3.11	set_channel()	528
10.117.3.12	set_channel_fsm()	528
10.117.3.13	set_doppler_center()	529
10.117.3.14	set_doppler_max()	529
10.117.3.15	set_doppler_step()	529
10.117.3.16	set_gnss_synchro()	529
10.117.3.17	set_local_code()	529
10.117.3.18	set_resampler_latency()	530
10.117.3.19	set_state()	530
10.117.3.20	set_threshold()	530
10.117.3.21	stop_acquisition()	530
10.118	GalileoE5bPcpsAcquisitionFpga Class Reference	531
10.118.1	Detailed Description	532
10.118.2	Constructor & Destructor Documentation	532
10.118.2.1	GalileoE5bPcpsAcquisitionFpga()	532
10.118.2.2	~GalileoE5bPcpsAcquisitionFpga()	532
10.118.3	Member Function Documentation	533
10.118.3.1	connect()	533
10.118.3.2	disconnect()	533
10.118.3.3	get_left_block()	533
10.118.3.4	get_right_block()	533
10.118.3.5	implementation()	534
10.118.3.6	init()	534
10.118.3.7	item_size()	534
10.118.3.8	mag()	534
10.118.3.9	reset()	534
10.118.3.10	le()	535
10.118.3.11	set_channel()	535
10.118.3.12	set_channel_fsm()	535

10.118.3.13	<a href="#">set_doppler_center()</a> . . . . .	535
10.118.3.14	<a href="#">set_doppler_max()</a> . . . . .	536
10.118.3.15	<a href="#">set_doppler_step()</a> . . . . .	536
10.118.3.16	<a href="#">set_gnss_synchro()</a> . . . . .	536
10.118.3.17	<a href="#">set_local_code()</a> . . . . .	536
10.118.3.18	<a href="#">set_resampler_latency()</a> . . . . .	536
10.118.3.19	<a href="#">set_single_doppler_flag()</a> . . . . .	537
10.118.3.20	<a href="#">set_state()</a> . . . . .	537
10.118.3.21	<a href="#">set_threshold()</a> . . . . .	537
10.118.3.22	<a href="#">stop_acquisition()</a> . . . . .	537
10.119	<a href="#">GalileoE5bTelemetryDecoder Class Reference</a> . . . . .	538
10.119.1	<a href="#">Detailed Description</a> . . . . .	538
10.119.2	<a href="#">Member Function Documentation</a> . . . . .	538
10.119.2.1	<a href="#">connect()</a> . . . . .	539
10.119.2.2	<a href="#">disconnect()</a> . . . . .	539
10.119.2.3	<a href="#">get_left_block()</a> . . . . .	539
10.119.2.4	<a href="#">get_right_block()</a> . . . . .	539
10.119.2.5	<a href="#">implementation()</a> . . . . .	539
10.120	<a href="#">GalileoE6DIIPIITracking Class Reference</a> . . . . .	540
10.120.1	<a href="#">Detailed Description</a> . . . . .	540
10.120.2	<a href="#">Member Function Documentation</a> . . . . .	540
10.120.2.1	<a href="#">connect()</a> . . . . .	541
10.120.2.2	<a href="#">disconnect()</a> . . . . .	541
10.120.2.3	<a href="#">get_left_block()</a> . . . . .	541
10.120.2.4	<a href="#">get_right_block()</a> . . . . .	541
10.120.2.5	<a href="#">implementation()</a> . . . . .	541
10.120.2.6	<a href="#">set_channel()</a> . . . . .	542
10.120.2.7	<a href="#">set_gnss_synchro()</a> . . . . .	542
10.120.2.8	<a href="#">stop_tracking()</a> . . . . .	542
10.121	<a href="#">GalileoE6PcpsAcquisition Class Reference</a> . . . . .	542

10.121.1	Detailed Description	543
10.121.2	Member Function Documentation	543
10.121.2.1	implementation()	544
10.121.2.2	init()	544
10.121.2.3	mag()	544
10.121.2.4	reset()	544
10.121.2.5	set_channel()	544
10.121.2.6	set_channel_fsm()	545
10.121.2.7	set_doppler_center()	545
10.121.2.8	set_doppler_max()	545
10.121.2.9	set_doppler_step()	545
10.121.2.10	set_gnss_synchro()	545
10.121.2.11	set_local_code()	546
10.121.2.12	set_resampler_latency()	546
10.121.2.13	set_state()	546
10.121.2.14	set_threshold()	546
10.121.2.15	stop_acquisition()	546
10.122	GalileoE6TelemetryDecoder Class Reference	547
10.122.1	Detailed Description	547
10.122.2	Member Function Documentation	547
10.122.2.1	connect()	548
10.122.2.2	disconnect()	548
10.122.2.3	get_left_block()	548
10.122.2.4	get_right_block()	548
10.122.2.5	implementation()	548
10.123	GenSignalSource Class Reference	549
10.123.1	Detailed Description	549
10.123.2	Constructor & Destructor Documentation	549
10.123.2.1	GenSignalSource()	549
10.123.2.2	~GenSignalSource()	550

10.123.3	Member Function Documentation	550
10.123.3.1	implementation()	550
10.124	GeoJSON_Printer Class Reference	550
10.124.1	Detailed Description	550
10.125	eph_t Struct Reference	551
10.125.1	Detailed Description	551
10.126	Glonass_Gnav_Almanac Class Reference	551
10.126.1	Detailed Description	552
10.126.2	Constructor & Destructor Documentation	553
10.126.2.1	Glonass_Gnav_Almanac()	553
10.126.3	Member Function Documentation	553
10.126.3.1	serialize()	553
10.126.4	Member Data Documentation	553
10.126.4.1	d_C_n	553
10.126.4.2	d_Delta_i_n_A	554
10.126.4.3	d_Delta_T_n_A	554
10.126.4.4	d_Delta_T_n_A_dot	554
10.126.4.5	d_epsilon_n_A	554
10.126.4.6	d_H_n_A	555
10.126.4.7	d_KP	555
10.126.4.8	d_I_n	555
10.126.4.9	d_lambda_n_A	555
10.126.4.10	d_M_n_A	556
10.126.4.11	d_n_A	556
10.126.4.12	d_omega_n_A	556
10.126.4.13	d_t_lambda_n_A	556
10.126.4.14	d_tau_n_A	557
10.126.4.15	satellite_freq_channel	557
10.126.4.16	satellite_PRN	557
10.126.4.17	satellite_slot_number	557

10.127.1 <a href="#">Glonass_Gnav_Ephemeris Class Reference</a>	558
10.127.2 <a href="#">Detailed Description</a>	560
10.127.3 <a href="#">Constructor &amp; Destructor Documentation</a>	560
10.127.4 <a href="#">Glonass_Gnav_Ephemeris()</a>	560
10.127.5 <a href="#">Member Function Documentation</a>	560
10.127.6 <a href="#">compute_GLONASS_time()</a>	561
10.127.7 <a href="#">glot_to_gpst()</a>	561
10.127.8 <a href="#">glot_to_utc()</a>	561
10.127.9 <a href="#">serialize()</a>	562
10.127.10 <a href="#">sv_clock_drift()</a>	563
10.127.11 <a href="#">Member Data Documentation</a>	563
10.127.12 <a href="#">d_AXn</a>	563
10.127.13 <a href="#">d_AYn</a>	564
10.127.14 <a href="#">d_AZn</a>	564
10.127.15 <a href="#">d_B_n</a>	564
10.127.16 <a href="#">d_Delta_tau_n</a>	564
10.127.17 <a href="#">d_dtr</a>	565
10.127.18 <a href="#">d_E_n</a>	565
10.127.19 <a href="#">d_F_T</a>	565
10.127.20 <a href="#">d_gamma_n</a>	565
10.127.21 <a href="#">d_iode</a>	566
10.127.22 <a href="#">d_l3rd_n</a>	566
10.127.23 <a href="#">d_l5th_n</a>	566
10.127.24 <a href="#">d_m</a>	566
10.127.25 <a href="#">d_M</a>	567
10.127.26 <a href="#">d_n</a>	567
10.127.27 <a href="#">d_N_T</a>	567
10.127.28 <a href="#">d_P</a>	567
10.127.29 <a href="#">d_P_1</a>	568
10.127.30 <a href="#">d_P_2</a>	568

10.127.4.20_P_3	568
10.127.4.21_P_4	568
10.127.4.22_satClkDrift	569
10.127.4.23_t_b	569
10.127.4.24_t_k	569
10.127.4.25_tau_c	569
10.127.4.26_tau_n	569
10.127.4.27_tod	570
10.127.4.28_TOW	570
10.127.4.29_VXn	570
10.127.4.30_VYn	570
10.127.4.31_VZn	570
10.127.4.32_WN	571
10.127.4.33_Xn	571
10.127.4.34_Yn	571
10.127.4.35_yr	571
10.127.4.36_Zn	571
10.127.4.37_satellite_freq_channel	572
10.127.4.38_satellite_PRN	572
10.127.4.39_satellite_slot_number	572
10.128.Glonass_Gnav_Navigation_Message Class Reference	572
10.128.1.Detailed Description	573
10.128.2.Constructor & Destructor Documentation	573
10.128.2.1Glonass_Gnav_Navigation_Message()	574
10.128.3.Member Function Documentation	574
10.128.3.1CRC_test()	574
10.128.3.2get_almanac()	574
10.128.3.3get_ephemeris()	574
10.128.3.4get_frame_number()	575
10.128.3.5get_utc_model()	575

10.128.3.6	<a href="#">have_new_almanac()</a> . . . . .	575
10.128.3.7	<a href="#">have_new_ephemeris()</a> . . . . .	575
10.128.3.8	<a href="#">have_new_utc_model()</a> . . . . .	575
10.128.3.9	<a href="#">string_decoder()</a> . . . . .	575
10.129.0	<a href="#">Glonass_Gnav_Utc_Model Class Reference</a> . . . . .	576
10.129.1	<a href="#">Detailed Description</a> . . . . .	577
10.129.2	<a href="#">Constructor &amp; Destructor Documentation</a> . . . . .	577
10.129.2.1	<a href="#">Glonass_Gnav_Utc_Model()</a> . . . . .	577
10.129.3	<a href="#">Member Function Documentation</a> . . . . .	577
10.129.3.1	<a href="#">serialize()</a> . . . . .	577
10.129.3.2	<a href="#">utc_time()</a> . . . . .	578
10.129.4	<a href="#">Member Data Documentation</a> . . . . .	578
10.129.4.1	<a href="#">d_B1</a> . . . . .	578
10.129.4.2	<a href="#">d_B2</a> . . . . .	578
10.129.4.3	<a href="#">d_N_4</a> . . . . .	578
10.129.4.4	<a href="#">d_N_A</a> . . . . .	579
10.129.4.5	<a href="#">d_tau_c</a> . . . . .	579
10.129.4.6	<a href="#">d_tau_gps</a> . . . . .	579
10.130.0	<a href="#">Glonass_I1_ca_dll_pll_c_aid_tracking_cc Class Reference</a> . . . . .	579
10.130.1	<a href="#">Detailed Description</a> . . . . .	580
10.131.0	<a href="#">Glonass_I1_ca_dll_pll_c_aid_tracking_sc Class Reference</a> . . . . .	580
10.131.1	<a href="#">Detailed Description</a> . . . . .	581
10.132.0	<a href="#">Glonass_L1_Ca_Dll_Pll_Tracking_cc Class Reference</a> . . . . .	581
10.132.1	<a href="#">Detailed Description</a> . . . . .	582
10.133.0	<a href="#">Glonass_I1_ca_telemetry_decoder_gs Class Reference</a> . . . . .	582
10.133.1	<a href="#">Detailed Description</a> . . . . .	583
10.133.2	<a href="#">Constructor &amp; Destructor Documentation</a> . . . . .	583
10.133.2.1	<a href="#">~glonass_I1_ca_telemetry_decoder_gs()</a> . . . . .	583
10.133.3	<a href="#">Member Function Documentation</a> . . . . .	583
10.133.3.1	<a href="#">general_work()</a> . . . . .	583

10.133.3.2	<a href="#">set_channel()</a> . . . . .	583
10.133.3.3	<a href="#">set_satellite()</a> . . . . .	584
10.134	<a href="#">Glonass_L2_ca_dll_pll_c_aid_tracking_cc Class Reference</a> . . . . .	584
10.134.1	Detailed Description . . . . .	584
10.135	<a href="#">Glonass_L2_ca_dll_pll_c_aid_tracking_sc Class Reference</a> . . . . .	585
10.135.1	Detailed Description . . . . .	585
10.136	<a href="#">Glonass_L2_Ca_DLL_Pll_Tracking_cc Class Reference</a> . . . . .	585
10.136.1	Detailed Description . . . . .	586
10.137	<a href="#">Glonass_L2_ca_telemetry_decoder_gs Class Reference</a> . . . . .	586
10.137.1	Detailed Description . . . . .	587
10.137.2	Constructor & Destructor Documentation . . . . .	587
10.137.2.1	<a href="#">Glonass_L2_ca_telemetry_decoder_gs()</a> . . . . .	587
10.137.3	Member Function Documentation . . . . .	587
10.137.3.1	<a href="#">general_work()</a> . . . . .	587
10.137.3.2	<a href="#">set_channel()</a> . . . . .	587
10.137.3.3	<a href="#">set_satellite()</a> . . . . .	588
10.138	<a href="#">GlonassL1CaDllPllCAidTracking Class Reference</a> . . . . .	588
10.138.1	Detailed Description . . . . .	589
10.138.2	Member Function Documentation . . . . .	589
10.138.2.1	<a href="#">implementation()</a> . . . . .	589
10.138.2.2	<a href="#">set_channel()</a> . . . . .	589
10.138.2.3	<a href="#">set_gnss_synchro()</a> . . . . .	589
10.138.2.4	<a href="#">stop_tracking()</a> . . . . .	590
10.139	<a href="#">GlonassL1CaDllPllTracking Class Reference</a> . . . . .	590
10.139.1	Detailed Description . . . . .	591
10.139.2	Member Function Documentation . . . . .	591
10.139.2.1	<a href="#">implementation()</a> . . . . .	591
10.139.2.2	<a href="#">set_channel()</a> . . . . .	591
10.139.2.3	<a href="#">set_gnss_synchro()</a> . . . . .	591
10.139.2.4	<a href="#">stop_tracking()</a> . . . . .	592

10.140. <a href="#">GlonassL1CaPcpsAcquisition Class Reference</a>	592
10.140.1. <a href="#">Detailed Description</a>	593
10.140.2. <a href="#">Member Function Documentation</a>	593
10.140.2.1. <a href="#">implementation()</a>	593
10.140.2.2. <a href="#">init()</a>	593
10.140.2.3. <a href="#">mag()</a>	594
10.140.2.4. <a href="#">reset()</a>	594
10.140.2.5. <a href="#">set_channel()</a>	594
10.140.2.6. <a href="#">set_channel_fsm()</a>	594
10.140.2.7. <a href="#">set_doppler_max()</a>	595
10.140.2.8. <a href="#">set_doppler_step()</a>	595
10.140.2.9. <a href="#">set_gnss_synchro()</a>	595
10.140.2.10. <a href="#">set_local_code()</a>	595
10.140.2.11. <a href="#">set_state()</a>	595
10.140.2.12. <a href="#">set_threshold()</a>	596
10.140.2.13. <a href="#">stop_acquisition()</a>	596
10.141. <a href="#">GlonassL1CaTelemetryDecoder Class Reference</a>	596
10.141.1. <a href="#">Detailed Description</a>	597
10.141.2. <a href="#">Member Function Documentation</a>	597
10.141.2.1. <a href="#">implementation()</a>	597
10.142. <a href="#">GlonassL2CaDIIPICAidTracking Class Reference</a>	597
10.142.1. <a href="#">Detailed Description</a>	598
10.142.2. <a href="#">Member Function Documentation</a>	598
10.142.2.1. <a href="#">implementation()</a>	598
10.142.2.2. <a href="#">set_channel()</a>	598
10.142.2.3. <a href="#">set_gnss_synchro()</a>	599
10.142.2.4. <a href="#">stop_tracking()</a>	599
10.143. <a href="#">GlonassL2CaDIIPIITracking Class Reference</a>	599
10.143.1. <a href="#">Detailed Description</a>	600
10.143.2. <a href="#">Member Function Documentation</a>	600

10.143.2.1implementation()	600
10.143.2.2set_channel()	600
10.143.2.3set_gnss_synchro()	600
10.143.2.4stop_tracking()	601
10.144.1GlonassL2CaPcpsAcquisition Class Reference	601
10.144.1.1Detailed Description	602
10.144.2Member Function Documentation	602
10.144.2.1implementation()	602
10.144.2.2nit()	602
10.144.2.3mag()	603
10.144.2.4reset()	603
10.144.2.5set_channel()	603
10.144.2.6set_channel_fsm()	603
10.144.2.7set_doppler_max()	604
10.144.2.8set_doppler_step()	604
10.144.2.9set_gnss_synchro()	604
10.144.2.10set_local_code()	604
10.144.2.11set_state()	604
10.144.2.12set_threshold()	605
10.144.2.13stop_acquisition()	605
10.145.1GlonassL2CaTelemetryDecoder Class Reference	605
10.145.1.1Detailed Description	606
10.145.2Member Function Documentation	606
10.145.2.1implementation()	606
10.146.1Gn3sSignalSource Class Reference	606
10.146.1.1Detailed Description	607
10.146.2Member Function Documentation	607
10.146.2.1implementation()	607
10.147.1GnMaxSignalSource Class Reference	607
10.147.1.1Detailed Description	608

10.147.2	Member Function Documentation	608
10.147.2.1	implementation()	608
10.148	Gnss_circular_deque< T > Class Template Reference	608
10.148.1	Detailed Description	609
10.149	Gnss_Satellite Class Reference	609
10.149.1	Detailed Description	610
10.149.2	Constructor & Destructor Documentation	610
10.149.2.1	Gnss_Satellite() [1/4]	610
10.149.2.2	Gnss_Satellite() [2/4]	610
10.149.2.3	~Gnss_Satellite()	610
10.149.2.4	Gnss_Satellite() [3/4]	611
10.149.2.5	Gnss_Satellite() [4/4]	611
10.149.3	Member Function Documentation	611
10.149.3.1	get_block()	611
10.149.3.2	get_PRN()	611
10.149.3.3	get_rf_link()	611
10.149.3.4	get_system()	612
10.149.3.5	get_system_short()	612
10.149.3.6	operator=() [1/2]	612
10.149.3.7	operator=() [2/2]	612
10.149.3.8	update_PRN()	612
10.149.3.9	what_block()	612
10.149.4	Friends And Related Function Documentation	613
10.149.4.1	operator<<	613
10.149.4.2	operator==	613
10.150	Gnss_sdr_fpga_sample_counter Class Reference	613
10.150.1	Detailed Description	614
10.151	Gnss_sdr_sample_counter Class Reference	614
10.151.1	Detailed Description	614
10.152	Gnss_Sdr_Supl_Client Class Reference	614

10.152.1 Detailed Description . . . . .	616
10.152.2 Member Function Documentation . . . . .	616
10.152.2.1 load_cnav_ephemeris_xml() . . . . .	617
10.152.2.2 load_cnav_utc_xml() . . . . .	617
10.152.2.3 load_ephemeris_xml() . . . . .	617
10.152.2.4 load_gal_almanac_xml() . . . . .	617
10.152.2.5 load_gal_ephemeris_xml() . . . . .	617
10.152.2.6 load_gal_iono_xml() . . . . .	617
10.152.2.7 load_gal_utc_xml() . . . . .	618
10.152.2.8 load_glo_utc_xml() . . . . .	618
10.152.2.9 load_gnav_ephemeris_xml() . . . . .	618
10.152.2.10 load_gps_almanac_xml() . . . . .	618
10.152.2.11 load_iono_xml() . . . . .	618
10.152.2.12 load_ref_location_xml() . . . . .	618
10.152.2.13 load_ref_time_xml() . . . . .	619
10.152.2.14 load_utc_xml() . . . . .	619
10.152.2.15 save_cnav_ephemeris_map_xml() . . . . .	619
10.152.2.16 save_cnav_utc_xml() . . . . .	619
10.152.2.17 save_ephemeris_map_xml() . . . . .	619
10.152.2.18 save_gal_almanac_xml() . . . . .	620
10.152.2.19 save_gal_ephemeris_map_xml() . . . . .	620
10.152.2.20 save_gal_iono_xml() . . . . .	620
10.152.2.21 save_gal_utc_xml() . . . . .	620
10.152.2.22 save_glo_utc_xml() . . . . .	620
10.152.2.23 save_gnav_ephemeris_map_xml() . . . . .	621
10.152.2.24 save_gps_almanac_xml() . . . . .	621
10.152.2.25 save_iono_xml() . . . . .	621
10.152.2.26 save_ref_location_xml() . . . . .	621
10.152.2.27 save_ref_time_xml() . . . . .	621
10.152.2.28 save_utc_xml() . . . . .	622

10.153	Gnss_sdr_time_counter Class Reference . . . . .	622
10.153.1	Detailed Description . . . . .	622
10.154	Gnss_Sdr_Valve Class Reference . . . . .	623
10.154.1	Detailed Description . . . . .	623
10.155	Gnss_Signal Class Reference . . . . .	623
10.155.1	Detailed Description . . . . .	624
10.155.2	Member Function Documentation . . . . .	624
10.155.2.1	get_satellite() . . . . .	624
10.155.2.2	get_signal_str() . . . . .	624
10.155.3	Friends And Related Function Documentation . . . . .	624
10.155.3.1	operator<< . . . . .	625
10.155.3.2	operator== . . . . .	625
10.156	Gnss_Synchro Class Reference . . . . .	625
10.156.1	Detailed Description . . . . .	627
10.156.2	Constructor & Destructor Documentation . . . . .	627
10.156.2.1	Gnss_Synchro() [1/3] . . . . .	627
10.156.2.2	~Gnss_Synchro() . . . . .	627
10.156.2.3	Gnss_Synchro() [2/3] . . . . .	627
10.156.2.4	Gnss_Synchro() [3/3] . . . . .	627
10.156.3	Member Function Documentation . . . . .	628
10.156.3.1	operator=() [1/2] . . . . .	628
10.156.3.2	operator=() [2/2] . . . . .	628
10.156.3.3	serialize() . . . . .	628
10.156.4	Member Data Documentation . . . . .	629
10.156.4.1	Acq_delay_samples . . . . .	629
10.156.4.2	Acq_doppler_hz . . . . .	629
10.156.4.3	Acq_doppler_step . . . . .	629
10.156.4.4	Acq_samplestamp_samples . . . . .	629
10.156.4.5	Carrier_Doppler_hz . . . . .	630
10.156.4.6	Carrier_phase_rads . . . . .	630

10.156.4.7Channel_ID . . . . .	630
10.156.4.8CNO_dB_hz . . . . .	630
10.156.4.9Code_phase_samples . . . . .	631
10.156.4.10Correlation_length_ms . . . . .	631
10.156.4.11Flag_valid_acquisition . . . . .	631
10.156.4.12Flag_valid_pseudorange . . . . .	631
10.156.4.13Flag_valid_symbol_output . . . . .	632
10.156.4.14Flag_valid_word . . . . .	632
10.156.4.15 . . . . .	632
10.156.4.16Interp_TOW_ms . . . . .	632
10.156.4.17PRN . . . . .	633
10.156.4.18Prompt_I . . . . .	633
10.156.4.19Prompt_Q . . . . .	633
10.156.4.20Pseudorange_m . . . . .	633
10.156.4.21RX_time . . . . .	634
10.156.4.22Signal . . . . .	634
10.156.4.23System . . . . .	634
10.156.4.24TOW_at_current_symbol_ms . . . . .	634
10.156.4.25Tracking_sample_counter . . . . .	635
10.157gnss_synchro_monitor Class Reference . . . . .	635
10.157.1Detailed Description . . . . .	635
10.157.2Constructor & Destructor Documentation . . . . .	636
10.157.2.1~gnss_synchro_monitor() . . . . .	636
10.158gnss_Synchro_Udp_Sink Class Reference . . . . .	636
10.158.1Detailed Description . . . . .	636
10.159GNSSBlockFactory Class Reference . . . . .	636
10.159.1Detailed Description . . . . .	637
10.159.2Member Function Documentation . . . . .	637
10.159.2.1GetBlock() . . . . .	637
10.160GNSSBlockInterface Class Reference . . . . .	638

10.160.1Detailed Description . . . . .	639
10.160.2Member Function Documentation . . . . .	639
10.160.2.1start() . . . . .	639
10.160GNSSFlowgraph Class Reference . . . . .	639
10.161.1Detailed Description . . . . .	640
10.161.2Constructor & Destructor Documentation . . . . .	640
10.161.2.1GNSSFlowgraph() . . . . .	640
10.161.2.2~GNSSFlowgraph() . . . . .	641
10.161.3Member Function Documentation . . . . .	641
10.161.3.1acquisition_manager() . . . . .	641
10.161.3.2apply_action() . . . . .	641
10.161.3.3connect() . . . . .	641
10.161.3.4disconnect() . . . . .	642
10.161.3.5get_pvt() . . . . .	642
10.161.3.6priorize_satellites() . . . . .	642
10.161.3.7send_telemetry_msg() . . . . .	642
10.161.3.8set_configuration() . . . . .	642
10.161.3.9start() . . . . .	642
10.161.3.10stop() . . . . .	643
10.161.3.11wait() . . . . .	643
10.162Gnuplot Class Reference . . . . .	643
10.162.1Detailed Description . . . . .	645
10.162.2Constructor & Destructor Documentation . . . . .	645
10.162.2.1Gnuplot() . . . . .	645
10.162.3Member Function Documentation . . . . .	645
10.162.3.1operator<<() . . . . .	645
10.162.3.2replot() . . . . .	645
10.162.3.3unset_title() . . . . .	646
10.163GnuplotException Class Reference . . . . .	646
10.163.1Detailed Description . . . . .	646

10.164	Gps_Acq_Assist Class Reference . . . . .	646
10.164.1	Detailed Description . . . . .	647
10.164.2	Constructor & Destructor Documentation . . . . .	647
10.164.2.1	Gps_Acq_Assist() . . . . .	647
10.164.3	Member Data Documentation . . . . .	648
10.164.3.1	Azimuth . . . . .	648
10.164.3.2	Code_Phase . . . . .	648
10.164.3.3	Code_Phase_int . . . . .	648
10.164.3.4	Code_Phase_window . . . . .	648
10.164.3.5	d_Doppler0 . . . . .	648
10.164.3.6	d_Doppler1 . . . . .	649
10.164.3.7	d_TOW . . . . .	649
10.164.3.8	dopplerUncertainty . . . . .	649
10.164.3.9	Elevation . . . . .	649
10.164.3.10	GPS_Bit_Number . . . . .	649
10.164.3.11	i_satellite_PRN . . . . .	650
10.165	Gps_Almanac Class Reference . . . . .	650
10.165.1	Detailed Description . . . . .	651
10.165.2	Constructor & Destructor Documentation . . . . .	651
10.165.2.1	Gps_Almanac() . . . . .	651
10.165.3	Member Data Documentation . . . . .	651
10.165.3.1	d_A_f0 . . . . .	651
10.165.3.2	d_A_f1 . . . . .	651
10.165.3.3	d_Delta_i . . . . .	651
10.165.3.4	d_e_eccentricity . . . . .	652
10.165.3.5	d_M_0 . . . . .	652
10.165.3.6	d_OMEGA . . . . .	652
10.165.3.7	d_OMEGA0 . . . . .	652
10.165.3.8	d_OMEGA_DOT . . . . .	652
10.165.3.9	d_sqrt_A . . . . .	653

10.165.3.10AS_status . . . . .	653
10.165.3.11satellite_PRN . . . . .	653
10.165.3.12SV_health . . . . .	653
10.165.3.13Toa . . . . .	653
10.165.3.14WNa . . . . .	654
10.166Gps_CNAV_Ephemeris Class Reference . . . . .	654
10.166.1Detailed Description . . . . .	656
10.166.2Constructor & Destructor Documentation . . . . .	656
10.166.2.1Gps_CNAV_Ephemeris() . . . . .	656
10.166.3Member Function Documentation . . . . .	657
10.166.3.1satellitePosition() . . . . .	657
10.166.3.2serialize() . . . . .	657
10.166.3.3sv_clock_drift() . . . . .	658
10.166.3.4sv_clock_relativistic_term() . . . . .	658
10.166.4Member Data Documentation . . . . .	659
10.166.4.1b_alert_flag . . . . .	659
10.166.4.2b_antispoofing_flag . . . . .	659
10.166.4.3b_integrity_status_flag . . . . .	659
10.166.4.4d_A_DOT . . . . .	660
10.166.4.5d_A_f0 . . . . .	660
10.166.4.6d_A_f1 . . . . .	660
10.166.4.7d_A_f2 . . . . .	660
10.166.4.8d_Cic . . . . .	661
10.166.4.9d_Cis . . . . .	661
10.166.4.10d_Crc . . . . .	661
10.166.4.11d_Crs . . . . .	661
10.166.4.12d_Cuc . . . . .	662
10.166.4.13d_Cus . . . . .	662
10.166.4.14d_DELTA_A . . . . .	662
10.166.4.15d_DELTA_DOT_N . . . . .	662

10.166.4.16_Delta_n	662
10.166.4.17_DELTA_OMEGA_DOT	663
10.166.4.18_dtr	663
10.166.4.19_e_eccentricity	663
10.166.4.20_i_0	663
10.166.4.21_IDOT	664
10.166.4.22_M_0	664
10.166.4.23_OMEGA	664
10.166.4.24_OMEGA0	664
10.166.4.25_satClkDrift	665
10.166.4.26_satpos_X	665
10.166.4.27_satpos_Y	665
10.166.4.28_satpos_Z	665
10.166.4.29_satvel_X	665
10.166.4.30_satvel_Y	666
10.166.4.31_satvel_Z	666
10.166.4.32_TGD	666
10.166.4.33_Toc	666
10.166.4.34_Toe1	666
10.166.4.35_Toe2	667
10.166.4.36_Top	667
10.166.4.37_TOW	667
10.166.4.38_URA0	667
10.166.4.39_URA1	667
10.166.4.40_URA2	668
10.166.4.41_GPS_week	668
10.166.4.42_signal_health	668
10.166.4.43_URA	668
10.166.4.44_Gps_CNAV_Iono Class Reference	668
10.167.1 Detailed Description	669

10.167.2	Constructor & Destructor Documentation . . . . .	669
10.167.2.1	Gps_CNAV_Iono() . . . . .	669
10.167.3	Member Function Documentation . . . . .	670
10.167.3.1	serialize() . . . . .	670
10.167.4	Member Data Documentation . . . . .	670
10.167.4.1	d_alpha0 . . . . .	670
10.167.4.2	d_alpha1 . . . . .	670
10.167.4.3	d_alpha2 . . . . .	671
10.167.4.4	d_alpha3 . . . . .	671
10.167.4.5	d_beta0 . . . . .	671
10.167.4.6	d_beta1 . . . . .	671
10.167.4.7	d_beta2 . . . . .	672
10.167.4.8	d_beta3 . . . . .	672
10.167.4.9	valid . . . . .	672
10.168	Gps_CNAV_Navigation_Message Class Reference . . . . .	672
10.168.1	Detailed Description . . . . .	673
10.168.2	Constructor & Destructor Documentation . . . . .	673
10.168.2.1	Gps_CNAV_Navigation_Message() . . . . .	673
10.168.3	Member Function Documentation . . . . .	673
10.168.3.1	get_ephemeris() . . . . .	673
10.168.3.2	get_iono() . . . . .	674
10.168.3.3	get_utc_model() . . . . .	674
10.168.3.4	have_new_ephemeris() . . . . .	674
10.168.3.5	have_new_iono() . . . . .	674
10.168.3.6	have_new_utc_model() . . . . .	674
10.169	Gps_CNAV_Utc_Model Class Reference . . . . .	674
10.169.1	Detailed Description . . . . .	675
10.169.2	Constructor & Destructor Documentation . . . . .	675
10.169.2.1	Gps_CNAV_Utc_Model() . . . . .	675
10.169.3	Member Data Documentation . . . . .	676

10.169.3.1d_A0 . . . . .	676
10.169.3.2d_A1 . . . . .	676
10.169.3.3d_A2 . . . . .	676
10.169.3.4d_DeltaT_LS . . . . .	676
10.169.3.5d_DeltaT_LSF . . . . .	676
10.169.3.6d_t_OT . . . . .	677
10.169.3.7_DN . . . . .	677
10.169.3.8_WN_LSF . . . . .	677
10.169.3.9_WN_T . . . . .	677
10.170.Gps_Ephemeris Class Reference . . . . .	677
10.170.1Detailed Description . . . . .	680
10.170.2Constructor & Destructor Documentation . . . . .	680
10.170.2.1Gps_Ephemeris() . . . . .	680
10.170.3Member Function Documentation . . . . .	680
10.170.3.1satellitePosition() . . . . .	680
10.170.3.2serialize() . . . . .	681
10.170.3.3sv_clock_drift() . . . . .	682
10.170.3.4sv_clock_relativistic_term() . . . . .	682
10.170.4Member Data Documentation . . . . .	682
10.170.4.1b_alert_flag . . . . .	683
10.170.4.2b_antispoofing_flag . . . . .	683
10.170.4.3b_fit_interval_flag . . . . .	683
10.170.4.4b_integrity_status_flag . . . . .	683
10.170.4.5b_L2_P_data_flag . . . . .	684
10.170.4.6d_A_f0 . . . . .	684
10.170.4.7d_A_f1 . . . . .	684
10.170.4.8d_A_f2 . . . . .	684
10.170.4.9d_Cic . . . . .	685
10.170.4.10_Cis . . . . .	685
10.170.4.1d_Crc . . . . .	685

10.170.4.12_Crs . . . . .	685
10.170.4.13_Cuc . . . . .	686
10.170.4.14_Cus . . . . .	686
10.170.4.15_Delta_n . . . . .	686
10.170.4.16_dtr . . . . .	686
10.170.4.17_e_eccentricity . . . . .	687
10.170.4.18_i_0 . . . . .	687
10.170.4.19_IDOT . . . . .	687
10.170.4.20_IODC . . . . .	687
10.170.4.21_IODE_SF2 . . . . .	688
10.170.4.22_IODE_SF3 . . . . .	688
10.170.4.23_M_0 . . . . .	688
10.170.4.24_OMEGA . . . . .	688
10.170.4.25_OMEGA0 . . . . .	689
10.170.4.26_OMEGA_DOT . . . . .	689
10.170.4.27_satClkDrift . . . . .	689
10.170.4.28_satpos_X . . . . .	689
10.170.4.29_satpos_Y . . . . .	689
10.170.4.30_satpos_Z . . . . .	690
10.170.4.31_satvel_X . . . . .	690
10.170.4.32_satvel_Y . . . . .	690
10.170.4.33_satvel_Z . . . . .	690
10.170.4.34_sqrt_A . . . . .	690
10.170.4.35_TGD . . . . .	691
10.170.4.36_Toc . . . . .	691
10.170.4.37_Toe . . . . .	691
10.170.4.38_TOW . . . . .	691
10.170.4.39AODO . . . . .	692
10.170.4.40code_on_L2 . . . . .	692
10.170.4.41GPS_week . . . . .	692

10.170.4.42SV_accuracy . . . . .	692
10.170.4.43SatelliteBlock . . . . .	693
10.171Gps_Iono Class Reference . . . . .	693
10.171.1Detailed Description . . . . .	694
10.171.2Constructor & Destructor Documentation . . . . .	694
10.171.2.1Gps_Iono() . . . . .	694
10.171.3Member Function Documentation . . . . .	694
10.171.3.1serialize() . . . . .	694
10.171.4Member Data Documentation . . . . .	694
10.171.4.1d_alpha0 . . . . .	694
10.171.4.2d_alpha1 . . . . .	695
10.171.4.3d_alpha2 . . . . .	695
10.171.4.4d_alpha3 . . . . .	695
10.171.4.5d_beta0 . . . . .	695
10.171.4.6d_beta1 . . . . .	696
10.171.4.7d_beta2 . . . . .	696
10.171.4.8d_beta3 . . . . .	696
10.171.4.9valid . . . . .	696
10.172Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc Class Reference . . . . .	697
10.172.1Detailed Description . . . . .	697
10.173Gps_L1_Ca_Kf_Tracking_cc Class Reference . . . . .	697
10.173.1Detailed Description . . . . .	698
10.174Gps_L1_Ca_Tcp_Connector_Tracking_cc Class Reference . . . . .	698
10.174.1Detailed Description . . . . .	699
10.175Gps_I1_ca_telemetry_decoder_gs Class Reference . . . . .	699
10.175.1Detailed Description . . . . .	700
10.175.2Member Function Documentation . . . . .	700
10.175.2.1general_work() . . . . .	700
10.175.2.2set_channel() . . . . .	700
10.175.2.3set_satellite() . . . . .	700

10.176. <a href="#">Gps_I2c_telemetry_decoder_gs Class Reference</a> . . . . .	701
10.176.1. <a href="#">Detailed Description</a> . . . . .	701
10.176.2. <a href="#">Member Function Documentation</a> . . . . .	701
10.176.2.1. <a href="#">general_work()</a> . . . . .	701
10.176.2.2. <a href="#">set_channel()</a> . . . . .	702
10.176.2.3. <a href="#">set_satellite()</a> . . . . .	702
10.177. <a href="#">Gps_I5_telemetry_decoder_gs Class Reference</a> . . . . .	702
10.177.1. <a href="#">Detailed Description</a> . . . . .	703
10.177.2. <a href="#">Member Function Documentation</a> . . . . .	703
10.177.2.1. <a href="#">set_channel()</a> . . . . .	703
10.177.2.2. <a href="#">set_satellite()</a> . . . . .	703
10.178. <a href="#">Gps_Navigation_Message Class Reference</a> . . . . .	703
10.178.1. <a href="#">Detailed Description</a> . . . . .	704
10.178.2. <a href="#">Constructor &amp; Destructor Documentation</a> . . . . .	704
10.178.2.1. <a href="#">Gps_Navigation_Message()</a> . . . . .	704
10.178.3. <a href="#">Member Function Documentation</a> . . . . .	705
10.178.3.1. <a href="#">get_ephemeris()</a> . . . . .	705
10.178.3.2. <a href="#">get_flag_iono_valid()</a> . . . . .	705
10.178.3.3. <a href="#">get_flag_utc_model_valid()</a> . . . . .	705
10.178.3.4. <a href="#">get_GPS_week()</a> . . . . .	705
10.178.3.5. <a href="#">get_iono()</a> . . . . .	705
10.178.3.6. <a href="#">get_satellite_PRN()</a> . . . . .	706
10.178.3.7. <a href="#">get_TOW()</a> . . . . .	706
10.178.3.8. <a href="#">get_utc_model()</a> . . . . .	706
10.178.3.9. <a href="#">set_channel()</a> . . . . .	706
10.178.3.10. <a href="#">set_satellite_PRN()</a> . . . . .	706
10.178.3.11. <a href="#">subframe_decoder()</a> . . . . .	707
10.178.3.12. <a href="#">utc_time()</a> . . . . .	707
10.179. <a href="#">Gps_Utc_Model Class Reference</a> . . . . .	707
10.179.1. <a href="#">Detailed Description</a> . . . . .	708

10.179.2	Constructor & Destructor Documentation . . . . .	708
10.179.2.1	Gps_Utc_Model() . . . . .	708
10.179.3	Member Data Documentation . . . . .	708
10.179.3.1	d_A0 . . . . .	708
10.179.3.2	d_A1 . . . . .	708
10.179.3.3	d_A2 . . . . .	708
10.179.3.4	d_DeltaT_LS . . . . .	709
10.179.3.5	d_DeltaT_LSF . . . . .	709
10.179.3.6	d_t_OT . . . . .	709
10.179.3.7	d_DN . . . . .	709
10.179.3.8	d_WN_LSF . . . . .	709
10.179.3.9	d_WN_T . . . . .	710
10.180	GpsL1CaDIIPIITracking Class Reference . . . . .	710
10.180.1	Detailed Description . . . . .	711
10.180.2	Member Function Documentation . . . . .	711
10.180.2.1	implementation() . . . . .	711
10.180.2.2	set_channel() . . . . .	711
10.180.2.3	set_gnss_synchro() . . . . .	711
10.180.2.4	stop_tracking() . . . . .	712
10.181	GpsL1CaDIIPIITrackingFpga Class Reference . . . . .	712
10.181.1	Detailed Description . . . . .	713
10.181.2	Constructor & Destructor Documentation . . . . .	713
10.181.2.1	GpsL1CaDIIPIITrackingFpga() . . . . .	713
10.181.2.2	~GpsL1CaDIIPIITrackingFpga() . . . . .	713
10.181.3	Member Function Documentation . . . . .	713
10.181.3.1	connect() . . . . .	714
10.181.3.2	disconnect() . . . . .	714
10.181.3.3	get_left_block() . . . . .	714
10.181.3.4	get_right_block() . . . . .	714
10.181.3.5	implementation() . . . . .	714

10.181.3.6	item_size()	715
10.181.3.7	role()	715
10.181.3.8	set_channel()	715
10.181.3.9	set_gnss_synchro()	715
10.181.3.10	start_tracking()	716
10.181.3.11	stop_tracking()	716
10.182	GpsL1CaDlPILTrackingGPU Class Reference	716
10.182.1	Detailed Description	717
10.182.2	Member Function Documentation	717
10.182.2.1	implementation()	717
10.182.2.2	set_channel()	717
10.182.2.3	set_gnss_synchro()	717
10.182.2.4	stop_tracking()	718
10.183	GpsL1CaKfTracking Class Reference	718
10.183.1	Detailed Description	719
10.183.2	Member Function Documentation	719
10.183.2.1	implementation()	719
10.183.2.2	set_channel()	719
10.183.2.3	set_gnss_synchro()	719
10.183.2.4	stop_tracking()	720
10.184	GpsL1CaPcpsAcquisition Class Reference	720
10.184.1	Detailed Description	721
10.184.2	Member Function Documentation	721
10.184.2.1	implementation()	721
10.184.2.2	init()	721
10.184.2.3	mag()	722
10.184.2.4	reset()	722
10.184.2.5	set_channel()	722
10.184.2.6	set_channel_fsm()	722
10.184.2.7	set_doppler_center()	722

10.184.2.8	<a href="#">set_doppler_max()</a> . . . . .	723
10.184.2.9	<a href="#">set_doppler_step()</a> . . . . .	723
10.184.2.10	<a href="#">set_gnss_synchro()</a> . . . . .	723
10.184.2.11	<a href="#">set_local_code()</a> . . . . .	723
10.184.2.12	<a href="#">set_resampler_latency()</a> . . . . .	723
10.184.2.13	<a href="#">set_state()</a> . . . . .	724
10.184.2.14	<a href="#">set_threshold()</a> . . . . .	724
10.184.2.15	<a href="#">stop_acquisition()</a> . . . . .	724
10.185	<a href="#">GpsL1CaPcpsAcquisitionFineDoppler Class Reference</a> . . . . .	724
10.185.1	<a href="#">Detailed Description</a> . . . . .	725
10.185.2	<a href="#">Member Function Documentation</a> . . . . .	725
10.185.2.1	<a href="#">implementation()</a> . . . . .	726
10.185.2.2	<a href="#">nit()</a> . . . . .	726
10.185.2.3	<a href="#">mag()</a> . . . . .	726
10.185.2.4	<a href="#">reset()</a> . . . . .	726
10.185.2.5	<a href="#">set_channel()</a> . . . . .	726
10.185.2.6	<a href="#">set_channel_fsm()</a> . . . . .	727
10.185.2.7	<a href="#">set_doppler_max()</a> . . . . .	727
10.185.2.8	<a href="#">set_doppler_step()</a> . . . . .	727
10.185.2.9	<a href="#">set_gnss_synchro()</a> . . . . .	727
10.185.2.10	<a href="#">set_state()</a> . . . . .	728
10.185.2.11	<a href="#">set_threshold()</a> . . . . .	728
10.185.2.12	<a href="#">stop_acquisition()</a> . . . . .	728
10.186	<a href="#">GpsL1CaPcpsAcquisitionFpga Class Reference</a> . . . . .	728
10.186.1	<a href="#">Detailed Description</a> . . . . .	730
10.186.2	<a href="#">Constructor &amp; Destructor Documentation</a> . . . . .	730
10.186.2.1	<a href="#">GpsL1CaPcpsAcquisitionFpga()</a> . . . . .	730
10.186.2.2	<a href="#">~GpsL1CaPcpsAcquisitionFpga()</a> . . . . .	730
10.186.3	<a href="#">Member Function Documentation</a> . . . . .	730
10.186.3.1	<a href="#">connect()</a> . . . . .	730

10.186.3.2	<a href="#">disconnect()</a>	731
10.186.3.3	<a href="#">get_left_block()</a>	731
10.186.3.4	<a href="#">get_right_block()</a>	731
10.186.3.5	<a href="#">implementation()</a>	731
10.186.3.6	<a href="#">init()</a>	731
10.186.3.7	<a href="#">item_size()</a>	732
10.186.3.8	<a href="#">mag()</a>	732
10.186.3.9	<a href="#">reset()</a>	732
10.186.3.10	<a href="#">role()</a>	732
10.186.3.11	<a href="#">set_channel()</a>	732
10.186.3.12	<a href="#">set_channel_fsm()</a>	733
10.186.3.13	<a href="#">set_doppler_center()</a>	733
10.186.3.14	<a href="#">set_doppler_max()</a>	733
10.186.3.15	<a href="#">set_doppler_step()</a>	733
10.186.3.16	<a href="#">set_gnss_synchro()</a>	733
10.186.3.17	<a href="#">set_local_code()</a>	734
10.186.3.18	<a href="#">set_resampler_latency()</a>	734
10.186.3.19	<a href="#">set_state()</a>	734
10.186.3.20	<a href="#">set_threshold()</a>	734
10.186.3.21	<a href="#">stop_acquisition()</a>	734
10.187	<a href="#">CpsL1CaPcpsAssistedAcquisition Class Reference</a>	735
10.187.1	<a href="#">Detailed Description</a>	736
10.187.2	<a href="#">Member Function Documentation</a>	736
10.187.2.1	<a href="#">implementation()</a>	736
10.187.2.2	<a href="#">init()</a>	736
10.187.2.3	<a href="#">mag()</a>	736
10.187.2.4	<a href="#">reset()</a>	736
10.187.2.5	<a href="#">set_channel()</a>	737
10.187.2.6	<a href="#">set_channel_fsm()</a>	737
10.187.2.7	<a href="#">set_doppler_max()</a>	737

10.187.2.8	<a href="#">set_doppler_step()</a> . . . . .	737
10.187.2.9	<a href="#">set_gnss_synchro()</a> . . . . .	738
10.187.2.10	<a href="#">set_threshold()</a> . . . . .	738
10.187.2.11	<a href="#">stop_acquisition()</a> . . . . .	738
10.188	<a href="#">GpsL1CaPcpsOpenCIAcquisition Class Reference</a> . . . . .	738
10.188.1	<a href="#">Detailed Description</a> . . . . .	739
10.188.2	<a href="#">Member Function Documentation</a> . . . . .	739
10.188.2.1	<a href="#">implementation()</a> . . . . .	740
10.188.2.2	<a href="#">nit()</a> . . . . .	740
10.188.2.3	<a href="#">mag()</a> . . . . .	740
10.188.2.4	<a href="#">reset()</a> . . . . .	740
10.188.2.5	<a href="#">set_channel()</a> . . . . .	740
10.188.2.6	<a href="#">set_channel_fsm()</a> . . . . .	741
10.188.2.7	<a href="#">set_doppler_max()</a> . . . . .	741
10.188.2.8	<a href="#">set_doppler_step()</a> . . . . .	741
10.188.2.9	<a href="#">set_gnss_synchro()</a> . . . . .	741
10.188.2.10	<a href="#">set_local_code()</a> . . . . .	742
10.188.2.11	<a href="#">set_threshold()</a> . . . . .	742
10.188.2.12	<a href="#">stop_acquisition()</a> . . . . .	742
10.189	<a href="#">GpsL1CaPcpsQuickSyncAcquisition Class Reference</a> . . . . .	742
10.189.1	<a href="#">Detailed Description</a> . . . . .	743
10.189.2	<a href="#">Member Function Documentation</a> . . . . .	743
10.189.2.1	<a href="#">implementation()</a> . . . . .	744
10.189.2.2	<a href="#">nit()</a> . . . . .	744
10.189.2.3	<a href="#">mag()</a> . . . . .	744
10.189.2.4	<a href="#">reset()</a> . . . . .	744
10.189.2.5	<a href="#">set_channel()</a> . . . . .	744
10.189.2.6	<a href="#">set_channel_fsm()</a> . . . . .	745
10.189.2.7	<a href="#">set_doppler_max()</a> . . . . .	745
10.189.2.8	<a href="#">set_doppler_step()</a> . . . . .	745

10.189.2.9set_gnss_synchro()	745
10.189.2.10set_local_code()	746
10.189.2.11set_state()	746
10.189.2.12set_threshold()	746
10.189.2.13stop_acquisition()	746
10.190GpsL1CaPcpsTongAcquisition Class Reference	747
10.190.1Detailed Description	748
10.190.2Member Function Documentation	748
10.190.2.1implementation()	748
10.190.2.2nit()	748
10.190.2.3mag()	748
10.190.2.4reset()	748
10.190.2.5set_channel()	749
10.190.2.6set_channel_fsm()	749
10.190.2.7set_doppler_max()	749
10.190.2.8set_doppler_step()	749
10.190.2.9set_gnss_synchro()	750
10.190.2.10set_local_code()	750
10.190.2.11set_state()	750
10.190.2.12set_threshold()	750
10.190.2.13stop_acquisition()	750
10.191GpsL1CaTcpConnectorTracking Class Reference	751
10.191.1Detailed Description	751
10.191.2Member Function Documentation	751
10.191.2.1implementation()	752
10.191.2.2set_channel()	752
10.191.2.3set_gnss_synchro()	752
10.191.2.4stop_tracking()	752
10.192GpsL1CaTelemetryDecoder Class Reference	753
10.192.1Detailed Description	753

10.192.2	Member Function Documentation	753
10.192.2.1	implementation()	753
10.193	GpsL2CTelemetryDecoder Class Reference	754
10.193.1	Detailed Description	754
10.193.2	Member Function Documentation	754
10.193.2.1	implementation()	754
10.194	GpsL2MDIIPITracking Class Reference	755
10.194.1	Detailed Description	755
10.194.2	Member Function Documentation	755
10.194.2.1	implementation()	756
10.194.2.2	set_channel()	756
10.194.2.3	set_gnss_synchro()	756
10.194.2.4	stop_tracking()	756
10.195	GpsL2MDIIPITrackingFpga Class Reference	757
10.195.1	Detailed Description	757
10.195.2	Member Function Documentation	757
10.195.2.1	implementation()	758
10.195.2.2	set_channel()	758
10.195.2.3	set_gnss_synchro()	758
10.195.2.4	stop_tracking()	758
10.196	GpsL2MPcpsAcquisition Class Reference	759
10.196.1	Detailed Description	760
10.196.2	Member Function Documentation	760
10.196.2.1	implementation()	760
10.196.2.2	init()	760
10.196.2.3	mag()	760
10.196.2.4	reset()	761
10.196.2.5	set_channel()	761
10.196.2.6	set_channel_fsm()	761
10.196.2.7	set_doppler_center()	761

10.196.2.8	set_doppler_max()	761
10.196.2.9	set_doppler_step()	762
10.196.2.10	set_gnss_synchro()	762
10.196.2.11	set_local_code()	762
10.196.2.12	set_resampler_latency()	762
10.196.2.13	set_state()	762
10.196.2.14	set_threshold()	763
10.196.2.15	stop_acquisition()	763
10.197	GpsL2MPcpsAcquisitionFpga Class Reference	763
10.197.1	Detailed Description	764
10.197.2	Member Function Documentation	764
10.197.2.1	implementation()	765
10.197.2.2	nit()	765
10.197.2.3	mag()	765
10.197.2.4	reset()	765
10.197.2.5	set_channel()	765
10.197.2.6	set_channel_fsm()	766
10.197.2.7	set_doppler_max()	766
10.197.2.8	set_doppler_step()	766
10.197.2.9	set_gnss_synchro()	766
10.197.2.10	set_local_code()	767
10.197.2.11	set_state()	767
10.197.2.12	set_threshold()	767
10.197.2.13	stop_acquisition()	767
10.198	GpsL5DIIPITracking Class Reference	768
10.198.1	Detailed Description	768
10.198.2	Member Function Documentation	768
10.198.2.1	implementation()	769
10.198.2.2	set_channel()	769
10.198.2.3	set_gnss_synchro()	769

10.198.2.4	stop_tracking()	769
10.199	GpsL5DIIPIITrackingFpga Class Reference	770
10.199.1	Detailed Description	770
10.199.2	Constructor & Destructor Documentation	771
10.199.2.1	GpsL5DIIPIITrackingFpga()	771
10.199.2.2	~GpsL5DIIPIITrackingFpga()	771
10.199.3	Member Function Documentation	771
10.199.3.1	connect()	771
10.199.3.2	disconnect()	771
10.199.3.3	get_left_block()	772
10.199.3.4	get_right_block()	772
10.199.3.5	implementation()	772
10.199.3.6	item_size()	772
10.199.3.7	role()	772
10.199.3.8	set_channel()	773
10.199.3.9	set_gnss_synchro()	773
10.199.3.10	start_tracking()	773
10.199.3.11	stop_tracking()	773
10.200	GpsL5iPcpsAcquisition Class Reference	774
10.200.1	Detailed Description	775
10.200.2	Member Function Documentation	775
10.200.2.1	implementation()	775
10.200.2.2	init()	775
10.200.2.3	mag()	775
10.200.2.4	reset()	776
10.200.2.5	set_channel()	776
10.200.2.6	set_channel_fsm()	776
10.200.2.7	set_doppler_center()	776
10.200.2.8	set_doppler_max()	776
10.200.2.9	set_doppler_step()	777

10.200.2.1	<del>set</del> _gnss_synchro()	777
10.200.2.1	<del>set</del> _local_code()	777
10.200.2.1	<del>set</del> _resampler_latency()	777
10.200.2.1	<del>set</del> _state()	777
10.200.2.1	<del>set</del> _threshold()	778
10.200.2.1	<del>stop</del> _acquisition()	778
10.201	GpsL5iPcpsAcquisitionFpga Class Reference	778
10.201.1	Detailed Description	780
10.201.2	Constructor & Destructor Documentation	780
10.201.2.1	GpsL5iPcpsAcquisitionFpga()	780
10.201.2.2	~GpsL5iPcpsAcquisitionFpga()	780
10.201.3	Member Function Documentation	780
10.201.3.1	connect()	780
10.201.3.2	disconnect()	781
10.201.3.3	get_left_block()	781
10.201.3.4	get_right_block()	781
10.201.3.5	implementation()	781
10.201.3.6	init()	781
10.201.3.7	item_size()	782
10.201.3.8	mag()	782
10.201.3.9	reset()	782
10.201.3.10	role()	782
10.201.3.11	<del>set</del> _channel()	782
10.201.3.12	<del>set</del> _channel_fsm()	783
10.201.3.13	<del>set</del> _doppler_center()	783
10.201.3.14	<del>set</del> _doppler_max()	783
10.201.3.15	<del>set</del> _doppler_step()	783
10.201.3.16	<del>set</del> _gnss_synchro()	783
10.201.3.17	<del>set</del> _local_code()	784
10.201.3.18	<del>set</del> _resampler_latency()	784

10.201.3.1	<a href="#">set_state()</a> . . . . .	784
10.201.3.2	<a href="#">set_threshold()</a> . . . . .	784
10.201.3.3	<a href="#">stop_acquisition()</a> . . . . .	784
10.202	<a href="#">SpsL5TelemetryDecoder Class Reference</a> . . . . .	785
10.202.1	<a href="#">Detailed Description</a> . . . . .	785
10.202.2	<a href="#">Member Function Documentation</a> . . . . .	785
10.202.2.1	<a href="#">implementation()</a> . . . . .	785
10.203	<a href="#">GPU_Complex Struct Reference</a> . . . . .	786
10.203.1	<a href="#">Detailed Description</a> . . . . .	786
10.204	<a href="#">GPU_Complex_Short Struct Reference</a> . . . . .	786
10.204.1	<a href="#">Detailed Description</a> . . . . .	786
10.205	<a href="#">Gpx_Printer Class Reference</a> . . . . .	787
10.205.1	<a href="#">Detailed Description</a> . . . . .	787
10.206	<a href="#">Gr_Complex_Ip_Packet_Source Class Reference</a> . . . . .	787
10.206.1	<a href="#">Detailed Description</a> . . . . .	788
10.207	<a href="#">time_t Struct Reference</a> . . . . .	788
10.207.1	<a href="#">Detailed Description</a> . . . . .	788
10.208	<a href="#">half_cyc_tag Struct Reference</a> . . . . .	788
10.208.1	<a href="#">Detailed Description</a> . . . . .	788
10.209	<a href="#">Hybrid_observables_gs Class Reference</a> . . . . .	789
10.209.1	<a href="#">Detailed Description</a> . . . . .	789
10.210	<a href="#">HybridObservables Class Reference</a> . . . . .	789
10.210.1	<a href="#">Detailed Description</a> . . . . .	790
10.210.2	<a href="#">Member Function Documentation</a> . . . . .	790
10.210.2.1	<a href="#">implementation()</a> . . . . .	790
10.210.2.2	<a href="#">item_size()</a> . . . . .	790
10.211	<a href="#">byteToCbyte Class Reference</a> . . . . .	791
10.211.1	<a href="#">Detailed Description</a> . . . . .	791
10.211.2	<a href="#">Member Function Documentation</a> . . . . .	791
10.211.2.1	<a href="#">implementation()</a> . . . . .	791

10.212	byteToComplex Class Reference	792
10.212.1	Detailed Description	792
10.212.2	Member Function Documentation	792
10.212.2.1	implementation()	792
10.213	byteToCshort Class Reference	793
10.213.1	Detailed Description	793
10.213.2	Member Function Documentation	793
10.213.2.1	implementation()	793
10.214	INIReader Class Reference	794
10.214.1	Detailed Description	794
10.214.2	Constructor & Destructor Documentation	794
10.214.2.1	INIReader()	794
10.214.3	Member Function Documentation	794
10.214.3.1	Get()	794
10.214.3.2	GetInteger()	795
10.214.3.3	ParseError()	795
10.215	MemoryConfiguration Class Reference	795
10.215.1	Detailed Description	796
10.216	Interleaved_byte_to_complex_byte Class Reference	796
10.216.1	Detailed Description	796
10.217	Interleaved_byte_to_complex_short Class Reference	797
10.217.1	Detailed Description	797
10.218	Interleaved_short_to_complex_short Class Reference	797
10.218.1	Detailed Description	798
10.219	shortToComplex Class Reference	798
10.219.1	Detailed Description	799
10.219.2	Member Function Documentation	799
10.219.2.1	implementation()	799
10.220	shortToCshort Class Reference	799
10.220.1	Detailed Description	800

10.220.2	Member Function Documentation	800
10.220.2.1	implementation()	800
10.221	kernel_info_t Struct Reference	800
10.221.1	Detailed Description	800
10.222	Xml_Printer Class Reference	801
10.222.1	Detailed Description	801
10.223	absat23_source Class Reference	801
10.223.1	Detailed Description	802
10.224	absatSignalSource Class Reference	802
10.224.1	Detailed Description	802
10.224.2	Member Function Documentation	802
10.224.2.1	implementation()	803
10.225	Ex_t Struct Reference	803
10.225.1	Detailed Description	803
10.226	Exeph_t Struct Reference	803
10.226.1	Detailed Description	804
10.227	Exion_t Struct Reference	804
10.227.1	Detailed Description	804
10.228	Exmsg_t Struct Reference	804
10.228.1	Detailed Description	804
10.229	MmseResamplerConditioner Class Reference	805
10.229.1	Detailed Description	805
10.230	ModelFunction Class Reference	805
10.230.1	Detailed Description	805
10.231	Monitor_Pvt Class Reference	806
10.231.1	Detailed Description	806
10.231.2	Member Function Documentation	807
10.231.2.1	serialize()	807
10.232	Monitor_Pvt_Udp_Sink Class Reference	807
10.232.1	Detailed Description	807

10.238	<a href="#">hsm_h_t Struct Reference</a>	807
10.233.1	<a href="#">Detailed Description</a>	808
10.234	<a href="#">ht1_header Struct Reference</a>	808
10.234.1	<a href="#">Detailed Description</a>	808
10.235	<a href="#">MultichannelFileSignalSource Class Reference</a>	808
10.235.1	<a href="#">Detailed Description</a>	809
10.235.2	<a href="#">Member Function Documentation</a>	809
10.235.2.1	<a href="#">implementation()</a>	809
10.236	<a href="#">lav_t Struct Reference</a>	810
10.236.1	<a href="#">Detailed Description</a>	811
10.237	<a href="#">Nmea_Printer Class Reference</a>	811
10.237.1	<a href="#">Detailed Description</a>	811
10.237.2	<a href="#">Constructor &amp; Destructor Documentation</a>	811
10.237.2.1	<a href="#">Nmea_Printer()</a>	811
10.237.2.2	<a href="#">~Nmea_Printer()</a>	812
10.237.3	<a href="#">Member Function Documentation</a>	812
10.237.3.1	<a href="#">Print_Nmea_Line()</a>	812
10.238	<a href="#">Notch Class Reference</a>	812
10.238.1	<a href="#">Detailed Description</a>	813
10.239	<a href="#">NotchFilter Class Reference</a>	813
10.239.1	<a href="#">Detailed Description</a>	813
10.239.2	<a href="#">Member Function Documentation</a>	813
10.239.2.1	<a href="#">implementation()</a>	814
10.240	<a href="#">NotchFilterLite Class Reference</a>	814
10.240.1	<a href="#">Detailed Description</a>	814
10.240.2	<a href="#">Member Function Documentation</a>	814
10.240.2.1	<a href="#">implementation()</a>	815
10.241	<a href="#">NotchLite Class Reference</a>	815
10.241.1	<a href="#">Detailed Description</a>	815
10.242	<a href="#">NsrFileSignalSource Class Reference</a>	816

10.242.1	Detailed Description	816
10.242.2	Member Function Documentation	816
10.242.2.1	implementation()	816
10.243	trip_t Struct Reference	817
10.243.1	Detailed Description	817
10.244	Obs_Conf Class Reference	817
10.244.1	Detailed Description	817
10.245	bs_t Struct Reference	818
10.245.1	Detailed Description	818
10.246	bsd_t Struct Reference	818
10.246.1	Detailed Description	818
10.247	Observables_Dump_Reader Class Reference	818
10.247.1	Detailed Description	819
10.248	ObservablesInterface Class Reference	819
10.248.1	Detailed Description	819
10.249	opt_t Struct Reference	820
10.249.1	Detailed Description	820
10.250	OsmosdrSignalSource Class Reference	820
10.250.1	Detailed Description	821
10.250.2	Member Function Documentation	821
10.250.2.1	implementation()	821
10.251	Pass_Through Class Reference	821
10.251.1	Detailed Description	822
10.251.2	Member Function Documentation	822
10.251.2.1	implementation()	822
10.252	clk_t Struct Reference	822
10.252.1	Detailed Description	822
10.253	pcps_acquisition Class Reference	823
10.253.1	Detailed Description	824
10.253.2	Member Function Documentation	824

10.253.2.1	general_work()	824
10.253.2.2	nit()	824
10.253.2.3	mag()	824
10.253.2.4	set_active()	824
10.253.2.5	set_channel()	825
10.253.2.6	set_channel_fsm()	825
10.253.2.7	set_doppler_center()	825
10.253.2.8	set_doppler_max()	826
10.253.2.9	set_doppler_step()	826
10.253.2.10	set_gnss_synchro()	826
10.253.2.11	set_local_code()	826
10.253.2.12	set_state()	827
10.253.2.13	set_threshold()	827
10.254	pcps_acquisition_fine_doppler_cc Class Reference	827
10.254.1	Detailed Description	828
10.254.2	Constructor & Destructor Documentation	828
10.254.2.1	~pcps_acquisition_fine_doppler_cc()	829
10.254.3	Member Function Documentation	829
10.254.3.1	general_work()	829
10.254.3.2	nit()	829
10.254.3.3	mag()	829
10.254.3.4	nextPowerOf2()	829
10.254.3.5	set_active()	830
10.254.3.6	set_channel()	830
10.254.3.7	set_channel_fsm()	830
10.254.3.8	set_doppler_max()	831
10.254.3.9	set_doppler_step()	832
10.254.3.10	set_gnss_synchro()	832
10.254.3.11	set_local_code()	832
10.254.3.12	set_state()	833

10.254.3.1	<a href="#">set_threshold()</a> . . . . .	833
10.255	<a href="#">pcps_acquisition_fpga Class Reference</a> . . . . .	833
10.255.1	<a href="#">Detailed Description</a> . . . . .	834
10.255.2	<a href="#">Constructor &amp; Destructor Documentation</a> . . . . .	834
10.255.2.1	<a href="#">~pcps_acquisition_fpga()</a> . . . . .	834
10.255.3	<a href="#">Member Function Documentation</a> . . . . .	835
10.255.3.1	<a href="#">init()</a> . . . . .	835
10.255.3.2	<a href="#">mag()</a> . . . . .	835
10.255.3.3	<a href="#">reset_acquisition()</a> . . . . .	835
10.255.3.4	<a href="#">set_active()</a> . . . . .	835
10.255.3.5	<a href="#">set_channel()</a> . . . . .	835
10.255.3.6	<a href="#">set_channel_fsm()</a> . . . . .	837
10.255.3.7	<a href="#">set_doppler_center()</a> . . . . .	837
10.255.3.8	<a href="#">set_doppler_max()</a> . . . . .	837
10.255.3.9	<a href="#">set_doppler_step()</a> . . . . .	838
10.255.3.10	<a href="#">set_gnss_synchro()</a> . . . . .	838
10.255.3.11	<a href="#">set_local_code()</a> . . . . .	838
10.255.3.12	<a href="#">set_state()</a> . . . . .	838
10.255.3.13	<a href="#">set_threshold()</a> . . . . .	839
10.256	<a href="#">pcps_assisted_acquisition_cc Class Reference</a> . . . . .	839
10.256.1	<a href="#">Detailed Description</a> . . . . .	840
10.256.2	<a href="#">Constructor &amp; Destructor Documentation</a> . . . . .	840
10.256.2.1	<a href="#">~pcps_assisted_acquisition_cc()</a> . . . . .	840
10.256.3	<a href="#">Member Function Documentation</a> . . . . .	840
10.256.3.1	<a href="#">general_work()</a> . . . . .	841
10.256.3.2	<a href="#">nit()</a> . . . . .	841
10.256.3.3	<a href="#">mag()</a> . . . . .	841
10.256.3.4	<a href="#">set_active()</a> . . . . .	841
10.256.3.5	<a href="#">set_channel()</a> . . . . .	841
10.256.3.6	<a href="#">set_channel_fsm()</a> . . . . .	842

10.256.3.7	<a href="#">set_doppler_max()</a> . . . . .	842
10.256.3.8	<a href="#">set_doppler_step()</a> . . . . .	842
10.256.3.9	<a href="#">set_gnss_synchro()</a> . . . . .	843
10.256.3.10	<a href="#">set_local_code()</a> . . . . .	843
10.256.3.11	<a href="#">set_threshold()</a> . . . . .	843
10.257	<a href="#">pcps_cccwsr_acquisition_cc Class Reference</a> . . . . .	844
10.257.1	<a href="#">Detailed Description</a> . . . . .	845
10.257.2	<a href="#">Constructor &amp; Destructor Documentation</a> . . . . .	845
10.257.2.1	<a href="#">~pcps_cccwsr_acquisition_cc()</a> . . . . .	845
10.257.3	<a href="#">Member Function Documentation</a> . . . . .	845
10.257.3.1	<a href="#">general_work()</a> . . . . .	845
10.257.3.2	<a href="#">nit()</a> . . . . .	845
10.257.3.3	<a href="#">mag()</a> . . . . .	845
10.257.3.4	<a href="#">set_active()</a> . . . . .	845
10.257.3.5	<a href="#">set_channel()</a> . . . . .	846
10.257.3.6	<a href="#">set_channel_fsm()</a> . . . . .	846
10.257.3.7	<a href="#">set_doppler_max()</a> . . . . .	846
10.257.3.8	<a href="#">set_doppler_step()</a> . . . . .	847
10.257.3.9	<a href="#">set_gnss_synchro()</a> . . . . .	847
10.257.3.10	<a href="#">set_local_code()</a> . . . . .	847
10.257.3.11	<a href="#">set_state()</a> . . . . .	847
10.257.3.12	<a href="#">set_threshold()</a> . . . . .	848
10.258	<a href="#">pcps_openc1_acquisition_cc Class Reference</a> . . . . .	848
10.258.1	<a href="#">Detailed Description</a> . . . . .	849
10.258.2	<a href="#">Constructor &amp; Destructor Documentation</a> . . . . .	849
10.258.2.1	<a href="#">~pcps_openc1_acquisition_cc()</a> . . . . .	850
10.258.3	<a href="#">Member Function Documentation</a> . . . . .	850
10.258.3.1	<a href="#">general_work()</a> . . . . .	850
10.258.3.2	<a href="#">nit()</a> . . . . .	850
10.258.3.3	<a href="#">mag()</a> . . . . .	850

10.258.3.4	<a href="#">set_active()</a> . . . . .	850
10.258.3.5	<a href="#">set_channel()</a> . . . . .	851
10.258.3.6	<a href="#">set_channel_fsm()</a> . . . . .	851
10.258.3.7	<a href="#">set_doppler_max()</a> . . . . .	851
10.258.3.8	<a href="#">set_doppler_step()</a> . . . . .	852
10.258.3.9	<a href="#">set_gnss_synchro()</a> . . . . .	852
10.258.3.10	<a href="#">set_local_code()</a> . . . . .	852
10.258.3.11	<a href="#">set_state()</a> . . . . .	852
10.258.3.12	<a href="#">set_threshold()</a> . . . . .	854
10.259	<a href="#">pcps_quicksync_acquisition_cc Class Reference</a> . . . . .	854
10.259.1	<a href="#">Detailed Description</a> . . . . .	855
10.259.2	<a href="#">Constructor &amp; Destructor Documentation</a> . . . . .	855
10.259.2.1	<a href="#">~pcps_quicksync_acquisition_cc()</a> . . . . .	855
10.259.3	<a href="#">Member Function Documentation</a> . . . . .	856
10.259.3.1	<a href="#">general_work()</a> . . . . .	856
10.259.3.2	<a href="#">nit()</a> . . . . .	856
10.259.3.3	<a href="#">mag()</a> . . . . .	856
10.259.3.4	<a href="#">set_active()</a> . . . . .	856
10.259.3.5	<a href="#">set_channel()</a> . . . . .	857
10.259.3.6	<a href="#">set_channel_fsm()</a> . . . . .	857
10.259.3.7	<a href="#">set_doppler_max()</a> . . . . .	857
10.259.3.8	<a href="#">set_doppler_step()</a> . . . . .	857
10.259.3.9	<a href="#">set_gnss_synchro()</a> . . . . .	858
10.259.3.10	<a href="#">set_local_code()</a> . . . . .	858
10.259.3.11	<a href="#">set_state()</a> . . . . .	858
10.259.3.12	<a href="#">set_threshold()</a> . . . . .	859
10.260	<a href="#">pcps_tong_acquisition_cc Class Reference</a> . . . . .	859
10.260.1	<a href="#">Detailed Description</a> . . . . .	860
10.260.2	<a href="#">Constructor &amp; Destructor Documentation</a> . . . . .	860
10.260.2.1	<a href="#">~pcps_tong_acquisition_cc()</a> . . . . .	860

10.260.3 Member Function Documentation . . . . .	860
10.260.3.1 general_work() . . . . .	861
10.260.3.2 nit() . . . . .	861
10.260.3.3 mag() . . . . .	861
10.260.3.4 set_active() . . . . .	861
10.260.3.5 set_channel() . . . . .	861
10.260.3.6 set_channel_fsm() . . . . .	862
10.260.3.7 set_doppler_max() . . . . .	862
10.260.3.8 set_doppler_step() . . . . .	862
10.260.3.9 set_gnss_synchro() . . . . .	863
10.260.3.10 set_local_code() . . . . .	863
10.260.3.11 set_state() . . . . .	863
10.260.3.12 set_threshold() . . . . .	863
10.261 pcpsconf_fpga_t Struct Reference . . . . .	864
10.261.1 Detailed Description . . . . .	864
10.262 pcv_t Struct Reference . . . . .	864
10.262.1 Detailed Description . . . . .	865
10.263 pcvs_t Struct Reference . . . . .	865
10.263.1 Detailed Description . . . . .	865
10.264 peph_t Struct Reference . . . . .	865
10.264.1 Detailed Description . . . . .	865
10.265 PlutosdrSignalSource Class Reference . . . . .	866
10.265.1 Detailed Description . . . . .	866
10.265.2 Member Function Documentation . . . . .	866
10.265.2.1 implementation() . . . . .	866
10.266 pppcorr_t Struct Reference . . . . .	867
10.266.1 Detailed Description . . . . .	867
10.267 prcopt_t Struct Reference . . . . .	867
10.267.1 Detailed Description . . . . .	868
10.268 pulse_blanking_cc Class Reference . . . . .	868

10.268.1 Detailed Description . . . . .	869
10.269 PulseBlankingFilter Class Reference . . . . .	869
10.269.1 Detailed Description . . . . .	869
10.269.2 Member Function Documentation . . . . .	870
10.269.2.1 implementation() . . . . .	870
10.270 Pvt_Conf Class Reference . . . . .	870
10.270.1 Detailed Description . . . . .	871
10.271 Pvt_Solution Class Reference . . . . .	871
10.271.1 Detailed Description . . . . .	873
10.271.2 Member Function Documentation . . . . .	873
10.271.2.1 get_avg_height() . . . . .	873
10.271.2.2 get_avg_latitude() . . . . .	873
10.271.2.3 get_avg_longitude() . . . . .	873
10.271.2.4 get_clock_drift_ppm() . . . . .	873
10.271.2.5 get_course_over_ground() . . . . .	873
10.271.2.6 get_height() . . . . .	874
10.271.2.7 get_latitude() . . . . .	874
10.271.2.8 get_longitude() . . . . .	874
10.271.2.9 get_num_valid_observations() . . . . .	874
10.271.2.10 get_speed_over_ground() . . . . .	874
10.271.2.11 get_time_offset_s() . . . . .	874
10.271.2.12 set_averaging_depth() . . . . .	875
10.271.2.13 set_clock_drift_ppm() . . . . .	875
10.271.2.14 set_course_over_ground() . . . . .	875
10.271.2.15 set_num_valid_observations() . . . . .	875
10.271.2.16 set_pre_2009_file() . . . . .	875
10.271.2.17 set_rx_pos() . . . . .	875
10.271.2.18 set_rx_vel() . . . . .	876
10.271.2.19 set_speed_over_ground() . . . . .	876
10.271.2.20 set_time_offset_s() . . . . .	876

10.272	<a href="#">PvtInterface Class Reference</a>	876
10.272.1	<a href="#">Detailed Description</a>	877
10.273	<a href="#">Raw_t Struct Reference</a>	877
10.273.1	<a href="#">Detailed Description</a>	878
10.274	<a href="#">RawArraySignalSource Class Reference</a>	878
10.274.1	<a href="#">Detailed Description</a>	879
10.274.2	<a href="#">Member Function Documentation</a>	879
10.274.2.1	<a href="#">implementation()</a>	879
10.275	<a href="#">Rinex_Printer Class Reference</a>	879
10.275.1	<a href="#">Detailed Description</a>	880
10.275.2	<a href="#">Constructor &amp; Destructor Documentation</a>	880
10.275.2.1	<a href="#">Rinex_Printer()</a>	880
10.275.2.2	<a href="#">~Rinex_Printer()</a>	880
10.275.3	<a href="#">Member Function Documentation</a>	880
10.275.3.1	<a href="#">get_navfilename()</a>	881
10.275.3.2	<a href="#">get_obsfilename()</a>	881
10.275.3.3	<a href="#">is_rinex_header_written()</a>	881
10.275.3.4	<a href="#">log_rinex_nav_bds_dnav()</a>	881
10.275.3.5	<a href="#">log_rinex_nav_gal_nav()</a>	881
10.275.3.6	<a href="#">log_rinex_nav_glo_gnav()</a>	882
10.275.3.7	<a href="#">log_rinex_nav_gps_cnav()</a>	882
10.275.3.8	<a href="#">log_rinex_nav_gps_nav()</a>	882
10.275.3.9	<a href="#">print_rinex_annotation()</a>	882
10.275.3.10	<a href="#">set_pre_2009_file()</a>	884
10.276	<a href="#">Rtcm Class Reference</a>	884
10.276.1	<a href="#">Detailed Description</a>	887
10.276.2	<a href="#">Constructor &amp; Destructor Documentation</a>	887
10.276.2.1	<a href="#">Rtcm()</a>	887
10.276.3	<a href="#">Member Function Documentation</a>	887
10.276.3.1	<a href="#">bin_to_binary_data()</a>	887

10.276.3.2	<a href="#">bin_to_double()</a>	888
10.276.3.3	<a href="#">bin_to_hex()</a>	888
10.276.3.4	<a href="#">bin_to_uint()</a>	888
10.276.3.5	<a href="#">binary_data_to_bin()</a>	888
10.276.3.6	<a href="#">check_CRC()</a>	888
10.276.3.7	<a href="#">hex_to_bin()</a>	888
10.276.3.8	<a href="#">hex_to_int()</a>	889
10.276.3.9	<a href="#">hex_to_uint()</a>	889
10.276.3.10	<a href="#">is_server_running()</a>	889
10.276.3.11	<a href="#">lock_time()</a> [1/4]	889
10.276.3.12	<a href="#">lock_time()</a> [2/4]	889
10.276.3.13	<a href="#">lock_time()</a> [3/4]	890
10.276.3.14	<a href="#">lock_time()</a> [4/4]	890
10.276.3.15	<a href="#">print_MSM_1()</a>	890
10.276.3.16	<a href="#">print_MSM_2()</a>	891
10.276.3.17	<a href="#">print_MSM_3()</a>	891
10.276.3.18	<a href="#">print_MSM_4()</a>	891
10.276.3.19	<a href="#">print_MSM_5()</a>	892
10.276.3.20	<a href="#">print_MSM_6()</a>	892
10.276.3.21	<a href="#">print_MSM_7()</a>	892
10.276.3.22	<a href="#">print_MT1001()</a>	893
10.276.3.23	<a href="#">print_MT1002()</a>	893
10.276.3.24	<a href="#">print_MT1003()</a>	893
10.276.3.25	<a href="#">print_MT1004()</a>	893
10.276.3.26	<a href="#">print_MT1005()</a>	894
10.276.3.27	<a href="#">print_MT1005_test()</a>	894
10.276.3.28	<a href="#">print_MT1006()</a>	894
10.276.3.29	<a href="#">print_MT1008()</a>	894
10.276.3.30	<a href="#">print_MT1009()</a>	895
10.276.3.31	<a href="#">print_MT1010()</a>	895

10.276.3.32	Print_MT1011()	896
10.276.3.33	Print_MT1012()	896
10.276.3.34	Print_MT1019()	897
10.276.3.35	Print_MT1020()	897
10.276.3.36	Print_MT1029()	897
10.276.3.37	Print_MT1045()	898
10.276.3.38	Read_MT1005()	898
10.276.3.39	Read_MT1019()	898
10.276.3.40	Read_MT1020()	898
10.276.3.41	Read_MT1045()	899
10.276.3.42	Run_server()	899
10.276.3.43	Send_message()	899
10.276.3.44	Stop_server()	899
10.277	Rtcm_Printer Class Reference	900
10.277.1	Detailed Description	900
10.277.2	Constructor & Destructor Documentation	900
10.277.2.1	Rtcm_Printer()	900
10.277.2.2	~Rtcm_Printer()	901
10.277.3	Member Function Documentation	901
10.277.3.1	lock_time()	901
10.277.3.2	print_MT1005_test()	901
10.277.3.3	Print_Rtcm_Messages()	902
10.278	Rtcm_t Struct Reference	902
10.278.1	Detailed Description	903
10.279	Rtk_t Struct Reference	903
10.279.1	Detailed Description	903
10.280	Rtklib_Pvt Class Reference	903
10.280.1	Detailed Description	904
10.280.2	Member Function Documentation	904
10.280.2.1	implementation()	904

10.280.2.2	item_size()	905
10.281	rtklib_pvt_gs Class Reference	905
10.281.1	Detailed Description	906
10.281.2	Constructor & Destructor Documentation	906
10.281.2.1	~rtklib_pvt_gs()	906
10.281.3	Member Function Documentation	906
10.281.3.1	clear_ephemeris()	906
10.281.3.2	get_beidou_dnav_almanac_map()	906
10.281.3.3	get_beidou_dnav_ephemeris_map()	906
10.281.3.4	get_galileo_almanac_map()	907
10.281.3.5	get_galileo_ephemeris_map()	907
10.281.3.6	get_gps_almanac_map()	907
10.281.3.7	get_gps_ephemeris_map()	907
10.281.3.8	get_latest_PVT()	907
10.281.3.9	work()	908
10.282	rtklib_Solver Class Reference	908
10.282.1	Detailed Description	909
10.282.2	Member Data Documentation	909
10.282.2.1	beidou_dnav_ephemeris_map	909
10.282.2.2	galileo_ephemeris_map	909
10.282.2.3	glonass_gnav_almanac	910
10.282.2.4	glonass_gnav_ephemeris_map	910
10.282.2.5	glonass_gnav_utc_model	910
10.282.2.6	gps_cnav_ephemeris_map	910
10.282.2.7	gps_ephemeris_map	910
10.283	rtklib_Solver_Dump_Reader Class Reference	911
10.283.1	Detailed Description	911
10.284	tksvr_t Struct Reference	911
10.284.1	Detailed Description	912
10.285	tl_Tcp_Dongle_Info Class Reference	912

10.285.1 Detailed Description . . . . .	913
10.286.1 <del>l_tcp_signal_source_c</del> Class Reference . . . . .	913
10.286.1 Detailed Description . . . . .	914
10.287.1 <del>Rtl</del> TcpSignalSource Class Reference . . . . .	914
10.287.1 Detailed Description . . . . .	914
10.287.2 Member Function Documentation . . . . .	914
10.287.2.1 implementation() . . . . .	915
10.288.1 <del>bas_Ephemeris</del> Class Reference . . . . .	915
10.288.1 Detailed Description . . . . .	916
10.288.2 Member Data Documentation . . . . .	916
10.288.2.1 <del>b_sv_do_not_use</del> . . . . .	916
10.288.2.2 <del>d_acc</del> . . . . .	916
10.288.2.3 <del>d_af0</del> . . . . .	916
10.288.2.4 <del>d_af1</del> . . . . .	916
10.288.2.5 <del>d_pos</del> . . . . .	917
10.288.2.6 <del>d_tof</del> . . . . .	917
10.288.2.7 <del>d_vel</del> . . . . .	917
10.288.2.8 <del>d_prn</del> . . . . .	917
10.288.2.9 <del>sv_ura</del> . . . . .	917
10.288.2.10 <del>QtoI</del> . . . . .	918
10.289.1 <del>bas_l1_telemetry_decoder_gs</del> Class Reference . . . . .	918
10.289.1 Detailed Description . . . . .	918
10.289.2 Member Function Documentation . . . . .	919
10.289.2.1 general_work() . . . . .	919
10.289.2.2 <del>set_channel()</del> . . . . .	919
10.289.2.3 <del>set_satellite()</del> . . . . .	919
10.290.1 <del>basL1TelemetryDecoder</del> Class Reference . . . . .	919
10.290.1 Detailed Description . . . . .	920
10.290.2 Member Function Documentation . . . . .	920
10.290.2.1 implementation() . . . . .	920

10.291bs_t Struct Reference . . . . .	920
10.291.1Detailed Description . . . . .	921
10.292bsfcrr_t Struct Reference . . . . .	921
10.292.1Detailed Description . . . . .	921
10.293bsign_t Struct Reference . . . . .	921
10.293.1Detailed Description . . . . .	921
10.294bsignband_t Struct Reference . . . . .	922
10.294.1Detailed Description . . . . .	922
10.295bsion_t Struct Reference . . . . .	922
10.295.1Detailed Description . . . . .	922
10.296bslcorr_t Struct Reference . . . . .	922
10.296.1Detailed Description . . . . .	923
10.297bsmsg_t Struct Reference . . . . .	923
10.297.1Detailed Description . . . . .	923
10.298bssat_t Struct Reference . . . . .	923
10.298.1Detailed Description . . . . .	923
10.299bssatp_t Struct Reference . . . . .	924
10.299.1Detailed Description . . . . .	924
10.300eph_t Struct Reference . . . . .	924
10.300.1Detailed Description . . . . .	924
10.301Serdes_Gnss_Synchro Class Reference . . . . .	924
10.301.1Detailed Description . . . . .	925
10.301.2Constructor & Destructor Documentation . . . . .	925
10.301.2.1Serdes_Gnss_Synchro() [1/2] . . . . .	925
10.301.2.2Serdes_Gnss_Synchro() [2/2] . . . . .	925
10.301.3Member Function Documentation . . . . .	925
10.301.3.1createProtobuffer() . . . . .	925
10.301.3.2operator=() [1/2] . . . . .	926
10.301.3.3operator=() [2/2] . . . . .	926
10.301.3.4readProtobuffer() . . . . .	926

10.302	<del>Serial_t</del> Serdes_Monitor_Pvt Class Reference . . . . .	927
10.302.1	Detailed Description . . . . .	927
10.302.2	Constructor & Destructor Documentation . . . . .	927
10.302.2.1	Serdes_Monitor_Pvt() [1/2] . . . . .	927
10.302.2.2	Serdes_Monitor_Pvt() [2/2] . . . . .	927
10.302.3	Member Function Documentation . . . . .	928
10.302.3.1	createProtobuffer() . . . . .	928
10.302.3.2	operator=() [1/2] . . . . .	929
10.302.3.3	operator=() [2/2] . . . . .	929
10.302.3.4	readProtobuffer() . . . . .	929
10.303	Serial_t Struct Reference . . . . .	929
10.303.1	Detailed Description . . . . .	930
10.304	short_x2_to_cshort Class Reference . . . . .	930
10.304.1	Detailed Description . . . . .	930
10.305	signal_generator_c Class Reference . . . . .	931
10.305.1	Detailed Description . . . . .	931
10.305.2	Friends And Related Function Documentation . . . . .	931
10.305.2.1	signal_make_generator_c . . . . .	932
10.306	SignalConditioner Class Reference . . . . .	932
10.306.1	Detailed Description . . . . .	933
10.306.2	Constructor & Destructor Documentation . . . . .	933
10.306.2.1	SignalConditioner() . . . . .	933
10.306.2.2	~SignalConditioner() . . . . .	933
10.306.3	Member Function Documentation . . . . .	933
10.306.3.1	implementation() . . . . .	933
10.307	SignalGenerator Class Reference . . . . .	934
10.307.1	Detailed Description . . . . .	934
10.307.2	Member Function Documentation . . . . .	934
10.307.2.1	implementation() . . . . .	934
10.308	nrmask_t Struct Reference . . . . .	935

10.308.1 Detailed Description . . . . .	935
10.309.1 sol_t Struct Reference . . . . .	935
10.309.1 Detailed Description . . . . .	935
10.310.1 solbuf_t Struct Reference . . . . .	935
10.310.1 Detailed Description . . . . .	936
10.311.1 solopt_t Struct Reference . . . . .	936
10.311.1 Detailed Description . . . . .	936
10.312.1 solstat_t Struct Reference . . . . .	936
10.312.1 Detailed Description . . . . .	937
10.313.1 solstatbuf_t Struct Reference . . . . .	937
10.313.1 Detailed Description . . . . .	937
10.314.1 spirent_Motion_Csv_Dump_Reader Class Reference . . . . .	937
10.314.1 Detailed Description . . . . .	938
10.315.1 SpirFileSignalSource Class Reference . . . . .	939
10.315.1 Detailed Description . . . . .	939
10.315.2 Member Function Documentation . . . . .	939
10.315.2.1 implementation() . . . . .	939
10.316.1 SpirGSS6450FileSignalSource Class Reference . . . . .	940
10.316.1 Detailed Description . . . . .	940
10.317.1 ssat_t Struct Reference . . . . .	941
10.317.1 Detailed Description . . . . .	941
10.318.1 sr_t Struct Reference . . . . .	941
10.318.1 Detailed Description . . . . .	942
10.319.1 sta_t Struct Reference . . . . .	942
10.319.1 Detailed Description . . . . .	942
10.320.1 tec_t Struct Reference . . . . .	942
10.320.1 Detailed Description . . . . .	943
10.321.1 stream_cfg Struct Reference . . . . .	943
10.321.1 Detailed Description . . . . .	943
10.322.1 stream_t Struct Reference . . . . .	943

10.322.1 Detailed Description . . . . .	944
10.323 StringConverter Class Reference . . . . .	944
10.323.1 Detailed Description . . . . .	944
10.324 cp_Communication Class Reference . . . . .	944
10.324.1 Detailed Description . . . . .	945
10.325 cp_Packet_Data Class Reference . . . . .	945
10.325.1 Detailed Description . . . . .	945
10.326 cp_t Struct Reference . . . . .	946
10.326.1 Detailed Description . . . . .	946
10.327 cpcli_t Struct Reference . . . . .	946
10.327.1 Detailed Description . . . . .	946
10.328 cpCmdInterface Class Reference . . . . .	946
10.328.1 Detailed Description . . . . .	947
10.328.2 Member Function Documentation . . . . .	947
10.328.2.1 get_LLH() . . . . .	947
10.328.2.2 get_utc_time() . . . . .	947
10.329 cpsvr_t Struct Reference . . . . .	947
10.329.1 Detailed Description . . . . .	947
10.330 ec_t Struct Reference . . . . .	948
10.330.1 Detailed Description . . . . .	948
10.331 TelemetryDecoderInterface Class Reference . . . . .	948
10.331.1 Detailed Description . . . . .	949
10.332 e_t Struct Reference . . . . .	949
10.332.1 Detailed Description . . . . .	949
10.333 ed_t Struct Reference . . . . .	949
10.333.1 Detailed Description . . . . .	950
10.334 Im_Conf Class Reference . . . . .	950
10.334.1 Detailed Description . . . . .	950
10.335 Im_Dump_Reader Class Reference . . . . .	950
10.335.1 Detailed Description . . . . .	951

10.336	Tracking_2nd_DLL_filter Class Reference	951
10.336.1	Detailed Description	951
10.336.2	Member Function Documentation	951
10.336.2.1	get_code_nco()	951
10.336.2.2	initialize()	952
10.336.2.3	set_DLL_BW()	952
10.336.2.4	set_pdi()	952
10.337	Tracking_2nd_PLL_filter Class Reference	952
10.337.1	Detailed Description	953
10.337.2	Member Function Documentation	953
10.337.2.1	set_pdi()	953
10.337.2.2	set_PLL_BW()	953
10.338	Tracking_Dump_Reader Class Reference	953
10.338.1	Detailed Description	954
10.339	Tracking_FLL_PLL_filter Class Reference	954
10.339.1	Detailed Description	955
10.340	Tracking_loop_filter Class Reference	955
10.340.1	Detailed Description	955
10.340.2	Constructor & Destructor Documentation	955
10.340.2.1	Tracking_loop_filter()	956
10.340.3	Member Function Documentation	956
10.340.3.1	operator=()	956
10.341	Tracking_True_Obs_Reader Class Reference	956
10.341.1	Detailed Description	956
10.342	TrackingInterface Class Reference	957
10.342.1	Detailed Description	957
10.343	top_t Struct Reference	958
10.343.1	Detailed Description	958
10.344	True_Observables_Reader Class Reference	958
10.344.1	Detailed Description	958

10.345woBitCpxFileSignalSource Class Reference . . . . .	959
10.345.1Detailed Description . . . . .	959
10.345.2Member Function Documentation . . . . .	959
10.345.2.1implementation() . . . . .	959
10.346woBitPackedFileSignalSource Class Reference . . . . .	960
10.346.1Detailed Description . . . . .	960
10.346.2Member Function Documentation . . . . .	960
10.346.2.1implementation() . . . . .	961
10.347UhdSignalSource Class Reference . . . . .	961
10.347.1Detailed Description . . . . .	961
10.347.2Member Function Documentation . . . . .	962
10.347.2.1implementation() . . . . .	962
10.348unpack_2bit_samples Class Reference . . . . .	962
10.348.1Detailed Description . . . . .	963
10.349unpack_byte_2bit_cpx_samples Class Reference . . . . .	963
10.349.1Detailed Description . . . . .	963
10.350unpack_byte_2bit_samples Class Reference . . . . .	964
10.350.1Detailed Description . . . . .	964
10.351unpack_byte_4bit_samples Class Reference . . . . .	964
10.351.1Detailed Description . . . . .	965
10.352unpack_intspir_1bit_samples Class Reference . . . . .	965
10.352.1Detailed Description . . . . .	965
10.353unpack_spir_gss6450_samples Class Reference . . . . .	966
10.353.1Detailed Description . . . . .	966
10.354UnscentedFilter Class Reference . . . . .	966
10.354.1Detailed Description . . . . .	967
10.355url_t Struct Reference . . . . .	967
10.355.1Detailed Description . . . . .	967
10.35627_decision_t Struct Reference . . . . .	967
10.356.1Detailed Description . . . . .	967
10.35727_poly_t Struct Reference . . . . .	967
10.357.1Detailed Description . . . . .	968
10.35827_t Struct Reference . . . . .	968
10.358.1Detailed Description . . . . .	968
10.359Viterbi_Decoder Class Reference . . . . .	968
10.359.1Detailed Description . . . . .	969
10.359.2Member Function Documentation . . . . .	969
10.359.2.1decode_block() . . . . .	969

<b>11 File Documentation</b>	<b>971</b>
11.1 acq_conf.h File Reference . . . . .	971
11.1.1 Detailed Description . . . . .	971
11.2 acquisition_dump_reader.h File Reference . . . . .	971
11.2.1 Detailed Description . . . . .	972
11.3 acquisition_interface.h File Reference . . . . .	972
11.3.1 Detailed Description . . . . .	972
11.4 acquisition_msg_rx.h File Reference . . . . .	973
11.4.1 Detailed Description . . . . .	973
11.5 ad9361_fpga_signal_source.h File Reference . . . . .	973
11.5.1 Detailed Description . . . . .	974
11.6 ad9361_manager.h File Reference . . . . .	974
11.6.1 Detailed Description . . . . .	975
11.7 agnss_ref_location.h File Reference . . . . .	975
11.7.1 Detailed Description . . . . .	976
11.8 agnss_ref_time.h File Reference . . . . .	976
11.8.1 Detailed Description . . . . .	976
11.9 array_signal_conditioner.h File Reference . . . . .	976
11.9.1 Detailed Description . . . . .	977
11.10 bayesian_estimation.h File Reference . . . . .	977
11.10.1 Detailed Description . . . . .	977
11.11 beamformer.h File Reference . . . . .	978
11.11.1 Detailed Description . . . . .	978
11.12 beamformer_filter.h File Reference . . . . .	978
11.12.1 Detailed Description . . . . .	979
11.13 Beidou_B1I.h File Reference . . . . .	979
11.13.1 Detailed Description . . . . .	980
11.14 beidou_b1i_dll_pll_tracking.h File Reference . . . . .	980
11.14.1 Detailed Description . . . . .	980
11.15 beidou_b1i_pcps_acquisition.h File Reference . . . . .	981

11.15.1 Detailed Description . . . . .	981
11.16beidou_b1i_signal_replica.h File Reference . . . . .	981
11.16.1 Detailed Description . . . . .	982
11.17beidou_b1i_telemetry_decoder.h File Reference . . . . .	982
11.17.1 Detailed Description . . . . .	982
11.18beidou_b1i_telemetry_decoder_gs.h File Reference . . . . .	983
11.18.1 Detailed Description . . . . .	983
11.19Beidou_B3I.h File Reference . . . . .	984
11.19.1 Detailed Description . . . . .	984
11.20beidou_b3i_dll_pll_tracking.h File Reference . . . . .	985
11.20.1 Detailed Description . . . . .	985
11.21beidou_b3i_pcps_acquisition.h File Reference . . . . .	985
11.21.1 Detailed Description . . . . .	986
11.22beidou_b3i_signal_replica.h File Reference . . . . .	986
11.22.1 Detailed Description . . . . .	986
11.23beidou_b3i_telemetry_decoder.h File Reference . . . . .	987
11.23.1 Detailed Description . . . . .	987
11.24beidou_b3i_telemetry_decoder_gs.h File Reference . . . . .	987
11.24.1 Detailed Description . . . . .	988
11.25Beidou_DNAV.h File Reference . . . . .	988
11.25.1 Detailed Description . . . . .	992
11.26beidou_dnav_almanac.h File Reference . . . . .	993
11.26.1 Detailed Description . . . . .	993
11.27beidou_dnav_ephemeris.h File Reference . . . . .	993
11.27.1 Detailed Description . . . . .	993
11.28beidou_dnav_iono.h File Reference . . . . .	994
11.28.1 Detailed Description . . . . .	994
11.29beidou_dnav_navigation_message.h File Reference . . . . .	994
11.29.1 Detailed Description . . . . .	995
11.30beidou_dnav_utc_model.h File Reference . . . . .	995

11.30.1 Detailed Description . . . . .	995
11.31bits.h File Reference . . . . .	995
11.31.1 Detailed Description . . . . .	996
11.32byte_to_short.h File Reference . . . . .	996
11.32.1 Detailed Description . . . . .	997
11.33byte_x2_to_complex_byte.h File Reference . . . . .	997
11.33.1 Detailed Description . . . . .	997
11.34channel.h File Reference . . . . .	998
11.34.1 Detailed Description . . . . .	998
11.35channel_event.h File Reference . . . . .	998
11.35.1 Detailed Description . . . . .	999
11.36channel_fsm.h File Reference . . . . .	999
11.36.1 Detailed Description . . . . .	999
11.37channel_interface.h File Reference . . . . .	1000
11.37.1 Detailed Description . . . . .	1000
11.38channel_msg_receiver_cc.h File Reference . . . . .	1000
11.38.1 Detailed Description . . . . .	1001
11.39channel_status_msg_receiver.h File Reference . . . . .	1001
11.39.1 Detailed Description . . . . .	1002
11.40clFFT.h File Reference . . . . .	1002
11.40.1 Detailed Description . . . . .	1003
11.41cnav_msg.h File Reference . . . . .	1003
11.41.1 Detailed Description . . . . .	1003
11.42command_event.h File Reference . . . . .	1004
11.42.1 Detailed Description . . . . .	1004
11.43complex_byte_to_float_x2.h File Reference . . . . .	1004
11.43.1 Detailed Description . . . . .	1005
11.44complex_float_to_complex_byte.h File Reference . . . . .	1005
11.44.1 Detailed Description . . . . .	1006
11.45concurrent_map.h File Reference . . . . .	1006

11.45.1 Detailed Description . . . . .	1006
11.46concurrent_queue.h File Reference . . . . .	1006
11.46.1 Detailed Description . . . . .	1007
11.47configuration_interface.h File Reference . . . . .	1007
11.47.1 Detailed Description . . . . .	1007
11.48conjugate_cc.h File Reference . . . . .	1008
11.48.1 Detailed Description . . . . .	1008
11.49conjugate_ic.h File Reference . . . . .	1008
11.49.1 Detailed Description . . . . .	1009
11.50conjugate_sc.h File Reference . . . . .	1009
11.50.1 Detailed Description . . . . .	1010
11.51control_thread.h File Reference . . . . .	1010
11.51.1 Detailed Description . . . . .	1010
11.52convolutional.h File Reference . . . . .	1011
11.52.1 Detailed Description . . . . .	1011
11.53cpu_multicorrelator.h File Reference . . . . .	1012
11.53.1 Detailed Description . . . . .	1012
11.54cpu_multicorrelator_16sc.h File Reference . . . . .	1012
11.54.1 Detailed Description . . . . .	1012
11.55cpu_multicorrelator_real_codes.h File Reference . . . . .	1013
11.55.1 Detailed Description . . . . .	1013
11.56cshort_to_float_x2.h File Reference . . . . .	1013
11.56.1 Detailed Description . . . . .	1014
11.57cuda_multicorrelator.h File Reference . . . . .	1014
11.57.1 Detailed Description . . . . .	1014
11.58custom_udp_signal_source.h File Reference . . . . .	1015
11.58.1 Detailed Description . . . . .	1015
11.59direct_resampler_conditioner.h File Reference . . . . .	1015
11.59.1 Detailed Description . . . . .	1016
11.60direct_resampler_conditioner_cb.h File Reference . . . . .	1016

11.60.1 Detailed Description . . . . .	1016
11.61direct_resampler_conditioner_cc.h File Reference . . . . .	1017
11.61.1 Detailed Description . . . . .	1017
11.62direct_resampler_conditioner_cs.h File Reference . . . . .	1017
11.62.1 Detailed Description . . . . .	1018
11.63display.h File Reference . . . . .	1018
11.63.1 Detailed Description . . . . .	1019
11.64dll_pll_conf.h File Reference . . . . .	1019
11.64.1 Detailed Description . . . . .	1019
11.65dll_pll_conf_fpga.h File Reference . . . . .	1019
11.65.1 Detailed Description . . . . .	1020
11.66dll_pll_veml_tracking.h File Reference . . . . .	1020
11.66.1 Detailed Description . . . . .	1021
11.67dll_pll_veml_tracking_fpga.h File Reference . . . . .	1021
11.67.1 Detailed Description . . . . .	1022
11.68edc.h File Reference . . . . .	1022
11.68.1 Detailed Description . . . . .	1022
11.69exponential_smoother.h File Reference . . . . .	1023
11.69.1 Detailed Description . . . . .	1023
11.70fec.h File Reference . . . . .	1023
11.70.1 Detailed Description . . . . .	1024
11.71fft_base_kernels.h File Reference . . . . .	1024
11.71.1 Detailed Description . . . . .	1024
11.72fft_internal.h File Reference . . . . .	1024
11.72.1 Detailed Description . . . . .	1025
11.73file_configuration.h File Reference . . . . .	1025
11.73.1 Detailed Description . . . . .	1025
11.74file_signal_source.h File Reference . . . . .	1026
11.74.1 Detailed Description . . . . .	1026
11.75fir_filter.h File Reference . . . . .	1026

11.75.1 Detailed Description . . . . .	1027
11.76flexiband_signal_source.h File Reference . . . . .	1027
11.76.1 Detailed Description . . . . .	1027
11.77fmcomms2_signal_source.h File Reference . . . . .	1028
11.77.1 Detailed Description . . . . .	1028
11.78fpga_acquisition.h File Reference . . . . .	1028
11.78.1 Detailed Description . . . . .	1029
11.79fpga_dynamic_bit_selection.h File Reference . . . . .	1029
11.79.1 Detailed Description . . . . .	1029
11.80fpga_multicorrelator.h File Reference . . . . .	1029
11.80.1 Detailed Description . . . . .	1030
11.81fpga_switch.h File Reference . . . . .	1030
11.81.1 Detailed Description . . . . .	1030
11.82freq_xlating_fir_filter.h File Reference . . . . .	1031
11.82.1 Detailed Description . . . . .	1031
11.83front_end_cal.h File Reference . . . . .	1031
11.83.1 Detailed Description . . . . .	1032
11.84galileo_almanac.h File Reference . . . . .	1032
11.84.1 Detailed Description . . . . .	1032
11.85galileo_almanac_helper.h File Reference . . . . .	1032
11.85.1 Detailed Description . . . . .	1033
11.86Galileo_CNAV.h File Reference . . . . .	1033
11.86.1 Detailed Description . . . . .	1034
11.87galileo_cnav_message.h File Reference . . . . .	1035
11.87.1 Detailed Description . . . . .	1035
11.88Galileo_E1.h File Reference . . . . .	1035
11.88.1 Detailed Description . . . . .	1036
11.89galileo_e1_dll_pll_veml_tracking.h File Reference . . . . .	1037
11.89.1 Detailed Description . . . . .	1037
11.90galileo_e1_dll_pll_veml_tracking_fpga.h File Reference . . . . .	1037

11.90.1 Detailed Description . . . . .	1038
11.91galileo_e1_pcps_8ms_ambiguous_acquisition.h File Reference . . . . .	1038
11.91.1 Detailed Description . . . . .	1038
11.92galileo_e1_pcps_ambiguous_acquisition.h File Reference . . . . .	1039
11.92.1 Detailed Description . . . . .	1039
11.93galileo_e1_pcps_ambiguous_acquisition_fpga.h File Reference . . . . .	1039
11.93.1 Detailed Description . . . . .	1040
11.94galileo_e1_pcps_cccwsr_ambiguous_acquisition.h File Reference . . . . .	1040
11.94.1 Detailed Description . . . . .	1040
11.95galileo_e1_pcps_quicksync_ambiguous_acquisition.h File Reference . . . . .	1040
11.95.1 Detailed Description . . . . .	1041
11.96galileo_e1_pcps_tong_ambiguous_acquisition.h File Reference . . . . .	1041
11.96.1 Detailed Description . . . . .	1041
11.97galileo_e1_signal_replica.h File Reference . . . . .	1042
11.97.1 Detailed Description . . . . .	1042
11.98galileo_e1_tcp_connector_tracking.h File Reference . . . . .	1042
11.98.1 Detailed Description . . . . .	1043
11.99galileo_e1_tcp_connector_tracking_cc.h File Reference . . . . .	1043
11.99.1 Detailed Description . . . . .	1044
11.100galileo_e1b_telemetry_decoder.h File Reference . . . . .	1044
11.100.1 Detailed Description . . . . .	1044
11.101galileo_e5_signal_replica.h File Reference . . . . .	1045
11.101.1 Detailed Description . . . . .	1045
11.102galileo_E5a.h File Reference . . . . .	1045
11.102.1 Detailed Description . . . . .	1047
11.103galileo_e5a_dll_pll_tracking.h File Reference . . . . .	1047
11.103.1 Detailed Description . . . . .	1047
11.104galileo_e5a_dll_pll_tracking_fpga.h File Reference . . . . .	1047
11.104.1 Detailed Description . . . . .	1048
11.105galileo_e5a_noncoherent_iq_acquisition_caf.h File Reference . . . . .	1048

11.105. Detailed Description . . . . .	1048
11.106. Galileo_e5a_noncoherent_iq_acquisition_caf_cc.h File Reference . . . . .	1049
11.106. Detailed Description . . . . .	1049
11.107. Galileo_e5a_pcps_acquisition.h File Reference . . . . .	1050
11.107. Detailed Description . . . . .	1050
11.108. Galileo_e5a_pcps_acquisition_fpga.h File Reference . . . . .	1050
11.108. Detailed Description . . . . .	1051
11.109. Galileo_e5a_telemetry_decoder.h File Reference . . . . .	1051
11.109. Detailed Description . . . . .	1051
11.110. Galileo_E5b.h File Reference . . . . .	1052
11.110. Detailed Description . . . . .	1053
11.111. Galileo_e5b_dll_pll_tracking.h File Reference . . . . .	1053
11.111. Detailed Description . . . . .	1053
11.112. Galileo_e5b_pcps_acquisition.h File Reference . . . . .	1054
11.112. Detailed Description . . . . .	1054
11.113. Galileo_e5b_pcps_acquisition_fpga.h File Reference . . . . .	1054
11.113. Detailed Description . . . . .	1055
11.114. Galileo_e5b_telemetry_decoder.h File Reference . . . . .	1055
11.114. Detailed Description . . . . .	1055
11.115. Galileo_E6.h File Reference . . . . .	1056
11.115. Detailed Description . . . . .	1056
11.116. Galileo_e6_dll_pll_tracking.h File Reference . . . . .	1057
11.116. Detailed Description . . . . .	1057
11.117. Galileo_e6_pcps_acquisition.h File Reference . . . . .	1057
11.117. Detailed Description . . . . .	1058
11.118. Galileo_e6_signal_replica.h File Reference . . . . .	1058
11.118. Detailed Description . . . . .	1059
11.119. Galileo_e6_telemetry_decoder.h File Reference . . . . .	1059
11.119. Detailed Description . . . . .	1059
11.120. Galileo_ephemeris.h File Reference . . . . .	1059

11.120. Detailed Description . . . . .	1060
11.120 Galileo_FNAV.h File Reference . . . . .	1060
11.121. Detailed Description . . . . .	1063
11.121 Galileo_fnav_message.h File Reference . . . . .	1063
11.122. Detailed Description . . . . .	1064
11.122 Galileo_has_data.h File Reference . . . . .	1064
11.123. Detailed Description . . . . .	1064
11.123 Galileo_INAV.h File Reference . . . . .	1064
11.124. Detailed Description . . . . .	1068
11.124 Galileo_inav_message.h File Reference . . . . .	1068
11.125. Detailed Description . . . . .	1069
11.125 Galileo_iono.h File Reference . . . . .	1069
11.126. Detailed Description . . . . .	1069
11.126 Galileo_pcps_8ms_acquisition_cc.h File Reference . . . . .	1070
11.127. Detailed Description . . . . .	1070
11.127 Galileo_telemetry_decoder_gs.h File Reference . . . . .	1071
11.128. Detailed Description . . . . .	1071
11.128 Galileo_utc_model.h File Reference . . . . .	1071
11.129. Detailed Description . . . . .	1072
11.129 Gen_signal_source.h File Reference . . . . .	1072
11.130. Detailed Description . . . . .	1072
11.130 Geofunctions.h File Reference . . . . .	1073
11.131. Detailed Description . . . . .	1074
11.131 Geojson_printer.h File Reference . . . . .	1074
11.132. Detailed Description . . . . .	1074
11.132 lonass_gnav_almanac.h File Reference . . . . .	1074
11.133. Detailed Description . . . . .	1075
11.133 lonass_gnav_ephemeris.h File Reference . . . . .	1075
11.134. Detailed Description . . . . .	1076
11.134 lonass_gnav_navigation_message.h File Reference . . . . .	1076

11.135. Detailed Description . . . . .	1076
11.136. lonass_gnav_utc_model.h File Reference . . . . .	1077
11.136. Detailed Description . . . . .	1077
11.137. lonass_l1_ca_dll_pll_c_aid_tracking.h File Reference . . . . .	1077
11.137. Detailed Description . . . . .	1078
11.138. lonass_l1_ca_dll_pll_c_aid_tracking_cc.h File Reference . . . . .	1078
11.138. Detailed Description . . . . .	1079
11.139. lonass_l1_ca_dll_pll_c_aid_tracking_sc.h File Reference . . . . .	1079
11.139. Detailed Description . . . . .	1080
11.140. lonass_l1_ca_dll_pll_tracking.h File Reference . . . . .	1080
11.140. Detailed Description . . . . .	1080
11.141. lonass_l1_ca_dll_pll_tracking_cc.h File Reference . . . . .	1081
11.141. Detailed Description . . . . .	1081
11.142. lonass_l1_ca_pcps_acquisition.h File Reference . . . . .	1082
11.142. Detailed Description . . . . .	1082
11.143. lonass_l1_ca_telemetry_decoder.h File Reference . . . . .	1082
11.143. Detailed Description . . . . .	1083
11.144. lonass_l1_ca_telemetry_decoder_gs.h File Reference . . . . .	1083
11.144. Detailed Description . . . . .	1084
11.145. LONASS_L1_L2_CA.h File Reference . . . . .	1084
11.145. Detailed Description . . . . .	1088
11.146. lonass_l2_ca_dll_pll_c_aid_tracking.h File Reference . . . . .	1088
11.146. Detailed Description . . . . .	1088
11.147. lonass_l2_ca_dll_pll_c_aid_tracking_cc.h File Reference . . . . .	1089
11.147. Detailed Description . . . . .	1089
11.148. lonass_l2_ca_dll_pll_c_aid_tracking_sc.h File Reference . . . . .	1090
11.148. Detailed Description . . . . .	1090
11.149. lonass_l2_ca_dll_pll_tracking.h File Reference . . . . .	1091
11.149. Detailed Description . . . . .	1091
11.150. lonass_l2_ca_dll_pll_tracking_cc.h File Reference . . . . .	1091

11.150. Detailed Description . . . . .	1092
11.151. lonass_l2_ca_pcps_acquisition.h File Reference . . . . .	1092
11.151. Detailed Description . . . . .	1093
11.152. lonass_l2_ca_telemetry_decoder.h File Reference . . . . .	1093
11.152. Detailed Description . . . . .	1093
11.153. lonass_l2_ca_telemetry_decoder_gs.h File Reference . . . . .	1094
11.153. Detailed Description . . . . .	1094
11.154. lonass_l2_signal_replica.h File Reference . . . . .	1095
11.154. Detailed Description . . . . .	1095
11.155. n3s_signal_source.h File Reference . . . . .	1095
11.155. Detailed Description . . . . .	1096
11.156. nmax_signal_source.h File Reference . . . . .	1096
11.156. Detailed Description . . . . .	1096
11.157. nss_block_factory.h File Reference . . . . .	1097
11.157. Detailed Description . . . . .	1097
11.158. nss_block_interface.h File Reference . . . . .	1097
11.158. Detailed Description . . . . .	1098
11.159. nss_circular_deque.h File Reference . . . . .	1098
11.159. Detailed Description . . . . .	1098
11.160. nss_flowgraph.h File Reference . . . . .	1099
11.160. Detailed Description . . . . .	1099
11.161. nss_frequencies.h File Reference . . . . .	1099
11.161. Detailed Description . . . . .	1100
11.162. nss_obs_codes.h File Reference . . . . .	1100
11.162. Detailed Description . . . . .	1103
11.163. nss_satellite.h File Reference . . . . .	1103
11.163. Detailed Description . . . . .	1104
11.164. nss_sdr_create_directory.h File Reference . . . . .	1104
11.164. Detailed Description . . . . .	1104
11.165. nss_sdr_flags.h File Reference . . . . .	1104

11.165. Detailed Description . . . . .	1105
11.166. gnss_sdr_fpga_sample_counter.h File Reference . . . . .	1106
11.166. Detailed Description . . . . .	1106
11.167. gnss_sdr_make_unique.h File Reference . . . . .	1106
11.167. Detailed Description . . . . .	1106
11.168. gnss_sdr_sample_counter.h File Reference . . . . .	1107
11.168. Detailed Description . . . . .	1107
11.169. gnss_sdr_supl_client.h File Reference . . . . .	1108
11.169. Detailed Description . . . . .	1108
11.170. gnss_sdr_time_counter.h File Reference . . . . .	1108
11.170. Detailed Description . . . . .	1109
11.171. gnss_sdr_valve.h File Reference . . . . .	1109
11.171. Detailed Description . . . . .	1110
11.172. gnss_signal.h File Reference . . . . .	1110
11.172. Detailed Description . . . . .	1110
11.173. gnss_signal_replica.h File Reference . . . . .	1110
11.173. Detailed Description . . . . .	1111
11.174. gnss_synchro.h File Reference . . . . .	1111
11.174. Detailed Description . . . . .	1112
11.175. gnss_synchro_monitor.h File Reference . . . . .	1112
11.175. Detailed Description . . . . .	1112
11.176. gnss_synchro_udp_sink.h File Reference . . . . .	1113
11.176. Detailed Description . . . . .	1113
11.177. gnuplot_i.h File Reference . . . . .	1113
11.177. Detailed Description . . . . .	1114
11.178. gps_acq_assist.h File Reference . . . . .	1114
11.178. Detailed Description . . . . .	1115
11.179. gps_almanac.h File Reference . . . . .	1115
11.179. Detailed Description . . . . .	1115
11.180. GPS_CNAV.h File Reference . . . . .	1115

11.180. Detailed Description . . . . .	1118
11.181. <a href="#">gps_cnav_ephemeris.h</a> File Reference . . . . .	1118
11.181. Detailed Description . . . . .	1118
11.182. <a href="#">gps_cnav_iono.h</a> File Reference . . . . .	1118
11.182. Detailed Description . . . . .	1119
11.183. <a href="#">gps_cnav_navigation_message.h</a> File Reference . . . . .	1119
11.183. Detailed Description . . . . .	1119
11.184. <a href="#">gps_cnav_utc_model.h</a> File Reference . . . . .	1120
11.184. Detailed Description . . . . .	1120
11.185. <a href="#">gps_ephemeris.h</a> File Reference . . . . .	1120
11.185. Detailed Description . . . . .	1120
11.186. <a href="#">gps_iono.h</a> File Reference . . . . .	1121
11.186. Detailed Description . . . . .	1121
11.187. <a href="#">GPS_L1_CA.h</a> File Reference . . . . .	1121
11.187. Detailed Description . . . . .	1125
11.188. <a href="#">gps_l1_ca_dll_pll_tracking.h</a> File Reference . . . . .	1125
11.188. Detailed Description . . . . .	1125
11.189. <a href="#">gps_l1_ca_dll_pll_tracking_fpga.h</a> File Reference . . . . .	1125
11.189. Detailed Description . . . . .	1126
11.190. <a href="#">gps_l1_ca_dll_pll_tracking_gpu.h</a> File Reference . . . . .	1126
11.190. Detailed Description . . . . .	1126
11.191. <a href="#">gps_l1_ca_dll_pll_tracking_gpu_cc.h</a> File Reference . . . . .	1127
11.191. Detailed Description . . . . .	1127
11.192. <a href="#">gps_l1_ca_kf_tracking.h</a> File Reference . . . . .	1128
11.192. Detailed Description . . . . .	1128
11.193. <a href="#">gps_l1_ca_kf_tracking_cc.h</a> File Reference . . . . .	1128
11.193. Detailed Description . . . . .	1129
11.194. <a href="#">gps_l1_ca_pcps_acquisition.h</a> File Reference . . . . .	1129
11.194. Detailed Description . . . . .	1130
11.195. <a href="#">gps_l1_ca_pcps_acquisition_fine_doppler.h</a> File Reference . . . . .	1130

11.195. Detailed Description . . . . .	1130
11.196. <a href="#">gps_l1_ca_pcps_acquisition_fpga.h</a> File Reference . . . . .	1131
11.196. Detailed Description . . . . .	1131
11.197. <a href="#">gps_l1_ca_pcps_assisted_acquisition.h</a> File Reference . . . . .	1131
11.197. Detailed Description . . . . .	1132
11.198. <a href="#">gps_l1_ca_pcps_openc1_acquisition.h</a> File Reference . . . . .	1132
11.198. Detailed Description . . . . .	1132
11.199. <a href="#">gps_l1_ca_pcps_quicksync_acquisition.h</a> File Reference . . . . .	1133
11.199. Detailed Description . . . . .	1133
11.200. <a href="#">gps_l1_ca_pcps_tong_acquisition.h</a> File Reference . . . . .	1133
11.200. Detailed Description . . . . .	1134
11.201. <a href="#">gps_l1_ca_tcp_connector_tracking.h</a> File Reference . . . . .	1134
11.201. Detailed Description . . . . .	1134
11.202. <a href="#">gps_l1_ca_tcp_connector_tracking_cc.h</a> File Reference . . . . .	1135
11.202. Detailed Description . . . . .	1135
11.203. <a href="#">gps_l1_ca_telemetry_decoder.h</a> File Reference . . . . .	1136
11.203. Detailed Description . . . . .	1136
11.204. <a href="#">gps_l1_ca_telemetry_decoder_gs.h</a> File Reference . . . . .	1136
11.204. Detailed Description . . . . .	1137
11.205. <a href="#">gps_l2_m_dll_pll_tracking.h</a> File Reference . . . . .	1137
11.205. Detailed Description . . . . .	1137
11.206. <a href="#">gps_l2_m_dll_pll_tracking_fpga.h</a> File Reference . . . . .	1138
11.206. Detailed Description . . . . .	1138
11.207. <a href="#">gps_l2_m_pcps_acquisition.h</a> File Reference . . . . .	1138
11.207. Detailed Description . . . . .	1139
11.208. <a href="#">gps_l2_m_pcps_acquisition_fpga.h</a> File Reference . . . . .	1139
11.208. Detailed Description . . . . .	1139
11.209. <a href="#">GPS_L2C.h</a> File Reference . . . . .	1139
11.209. Detailed Description . . . . .	1140
11.210. <a href="#">gps_l2c_signal_replica.h</a> File Reference . . . . .	1140

11.210. Detailed Description . . . . .	1141
11.211. gps_l2c_telemetry_decoder.h File Reference . . . . .	1141
11.211. Detailed Description . . . . .	1141
11.212. gps_l2c_telemetry_decoder_gs.h File Reference . . . . .	1142
11.212. Detailed Description . . . . .	1142
11.213. GPS_L5.h File Reference . . . . .	1142
11.213. Detailed Description . . . . .	1143
11.214. gps_l5_dll_pll_tracking.h File Reference . . . . .	1144
11.214. Detailed Description . . . . .	1144
11.215. gps_l5_dll_pll_tracking_fpga.h File Reference . . . . .	1144
11.215. Detailed Description . . . . .	1145
11.216. gps_l5_signal_replica.h File Reference . . . . .	1145
11.216. Detailed Description . . . . .	1145
11.217. gps_l5_telemetry_decoder.h File Reference . . . . .	1146
11.217. Detailed Description . . . . .	1146
11.218. gps_l5_telemetry_decoder_gs.h File Reference . . . . .	1146
11.218. Detailed Description . . . . .	1147
11.219. gps_l5i_pcps_acquisition.h File Reference . . . . .	1147
11.219. Detailed Description . . . . .	1147
11.220. gps_l5i_pcps_acquisition_fpga.h File Reference . . . . .	1148
11.220. Detailed Description . . . . .	1148
11.221. gps_navigation_message.h File Reference . . . . .	1148
11.221. Detailed Description . . . . .	1149
11.222. gps_sdr_signal_replica.h File Reference . . . . .	1149
11.222. Detailed Description . . . . .	1149
11.223. gps_utc_model.h File Reference . . . . .	1150
11.223. Detailed Description . . . . .	1150
11.224. gpx_printer.h File Reference . . . . .	1150
11.224. Detailed Description . . . . .	1150
11.225. gr_complex_ip_packet_source.h File Reference . . . . .	1151

11.225. Detailed Description . . . . .	1151
11.226. hybrid_observables.h File Reference . . . . .	1151
11.226. Detailed Description . . . . .	1152
11.227. hybrid_observables_gs.h File Reference . . . . .	1152
11.227. Detailed Description . . . . .	1153
11.228. byte_to_cbyte.h File Reference . . . . .	1153
11.228. Detailed Description . . . . .	1153
11.229. byte_to_complex.h File Reference . . . . .	1153
11.229. Detailed Description . . . . .	1154
11.230. byte_to_cshort.h File Reference . . . . .	1154
11.230. Detailed Description . . . . .	1154
11.231. h_memory_configuration.h File Reference . . . . .	1155
11.231. Detailed Description . . . . .	1155
11.232. hi.h File Reference . . . . .	1155
11.232. Detailed Description . . . . .	1156
11.233. NIReader.h File Reference . . . . .	1156
11.233. Detailed Description . . . . .	1156
11.234. interleaved_byte_to_complex_byte.h File Reference . . . . .	1157
11.234. Detailed Description . . . . .	1157
11.235. interleaved_byte_to_complex_short.h File Reference . . . . .	1157
11.235. Detailed Description . . . . .	1158
11.236. interleaved_short_to_complex_short.h File Reference . . . . .	1158
11.236. Detailed Description . . . . .	1159
11.237. short_to_complex.h File Reference . . . . .	1159
11.237. Detailed Description . . . . .	1159
11.238. short_to_cshort.h File Reference . . . . .	1159
11.238. Detailed Description . . . . .	1160
11.239. sem_type_helpers.h File Reference . . . . .	1160
11.239. Detailed Description . . . . .	1160
11.240. ml_printer.h File Reference . . . . .	1161

11.240. Detailed Description . . . . .	1161
11.241. absat23_source.h File Reference . . . . .	1161
11.241. Detailed Description . . . . .	1162
11.242. absat_signal_source.h File Reference . . . . .	1162
11.242. Detailed Description . . . . .	1162
11.243. back_detectors.h File Reference . . . . .	1163
11.243. Detailed Description . . . . .	1163
11.244. MATH_CONSTANTS.h File Reference . . . . .	1163
11.244. Detailed Description . . . . .	1167
11.245. hmse_resampler_conditioner.h File Reference . . . . .	1167
11.245. Detailed Description . . . . .	1167
11.246. monitor_pvt.h File Reference . . . . .	1167
11.246. Detailed Description . . . . .	1168
11.247. monitor_pvt_udp_sink.h File Reference . . . . .	1168
11.247. Detailed Description . . . . .	1168
11.248. multichannel_file_signal_source.h File Reference . . . . .	1169
11.248. Detailed Description . . . . .	1169
11.249. mea_printer.h File Reference . . . . .	1169
11.249. Detailed Description . . . . .	1170
11.250. nonlinear_tracking.h File Reference . . . . .	1170
11.250. Detailed Description . . . . .	1170
11.251. notch_cc.h File Reference . . . . .	1171
11.251. Detailed Description . . . . .	1171
11.252. notch_filter.h File Reference . . . . .	1171
11.252. Detailed Description . . . . .	1172
11.253. notch_filter_lite.h File Reference . . . . .	1172
11.253. Detailed Description . . . . .	1172
11.254. notch_lite_cc.h File Reference . . . . .	1173
11.254. Detailed Description . . . . .	1173
11.255. sr_file_signal_source.h File Reference . . . . .	1173

11.255. Detailed Description . . . . .	1174
11.256. bs_conf.h File Reference . . . . .	1174
11.256. Detailed Description . . . . .	1174
11.257. bsdiff_flags.h File Reference . . . . .	1175
11.257. Detailed Description . . . . .	1175
11.258. observable_tests_flags.h File Reference . . . . .	1175
11.258. Detailed Description . . . . .	1176
11.259. observables_dump_reader.h File Reference . . . . .	1176
11.259. Detailed Description . . . . .	1176
11.260. observables_interface.h File Reference . . . . .	1177
11.260. Detailed Description . . . . .	1177
11.261. smosdr_signal_source.h File Reference . . . . .	1177
11.261. Detailed Description . . . . .	1178
11.262. pass_through.h File Reference . . . . .	1178
11.262. Detailed Description . . . . .	1178
11.263. pcps_acquisition.h File Reference . . . . .	1179
11.263. Detailed Description . . . . .	1179
11.264. pcps_acquisition_fine_doppler_cc.h File Reference . . . . .	1180
11.264. Detailed Description . . . . .	1181
11.265. pcps_acquisition_fpga.h File Reference . . . . .	1181
11.265. Detailed Description . . . . .	1182
11.266. pcps_assisted_acquisition_cc.h File Reference . . . . .	1182
11.266. Detailed Description . . . . .	1183
11.267. pcps_cccwsr_acquisition_cc.h File Reference . . . . .	1183
11.267. Detailed Description . . . . .	1184
11.268. pcps_openccl_acquisition_cc.h File Reference . . . . .	1184
11.268. Detailed Description . . . . .	1185
11.269. pcps_quicksync_acquisition_cc.h File Reference . . . . .	1185
11.269. Detailed Description . . . . .	1186
11.270. pcps_tong_acquisition_cc.h File Reference . . . . .	1187

11.270. Detailed Description . . . . .	1187
11.271. plutosdr_signal_source.h File Reference . . . . .	1188
11.271. Detailed Description . . . . .	1188
11.272. position_test_flags.h File Reference . . . . .	1189
11.272. Detailed Description . . . . .	1189
11.273. pulse_blanking_cc.h File Reference . . . . .	1190
11.273. Detailed Description . . . . .	1190
11.274. pulse_blanking_filter.h File Reference . . . . .	1190
11.274. Detailed Description . . . . .	1191
11.275. pvt_conf.h File Reference . . . . .	1191
11.275. Detailed Description . . . . .	1191
11.276. pvt_interface.h File Reference . . . . .	1191
11.276. Detailed Description . . . . .	1192
11.277. pvt_solution.h File Reference . . . . .	1192
11.277. Detailed Description . . . . .	1192
11.278. raw_array_signal_source.h File Reference . . . . .	1193
11.278. Detailed Description . . . . .	1193
11.279. rex_printer.h File Reference . . . . .	1193
11.279. Detailed Description . . . . .	1194
11.280. tcm.h File Reference . . . . .	1194
11.280. Detailed Description . . . . .	1195
11.281. tcm_printer.h File Reference . . . . .	1195
11.281. Detailed Description . . . . .	1195
11.282. tklib.h File Reference . . . . .	1196
11.282. Detailed Description . . . . .	1205
11.283. tklib_conversions.h File Reference . . . . .	1205
11.283. Detailed Description . . . . .	1206
11.284. tklib_ephemeris.h File Reference . . . . .	1206
11.284. Detailed Description . . . . .	1207
11.285. tklib_ionex.h File Reference . . . . .	1207

11.285.1 Detailed Description . . . . .	1208
11.286.1 <del>klib_lambda.h</del> File Reference . . . . .	1208
11.286.1 Detailed Description . . . . .	1209
11.286.2 Macro Definition Documentation . . . . .	1209
11.286.2.1 SWAP_LAMBDA . . . . .	1210
11.287.1 <del>klib_pntpos.h</del> File Reference . . . . .	1210
11.287.1 bf estimated parameters . . . . .	1211
11.287.2 Detailed Description . . . . .	1211
11.287.3 Function Documentation . . . . .	1211
11.287.3.1 pntpos() . . . . .	1212
11.287.4 Variable Documentation . . . . .	1212
11.287.4.1 ERR_ION . . . . .	1212
11.287.4.2 ERR_TROP . . . . .	1212
11.287.4.3 MAXITR . . . . .	1212
11.287.4.4 NX . . . . .	1213
11.287.5 f estimated parameters . . . . .	1213
11.288.1 <del>klib_ppp.h</del> File Reference . . . . .	1213
11.288.1 Detailed Description . . . . .	1214
11.288.2 Macro Definition Documentation . . . . .	1215
11.288.2.1 SWAP_D . . . . .	1215
11.288.2.2 SWAP_I . . . . .	1215
11.289.1 <del>klib_preceph.h</del> File Reference . . . . .	1215
11.289.1 Detailed Description . . . . .	1216
11.290.1 <del>klib_pvt.h</del> File Reference . . . . .	1217
11.290.1 Detailed Description . . . . .	1217
11.291.1 <del>klib_pvt_gs.h</del> File Reference . . . . .	1217
11.291.1 Detailed Description . . . . .	1218
11.292.1 <del>klib_rtcn.h</del> File Reference . . . . .	1218
11.292.1 Detailed Description . . . . .	1219
11.293.1 <del>klib_rtcn2.h</del> File Reference . . . . .	1219

11.293.1 Detailed Description . . . . .	1220
11.294.1 <del>klib_rtc3.h</del> File Reference . . . . .	1220
11.294.1 Detailed Description . . . . .	1222
11.294.2 <del>/variable</del> Documentation . . . . .	1222
11.294.2.1 CODES_BDS . . . . .	1222
11.294.2.2 CODES_GAL . . . . .	1223
11.294.2.3 CODES_GLO . . . . .	1223
11.294.2.4 CODES_GPS . . . . .	1223
11.294.2.5 CODES_QZS . . . . .	1224
11.294.2.6 CODES_SBS . . . . .	1224
11.294.2.7 SSRUDINT . . . . .	1224
11.295.1 <del>klib_rtkcmn.h</del> File Reference . . . . .	1224
11.295.1 Detailed Description . . . . .	1227
11.295.2 Macro Definition Documentation . . . . .	1228
11.295.2.1 Rx . . . . .	1228
11.295.2.2 Ry . . . . .	1229
11.295.2.3 Rz . . . . .	1229
11.296.1 <del>klib_rtkpos.h</del> File Reference . . . . .	1229
11.296.1 Detailed Description . . . . .	1231
11.297.1 <del>klib_rtksvr.h</del> File Reference . . . . .	1231
11.297.1 Detailed Description . . . . .	1232
11.297.2 <del>/variable</del> Documentation . . . . .	1233
11.297.2.1 PRCOPT_DEFAULT . . . . .	1233
11.297.2.2 SOLOPT_DEFAULT . . . . .	1233
11.298.1 <del>klib_sbas.h</del> File Reference . . . . .	1233
11.298.1 Detailed Description . . . . .	1234
11.298.2 <del>/variable</del> Documentation . . . . .	1235
11.298.2.1 IGPBAND1 . . . . .	1235
11.298.2.2 IGPBAND2 . . . . .	1236
11.299.1 <del>klib_solution.h</del> File Reference . . . . .	1236

11.299. Detailed Description . . . . .	1237
11.300. <a href="#">tklib_solver.h</a> File Reference . . . . .	1238
11.300. Detailed Description . . . . .	1238
11.301. <a href="#">tklib_solver_dump_reader.h</a> File Reference . . . . .	1239
11.301. Detailed Description . . . . .	1239
11.302. <a href="#">tklib_stream.h</a> File Reference . . . . .	1239
11.302. Detailed Description . . . . .	1241
11.303. <a href="#">tklib_tides.h</a> File Reference . . . . .	1242
11.303. Detailed Description . . . . .	1242
11.304. <a href="#">tkl_tcp_commands.h</a> File Reference . . . . .	1243
11.304. Detailed Description . . . . .	1243
11.305. <a href="#">tkl_tcp_dongle_info.h</a> File Reference . . . . .	1243
11.305. Detailed Description . . . . .	1244
11.306. <a href="#">tkl_tcp_signal_source.h</a> File Reference . . . . .	1244
11.306. Detailed Description . . . . .	1244
11.307. <a href="#">tkl_tcp_signal_source_c.h</a> File Reference . . . . .	1245
11.307. Detailed Description . . . . .	1245
11.308. <a href="#">tkbas_ephemeris.h</a> File Reference . . . . .	1246
11.308. Detailed Description . . . . .	1246
11.309. <a href="#">tkbas_l1_telemetry_decoder.h</a> File Reference . . . . .	1246
11.309. Detailed Description . . . . .	1247
11.310. <a href="#">tkbas_l1_telemetry_decoder_gs.h</a> File Reference . . . . .	1247
11.310. Detailed Description . . . . .	1248
11.311. <a href="#">tkdes_gnss_synchro.h</a> File Reference . . . . .	1248
11.311. Detailed Description . . . . .	1248
11.312. <a href="#">tkdes_monitor_pvt.h</a> File Reference . . . . .	1248
11.312. Detailed Description . . . . .	1249
11.313. <a href="#">tkshort_x2_to_cshort.h</a> File Reference . . . . .	1249
11.313. Detailed Description . . . . .	1249
11.314. <a href="#">tksignal_conditioner.h</a> File Reference . . . . .	1250

11.314. Detailed Description . . . . .	1250
11.315. signal_generator.h File Reference . . . . .	1250
11.315. Detailed Description . . . . .	1251
11.316. signal_generator_c.h File Reference . . . . .	1251
11.316. Detailed Description . . . . .	1251
11.316.2. Function Documentation . . . . .	1252
11.316.2.1. signal_make_generator_c() . . . . .	1252
11.317. signal_generator_flags.h File Reference . . . . .	1252
11.317. Detailed Description . . . . .	1253
11.318. pir_file_signal_source.h File Reference . . . . .	1253
11.318. Detailed Description . . . . .	1253
11.319. pir_gss6450_file_signal_source.h File Reference . . . . .	1254
11.319. Detailed Description . . . . .	1254
11.320. parent_motion_csv_dump_reader.h File Reference . . . . .	1254
11.320. Detailed Description . . . . .	1255
11.321. string_converter.h File Reference . . . . .	1255
11.321. Detailed Description . . . . .	1255
11.322. swift_common.h File Reference . . . . .	1255
11.322. Detailed Description . . . . .	1256
11.323. tcp_cmd_interface.h File Reference . . . . .	1256
11.323. Detailed Description . . . . .	1257
11.324. tcp_communication.h File Reference . . . . .	1257
11.324. Detailed Description . . . . .	1257
11.325. tcp_packet_data.h File Reference . . . . .	1258
11.325. Detailed Description . . . . .	1258
11.326. telemetry_decoder_interface.h File Reference . . . . .	1258
11.326. Detailed Description . . . . .	1258
11.327. test_flags.h File Reference . . . . .	1259
11.327. Detailed Description . . . . .	1259
11.328. tm_conf.h File Reference . . . . .	1259

11.328. Detailed Description . . . . .	1259
11.329. m_dump_reader.h File Reference . . . . .	1260
11.329. Detailed Description . . . . .	1260
11.330. m_utils.h File Reference . . . . .	1260
11.330. Detailed Description . . . . .	1260
11.331. tracking_2nd_DLL_filter.h File Reference . . . . .	1261
11.331. Detailed Description . . . . .	1261
11.332. tracking_2nd_PLL_filter.h File Reference . . . . .	1261
11.332. Detailed Description . . . . .	1261
11.333. tracking_discriminators.h File Reference . . . . .	1262
11.333. Detailed Description . . . . .	1262
11.334. tracking_dump_reader.h File Reference . . . . .	1263
11.334. Detailed Description . . . . .	1263
11.335. tracking_FLL_PLL_filter.h File Reference . . . . .	1263
11.335. Detailed Description . . . . .	1263
11.336. tracking_interface.h File Reference . . . . .	1264
11.336. Detailed Description . . . . .	1264
11.337. tracking_loop_filter.h File Reference . . . . .	1264
11.337. Detailed Description . . . . .	1265
11.338. tracking_tests_flags.h File Reference . . . . .	1265
11.338. Detailed Description . . . . .	1266
11.339. tracking_true_obs_reader.h File Reference . . . . .	1266
11.339. Detailed Description . . . . .	1267
11.340. true_observables_reader.h File Reference . . . . .	1267
11.340. Detailed Description . . . . .	1267
11.341. two_bit_cpx_file_signal_source.h File Reference . . . . .	1267
11.341. Detailed Description . . . . .	1268
11.342. two_bit_packed_file_signal_source.h File Reference . . . . .	1268
11.342. Detailed Description . . . . .	1268
11.343. xhd_signal_source.h File Reference . . . . .	1269

11.343. Detailed Description . . . . .	1269
11.344. io_fpga.h File Reference . . . . .	1269
11.344. Detailed Description . . . . .	1270
11.345. npack_2bit_samples.h File Reference . . . . .	1270
11.345. Detailed Description . . . . .	1271
11.346. npack_byte_2bit_cpx_samples.h File Reference . . . . .	1271
11.346. Detailed Description . . . . .	1272
11.347. npack_byte_2bit_samples.h File Reference . . . . .	1272
11.347. Detailed Description . . . . .	1273
11.348. npack_byte_4bit_samples.h File Reference . . . . .	1273
11.348. Detailed Description . . . . .	1273
11.349. npack_intspir_1bit_samples.h File Reference . . . . .	1274
11.349. Detailed Description . . . . .	1274
11.350. npack_spir_gss6450_samples.h File Reference . . . . .	1274
11.350. Detailed Description . . . . .	1275
11.351. viterbi_decoder.h File Reference . . . . .	1275
11.351. Detailed Description . . . . .	1275
<b>Index</b>	<b>1277</b>

# Chapter 1

## Main Page



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.

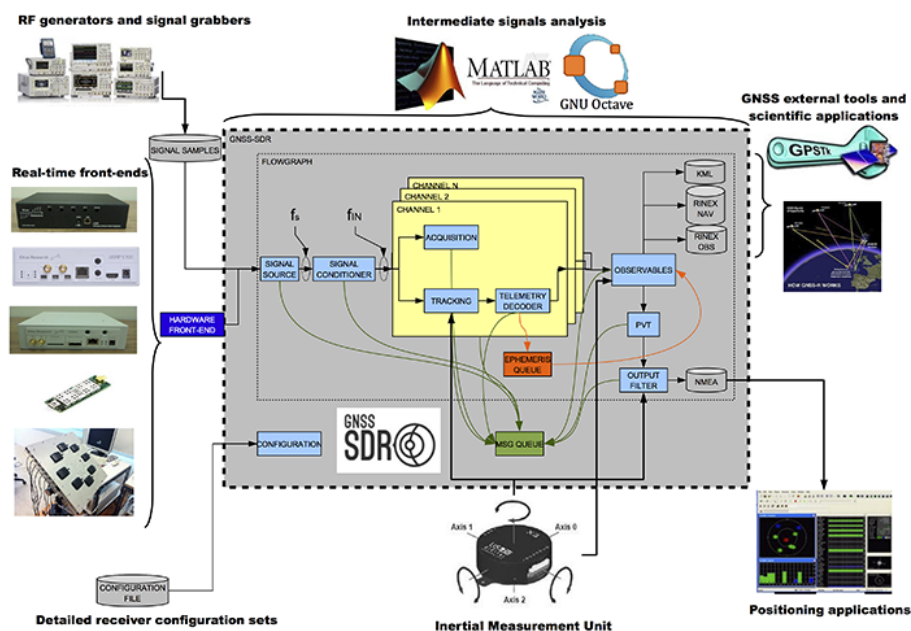


Figure 1.2 Overview

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.
- The [GN3S v2 USB dongle](#) (GN3S v3 might work with small modifications).
- 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
|---build      <- where gnss-sdr is built
|---cmake      <- CMake-related files
|---conf       <- Configuration files. Each file represents one receiver.
|---data       <- Populate this folder with your captured data.
|---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/build
$ cmake ..
$ make
```

If everything goes well, three new executables will be created at `gnss-sdr/install`, namely `gnss-sdr`, `volk_gnssssdr_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 `gnss-sdr/build` 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. From the `gnss-sdr/build` 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-get 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/build
$ cmake -DCMAKE_BUILD_TYPE=Debug ..
$ make
$ sudo make install
```

### 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 gnss-sdr/build
$ sudo make uninstall
$ 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 with 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=../data/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.

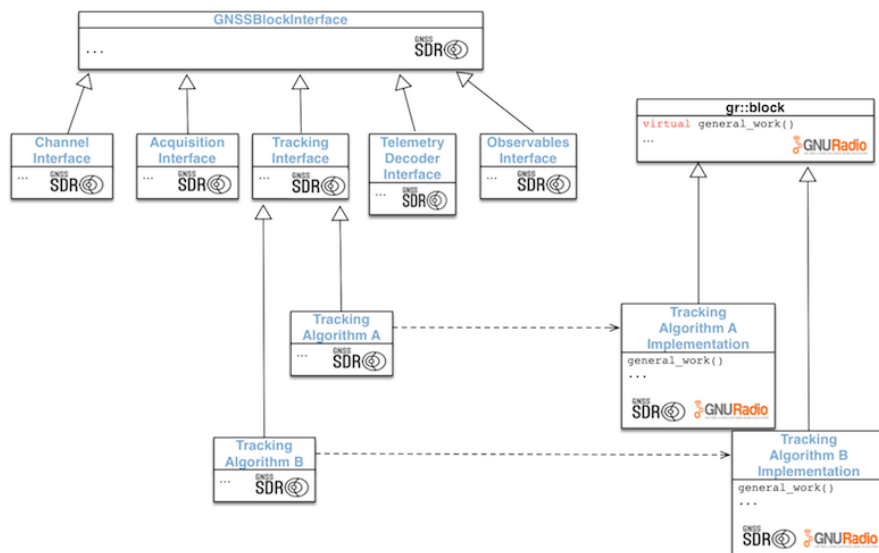


Figure 1.3 Class hierarchy of signal processing blocks

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 run-time functions used during the execution of the program: `run()`, `start()`, `stop()`, `wait()`, etc. 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 [GpsL1CaPcpsAcquisition](#) and [GalileoE1PcpsAmbiguousAcquisition](#) for examples of adapters from a Parallel Code Phase Search (PCPS) acquisition block, and `pcps_acquisition_cc` 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 [GpsL1CaDlIPllTracking](#) or [GalileoE1DlIPllVemlTracking](#) for examples of adapters, and [Gps\\_L1\\_Ca\\_Dll\\_Pll\\_Tracking\\_cc](#) 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 channel
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 and 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 [GpsL1CaTelemetryDecoder](#) for an example of the GPS L1 NAV message decoding adapter, and `gps_l1_ca_telemetry_decoder_cc` 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

Although data processing for obtaining high-accuracy PVT solutions is out of the scope of GNSS-SDR, we provide a module that can compute a simple least square solution and leaves room for more sophisticated positioning methods. The integration with libraries and software tools that are able to deal with multi-constellation data such as [GPSTk](#) or [gLAB](#) appears as a viable solution for high performance, completely customizable GNSS receivers.

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.iono_model=Broadcast             ; options: OFF, Broadcast
PVT.trop_model=Saastamoinen         ; options: OFF, Saastamoinen
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 or disables a TCP/IP server dispatching RTCM messages
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

```

## 1.7 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](#)), for the GN3S v2 USB dongle 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](https://www.gps.gov), the official U.S. Government webpage for GPS.

- GPS L1 and L2C: Global Positioning System Directorate, **Interface Specification IS-GPS-200 Revision K**. March, 2019.
- GPS L1C (available with first Block III launch): Global Positioning System Directorate, **Interface Specification IS-GPS-800 Revision F**. March, 2019.
- GPS L5 (first Block IIF satellite launched on May, 2010): Global Positioning System Directorate, **Interface Specification IS-GPS-705 Revision F**. March, 2019.

#### 2.1.2 GLONASS

Official GLONASS webpage: [Information-analytical centre official website](https://www.glonass-iac.ru/).

- 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 1.3**, European Commission, Dec. 2016.

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](http://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 DO-229D - 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++20:** Check the [C++ standard draft sources at GitHub](#).
- **C++17:** The current ISO C++ standard is 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:** The 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 \(ITU-R\)](https://www.itu.int/en/ITU-R/Pages/default.aspx?target=_blank) in the global standard for international mobile telecommunications (IMT Advanced, also referred to as 4G) access technologies.

Control plane protocols:

- Radio Resource LCS Protocol (RRLP): [3GPP Technical Specification 44.031](#).

- LTE Positioning Protocol (LPP): [3GPP Technical Specification 36.355](#).

User plane protocols:

- Open Mobile Alliance (OMA), [Secure User Plane Location Architecture Version 1 \(SUPL 1.0\)](#), June 2007.
- Open Mobile Alliance (OMA), [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 [OMA Specifications webpage](#) for updated information about LPP Extensions (LPPE) Specification.

- The [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:

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

where  $(x_i^{\text{Tx}}, y_i^{\text{Tx}}, z_i^{\text{Tx}})$  is the satellite's position (known from the navigation message),  $(x, y, z)$  the receiver's position, and  $\sigma_e$  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

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

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 K**, this band is centered at  $f_{\text{GPS L1}} = 1575.42$  MHz. The complex baseband transmitted signal can be written as

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

with

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

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

where  $\oplus$  is the exclusive-or operation (modulo-2 addition),  $|l|_L$  means  $l$  modulo  $L$ ,  $[l]_L$  means the integer part of  $\frac{l}{L}$ ,  $D_{\text{NAV}}$  is the GPS navigation message bit sequence, transmitted at 50 bps,  $T_{c,P(Y)} = \frac{1}{10.23} \mu\text{s}$ ,  $T_{c,C/A} = \frac{1}{1.023} \mu\text{s}$ ,  $L_{P(Y)} = 6.1871 \cdot 10^{12}$ , 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 F**, to be broadcast on the same L1 frequency that currently contains the C/A signal. The L1C will be available with first Block III launch, currently scheduled for 2013. The implementation will provide C/A code to ensure backward compatibility.

- **GPS L2C.** Defined at **Interface Specification IS-GPS-200 Revision K**, 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  $D_{\text{NAV}}$ . For the Quadrature-phase component, three options are defined:

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

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

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

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

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  $C_{\text{CL}}$  is  $L_{\text{CL}} = 767250$  chips long, repeating every 1.5 s, while the civilian moderate code  $C_{\text{CM}}$  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 F**, 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:

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

$$e_{L5I}(t) = \sum_{m=-\infty}^{+\infty} C_{nh10}[m]_{10} \oplus D_{\text{CNAV}}[m]_{10} \oplus \sum_{l=1}^{102300} C_{L5I}[l]_{10230} p(t - mT_{c,nh} - lT_{c,L5}), \quad (3.11)$$

$$e_{L5Q}(t) = \sum_{m=-\infty}^{+\infty} C_{nh20}[m]_{20} \oplus \sum_{l=1}^{102300} C_{L5Q}[l]_{10230} \cdot p(t - mT_{c,nh} - lT_{c,L5}), \quad (3.12)$$

where  $T_{c,nh} = 1$  ms and  $T_{c,L5} = \frac{1}{10.23}$   $\mu$ s, thus defining a BPSK(10) modulation. Both L5I and L5Q contain synchronization sequences. {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  $64.8^\circ$  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  $120^\circ$  right ascension of the ascending node. Eight satellites are equally spaced in each plane with  $45^\circ$  argument of latitude. Moreover, the orbital planes have an argument of latitude displacement of  $15^\circ$  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:  $f_{\text{GLOL1}}^{(k)} = 1602 + k \cdot 0.5625$  MHz and  $f_{\text{GLOL2}}^{(k)} = 1246 + k \cdot 0.4375$  MHz, where  $k \in \{-7, -6, \dots, 5, 6\}$  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

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

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

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

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

where  $T_{c,\text{HP}} = \frac{1}{5.11} \mu\text{s}$ ,  $T_{c,\text{SP}} = \frac{1}{0.511} \mu\text{s}$ , and  $L_{\text{HP}} = 3.3554 \cdot 10^7$ . The navigation message  $D_{\text{GNAV}}$  is transmitted at 50 bps. Details of its content and structure, as well as the generation of the  $C_{\text{SP}}$  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  $56^\circ$  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 1.3](#), 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:

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

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

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

$$sc_A(t) = \text{sign} \left( \sin(2\pi f_{s,E1A} t) \right), \quad (3.18)$$

$$sc_B(t) = \text{sign} \left( \sin(2\pi f_{s,E1B} t) \right), \quad (3.19)$$

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,  $D_{I/NAV}$ , intended for Safety-of-Life (SoL) services:

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

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

$$e_{E1C}(t) = \sum_{m=-\infty}^{+\infty} C_{E1Cs} \left[ [m]_{25} \right] \oplus \sum_{l=1}^{4092} C_{E1Cp} \left[ l \right] \cdot p(t - mT_{c,E1Cs} - lT_{c,E1Cp}), \quad (3.21)$$

with  $T_{c,E1B} = T_{c,E1Cp} = \frac{1}{1.023} \mu\text{s}$  and  $T_{c,E1Cs} = 4$  ms. The  $C_{E1B}$  and  $C_{E1Cp}$  primary codes are pseudorandom memory code sequences defined at Annex C.7 and C.8 of OS SIS ICD. The binary sequence of the secondary code  $C_{E1Cs}$  is 0011100000001010110110010. 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  $T_{c,E1A} = \frac{1}{2.5575} \mu\text{s}$ . 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

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

$$e_{E6B}(t) = \sum_{m=-\infty}^{+\infty} D_{C/NAV} \left[ [l]_{5115} \right] \oplus C_{E6B} \left[ [l]_{L_{E6B}} \right] \cdot p(t - lT_{c,E6}), \quad (3.23)$$

$$e_{E6C}(t) = \sum_{m=-\infty}^{+\infty} C_{E6Cs} \left[ [m]_{100} \right] \oplus \sum_{l=1}^{L_{E6C}} C_{E6Cp} \left[ l \right] \cdot p(t - mT_{c,E6s} - lT_{c,E6p}), \quad (3.24)$$

where  $D_{C/NAV}$  is the C/NAV navigation data stream, which is modulated with the encrypted ranging code  $C_{E6B}$  with chip period  $T_{c,E6} = \frac{1}{5.115} \mu\text{s}$ , 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,  $C_{E6Cp}$ , 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

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

where  $v(t) = \frac{1}{\sqrt{2}} (\text{sign}(\cos(2\pi f_s t)) + j \text{sign}(\sin(2\pi f_s t)))$  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:

$$s_T^{(\text{Gal E5})}(t) = e_{E5a}(t)ss_c^*(t) + e_{E5b}(t)ss_c(t) + \bar{e}_{E5a}(t)ss_c^*(t) + \bar{e}_{E5b}(t)ss_c(t), \quad (3.26)$$

where the single and product side-band signal subcarriers are

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

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

and

$$e_{E5a}(t) = e_{E5aI}(t) + je_{E5aQ}(t), \quad (3.29)$$

$$e_{E5b}(t) = e_{E5bI}(t) + je_{E5bQ}(t), \quad (3.30)$$

$$\bar{e}_{E5a}(t) = \bar{e}_{E5aI}(t) + j\bar{e}_{E5aQ}(t), \quad (3.31)$$

$$\bar{e}_{E5b}(t) = \bar{e}_{E5bI}(t) + j\bar{e}_{E5bQ}(t), \quad (3.32)$$

$$\bar{e}_{E5aI}(t) = e_{E5aQ}(t)e_{E5bI}(t)e_{E5bQ}(t), \quad (3.33)$$

$$\bar{e}_{E5aQ}(t) = e_{E5aI}(t)e_{E5bI}(t)e_{E5bQ}(t), \quad (3.34)$$

$$\bar{e}_{E5bI}(t) = e_{E5bQ}(t)e_{E5aI}(t)e_{E5aQ}(t), \quad (3.35)$$

$$\bar{e}_{E5bQ}(t) = e_{E5bI}(t)e_{E5aI}(t)e_{E5aQ}(t). \quad (3.36)$$

The signal components are defined as

$$e_{E5aI}(t) = \sum_{m=-\infty}^{+\infty} C_{E5aIs} \left[ |m|_{20} \right] \oplus \sum_{l=1}^{10230} C_{E5aIp} \left[ l \right] \oplus \quad (3.37)$$

$$\oplus D_{F/NAV} \left[ [l]_{204600} \right] p(t - mT_{c,E5s} - lT_{c,E5p}), \quad (3.38)$$

$$e_{E5aQ}(t) = \sum_{m=-\infty}^{+\infty} C_{E5aQs} \left[ |m|_{100} \right] \oplus \sum_{l=1}^{10230} C_{E5aQp} \left[ l \right] \cdot \quad (3.39)$$

$$\cdot p(t - mT_{c,E5s} - lT_{c,E5p}), \quad (3.40)$$

$$e_{E5bI}(t) = \sum_{m=-\infty}^{+\infty} C_{E5bIs} \left[ |m|_4 \right] \oplus \sum_{l=1}^{10230} C_{E5bIp} \left[ l \right] \oplus \quad (3.41)$$

$$\oplus D_{I/NAV} \left[ [l]_{40920} \right] p(t - mT_{c,E5s} - lT_{c,E5p}), \quad (3.42)$$

$$e_{E5bQ}(t) = \sum_{m=-\infty}^{+\infty} C_{E5bQs} \left[ |m|_{100} \right] \oplus \sum_{l=1}^{10230} C_{E5bQp} \left[ l \right] \cdot \quad (3.43)$$

$$\cdot p(t - mT_{c,E5s} - lT_{c,E5p}), \quad (3.44)$$

where  $T_{c,E5s} = 1$  ms and  $T_{c,E5p} = \frac{1}{10.23} \mu\text{s}$ . **Channel A** contains the F/NAV type of navigation message,  $D_{F/NAV}$ , 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:

$$sc_s(t) = \frac{\sqrt{2}}{4} \text{sign} \left( \cos \left( 2\pi f_s t - \frac{\pi}{4} \right) \right) + \quad (3.45)$$

$$+ \frac{1}{2} \text{sign} \left( \cos (2\pi f_s t) \right) + \frac{\sqrt{2}}{4} \text{sign} \left( \cos \left( 2\pi f_s t + \frac{\pi}{4} \right) \right), \quad (3.46)$$

$$sc_p(t) = -\frac{\sqrt{2}}{4} \text{sign} \left( \cos \left( 2\pi f_s t - \frac{\pi}{4} \right) \right) + \quad (3.47)$$

$$+ \frac{1}{2} \text{sign} \left( \cos (2\pi f_s t) \right) - \frac{\sqrt{2}}{4} \text{sign} \left( \cos \left( 2\pi f_s t + \frac{\pi}{4} \right) \right), \quad (3.48)$$

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  $\bar{e}_{E5b}(t)$ :

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

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. Falletti, *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/JPR↔OC.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

# Module Index

### 5.1 Modules

Here is a list of all modules:

Acquisition . . . . .	71
acquisition_adapters . . . . .	72
acquisition_gr_blocks . . . . .	74
acquisition_libs . . . . .	76
Channel . . . . .	77
channel_adapters . . . . .	78
channel_libs . . . . .	79
Signal Conditioner . . . . .	80
conditioner_adapters . . . . .	81
Data Type Adapters . . . . .	82
data_type_adapters . . . . .	83
data_type_gr_blocks . . . . .	84
Input Filter . . . . .	85
input_filter_adapters . . . . .	86
input_filter_gr_blocks . . . . .	87
Algorithms Common Library . . . . .	88
algorithms_libs . . . . .	89
gnss_sdr_flags . . . . .	113
PVT . . . . .	117
algorithms_libs_rtklib . . . . .	118
pvt_adapters . . . . .	163
pvt_gr_blocks . . . . .	164
pvt_libs . . . . .	165
Observables . . . . .	159
obs_adapters . . . . .	160
obs_gr_blocks . . . . .	161
observables_libs . . . . .	162
Resampler . . . . .	166
resampler_adapters . . . . .	167
resampler_gr_blocks . . . . .	168
Signal Source . . . . .	169
signal_source_adapters . . . . .	170

signal_source_gr_blocks . . . . .	171
signal_source_libs . . . . .	172
Telemetry Decoder . . . . .	174
telemetry_decoder_adapters . . . . .	175
telemetry_decoder_gr_blocks . . . . .	176
telemetry_decoder_libs . . . . .	178
telemetry_decoder_libswiftcnv . . . . .	182
Tracking . . . . .	184
tracking_adapters . . . . .	185
tracking_gr_blocks . . . . .	186
tracking_libs . . . . .	188
Core GNSS Receiver . . . . .	193
GNSS block interfaces . . . . .	194
core_libs . . . . .	195
core_monitor . . . . .	197
core_receiver . . . . .	198
core_system_parameters . . . . .	199

## Chapter 6

# Hierarchical Index

### 6.1 Class Hierarchy

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

Acq_Conf . . . . .	293
Acquisition_Dump_Reader . . . . .	294
Agnss_Ref_Location . . . . .	299
Agnss_Ref_Time . . . . .	300
alm_t . . . . .	302
ambc_t . . . . .	302
Bayesian_estimator . . . . .	304
Beidou_Dnav_Almanac . . . . .	311
Beidou_Dnav_Ephemeris . . . . .	314
Beidou_Dnav_Iono . . . . .	330
Beidou_Dnav_Navigation_Message . . . . .	333
Beidou_Dnav_Utc_Model . . . . .	337
block	
Acquisition_msg_rx . . . . .	295
beidou_b1i_telemetry_decoder_gs . . . . .	307
beidou_b3i_telemetry_decoder_gs . . . . .	309
channel_msg_receiver_cc . . . . .	361
channel_status_msg_receiver . . . . .	362
direct_resampler_conditioner_cb . . . . .	386
direct_resampler_conditioner_cc . . . . .	387
direct_resampler_conditioner_cs . . . . .	387
dll_pll_veml_tracking . . . . .	392
dll_pll_veml_tracking_fpga . . . . .	393
Galileo_E1_Tcp_Connector_Tracking_cc . . . . .	427
galileo_e5a_noncoherentIQ_acquisition_caf_cc . . . . .	428
galileo_pcps_8ms_acquisition_cc . . . . .	452
galileo_telemetry_decoder_gs . . . . .	458
glonass_l1_ca_dll_pll_c_aid_tracking_cc . . . . .	579
glonass_l1_ca_dll_pll_c_aid_tracking_sc . . . . .	580
Glonass_L1_Ca_Dll_Pll_Tracking_cc . . . . .	581
glonass_l1_ca_telemetry_decoder_gs . . . . .	582
glonass_l2_ca_dll_pll_c_aid_tracking_cc . . . . .	584
glonass_l2_ca_dll_pll_c_aid_tracking_sc . . . . .	585
Glonass_L2_Ca_Dll_Pll_Tracking_cc . . . . .	585
glonass_l2_ca_telemetry_decoder_gs . . . . .	586

gnss_sdr_fpga_sample_counter . . . . .	613
gnss_sdr_time_counter . . . . .	622
gnss_synchro_monitor . . . . .	635
Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc . . . . .	697
Gps_L1_Ca_Kf_Tracking_cc . . . . .	697
Gps_L1_Ca_Tcp_Connector_Tracking_cc . . . . .	698
gps_l1_ca_telemetry_decoder_gs . . . . .	699
gps_l2c_telemetry_decoder_gs . . . . .	701
gps_l5_telemetry_decoder_gs . . . . .	702
hybrid_observables_gs . . . . .	789
labsat23_source . . . . .	801
Notch . . . . .	812
NotchLite . . . . .	815
pcps_acquisition . . . . .	823
pcps_acquisition_fine_doppler_cc . . . . .	827
pcps_assisted_acquisition_cc . . . . .	839
pcps_cccwsr_acquisition_cc . . . . .	844
pcps_openc1_acquisition_cc . . . . .	848
pcps_quicksync_acquisition_cc . . . . .	854
pcps_tong_acquisition_cc . . . . .	859
pulse_blanking_cc . . . . .	868
sbas_l1_telemetry_decoder_gs . . . . .	918
signal_generator_c . . . . .	931
Channel_Event . . . . .	360
ChannelFsm . . . . .	363
cl_fft_plan . . . . .	365
clFFT_Complex . . . . .	365
clFFT_Dim3 . . . . .	366
clFFT_SplitComplex . . . . .	366
cnav_msg_decoder_t . . . . .	366
cnav_msg_t . . . . .	367
cnav_v27_part_t . . . . .	369
Command_Event . . . . .	372
Concurrent_Map< Data > . . . . .	374
Concurrent_Queue< Data > . . . . .	375
Concurrent_Queue< pmt::pmt_t > . . . . .	375
ConfigurationInterface . . . . .	375
FileConfiguration . . . . .	400
InMemoryConfiguration . . . . .	795
ControlThread . . . . .	378
Cpu_Multicorrelator . . . . .	381
Cpu_Multicorrelator_16sc . . . . .	381
Cpu_Multicorrelator_Real_Codes . . . . .	382
CubatureFilter . . . . .	383
cuda_multicorrelator . . . . .	384
dgps_t . . . . .	385
Dll_Pll_Conf . . . . .	389
Dll_Pll_Conf_Fpga . . . . .	390
eph_t . . . . .	395
erp_t . . . . .	396
erpd_t . . . . .	396
std::exception . . . . .	
std::runtime_error . . . . .	
GnuplotException . . . . .	646
Exponential_Smoother . . . . .	397
exterr_t . . . . .	399
fcdb_t . . . . .	399
file_t . . . . .	400

Fpga_Acquisition	406
Fpga_dynamic_bit_selection	410
Fpga_Multicorrelator_8sc	412
Fpga_Switch	416
FrontEndCal	419
ftp_t	421
Galileo_Almanac	421
Galileo_Almanac_Helper	425
Galileo_Cnav_Message	426
Galileo_Ephemeris	432
Galileo_Fnav_Message	445
Galileo_HAS_data	446
Galileo_Inav_Message	447
Galileo_Iono	448
Galileo_Utc_Model	460
GeoJSON_Printer	550
geph_t	551
Glonass_Gnav_Almanac	551
Glonass_Gnav_Ephemeris	558
Glonass_Gnav_Navigation_Message	572
Glonass_Gnav_Utc_Model	576
Gnss_circular_deque< T >	608
Gnss_circular_deque< Gnss_Synchro >	608
Gnss_Satellite	609
Gnss_Sdr_Supl_Client	614
Gnss_Signal	623
Gnss_Synchro	625
Gnss_Synchro_Udp_Sink	636
GNSSBlockFactory	636
GNSSBlockInterface	638
AcquisitionInterface	297
BeidouB1iPcpsAcquisition	342
BeidouB3iPcpsAcquisition	349
GalileoE1Pcps8msAmbiguousAcquisition	469
GalileoE1PcpsAmbiguousAcquisition	473
GalileoE1PcpsAmbiguousAcquisitionFpga	478
GalileoE1PcpsCccwsrAmbiguousAcquisition	484
GalileoE1PcpsQuickSyncAmbiguousAcquisition	488
GalileoE1PcpsTongAmbiguousAcquisition	492
GalileoE5aNoncoherentIQAcquisitionCaf	505
GalileoE5aPcpsAcquisition	509
GalileoE5aPcpsAcquisitionFpga	513
GalileoE5bPcpsAcquisition	524
GalileoE5bPcpsAcquisitionFpga	531
GalileoE6PcpsAcquisition	542
GlonassL1CaPcpsAcquisition	592
GlonassL2CaPcpsAcquisition	601
GpsL1CaPcpsAcquisition	720
GpsL1CaPcpsAcquisitionFineDoppler	724
GpsL1CaPcpsAcquisitionFpga	728
GpsL1CaPcpsAssistedAcquisition	735
GpsL1CaPcpsOpenCIAcquisition	738
GpsL1CaPcpsQuickSyncAcquisition	742
GpsL1CaPcpsTongAcquisition	747
GpsL2MPcpsAcquisition	759
GpsL2MPcpsAcquisitionFpga	763
GpsL5iPcpsAcquisition	774
GpsL5iPcpsAcquisitionFpga	778

Ad9361FpgaSignalSource . . . . .	298
ArraySignalConditioner . . . . .	303
BeamformerFilter . . . . .	306
ByteToShort . . . . .	355
ChannelInterface . . . . .	364
Channel . . . . .	356
CustomUDPSignalSource . . . . .	384
DirectResamplerConditioner . . . . .	388
FileSignalSource . . . . .	401
FirFilter . . . . .	402
FlexibandSignalSource . . . . .	404
Fmcomms2SignalSource . . . . .	405
FreqXlatingFirFilter . . . . .	418
GenSignalSource . . . . .	549
Gn3sSignalSource . . . . .	606
GnMaxSignalSource . . . . .	607
lbyteToCbyte . . . . .	791
lbyteToComplex . . . . .	792
lbyteToCshort . . . . .	793
lshortToComplex . . . . .	798
lshortToCshort . . . . .	799
LabsatSignalSource . . . . .	802
MmseResamplerConditioner . . . . .	805
MultichannelFileSignalSource . . . . .	808
NotchFilter . . . . .	813
NotchFilterLite . . . . .	814
NsrFileSignalSource . . . . .	816
ObservablesInterface . . . . .	819
HybridObservables . . . . .	789
OsmosdrSignalSource . . . . .	820
Pass_Through . . . . .	821
PlutosdrSignalSource . . . . .	866
PulseBlankingFilter . . . . .	869
PvtInterface . . . . .	876
Rtklib_Pvt . . . . .	903
RawArraySignalSource . . . . .	878
RtITcpSignalSource . . . . .	914
SignalConditioner . . . . .	932
SignalGenerator . . . . .	934
SpirFileSignalSource . . . . .	939
SpirGSS6450FileSignalSource . . . . .	940
TelemetryDecoderInterface . . . . .	948
BeidouB1iTelemetryDecoder . . . . .	346
BeidouB3iTelemetryDecoder . . . . .	353
GalileoE1BTelemetryDecoder . . . . .	462
GalileoE5aTelemetryDecoder . . . . .	520
GalileoE5bTelemetryDecoder . . . . .	538
GalileoE6TelemetryDecoder . . . . .	547
GlonassL1CaTelemetryDecoder . . . . .	596
GlonassL2CaTelemetryDecoder . . . . .	605
GpsL1CaTelemetryDecoder . . . . .	753
GpsL2CTelemetryDecoder . . . . .	754
GpsL5TelemetryDecoder . . . . .	785
SbasL1TelemetryDecoder . . . . .	919
TrackingInterface . . . . .	957
BeidouB1iDIIPITracking . . . . .	340
BeidouB3iDIIPITracking . . . . .	347
GalileoE1DIIPIVemITracking . . . . .	463

GalileoE1DIIPIIVemlTrackingFpga . . . . .	465
GalileoE1TcpConnectorTracking . . . . .	497
GalileoE5aDIIPIITracking . . . . .	499
GalileoE5aDIIPIITrackingFpga . . . . .	501
GalileoE5bDIIPIITracking . . . . .	521
GalileoE6DIIPIITracking . . . . .	540
GlonassL1CaDIIPIICAidTracking . . . . .	588
GlonassL1CaDIIPIITracking . . . . .	590
GlonassL2CaDIIPIICAidTracking . . . . .	597
GlonassL2CaDIIPIITracking . . . . .	599
GpsL1CaDIIPIITracking . . . . .	710
GpsL1CaDIIPIITrackingFpga . . . . .	712
GpsL1CaDIIPIITrackingGPU . . . . .	716
GpsL1CaKfTracking . . . . .	718
GpsL1CaTcpConnectorTracking . . . . .	751
GpsL2MDIIPIITracking . . . . .	755
GpsL2MDIIPIITrackingFpga . . . . .	757
GpsL5DIIPIITracking . . . . .	768
GpsL5DIIPIITrackingFpga . . . . .	770
TwoBitCpxFileSignalSource . . . . .	959
TwoBitPackedFileSignalSource . . . . .	960
UhdSignalSource . . . . .	961
GNSSFlowgraph . . . . .	639
Gnuplot . . . . .	643
Gps_Acq_Assist . . . . .	646
Gps_Almanac . . . . .	650
Gps_CNAV_Ephemeris . . . . .	654
Gps_CNAV_Iono . . . . .	668
Gps_CNAV_Navigation_Message . . . . .	672
Gps_CNAV_Utc_Model . . . . .	674
Gps_Ephemeris . . . . .	677
Gps_Iono . . . . .	693
Gps_Navigation_Message . . . . .	703
Gps_Utc_Model . . . . .	707
GPU_Complex . . . . .	786
GPU_Complex_Short . . . . .	786
Gpx_Printer . . . . .	787
gtime_t . . . . .	788
half_cyc_tag . . . . .	788
INIReader . . . . .	794
kernel_info_t . . . . .	800
Kml_Printer . . . . .	801
lex_t . . . . .	803
lexeph_t . . . . .	803
lexion_t . . . . .	804
lexmsg_t . . . . .	804
ModelFunction . . . . .	805
Monitor_Pvt . . . . .	806
Monitor_Pvt_Udp_Sink . . . . .	807
msm_h_t . . . . .	807
mt1_header . . . . .	808
nav_t . . . . .	810
Nmea_Printer . . . . .	811
ntrip_t . . . . .	817
Obs_Conf . . . . .	817
obs_t . . . . .	818
obsd_t . . . . .	818
Observables_Dump_Reader . . . . .	818

opt_t . . . . .	820
pclk_t . . . . .	822
pcps_acquisition_fpga . . . . .	833
pcpsconf_fpga_t . . . . .	864
pcv_t . . . . .	864
pcvs_t . . . . .	865
peph_t . . . . .	865
pppcorr_t . . . . .	867
prcopt_t . . . . .	867
Pvt_Conf . . . . .	870
Pvt_Solution . . . . .	871
Rtklib_Solver . . . . .	908
raw_t . . . . .	877
Rinex_Printer . . . . .	879
Rtcm . . . . .	884
Rtcm_Printer . . . . .	900
rtcm_t . . . . .	902
rtk_t . . . . .	903
Rtklib_Solver_Dump_Reader . . . . .	911
rtksvr_t . . . . .	911
Rtl_Tcp_Dongle_Info . . . . .	912
Sbas_Ephemeris . . . . .	915
sbs_t . . . . .	920
sbsfcorr_t . . . . .	921
sbsigp_t . . . . .	921
sbsigpband_t . . . . .	922
sbsion_t . . . . .	922
sbslcorr_t . . . . .	922
sbsmsg_t . . . . .	923
sbssat_t . . . . .	923
sbssatp_t . . . . .	924
seph_t . . . . .	924
Serdes_Gnss_Synchro . . . . .	924
Serdes_Monitor_Pvt . . . . .	927
serial_t . . . . .	929
snrmask_t . . . . .	935
sol_t . . . . .	935
solbuf_t . . . . .	935
solo_t . . . . .	936
solstat_t . . . . .	936
solstatbuf_t . . . . .	937
Spirent_Motion_Csv_Dump_Reader . . . . .	937
ssat_t . . . . .	941
ssr_t . . . . .	941
sta_t . . . . .	942
stec_t . . . . .	942
stream_cfg . . . . .	943
stream_t . . . . .	943
StringConverter . . . . .	944
sync_block . . . . .	
beamformer . . . . .	305
byte_x2_to_complex_byte . . . . .	354
complex_byte_to_float_x2 . . . . .	373
complex_float_to_complex_byte . . . . .	373
conjugate_cc . . . . .	376
conjugate_ic . . . . .	377
conjugate_sc . . . . .	378
cshort_to_float_x2 . . . . .	383

Gnss_Sdr_Valve . . . . .	623
Gr_Complex_Ip_Packet_Source . . . . .	787
rtklib_pvt_gs . . . . .	905
rtl_tcp_signal_source_c . . . . .	913
short_x2_to_cshort . . . . .	930
sync_decimator	
gnss_sdr_sample_counter . . . . .	614
interleaved_byte_to_complex_byte . . . . .	796
interleaved_byte_to_complex_short . . . . .	797
interleaved_short_to_complex_short . . . . .	797
sync_interpolator	
unpack_2bit_samples . . . . .	962
unpack_byte_2bit_cpx_samples . . . . .	963
unpack_byte_2bit_samples . . . . .	964
unpack_byte_4bit_samples . . . . .	964
unpack_intspir_1bit_samples . . . . .	965
unpack_spir_gss6450_samples . . . . .	966
Tcp_Communication . . . . .	944
Tcp_Packet_Data . . . . .	945
tcp_t . . . . .	946
tcpcli_t . . . . .	946
TcpCmdInterface . . . . .	946
tcpsvr_t . . . . .	947
tec_t . . . . .	948
tfe_t . . . . .	949
tled_t . . . . .	949
Tlm_Conf . . . . .	950
Tlm_Dump_Reader . . . . .	950
Tracking_2nd_DLL_filter . . . . .	951
Tracking_2nd_PLL_filter . . . . .	952
Tracking_Dump_Reader . . . . .	953
Tracking_FLL_PLL_filter . . . . .	954
Tracking_loop_filter . . . . .	955
Tracking_True_Obs_Reader . . . . .	956
trop_t . . . . .	958
True_Observables_Reader . . . . .	958
UnscentedFilter . . . . .	966
url_t . . . . .	967
v27_decision_t . . . . .	967
v27_poly_t . . . . .	967
v27_t . . . . .	968
Viterbi_Decoder . . . . .	968



## Chapter 7

# Class Index

### 7.1 Class List

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

<a href="#">Acq_Conf</a>	293
<a href="#">Acquisition_Dump_Reader</a>	294
<a href="#">Acquisition_msg_rx</a>	295
<a href="#">AcquisitionInterface</a>	
This abstract class represents an interface to an acquisition GNSS block	297
<a href="#">Ad9361FpgaSignalSource</a>	298
<a href="#">Agnss_Ref_Location</a>	
Interface of an Assisted GNSS REFERENCE LOCATION storage	299
<a href="#">Agnss_Ref_Time</a>	
Interface of an Assisted GNSS REFERENCE TIME storage	300
<a href="#">alm_t</a>	302
<a href="#">ambc_t</a>	302
<a href="#">ArraySignalConditioner</a>	
This class wraps blocks to change data_type_adapter, input_filter and resampler to be applied to the input flow of sampled signal	303
<a href="#">Bayesian_estimator</a>	
<a href="#">Bayesian_estimator</a> is an estimator of noise characteristics (i.e. mean, covariance)	304
<a href="#">beamformer</a>	
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	305
<a href="#">BeamformerFilter</a>	
Interface of an adapter of a digital beamformer block to a <a href="#">GNSSBlockInterface</a>	306
<a href="#">beidou_b1i_telemetry_decoder_gs</a>	
This class implements a block that decodes the BeiDou DNAV data	307
<a href="#">beidou_b3i_telemetry_decoder_gs</a>	
This class implements a block that decodes the BeiDou DNAV data	309
<a href="#">Beidou_Dnav_Almanac</a>	
This class is a storage for the BeiDou D1 almanac	311
<a href="#">Beidou_Dnav_Ephemeris</a>	
This class is a storage and orbital model functions for the GPS SV ephemeris data as described in BeiDou Navigation Satellite System Signal In Space Interface Control Document Open Service Signal B1I (Version 3.0)	314
<a href="#">Beidou_Dnav_Iono</a>	
This class is a storage for the BEIDOU IONOSPHERIC data as described in ICD v2.1	330
<a href="#">Beidou_Dnav_Navigation_Message</a>	
This class decodes a BeiDou D1 NAV Data message	333

<a href="#">Beidou_Dnav_Utc_Model</a>	
This class is a storage for the BeiDou DNAV UTC Model	337
<a href="#">BeidouB1iDlIPllTracking</a>	
This class implements a code DLL + carrier PLL tracking loop	340
<a href="#">BeidouB1iPcpsAcquisition</a>	
This class adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L1 C/A signals	342
<a href="#">BeidouB1iTelemetryDecoder</a>	
This class implements a NAV data decoder for BEIDOU B1I	346
<a href="#">BeidouB3iDlIPllTracking</a>	
This class implements a code DLL + carrier PLL tracking loop	347
<a href="#">BeidouB3iPcpsAcquisition</a>	
This class adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for BeiDou B3I signals	349
<a href="#">BeidouB3iTelemetryDecoder</a>	
This class implements a NAV data decoder for BEIDOU B1I	353
<a href="#">byte_x2_to_complex_byte</a>	
This class adapts two signed char streams into a std::complex<signed char> stream	354
<a href="#">ByteToShort</a>	
Adapts an 8-bits sample stream (IF) to a short int stream (IF)	355
<a href="#">Channel</a>	
This class represents a GNSS channel. It wraps an <a href="#">AcquisitionInterface</a> , a <a href="#">TrackingInterface</a> and a <a href="#">TelemetryDecoderInterface</a> , and handles their interaction through a Finite State Machine	356
<a href="#">Channel_Event</a>	360
<a href="#">channel_msg_receiver_cc</a>	
GNU Radio block that receives asynchronous channel messages from acquisition and tracking blocks	361
<a href="#">channel_status_msg_receiver</a>	
GNU Radio block that receives asynchronous channel messages from tlm blocks	362
<a href="#">ChannelFsm</a>	
This class implements a State Machine for channel	363
<a href="#">ChannelInterface</a>	
This abstract class represents an interface to a channel GNSS block	364
<a href="#">cl_fft_plan</a>	365
<a href="#">clFFT_Complex</a>	365
<a href="#">clFFT_Dim3</a>	366
<a href="#">clFFT_SplitComplex</a>	366
<a href="#">cnav_msg_decoder_t</a>	366
<a href="#">cnav_msg_t</a>	367
<a href="#">cnav_v27_part_t</a>	369
<a href="#">Command_Event</a>	372
<a href="#">complex_byte_to_float_x2</a>	
This class adapts a std::complex<signed char> stream into two 16-bits (short) streams	373
<a href="#">complex_float_to_complex_byte</a>	
This class adapts a gr_complex stream into a std::complex<signed char> stream	373
<a href="#">Concurrent_Map&lt; Data &gt;</a>	
This class implements a thread-safe std::map	374
<a href="#">Concurrent_Queue&lt; Data &gt;</a>	
This class implements a thread-safe std::queue	375
<a href="#">ConfigurationInterface</a>	
This abstract class represents an interface to configuration parameters	375
<a href="#">conjugate_cc</a>	
This class adapts a std::complex<short> stream into two 32-bits (float) streams	376
<a href="#">conjugate_ic</a>	
This class adapts a std::complex<short> stream into two 32-bits (float) streams	377
<a href="#">conjugate_sc</a>	
This class adapts a std::complex<short> stream into two 32-bits (float) streams	378

<a href="#">ControlThread</a>	
This class represents the main thread of the application, so the name is <a href="#">ControlThread</a> . 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	378
<a href="#">Cpu_Multicorrelator</a>	
Class that implements carrier wipe-off and correlators	381
<a href="#">Cpu_Multicorrelator_16sc</a>	
Class that implements carrier wipe-off and correlators	381
<a href="#">Cpu_Multicorrelator_Real_Codes</a>	
Class that implements carrier wipe-off and correlators	382
<a href="#">cshort_to_float_x2</a>	
This class adapts a <code>std::complex&lt;short&gt;</code> stream into two 32-bits (float) streams	383
<a href="#">CubatureFilter</a>	383
<a href="#">cuda_multicorrelator</a>	
Class that implements carrier wipe-off and correlators using NVIDIA CUDA GPU accelerators	384
<a href="#">CustomUDPSignalSource</a>	
This class reads from UDP packets, which streams interleaved I/Q samples over a network	384
<a href="#">dgps_t</a>	385
<a href="#">direct_resampler_conditioner_cb</a>	
This class implements a direct resampler conditioner for <code>std::complex&lt;signed char&gt;</code>	386
<a href="#">direct_resampler_conditioner_cc</a>	
This class implements a direct resampler conditioner for complex data	387
<a href="#">direct_resampler_conditioner_cs</a>	
This class implements a direct resampler conditioner for <code>std::complex&lt;short&gt;</code>	387
<a href="#">DirectResamplerConditioner</a>	
Interface of an adapter of a direct resampler conditioner block to a <code>SignalConditionerInterface</code>	388
<a href="#">Dll_Pll_Conf</a>	389
<a href="#">Dll_Pll_Conf_Fpga</a>	390
<a href="#">dll_pll_veml_tracking</a>	
This class implements a code DLL + carrier PLL tracking block	392
<a href="#">dll_pll_veml_tracking_fpga</a>	
This class implements a code DLL + carrier PLL tracking block	393
<a href="#">eph_t</a>	395
<a href="#">erp_t</a>	396
<a href="#">erpd_t</a>	396
<a href="#">Exponential_Smoother</a>	
Class that implements a first-order exponential smoother	397
<a href="#">exterr_t</a>	399
<a href="#">fcbd_t</a>	399
<a href="#">file_t</a>	400
<a href="#">FileConfiguration</a>	
This class is an implementation of the interface <a href="#">ConfigurationInterface</a>	400
<a href="#">FileSignalSource</a>	
Class that reads signals samples from a file and adapts it to a <code>SignalSourceInterface</code>	401
<a href="#">FirFilter</a>	
This class adapts a GNU Radio <code>gr_fir_filter</code> designed with <code>pm_remez</code>	402
<a href="#">FlexibandSignalSource</a>	
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)	404
<a href="#">Fmcomms2SignalSource</a>	405
<a href="#">Fpga_Acquisition</a>	
Class that implements carrier wipe-off and correlators	406
<a href="#">Fpga_dynamic_bit_selection</a>	
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	410
<a href="#">Fpga_Multicorrelator_8sc</a>	
Class that implements carrier wipe-off and correlators	412

<a href="#">Fpga_Switch</a>	
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	416
<a href="#">FreqXlatingFirFilter</a>	
This class adapts a gnuradio gr_freq_xlating_fir_filter designed with pm_remez	418
<a href="#">FrontEndCal</a>	419
<a href="#">ftp_t</a>	421
<a href="#">Galileo_Almanac</a>	
This class is a storage for the Galileo SV ALMANAC data	421
<a href="#">Galileo_Almanac_Helper</a>	
This class is a storage for the GALILEO ALMANAC data as described in GALILEO ICD	425
<a href="#">Galileo_Cnav_Message</a>	
This class handles the Galileo CNAV Data message, as described in the Galileo High Accuracy Service E6-B Signal-In-Space Message Specification v1.2 (April 2020)	426
<a href="#">Galileo_E1_Tcp_Connector_Tracking_cc</a>	
This class implements a code DLL + carrier PLL VEML (Very Early Minus Late) tracking block for Galileo E1 signals	427
<a href="#">galileo_e5a_noncoherentIQ_acquisition_caf_cc</a>	
This class implements a Parallel Code Phase Search Acquisition	428
<a href="#">Galileo_Ephemeris</a>	
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 <a href="https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo-OS-SIS-ICD.pdf">https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo-OS-SIS-ICD.pdf</a> )	432
<a href="#">Galileo_Fnav_Message</a>	
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 1.2 (Nov 2015). See <a href="https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo-OS-SIS-ICD.pdf">https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo-OS-SIS-ICD.pdf</a>	445
<a href="#">Galileo_HAS_data</a>	
This class is a storage for Galileo HAS message type 1, as defined in Galileo High Accuracy Service E6-B Signal-In-Space Message Specification v1.2 (April 2020)	446
<a href="#">Galileo_Inav_Message</a>	
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 1.2 (Nov 2015). See <a href="https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo-OS-SIS-ICD.pdf">https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo-OS-SIS-ICD.pdf</a>	447
<a href="#">Galileo_Iono</a>	
This class is a storage for the GALILEO IONOSPHERIC data as described in Galileo ICD paragraph 5.1.6	448
<a href="#">galileo_pcps_8ms_acquisition_cc</a>	
This class implements a Parallel Code Phase Search Acquisition for Galileo E1 signals with coherent integration time = 8 ms (two codes)	452
<a href="#">galileo_telemetry_decoder_gs</a>	
This class implements a block that decodes the INAV and FNAV data defined in Galileo ICD	458
<a href="#">Galileo_Utc_Model</a>	
This class is a storage for the GALILEO UTC MODEL data as described in Galileo ICD <a href="https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo-OS-SIS-ICD.pdf">https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo-OS-SIS-ICD.pdf</a> paragraph 5.1.7	460
<a href="#">GalileoE1BTelemetryDecoder</a>	
This class implements a NAV data decoder for Galileo INAV frames in E1B radio link	462
<a href="#">GalileoE1DIIPIVemlTracking</a>	
This class Adapts a DLL+PLL VEML (Very Early Minus Late) tracking loop block to a <a href="#">TrackingInterface</a> for Galileo E1 signals	463
<a href="#">GalileoE1DIIPIVemlTrackingFpga</a>	
This class Adapts a DLL+PLL VEML (Very Early Minus Late) tracking loop block to a <a href="#">TrackingInterface</a> for Galileo E1 signals	465

<a href="#">GalileoE1Pcps8msAmbiguousAcquisition</a>	Adapts a PCPS 8ms acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E1 Signals . . . .	469
<a href="#">GalileoE1PcpsAmbiguousAcquisition</a>	This class adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E1 Signals . .	473
<a href="#">GalileoE1PcpsAmbiguousAcquisitionFpga</a>	This class adapts a PCPS acquisition block off-loaded on an FPGA to an <a href="#">AcquisitionInterface</a> for Galileo E1 Signals . . . . .	478
<a href="#">GalileoE1PcpsCccwsrAmbiguousAcquisition</a>	Adapts a PCPS CCCWSR acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E1 Signals .	484
<a href="#">GalileoE1PcpsQuickSyncAmbiguousAcquisition</a>	This class adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E1 Signals . .	488
<a href="#">GalileoE1PcpsTongAmbiguousAcquisition</a>	Adapts a PCPS Tong acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E1 Signals . . . .	492
<a href="#">GalileoE1TcpConnectorTracking</a>	This class implements a code DLL + carrier PLL tracking loop . . . . .	497
<a href="#">GalileoE5aDIIPILTracking</a>	This class implements a code DLL + carrier PLL tracking loop . . . . .	499
<a href="#">GalileoE5aDIIPILTrackingFpga</a>	This class implements a code DLL + carrier PLL tracking loop . . . . .	501
<a href="#">GalileoE5aNoncoherentIQAcquisitionCaf</a>		505
<a href="#">GalileoE5aPcpsAcquisition</a>		509
<a href="#">GalileoE5aPcpsAcquisitionFpga</a>	This class adapts a PCPS acquisition block off-loaded on an FPGA to an <a href="#">AcquisitionInterface</a> for Galileo E5a signals . . . . .	513
<a href="#">GalileoE5aTelemetryDecoder</a>	This class implements a NAV data decoder for Galileo INAV frames in E1B radio link . . . . .	520
<a href="#">GalileoE5bDIIPILTracking</a>	This class implements a code DLL + carrier PLL tracking loop . . . . .	521
<a href="#">GalileoE5bPcpsAcquisition</a>		524
<a href="#">GalileoE5bPcpsAcquisitionFpga</a>	This class adapts a PCPS acquisition block off-loaded on an FPGA to an <a href="#">AcquisitionInterface</a> for Galileo E5b signals . . . . .	531
<a href="#">GalileoE5bTelemetryDecoder</a>	This class implements a NAV data decoder for Galileo INAV frames in E5b radio link . . . . .	538
<a href="#">GalileoE6DIIPILTracking</a>	This class implements a code DLL + carrier PLL tracking loop . . . . .	540
<a href="#">GalileoE6PcpsAcquisition</a>	This class adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E6 Signals . .	542
<a href="#">GalileoE6TelemetryDecoder</a>	This class implements a NAV data decoder for Galileo CNAV frames in E6 radio link . . . . .	547
<a href="#">GenSignalSource</a>	This class wraps blocks that generates synthesized GNSS signal and filters the signal . . . . .	549
<a href="#">GeoJSON_Printer</a>	Prints PVT solutions in GeoJSON format file . . . . .	550
<a href="#">geph_t</a>		551
<a href="#">Glonass_Gnav_Almanac</a>	This class is a storage for the GLONASS SV ALMANAC data as described GLONASS ICD (Edition 5.1) . . . . .	551
<a href="#">Glonass_Gnav_Ephemeris</a>	This class is a storage and orbital model functions for the GLONASS SV ephemeris data as described in GLONASS ICD (Edition 5.1) . . . . .	558
<a href="#">Glonass_Gnav_Navigation_Message</a>	This class decodes a GLONASS GNAV Data message as described in GLONASS ICD (Edition 5.1) . . . . .	572
<a href="#">Glonass_Gnav_Utc_Model</a>	This class is a storage for the GLONASS GNAV UTC MODEL data as described in GLONASS ICD (Edition 5.1) . . . . .	576

<a href="#">glonass_l1_ca_dll_pll_c_aid_tracking_cc</a>	
This class implements a DLL + PLL tracking loop block . . . . .	579
<a href="#">glonass_l1_ca_dll_pll_c_aid_tracking_sc</a>	
This class implements a DLL + PLL tracking loop block . . . . .	580
<a href="#">Glonass_L1_Ca_DLL_Pll_Tracking_cc</a>	
This class implements a DLL + PLL tracking loop block . . . . .	581
<a href="#">glonass_l1_ca_telemetry_decoder_gs</a>	
This class implements a block that decodes the GNAV data defined in GLONASS ICD v5.1 . . .	582
<a href="#">glonass_l2_ca_dll_pll_c_aid_tracking_cc</a>	
This class implements a DLL + PLL tracking loop block . . . . .	584
<a href="#">glonass_l2_ca_dll_pll_c_aid_tracking_sc</a>	
This class implements a DLL + PLL tracking loop block . . . . .	585
<a href="#">Glonass_L2_Ca_DLL_Pll_Tracking_cc</a>	
This class implements a DLL + PLL tracking loop block . . . . .	585
<a href="#">glonass_l2_ca_telemetry_decoder_gs</a>	
This class implements a block that decodes the GNAV data defined in GLONASS ICD v5.1 . . .	586
<a href="#">GlonassL1CaDIPIICAidTracking</a>	
This class implements a code DLL + carrier PLL tracking loop . . . . .	588
<a href="#">GlonassL1CaDIPIITracking</a>	
This class implements a code DLL + carrier PLL tracking loop . . . . .	590
<a href="#">GlonassL1CaPcpsAcquisition</a>	
This class adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L1 C/A signals .	592
<a href="#">GlonassL1CaTelemetryDecoder</a>	
This class implements a NAV data decoder for GLONASS L1 C/A . . . . .	596
<a href="#">GlonassL2CaDIPIICAidTracking</a>	
This class implements a code DLL + carrier PLL tracking loop . . . . .	597
<a href="#">GlonassL2CaDIPIITracking</a>	
This class implements a code DLL + carrier PLL tracking loop . . . . .	599
<a href="#">GlonassL2CaPcpsAcquisition</a>	
This class adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GLONASS L2 C/A signals . . . . .	601
<a href="#">GlonassL2CaTelemetryDecoder</a>	
This class implements a NAV data decoder for GLONASS L2 C/A . . . . .	605
<a href="#">Gn3sSignalSource</a>	
This class reads samples from a GN3S USB dongle, a RF front-end signal sampler . . . . .	606
<a href="#">GnMaxSignalSource</a>	
This class reads samples from a gnMAX2769 USB dongle, a RF front-end signal sampler . . .	607
<a href="#">Gnss_circular_deque&lt; T &gt;</a>	608
<a href="#">Gnss_Satellite</a>	
This class represents a GNSS satellite . . . . .	609
<a href="#">gnss_sdr_fpga_sample_counter</a>	613
<a href="#">gnss_sdr_sample_counter</a>	614
<a href="#">Gnss_Sdr_Supl_Client</a>	
Class that implements a C++ interface to external Secure User Location Protocol (SUPL) client library. . . . .	614
<a href="#">gnss_sdr_time_counter</a>	622
<a href="#">Gnss_Sdr_Valve</a>	
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 . . . . .	623
<a href="#">Gnss_Signal</a>	
This class represents a GNSS signal . . . . .	623
<a href="#">Gnss_Synchro</a>	
This is the class that contains the information that is shared by the processing blocks . . . . .	625
<a href="#">gnss_synchro_monitor</a>	
This class implements a monitoring block which allows sending a data stream with the receiver internal parameters ( <a href="#">Gnss_Synchro</a> objects) to local or remote clients over UDP . . . . .	635
<a href="#">Gnss_Synchro_Udp_Sink</a>	
This class sends serialized <a href="#">Gnss_Synchro</a> objects over UDP to one or multiple endpoints . . .	636

<a href="#">GNSSBlockFactory</a>	
Class that produces all kinds of GNSS blocks . . . . .	636
<a href="#">GNSSBlockInterface</a>	
This abstract class represents an interface to GNSS blocks . . . . .	638
<a href="#">GNSSFlowgraph</a>	
This class represents a GNSS flow graph . . . . .	639
<a href="#">Gnuplot</a> . . . . .	643
<a href="#">GnuplotException</a> . . . . .	646
<a href="#">Gps_Acq_Assist</a>	
This class is a storage for the GPS GSM RRLT 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
<a href="#">Gps_Almanac</a>	
This class is a storage for the GPS SV ALMANAC data as described in IS-GPS-200K . . . . .	650
<a href="#">Gps_CNAV_Ephemeris</a>	
This class is a storage and orbital model functions for the GPS SV ephemeris data as described in IS-GPS-200K . . . . .	654
<a href="#">Gps_CNAV_Iono</a>	
This class is a storage for the GPS IONOSPHERIC data as described in IS-GPS-200K . . . . .	668
<a href="#">Gps_CNAV_Navigation_Message</a>	
This class decodes a GPS CNAV Data message as described in IS-GPS-200K . . . . .	672
<a href="#">Gps_CNAV_Utc_Model</a>	
This class is a storage for the GPS UTC MODEL data as described in IS-GPS-200K . . . . .	674
<a href="#">Gps_Ephemeris</a>	
This class is a storage and orbital model functions for the GPS SV ephemeris data as described in IS-GPS-200K . . . . .	677
<a href="#">Gps_Iono</a>	
This class is a storage for the GPS IONOSPHERIC data as described in IS-GPS-200K . . . . .	693
<a href="#">Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc</a>	
This class implements a DLL + PLL tracking loop block . . . . .	697
<a href="#">Gps_L1_Ca_Kf_Tracking_cc</a>	
This class implements a DLL + PLL tracking loop block . . . . .	697
<a href="#">Gps_L1_Ca_Tcp_Connector_Tracking_cc</a>	
This class implements a DLL + PLL tracking loop block . . . . .	698
<a href="#">gps_l1_ca_telemetry_decoder_gs</a>	
This class implements a block that decodes the NAV data defined in IS-GPS-200K . . . . .	699
<a href="#">gps_l2c_telemetry_decoder_gs</a>	
This class implements a block that decodes CNAV data defined in IS-GPS-200K . . . . .	701
<a href="#">gps_l5_telemetry_decoder_gs</a>	
This class implements a GPS L5 Telemetry decoder . . . . .	702
<a href="#">Gps_Navigation_Message</a>	
This class decodes a GPS NAV Data message as described in IS-GPS-200K . . . . .	703
<a href="#">Gps_Utc_Model</a>	
This class is a storage for the GPS UTC MODEL data as described in IS-GPS-200K . . . . .	707
<a href="#">GpsL1CaDllPllTracking</a>	
This class implements a code DLL + carrier PLL tracking loop . . . . .	710
<a href="#">GpsL1CaDllPllTrackingFpga</a>	
This class implements a code DLL + carrier PLL tracking loop . . . . .	712
<a href="#">GpsL1CaDllPllTrackingGPU</a>	
This class implements a code DLL + carrier PLL tracking loop using GPU accelerated functions . . . . .	716
<a href="#">GpsL1CaKfTracking</a>	
This class implements a code DLL + carrier PLL tracking loop . . . . .	718
<a href="#">GpsL1CaPcpsAcquisition</a>	
This class adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L1 C/A signals . . . . .	720
<a href="#">GpsL1CaPcpsAcquisitionFineDoppler</a>	
This class Adapts a PCPS acquisition block with fine Doppler estimation to an <a href="#">AcquisitionInterface</a> for GPS L1 C/A signals . . . . .	724

<a href="#">GpsL1CaPcpsAcquisitionFpga</a>	
This class adapts a PCPS acquisition block off-loaded on an FPGA to an <a href="#">AcquisitionInterface</a> for GPS L1 C/A signals . . . . .	728
<a href="#">GpsL1CaPcpsAssistedAcquisition</a>	
This class adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L1 C/A signals . . . . .	735
<a href="#">GpsL1CaPcpsOpenCLAcquisition</a>	
This class adapts an OpenCL PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L1 C/A signals . . . . .	738
<a href="#">GpsL1CaPcpsQuickSyncAcquisition</a>	
This class adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L1 C/A signals . . . . .	742
<a href="#">GpsL1CaPcpsTongAcquisition</a>	
This class adapts a PCPS Tong acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L1 C/A signals . . . . .	747
<a href="#">GpsL1CaTcpConnectorTracking</a>	
This class implements a code DLL + carrier PLL tracking loop . . . . .	751
<a href="#">GpsL1CaTelemetryDecoder</a>	
This class implements a NAV data decoder for GPS L1 C/A . . . . .	753
<a href="#">GpsL2CTelemetryDecoder</a>	
This class implements a NAV data decoder for GPS L2 M . . . . .	754
<a href="#">GpsL2MDIIPITracking</a>	
This class implements a code DLL + carrier PLL tracking loop . . . . .	755
<a href="#">GpsL2MDIIPITrackingFpga</a>	
This class implements a code DLL + carrier PLL tracking loop . . . . .	757
<a href="#">GpsL2MPcpsAcquisition</a>	
This class adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L2 M signals . . . . .	759
<a href="#">GpsL2MPcpsAcquisitionFpga</a>	
This class adapts a PCPS acquisition block off-loaded on an FPGA to an <a href="#">AcquisitionInterface</a> for GPS L2 M signals . . . . .	763
<a href="#">GpsL5DIIPITracking</a>	
This class implements a code DLL + carrier PLL tracking loop . . . . .	768
<a href="#">GpsL5DIIPITrackingFpga</a>	
This class implements a code DLL + carrier PLL tracking loop . . . . .	770
<a href="#">GpsL5iPcpsAcquisition</a>	
This class adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L5i signals . . . . .	774
<a href="#">GpsL5iPcpsAcquisitionFpga</a>	
This class adapts a PCPS acquisition block off-loaded on an FPGA to an <a href="#">AcquisitionInterface</a> for GPS L5i signals . . . . .	778
<a href="#">GpsL5TelemetryDecoder</a>	
This class implements a NAV data decoder for GPS L5 . . . . .	785
<a href="#">GPU_Complex</a>	786
<a href="#">GPU_Complex_Short</a>	786
<a href="#">Gpx_Printer</a>	
Prints PVT information to GPX format file . . . . .	787
<a href="#">Gr_Complex_Ip_Packet_Source</a>	787
<a href="#">gtime_t</a>	788
<a href="#">half_cyc_tag</a>	788
<a href="#">hybrid_observables_gs</a>	
This class implements a block that computes observables . . . . .	789
<a href="#">HybridObservables</a>	
This class implements an <a href="#">ObservablesInterface</a> for observables of all kind of GNSS signals . . . . .	789
<a href="#">lbyteToCbyte</a>	791
<a href="#">lbyteToComplex</a>	
Adapts an I/Q interleaved byte integer sample stream to a gr_complex (float) stream . . . . .	792
<a href="#">lbyteToCshort</a>	
Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream . . . . .	793
<a href="#">INIReader</a>	
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.) . . . . .	794

<a href="#">InMemoryConfiguration</a>	
This class is an implementation of the interface <a href="#">ConfigurationInterface</a>	795
<a href="#">interleaved_byte_to_complex_byte</a>	
This class adapts an 8-bits interleaved sample stream into a 16-bits complex stream (std::complex<unsigned char>)	796
<a href="#">interleaved_byte_to_complex_short</a>	
This class adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream	797
<a href="#">interleaved_short_to_complex_short</a>	
This class adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream	797
<a href="#">IshortToComplex</a>	
Adapts an I/Q interleaved short integer sample stream to a gr_complex (float) stream	798
<a href="#">IshortToCshort</a>	
Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream	799
<a href="#">kernel_info_t</a>	800
<a href="#">Kml_Printer</a>	
Prints PVT information to OGC KML format file (can be viewed with Google Earth)	801
<a href="#">labsat23_source</a>	
This class implements conversion between Labsat2 and 3 format byte packet samples to gr_complex	801
<a href="#">LabsatSignalSource</a>	
This class reads samples stored by a LabSat 2 or LabSat 3 device	802
<a href="#">lex_t</a>	803
<a href="#">lexeph_t</a>	803
<a href="#">lexion_t</a>	804
<a href="#">lexmsg_t</a>	804
<a href="#">MmseResamplerConditioner</a>	
Interface of a MMSE resampler block adapter to a SignalConditionerInterface	805
<a href="#">ModelFunction</a>	805
<a href="#">Monitor_Pvt</a>	
This class contains parameters and outputs of the PVT block	806
<a href="#">Monitor_Pvt_Udp_Sink</a>	807
<a href="#">msm_h_t</a>	807
<a href="#">mt1_header</a>	808
<a href="#">MultichannelFileSignalSource</a>	
Class that reads signals samples from files at different frequency bands and adapts it to a SignalSourceInterface	808
<a href="#">nav_t</a>	810
<a href="#">Nmea_Printer</a>	
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)	811
<a href="#">Notch</a>	
This class implements a real-time software-defined multi state notch filter	812
<a href="#">NotchFilter</a>	813
<a href="#">NotchFilterLite</a>	814
<a href="#">NotchLite</a>	
This class implements a real-time software-defined multi state notch filter light version	815
<a href="#">NsrFileSignalSource</a>	
Class that reads signals samples from a file and adapts it to a SignalSourceInterface	816
<a href="#">ntrip_t</a>	817
<a href="#">Obs_Conf</a>	817
<a href="#">obs_t</a>	818
<a href="#">obsd_t</a>	818
<a href="#">Observables_Dump_Reader</a>	818
<a href="#">ObservablesInterface</a>	
This abstract class represents an interface to an observables block	819
<a href="#">opt_t</a>	820

<a href="#">OsmosdrSignalSource</a>	
This class reads samples OsmoSDR-compatible front-ends, such as HackRF or Realtek's RTL2832U-based USB dongle DVB-T receivers (see <a href="https://osmocom.org/projects/rtl-sdr/wiki">https://osmocom.org/projects/rtl-sdr/wiki</a> )	820
<a href="#">Pass_Through</a>	
This class implements a block that connects input and output (does nothing)	821
<a href="#">pclk_t</a>	822
<a href="#">pcps_acquisition</a>	
This class implements a Parallel Code Phase Search Acquisition	823
<a href="#">pcps_acquisition_fine_doppler_cc</a>	
This class implements a Parallel Code Phase Search Acquisition	827
<a href="#">pcps_acquisition_fpga</a>	
This class implements a Parallel Code Phase Search Acquisition that uses the FPGA	833
<a href="#">pcps_assisted_acquisition_cc</a>	
This class implements a Parallel Code Phase Search Acquisition	839
<a href="#">pcps_cccwsr_acquisition_cc</a>	
This class implements a Parallel Code Phase Search Acquisition with Coherent Channel Combining With Sign Recovery scheme	844
<a href="#">pcps_opencl_acquisition_cc</a>	
This class implements a Parallel Code Phase Search Acquisition	848
<a href="#">pcps_quicksync_acquisition_cc</a>	
This class implements a Parallel Code Phase Search Acquisition with the implementation of the Sparse QuickSync Algorithm	854
<a href="#">pcps_tong_acquisition_cc</a>	
This class implements a Parallel Code Phase Search Acquisition with Tong algorithm	859
<a href="#">pcpsconf_fpga_t</a>	864
<a href="#">pcv_t</a>	864
<a href="#">pcvs_t</a>	865
<a href="#">peph_t</a>	865
<a href="#">PlutosdrSignalSource</a>	866
<a href="#">pppcorr_t</a>	867
<a href="#">prcopt_t</a>	867
<a href="#">pulse_blanking_cc</a>	868
<a href="#">PulseBlankingFilter</a>	869
<a href="#">Pvt_Conf</a>	870
<a href="#">Pvt_Solution</a>	
Base class for a PVT solution	871
<a href="#">PvtInterface</a>	
This class represents an interface to a PVT block	876
<a href="#">raw_t</a>	877
<a href="#">RawArraySignalSource</a>	
This class reads samples from a GN3S USB dongle, a RF front-end signal sampler	878
<a href="#">Rinex_Printer</a>	
Class that handles the generation of Receiver INdependent EXchange format (RINEX) files	879
<a href="#">Rtcm</a>	
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
<a href="#">Rtcm_Printer</a>	
This class provides a implementation of a subset of the RTCM Standard 10403.2 messages	900
<a href="#">rtcm_t</a>	902
<a href="#">rtk_t</a>	903
<a href="#">Rtklib_Pvt</a>	
This class implements a <a href="#">PvtInterface</a> for the RTKLIB PVT block	903
<a href="#">rtklib_pvt_gs</a>	
This class implements a block that computes the PVT solution using the RTKLIB integrated library	905
<a href="#">Rtklib_Solver</a>	
This class implements a PVT solution based on RTKLIB	908

Rtklib_Solver_Dump_Reader	911
rtksvr_t	911
Rtl_Tcp_Dongle_Info	
This class represents the dongle information which is sent by rtl_tcp	912
rtl_tcp_signal_source_c	
This class reads interleaved I/Q samples from an rtl_tcp server and outputs complex types	913
RtlTcpSignalSource	
This class reads from rtl_tcp, which streams interleaved I/Q samples over TCP. (see <a href="https://osmocom.org/projects/rtl-sdr/wiki">https://osmocom.org/projects/rtl-sdr/wiki</a> )	914
Sbas_Ephemeris	
This class stores SBAS SV ephemeris data	915
sbas_l1_telemetry_decoder_gs	
This class implements a block that decodes the SBAS integrity and corrections data defined in RTCA MOPS DO-229	918
SbasL1TelemetryDecoder	
This class implements a NAV data decoder for SBAS frames in L1 radio link	919
sbs_t	920
sbsfcrr_t	921
sbsigp_t	921
sbsigpband_t	922
sbsion_t	922
sbslcorr_t	922
sbsmsg_t	923
sbssat_t	923
sbssatp_t	924
seph_t	924
Serdes_Gnss_Synchro	
This class implements serialization and deserialization of Gnss_Synchro objects using Protocol Buffers	924
Serdes_Monitor_Pvt	
This class implements serialization and deserialization of Monitor_Pvt objects using Protocol Buffers	927
serial_t	929
short_x2_to_cshort	
This class adapts two short streams into a std::complex<short> stream	930
signal_generator_c	
This class generates synthesized GNSS signal	931
SignalConditioner	
This class wraps blocks to change data_type_adapter, input_filter and resampler to be applied to the input flow of sampled signal	932
SignalGenerator	
This class generates synthesized GNSS signal	934
snrmask_t	935
sol_t	935
solbuf_t	935
soloft_t	936
solstat_t	936
solstatbuf_t	937
Spirent_Motion_Csv_Dump_Reader	937
SpirFileSignalSource	
Class that reads signals samples from a file and adapts it to a SignalSourceInterface	939
SpirGSS6450FileSignalSource	
Class that reads signals samples from a file and adapts it to a SignalSourceInterface	940
ssat_t	941
ssr_t	941
sta_t	942
stec_t	942
stream_cfg	943

<a href="#">stream_t</a>	943
<a href="#">StringConverter</a>	
Class that interprets the contents of a string and converts it into different types	944
<a href="#">Tcp_Communication</a>	
TCP communication class	944
<a href="#">Tcp_Packet_Data</a>	
Class that implements a TCP data packet	945
<a href="#">tcp_t</a>	946
<a href="#">tcpcli_t</a>	946
<a href="#">TcpCmdInterface</a>	946
<a href="#">tcpsvr_t</a>	947
<a href="#">tec_t</a>	948
<a href="#">TelemetryDecoderInterface</a>	
This abstract class represents an interface to a navigation GNSS block	948
<a href="#">tle_t</a>	949
<a href="#">tled_t</a>	949
<a href="#">Tlm_Conf</a>	950
<a href="#">Tlm_Dump_Reader</a>	950
<a href="#">Tracking_2nd_DLL_filter</a>	
This class implements a 2nd order DLL filter for code tracking loop	951
<a href="#">Tracking_2nd_PLL_filter</a>	
This class implements a 2nd order PLL filter for carrier tracking loop	952
<a href="#">Tracking_Dump_Reader</a>	953
<a href="#">Tracking_FLL_PLL_filter</a>	
This class implements a hybrid FLL and PLL filter for tracking carrier loop	954
<a href="#">Tracking_loop_filter</a>	
This class implements a generic 1st, 2nd or 3rd order loop filter	955
<a href="#">Tracking_True_Obs_Reader</a>	956
<a href="#">TrackingInterface</a>	
This abstract class represents an interface to a tracking block	957
<a href="#">trop_t</a>	958
<a href="#">True_Observables_Reader</a>	958
<a href="#">TwoBitCpxFileSignalSource</a>	
Class that reads signals samples from a file and adapts it to a SignalSourceInterface	959
<a href="#">TwoBitPackedFileSignalSource</a>	
Class that reads signals samples from a file and adapts it to a SignalSourceInterface	960
<a href="#">UhdSignalSource</a>	
This class reads samples from a UHD device (see <a href="http://code.ettus.com/redmine/ettus/projects/uhd">http://code.ettus.com/redmine/ettus/projects/uhd</a> )	961
<a href="#">unpack_2bit_samples</a>	
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)	962
<a href="#">unpack_byte_2bit_cpx_samples</a>	
This class implements conversion between byte packet samples to 2bit_cpx samples 1 byte = 2 x complex 2bit I, + 2bit Q samples	963
<a href="#">unpack_byte_2bit_samples</a>	
This class implements conversion between byte packet samples to 2bit samples 1 byte = 4 2bit samples	964
<a href="#">unpack_byte_4bit_samples</a>	
This class implements conversion between byte packet samples to 4bit_cpx samples 1 byte = 1 x complex 4bit I, + 4bit Q samples	964
<a href="#">unpack_intspir_1bit_samples</a>	
This class implements conversion between byte packet samples to 2bit samples 1 byte = 4 2bit samples	965
<a href="#">unpack_spir_gss6450_samples</a>	966
<a href="#">UnscentedFilter</a>	966
<a href="#">url_t</a>	967

<a href="#">v27_decision_t</a>	967
<a href="#">v27_poly_t</a>	967
<a href="#">v27_t</a>	968
<a href="#">Viterbi_Decoder</a>	
Class that implements a Viterbi decoder	968



## Chapter 8

# File Index

### 8.1 File List

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

<a href="#">acq_conf.h</a>	Class that contains all the configuration parameters for generic acquisition block based on the PCPS algorithm . . . . .	971
<a href="#">acquisition_dump_reader.h</a>	Helper file for unit testing . . . . .	971
<a href="#">acquisition_interface.h</a>	Header file of the interface to an acquisition GNSS block . . . . .	972
<a href="#">acquisition_msg_rx.h</a>	This is a helper class to catch the asynchronous messages emitted by an acquisition block . . .	973
<a href="#">ad9361_fpga_signal_source.h</a>	Signal source for Analog Devices front-end AD9361 connected directly to 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 . . . . .	973
<a href="#">ad9361_manager.h</a>	An Analog Devices AD9361 front-end configuration library wrapper for configure some functions via iiod link . . . . .	974
<a href="#">agnss_ref_location.h</a>	Interface of an Assisted GNSS REFERENCE LOCATION storage . . . . .	975
<a href="#">agnss_ref_time.h</a>	Interface of an Assisted GNSS REFERENCE TIME storage . . . . .	976
<a href="#">array_signal_conditioner.h</a>	It wraps blocks to change data type, filter and resample input data, adapted to array receiver . .	976
<a href="#">bayesian_estimation.h</a>	Interface of a library with Bayesian noise statistic estimation . . . . .	977
<a href="#">beamformer.h</a>	Simple spatial filter using RAW array input and beamforming coefficients . . . . .	978
<a href="#">beamformer_filter.h</a>	Interface of an adapter of a digital beamformer . . . . .	978
<a href="#">Beidou_B1I.h</a>	Defines system parameters for BeiDou B1I signal and DNAV data . . . . .	979
<a href="#">beidou_b1i_dll_pll_tracking.h</a>	Interface of an adapter of a DLL+PLL tracking loop block for Beidou B1I to a <a href="#">TrackingInterface</a> .	980
<a href="#">beidou_b1i_pcps_acquisition.h</a>	Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Beidou B1I signals . . . . .	981

<a href="#">beidou_b1i_signal_replica.h</a>	
This file implements various functions for BeiDou B1I signal replica generation . . . . .	981
<a href="#">beidou_b1i_telemetry_decoder.h</a>	
Interface of an adapter of a Beidou B1I NAV data decoder block to a <a href="#">TelemetryDecoderInterface</a>	982
<a href="#">beidou_b1i_telemetry_decoder_gs.h</a>	
Implementation of a BEIDOU B1I DNAV data decoder block . . . . .	983
<a href="#">Beidou_B3I.h</a>	
Defines system parameters for BeiDou B3I signal and DNAV data . . . . .	984
<a href="#">beidou_b3i_dll_pll_tracking.h</a>	
Interface of an adapter of a DLL+PLL tracking loop block for Beidou B3I to a <a href="#">TrackingInterface</a>	985
<a href="#">beidou_b3i_pcps_acquisition.h</a>	
Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Beidou B3I signals . . . . .	985
<a href="#">beidou_b3i_signal_replica.h</a>	
This file implements various functions for BeiDou B3I signal replica generation . . . . .	986
<a href="#">beidou_b3i_telemetry_decoder.h</a>	
Interface of an adapter of a Beidou B3I NAV data decoder block to a <a href="#">TelemetryDecoderInterface</a>	987
<a href="#">beidou_b3i_telemetry_decoder_gs.h</a>	
Implementation of a BEIDOU B3I DNAV data decoder block . . . . .	987
<a href="#">Beidou_DNAV.h</a>	
Defines system parameters for BeiDou DNAV data processing . . . . .	988
<a href="#">beidou_dnav_almanac.h</a>	
Interface of a Beidou DNAV Almanac storage . . . . .	993
<a href="#">beidou_dnav_ephemeris.h</a>	
Interface of a BEIDOU EPHEMERIS storage . . . . .	993
<a href="#">beidou_dnav_iono.h</a>	
Interface of a BEIDOU IONOSPHERIC MODEL storage . . . . .	994
<a href="#">beidou_dnav_navigation_message.h</a>	
Interface of a BeiDou DNAV Data message decoder . . . . .	994
<a href="#">beidou_dnav_utc_model.h</a>	
Interface of a BeiDou UTC MODEL storage . . . . .	995
<a href="#">bits.h</a>	
Utilities for bit manipulation of the libswiftnav library . . . . .	995
<a href="#">byte_to_short.h</a>	
Adapts an 8-bits sample stream (IF) to a short int stream (IF) . . . . .	996
<a href="#">byte_x2_to_complex_byte.h</a>	
Adapts two signed char streams into a std::complex<signed char> stream . . . . .	997
<a href="#">channel.h</a>	
Interface of a GNSS channel . . . . .	998
<a href="#">channel_event.h</a>	
Class that defines a channel event . . . . .	998
<a href="#">channel_fsm.h</a>	
Interface of the State Machine for channel . . . . .	999
<a href="#">channel_interface.h</a>	
This class represents an interface to a channel GNSS block . . . . .	1000
<a href="#">channel_msg_receiver_cc.h</a>	
GNU Radio block that receives asynchronous channel messages from acquisition and tracking blocks . . . . .	1000
<a href="#">channel_status_msg_receiver.h</a>	
GNU Radio block that receives asynchronous channel messages from acquisition and tracking blocks . . . . .	1001
<a href="#">cIFFT.h</a>	
FFT in OpenCL . . . . .	1002
<a href="#">cnav_msg.h</a>	
Utilities for CNAV message manipulation of the libswiftnav library . . . . .	1003
<a href="#">command_event.h</a>	
Class that defines a receiver command event . . . . .	1004
<a href="#">complex_byte_to_float_x2.h</a>	
Adapts a std::complex<signed char> stream into two 16-bits (short) streams . . . . .	1004

<a href="#">complex_float_to_complex_byte.h</a>	Adapts a gr_complex stream into a std::complex<signed char> stream . . . . .	1005
<a href="#">concurrent_map.h</a>	Interface of a thread-safe std::map . . . . .	1006
<a href="#">concurrent_queue.h</a>	Interface of a thread-safe std::queue . . . . .	1006
<a href="#">configuration_interface.h</a>	This class represents an interface to configuration parameters . . . . .	1007
<a href="#">conjugate_cc.h</a>	Conjugate a stream of gr_complex . . . . .	1008
<a href="#">conjugate_ic.h</a>	Conjugate a stream of lv_8sc_t ( std::complex<char> ) . . . . .	1008
<a href="#">conjugate_sc.h</a>	Conjugate a stream of lv_16sc_t ( std::complex<short> ) . . . . .	1009
<a href="#">control_thread.h</a>	Interface of the receiver control plane . . . . .	1010
<a href="#">convolutional.h</a>	General functions used to implement convolutional encoding . . . . .	1011
<a href="#">cpu_multicorrelator.h</a>	High optimized CPU vector multiTAP correlator class . . . . .	1012
<a href="#">cpu_multicorrelator_16sc.h</a>	Highly optimized CPU vector multiTAP correlator class for lv_16sc_t (short int complex) . . . . .	1012
<a href="#">cpu_multicorrelator_real_codes.h</a>	Highly optimized CPU vector multiTAP correlator class using real-valued local codes . . . . .	1013
<a href="#">cshort_to_float_x2.h</a>	Adapts a std::complex<short> stream into two float streams . . . . .	1013
<a href="#">cuda_multicorrelator.h</a>	Highly optimized CUDA GPU vector multiTAP correlator class . . . . .	1014
<a href="#">custom_udp_signal_source.h</a>	Receives ip frames containing samples in UDP frame encapsulation using a high performance packet capture library (libpcap) . . . . .	1015
<a href="#">direct_resampler_conditioner.h</a>	Interface of an adapter of a direct resampler conditioner block to a SignalConditionerInterface . . . . .	1015
<a href="#">direct_resampler_conditioner_cb.h</a>	Nearest neighborhood resampler with std::complex<signed char> input and std::complex<signed char> output . . . . .	1016
<a href="#">direct_resampler_conditioner_cc.h</a>	Nearest neighborhood resampler with gr_complex input and gr_complex output . . . . .	1017
<a href="#">direct_resampler_conditioner_cs.h</a>	Nearest neighborhood resampler with std::complex<short> input and std::complex<short> output . . . . .	1017
<a href="#">display.h</a>	Defines useful display constants . . . . .	1018
<a href="#">dll_pll_conf.h</a>	Class that contains all the configuration parameters for generic tracking block based on a DLL and a PLL . . . . .	1019
<a href="#">dll_pll_conf_fpga.h</a>	Class that contains all the configuration parameters for generic tracking block based on a DLL and a PLL for the FPGA . . . . .	1019
<a href="#">dll_pll_veml_tracking.h</a>	Implementation of a code DLL + carrier PLL tracking block . . . . .	1020
<a href="#">dll_pll_veml_tracking_fpga.h</a>	Implementation of a code DLL + carrier PLL tracking block using an FPGA . . . . .	1021
<a href="#">edc.h</a>	Utilities for CRC computation of the libswiftnav library . . . . .	1022
<a href="#">exponential_smoother.h</a>	Class that implements an exponential smoother . . . . .	1023

<a href="#">fec.h</a>	Utilities for the convolutional encoder of the libswiftnav library . . . . .	1023
<a href="#">fft_base_kernels.h</a>	FFT base kernels for OpenCL . . . . .	1024
<a href="#">fft_internal.h</a>	Internals of FFT for OpenCL . . . . .	1024
<a href="#">file_configuration.h</a>	A <a href="#">ConfigurationInterface</a> that reads the configuration from a file . . . . .	1025
<a href="#">file_signal_source.h</a>	Interface of a class that reads signals samples from a file and adapts it to a <a href="#">SignalSourceInterface</a> . . . . .	1026
<a href="#">fir_filter.h</a>	Adapts a gnuradio gr_fir_filter designed with pm_remez . . . . .	1026
<a href="#">flexiband_signal_source.h</a>	Signal Source adapter for the Teleorbit Flexiband front-end device. This adapter requires a Flexiband GNU Radio driver installed (not included with GNSS-SDR) . . . . .	1027
<a href="#">fmcomms2_signal_source.h</a>	Interface to use SDR hardware based in FMCOMMS2 driver from analog devices, for example FMCOMMS4 and ADALM-PLUTO (PlutoSdr) . . . . .	1028
<a href="#">fpga_acquisition.h</a>	Highly optimized FPGA vector correlator class . . . . .	1028
<a href="#">fpga_dynamic_bit_selection.h</a>	Dynamic bit selection in the received signal . . . . .	1029
<a href="#">fpga_multicorrelator.h</a>	FPGA vector correlator class . . . . .	1029
<a href="#">fpga_switch.h</a>	Switch that connects the HW accelerator queues to the analog front end or the DMA . . . . .	1030
<a href="#">freq_xlating_fir_filter.h</a>	Adapts a gnuradio gr_freq_xlating_fir_filter designed with gr_remez . . . . .	1031
<a href="#">front_end_cal.h</a>	Interface of the Front-end calibration program . . . . .	1031
<a href="#">galileo_almanac.h</a>	Interface of a Galileo ALMANAC storage . . . . .	1032
<a href="#">galileo_almanac_helper.h</a>	Interface of a Galileo ALMANAC storage helper . . . . .	1032
<a href="#">Galileo_CNAV.h</a>	Galileo CNAV message constants. Data from: Galileo High Accuracy Service E6-B Signal-In-Space Message Specification v1.2 (April 2020) . . . . .	1033
<a href="#">galileo_cnav_message.h</a>	Implementation of a Galileo CNAV Data message as described in Galileo High Accuracy Service E6-B Signal-In-Space Message Specification v1.2 (April 2020) . . . . .	1035
<a href="#">Galileo_E1.h</a>	Defines system parameters for Galileo E1 signal and NAV data . . . . .	1035
<a href="#">galileo_e1_dll_pll_veml_tracking.h</a>	Adapts a DLL+PLL VEML (Very Early Minus Late) tracking loop block to a <a href="#">TrackingInterface</a> for Galileo E1 signals . . . . .	1037
<a href="#">galileo_e1_dll_pll_veml_tracking_fpga.h</a>	Adapts a DLL+PLL VEML (Very Early Minus Late) tracking loop block to a <a href="#">TrackingInterface</a> for Galileo E1 signals for the FPGA . . . . .	1037
<a href="#">galileo_e1_pcps_8ms_ambiguous_acquisition.h</a>	Adapts a PCPS 8ms acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E1 Signals . . . . .	1038
<a href="#">galileo_e1_pcps_ambiguous_acquisition.h</a>	Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E1 Signals . . . . .	1039
<a href="#">galileo_e1_pcps_ambiguous_acquisition_fpga.h</a>	Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E1 Signals for the FPGA . . . . .	1039
<a href="#">galileo_e1_pcps_cccwsr_ambiguous_acquisition.h</a>	Adapts a PCPS CCCWSR acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E1 Signals . . . . .	1040
<a href="#">galileo_e1_pcps_quicksync_ambiguous_acquisition.h</a>	Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E1 Signals . . . . .	1040

<a href="#">galileo_e1_pcps_tong_ambiguous_acquisition.h</a>	
Adapts a PCPS Tong acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E1 Signals	1041
<a href="#">galileo_e1_signal_replica.h</a>	
This library implements various functions for Galileo E1 signal replica generation	1042
<a href="#">galileo_e1_tcp_connector_tracking.h</a>	
Interface of an adapter of a TCP connector block based on code DLL + carrier PLL for Galileo E1 to a <a href="#">TrackingInterface</a>	1042
<a href="#">galileo_e1_tcp_connector_tracking_cc.h</a>	
Interface of a TCP connector block based on code DLL + carrier PLL VEML (Very Early Minus Late) tracking block for Galileo E1 signals	1043
<a href="#">galileo_e1b_telemetry_decoder.h</a>	
Interface of an adapter of a GALILEO E1B NAV data decoder block to a <a href="#">TelemetryDecoderInterface</a>	1044
<a href="#">galileo_e5_signal_replica.h</a>	
This library implements various functions for Galileo E5 signal replica generation	1045
<a href="#">Galileo_E5a.h</a>	
Defines system parameters for Galileo E5a signal and NAV data	1045
<a href="#">galileo_e5a_dll_pll_tracking.h</a>	
Adapts a code DLL + carrier PLL tracking block to a <a href="#">TrackingInterface</a> for Galileo E5a signals	1047
<a href="#">galileo_e5a_dll_pll_tracking_fpga.h</a>	
Adapts a code DLL + carrier PLL tracking block to a <a href="#">TrackingInterface</a> for Galileo E5a signals for the FPGA	1047
<a href="#">galileo_e5a_noncoherent_iq_acquisition_caf.h</a>	
Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E5a data and pilot Signals	1048
<a href="#">galileo_e5a_noncoherent_iq_acquisition_caf_cc.h</a>	
Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E5a data and pilot Signals	1049
<a href="#">galileo_e5a_pcps_acquisition.h</a>	
Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E5a data and pilot Signals	1050
<a href="#">galileo_e5a_pcps_acquisition_fpga.h</a>	
Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E5a data and pilot Signals for the FPGA	1050
<a href="#">galileo_e5a_telemetry_decoder.h</a>	
Interface of an adapter of a GALILEO E5a FNAV data decoder block to a <a href="#">TelemetryDecoderInterface</a>	1051
<a href="#">Galileo_E5b.h</a>	
Defines system parameters for Galileo E5b signal and NAV data	1052
<a href="#">galileo_e5b_dll_pll_tracking.h</a>	
Adapts a code DLL + carrier PLL tracking block to a <a href="#">TrackingInterface</a> for Galileo E5b signals	1053
<a href="#">galileo_e5b_pcps_acquisition.h</a>	
Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E5b data and pilot Signals	1054
<a href="#">galileo_e5b_pcps_acquisition_fpga.h</a>	
Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E5b data and pilot Signals for the FPGA	1054
<a href="#">galileo_e5b_telemetry_decoder.h</a>	
Interface of an adapter of a GALILEO E5B NAV data decoder block to a <a href="#">TelemetryDecoderInterface</a>	1055
<a href="#">Galileo_E6.h</a>	
Defines system parameters for Galileo E6 B/C signal, as published at: European Union, E6-B/C Codes Technical Note, Issue 1, January 2019	1056
<a href="#">galileo_e6_dll_pll_tracking.h</a>	
Adapts a code DLL + carrier PLL tracking block to a <a href="#">TrackingInterface</a> for Galileo E6 signals	1057
<a href="#">galileo_e6_pcps_acquisition.h</a>	
Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Galileo E6 B/C Signals	1057
<a href="#">galileo_e6_signal_replica.h</a>	
This library implements various functions for Galileo E6 signal replica generation	1058
<a href="#">galileo_e6_telemetry_decoder.h</a>	
Interface of an adapter of a GALILEO E6 CNAV data decoder block to a <a href="#">TelemetryDecoderInterface</a>	1059

<a href="#">galileo_ephemeris.h</a>	Interface of a Galileo EPHEMERIS storage . . . . .	1059
<a href="#">Galileo_FNAV.h</a>	Galileo FNAV message constants . . . . .	1060
<a href="#">galileo_fnav_message.h</a>	Implementation of a Galileo F/NAV Data message as described in Galileo OS SIS ICD Issue 1.2 (Nov. 2015) . . . . .	1063
<a href="#">galileo_has_data.h</a>	Class for Galileo HAS message type 1 data storage . . . . .	1064
<a href="#">Galileo_INAV.h</a>	Galileo INAV message constants . . . . .	1064
<a href="#">galileo_inav_message.h</a>	Implementation of a Galileo I/NAV Data message as described in Galileo OS SIS ICD Issue 1.2 (Nov. 2015) . . . . .	1068
<a href="#">galileo_iono.h</a>	Interface of a Galileo Ionospheric Model storage . . . . .	1069
<a href="#">galileo_pcps_8ms_acquisition_cc.h</a>	This class implements a Parallel Code Phase Search Acquisition for Galileo E1 signals with coherent integration time = 8 ms (two codes) . . . . .	1070
<a href="#">galileo_telemetry_decoder_gs.h</a>	Implementation of a Galileo unified INAV and FNAV message demodulator block . . . . .	1071
<a href="#">galileo_utc_model.h</a>	Interface of a Galileo UTC MODEL storage . . . . .	1071
<a href="#">gen_signal_source.h</a>	It wraps blocks that generates synthesized GNSS signal and filters it . . . . .	1072
<a href="#">geofunctions.h</a>	A set of coordinate transformations functions and helpers, some of them migrated from MATLAB, for geographic information systems . . . . .	1073
<a href="#">geojson_printer.h</a>	Interface of a class that prints PVT solutions in GeoJSON format . . . . .	1074
<a href="#">glonass_gnav_almanac.h</a>	Interface of a GLONASS GNAV ALMANAC storage . . . . .	1074
<a href="#">glonass_gnav_ephemeris.h</a>	Interface of a GLONASS EPHEMERIS storage . . . . .	1075
<a href="#">glonass_gnav_navigation_message.h</a>	Interface of a GLONASS GNAV Data message decoder as described in GLONASS ICD (Edition 5.1) . . . . .	1076
<a href="#">glonass_gnav_utc_model.h</a>	Interface of a GLONASS GNAV UTC MODEL storage . . . . .	1077
<a href="#">glonass_l1_ca_dll_pll_c_aid_tracking.h</a>	Interface of an adapter of a DLL+PLL tracking loop block for Glonass L1 C/A to a <a href="#">TrackingInterface</a> . . . . .	1077
<a href="#">glonass_l1_ca_dll_pll_c_aid_tracking_cc.h</a>	Implementation of a code DLL + carrier PLL tracking block . . . . .	1078
<a href="#">glonass_l1_ca_dll_pll_c_aid_tracking_sc.h</a>	Implementation of a code DLL + carrier PLL tracking block . . . . .	1079
<a href="#">glonass_l1_ca_dll_pll_tracking.h</a>	Interface of an adapter of a DLL+PLL tracking loop block for Glonass L1 C/A to a <a href="#">TrackingInterface</a> . . . . .	1080
<a href="#">glonass_l1_ca_dll_pll_tracking_cc.h</a>	Implementation of a code DLL + carrier PLL tracking block . . . . .	1081
<a href="#">glonass_l1_ca_pcps_acquisition.h</a>	Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Glonass L1 C/A signals . . . . .	1082
<a href="#">glonass_l1_ca_telemetry_decoder.h</a>	Interface of an adapter of a GLONASS L1 C/A NAV data decoder block to a <a href="#">TelemetryDecoderInterface</a> . . . . .	1082
<a href="#">glonass_l1_ca_telemetry_decoder_gs.h</a>	Implementation of a GLONASS L1 C/A NAV data decoder block . . . . .	1083

<a href="#">GLONASS_L1_L2_CA.h</a>	Defines system parameters for GLONASS L1 C/A signal and NAV data	1084
<a href="#"><b>glonass_l1_signal_replica.h</b></a>		??
<a href="#">glonass_l2_ca_dll_pll_c_aid_tracking.h</a>	Interface of an adapter of a DLL+PLL tracking loop block for Glonass L2 C/A to a <a href="#">TrackingInterface</a>	1088
<a href="#">glonass_l2_ca_dll_pll_c_aid_tracking_cc.h</a>	Implementation of a code DLL + carrier PLL tracking block	1089
<a href="#">glonass_l2_ca_dll_pll_c_aid_tracking_sc.h</a>	Implementation of a code DLL + carrier PLL tracking block	1090
<a href="#">glonass_l2_ca_dll_pll_tracking.h</a>	Interface of an adapter of a DLL+PLL tracking loop block for Glonass L2 C/A to a <a href="#">TrackingInterface</a>	1091
<a href="#">glonass_l2_ca_dll_pll_tracking_cc.h</a>	Implementation of a code DLL + carrier PLL tracking block	1091
<a href="#">glonass_l2_ca_pcps_acquisition.h</a>	Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for Glonass L2 C/A signals	1092
<a href="#">glonass_l2_ca_telemetry_decoder.h</a>	Interface of an adapter of a GLONASS L2 C/A NAV data decoder block to a <a href="#">TelemetryDecoderInterface</a>	1093
<a href="#">glonass_l2_ca_telemetry_decoder_gs.h</a>	Implementation of a GLONASS L2 C/A NAV data decoder block	1094
<a href="#">glonass_l2_signal_replica.h</a>	This file implements various functions for GLONASS L2 CA signal replica generation	1095
<a href="#">gn3s_signal_source.h</a>	GN3S USB dongle GPS RF front-end signal sampler driver	1095
<a href="#">gnmax_signal_source.h</a>	GnMAX2769 USB dongle GPS RF front-end signal sampler driver	1096
<a href="#">gnss_block_factory.h</a>	Interface of a factory that returns smart pointers to GNSS blocks	1097
<a href="#">gnss_block_interface.h</a>	This interface represents a GNSS block	1097
<a href="#">gnss_circular_deque.h</a>	This class implements a circular deque for <a href="#">Gnss_Synchro</a>	1098
<a href="#">gnss_flowgraph.h</a>	Interface of a GNSS receiver flow graph	1099
<a href="#">gnss_frequencies.h</a>	GNSS Frequencies	1099
<a href="#">gnss_obs_codes.h</a>	GNSS Observable codes	1100
<a href="#">gnss_satellite.h</a>	Interface of the <a href="#">Gnss_Satellite</a> class	1103
<a href="#">gnss_sdr_create_directory.h</a>	Create a directory	1104
<a href="#">gnss_sdr_flags.h</a>	Helper file for gnss-sdr commandline flags	1104
<a href="#">gnss_sdr_fpga_sample_counter.h</a>	Simple block to report the current receiver time based on the output of the tracking or telemetry blocks	1106
<a href="#">gnss_sdr_make_unique.h</a>	This file implements std::make_unique for C++11	1106
<a href="#">gnss_sdr_sample_counter.h</a>	Simple block to report the current receiver time based on the output of the tracking or telemetry blocks	1107
<a href="#">gnss_sdr_supl_client.h</a>	Class that implements a C++ interface to external Secure User Location Protocol (SUPL) client library	1108

<a href="#">gnss_sdr_time_counter.h</a>	
Simple block to report the current receiver time based on the output of the tracking or telemetry blocks . . . . .	1108
<a href="#">gnss_sdr_valve.h</a>	
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 . . . . .	1109
<a href="#">gnss_signal.h</a>	
Implementation of the <a href="#">Gnss_Signal</a> class . . . . .	1110
<a href="#">gnss_signal_replica.h</a>	
This library gathers a few functions used for GNSS signal replica generation regardless of system used . . . . .	1110
<a href="#">gnss_synchro.h</a>	
Interface of the <a href="#">Gnss_Synchro</a> class . . . . .	1111
<a href="#">gnss_synchro_monitor.h</a>	
Interface of a receiver monitoring block which allows sending a data stream with the receiver internal parameters ( <a href="#">Gnss_Synchro</a> objects) to local or remote clients over UDP . . . . .	1112
<a href="#">gnss_synchro_udp_sink.h</a>	
Interface of a class that sends serialized <a href="#">Gnss_Synchro</a> objects over udp to one or multiple endpoints . . . . .	1113
<a href="#">gnuplot_i.h</a>	
A C++ interface to gnuplot . . . . .	1113
<a href="#">gps_acq_assist.h</a>	
Interface of a GPS RRLL ACQUISITION ASSISTANCE storage . . . . .	1114
<a href="#">gps_almanac.h</a>	
Interface of a GPS ALMANAC storage . . . . .	1115
<a href="#">GPS_CNAV.h</a>	
Defines parameters for GPS CNAV . . . . .	1115
<a href="#">gps_cnav_ephemeris.h</a>	
Interface of a GPS CNAV EPHEMERIS storage . . . . .	1118
<a href="#">gps_cnav_iono.h</a>	
Interface of a GPS CNAV IONOSPHERIC MODEL storage . . . . .	1118
<a href="#">gps_cnav_navigation_message.h</a>	
Interface of a GPS CNAV Data message decoder . . . . .	1119
<a href="#">gps_cnav_utc_model.h</a>	
Interface of a GPS CNAV UTC MODEL storage . . . . .	1120
<a href="#">gps_ephemeris.h</a>	
Interface of a GPS EPHEMERIS storage . . . . .	1120
<a href="#">gps_iono.h</a>	
Interface of a GPS IONOSPHERIC MODEL storage . . . . .	1121
<a href="#">GPS_L1_CA.h</a>	
Defines system parameters for GPS L1 C/A signal and NAV data . . . . .	1121
<a href="#">gps_l1_ca_dll_pll_tracking.h</a>	
Interface of an adapter of a DLL+PLL tracking loop block for GPS L1 C/A to a <a href="#">TrackingInterface</a> . . . . .	1125
<a href="#">gps_l1_ca_dll_pll_tracking_fpga.h</a>	
Interface of an adapter of a DLL+PLL tracking loop block for GPS L1 C/A to a <a href="#">TrackingInterface</a> for the FPGA . . . . .	1125
<a href="#">gps_l1_ca_dll_pll_tracking_gpu.h</a>	
Implementation of an adapter of a DLL+PLL tracking loop block using GPU accelerated functions for GPS L1 C/A to a <a href="#">TrackingInterface</a> . . . . .	1126
<a href="#">gps_l1_ca_dll_pll_tracking_gpu_cc.h</a>	
Implementation of a code DLL + carrier PLL tracking block, GPU ACCELERATED . . . . .	1127
<a href="#">gps_l1_ca_kf_tracking.h</a>	
Interface of an adapter of a DLL + Kalman carrier tracking loop block for GPS L1 C/A signals . . . . .	1128
<a href="#">gps_l1_ca_kf_tracking_cc.h</a>	
Interface of a processing block of a DLL + Kalman carrier tracking loop for GPS L1 C/A signals . . . . .	1128
<a href="#">gps_l1_ca_pcps_acquisition.h</a>	
Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L1 C/A signals . . . . .	1129

<a href="#">gps_l1_ca_pcps_acquisition_fine_doppler.h</a>	
Adapts a PCPS acquisition block with fine Doppler estimation to an <a href="#">AcquisitionInterface</a> for GPS L1 C/A signals	1130
<a href="#">gps_l1_ca_pcps_acquisition_fpga.h</a>	
Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L1 C/A signals for the FPGA	1131
<a href="#">gps_l1_ca_pcps_assisted_acquisition.h</a>	
Adapts a PCPS Assisted acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L1 C/A signals	1131
<a href="#">gps_l1_ca_pcps_opencl_acquisition.h</a>	
Adapts an OpenCL PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L1 C/A signals	1132
<a href="#">gps_l1_ca_pcps_quicksync_acquisition.h</a>	
Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L1 C/A signals implementing the QuickSync Algorithm	1133
<a href="#">gps_l1_ca_pcps_tong_acquisition.h</a>	
Adapts a PCPS Tong acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L1 C/A signals	1133
<a href="#">gps_l1_ca_tcp_connector_tracking.h</a>	
Interface of an adapter of a TCP connector block based on code DLL + carrier PLL for GPS L1 C/A to a <a href="#">TrackingInterface</a>	1134
<a href="#">gps_l1_ca_tcp_connector_tracking_cc.h</a>	
Interface of a TCP connector block based on code DLL + carrier PLL	1135
<a href="#">gps_l1_ca_telemetry_decoder.h</a>	
Interface of an adapter of a GPS L1 C/A NAV data decoder block to a <a href="#">TelemetryDecoderInterface</a>	1136
<a href="#">gps_l1_ca_telemetry_decoder_gs.h</a>	
Interface of a NAV message demodulator block based on Kay Borre book MATLAB-based GPS receiver	1136
<a href="#">gps_l2_m_dll_pll_tracking.h</a>	
Interface of an adapter of a DLL+PLL tracking loop block for GPS L1 C/A to a <a href="#">TrackingInterface</a>	1137
<a href="#">gps_l2_m_dll_pll_tracking_fpga.h</a>	
Interface of an adapter of a DLL+PLL tracking loop block for GPS L2C to a <a href="#">TrackingInterface</a> for the FPGA	1138
<a href="#">gps_l2_m_pcps_acquisition.h</a>	
Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L2 M signals	1138
<a href="#">gps_l2_m_pcps_acquisition_fpga.h</a>	
Adapts an FPGA-offloaded PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L2 M signals	1139
<a href="#">GPS_L2C.h</a>	
Defines system parameters for GPS L2C signal	1139
<a href="#">gps_l2c_signal_replica.h</a>	
This file implements signal generators for GPS L2C signals	1140
<a href="#">gps_l2c_telemetry_decoder.h</a>	
Interface of an adapter of a GPS L2C (CNAV) data decoder block to a <a href="#">TelemetryDecoderInterface</a>	1141
<a href="#">gps_l2c_telemetry_decoder_gs.h</a>	
Interface of a CNAV message demodulator block	1142
<a href="#">GPS_L5.h</a>	
Defines system parameters for GPS L5 signal	1142
<a href="#">gps_l5_dll_pll_tracking.h</a>	
Interface of an adapter of a DLL+PLL tracking loop block for GPS L5 to a <a href="#">TrackingInterface</a>	1144
<a href="#">gps_l5_dll_pll_tracking_fpga.h</a>	
Interface of an adapter of a DLL+PLL tracking loop block for GPS L5 to a <a href="#">TrackingInterface</a> for the FPGA	1144
<a href="#">gps_l5_signal_replica.h</a>	
This file implements signal generators for GPS L5 signals	1145
<a href="#">gps_l5_telemetry_decoder.h</a>	
Interface of an adapter of a GPS L5 (CNAV) data decoder block to a <a href="#">TelemetryDecoderInterface</a>	1146
<a href="#">gps_l5_telemetry_decoder_gs.h</a>	
Interface of a CNAV message demodulator block	1146
<a href="#">gps_l5i_pcps_acquisition.h</a>	
Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L5i signals	1147

<a href="#">gps_l5i_pcps_acquisition_fpga.h</a>	Adapts a PCPS acquisition block to an <a href="#">AcquisitionInterface</a> for GPS L5i signals for the FPGA . . . . .	1148
<a href="#">gps_navigation_message.h</a>	Interface of a GPS NAV Data message decoder . . . . .	1148
<a href="#">gps_sdr_signal_replica.h</a>	This file implements functions for GPS L1 C/A signal replica generation . . . . .	1149
<a href="#">gps_utc_model.h</a>	Interface of a GPS UTC MODEL storage . . . . .	1150
<a href="#">gpx_printer.h</a>	Interface of a class that prints PVT information to a gpx file . . . . .	1150
<a href="#">gr_complex_ip_packet_source.h</a>	Receives ip frames containing samples in UDP frame encapsulation using a high performance packet capture library (libpcap) . . . . .	1151
<a href="#">hybrid_observables.h</a>	Implementation of an adapter of an observables block accepting all kind of signals to a <a href="#">ObservablesInterface</a> . . . . .	1151
<a href="#">hybrid_observables_gs.h</a>	Interface of the observables computation block . . . . .	1152
<a href="#">ibyte_to_cbyte.h</a>	Adapts an I/Q interleaved byte (unsigned char) sample stream into a std::complex<unsigned char> stream . . . . .	1153
<a href="#">ibyte_to_complex.h</a>	Adapts an I/Q interleaved byte integer sample stream to a gr_complex (float) stream . . . . .	1153
<a href="#">ibyte_to_cshort.h</a>	Adapts a short interleaved sample stream into a std::complex<short> stream . . . . .	1154
<a href="#">in_memory_configuration.h</a>	A <a href="#">ConfigurationInterface</a> for testing purposes . . . . .	1155
<a href="#">ini.h</a>	This function parses an INI file into easy-to-access name/value pairs . . . . .	1155
<a href="#">INIRedReader.h</a>	This class reads an INI file into easy-to-access name/value pairs . . . . .	1156
<a href="#">interleaved_byte_to_complex_byte.h</a>	Adapts an 8-bits interleaved sample stream into a 16-bits complex stream . . . . .	1157
<a href="#">interleaved_byte_to_complex_short.h</a>	Adapts a byte (8-bits) interleaved sample stream into a std::complex<short> stream . . . . .	1157
<a href="#">interleaved_short_to_complex_short.h</a>	Adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream . . . . .	1158
<a href="#">ishort_to_complex.h</a>	Adapts an I/Q interleaved short integer sample stream to a gr_complex (float) stream . . . . .	1159
<a href="#">ishort_to_cshort.h</a>	Adapts a short interleaved sample stream into a std::complex<short> stream . . . . .	1159
<a href="#">item_type_helpers.h</a>	Utility functions for converting between item types . . . . .	1160
<a href="#">kml_printer.h</a>	Interface of a class that prints PVT information to a kml file . . . . .	1161
<a href="#">labsat23_source.h</a>	Unpacks the Labsat 2 (ls2) and (ls3) capture files . . . . .	1161
<a href="#">labsat_signal_source.h</a>	Labsat 2 and 3 front-end signal sampler driver . . . . .	1162
<a href="#">lock_detectors.h</a>	Interface of a library with a set of code and carrier phase lock detectors . . . . .	1163
<a href="#">MATH_CONSTANTS.h</a>	Defines useful mathematical constants and their scaled versions . . . . .	1163
<a href="#">mmse_resampler_conditioner.h</a>	Interface of an adapter of a mmse resampler conditioner block to a <a href="#">SignalConditionerInterface</a> . . . . .	1167
<a href="#">monitor_pvt.h</a>	Interface of the <a href="#">Monitor_Pvt</a> class . . . . .	1167

<a href="#">monitor_pvt_udp_sink.h</a>	Interface of a class that sends serialized <a href="#">Monitor_Pvt</a> objects over udp to one or multiple endpoints . . . . .	1168
<a href="#">multichannel_file_signal_source.h</a>	Implementation of a class that reads signals samples from files at different frequency band and adapts it to a <a href="#">SignalSourceInterface</a> . . . . .	1169
<a href="#">nmea_printer.h</a>	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 <a href="https://www.nmea.org/">https://www.nmea.org/</a> for the NMEA 183 standard . . . . .	1169
<a href="#">nonlinear_tracking.h</a>	Interface of a library for nonlinear tracking algorithms . . . . .	1170
<a href="#">notch_cc.h</a>	Implements a notch filter algorithm . . . . .	1171
<a href="#">notch_filter.h</a>	Adapter of a multistate <a href="#">Notch</a> filter . . . . .	1171
<a href="#">notch_filter_lite.h</a>	Adapts a light version of a multistate notch filter . . . . .	1172
<a href="#">notch_lite_cc.h</a>	Implements a notch filter light algorithm . . . . .	1173
<a href="#">nsr_file_signal_source.h</a>	Implementation of a class that reads signals samples from a NSR 2 bits sampler front-end file and adapts it to a <a href="#">SignalSourceInterface</a> . More information about the front-end here <a href="http://www.ifen.com/products/sx-scientific-gnss-solutions/nsr-software-receiver.html">http://www.ifen.com/products/sx-scientific-gnss-solutions/nsr-software-receiver.html</a> . . . . .	1173
<a href="#">obs_conf.h</a>	Class that contains all the configuration parameters for generic observables block . . . . .	1174
<a href="#">obsdiff_flags.h</a>	Helper file for unit testing . . . . .	1175
<a href="#">observable_tests_flags.h</a>	Helper file for unit testing . . . . .	1175
<a href="#">observables_dump_reader.h</a>	Helper file for unit testing . . . . .	1176
<a href="#">observables_interface.h</a>	This class represents an interface to an <a href="#">Observables</a> block . . . . .	1177
<a href="#">osmosdr_signal_source.h</a>	Signal source wrapper for OsmoSDR-compatible front-ends, such as HackRF or Realtek's RTL2832U-based USB dongle DVB-T receivers (see <a href="https://osmocom.org/projects/rtl-sdr/wiki">https://osmocom.org/projects/rtl-sdr/wiki</a> for more information) . . . . .	1177
<a href="#">pass_through.h</a>	Interface of a block that just puts its input in its output . . . . .	1178
<a href="#">pcps_acquisition.h</a>	This class implements a Parallel Code Phase Search Acquisition . . . . .	1179
<a href="#">pcps_acquisition_fine_doppler_cc.h</a>	This class implements a Parallel Code Phase Search Acquisition with multi-dwells and fine Doppler estimation for GPS L1 C/A signal . . . . .	1180
<a href="#">pcps_acquisition_fpga.h</a>	This class implements a Parallel Code Phase Search Acquisition for the FPGA . . . . .	1181
<a href="#">pcps_assisted_acquisition_cc.h</a>	This class implements a Parallel Code Phase Search Acquisition with assistance and multi-dwells . . . . .	1182
<a href="#">pcps_cccwsr_acquisition_cc.h</a>	This class implements a Parallel Code Phase Search acquisition with Coherent <a href="#">Channel</a> Combining With Sign Recovery scheme . . . . .	1183
<a href="#">pcps_opencl_acquisition_cc.h</a>	This class implements a Parallel Code Phase Search Acquisition using OpenCL to offload some functions to the GPU . . . . .	1184

<a href="#">pcps_quicksync_acquisition_cc.h</a>	This class implements a Parallel Code Phase Search Acquisition with the QuickSync Algorithm	1185
<a href="#">pcps_tong_acquisition_cc.h</a>	This class implements a Parallel Code Phase Search Acquisition with Tong algorithm	1187
<a href="#">plutosdr_signal_source.h</a>	Signal source for PlutoSDR	1188
<a href="#">position_test_flags.h</a>	Helper file for unit testing	1189
<a href="#">pulse_blanking_cc.h</a>	Implements a pulse blanking algorithm	1190
<a href="#">pulse_blanking_filter.h</a>	Instantiates the GNSS-SDR pulse blanking filter	1190
<a href="#">pvt_conf.h</a>	Class that contains all the configuration parameters for the PVT block	1191
<a href="#">pvt_interface.h</a>	This class represents an interface to a PVT block	1191
<a href="#">pvt_solution.h</a>	Interface of a base class for a PVT solution	1192
<a href="#">raw_array_signal_source.h</a>	CTTC Experimental GNSS 8 channels array signal source	1193
<a href="#">rinex_printer.h</a>	Interface of a RINEX 2.11 / 3.01 printer See <a href="ftp://igs.org/pub/data/format/rinex301.pdf">ftp://igs.org/pub/data/format/rinex301.pdf</a>	1193
<a href="#">rtcm.h</a>	Interface for the RTCM 3.2 Standard	1194
<a href="#">rtcm_printer.h</a>	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	1195
<a href="#">rtklib.h</a>	Main header file for the rtklib library	1196
<a href="#">rtklib_conversions.h</a>	GNSS-SDR to RTKLIB data structures conversion functions	1205
<a href="#">rtklib_ephemeris.h</a>	Satellite ephemeris and clock functions	1206
<a href="#">rtklib_ionex.h</a>	Ionex functions	1207
<a href="#">rtklib_lambda.h</a>	Integer ambiguity resolution	1208
<a href="#">rtklib_pntpos.h</a>	Standard code-based positioning	1210
<a href="#">rtklib_ppp.h</a>	Precise Point Positioning	1213
<a href="#">rtklib_preceph.h</a>	Precise ephemeris and clock functions	1215
<a href="#">rtklib_pvt.h</a>	Interface of a Position Velocity and Time computation block	1217
<a href="#">rtklib_pvt_gs.h</a>	Interface of a Position Velocity and Time computation block	1217
<a href="#">rtklib_rtcn.h</a>	RTCM functions headers	1218
<a href="#">rtklib_rtcn2.h</a>	RTCM v2 functions headers	1219
<a href="#">rtklib_rtcn3.h</a>	RTCM v3 functions headers	1220
<a href="#">rtklib_rtkcmn.h</a>	Rtklib common functions	1224
<a href="#">rtklib_rtkpos.h</a>	Rtklib ppp-related functions	1229

<a href="#">rtklib_rtksvr.h</a>	
Rtk server functions	1231
<a href="#">rtklib_sbas.h</a>	
Sbas functions	1233
<a href="#">rtklib_solution.h</a>	
Solution functions headers	1236
<a href="#">rtklib_solver.h</a>	
PVT solver based on rtklib library functions adapted to the GNSS-SDR data flow and structures	1238
<a href="#">rtklib_solver_dump_reader.h</a>	
Helper file for unit testing	1239
<a href="#">rtklib_stream.h</a>	
Streaming functions	1239
<a href="#">rtklib_tides.h</a>	
Tidal displacement corrections	1242
<a href="#">rtl_tcp_commands.h</a>	
Defines structures and constants for communicating with rtl_tcp	1243
<a href="#">rtl_tcp_dongle_info.h</a>	
Interface for a structure sent by rtl_tcp defining the hardware	1243
<a href="#">rtl_tcp_signal_source.h</a>	
Signal source which reads from rtl_tcp. (see <a href="https://osmocom.org/projects/rtl-sdr/wiki">https://osmocom.org/projects/rtl-sdr/wiki</a> for more information)	1244
<a href="#">rtl_tcp_signal_source_c.h</a>	
Interface of an rtl_tcp signal source reader	1245
<a href="#">sbas_ephemeris.h</a>	
Interface of a SBAS REFERENCE LOCATION storage	1246
<a href="#">sbas_l1_telemetry_decoder.h</a>	
Interface of an adapter of a SBAS telemetry data decoder block to a <a href="#">TelemetryDecoderInterface</a>	1246
<a href="#">sbas_l1_telemetry_decoder_gs.h</a>	
Interface of a SBAS telemetry data decoder block	1247
<a href="#">serdes_gnss_synchro.h</a>	
Serialization / Deserialization of <a href="#">Gnss_Synchro</a> objects using Protocol Buffers	1248
<a href="#">serdes_monitor_pvt.h</a>	
Serialization / Deserialization of <a href="#">Monitor_Pvt</a> objects using Protocol Buffers	1248
<a href="#">short_x2_to_cshort.h</a>	
Adapts two short streams into a <code>std::complex&lt;short&gt;</code> stream	1249
<a href="#">signal_conditioner.h</a>	
It wraps blocks to change data type, filter and resample input data	1250
<a href="#">signal_generator.h</a>	
Adapter of a class that generates synthesized GNSS signal	1250
<a href="#">signal_generator_c.h</a>	
GNU Radio source block that generates synthesized GNSS signal	1251
<a href="#">signal_generator_flags.h</a>	
Helper file for unit testing	1252
<a href="#">spir_file_signal_source.h</a>	
Implementation of a class that reads signals samples from a SPIR file and adapts it to a <a href="#">SignalSourceInterface</a>	1253
<a href="#">spir_gss6450_file_signal_source.h</a>	
Implementation of a class that reads signals samples from a SPIR file and adapts it to a <a href="#">SignalSourceInterface</a>	1254
<a href="#">spirent_motion_csv_dump_reader.h</a>	
Helper file for unit testing	1254
<a href="#">string_converter.h</a>	
Interface of a class that interprets the contents of a string and converts it into different types	1255
<a href="#">swift_common.h</a>	
Common definitions used throughout the libswiftnav library	1255
<a href="#">tcp_cmd_interface.h</a>	
Class that implements a TCP/IP telecommand command line interface for GNSS-SDR	1256

<a href="#">tcp_communication.h</a>	Interface of the TCP communication class . . . . .	1257
<a href="#">tcp_packet_data.h</a>	Interface of the TCP data packet class . . . . .	1258
<a href="#">telemetry_decoder_interface.h</a>	This class represents an interface to a telemetry decoder block . . . . .	1258
<a href="#">test_flags.h</a>	Helper file for unit testing . . . . .	1259
<a href="#">tlm_conf.h</a>	Class that contains all the configuration parameters for generic telemetry decoder block . . . . .	1259
<a href="#">tlm_dump_reader.h</a>	Helper file for unit testing . . . . .	1260
<a href="#">tlm_utils.h</a>	Utilities for the telemetry decoder blocks . . . . .	1260
<a href="#">tracking_2nd_DLL_filter.h</a>	Interface of a 2nd order DLL filter for code tracking loop . . . . .	1261
<a href="#">tracking_2nd_PLL_filter.h</a>	Interface of a 2nd order PLL filter for carrier tracking loop . . . . .	1261
<a href="#">tracking_discriminators.h</a>	Interface of a library with a set of code tracking and carrier tracking discriminators . . . . .	1262
<a href="#">tracking_dump_reader.h</a>	Helper file for unit testing . . . . .	1263
<a href="#">tracking_FLL_PLL_filter.h</a>	Interface of a hybrid FLL and PLL filter for tracking carrier loop . . . . .	1263
<a href="#">tracking_interface.h</a>	This class represents an interface to a tracking block . . . . .	1264
<a href="#">tracking_loop_filter.h</a>	Generic 1st to 3rd order loop filter implementation . . . . .	1264
<a href="#">tracking_tests_flags.h</a>	Helper file for unit testing . . . . .	1265
<a href="#">tracking_true_obs_reader.h</a>	Helper file for unit testing . . . . .	1266
<a href="#">true_observables_reader.h</a>	Helper file for unit testing . . . . .	1267
<a href="#">two_bit_cpx_file_signal_source.h</a>	Interface of a class that reads signals samples from a 2 bit complex sampler front-end file and adapts it to a SignalSourceInterface . . . . .	1267
<a href="#">two_bit_packed_file_signal_source.h</a>	Interface of a class that reads signals samples from a file. Each sample is two bits, which are packed into bytes or shorts . . . . .	1268
<a href="#">uhd_signal_source.h</a>	Interface for the Universal Hardware Driver signal source . . . . .	1269
<a href="#">uio_fpga.h</a>	This library contains functions to determine the uio device driver file that corresponds to a hardware accelerator device name in the FPGA . . . . .	1269
<a href="#">unpack_2bit_samples.h</a>	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 . . . . .	1270
<a href="#">unpack_byte_2bit_cpx_samples.h</a>	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 . . . . .	1271
<a href="#">unpack_byte_2bit_samples.h</a>	Unpacks byte samples to NSR 2 bits samples . . . . .	1272

<a href="#">unpack_byte_4bit_samples.h</a>	
Unpacks byte samples to 4 bits samples. Packing Order Packing order in Nibble I0 I1 I2 I3 I0 I1 I2 I3 . . . . .	1273
<a href="#">unpack_intspir_1bit_samples.h</a>	
Unpacks SPIR int samples to NSR 1 bit samples . . . . .	1274
<a href="#">unpack_spir_gss6450_samples.h</a>	
Unpacks SPIR int samples . . . . .	1274
<a href="#">viterbi_decoder.h</a>	
Interface of a Viterbi decoder class based on the Iterative Solutions Coded Modulation Library by Matthew C. Valenti . . . . .	1275



## Chapter 9

# Module Documentation

### 9.1 Acquisition

#### Modules

- [acquisition\\_adapters](#)
- [acquisition\\_gr\\_blocks](#)
- [acquisition\\_libs](#)

#### 9.1.1 Detailed Description

Classes for GNSS signal acquisition

## 9.2 acquisition\_adapters

### Classes

- class [BeidouB1iPcpsAcquisition](#)  
This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.
- class [BeidouB3iPcpsAcquisition](#)  
This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for BeiDou B3I signals.
- class [GalileoE1Pcps8msAmbiguousAcquisition](#)  
Adapts a PCPS 8ms acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.
- class [GalileoE1PcpsAmbiguousAcquisition](#)  
This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.
- class [GalileoE1PcpsAmbiguousAcquisitionFpga](#)  
This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for Galileo E1 Signals.
- class [GalileoE1PcpsCccwsrAmbiguousAcquisition](#)  
Adapts a PCPS CCCWSR acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.
- class [GalileoE1PcpsQuickSyncAmbiguousAcquisition](#)  
This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.
- class [GalileoE1PcpsTongAmbiguousAcquisition](#)  
Adapts a PCPS Tong acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.
- class [GalileoE5aNoncoherentIQAcquisitionCaf](#)
- class [GalileoE5aPcpsAcquisition](#)
- class [GalileoE5aPcpsAcquisitionFpga](#)  
This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for Galileo E5a signals.
- class [GalileoE5bPcpsAcquisition](#)
- class [GalileoE5bPcpsAcquisitionFpga](#)  
This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for Galileo E5b signals.
- class [GalileoE6PcpsAcquisition](#)  
This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E6 Signals.
- class [GlonassL1CaPcpsAcquisition](#)  
This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.
- class [GlonassL2CaPcpsAcquisition](#)  
This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GLONASS L2 C/A signals.
- class [GpsL1CaPcpsAcquisition](#)  
This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.
- class [GpsL1CaPcpsAcquisitionFineDoppler](#)  
This class Adapts a PCPS acquisition block with fine Doppler estimation to an [AcquisitionInterface](#) for GPS L1 C/A signals.
- class [GpsL1CaPcpsAcquisitionFpga](#)  
This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for GPS L1 C/A signals.
- class [GpsL1CaPcpsAssistedAcquisition](#)  
This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.
- class [GpsL1CaPcpsOpenCLAcquisition](#)  
This class adapts an OpenCL PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.
- class [GpsL1CaPcpsQuickSyncAcquisition](#)  
This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.
- class [GpsL1CaPcpsTongAcquisition](#)  
This class adapts a PCPS Tong acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.
- class [GpsL2MPcpsAcquisition](#)  
This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L2 M signals.
- class [GpsL2MPcpsAcquisitionFpga](#)

*This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for GPS L2 M signals.*

- class [GpsL5iPcpsAcquisition](#)

*This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L5i signals.*

- class [GpsL5iPcpsAcquisitionFpga](#)

*This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for GPS L5i signals.*

## Typedefs

- using `pcps_acquisition_fine_doppler_cc_sptr` = `gnss_shared_ptr`< [pcps\\_acquisition\\_fine\\_doppler\\_cc](#) >

### 9.2.1 Detailed Description

Wrap GNU Radio acquisition blocks with an [AcquisitionInterface](#)

## 9.3 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.*
- struct [pcpsconf\\_fpga\\_t](#)
- 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\\_cc](#) >
- 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** (unsigned int sampled\_ms, unsigned int max\_dwells, unsigned int doppler\_max, int64\_t fs\_in, int samples\_per\_ms, int samples\_per\_code, bool bit\_transition\_flag, bool dump, const std::string &dump\_filename, bool both\_signal\_components\_, int CAF\_window\_hz\_, int Zero\_padding\_, bool enable\_monitor\_output)
- `galileo_pcps_8ms_acquisition_cc_sptr` **galileo\_pcps\_8ms\_make\_acquisition\_cc** (uint32\_t sampled\_ms, uint32\_t max\_dwells, uint32\_t doppler\_max, int64\_t fs\_in, int32\_t samples\_per\_ms, int32\_t samples\_per\_code, bool dump, const std::string &dump\_filename, bool enable\_monitor\_output)
- `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** ([pcpsconf\\_fpga\\_t](#) conf\_)
- `pcps_assisted_acquisition_cc_sptr` **pcps\_make\_assisted\_acquisition\_cc** (int32\_t max\_dwells, uint32\_t sampled\_ms, int32\_t doppler\_max, int32\_t doppler\_min, int64\_t fs\_in, int32\_t samples\_per\_ms, bool dump, const std::string &dump\_filename, bool enable\_monitor\_output)
- `pcps_cccwsr_acquisition_cc_sptr` **pcps\_cccwsr\_make\_acquisition\_cc** (uint32\_t sampled\_ms, uint32\_t max\_dwells, uint32\_t doppler\_max, int64\_t fs\_in, int32\_t samples\_per\_ms, int32\_t samples\_per\_code, bool dump, const std::string &dump\_filename, bool enable\_monitor\_output)
- `pcps_opencl_acquisition_cc_sptr` **pcps\_make\_opencl\_acquisition\_cc** (uint32\_t sampled\_ms, uint32\_t max\_dwells, uint32\_t doppler\_max, int64\_t fs\_in, int samples\_per\_ms, int samples\_per\_code, bool bit\_transition\_flag, bool dump, const std::string &dump\_filename, bool enable\_monitor\_output)
- `pcps_quicksync_acquisition_cc_sptr` **pcps\_quicksync\_make\_acquisition\_cc** (uint32\_t folding\_factor, uint32\_t sampled\_ms, uint32\_t max\_dwells, uint32\_t doppler\_max, int64\_t fs\_in, int32\_t samples\_per\_ms, int32\_t samples\_per\_code, bool bit\_transition\_flag, bool dump, const std::string &dump\_filename, bool enable\_monitor\_output)
- `pcps_tong_acquisition_cc_sptr` **pcps\_tong\_make\_acquisition\_cc** (uint32\_t sampled\_ms, uint32\_t doppler\_max, int64\_t fs\_in, int32\_t samples\_per\_ms, int32\_t samples\_per\_code, uint32\_t tong\_init\_val, uint32\_t tong\_max\_val, uint32\_t tong\_max\_dwells, bool dump, const std::string &dump\_filename, bool enable\_monitor\_output)

### 9.3.1 Detailed Description

GNU Radio processing blocks for GNSS signal acquisition

## 9.4 acquisition\_libs

### Classes

- class [Acq\\_Conf](#)
- class [Fpga\\_Acquisition](#)

*Class that implements carrier wipe-off and correlators.*

### 9.4.1 Detailed Description

Library with utilities for GNSS signal acquisition

## 9.5 Channel

### Modules

- [channel\\_adapters](#)
- [channel\\_libs](#)

### 9.5.1 Detailed Description

Classes containing a GNSS channel.

## 9.6 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.*

#### 9.6.1 Detailed Description

Classes that wrap an [AcquisitionInterface](#), a [TrackingInterface](#) and a [TelemetryDecoderInterface](#), and handles their interaction.

## 9.7 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)

#### 9.7.1 Detailed Description

Library with utilities for a GNSS [Channel](#).

## 9.8 Signal Conditioner

### Modules

- [conditioner\\_adapters](#)

### 9.8.1 Detailed Description

Signal Conditioner wrapper block

## 9.9 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.*

### 9.9.1 Detailed Description

Wrap a Signal Conditioner with a [GNSSBlockInterface](#)

## 9.10 Data Type Adapters

### Modules

- [data\\_type\\_adapters](#)
- [data\\_type\\_gr\\_blocks](#)

### 9.10.1 Detailed Description

Classes for data type conversion

## 9.11 data\_type\_adapters

### Classes

- class [ByteToShort](#)  
*Adapts an 8-bits sample stream (IF) to a short int stream (IF)*
- class [lbyteToCbyte](#)
- class [lbyteToComplex](#)  
*Adapts an I/Q interleaved byte integer sample stream to a gr\_complex (float) stream.*
- class [lbyteToCshort](#)  
*Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream.*
- class [lshortToComplex](#)  
*Adapts an I/Q interleaved short integer sample stream to a gr\_complex (float) stream.*
- class [lshortToCshort](#)  
*Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream.*

### 9.11.1 Detailed Description

Wrap GNU Radio data type adapter blocks with a [GNSSBlockInterface](#)

## 9.12 data\_type\_gr\_blocks

### 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>)*
- 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 **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

- 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** ()

#### 9.12.1 Detailed Description

GNU Radio Blocks for data type conversion

## 9.13 Input Filter

### Modules

- [input\\_filter\\_adapters](#)
- [input\\_filter\\_gr\\_blocks](#)

### 9.13.1 Detailed Description

Classes for input signal filtering

## 9.14 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](#)

### 9.14.1 Detailed Description

Classes that wrap GNU Radio input filters with a [GNSSBlockInterface](#)

## 9.15 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

#### 9.15.1 Detailed Description

GNU Radio blocks implementing input filters,

## 9.16 Algorithms Common Library

### Modules

- [algorithms\\_libs](#)
- [gnss\\_sdr\\_flags](#)

### 9.16.1 Detailed Description

Common utilities for the GNSS receiver.

## 9.17 algorithms\_libs

### 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)*
- class [short\\_x2\\_to\\_cshort](#)  
*This class adapts two short streams into a `std::complex<short>` stream.*

### 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\\_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\\_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 ellipsoid\_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 [clkisin](#) (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\\_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\\_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\\_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\\_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.*
- 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.*

### 9.17.1 Detailed Description

Common utilities for GNSS algorithms.

### 9.17.2 Function Documentation

#### 9.17.2.1 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`.

#### 9.17.2.2 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`.

#### 9.17.2.3 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.

#### 9.17.2.4 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.

#### 9.17.2.5 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.

#### 9.17.2.6 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.

#### 9.17.2.7 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.

#### 9.17.2.8 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.

#### 9.17.2.9 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.

#### 9.17.2.10 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.

#### 9.17.2.11 cart2geo()

```
arma::vec cart2geo (
    const arma::vec & XYZ,
    int ellipsoid_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 0. International Ellipsoid 1924

1. International Ellipsoid 1967
2. World Geodetic System 1972
3. Geodetic Reference System 1980
4. World Geodetic System 1984

#### 9.17.2.12 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'.

#### 9.17.2.13 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.

**9.17.2.14 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.

**9.17.2.15 clsin()**

```
double clsin (
    const arma::colvec & ar,
    int degree,
    double argument )
```

Clenshaw summation of sinus of argument.

**9.17.2.16 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.

**9.17.2.17 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.

#### 9.17.2.18 findUtmZone()

```
int findUtmZone (
    double latitude_deg,
    double longitude_deg )
```

Function finds the UTM zone number for given longitude and latitude.

#### 9.17.2.19 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.

#### 9.17.2.20 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,
    bool secondary_flag )
```

This function generates Galileo E1 code (can select E1B or E1C, cboc or sinboc and the sample frequency `sampling_freq`).

#### 9.17.2.21 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 )
```

`galileo_e1_code_gen_complex_sampled` without `secondary_flag` for backward compatibility.

**9.17.2.22 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,
    bool secondary_flag )
```

This function generates Galileo E1 code (can select E1B or E1C, cboc or sinboc and the sample frequency `sampling_freq`).

**9.17.2.23 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 )
```

This function generates Galileo E1 code (can select E1B or E1C, cboc or sinboc and the sample frequency `sampling_freq`).

**9.17.2.24 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).

**9.17.2.25 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.

#### 9.17.2.26 `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`.

#### 9.17.2.27 `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.

#### 9.17.2.28 `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`.

#### 9.17.2.29 `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.

**9.17.2.30 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`.

**9.17.2.31 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.

**9.17.2.32 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.

**9.17.2.33 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`.

**9.17.2.34 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.

**9.17.2.35 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.

**9.17.2.36 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.

**9.17.2.37 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.

**9.17.2.38 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

**9.17.2.39 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.

**9.17.2.40 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.

**9.17.2.41 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.

**9.17.2.42 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.

**9.17.2.43 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.

**9.17.2.44 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.

**9.17.2.45 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.

**9.17.2.46 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.

**9.17.2.47 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.

**9.17.2.48 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.

**9.17.2.49 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.

**9.17.2.50 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.

**9.17.2.51 gps\_l2c\_m\_code\_gen\_complex\_sampled()**

```
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.

**9.17.2.52 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.

#### 9.17.2.53 `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.

#### 9.17.2.54 `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.

#### 9.17.2.55 `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.

#### 9.17.2.56 `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.

#### 9.17.2.57 `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.

**9.17.2.58 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.

**9.17.2.59 Gravity\_ECEF()**

```
arma::vec Gravity_ECEF (
    const arma::vec & r_eb_e )
```

Calculates acceleration due to gravity resolved about ECEF-frame.

**9.17.2.60 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.

**9.17.2.61 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).

**9.17.2.62 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).

**9.17.2.63 item\_type\_is\_complex()**

```
bool item_type_is_complex (
    const std::string & item_type )
```

Determine if an `item_type` is complex.

**9.17.2.64 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.

**9.17.2.65 item\_type\_valid()**

```
bool item_type_valid (
    const std::string & item_type )
```

Check if a string is a valid item type.

Valid item types include: "byte", "short", "float", "ibyte", "ishort", "cbyte", "cshort", "gr\_complex"

**9.17.2.66 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`.

Provides a generic interface to generate conversion functions for mapping arrays of items.

**Parameters**

<i>input_type</i>	- String representation of the input item type
<i>output_type</i>	- 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 );`

##### 9.17.2.67 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`.

##### 9.17.2.68 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`.

##### 9.17.2.69 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)

**9.17.2.70 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.

**9.17.2.71 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.

**9.17.2.72 reset()** [1/2]

```
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`.

**9.17.2.73 reset()** [2/2]

```
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`.

**9.17.2.74 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.

**9.17.2.75 Skew\_symmetric()**

```
arma::mat Skew_symmetric (
    const arma::vec & a )
```

Calculates skew-symmetric matrix.

**9.17.2.76 togeod()**

```
int togeod (
    double * dphi,
    double * dlambd,
    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

### 9.17.2.77 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

## 9.18 gnss\_sdr\_flags

### Functions

- [DECLARE\\_string](#) (c)  
*Path to the configuration file.*
- [DECLARE\\_string](#) (config\_file)  
*Path to the configuration file.*
- [DECLARE\\_string](#) (log\_dir)  
*Path to the folder in which logging will be stored.*
- [DECLARE\\_string](#) (s)  
*Path to the file containing the signal samples.*
- [DECLARE\\_string](#) (signal\_source)  
*Path to the file containing the signal samples.*
- [DECLARE\\_bool](#) (rf\_shutdown)  
*Shutdown RF when program exits.*
- [DECLARE\\_int32](#) (doppler\_max)  
*If defined, maximum Doppler value in the search grid, in Hz (overrides the configuration file).*
- [DECLARE\\_int32](#) (doppler\_step)  
*If defined, sets the frequency step in the search grid, in Hz, in Hz (overrides the configuration file).*
- [DECLARE\\_int32](#) (cn0\_samples)  
*Number of correlator outputs used for CN0 estimation.*
- [DECLARE\\_int32](#) (cn0\_min)  
*Minimum valid CN0 (in dB-Hz).*
- [DECLARE\\_int32](#) (max\_lock\_fail)  
*Maximum number of code lock failures before dropping a satellite.*
- [DECLARE\\_int32](#) (max\_carrier\_lock\_fail)  
*Maximum number of carrier lock failures before dropping a satellite.*
- [DECLARE\\_double](#) (carrier\_lock\_th)  
*Carrier lock threshold (in rad).*
- [DECLARE\\_double](#) (dll\_bw\_hz)  
*Bandwidth of the DLL low pass filter, in Hz (overrides the configuration file).*
- [DECLARE\\_double](#) (pll\_bw\_hz)  
*Bandwidth of the PLL low pass filter, in Hz (overrides the configuration file).*
- [DECLARE\\_int32](#) (carrier\_smoothing\_factor)  
*Sets carrier smoothing factor M (overrides the configuration file).*
- [DECLARE\\_string](#) (RINEX\_version)  
*If defined, specifies the RINEX version (2.11 or 3.02). Overrides the configuration file.*
- [DECLARE\\_string](#) (RINEX\_name)  
*If defined, specifies the RINEX files base name.*

### Variables

- `const int32_t DEFAULT_CARRIER_SMOOTHING_FACTOR = 200`

#### 9.18.1 Detailed Description

Library for command-line handling.

## 9.18.2 Function Documentation

### 9.18.2.1 DECLARE\_bool()

```
DECLARE_bool (
    rf_shutdown )
```

Shutdown RF when program exits.

### 9.18.2.2 DECLARE\_double() [1/3]

```
DECLARE_double (
    carrier_lock_th )
```

Carrier lock threshold (in rad).

### 9.18.2.3 DECLARE\_double() [2/3]

```
DECLARE_double (
    dll_bw_hz )
```

Bandwidth of the DLL low pass filter, in Hz (overrides the configuration file).

### 9.18.2.4 DECLARE\_double() [3/3]

```
DECLARE_double (
    pll_bw_hz )
```

Bandwidth of the PLL low pass filter, in Hz (overrides the configuration file).

### 9.18.2.5 DECLARE\_int32() [1/7]

```
DECLARE_int32 (
    doppler_max )
```

If defined, maximum Doppler value in the search grid, in Hz (overrides the configuration file).

**9.18.2.6 DECLARE\_int32()** [2/7]

```
DECLARE_int32 (
    doppler_step )
```

If defined, sets the frequency step in the search grid, in Hz, in Hz (overrides the configuration file).

**9.18.2.7 DECLARE\_int32()** [3/7]

```
DECLARE_int32 (
    cn0_samples )
```

Number of correlator outputs used for CN0 estimation.

**9.18.2.8 DECLARE\_int32()** [4/7]

```
DECLARE_int32 (
    cn0_min )
```

Minimum valid CN0 (in dB-Hz).

**9.18.2.9 DECLARE\_int32()** [5/7]

```
DECLARE_int32 (
    max_lock_fail )
```

Maximum number of code lock failures before dropping a satellite.

**9.18.2.10 DECLARE\_int32()** [6/7]

```
DECLARE_int32 (
    max_carrier_lock_fail )
```

Maximum number of carrier lock failures before dropping a satellite.

**9.18.2.11 DECLARE\_int32()** [7/7]

```
DECLARE_int32 (
    carrier_smoothing_factor )
```

Sets carrier smoothing factor M (overrides the configuration file).

**9.18.2.12 DECLARE\_string()** [1/7]

```
DECLARE_string (  
    c )
```

Path to the configuration file.

**9.18.2.13 DECLARE\_string()** [2/7]

```
DECLARE_string (  
    config_file )
```

Path to the configuration file.

**9.18.2.14 DECLARE\_string()** [3/7]

```
DECLARE_string (  
    log_dir )
```

Path to the folder in which logging will be stored.

**9.18.2.15 DECLARE\_string()** [4/7]

```
DECLARE_string (  
    s )
```

Path to the file containing the signal samples.

**9.18.2.16 DECLARE\_string()** [5/7]

```
DECLARE_string (  
    signal_source )
```

Path to the file containing the signal samples.

**9.18.2.17 DECLARE\_string()** [6/7]

```
DECLARE_string (  
    RINEX_version )
```

If defined, specifies the RINEX version (2.11 or 3.02). Overrides the configuration file.

**9.18.2.18 DECLARE\_string()** [7/7]

```
DECLARE_string (  
    RINEX_name )
```

If defined, specifies the RINEX files base name.

## 9.19 PVT

### Modules

- [algorithms\\_libs\\_rtklib](#)
- [pvt\\_adapters](#)
- [pvt\\_gr\\_blocks](#)
- [pvt\\_libs](#)

### 9.19.1 Detailed Description

Computation of Position, Velocity and Time from GNSS observables.

## 9.20 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 [sbsfcrr\\_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) pthread\_mutex\_init(f, NULL)
- #define **rtk\_lock**(f) pthread\_mutex\_lock(f)
- #define **rtk\_unlock**(f) pthread\_mutex\_unlock(f)
- #define **VER\_RTKLIB** "2.4.2"
- #define **NTRIP\_AGENT** "RTKLIB/" VER\_RTKLIB
- #define **NTRIP\_CLI\_PORT** 2101 /\* default ntrip-client connection port \*/
- #define **NTRIP\_SVR\_PORT** 80 /\* default ntrip-server connection port \*/
- #define **NTRIP\_MAXRSP** 32768 /\* max size of ntrip response \*/
- #define **NTRIP\_MAXSTR** 256 /\* max length of mountpoint string \*/
- #define **NTRIP\_RSP\_OK\_CLI** "ICY 200 OK\r\n" /\* ntrip response: client \*/
- #define **NTRIP\_RSP\_OK\_SVR** "OK\r\n" /\* ntrip response: server \*/
- #define **NTRIP\_RSP\_SRCTBL** "SOURCETABLE 200 OK\r\n" /\* ntrip response: source table \*/
- #define **NTRIP\_RSP\_TBLEND** "ENDSOURCETABLE"
- #define **NTRIP\_RSP\_HTTP** "HTTP/" /\* ntrip response: http \*/
- #define **NTRIP\_RSP\_ERROR** "ERROR" /\* ntrip response: error \*/
- #define **FTP\_CMD** "wget" /\* ftp/http command \*/
- #define **ENAGLO**
- #define **ENABDS**
- #define **STR\_MODE\_R** 0x1 /\* stream mode: read \*/
- #define **STR\_MODE\_W** 0x2 /\* stream mode: write \*/
- #define **STR\_MODE\_RW** 0x3 /\* stream mode: read/write \*/
- #define **STR\_NONE** 0 /\* stream type: none \*/
- #define **STR\_SERIAL** 1 /\* stream type: serial \*/
- #define **STR\_FILE** 2 /\* stream type: file \*/
- #define **STR\_TCPSVR** 3 /\* stream type: TCP server \*/
- #define **STR\_TCPCLI** 4 /\* stream type: TCP client \*/
- #define **STR\_UDP** 5 /\* stream type: UDP stream \*/
- #define **STR\_NTRIPSVR** 6 /\* stream type: NTRIP server \*/
- #define **STR\_NTRIPCLI** 7 /\* stream type: NTRIP client \*/
- #define **STR\_FTP** 8 /\* stream type: ftp \*/
- #define **STR\_HTTP** 9 /\* stream type: http \*/
- #define **NP\_PPP**(opt) ((opt)->dynamics ? 9 : 3) /\* number of pos solution \*/
- #define **IC\_PPP**(s, opt) (NP\_PPP(opt) + (s)) /\* state index of clocks (s=0:gps,1:glo) \*/
- #define **IT\_PPP**(opt) (IC\_PPP(0, opt) + **NSYS**) /\* state index of tropos \*/

- `#define NR_PPP(opt) (IT_PPP(opt) + ((opt)->tropopt < TROPOPT_EST ? 0 : ((opt)->tropopt == TROPOPT_EST ? 1 : 3)))` /\* number of solutions \*/
- `#define IB_PPP(s, opt) (NR_PPP(opt) + (s)-1)` /\* state index of phase bias \*/
- `#define NX_PPP(opt) (IB_PPP(MAXSAT, opt) + 1)` /\* number of estimated states \*/
- `#define NF_RTK(opt) ((opt)->ionoopt == IONOOPT_IFLC ? 1 : (opt)->nf)`
- `#define NP_RTK(opt) ((opt)->dynamics == 0 ? 3 : 9)`
- `#define NI_RTK(opt) ((opt)->ionoopt != IONOOPT_EST ? 0 : MAXSAT)`
- `#define NT_RTK(opt) ((opt)->tropopt < TROPOPT_EST ? 0 : ((opt)->tropopt < TROPOPT_ESTG ? 2 : 6))`
- `#define NL_RTK(opt) ((opt)->glomodear != 2 ? 0 : NFREQGLO)`
- `#define NB_RTK(opt) ((opt)->mode <= PMODE_DGPS ? 0 : MAXSAT * NF_RTK(opt))`
- `#define NR_RTK(opt) (NP_RTK(opt) + NI_RTK(opt) + NT_RTK(opt) + NL_RTK(opt))`
- `#define NX_RTK(opt) (NR_RTK(opt) + NB_RTK(opt))`
- `#define II_RTK(s, opt) (NP_RTK(opt) + (s)-1)` /\* ionos (s:satellite no) \*/
- `#define IT_RTK(r, opt) (NP_RTK(opt) + NI_RTK(opt) + NT_RTK(opt) / 2 * (r))` /\* tropos (r:0=rov,1:ref) \*/
- `#define IL_RTK(f, opt) (NP_RTK(opt) + NI_RTK(opt) + NT_RTK(opt) + (f))` /\* receiver h/w bias \*/
- `#define IB_RTK(s, f, opt) (NR_RTK(opt) + MAXSAT * (f) + (s)-1)` /\* phase bias (s:satno,f:freq) \*/
- `#define COMMENTH "%" /* comment line indicator for solution */`
- `#define MSG_DISCONN "$_DISCONNECT\r\n" /* disconnect message */`

## Typedefs

- using `fatafunc_t` = void(const char \*)  
*fatal callback function type*
- typedef struct `half_cyc_tag` `half_cyc_t`

## Functions

- `eph_t eph_to_rtklib` (const `Galileo_Ephemeris` &gal\_eph)
- `eph_t eph_to_rtklib` (const `Gps_Ephemeris` &gps\_eph, bool pre\_2009\_file)
- `eph_t eph_to_rtklib` (const `Gps_CNAV_Ephemeris` &gps\_cnav\_eph)
- `eph_t eph_to_rtklib` (const `Beidou_Dnav_Ephemeris` &bei\_eph)
- `alm_t alm_to_rtklib` (const `Gps_Almanac` &gps\_alm)
- `alm_t alm_to_rtklib` (const `Galileo_Almanac` &gal\_alm)
- `geph_t eph_to_rtklib` (const `Glonass_Gnav_Ephemeris` &glonass\_gnav\_eph, const `Glonass_Gnav_Utc_Model` &gnav\_clock\_model)  
*Transforms a `Glonass_Gnav_Ephemeris` to its RTKLIB counterpart.*
- `obsd_t insert_obs_to_rtklib` (`obsd_t` &rtklib\_obs, const `Gnss_Synchro` &gnss\_synchro, int week, int band, bool pre\_2009\_file=false)
- 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 `sdocs` (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 bl, double dt, int f, const `prcopt_t` \*opt)
- 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, int i, 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 **zdras\_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 **zdras** (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 **gloibcorr** (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 *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 **ddmat** (*rtk\_t* \*rtk, double \*D)
- 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 **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 **outsolsex** (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 **outsolsex** (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-kinematic*
- 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 **MINPRNQZS** = 0
- const int **MAXPRNQZS** = 0
- const int **MINPRNQZS\_S** = 0
- const int **MAXPRNQZS\_S** = 0
- const int **NSATQZS** = 0
- const int **NSYSQZS** = 0
- const int **MINPRNBDS** = 1  
*min satellite sat number of BeiDou*
- const int **MAXPRNBDS** = 37  
*max satellite sat number of BeiDou*
- const int **NSATBDS** = (**MAXPRNBDS** - **MINPRNBDS** + 1)  
*number of BeiDou satellites*

- const int **NSYSBDS** = 1
- const int **MINPRNIRN** = 0
- const int **MAXPRNIRN** = 0
- const int **NSATIRN** = 0
- const int **NSYSIRN** = 0
- const int **MINPRNLEO** = 0
- const int **MAXPRNLEO** = 0
- 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 **MAXOBS** = 64  
*max number of obs type in RINEX*
- const double **MAXD** = 7200.0  
*max time difference to GPS Toe (s)*
- const double **MAXD\_QZS** = 7200.0  
*max time difference to QZSS Toe (s)*
- const double **MAXD\_GAL** = 10800.0  
*max time difference to Galileo Toe (s)*
- const double **MAXD\_BDS** = 21600.0  
*max time difference to BeiDou Toe (s)*
- const double **MAXD\_GLO** = 1800.0  
*max time difference to GLONASS Toe (s)*
- const double **MAXD\_SBS** = 360.0  
*max time difference to SBAS Toe (s)*
- const double **MAXD\_S** = 86400.0  
*max time difference to ephemeris 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 ionosphere*
- 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*
- const double EFACT\_GPS = 1.0
- error factor: GPS*
- const double EFACT\_GLO = 1.5
- error factor: GLONASS*

- const double **EFACT\_GAL** = 1.0  
*error factor: Galileo*
- const double **EFACT\_QZS** = 1.0  
*error factor: QZSS*
- const double **EFACT\_BDS** = 1.0  
*error factor: BeiDou*
- const double **EFACT\_IRN** = 1.5  
*error factor: IRNSS*
- const 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**)

### 9.20.1 Detailed Description

Our version of the RTKLIB core library (see <http://www.rtklib.com/>)

### 9.20.2 Typedef Documentation

#### 9.20.2.1 fatalfunc\_t

```
using fatalfunc_t = void(const char *)
```

fatal callback function type

Definition at line 327 of file rtklib.h.

### 9.20.3 Function Documentation

#### 9.20.3.1 eph\_to\_rtklib()

```
geph_t eph_to_rtklib (
    const Glonass_Gnav_Ephemeris & glonass_gnav_eph,
    const Glonass_Gnav_Utc_Model & gnav_clock_model )
```

Transforms a [Glonass\\_Gnav\\_Ephemeris](#) to its RTKLIB counterpart.

##### Parameters

<i>glonass_gnav_eph</i>	GLONASS GNAV Ephemeris structure
-------------------------	----------------------------------

##### Returns

Ephemeris structure for RTKLIB parsing

### 9.20.4 Variable Documentation

#### 9.20.4.1 ARMODE\_CONT

```
const int ARMODE_CONT = 1
```

AR mode: continuous.

Definition at line 314 of file rtklib.h.

#### 9.20.4.2 ARMODE\_FIXHOLD

```
const int ARMODE_FIXHOLD = 3
```

AR mode: fix and hold.

Definition at line 316 of file rtklib.h.

#### 9.20.4.3 ARMODE\_INST

```
const int ARMODE_INST = 2
```

AR mode: instantaneous.

Definition at line 315 of file rtklib.h.

#### 9.20.4.4 ARMODE\_OFF

```
const int ARMODE_OFF = 0
```

AR mode: off.

Definition at line 313 of file rtklib.h.

#### 9.20.4.5 ARMODE\_PPPAR

```
const int ARMODE_PPPAR = 4
```

AR mode: PPP-AR.

Definition at line 317 of file rtklib.h.

#### 9.20.4.6 ARMODE\_PPPAR\_ILS

```
const int ARMODE_PPPAR_ILS = 5
```

AR mode: AR mode: PPP-AR ILS.

Definition at line 318 of file rtklib.h.

#### 9.20.4.7 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 1292 of file rtklib.h.

#### 9.20.4.8 DTTOL

```
const double DTTOL = 0.005
```

tolerance of time difference (s)

Definition at line 146 of file rtklib.h.

#### 9.20.4.9 EFACT\_BDS

```
const double EFACT_BDS = 1.0
```

error factor: BeiDou

Definition at line 305 of file rtklib.h.

#### 9.20.4.10 EFACT\_GAL

```
const double EFACT_GAL = 1.0
```

error factor: Galileo

Definition at line 303 of file rtklib.h.

#### 9.20.4.11 EFACT\_GLO

```
const double EFACT_GLO = 1.5
```

error factor: GLONASS

Definition at line 302 of file rtklib.h.

#### 9.20.4.12 EFACT\_GPS

```
const double EFACT_GPS = 1.0
```

error factor: GPS

Definition at line 301 of file rtklib.h.

#### 9.20.4.13 EFACT\_IRN

```
const double EFACT_IRN = 1.5
```

error factor: IRNSS

Definition at line 306 of file rtklib.h.

#### 9.20.4.14 EFACT\_QZS

```
const double EFACT_QZS = 1.0
```

error factor: QZSS

Definition at line 304 of file rtklib.h.

#### 9.20.4.15 EFACT\_SBS

```
const double EFACT_SBS = 3.0
```

error factor: SBAS

Definition at line 307 of file rtklib.h.

**9.20.4.16 EPHOPT\_BRDC**

```
const int EPHOPT_BRDC = 0
```

ephemeris option: broadcast ephemeris

Definition at line 294 of file rtklib.h.

**9.20.4.17 EPHOPT\_LEX**

```
const int EPHOPT_LEX = 5
```

ephemeris option: QZSS LEX ephemeris

Definition at line 299 of file rtklib.h.

**9.20.4.18 EPHOPT\_PREC**

```
const int EPHOPT_PREC = 1
```

ephemeris option: precise ephemeris

Definition at line 295 of file rtklib.h.

**9.20.4.19 EPHOPT\_SBAS**

```
const int EPHOPT_SBAS = 2
```

ephemeris option: broadcast + SBAS

Definition at line 296 of file rtklib.h.

**9.20.4.20 EPHOPT\_SSRAPC**

```
const int EPHOPT_SSRAPC = 3
```

ephemeris option: broadcast + SSR\_APC

Definition at line 297 of file rtklib.h.

#### 9.20.4.21 EPHOPT\_SSRCOM

```
const int EPHOPT_SSRCOM = 4
```

ephemeris option: broadcast + SSR\_COM

Definition at line 298 of file rtklib.h.

#### 9.20.4.22 ERR\_BRDCI

```
const double ERR_BRDCI = 0.5
```

broadcast iono model error factor

Definition at line 138 of file rtklib.h.

#### 9.20.4.23 ERR\_CBIAS

```
const double ERR_CBIAS = 0.3
```

code bias error std (m)

Definition at line 139 of file rtklib.h.

#### 9.20.4.24 ERR\_SAAS

```
const double ERR_SAAS = 0.3
```

saastamoinen model error std (m)

Definition at line 137 of file rtklib.h.

#### 9.20.4.25 FE\_WGS84

```
const double FE_WGS84 = (1.0 / 298.257223563)
```

earth flattening (WGS84)

Definition at line 94 of file rtklib.h.

**9.20.4.26 FTP\_TIMEOUT**

```
const int FTP_TIMEOUT = 30
```

ftp/http timeout (s)

Definition at line 86 of file rtklib.h.

**9.20.4.27 GAP\_RESION**

```
const double GAP_RESION = 120
```

default gap to reset ionos parameters (ep)

Definition at line 141 of file rtklib.h.

**9.20.4.28 HION**

```
const double HION = 350000.0
```

ionosphere height (m)

Definition at line 96 of file rtklib.h.

**9.20.4.29 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.

**9.20.4.30 INT\_SWAP\_TRAC**

```
const double INT_SWAP_TRAC = 86400.0
```

swap interval of trace file (s)

Definition at line 100 of file rtklib.h.

#### 9.20.4.31 IONOOPT\_BRDC

```
const int IONOOPT_BRDC = 1
```

ionosphere option: broadcast model

Definition at line 276 of file rtklib.h.

#### 9.20.4.32 IONOOPT\_EST

```
const int IONOOPT_EST = 4
```

ionosphere option: estimation

Definition at line 279 of file rtklib.h.

#### 9.20.4.33 IONOOPT\_IFLC

```
const int IONOOPT_IFLC = 3
```

ionosphere option: L1/L2 or L1/L5 iono-free LC

Definition at line 278 of file rtklib.h.

#### 9.20.4.34 IONOOPT\_LEX

```
const int IONOOPT_LEX = 7
```

ionosphere option: QZSS LEX ionosphere

Definition at line 282 of file rtklib.h.

#### 9.20.4.35 IONOOPT\_OFF

```
const int IONOOPT_OFF = 0
```

ionosphere option: correction off

Definition at line 275 of file rtklib.h.

#### 9.20.4.36 IONOOPT\_QZS

```
const int IONOOPT_QZS = 6
```

ionosphere option: QZSS broadcast model

Definition at line 281 of file rtklib.h.

#### 9.20.4.37 IONOOPT\_SBAS

```
const int IONOOPT_SBAS = 2
```

ionosphere option: SBAS model

Definition at line 277 of file rtklib.h.

#### 9.20.4.38 IONOOPT\_STEC

```
const int IONOOPT_STEC = 8
```

ionosphere option: SLANT TEC model

Definition at line 283 of file rtklib.h.

#### 9.20.4.39 IONOOPT\_TEC

```
const int IONOOPT_TEC = 5
```

ionosphere option: IONEX TEC model

Definition at line 280 of file rtklib.h.

#### 9.20.4.40 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 1305 of file rtklib.h.

#### 9.20.4.41 MAXANT

```
const int MAXANT = 64
```

max length of station name/antenna type

Definition at line 151 of file rtklib.h.

#### 9.20.4.42 MAXBAND

```
const int MAXBAND = 10
```

max SBAS band of IGP

Definition at line 268 of file rtklib.h.

#### 9.20.4.43 MAXCLI

```
const int MAXCLI = 32
```

max client connection for tcp svr

Definition at line 83 of file rtklib.h.

#### 9.20.4.44 MAXDToe

```
const double MAXDToe = 7200.0
```

max time difference to GPS Toe (s)

Definition at line 258 of file rtklib.h.

#### 9.20.4.45 MAXDToe\_BDS

```
const double MAXDToe_BDS = 21600.0
```

max time difference to BeiDou Toe (s)

Definition at line 261 of file rtklib.h.

**9.20.4.46 MAXDToe\_GAL**

```
const double MAXDToe_GAL = 10800.0
```

max time difference to Galileo Toe (s)

Definition at line 260 of file rtklib.h.

**9.20.4.47 MAXDToe\_GLO**

```
const double MAXDToe_GLO = 1800.0
```

max time difference to GLONASS Toe (s)

Definition at line 262 of file rtklib.h.

**9.20.4.48 MAXDToe\_QZS**

```
const double MAXDToe_QZS = 7200.0
```

max time difference to QZSS Toe (s)

Definition at line 259 of file rtklib.h.

**9.20.4.49 MAXDToe\_S**

```
const double MAXDToe_S = 86400.0
```

max time difference to ephemeris toe (s) for other

Definition at line 264 of file rtklib.h.

**9.20.4.50 MAXDToe\_SBS**

```
const double MAXDToe_SBS = 360.0
```

max time difference to SBAS Toe (s)

Definition at line 263 of file rtklib.h.

#### 9.20.4.51 MAXERRMSG

```
const int MAXERRMSG = 4096
```

max length of error/warning message

Definition at line 273 of file rtklib.h.

#### 9.20.4.52 MAXEXFILE

```
const int MAXEXFILE = 1024
```

max number of expanded files

Definition at line 309 of file rtklib.h.

#### 9.20.4.53 MAXFREQ

```
const int MAXFREQ = 7
```

max NFREQ

Definition at line 143 of file rtklib.h.

#### 9.20.4.54 MAXGDOP

```
const double MAXGDOP = 300.0
```

max GDOP

Definition at line 265 of file rtklib.h.

#### 9.20.4.55 MAXLEAPS

```
const int MAXLEAPS = 64
```

max number of leap seconds table

Definition at line 145 of file rtklib.h.

**9.20.4.56 MAXNGEO**

```
const int MAXNGEO = 4
```

max number of GEO satellites

Definition at line 270 of file rtklib.h.

**9.20.4.57 MAXNIGP**

```
const int MAXNIGP = 201
```

max number of IGP in SBAS band

Definition at line 269 of file rtklib.h.

**9.20.4.58 MAXOBS**

```
const int MAXOBS = 64
```

max number of obs in an epoch

Definition at line 253 of file rtklib.h.

**9.20.4.59 MAXOBSBUF**

```
const int MAXOBSBUF = 128
```

max number of observation data buffer

Definition at line 90 of file rtklib.h.

**9.20.4.60 MAXOBSTYPE**

```
const int MAXOBSTYPE = 64
```

max number of obs type in RINEX

Definition at line 257 of file rtklib.h.

#### 9.20.4.61 MAXPRNBDS

```
const int MAXPRNBDS = 37
```

max satellite sat number of BeiDou

Definition at line 208 of file rtklib.h.

#### 9.20.4.62 MAXPRNGAL

```
const int MAXPRNGAL = 36
```

max satellite PRN number of Galileo

Definition at line 185 of file rtklib.h.

#### 9.20.4.63 MAXPRNGLO

```
const int MAXPRNGLO = 27
```

max satellite slot number of GLONASS

Definition at line 173 of file rtklib.h.

#### 9.20.4.64 MAXPRNGPS

```
const int MAXPRNGPS = 32
```

max satellite PRN number of GPS

Definition at line 154 of file rtklib.h.

#### 9.20.4.65 MAXPRNSBS

```
const int MAXPRNSBS = 142
```

max satellite PRN number of SBAS

Definition at line 245 of file rtklib.h.

**9.20.4.66 MAXRAWLEN**

```
const int MAXRAWLEN = 4096
```

max length of receiver raw message

Definition at line 87 of file rtklib.h.

**9.20.4.67 MAXRCV**

```
const int MAXRCV = 64
```

max receiver number (1 to MAXRCV)

Definition at line 256 of file rtklib.h.

**9.20.4.68 MAXSBSAGEF**

```
const double MAXSBSAGEF = 30.0
```

max age of SBAS fast correction (s)

Definition at line 310 of file rtklib.h.

**9.20.4.69 MAXSBSAGEL**

```
const double MAXSBSAGEL = 1800.0
```

max age of SBAS long term corr (s)

Definition at line 311 of file rtklib.h.

**9.20.4.70 MAXSBSMSG**

```
const int MAXSBSMSG = 32
```

max number of SBAS msg in RTK server

Definition at line 89 of file rtklib.h.

#### 9.20.4.71 MAXSBSURA

```
const int MAXSBSURA = 8
```

max URA of SBAS satellite

Definition at line 267 of file rtklib.h.

#### 9.20.4.72 MAXSOLBUF

```
const int MAXSOLBUF = 256
```

max number of solution buffer

Definition at line 88 of file rtklib.h.

#### 9.20.4.73 MAXSOLMSG

```
const int MAXSOLMSG = 8191
```

max length of solution message

Definition at line 272 of file rtklib.h.

#### 9.20.4.74 MAXSOLQ

```
const int MAXSOLQ = 7
```

max number of solution status

Definition at line 130 of file rtklib.h.

#### 9.20.4.75 MAXSTATMSG

```
const int MAXSTATMSG = 32
```

max length of status message

Definition at line 84 of file rtklib.h.

**9.20.4.76 MAXSTRMSG**

```
const int MAXSTRMSG = 1024
```

max length of stream message

Definition at line 325 of file rtklib.h.

**9.20.4.77 MAXSTRPATH**

```
const int MAXSTRPATH = 1024
```

max length of stream path

Definition at line 324 of file rtklib.h.

**9.20.4.78 MINPRNBDS**

```
const int MINPRNBDS = 1
```

min satellite sat number of BeiDou

Definition at line 207 of file rtklib.h.

**9.20.4.79 MINPRNGAL**

```
const int MINPRNGAL = 1
```

min satellite PRN number of Galileo

Definition at line 184 of file rtklib.h.

**9.20.4.80 MINPRNGLO**

```
const int MINPRNGLO = 1
```

min satellite slot number of GLONASS

Definition at line 172 of file rtklib.h.

#### 9.20.4.81 MINPRNGPS

```
const int MINPRNGPS = 1
```

min satellite PRN number of GPS

Definition at line 153 of file rtklib.h.

#### 9.20.4.82 MINPRNSBS

```
const int MINPRNSBS = 120
```

min satellite PRN number of SBAS

Definition at line 244 of file rtklib.h.

#### 9.20.4.83 NEXOBS

```
const int NEXOBS = 0
```

number of extended obs codes

Definition at line 150 of file rtklib.h.

#### 9.20.4.84 NFREQ

```
const int NFREQ = 3
```

number of carrier frequencies

Definition at line 148 of file rtklib.h.

#### 9.20.4.85 NFREQGLO

```
const int NFREQGLO = 2
```

number of carrier frequencies of GLONASS

Definition at line 149 of file rtklib.h.

**9.20.4.86 NSATBDS**

```
const int NSATBDS = (MAXPRNBDS - MINPRNBDS + 1)
```

number of BeiDou satellites

Definition at line 209 of file rtklib.h.

**9.20.4.87 NSATGAL**

```
const int NSATGAL = (MAXPRNGAL - MINPRNGAL + 1)
```

number of Galileo satellites

Definition at line 186 of file rtklib.h.

**9.20.4.88 NSATGLO**

```
const int NSATGLO = (MAXPRNGLO - MINPRNGLO + 1)
```

number of GLONASS satellites

Definition at line 174 of file rtklib.h.

**9.20.4.89 NSATGPS**

```
const int NSATGPS = (MAXPRNGPS - MINPRNGPS + 1)
```

number of GPS satellites

Definition at line 155 of file rtklib.h.

**9.20.4.90 NSATSBS**

```
const int NSATSBS = (MAXPRNSBS - MINPRNSBS + 1)
```

number of SBAS satellites

Definition at line 246 of file rtklib.h.

#### 9.20.4.91 NSYS

```
const int NSYS = (NSYSGPS + NSYSGLO + NSYSGAL + NSYSQZS + NSYSBDS + NSYSIRN + NSYSLEO)
```

number of systems

Definition at line 242 of file rtklib.h.

#### 9.20.4.92 PMODE\_DGPS

```
const int PMODE_DGPS = 1
```

positioning mode: DGPS/DGNSS

Definition at line 106 of file rtklib.h.

#### 9.20.4.93 PMODE\_FIXED

```
const int PMODE_FIXED = 5
```

positioning mode: fixed

Definition at line 110 of file rtklib.h.

#### 9.20.4.94 PMODE\_KINEMA

```
const int PMODE_KINEMA = 2
```

positioning mode: kinematic

Definition at line 107 of file rtklib.h.

#### 9.20.4.95 PMODE\_MOVEB

```
const int PMODE_MOVEB = 4
```

positioning mode: moving-base

Definition at line 109 of file rtklib.h.

**9.20.4.96 PMODE\_PPP\_FIXED**

```
const int PMODE_PPP_FIXED = 8
```

positioning mode: PPP-fixed

Definition at line 113 of file rtklib.h.

**9.20.4.97 PMODE\_PPP\_KINEMA**

```
const int PMODE_PPP_KINEMA = 6
```

positioning mode: PPP-kinematic

Definition at line 111 of file rtklib.h.

**9.20.4.98 PMODE\_PPP\_STATIC**

```
const int PMODE_PPP_STATIC = 7
```

positioning mode: PPP-static

Definition at line 112 of file rtklib.h.

**9.20.4.99 PMODE\_SINGLE**

```
const int PMODE_SINGLE = 0
```

positioning mode: single

Definition at line 105 of file rtklib.h.

**9.20.4.100 PMODE\_STATIC**

```
const int PMODE_STATIC = 3
```

positioning mode: static

Definition at line 108 of file rtklib.h.

#### 9.20.4.101 POLYCRC24Q

```
const unsigned int POLYCRC24Q = 0x1864CFBu
```

CRC24Q polynomial.

Definition at line 103 of file rtklib.h.

#### 9.20.4.102 POLYCRC32

```
const unsigned int POLYCRC32 = 0xEDB88320u
```

CRC32 polynomial.

Definition at line 102 of file rtklib.h.

#### 9.20.4.103 POSOPT\_RINEX

```
const int POSOPT_RINEX = 3
```

pos option: rinex header pos

Definition at line 323 of file rtklib.h.

#### 9.20.4.104 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.

#### 9.20.4.105 RE\_WGS84

```
const double RE_WGS84 = 6378137.0
```

earth semimajor axis (WGS84) (m)

Definition at line 93 of file rtklib.h.

**9.20.4.106 REL\_HUMI**

```
const double REL_HUMI = 0.7
```

relative humidity for saastamoinen model

Definition at line 140 of file rtklib.h.

**9.20.4.107 SERIBUFFSIZE**

```
const int SERIBUFFSIZE = 4096
```

serial buffer size (bytes)

Definition at line 81 of file rtklib.h.

**9.20.4.108 SOLF\_ENU**

```
const int SOLF_ENU = 2
```

solution format: e/n/u-baseline

Definition at line 117 of file rtklib.h.

**9.20.4.109 SOLF\_GSIF**

```
const int SOLF_GSIF = 5
```

solution format: GSI F1/F2

Definition at line 120 of file rtklib.h.

**9.20.4.110 SOLF\_LLH**

```
const int SOLF_LLH = 0
```

solution format: lat/lon/height

Definition at line 115 of file rtklib.h.

#### 9.20.4.111 SOLF\_NMEA

```
const int SOLF_NMEA = 3
```

solution format: NMEA-183

Definition at line 118 of file rtklib.h.

#### 9.20.4.112 SOLF\_STAT

```
const int SOLF_STAT = 4
```

solution format: solution status

Definition at line 119 of file rtklib.h.

#### 9.20.4.113 SOLF\_XYZ

```
const int SOLF_XYZ = 1
```

solution format: x/y/z-ecef

Definition at line 116 of file rtklib.h.

#### 9.20.4.114 SOLQ\_DGPS

```
const int SOLQ_DGPS = 4
```

solution status: DGPS/DGNSS

Definition at line 126 of file rtklib.h.

#### 9.20.4.115 SOLQ\_DR

```
const int SOLQ_DR = 7
```

solution status: dead reckoning

Definition at line 129 of file rtklib.h.

**9.20.4.116 SOLQ\_FIX**

```
const int SOLQ_FIX = 1
```

solution status: fix

Definition at line 123 of file rtklib.h.

**9.20.4.117 SOLQ\_FLOAT**

```
const int SOLQ_FLOAT = 2
```

solution status: float

Definition at line 124 of file rtklib.h.

**9.20.4.118 SOLQ\_NONE**

```
const int SOLQ_NONE = 0
```

solution status: no solution

Definition at line 122 of file rtklib.h.

**9.20.4.119 SOLQ\_PPP**

```
const int SOLQ_PPP = 6
```

solution status: PPP

Definition at line 128 of file rtklib.h.

**9.20.4.120 SOLQ\_SBAS**

```
const int SOLQ_SBAS = 3
```

solution status: SBAS

Definition at line 125 of file rtklib.h.

**9.20.4.121 SOLQ\_SINGLE**

```
const int SOLQ_SINGLE = 5
```

solution status: single

Definition at line 127 of file rtklib.h.

**9.20.4.122 SYS\_ALL**

```
const int SYS_ALL = 0xFF
```

navigation system: all

Definition at line 167 of file rtklib.h.

**9.20.4.123 SYS\_BDS**

```
const int SYS_BDS = 0x20
```

navigation system: BeiDou

Definition at line 164 of file rtklib.h.

**9.20.4.124 SYS\_GAL**

```
const int SYS_GAL = 0x08
```

navigation system: Galileo

Definition at line 162 of file rtklib.h.

**9.20.4.125 SYS\_GLO**

```
const int SYS_GLO = 0x04
```

navigation system: GLONASS

Definition at line 161 of file rtklib.h.

**9.20.4.126 SYS\_GPS**

```
const int SYS_GPS = 0x01
```

navigation system: GPS

Definition at line 159 of file rtklib.h.

**9.20.4.127 SYS\_IRN**

```
const int SYS_IRN = 0x40
```

navigation system: IRNS

Definition at line 165 of file rtklib.h.

**9.20.4.128 SYS\_LEO**

```
const int SYS_LEO = 0x80
```

navigation system: LEO

Definition at line 166 of file rtklib.h.

**9.20.4.129 SYS\_NONE**

```
const int SYS_NONE = 0x00
```

navigation system: none

Definition at line 158 of file rtklib.h.

**9.20.4.130 SYS\_QZS**

```
const int SYS_QZS = 0x10
```

navigation system: QZSS

Definition at line 163 of file rtklib.h.

#### 9.20.4.131 SYS\_SBS

```
const int SYS_SBS = 0x02
```

navigation system: SBAS

Definition at line 160 of file rtklib.h.

#### 9.20.4.132 TIMES\_GPST

```
const int TIMES_GPST = 0
```

time system: gps time

Definition at line 132 of file rtklib.h.

#### 9.20.4.133 TIMES\_JST

```
const int TIMES_JST = 2
```

time system: jst

Definition at line 134 of file rtklib.h.

#### 9.20.4.134 TIMES\_UTC

```
const int TIMES_UTC = 1
```

time system: utc

Definition at line 133 of file rtklib.h.

#### 9.20.4.135 TIMETAGH\_LEN

```
const int TIMETAGH_LEN = 64
```

time tag file header length

Definition at line 82 of file rtklib.h.

**9.20.4.136 TINTACT**

```
const int TINTACT = 200
```

period for stream active (ms)

Definition at line 80 of file rtklib.h.

**9.20.4.137 TROPOPT\_COR**

```
const int TROPOPT_COR = 5
```

troposphere option: ZTD correction

Definition at line 290 of file rtklib.h.

**9.20.4.138 TROPOPT\_CORG**

```
const int TROPOPT_CORG = 6
```

troposphere option: ZTD+grad correction

Definition at line 291 of file rtklib.h.

**9.20.4.139 TROPOPT\_EST**

```
const int TROPOPT_EST = 3
```

troposphere option: ZTD estimation

Definition at line 288 of file rtklib.h.

**9.20.4.140 TROPOPT\_ESTG**

```
const int TROPOPT_ESTG = 4
```

troposphere option: ZTD+grad estimation

Definition at line 289 of file rtklib.h.

**9.20.4.141 TROPOPT\_OFF**

```
const int TROPOPT_OFF = 0
```

troposphere option: correction off

Definition at line 285 of file rtklib.h.

**9.20.4.142 TROPOPT\_SAAS**

```
const int TROPOPT_SAAS = 1
```

troposphere option: Saastamoinen model

Definition at line 286 of file rtklib.h.

**9.20.4.143 TROPOPT\_SBAS**

```
const int TROPOPT_SBAS = 2
```

troposphere option: SBAS model

Definition at line 287 of file rtklib.h.

## 9.21 Observables

### Modules

- [obs\\_adapters](#)
- [obs\\_gr\\_blocks](#)
- [observables\\_libs](#)

### 9.21.1 Detailed Description

Classes for the computation of GNSS observables

## 9.22 obs\_adapters

### Classes

- class [HybridObservables](#)

*This class implements an [ObservablesInterface](#) for observables of all kind of GNSS signals.*

### 9.22.1 Detailed Description

Wrap GNU Radio observables blocks with an [ObservablesInterface](#)

## 9.23 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\_)

#### 9.23.1 Detailed Description

GNU Radio blocks for the computation of GNSS observables

## 9.24 observables\_libs

### Classes

- class [Obs\\_Conf](#)

### 9.24.1 Detailed Description

Utilities for GNSS observables configuration.

## 9.25 pvt\_adapters

### Classes

- class [Rtklib\\_Pvt](#)

*This class implements a [PvtInterface](#) for the RTKLIB PVT block.*

### 9.25.1 Detailed Description

Wrap GNU Radio PVT solvers with a [PvtInterface](#)

## 9.26 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)

#### 9.26.1 Detailed Description

GNU Radio blocks for the computation of PVT solutions.

## 9.27 pvt\_libs

### Classes

- class [GeoJSON\\_Printer](#)  
*Prints PVT solutions in GeoJSON format file.*
- class [Gpx\\_Printer](#)  
*Prints PVT information to GPX format file.*
- class [Kml\\_Printer](#)  
*Prints PVT information to OGC KML format file (can be viewed with Google Earth)*
- class [Monitor\\_Pvt](#)  
*This class contains parameters and outputs of the PVT block.*
- class [Monitor\\_Pvt\\_Udp\\_Sink](#)
- 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 [Pvt\\_Conf](#)
- class [Pvt\\_Solution](#)  
*Base class for a PVT solution.*
- class [Rinex\\_Printer](#)  
*Class that handles the generation of Receiver INdependent EXchange format (RINEX) files.*
- 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 [Rtcm\\_Printer](#)  
*This class provides a implementation of a subset of the RTCM Standard 10403.2 messages.*
- class [Rtklib\\_Solver](#)  
*This class implements a PVT solution based on RTKLIB.*
- class [Serdes\\_Monitor\\_Pvt](#)  
*This class implements serialization and deserialization of [Monitor\\_Pvt](#) objects using Protocol Buffers.*

### Typedefs

- using **b\_io\_context** = boost::asio::io\_service
- using **b\_io\_context** = boost::asio::io\_service

### Functions

- std::string **asString** (long double x, std::string::size\_type precision)
- int64\_t **asInt** (const std::string &s)

#### 9.27.1 Detailed Description

Library for the computation of PVT solutions.

## 9.28 Resampler

### Modules

- [resampler\\_adapters](#)
- [resampler\\_gr\\_blocks](#)

### 9.28.1 Detailed Description

Classes for input signal resampling

## 9.29 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`.*

### 9.29.1 Detailed Description

Classes that wrap GNU Radio resampler blocks with a [GNSSBlockInterface](#)

## 9.30 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)

#### 9.30.1 Detailed Description

GNU Radio blocks for input signal resampling

## 9.31 Signal Source

### Modules

- [signal\\_source\\_adapters](#)
- [signal\\_source\\_gr\\_blocks](#)
- [signal\\_source\\_libs](#)

### 9.31.1 Detailed Description

Classes for Signal Source management.

## 9.32 signal\_source\_adapters

### Classes

- class [Ad9361FpgaSignalSource](#)
- class [CustomUDPSignalSource](#)

*This class reads from UDP packets, which streams interleaved I/Q samples over a network.*
- class [FileSignalSource](#)

*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 [GenSignalSource](#)

*This class wraps blocks that generates synthesized GNSS signal and filters the signal.*
- class [Gn3sSignalSource](#)

*This class reads samples from a GN3S USB dongle, a RF front-end signal sampler.*
- class [LabsatSignalSource](#)

*This class reads samples stored by a LabSat 2 or LabSat 3 device.*
- 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 [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 [RawArraySignalSource](#)

*This class reads samples from a GN3S USB dongle, a 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 [SpirGSS6450FileSignalSource](#)

*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>)*

### 9.32.1 Detailed Description

Classes that wrap GNU Radio signal sources with a [GNSSBlockInterface](#)

## 9.33 signal\_source\_gr\_blocks

### Classes

- class [Gr\\_Complex\\_Ip\\_Packet\\_Source](#)
- class [labsat23\\_source](#)  
*This class implements conversion between Labsat2 and 3 format byte packet samples 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\\_intspir\\_1bit\\_samples](#)  
*This class implements conversion between byte packet samples to 2bit samples 1 byte = 4 2bit samples.*
- class [unpack\\_spir\\_gss6450\\_samples](#)

### Typedefs

- 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](#) = std::shared\_ptr< [unpack\\_byte\\_4bit\\_samples](#) >
- using [unpack\\_intspir\\_1bit\\_samples\\_sptr](#) = gnss\_shared\_ptr< [unpack\\_intspir\\_1bit\\_samples](#) >
- using [unpack\\_spir\\_gss6450\\_samples\\_sptr](#) = gnss\_shared\_ptr< [unpack\\_spir\\_gss6450\\_samples](#) >

### Functions

- [labsat23\\_source\\_sptr](#) **labsat23\_make\_source\_sptr** (const char \*signal\_file\_basename, int channel\_↵ selector, [Concurrent\\_Queue](#)< pmt::pmt\_t > \*queue)
- [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\\_intspir\\_1bit\\_samples\\_sptr](#) **make\_unpack\_intspir\_1bit\_samples** ()
- [unpack\\_spir\\_gss6450\\_samples\\_sptr](#) **make\_unpack\_spir\_gss6450\_samples** (int adc\_nbit\_)

#### 9.33.1 Detailed Description

GNU Radio blocks for signal sources.

## 9.34 signal\_source\_libs

### Classes

- struct [stream\\_cfg](#)
- class [Fpga\\_dynamic\\_bit\\_selection](#)  
*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 [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\\_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.*
- 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_lll` (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 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_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\\_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)
- 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.*

### 9.34.1 Detailed Description

Library with utilities for signal sources.

### 9.34.2 Enumeration Type Documentation

#### 9.34.2.1 RTL\_TCP\_COMMAND

enum [RTL\\_TCP\\_COMMAND](#)

Command IDs for configuration rtl\_tcp.

Definition at line 32 of file rtl\_tcp\_commands.h.

### 9.34.3 Function Documentation

#### 9.34.3.1 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.

## 9.35 Telemetry Decoder

### Modules

- [telemetry\\_decoder\\_adapters](#)
- [telemetry\\_decoder\\_gr\\_blocks](#)
- [telemetry\\_decoder\\_libs](#)
- [telemetry\\_decoder\\_libswiftcnv](#)

### 9.35.1 Detailed Description

Classes for the decoding of GNSS Navigation messages.

## 9.36 telemetry\_decoder\_adapters

### Classes

- class [BeidouB1iTelemetryDecoder](#)  
*This class implements a NAV data decoder for BEIDOU B1I.*
- class [BeidouB3iTelemetryDecoder](#)  
*This class implements a NAV data decoder for BEIDOU B1I.*
- class [GalileoE1BTelemetryDecoder](#)  
*This class implements a NAV data decoder for Galileo INAV frames in E1B radio link.*
- class [GalileoE5aTelemetryDecoder](#)  
*This class implements a NAV data decoder for Galileo INAV frames in E1B radio link.*
- class [GalileoE5bTelemetryDecoder](#)  
*This class implements a NAV data decoder for Galileo INAV frames in E5b radio link.*
- class [GalileoE6TelemetryDecoder](#)  
*This class implements a NAV data decoder for Galileo CNAV frames in E6 radio link.*
- class [GlonassL1CaTelemetryDecoder](#)  
*This class implements a NAV data decoder for GLONASS L1 C/A.*
- class [GlonassL2CaTelemetryDecoder](#)  
*This class implements a NAV data decoder for GLONASS L2 C/A.*
- class [GpsL1CaTelemetryDecoder](#)  
*This class implements a NAV data decoder for GPS L1 C/A.*
- class [GpsL2CTelemetryDecoder](#)  
*This class implements a NAV data decoder for GPS L2 M.*
- class [GpsL5TelemetryDecoder](#)  
*This class implements a NAV data decoder for GPS L5.*
- class [SbasL1TelemetryDecoder](#)  
*This class implements a NAV data decoder for SBAS frames in L1 radio link.*

### 9.36.1 Detailed Description

Wrap GNU Radio blocs for the decoding of GNSS Navigation messages with a [TelemetryDecoderInterface](#)

## 9.37 telemetry\_decoder\_gr\_blocks

### Classes

- class [beidou\\_b1i\\_telemetry\\_decoder\\_gs](#)  
*This class implements a block that decodes the BeiDou DNAV data.*
- class [beidou\\_b3i\\_telemetry\\_decoder\\_gs](#)  
*This class implements a block that decodes the BeiDou DNAV data.*
- class [galileo\\_telemetry\\_decoder\\_gs](#)  
*This class implements a block that decodes the INAV and FNAV data defined in Galileo ICD.*
- class [glonass\\_l1\\_ca\\_telemetry\\_decoder\\_gs](#)  
*This class implements a block that decodes the GNAV data defined in GLONASS ICD v5.1.*
- class [glonass\\_l2\\_ca\\_telemetry\\_decoder\\_gs](#)  
*This class implements a block that decodes the GNAV data defined in GLONASS ICD v5.1.*
- class [gps\\_l1\\_ca\\_telemetry\\_decoder\\_gs](#)  
*This class implements a block that decodes the NAV data defined in IS-GPS-200K.*
- class [gps\\_l2c\\_telemetry\\_decoder\\_gs](#)  
*This class implements a block that decodes CNAV data defined in IS-GPS-200K.*
- class [gps\\_l5\\_telemetry\\_decoder\\_gs](#)  
*This class implements a GPS L5 Telemetry decoder.*
- 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 [beidou\\_b1i\\_telemetry\\_decoder\\_gs\\_sptr](#) = gnss\_shared\_ptr< [beidou\\_b1i\\_telemetry\\_decoder\\_gs](#) >
- using [beidou\\_b3i\\_telemetry\\_decoder\\_gs\\_sptr](#) = gnss\_shared\_ptr< [beidou\\_b3i\\_telemetry\\_decoder\\_gs](#) >
- using [galileo\\_telemetry\\_decoder\\_gs\\_sptr](#) = gnss\_shared\_ptr< [galileo\\_telemetry\\_decoder\\_gs](#) >
- using [glonass\\_l1\\_ca\\_telemetry\\_decoder\\_gs\\_sptr](#) = gnss\_shared\_ptr< [glonass\\_l1\\_ca\\_telemetry\\_decoder\\_gs](#) >
- using [glonass\\_l2\\_ca\\_telemetry\\_decoder\\_gs\\_sptr](#) = gnss\_shared\_ptr< [glonass\\_l2\\_ca\\_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](#) >

### Functions

- [beidou\\_b1i\\_telemetry\\_decoder\\_gs\\_sptr](#) [beidou\\_b1i\\_make\\_telemetry\\_decoder\\_gs](#) (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)
- [beidou\\_b3i\\_telemetry\\_decoder\\_gs\\_sptr](#) [beidou\\_b3i\\_make\\_telemetry\\_decoder\\_gs](#) (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)
- [galileo\\_telemetry\\_decoder\\_gs\\_sptr](#) [galileo\\_make\\_telemetry\\_decoder\\_gs](#) (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf, int frame\_type)
- [glonass\\_l1\\_ca\\_telemetry\\_decoder\\_gs\\_sptr](#) [glonass\\_l1\\_ca\\_make\\_telemetry\\_decoder\\_gs](#) (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)
- [glonass\\_l2\\_ca\\_telemetry\\_decoder\\_gs\\_sptr](#) [glonass\\_l2\\_ca\\_make\\_telemetry\\_decoder\\_gs](#) (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)
- [gps\\_l1\\_ca\\_telemetry\\_decoder\\_gs\\_sptr](#) [gps\\_l1\\_ca\\_make\\_telemetry\\_decoder\\_gs](#) (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)
- [gps\\_l2c\\_telemetry\\_decoder\\_gs\\_sptr](#) [gps\\_l2c\\_make\\_telemetry\\_decoder\\_gs](#) (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)
- [gps\\_l5\\_telemetry\\_decoder\\_gs\\_sptr](#) [gps\\_l5\\_make\\_telemetry\\_decoder\\_gs](#) (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)
- [sbas\\_l1\\_telemetry\\_decoder\\_gs\\_sptr](#) [sbas\\_l1\\_make\\_telemetry\\_decoder\\_gs](#) (const [Gnss\\_Satellite](#) &satellite, bool dump)

### 9.37.1 Detailed Description

GNU Radio blocks for the demodulation of GNSS navigation messages.

## 9.38 telemetry\_decoder\_libs

### Classes

- class [Tlm\\_Conf](#)
- class [Viterbi\\_Decoder](#)

*Class that implements a Viterbi decoder.*

### Functions

- int [parity\\_counter](#) (int symbol, int length)  
*Determines if a symbol has odd (1) or even (0) parity Output parameters:*
- int [nsc\\_enc\\_bit](#) (int state\_out\_p[], int input, int state\_in, const int g[], int KK, int nn)  
*Convolutionally encodes a single bit using a rate 1/n encoder. Takes in one input bit at a time, and produces a n-bit output.*
- void [nsc\\_transit](#) (int output\_p[], int trans\_p[], int input, int g[], int KK, int nn)  
*Function that creates the transit and output vectors.*
- float [Gamma](#) (const float rec\_array[], int symbol, int nn)  
*Computes the branch metric used for decoding.*
- void [Viterbi](#) (int output\_u\_int[], const int out0[], const int state0[], const int out1[], const int state1[], const float input\_c[], int KK, int nn, int LL)  
*Uses the Viterbi algorithm to perform hard-decision decoding of a convolutional code.*
- int [save\\_tlm\\_matfile](#) (const std::string &dumpfile)
- bool [tlm\\_remove\\_file](#) (const std::string &file\_to\_remove)

### Variables

- const float **MAXLOG** = 1e7

### 9.38.1 Detailed Description

Utilities for the decoding of GNSS navigation messages.

### 9.38.2 Function Documentation

#### 9.38.2.1 Gamma()

```
float Gamma (
    const float rec_array[],
    int symbol,
    int nn ) [inline]
```

Computes the branch metric used for decoding.

#### Returns

(returned float) The metric between the hypothetical symbol and the received vector

## Parameters

in	<i>rec_array</i>	The received vector, of length nn
in	<i>symbol</i>	The hypothetical symbol
in	<i>nn</i>	The length of the received vector

Definition at line 149 of file convolutional.h.

## 9.38.2.2 nsc\_enc\_bit()

```
int nsc_enc_bit (
    int state_out_p[],
    int input,
    int state_in,
    const int g[],
    int KK,
    int nn ) [inline]
```

Convolutionally encodes a single bit using a rate 1/n encoder. Takes in one input bit at a time, and produces a n-bit output.

## Parameters

in	<i>input</i>	The input data bit (i.e. a 0 or 1).
in	<i>state_in</i>	The starting state of the encoder (an int from 0 to $2^m-1$ ).
in	<i>g[]</i>	An n-element vector containing the code generators in binary form.
in	<i>KK</i>	The constraint length of the convolutional code.
out	<i>output_p[]</i>	An n-element vector containing the encoded bits.
out	<i>state_out_p[]</i>	An integer containing the final state of the encoder (i.e. the state after encoding this bit)

This function is used by [nsc\\_transit\(\)](#)

Definition at line 90 of file convolutional.h.

References [parity\\_counter\(\)](#).

Referenced by [nsc\\_transit\(\)](#).

## 9.38.2.3 nsc\_transit()

```
void nsc_transit (
    int output_p[],
    int trans_p[],
    int input,
    int g[],
```

```
int KK,
int nn ) [inline]
```

Function that creates the transit and output vectors.

Definition at line 120 of file convolutional.h.

References [nsc\\_enc\\_bit\(\)](#).

#### 9.38.2.4 parity\_counter()

```
int parity_counter (
    int symbol,
    int length ) [inline]
```

Determines if a symbol has odd (1) or even (0) parity Output parameters:

##### Returns

(returned int): The symbol's parity = 1 for odd and 0 for even

##### Parameters

in	<i>symbol</i>	The integer-valued symbol
in	<i>length</i>	The highest bit position in the symbol

This function is used by [nsc\\_enc\\_bit\(\)](#), [rsc\\_enc\\_bit\(\)](#), and [rsc\\_tail\(\)](#)

Definition at line 62 of file convolutional.h.

Referenced by [nsc\\_enc\\_bit\(\)](#).

#### 9.38.2.5 Viterbi()

```
void Viterbi (
    int output_u_int[],
    const int out0[],
    const int state0[],
    const int out1[],
    const int state1[],
    const float input_c[],
    int KK,
    int nn,
    int LL ) [inline]
```

Uses the Viterbi algorithm to perform hard-decision decoding of a convolutional code.

## Parameters

in	<i>out0[]</i>	The output bits for each state if input is a 0.
in	<i>state0[]</i>	The next state if input is a 0.
in	<i>out1[]</i>	The output bits for each state if input is a 1.
in	<i>state1[]</i>	The next state if input is a 1.
in	<i>r[]</i>	The received signal in LLR-form. For BPSK, must be in form $r = 2*a*y/(\sigma^2)$ .
in	<i>KK</i>	The constraint length of the convolutional code.
in	<i>LL</i>	The number of data bits.
out	<i>output_u_int[]</i>	Hard decisions on the data bits

Definition at line 181 of file convolutional.h.

## 9.39 telemetry\_decoder\_libswiftcnv

### Classes

- struct [cnv\\_msg\\_t](#)
- struct [cnv\\_v27\\_part\\_t](#)
- struct [cnv\\_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) ((x) < 0 ? -(x) : (x))
- #define [MIN](#)(x, y) (((x) < (y)) ? (x) : (y))
- #define [MAX](#)(x, y) (((x) > (y)) ? (x) : (y))
- #define [CLAMP\\_DIFF](#)(a, b) (MAX((a), (b)) - (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](#) \* [cnv\\_msg\\_decoder\\_get\\_poly](#) (void)
- void [cnv\\_msg\\_decoder\\_init](#) ([cnv\\_msg\\_decoder\\_t](#) \*dec)
- bool [cnv\\_msg\\_decoder\\_add\\_symbol](#) ([cnv\\_msg\\_decoder\\_t](#) \*dec, unsigned char symbol, [cnv\\_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)

#### 9.39.1 Detailed Description

Utilities for CNAV message decoding by Swift Navigation Inc.

#### 9.39.2 Macro Definition Documentation

#### 9.39.2.1 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.

#### 9.39.2.2 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.

#### 9.39.2.3 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.

#### 9.39.2.4 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.

## 9.40 Tracking

### Modules

- [tracking\\_adapters](#)
- [tracking\\_gr\\_blocks](#)
- [tracking\\_libs](#)

### 9.40.1 Detailed Description

Classes for GNSS signal tracking.

## 9.41 tracking\_adapters

### Classes

- class [BeidouB1iDIIPITracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [BeidouB3iDIIPITracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GalileoE1DIPIIVemITracking](#)  
*This class Adapts a DLL+PLL VEML (Very Early Minus Late) tracking loop block to a [TrackingInterface](#) for Galileo E1 signals.*
- class [GalileoE1DIPIIVemITrackingFpga](#)  
*This class Adapts a DLL+PLL VEML (Very Early Minus Late) tracking loop block to a [TrackingInterface](#) for Galileo E1 signals.*
- class [GalileoE1TcpConnectorTracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GalileoE5aDIIPITracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GalileoE5aDIIPITrackingFpga](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GalileoE5bDIIPITracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GalileoE6DIIPITracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GlonassL1CaDIPIICAidTracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GlonassL1CaDIPIITracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GlonassL2CaDIPIICAidTracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GlonassL2CaDIPIITracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GpsL1CaDIPIITracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GpsL1CaDIPIITrackingFpga](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GpsL1CaDIPIITrackingGPU](#)  
*This class implements a code DLL + carrier PLL tracking loop using GPU accelerated functions.*
- class [GpsL1CaKfTracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GpsL1CaTcpConnectorTracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GpsL2MDIPIITracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GpsL2MDIPIITrackingFpga](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GpsL5DIPIITracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*
- class [GpsL5DIPIITrackingFpga](#)  
*This class implements a code DLL + carrier PLL tracking loop.*

### 9.41.1 Detailed Description

Wrap GNU Radio blocks for GNSS signal tracking with a [TrackingInterface](#)

## 9.42 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 [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_cc](#)  
*This class implements a DLL + PLL tracking loop block.*
- class [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_sc](#)  
*This class implements a DLL + PLL tracking loop block.*
- class [Glonass\\_L1\\_Ca\\_Dll\\_Pll\\_Tracking\\_cc](#)  
*This class implements a DLL + PLL tracking loop block.*
- class [glonass\\_l2\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_cc](#)  
*This class implements a DLL + PLL tracking loop block.*
- class [glonass\\_l2\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_sc](#)  
*This class implements a DLL + PLL tracking loop block.*
- class [Glonass\\_L2\\_Ca\\_Dll\\_Pll\\_Tracking\\_cc](#)  
*This class implements a DLL + PLL tracking loop block.*
- class [Gps\\_L1\\_Ca\\_Dll\\_Pll\\_Tracking\\_GPU\\_cc](#)  
*This class implements a DLL + PLL tracking loop block.*
- class [Gps\\_L1\\_Ca\\_Kf\\_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 [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_cc\\_sptr](#) = gnss\_shared\_ptr< [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_cc](#) >
- using [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_sc\\_sptr](#) = gnss\_shared\_ptr< [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_sc](#) >
- using [glonass\\_l1\\_ca\\_dll\\_pll\\_tracking\\_cc\\_sptr](#) = gnss\_shared\_ptr< [Glonass\\_L1\\_Ca\\_Dll\\_Pll\\_Tracking\\_cc](#) >
- using [glonass\\_l2\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_cc\\_sptr](#) = gnss\_shared\_ptr< [glonass\\_l2\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_cc](#) >
- using [glonass\\_l2\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_sc\\_sptr](#) = gnss\_shared\_ptr< [glonass\\_l2\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_sc](#) >
- using [glonass\\_l2\\_ca\\_dll\\_pll\\_tracking\\_cc\\_sptr](#) = gnss\_shared\_ptr< [Glonass\\_L2\\_Ca\\_Dll\\_Pll\\_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\\_kf\\_tracking\\_cc\\_sptr](#) = gnss\_shared\_ptr< [Gps\\_L1\\_Ca\\_Kf\\_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)`
- `glonass_l1_ca_dll_pll_c_aid_tracking_cc_sptr glonass_l1_ca_dll_pll_c_aid_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 pll_bw_narrow_hz, float dll_bw_narrow_hz, int32_t extend_correlation_ms, float early_late_space_chips)`
- `glonass_l1_ca_dll_pll_c_aid_tracking_sc_sptr glonass_l1_ca_dll_pll_c_aid_make_tracking_sc (int64_t fs_in, uint32_t vector_length, bool dump, const std::string &dump_filename, float pll_bw_hz, float dll_bw_hz, float pll_bw_narrow_hz, float dll_bw_narrow_hz, int32_t extend_correlation_ms, float early_late_space_chips)`
- `glonass_l1_ca_dll_pll_tracking_cc_sptr glonass_l1_ca_dll_pll_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)`
- `glonass_l2_ca_dll_pll_c_aid_tracking_cc_sptr glonass_l2_ca_dll_pll_c_aid_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 pll_bw_narrow_hz, float dll_bw_narrow_hz, int32_t extend_correlation_ms, float early_late_space_chips)`
- `glonass_l2_ca_dll_pll_c_aid_tracking_sc_sptr glonass_l2_ca_dll_pll_c_aid_make_tracking_sc (int64_t fs_in, uint32_t vector_length, bool dump, const std::string &dump_filename, float pll_bw_hz, float dll_bw_hz, float pll_bw_narrow_hz, float dll_bw_narrow_hz, int32_t extend_correlation_ms, float early_late_space_chips)`
- `glonass_l2_ca_dll_pll_tracking_cc_sptr glonass_l2_ca_dll_pll_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)`
- `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, const std::string &dump_filename, float pll_bw_hz, float dll_bw_hz, float early_late_space_chips)`
- `gps_l1_ca_kf_tracking_cc_sptr gps_l1_ca_kf_make_tracking_cc (uint32_t order, int64_t if_freq, 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)`

### 9.42.1 Detailed Description

GNU Radio blocks for GNSS signal tracking.

## 9.43 tracking\_libs

### Classes

- class [Bayesian\\_estimator](#)  
*Bayesian\_estimator is an estimator of noise characteristics (i.e. mean, covariance)*
- 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](#) (gr\_complex \*Prompt\_buffer, int length)  
*A carrier lock detector.*
- double [fil\\_four\\_quadrant\\_atan](#) (gr\_complex prompt\_s1, gr\_complex prompt\_s2, double t1, double t2)
- double [fil\\_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)

### 9.43.1 Detailed Description

Utilities for GNSS signal tracking.

### 9.43.2 Function Documentation

### 9.43.2.1 carrier\_lock\_detector()

```
float carrier_lock_detector (
    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

$$C2\phi = \frac{NBD}{NBP}, \quad (9.1)$$

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.

### 9.43.2.2 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) ( $\rho$ ) estimator using the Moments Method:

$$\hat{\rho} = \frac{\sqrt{2\hat{M}_2^2 - \hat{M}_4}}{\hat{M}_2 - \sqrt{2\hat{M}_2^2 - \hat{M}_4}}, \quad (9.2)$$

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:

$$CN0_{dB} = 10 * \log(\hat{\rho}) - 10 * \log(T_{int}), \quad (9.3)$$

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.

## 9.43.2.3 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) (  $\rho$  ) estimator using the Signal-to-Noise Variance (SNV) estimator:

$$\hat{\rho} = \frac{\hat{P}_s}{\hat{P}_n} = \frac{\hat{P}_s}{\hat{P}_{tot} - \hat{P}_s}, \quad (9.4)$$

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:

$$CN0_{dB} = 10 * \log(\hat{\rho}) - 10 * \log(T_{int}), \quad (9.5)$$

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.

## 9.43.2.4 dll\_nc\_e\_minus\_l\_normalized()

```
double dll_nc_e_minus_l_normalized (
    gr_complex early_sl,
    gr_complex late_sl,
    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:

$$error = \frac{y_{intercept} - slope * \epsilon}{slope} \frac{E - L}{E + L}, \quad (9.6)$$

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].

## 9.43.2.5 dll\_nc\_vemlp\_normalized()

```
double dll_nc_vemlp_normalized (
    gr_complex very_early_sl,
    gr_complex early_sl,
    gr_complex late_sl,
    gr_complex very_late_sl )
```

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):

$$error = \frac{E - L}{E + L}, \quad (9.7)$$

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].

#### 9.43.2.6 fl\_four\_quadrant\_atan()

```
double fl_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:

$$\frac{\phi_2 - \phi_1}{t_2 - t_1} = \frac{ATAN2(cross, dot)}{t_1 - t_2}, \quad (9.8)$$

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].

#### 9.43.2.7 phase\_unwrap()

```
double phase_unwrap (
    double phase_rad )
```

Phase unwrapping function, input is [rad].

#### 9.43.2.8 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:

$$\phi = ATAN\left(\frac{Q_{PS}}{I_{PS}}\right), \quad (9.9)$$

where  $I_{PS1}, Q_{PS1}$  are the inphase and quadrature prompt correlator outputs respectively. The output is in [radians].

#### 9.43.2.9 pll\_four\_quadrant\_atan()

```
double pll_four_quadrant_atan (
    gr_complex prompt_s1 )
```

PLL four quadrant arctan discriminator.

PLL four quadrant arctan discriminator:

$$\phi = ATAN2(Q_{PS}, I_{PS}), \quad (9.10)$$

where  $I_{PS1}, Q_{PS1}$  are the inphase and quadrature prompt correlator outputs respectively. The output is in [radians].

## 9.44 Core GNSS Receiver

### Modules

- [GNSS block interfaces](#)
- [core\\_libs](#)
- [core\\_monitor](#)
- [core\\_receiver](#)
- [core\\_system\\_parameters](#)

### 9.44.1 Detailed Description

Core GNSS Receiver.

## 9.45 GNSS block interfaces

### Classes

- 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 [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

- `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)`

#### 9.45.1 Detailed Description

GNSS block interfaces.

## 9.46 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 [gnss\\_sdr\\_fpga\\_sample\\_counter](#)
- 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 [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 `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 >`

### 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)`
- `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 pairs (whitespace stripped), and comments starting with ';' (semicolon). Section is "" if name=value pair parsed before any section heading.*
- `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.*

## 9.46.1 Detailed Description

Utilities for the core GNSS receiver.

## 9.46.2 Function Documentation

### 9.46.2.1 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.

### 9.46.2.2 ini\_parse()

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

Parse given INI-style file. May have [section]s, name=value pairs (whitespace stripped), and comments starting with ';' (semicolon). Section is "" if name=value pair parsed before any section heading.

For each name=value pair 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.

## 9.47 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, int udp\_port, const std::vector< std::string > &udp\_addresses, bool enable\_protobuf)

#### 9.47.1 Detailed Description

Classes for the [Gnss\\_Synchro](#) monitor.

## 9.48 core\_receiver

### 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 [GNSSBlockFactory](#)

*Class that produces all kinds of GNSS blocks.*

- class [GNSSFlowgraph](#)

*This class represents a GNSS flow graph.*

- class [InMemoryConfiguration](#)

*This class is an implementation of the interface [ConfigurationInterface](#).*

- class [TcpCmdInterface](#)

### 9.48.1 Detailed Description

Classes for the core GNSS receiver.

## 9.49 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\\_Dnav\\_Almanac](#)  
*This class is a storage for the BeiDou D1 almanac.*
- class [Beidou\\_Dnav\\_Ephemeris](#)  
*This class is a storage and orbital model functions for the GPS 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 a BeiDou D1 NAV Data message.*
- 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 E6-B Signal-In-Space Message Specification v1.2 (April 2020)*
- class [Galileo\\_Ephemeris](#)  
*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.pdf>)*
- 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 1.2 (Nov 2015). See <https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo-OS-SIS-ICD.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 E6-B Signal-In-Space Message Specification v1.2 (April 2020).*
- 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 1.2 (Nov 2015). See <https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo-OS-SIS-ICD.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\\_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.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\\_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 RRLT 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-200K.*
- class [Gps\\_CNAV\\_Ephemeris](#)  
*This class is a storage and orbital model functions for the GPS SV ephemeris data as described in IS-GPS-200K.*
- class [Gps\\_CNAV\\_Iono](#)  
*This class is a storage for the GPS IONOSPHERIC data as described in IS-GPS-200K.*
- class [Gps\\_CNAV\\_Navigation\\_Message](#)  
*This class decodes a GPS CNAV Data message as described in IS-GPS-200K.*
- class [Gps\\_CNAV\\_Utc\\_Model](#)  
*This class is a storage for the GPS UTC MODEL data as described in in IS-GPS-200K.*
- class [Gps\\_Ephemeris](#)  
*This class is a storage and orbital model functions for the GPS SV ephemeris data as described in IS-GPS-200K.*
- class [Gps\\_Iono](#)  
*This class is a storage for the GPS IONOSPHERIC data as described in IS-GPS-200K.*
- class [Gps\\_Navigation\\_Message](#)  
*This class decodes a GPS NAV Data message as described in IS-GPS-200K.*
- class [Gps\\_Utc\\_Model](#)  
*This class is a storage for the GPS UTC MODEL data as described in IS-GPS-200K.*
- class [Sbas\\_Ephemeris](#)  
*This class stores SBAS SV ephemeris data.*

## Macros

- `#define DISPLAY_COLORS 1`
- `#define GLONASS_GNAV_PREAMBLE`

## Functions

- `const std::vector< std::pair< int32_t, int32_t > > D1_PRE {{{1, 1}}}`
- `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, 11}}}`
- `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_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_PNUM ({43, 4})`
- `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::pair< int32_t, int32_t > GALILEO_HAS_STATUS ({1, 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_URA_FLAG ({19, 1})`
- `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_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_DELTA12_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_DELTA1_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\_DELTA12\_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\_DELTA12\_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 > > A10_5_BIT {{{7, 11}}}`
- `const std::vector< std::pair< int32_t, int32_t > > A11_5_BIT {{{18, 11}}}`
- `const std::vector< std::pair< int32_t, int32_t > > A12_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 > > 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::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 > > 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_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 {{{271, 1}}}`
- `const std::vector< std::pair< int32_t, int32_t > > AODO {{{272, 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 > > 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}})

## 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 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] = "1111110000001100001100"
- constexpr char **BEIDOU\_B1I\_D2\_SECONDARY\_CODE\_STR** [3] = "00"
- 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 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\_SECOND** \* BEIDOU\_B3I\_TELEMETRY\_SYMBOLS\_PER\_BIT
- constexpr char **BEIDOU\_B3I\_SECONDARY\_CODE\_STR** [21] = "00000100110101001110"

- constexpr char **BEIDOU\_B3I\_GEO\_PREAMBLE\_SYMBOLS\_STR** [23] = "1111110000001100001100"
- constexpr char **BEIDOU\_B3I\_D2\_SECONDARY\_CODE\_STR** [3] = "00"
- 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 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 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"`
- `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_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 = 14`
- `constexpr size_t HAS_MSG_DELTA_ALONG_TRACK_LENGTH = 12`
- `constexpr size_t HAS_MSG_DELTA_CROSS_TRACK_LENGTH = 12`
- `constexpr size_t HAS_MSG_DELTA_CLOCK_C0_MULTIPLIER_LENGTH = 2`
- `constexpr size_t HAS_MSG_DELTA_CLOCK_C0_LENGTH = 14`
- `constexpr size_t HAS_MSG_NSYSPRIME_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_C0_SUBSET_LENGTH = 14`
- `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 size_t HAS_MSG_URA_LENGTH = 2`
- `constexpr int32_t GALILEO_CNAV_SYMBOLS_PER_PAGE = 1000`

*Total number of symbols per HAS page including the sync pattern.*

- `constexpr int32_t GALILEO_CNAV_PREAMBLE_PERIOD_SYMBOLS = 1000`
- `constexpr int32_t GALILEO_CNAV_PAGE_MS = 1`

*Duration in ms of a CNAV page.*

- `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_BITS_FOR_CRC = GALILEO_CNAV_HAS_PAGE_DATA_BITS + GALILEO_CNAV_PAGE_RESERVED_BITS`
- `constexpr int32_t GALILEO_CNAV_BYTES_FOR_CRC = 60`
- `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_ENCODED_BLOCKS = 255`
- `constexpr int32_t GALILEO_CNAV_MT1_HEADER_BITS = 32`
- `constexpr int32_t HAS_MSG_MAX_SATS = 40`
- `constexpr int32_t HAS_MSG_MAX_SIGNALS = 16`
- `constexpr uint8_t HAS_MSG_GPS_SYSTEM = 0`
- `constexpr uint8_t HAS_MSG_GALILEO_SYSTEM = 2`
- `constexpr char GALILEO_CNAV_PREAMBLE [17] = "1011011101110000"`
- `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 interpolation.*
- `constexpr char GALILEO_E1_C_SECONDARY_CODE [26] = "0011100000001010110110010"`
- `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\_DELTA12\_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 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 **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_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_CHIP_PERIOD_S` = 1.9569e-06  
*GLONASS L1 C/A chip period  $[seconds]$ .*
- constexpr double `GLONASS_L1_CA_SYMBOL_RATE_BPS` = 1000.0
- constexpr int32\_t `GLONASS_CA_NBR_SATS` = 24
- constexpr int32\_t `GLONASS_L1_CA_HISTORY_DEEP` = 100
- constexpr double `GLONASS_GNAV_PREAMBLE_DURATION_S` = 0.300
- 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_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_BITS_SECOND` \* `GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_BIT`  
*NAV message bit rate  $[symbols/s]$ .*
- constexpr int32\_t `GLONASS_GNAV_STRING_SYMBOLS` = 2000  
*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}
- 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)*

- 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 int32_t MAXCODE = 55`
- 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 double CNAV_A0_LSB = TWO_N35`

- Generated by Doxygen

- 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 **T\_OA\_LSB** = TWO\_P12
- 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\_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 int32\_t **GPS\_L2\_M\_CODE\_LENGTH\_CHIPS** = 10230  
*GPS L2 M code length [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, 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-200K, pag. 92.*
- constexpr double `GPS_F` = -4.442807633e-10  
*Constant,  $[s/(m)^{1/2}]$ , IS-GPS-200K, pag. 92.*
- constexpr double `GALILEO_GM` = 3.986004418e14  
*Geocentric gravitational constant  $[m^3/s^2]$ , OS SIS ICD v1.3, pag. 44.*
- constexpr double `GALILEO_F` = -4.442807309e-10  
*Constant,  $[s/(m)^{1/2}]$ . OS SIS ICD v1.3, 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_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 = 0.00003051757813
- $2^{-15}$
- constexpr double TWO\_N16 = 0.0000152587890625
- $2^{-16}$
- 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\_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\_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.*

### 9.49.1 Detailed Description

Classes containing info about system parameters for the different GNSS.

GNSS parameters

## 9.49.2 Macro Definition Documentation

### 9.49.2.1 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 90 of file GLONASS\_L1\_L2\_CA.h.

## 9.49.3 Function Documentation

### 9.49.3.1 ALPHA\_0()

```
const std::vector<std::pair<int32_t, int32_t> > ALPHA_0 (  
    {{69, 8}} )
```

**Todo** read all pages of subframe 4

### 9.49.3.2 T\_OA()

```
const std::vector<std::pair<int32_t, int32_t> > T_OA (  
    {{69, 8}} )
```

**Todo** read all pages of subframe 5

## 9.49.4 Variable Documentation

#### 9.49.4.1 AS2R

```
constexpr double AS2R = D2R / 3600.0
```

arc sec to radian

Definition at line 123 of file MATH\_CONSTANTS.h.

#### 9.49.4.2 AU

```
constexpr double AU = 149597870691.0
```

1 Astronomical Unit AU (m) distance from Earth to the Sun.

Definition at line 125 of file MATH\_CONSTANTS.h.

#### 9.49.4.3 BEIDOU\_B1I\_CODE\_LENGTH\_CHIPS

```
constexpr double BEIDOU_B1I_CODE_LENGTH_CHIPS = 2046.0
```

Beidou B1I code length [chips].

Definition at line 34 of file Beidou\_B1I.h.

#### 9.49.4.4 BEIDOU\_B1I\_CODE\_PERIOD\_MS

```
constexpr uint32_t BEIDOU_B1I_CODE_PERIOD_MS = 1
```

Beidou B1I code period [ms].

Definition at line 37 of file Beidou\_B1I.h.

#### 9.49.4.5 BEIDOU\_B1I\_CODE\_PERIOD\_S

```
constexpr double BEIDOU_B1I_CODE_PERIOD_S = 0.001
```

Beidou B1I code period [seconds].

Definition at line 35 of file Beidou\_B1I.h.

#### 9.49.4.6 BEIDOU\_B1I\_CODE\_RATE\_CPS

```
constexpr double BEIDOU_B1I_CODE_RATE_CPS = 2.046e6
```

Beidou B1I code rate [chips/s].

Definition at line 33 of file Beidou\_B1I.h.

#### 9.49.4.7 BEIDOU\_B1I\_FREQ\_HZ

```
constexpr double BEIDOU_B1I_FREQ_HZ = FREQ1\_BDS
```

B1I [Hz].

Definition at line 32 of file Beidou\_B1I.h.

#### 9.49.4.8 BEIDOU\_B3I\_CODE\_LENGTH\_CHIPS

```
constexpr double BEIDOU_B3I_CODE_LENGTH_CHIPS = 10230.0
```

BeiDou B3I code length [chips].

Definition at line 33 of file Beidou\_B3I.h.

#### 9.49.4.9 BEIDOU\_B3I\_CODE\_PERIOD\_MS

```
constexpr uint32_t BEIDOU_B3I_CODE_PERIOD_MS = 1
```

BeiDou B3I code period [ms].

Definition at line 36 of file Beidou\_B3I.h.

#### 9.49.4.10 BEIDOU\_B3I\_CODE\_PERIOD\_S

```
constexpr double BEIDOU_B3I_CODE_PERIOD_S = 0.001
```

BeiDou B3I code period [seconds].

Definition at line 34 of file Beidou\_B3I.h.

#### 9.49.4.11 BEIDOU\_B3I\_CODE\_RATE\_CPS

```
constexpr double BEIDOU_B3I_CODE_RATE_CPS = 10.23e6
```

BeiDou B3I code rate [chips/s].

Definition at line 32 of file Beidou\_B3I.h.

#### 9.49.4.12 BEIDOU\_B3I\_FREQ\_HZ

```
constexpr double BEIDOU_B3I_FREQ_HZ = FREQ3\_BDS
```

BeiDou B3I [Hz].

Definition at line 31 of file Beidou\_B3I.h.

#### 9.49.4.13 BEIDOU\_B3I\_TELEMETRY\_RATE\_BITS\_SECOND

```
constexpr int32_t BEIDOU_B3I_TELEMETRY_RATE_BITS_SECOND = 50
```

D1 NAV message bit rate [bits/s].

Definition at line 42 of file Beidou\_B3I.h.

#### 9.49.4.14 BEIDOU\_DNAV\_SUBFRAME\_DATA\_BITS

```
constexpr int32_t BEIDOU_DNAV_SUBFRAME_DATA_BITS = 300
```

Number of bits per subframe in the NAV message [bits].

Definition at line 87 of file Beidou\_DNAV.h.

#### 9.49.4.15 BEIDOU\_F

```
constexpr double BEIDOU_F = -4.442807309e-10
```

Constant,  $[s/(m)^{(1/2)}] F = -2(GM)^{.5}/C^2$ .

Definition at line 45 of file MATH\_CONSTANTS.h.

**9.49.4.16 BEIDOU\_GM**

```
constexpr double BEIDOU_GM = 3.986004418e14
```

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.

**9.49.4.17 BEIDOU\_OMEGA\_EARTH\_DOT**

```
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.

Definition at line 43 of file MATH\_CONSTANTS.h.

**9.49.4.18 CODE\_L1A**

```
constexpr uint32_t CODE_L1A = 10
```

obs code: E1A (GAL)

Definition at line 40 of file gnss\_obs\_codes.h.

**9.49.4.19 CODE\_L1B**

```
constexpr uint32_t CODE_L1B = 11
```

obs code: E1B (GAL)

Definition at line 41 of file gnss\_obs\_codes.h.

**9.49.4.20 CODE\_L1C**

```
constexpr uint32_t CODE_L1C = 1
```

obs code: L1C/A,G1C/A,E1C (GPS,GLO,GAL,QZS,SBS)

Definition at line 31 of file gnss\_obs\_codes.h.

#### 9.49.4.21 CODE\_L1E

```
constexpr uint32_t CODE_L1E = 9
```

(not used)

Definition at line 39 of file gnss\_obs\_codes.h.

#### 9.49.4.22 CODE\_L1I

```
constexpr uint32_t CODE_L1I = 47
```

obs code: B1I (BDS)

Definition at line 77 of file gnss\_obs\_codes.h.

#### 9.49.4.23 CODE\_L1L

```
constexpr uint32_t CODE_L1L = 8
```

obs code: L1C(P) (GPS,QZS)

Definition at line 38 of file gnss\_obs\_codes.h.

#### 9.49.4.24 CODE\_L1M

```
constexpr uint32_t CODE_L1M = 5
```

obs code: L1M (GPS)

Definition at line 35 of file gnss\_obs\_codes.h.

#### 9.49.4.25 CODE\_L1N

```
constexpr uint32_t CODE_L1N = 6
```

obs code: L1codeless (GPS)

Definition at line 36 of file gnss\_obs\_codes.h.

**9.49.4.26 CODE\_L1P**

```
constexpr uint32_t CODE_L1P = 2
```

obs code: L1P,G1P (GPS,GLO)

Definition at line 32 of file gnss\_obs\_codes.h.

**9.49.4.27 CODE\_L1Q**

```
constexpr uint32_t CODE_L1Q = 48
```

obs code: B1Q (BDS)

Definition at line 78 of file gnss\_obs\_codes.h.

**9.49.4.28 CODE\_L1S**

```
constexpr uint32_t CODE_L1S = 7
```

obs code: L1C(D) (GPS,QZS)

Definition at line 37 of file gnss\_obs\_codes.h.

**9.49.4.29 CODE\_L1W**

```
constexpr uint32_t CODE_L1W = 3
```

obs code: L1 Z-track (GPS)

Definition at line 33 of file gnss\_obs\_codes.h.

**9.49.4.30 CODE\_L1X**

```
constexpr uint32_t CODE_L1X = 12
```

obs code: E1B+C,L1C(D+P) (GAL,QZS)

Definition at line 42 of file gnss\_obs\_codes.h.

#### 9.49.4.31 CODE\_L1Y

```
constexpr uint32_t CODE_L1Y = 4
```

obs code: L1Y (GPS)

Definition at line 34 of file gnss\_obs\_codes.h.

#### 9.49.4.32 CODE\_L1Z

```
constexpr uint32_t CODE_L1Z = 13
```

obs code: E1A+B+C,L1SAIF (GAL,QZS)

Definition at line 43 of file gnss\_obs\_codes.h.

#### 9.49.4.33 CODE\_L2C

```
constexpr uint32_t CODE_L2C = 14
```

obs code: L2C/A,G1C/A (GPS,GLO)

Definition at line 44 of file gnss\_obs\_codes.h.

#### 9.49.4.34 CODE\_L2D

```
constexpr uint32_t CODE_L2D = 15
```

obs code: L2 L1C/A-(P2-P1) (GPS)

Definition at line 45 of file gnss\_obs\_codes.h.

#### 9.49.4.35 CODE\_L2I

```
constexpr uint32_t CODE_L2I = 40
```

obs code: B1I (BDS)

Definition at line 70 of file gnss\_obs\_codes.h.

**9.49.4.36 CODE\_L2L**

```
constexpr uint32_t CODE_L2L = 17
```

obs code: L2C(L) (GPS,QZS)

Definition at line 47 of file gnss\_obs\_codes.h.

**9.49.4.37 CODE\_L2M**

```
constexpr uint32_t CODE_L2M = 22
```

obs code: L2M (GPS)

Definition at line 52 of file gnss\_obs\_codes.h.

**9.49.4.38 CODE\_L2N**

```
constexpr uint32_t CODE_L2N = 23
```

obs code: L2codeless (GPS)

Definition at line 53 of file gnss\_obs\_codes.h.

**9.49.4.39 CODE\_L2P**

```
constexpr uint32_t CODE_L2P = 19
```

obs code: L2P,G2P (GPS,GLO)

Definition at line 49 of file gnss\_obs\_codes.h.

**9.49.4.40 CODE\_L2Q**

```
constexpr uint32_t CODE_L2Q = 41
```

obs code: B1Q (BDS)

Definition at line 71 of file gnss\_obs\_codes.h.

#### 9.49.4.41 CODE\_L2S

```
constexpr uint32_t CODE_L2S = 16
```

obs code: L2C(M) (GPS,QZS)

Definition at line 46 of file gnss\_obs\_codes.h.

#### 9.49.4.42 CODE\_L2W

```
constexpr uint32_t CODE_L2W = 20
```

obs code: L2 Z-track (GPS)

Definition at line 50 of file gnss\_obs\_codes.h.

#### 9.49.4.43 CODE\_L2X

```
constexpr uint32_t CODE_L2X = 18
```

obs code: L2C(M+L),B1I+Q (GPS,QZS,BDS)

Definition at line 48 of file gnss\_obs\_codes.h.

#### 9.49.4.44 CODE\_L2Y

```
constexpr uint32_t CODE_L2Y = 21
```

obs code: L2Y (GPS)

Definition at line 51 of file gnss\_obs\_codes.h.

#### 9.49.4.45 CODE\_L3I

```
constexpr uint32_t CODE_L3I = 44
```

obs code: G3I (GLO)

Definition at line 74 of file gnss\_obs\_codes.h.

**9.49.4.46 CODE\_L3Q**

```
constexpr uint32_t CODE_L3Q = 45
```

obs code: G3Q (GLO)

Definition at line 75 of file gnss\_obs\_codes.h.

**9.49.4.47 CODE\_L3X**

```
constexpr uint32_t CODE_L3X = 46
```

obs code: G3I+Q (GLO)

Definition at line 76 of file gnss\_obs\_codes.h.

**9.49.4.48 CODE\_L5A**

```
constexpr uint32_t CODE_L5A = 49
```

obs code: L5A SPS (IRN)

Definition at line 79 of file gnss\_obs\_codes.h.

**9.49.4.49 CODE\_L5B**

```
constexpr uint32_t CODE_L5B = 50
```

obs code: L5B RS(D) (IRN)

Definition at line 80 of file gnss\_obs\_codes.h.

**9.49.4.50 CODE\_L5C**

```
constexpr uint32_t CODE_L5C = 51
```

obs code: L5C RS(P) (IRN)

Definition at line 81 of file gnss\_obs\_codes.h.

#### 9.49.4.51 CODE\_L5I

```
constexpr uint32_t CODE_L5I = 24
```

obs code: L5/E5aI (GPS,GAL,QZS,SBS)

Definition at line 54 of file gnss\_obs\_codes.h.

#### 9.49.4.52 CODE\_L5Q

```
constexpr uint32_t CODE_L5Q = 25
```

obs code: L5/E5aQ (GPS,GAL,QZS,SBS)

Definition at line 55 of file gnss\_obs\_codes.h.

#### 9.49.4.53 CODE\_L5X

```
constexpr uint32_t CODE_L5X = 26
```

obs code: L5/E5aI+Q/L5B+C (GPS,GAL,QZS,IRN,SBS)

Definition at line 56 of file gnss\_obs\_codes.h.

#### 9.49.4.54 CODE\_L6A

```
constexpr uint32_t CODE_L6A = 30
```

obs code: E6A (GAL)

Definition at line 60 of file gnss\_obs\_codes.h.

#### 9.49.4.55 CODE\_L6B

```
constexpr uint32_t CODE_L6B = 31
```

obs code: E6B (GAL)

Definition at line 61 of file gnss\_obs\_codes.h.

**9.49.4.56 CODE\_L6C**

```
constexpr uint32_t CODE_L6C = 32
```

obs code: E6C (GAL)

Definition at line 62 of file gnss\_obs\_codes.h.

**9.49.4.57 CODE\_L6I**

```
constexpr uint32_t CODE_L6I = 42
```

obs code: B3I (BDS)

Definition at line 72 of file gnss\_obs\_codes.h.

**9.49.4.58 CODE\_L6L**

```
constexpr uint32_t CODE_L6L = 36
```

obs code: LEXL (QZS)

Definition at line 66 of file gnss\_obs\_codes.h.

**9.49.4.59 CODE\_L6Q**

```
constexpr uint32_t CODE_L6Q = 43
```

obs code: B3Q (BDS)

Definition at line 73 of file gnss\_obs\_codes.h.

**9.49.4.60 CODE\_L6S**

```
constexpr uint32_t CODE_L6S = 35
```

obs code: LEXS (QZS)

Definition at line 65 of file gnss\_obs\_codes.h.

#### 9.49.4.61 CODE\_L6X

```
constexpr uint32_t CODE_L6X = 33
```

obs code: E6B+C,LEXS+L,B3I+Q (GAL,QZS,BDS)

Definition at line 63 of file gnss\_obs\_codes.h.

#### 9.49.4.62 CODE\_L6Z

```
constexpr uint32_t CODE_L6Z = 34
```

obs code: E6A+B+C (GAL)

Definition at line 64 of file gnss\_obs\_codes.h.

#### 9.49.4.63 CODE\_L7I

```
constexpr uint32_t CODE_L7I = 27
```

obs code: E5bI,B2I (GAL,BDS)

Definition at line 57 of file gnss\_obs\_codes.h.

#### 9.49.4.64 CODE\_L7Q

```
constexpr uint32_t CODE_L7Q = 28
```

obs code: E5bQ,B2Q (GAL,BDS)

Definition at line 58 of file gnss\_obs\_codes.h.

#### 9.49.4.65 CODE\_L7X

```
constexpr uint32_t CODE_L7X = 29
```

obs code: E5bI+Q,B2I+Q (GAL,BDS)

Definition at line 59 of file gnss\_obs\_codes.h.

**9.49.4.66 CODE\_L8I**

```
constexpr uint32_t CODE_L8I = 37
```

obs code: E5(a+b)I (GAL)

Definition at line 67 of file gnss\_obs\_codes.h.

**9.49.4.67 CODE\_L8Q**

```
constexpr uint32_t CODE_L8Q = 38
```

obs code: E5(a+b)Q (GAL)

Definition at line 68 of file gnss\_obs\_codes.h.

**9.49.4.68 CODE\_L8X**

```
constexpr uint32_t CODE_L8X = 39
```

obs code: E5(a+b)I+Q (GAL)

Definition at line 69 of file gnss\_obs\_codes.h.

**9.49.4.69 CODE\_L9A**

```
constexpr uint32_t CODE_L9A = 52
```

obs code: SA SPS (IRN)

Definition at line 82 of file gnss\_obs\_codes.h.

**9.49.4.70 CODE\_L9B**

```
constexpr uint32_t CODE_L9B = 53
```

obs code: SB RS(D) (IRN)

Definition at line 83 of file gnss\_obs\_codes.h.

#### 9.49.4.71 CODE\_L9C

```
constexpr uint32_t CODE_L9C = 54
```

obs code: SC RS(P) (IRN)

Definition at line 84 of file gnss\_obs\_codes.h.

#### 9.49.4.72 CODE\_L9X

```
constexpr uint32_t CODE_L9X = 55
```

obs code: SB+C (IRN)

Definition at line 85 of file gnss\_obs\_codes.h.

#### 9.49.4.73 CODE\_NONE

```
constexpr uint32_t CODE_NONE = 0
```

obs code: none or unknown

Definition at line 30 of file gnss\_obs\_codes.h.

#### 9.49.4.74 D2R

```
constexpr double D2R = GNSS_PI / 180.0
```

deg to rad

Definition at line 120 of file MATH\_CONSTANTS.h.

#### 9.49.4.75 DFRQ1\_GLO

```
constexpr double DFRQ1_GLO = 0.56250e6
```

GLONASS G1 bias frequency (Hz/n)

Definition at line 36 of file gnss\_frequencies.h.

**9.49.4.76 DFRQ2\_GLO**

```
constexpr double DFRQ2_GLO = 0.43750e6
```

GLONASS G2 bias frequency (Hz/n)

Definition at line 38 of file gnss\_frequencies.h.

**9.49.4.77 FREQ1**

```
constexpr double FREQ1 = 1.57542e9
```

L1/E1 frequency (Hz)

Definition at line 28 of file gnss\_frequencies.h.

**9.49.4.78 FREQ1\_BDS**

```
constexpr double FREQ1_BDS = 1.561098e9
```

BeiDou B1 frequency (Hz)

Definition at line 40 of file gnss\_frequencies.h.

**9.49.4.79 FREQ1\_GLO**

```
constexpr double FREQ1_GLO = 1.60200e9
```

GLONASS G1 base frequency (Hz)

Definition at line 35 of file gnss\_frequencies.h.

**9.49.4.80 FREQ2**

```
constexpr double FREQ2 = 1.22760e9
```

L2 frequency (Hz)

Definition at line 29 of file gnss\_frequencies.h.

#### 9.49.4.81 `FREQ2_BDS`

```
constexpr double FREQ2_BDS = 1.20714e9
```

BeiDou B2 frequency (Hz)

Definition at line 41 of file `gnss_frequencies.h`.

#### 9.49.4.82 `FREQ2_GLO`

```
constexpr double FREQ2_GLO = 1.24600e9
```

GLONASS G2 base frequency (Hz)

Definition at line 37 of file `gnss_frequencies.h`.

#### 9.49.4.83 `FREQ3_BDS`

```
constexpr double FREQ3_BDS = 1.26852e9
```

BeiDou B3 frequency (Hz)

Definition at line 42 of file `gnss_frequencies.h`.

#### 9.49.4.84 `FREQ3_GLO`

```
constexpr double FREQ3_GLO = 1.202025e9
```

GLONASS G3 frequency (Hz)

Definition at line 39 of file `gnss_frequencies.h`.

#### 9.49.4.85 `FREQ5`

```
constexpr double FREQ5 = 1.17645e9
```

L5/E5a frequency (Hz)

Definition at line 30 of file `gnss_frequencies.h`.

**9.49.4.86   FREQ6**

```
constexpr double FREQ6 = 1.27875e9
```

E6/LEX frequency (Hz)

Definition at line 31 of file gnss\_frequencies.h.

**9.49.4.87   FREQ7**

```
constexpr double FREQ7 = 1.20714e9
```

E5b frequency (Hz)

Definition at line 32 of file gnss\_frequencies.h.

**9.49.4.88   FREQ8**

```
constexpr double FREQ8 = 1.191795e9
```

E5a+b frequency (Hz)

Definition at line 33 of file gnss\_frequencies.h.

**9.49.4.89   FREQ9**

```
constexpr double FREQ9 = 2.492028e9
```

S frequency (Hz)

Definition at line 34 of file gnss\_frequencies.h.

**9.49.4.90   GALILEO\_CNAV\_PAGE\_MS**

```
constexpr int32_t GALILEO_CNAV_PAGE_MS = 1
```

Duration in ms of a CNAV page.

Definition at line 59 of file Galileo\_CNAV.h.

#### 9.49.4.91 GALILEO\_CNAV\_SYMBOLS\_PER\_PAGE

```
constexpr int32_t GALILEO_CNAV_SYMBOLS_PER_PAGE = 1000
```

Total number of symbols per HAS page including the sync pattern.

Definition at line 57 of file Galileo\_CNAV.h.

#### 9.49.4.92 GALILEO\_E1\_B\_CODE\_LENGTH\_CHIPS

```
constexpr double GALILEO_E1_B_CODE_LENGTH_CHIPS = 4092.0
```

Galileo E1-B code length [chips].

Definition at line 39 of file Galileo\_E1.h.

#### 9.49.4.93 GALILEO\_E1\_B\_SAMPLES\_PER\_SYMBOL

```
constexpr int32_t GALILEO_E1_B_SAMPLES_PER_SYMBOL = 1
```

$(\text{Galileo\_E1\_CODE\_CHIP\_RATE\_HZ} / \text{Galileo\_E1\_B\_CODE\_LENGTH\_CHIPS}) / \text{Galileo\_E1\_B\_SYMBOL\_RATE\_BPS}$

Definition at line 42 of file Galileo\_E1.h.

#### 9.49.4.94 GALILEO\_E1\_B\_SYMBOL\_RATE\_BPS

```
constexpr double GALILEO_E1_B_SYMBOL_RATE_BPS = 250.0
```

Galileo E1-B symbol rate [bits/second].

Definition at line 40 of file Galileo\_E1.h.

#### 9.49.4.95 GALILEO\_E1\_C\_SECONDARY\_CODE\_LENGTH

```
constexpr int32_t GALILEO_E1_C_SECONDARY_CODE_LENGTH = 25
```

Galileo E1-C secondary code length [chips].

Definition at line 43 of file Galileo\_E1.h.

**9.49.4.96 GALILEO\_E1\_CODE\_CHIP\_RATE\_CPS**

```
constexpr double GALILEO_E1_CODE_CHIP_RATE_CPS = 1.023e6
```

Galileo E1 code rate [chips/s].

Definition at line 35 of file Galileo\_E1.h.

**9.49.4.97 GALILEO\_E1\_CODE\_PERIOD\_MS**

```
constexpr uint32_t GALILEO_E1_CODE_PERIOD_MS = 4
```

Galileo E1 code period [ms].

Definition at line 41 of file Galileo\_E1.h.

**9.49.4.98 GALILEO\_E1\_CODE\_PERIOD\_S**

```
constexpr double GALILEO_E1_CODE_PERIOD_S = 0.004
```

Galileo E1 code period [s].

Definition at line 36 of file Galileo\_E1.h.

**9.49.4.99 GALILEO\_E1\_FREQ\_HZ**

```
constexpr double GALILEO_E1_FREQ_HZ = FREQ1
```

Galileo E1 carrier frequency [Hz].

Definition at line 34 of file Galileo\_E1.h.

**9.49.4.100 GALILEO\_E1\_HISTORY\_DEEP**

```
constexpr int32_t GALILEO_E1_HISTORY_DEEP = 100
```

Observable history length for interpolation.

Definition at line 50 of file Galileo\_E1.h.

**9.49.4.101 GALILEO\_E1\_OPT\_ACQ\_FS\_SPS**

```
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.

Definition at line 48 of file Galileo\_E1.h.

**9.49.4.102 GALILEO\_E1\_SUB\_CARRIER\_A\_RATE\_HZ**

```
constexpr double GALILEO_E1_SUB_CARRIER_A_RATE_HZ = 1.023e6
```

Galileo E1 sub-carrier 'a' rate [Hz].

Definition at line 37 of file Galileo\_E1.h.

**9.49.4.103 GALILEO\_E1\_SUB\_CARRIER\_B\_RATE\_HZ**

```
constexpr double GALILEO_E1_SUB_CARRIER_B_RATE_HZ = 6.138e6
```

Galileo E1 sub-carrier 'b' rate [Hz].

Definition at line 38 of file Galileo\_E1.h.

**9.49.4.104 GALILEO\_E5A\_CODE\_CHIP\_RATE\_CPS**

```
constexpr double GALILEO_E5A_CODE_CHIP_RATE_CPS = 1.023e7
```

Galileo E5a code rate [chips/s].

Definition at line 33 of file Galileo\_E5a.h.

**9.49.4.105 GALILEO\_E5A\_CODE\_LENGTH\_CHIPS**

```
constexpr int32_t GALILEO_E5A_CODE_LENGTH_CHIPS = 10230
```

Galileo E5a primary code length [chips].

Definition at line 37 of file Galileo\_E5a.h.

**9.49.4.106 GALILEO\_E5A\_CODE\_PERIOD\_MS**

```
constexpr int32_t GALILEO_E5A_CODE_PERIOD_MS = 1
```

Galileo E5a primary code period [ms].

Definition at line 40 of file Galileo\_E5a.h.

**9.49.4.107 GALILEO\_E5A\_CODE\_PERIOD\_S**

```
constexpr double GALILEO_E5A_CODE_PERIOD_S = 0.001
```

Galileo E5a primary code period [s].

Definition at line 36 of file Galileo\_E5a.h.

**9.49.4.108 GALILEO\_E5A\_FREQ\_HZ**

```
constexpr double GALILEO_E5A_FREQ_HZ = FREQ5
```

Galileo E5a carrier frequency [Hz].

Definition at line 32 of file Galileo\_E5a.h.

**9.49.4.109 GALILEO\_E5A\_I\_SECONDARY\_CODE\_LENGTH**

```
constexpr int32_t GALILEO_E5A_I_SECONDARY_CODE_LENGTH = 20
```

Galileo E5a-I secondary code length [chips].

Definition at line 38 of file Galileo\_E5a.h.

**9.49.4.110 GALILEO\_E5A\_I\_TIERED\_CODE\_PERIOD\_S**

```
constexpr double GALILEO_E5A_I_TIERED_CODE_PERIOD_S = 0.020
```

Galileo E5a-I tiered code period [s].

Definition at line 34 of file Galileo\_E5a.h.

**9.49.4.111 GALILEO\_E5A\_OPT\_ACQ\_FS\_SPS**

```
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.

Definition at line 50 of file Galileo\_E5a.h.

**9.49.4.112 GALILEO\_E5A\_Q\_SECONDARY\_CODE\_LENGTH**

```
constexpr int32_t GALILEO_E5A_Q_SECONDARY_CODE_LENGTH = 100
```

Galileo E5a-Q secondary code length [chips].

Definition at line 39 of file Galileo\_E5a.h.

**9.49.4.113 GALILEO\_E5A\_Q\_TIERED\_CODE\_PERIOD\_S**

```
constexpr double GALILEO_E5A_Q_TIERED_CODE_PERIOD_S = 0.100
```

Galileo E5a-Q tiered code period [s].

Definition at line 35 of file Galileo\_E5a.h.

**9.49.4.114 GALILEO\_E5A\_SYMBOL\_RATE\_BPS**

```
constexpr int32_t GALILEO_E5A_SYMBOL_RATE_BPS = 50
```

Galileo E5a symbol rate [bits/second].

Definition at line 41 of file Galileo\_E5a.h.

**9.49.4.115 GALILEO\_E5B\_CODE\_CHIP\_RATE\_CPS**

```
constexpr double GALILEO_E5B_CODE_CHIP_RATE_CPS = 1.023e7
```

Galileo E5b code rate [chips/s].

Definition at line 34 of file Galileo\_E5b.h.

**9.49.4.116 GALILEO\_E5B\_CODE\_LENGTH\_CHIPS**

```
constexpr int32_t GALILEO_E5B_CODE_LENGTH_CHIPS = 10230
```

Galileo E5b primary code length [chips].

Definition at line 39 of file Galileo\_E5b.h.

**9.49.4.117 GALILEO\_E5B\_CODE\_PERIOD\_MS**

```
constexpr int32_t GALILEO_E5B_CODE_PERIOD_MS = 1
```

Galileo E5b primary code period [ms].

Definition at line 38 of file Galileo\_E5b.h.

**9.49.4.118 GALILEO\_E5B\_CODE\_PERIOD\_S**

```
constexpr double GALILEO_E5B_CODE_PERIOD_S = 0.001
```

Galileo E5b primary code period [s].

Definition at line 37 of file Galileo\_E5b.h.

**9.49.4.119 GALILEO\_E5B\_FREQ\_HZ**

```
constexpr double GALILEO_E5B_FREQ_HZ = FREQ7
```

Galileo E5b carrier frequency [Hz].

Definition at line 33 of file Galileo\_E5b.h.

**9.49.4.120 GALILEO\_E5B\_I\_SECONDARY\_CODE\_LENGTH**

```
constexpr int32_t GALILEO_E5B_I_SECONDARY_CODE_LENGTH = 4
```

Galileo E5b-I secondary code length [chips].

Definition at line 40 of file Galileo\_E5b.h.

**9.49.4.121 GALILEO\_E5B\_I\_TIERED\_CODE\_PERIOD\_S**

```
constexpr double GALILEO_E5B_I_TIERED_CODE_PERIOD_S = 0.004
```

Galileo E5b-I tiered code period [s].

Definition at line 35 of file Galileo\_E5b.h.

**9.49.4.122 GALILEO\_E5B\_OPT\_ACQ\_FS\_SPS**

```
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.

Definition at line 50 of file Galileo\_E5b.h.

**9.49.4.123 GALILEO\_E5B\_Q\_SECONDARY\_CODE\_LENGTH**

```
constexpr int32_t GALILEO_E5B_Q_SECONDARY_CODE_LENGTH = 100
```

Galileo E5b-Q secondary code length [chips].

Definition at line 41 of file Galileo\_E5b.h.

**9.49.4.124 GALILEO\_E5B\_Q\_TIERED\_CODE\_PERIOD\_S**

```
constexpr double GALILEO_E5B_Q_TIERED_CODE_PERIOD_S = 0.100
```

Galileo E5b-Q tiered code period [s].

Definition at line 36 of file Galileo\_E5b.h.

**9.49.4.125 GALILEO\_E5B\_SYMBOL\_RATE\_BPS**

```
constexpr int32_t GALILEO_E5B_SYMBOL_RATE_BPS = 250
```

Galileo E5b symbol rate [bits/second].

Definition at line 42 of file Galileo\_E5b.h.

**9.49.4.126 GALILEO\_E6\_B\_CODE\_CHIP\_RATE\_CPS**

```
constexpr double GALILEO_E6_B_CODE_CHIP_RATE_CPS = 5.115e6
```

Galileo E6 B code rate [chips/s].

Definition at line 31 of file Galileo\_E6.h.

**9.49.4.127 GALILEO\_E6\_B\_CODE\_LENGTH\_CHIPS**

```
constexpr double GALILEO_E6_B_CODE_LENGTH_CHIPS = 5115.0
```

Galileo E6 B code length [chips].

Definition at line 35 of file Galileo\_E6.h.

**9.49.4.128 GALILEO\_E6\_C\_CODE\_CHIP\_RATE\_CPS**

```
constexpr double GALILEO_E6_C_CODE_CHIP_RATE_CPS = 5.115e6
```

Galileo E6 C code rate [chips/s].

Definition at line 32 of file Galileo\_E6.h.

**9.49.4.129 GALILEO\_E6\_C\_CODE\_LENGTH\_CHIPS**

```
constexpr double GALILEO_E6_C_CODE_LENGTH_CHIPS = 5115.0
```

Galileo E6 C code length [chips].

Definition at line 36 of file Galileo\_E6.h.

**9.49.4.130 GALILEO\_E6\_C\_SECONDARY\_CODE\_LENGTH\_CHIPS**

```
constexpr double GALILEO_E6_C_SECONDARY_CODE_LENGTH_CHIPS = 100.0
```

Galileo E6 C secondary code length [chips].

Definition at line 37 of file Galileo\_E6.h.

**9.49.4.131 GALILEO\_E6\_CODE\_PERIOD\_MS**

```
constexpr uint32_t GALILEO_E6_CODE_PERIOD_MS = 1
```

Galileo E& B/C code period [ms].

Definition at line 38 of file Galileo\_E6.h.

**9.49.4.132 GALILEO\_E6\_CODE\_PERIOD\_S**

```
constexpr double GALILEO_E6_CODE_PERIOD_S = 0.001
```

Galileo E6 code period [s].

Definition at line 33 of file Galileo\_E6.h.

**9.49.4.133 GALILEO\_E6\_FREQ\_HZ**

```
constexpr double GALILEO_E6_FREQ_HZ = FREQ6
```

Galileo E6 carrier frequency [Hz].

Definition at line 30 of file Galileo\_E6.h.

**9.49.4.134 GALILEO\_F**

```
constexpr double GALILEO_F = -4.442807309e-10
```

Constant,  $[s/(m)^{(1/2)}]$ . OS SIS ICD v1.3, pag. 47.

Definition at line 36 of file MATH\_CONSTANTS.h.

**9.49.4.135 GALILEO\_GM**

```
constexpr double GALILEO_GM = 3.986004418e14
```

Geocentric gravitational constant  $[m^3/s^2]$ , OS SIS ICD v1.3, pag. 44.

Definition at line 35 of file MATH\_CONSTANTS.h.

**9.49.4.136 GALILEO\_INAV\_PAGE\_PART\_SYMBOLS**

```
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.

Definition at line 37 of file Galileo\_INAV.h.

**9.49.4.137 GALILEO\_INAV\_PAGE\_PART\_WITH\_PREABLE\_SECONDS**

```
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))

Definition at line 33 of file Galileo\_INAV.h.

**9.49.4.138 GALILEO\_INAV\_PAGE\_SYMBOLS**

```
constexpr uint32_t GALILEO_INAV_PAGE_SYMBOLS = 500
```

The complete Galileo INAV page length.

Definition at line 34 of file Galileo\_INAV.h.

**9.49.4.139 GLONASS\_C20**

```
constexpr double GLONASS_C20 = -1082.63e-6
```

Second zonal coefficient of spherical harmonic expansion.

Definition at line 45 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.140 GLONASS\_EARTH\_INCLINATION**

```
constexpr double GLONASS_EARTH_INCLINATION = 0.000409148809899e3
```

Mean inclination of ecliptic to equator (23 deg 26 min 33 sec) [rad].

Definition at line 47 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.141 GLONASS\_EARTH\_RADIUS

```
constexpr double GLONASS_EARTH_RADIUS = 6378.136
```

Equatorial radius of Earth [km].

Definition at line 46 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.142 GLONASS\_F\_M\_A

```
constexpr double GLONASS_F_M_A = 0.35e9
```

Gravitational constant of atmosphere [ $\text{m}^3/\text{s}^2$ ].

Definition at line 35 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.143 GLONASS\_FLATTENING

```
constexpr double GLONASS_FLATTENING = 1.0 / 29825784.0
```

Flattening parameter.

Definition at line 37 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.144 GLONASS\_GM

```
constexpr double GLONASS_GM = 398600.44e9
```

Universal gravitational constant times the mass of the Earth, [ $\text{m}^3/\text{s}^2$ ].

Definition at line 40 of file MATH\_CONSTANTS.h.

#### 9.49.4.145 GLONASS\_GNAV\_HAMMING\_CODE\_BITS

```
constexpr int32_t GLONASS_GNAV_HAMMING_CODE_BITS = 8
```

Number of bits in hamming code sequence of GNAV message.

Definition at line 104 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.146 GLONASS\_GNAV\_STRING\_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].

Definition at line 103 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.147 GLONASS\_GNAV\_STRING\_SYMBOLS**

```
constexpr int32_t GLONASS_GNAV_STRING_SYMBOLS = 2000
```

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

Definition at line 102 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.148 GLONASS\_GNAV\_TELEMETRY\_RATE\_BITS\_SECOND**

```
constexpr int32_t GLONASS_GNAV_TELEMETRY_RATE_BITS_SECOND = 50
```

NAV message bit rate [bits/s].

Definition at line 98 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.149 GLONASS\_GNAV\_TELEMETRY\_RATE\_SYMBOLS\_SECOND**

```
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].

Definition at line 101 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.150 GLONASS\_GRAVITY**

```
constexpr double GLONASS_GRAVITY = 97803284.0
```

Equatorial acceleration of gravity [mGal].

Definition at line 38 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.151 GLONASS\_GRAVITY\_CORRECTION

```
constexpr double GLONASS_GRAVITY_CORRECTION = 0.87
```

Correction to acceleration of gravity at sea-level due to Atmosphere[uGal].

Definition at line 39 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.152 GLONASS\_J2

```
constexpr double GLONASS_J2 = 1082625.75e-9
```

Second zonal harmonic of the geopotential.

Definition at line 40 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.153 GLONASS\_J4

```
constexpr double GLONASS_J4 = -2370.89e-9
```

Fourth zonal harmonic of the geopotential.

Definition at line 41 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.154 GLONASS\_J6

```
constexpr double GLONASS_J6 = 6.08e-9
```

Sixth zonal harmonic of the geopotential.

Definition at line 42 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.155 GLONASS\_J8

```
constexpr double GLONASS_J8 = 1.40e-11
```

Eighth zonal harmonic of the geopotential.

Definition at line 43 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.156 GLONASS\_L1\_CA\_CHIP\_PERIOD\_S**

```
constexpr double GLONASS_L1_CA_CHIP_PERIOD_S = 1.9569e-06
```

GLONASS L1 C/A chip period [seconds].

Definition at line 81 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.157 GLONASS\_L1\_CA\_CODE\_LENGTH\_CHIPS**

```
constexpr double GLONASS_L1_CA_CODE_LENGTH_CHIPS = 511.0
```

GLONASS L1 C/A code length [chips].

Definition at line 79 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.158 GLONASS\_L1\_CA\_CODE\_PERIOD\_S**

```
constexpr double GLONASS_L1_CA_CODE_PERIOD_S = 0.001
```

GLONASS L1 C/A code period [seconds].

Definition at line 80 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.159 GLONASS\_L1\_CA\_CODE\_RATE\_CPS**

```
constexpr double GLONASS_L1_CA_CODE_RATE_CPS = 0.511e6
```

GLONASS L1 C/A code rate [chips/s].

Definition at line 78 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.160 GLONASS\_L1\_CA\_DFREQ\_HZ**

```
constexpr double GLONASS_L1_CA_DFREQ_HZ = DFRQ1_GLO
```

Freq Bias for GLONASS L1 [Hz].

Definition at line 77 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.161 GLONASS\_L1\_CA\_FREQ\_HZ**

```
constexpr double GLONASS_L1_CA_FREQ_HZ = FREQ1\_GLO
```

L1 [Hz].

Definition at line 76 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.162 GLONASS\_L2\_CA\_CHIP\_PERIOD\_S**

```
constexpr double GLONASS_L2_CA_CHIP_PERIOD_S = 1.9569e-06
```

GLONASS L1 C/A chip period [seconds].

Definition at line 73 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.163 GLONASS\_L2\_CA\_CODE\_LENGTH\_CHIPS**

```
constexpr double GLONASS_L2_CA_CODE_LENGTH_CHIPS = 511.0
```

GLONASS L1 C/A code length [chips].

Definition at line 71 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.164 GLONASS\_L2\_CA\_CODE\_PERIOD\_S**

```
constexpr double GLONASS_L2_CA_CODE_PERIOD_S = 0.001
```

GLONASS L1 C/A code period [seconds].

Definition at line 72 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.165 GLONASS\_L2\_CA\_CODE\_RATE\_CPS**

```
constexpr double GLONASS_L2_CA_CODE_RATE_CPS = 0.511e6
```

GLONASS L1 C/A code rate [chips/s].

Definition at line 70 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.166 GLONASS\_L2\_CA\_DFREQ\_HZ**

```
constexpr double GLONASS_L2_CA_DFREQ_HZ = DFRQ2_GLO
```

Freq Bias for GLONASS L1 [Hz].

Definition at line 69 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.167 GLONASS\_L2\_CA\_FREQ\_HZ**

```
constexpr double GLONASS_L2_CA_FREQ_HZ = FREQ2_GLO
```

L2 [Hz].

Definition at line 68 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.168 GLONASS\_LEAP\_SECONDS**

```
constexpr double GLONASS_LEAP_SECONDS[19][7]
```

**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 rtklibGLONASS SV's orbital slots PRN = (orbital\_slot - 1)

Definition at line 112 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.169 GLONASS\_MOON\_ECCENTRICITY

```
constexpr double GLONASS_MOON_ECCENTRICITY = 0.054900489
```

Eccentricity of lunar orbit.

Definition at line 58 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.170 GLONASS\_MOON\_GM

```
constexpr double GLONASS_MOON_GM = 4902.835
```

Lunar gravitational constant [ $\text{km}^3/\text{s}^2$ ].

Definition at line 56 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.171 GLONASS\_MOON\_INCLINATION

```
constexpr double GLONASS_MOON_INCLINATION = 0.000089803977407e3
```

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.

#### 9.49.4.172 GLONASS\_MOON\_OMEGA\_0

```
constexpr double GLONASS_MOON_OMEGA_0 = 0.004523601514852e3
```

(259 deg 10 min 59.79 sec) [rad]

Definition at line 54 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.173 GLONASS\_MOON\_OMEGA\_1

```
constexpr double GLONASS_MOON_OMEGA_1 = -0.033757146246552e3
```

(-1934 deg 08 min 31.23 sec) [rad]

Definition at line 55 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.174 GLONASS\_MOON\_Q0**

```
constexpr double GLONASS_MOON_Q0 = -0.001115184961435e3
```

(-63 deg 53 min 43.41 sec) [rad]

Definition at line 52 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.175 GLONASS\_MOON\_Q1**

```
constexpr double GLONASS_MOON_Q1 = 8.328691103668023e3
```

(477198 deg 50 min 56.79 sec) [rad]

Definition at line 53 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.176 GLONASS\_MOON\_SEMI\_MAJOR\_AXIS**

```
constexpr double GLONASS_MOON_SEMI_MAJOR_AXIS = 3.84385243e5
```

Semi-major axis of lunar orbit [km];.

Definition at line 57 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.177 GLONASS\_OMEGA\_EARTH\_DOT**

```
constexpr double GLONASS_OMEGA_EARTH_DOT = 7.292115e-5
```

Earth rotation rate, [rad/s] ICD L1, L2 GLONASS Edition 5.1 2008 pag. 55.

Definition at line 39 of file MATH\_CONSTANTS.h.

**9.49.4.178 GLONASS\_SEMI\_MAJOR\_AXIS**

```
constexpr double GLONASS_SEMI_MAJOR_AXIS = 6378136
```

Semi-major axis of Earth [m].

Definition at line 36 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.179 GLONASS\_SUN\_ECCENTRICITY

```
constexpr double GLONASS_SUN_ECCENTRICITY = 0.016719
```

Eccentricity of solar orbit.

Definition at line 66 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.180 GLONASS\_SUN\_GM

```
constexpr double GLONASS_SUN_GM = 0.1325263e12
```

Solar gravitational constant [ $\text{km}^3/\text{s}^2$ ].

Definition at line 64 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.181 GLONASS\_SUN\_OMEGA

```
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].

Definition at line 61 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.182 GLONASS\_SUN\_Q0

```
constexpr double GLONASS_SUN_Q0 = 0.006256583774423e3
```

(358 deg 28 min 33.04 sec) [rad]

Definition at line 62 of file GLONASS\_L1\_L2\_CA.h.

#### 9.49.4.183 GLONASS\_SUN\_Q1

```
constexpr double GLONASS_SUN_Q1 = 0e3
```

TODO Why is the value greater than 60?(129596579.10 sec) [rad].

Definition at line 63 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.184 GLONASS\_SUN\_SEMI\_MAJOR\_AXIS**

```
constexpr double GLONASS_SUN_SEMI_MAJOR_AXIS = 1.49598e8
```

Semi-major axis of solar orbit [km];.

Definition at line 65 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.185 GLONASS\_TAU\_0**

```
constexpr double GLONASS_TAU_0 = -0.005835151531174e3
```

(-334 deg 19 min 46.40 sec) [rad];

Definition at line 49 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.186 GLONASS\_TAU\_1**

```
constexpr double GLONASS_TAU_1 = 0.071018041257371e3
```

(4069 deg 02 min 02.52 sec) [rad];

Definition at line 50 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.187 GLONASS\_U0**

```
constexpr double GLONASS_U0 = 62636861.4
```

Normal potential at surface of common terrestrial ellipsoid [ $\text{m}^2/\text{s}^2$ ].

Definition at line 44 of file GLONASS\_L1\_L2\_CA.h.

**9.49.4.188 GNSS\_OMEGA\_EARTH\_DOT**

```
constexpr double GNSS_OMEGA_EARTH_DOT = 7.2921151467e-5
```

Default Earth rotation rate, [rad/s].

Definition at line 26 of file MATH\_CONSTANTS.h.

#### 9.49.4.189 GNSS\_PI

```
constexpr double GNSS_PI = 3.1415926535898
```

pi constant as defined for GNSS

Definition at line 47 of file MATH\_CONSTANTS.h.

#### 9.49.4.190 GPS\_CA\_TELEMETRY\_RATE\_BITS\_SECOND

```
constexpr int32_t GPS_CA_TELEMETRY_RATE_BITS_SECOND = 50
```

NAV message bit rate [bits/s].

Definition at line 63 of file GPS\_L1\_CA.h.

#### 9.49.4.191 GPS\_CA\_TELEMETRY\_RATE\_SYMBOLS\_SECOND

```
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].

Definition at line 65 of file GPS\_L1\_CA.h.

#### 9.49.4.192 GPS\_F

```
constexpr double GPS_F = -4.442807633e-10
```

Constant,  $[s/(m)^{(1/2)}]$ , IS-GPS-200K, pag. 92.

Definition at line 32 of file MATH\_CONSTANTS.h.

#### 9.49.4.193 GPS\_GM

```
constexpr double GPS_GM = 3.986005e14
```

Universal gravitational constant times the mass of the Earth,  $[m^3/s^2]$  IS-GPS-200K, pag 92.

Definition at line 31 of file MATH\_CONSTANTS.h.

**9.49.4.194 GPS\_L1\_CA\_BIT\_PERIOD\_MS**

```
constexpr uint32_t GPS_L1_CA_BIT_PERIOD_MS = 20U
```

GPS L1 C/A bit period [ms].

Definition at line 40 of file GPS\_L1\_CA.h.

**9.49.4.195 GPS\_L1\_CA\_CHIP\_PERIOD\_S**

```
constexpr double GPS_L1_CA_CHIP_PERIOD_S = 9.7752e-07
```

GPS L1 C/A chip period [seconds].

Definition at line 38 of file GPS\_L1\_CA.h.

**9.49.4.196 GPS\_L1\_CA\_CODE\_LENGTH\_CHIPS**

```
constexpr double GPS_L1_CA_CODE_LENGTH_CHIPS = 1023.0
```

GPS L1 C/A code length [chips].

Definition at line 36 of file GPS\_L1\_CA.h.

**9.49.4.197 GPS\_L1\_CA\_CODE\_PERIOD\_MS**

```
constexpr uint32_t GPS_L1_CA_CODE_PERIOD_MS = 1U
```

GPS L1 C/A code period [ms].

Definition at line 39 of file GPS\_L1\_CA.h.

**9.49.4.198 GPS\_L1\_CA\_CODE\_PERIOD\_S**

```
constexpr double GPS_L1_CA_CODE_PERIOD_S = 0.001
```

GPS L1 C/A code period [seconds].

Definition at line 37 of file GPS\_L1\_CA.h.

**9.49.4.199 GPS\_L1\_CA\_CODE\_RATE\_CPS**

```
constexpr double GPS_L1_CA_CODE_RATE_CPS = 1.023e6
```

GPS L1 C/A code rate [chips/s].

Definition at line 35 of file GPS\_L1\_CA.h.

**9.49.4.200 GPS\_L1\_CA\_OPT\_ACQ\_FS\_SPS**

```
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.

Definition at line 53 of file GPS\_L1\_CA.h.

**9.49.4.201 GPS\_L1\_FREQ\_HZ**

```
constexpr double GPS_L1_FREQ_HZ = FREQ1
```

L1 [Hz].

Definition at line 34 of file GPS\_L1\_CA.h.

**9.49.4.202 GPS\_L2\_CNAV\_DATA\_PAGE\_BITS**

```
constexpr int32_t GPS_L2_CNAV_DATA_PAGE_BITS = 300
```

GPS L2 CNAV page length, including preamble and CRC [bits].

Definition at line 42 of file GPS\_L2C.h.

**9.49.4.203 GPS\_L2\_FREQ\_HZ**

```
constexpr double GPS_L2_FREQ_HZ = FREQ2
```

L2 [Hz].

Definition at line 35 of file GPS\_L2C.h.

**9.49.4.204 GPS\_L2\_L\_CODE\_LENGTH\_CHIPS**

```
constexpr int32_t GPS_L2_L_CODE_LENGTH_CHIPS = 767250
```

GPS L2 L code length [chips].

Definition at line 41 of file GPS\_L2C.h.

**9.49.4.205 GPS\_L2\_L\_CODE\_RATE\_CPS**

```
constexpr double GPS_L2_L_CODE_RATE_CPS = 0.5115e6
```

GPS L2 L code rate [chips/s].

Definition at line 39 of file GPS\_L2C.h.

**9.49.4.206 GPS\_L2\_L\_PERIOD\_S**

```
constexpr double GPS_L2_L_PERIOD_S = 1.5
```

GPS L2 L code period [seconds].

Definition at line 36 of file GPS\_L2C.h.

**9.49.4.207 GPS\_L2\_M\_CODE\_LENGTH\_CHIPS**

```
constexpr int32_t GPS_L2_M_CODE_LENGTH_CHIPS = 10230
```

GPS L2 M code length [chips].

Definition at line 40 of file GPS\_L2C.h.

**9.49.4.208 GPS\_L2\_M\_CODE\_RATE\_CPS**

```
constexpr double GPS_L2_M_CODE_RATE_CPS = 0.5115e6
```

GPS L2 M code rate [chips/s].

Definition at line 37 of file GPS\_L2C.h.

#### 9.49.4.209 GPS\_L2\_M\_PERIOD\_S

```
constexpr double GPS_L2_M_PERIOD_S = 0.02
```

GPS L2 M code period [seconds].

Definition at line 38 of file GPS\_L2C.h.

#### 9.49.4.210 GPS\_L2C\_M\_INIT\_REG

```
constexpr int32_t GPS_L2C_M_INIT_REG[115]
```

**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 53 of file GPS\_L2C.h.

#### 9.49.4.211 GPS\_L2C\_OPT\_ACQ\_FS\_SPS

```
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.

Definition at line 51 of file GPS\_L2C.h.

**9.49.4.212 GPS\_L5\_CNAV\_DATA\_PAGE\_BITS**

```
constexpr int32_t GPS_L5_CNAV_DATA_PAGE_BITS = 300
```

GPS L5 CNAV page length, including preamble and CRC [bits].

Definition at line 162 of file GPS\_L5.h.

**9.49.4.213 GPS\_L5\_FREQ\_HZ**

```
constexpr double GPS_L5_FREQ_HZ = FREQ5
```

L5 [Hz].

Definition at line 32 of file GPS\_L5.h.

**9.49.4.214 GPS\_L5\_OPT\_ACQ\_FS\_SPS**

```
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.

Definition at line 46 of file GPS\_L5.h.

**9.49.4.215 GPS\_L5I\_CODE\_LENGTH\_CHIPS**

```
constexpr int32_t GPS_L5I_CODE_LENGTH_CHIPS = 10230
```

GPS L5I code length [chips].

Definition at line 39 of file GPS\_L5.h.

**9.49.4.216 GPS\_L5I\_CODE\_RATE\_CPS**

```
constexpr double GPS_L5I_CODE_RATE_CPS = 10.23e6
```

GPS L5I code rate [chips/s].

Definition at line 33 of file GPS\_L5.h.

**9.49.4.217 GPS\_L5I\_PERIOD\_MS**

```
constexpr int32_t GPS_L5I_PERIOD_MS = 1
```

GPS L5I code period [ms].

Definition at line 40 of file GPS\_L5.h.

**9.49.4.218 GPS\_L5I\_PERIOD\_S**

```
constexpr double GPS_L5I_PERIOD_S = 0.001
```

GPS L5I code period [seconds].

Definition at line 34 of file GPS\_L5.h.

**9.49.4.219 GPS\_L5I\_SYMBOL\_PERIOD\_MS**

```
constexpr int32_t GPS_L5I_SYMBOL_PERIOD_MS = 10
```

GPS L5I symbol period [ms].

Definition at line 41 of file GPS\_L5.h.

**9.49.4.220 GPS\_L5I\_SYMBOL\_PERIOD\_S**

```
constexpr double GPS_L5I_SYMBOL_PERIOD_S = 0.01
```

GPS L5I symbol period [seconds].

Definition at line 35 of file GPS\_L5.h.

**9.49.4.221 GPS\_L5Q\_CODE\_LENGTH\_CHIPS**

```
constexpr int32_t GPS_L5Q_CODE_LENGTH_CHIPS = 10230
```

GPS L5Q code length [chips].

Definition at line 38 of file GPS\_L5.h.

**9.49.4.222 GPS\_L5Q\_CODE\_RATE\_CPS**

```
constexpr double GPS_L5Q_CODE_RATE_CPS = 10.23e6
```

GPS L5Q code rate [chips/s].

Definition at line 36 of file GPS\_L5.h.

**9.49.4.223 GPS\_L5Q\_PERIOD\_S**

```
constexpr double GPS_L5Q_PERIOD_S = 0.001
```

GPS L5Q code period [seconds].

Definition at line 37 of file GPS\_L5.h.

**9.49.4.224 GPS\_SUBFRAME\_BITS**

```
constexpr int32_t GPS_SUBFRAME_BITS = 300
```

Number of bits per subframe in the NAV message [bits].

Definition at line 68 of file GPS\_L1\_CA.h.

**9.49.4.225 GPS\_SUBFRAME\_LENGTH**

```
constexpr int32_t GPS_SUBFRAME_LENGTH = 40
```

GPS\_WORD\_LENGTH x 10 = 40 bytes.

Definition at line 67 of file GPS\_L1\_CA.h.

**9.49.4.226 GPS\_SUBFRAME\_MS**

```
constexpr int32_t GPS_SUBFRAME_MS = 6000
```

Subframe duration [seconds].

Definition at line 70 of file GPS\_L1\_CA.h.

**9.49.4.227 GPS\_SUBFRAME\_SECONDS**

```
constexpr int32_t GPS_SUBFRAME_SECONDS = 6
```

Subframe duration [seconds].

Definition at line 69 of file GPS\_L1\_CA.h.

**9.49.4.228 GPS\_WORD\_BITS**

```
constexpr int32_t GPS_WORD_BITS = 30
```

Number of bits per word in the NAV message [bits].

Definition at line 71 of file GPS\_L1\_CA.h.

**9.49.4.229 GPS\_WORD\_LENGTH**

```
constexpr int32_t GPS_WORD_LENGTH = 4
```

CRC + GPS WORD (-2 -1 0 ... 29) Bits = 4 bytes.

Definition at line 66 of file GPS\_L1\_CA.h.

**9.49.4.230 HALF\_PI**

```
constexpr double HALF_PI = GNSS_PI / 2.0
```

pi/2

Definition at line 48 of file MATH\_CONSTANTS.h.

**9.49.4.231 MAX\_TOA\_DELAY\_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.

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.

**9.49.4.232 MAXCODE**

```
constexpr int32_t MAXCODE = 55
```

max number of obs code

Definition at line 86 of file gnss\_obs\_codes.h.

**9.49.4.233 PI\_TWO\_N19**

```
constexpr double PI_TWO_N19 = 5.992112452678286e-006
```

$\text{Pi} \cdot 2^{-19}$ .

Definition at line 114 of file MATH\_CONSTANTS.h.

**9.49.4.234 PI\_TWO\_N23**

```
constexpr double PI_TWO_N23 = 3.745070282923929e-007
```

$\text{Pi} \cdot 2^{-23}$ .

Definition at line 118 of file MATH\_CONSTANTS.h.

**9.49.4.235 PI\_TWO\_N31**

```
constexpr double PI_TWO_N31 = 1.462918079267160e-009
```

$\text{Pi} \cdot 2^{-31}$ .

Definition at line 116 of file MATH\_CONSTANTS.h.

**9.49.4.236 PI\_TWO\_N38**

```
constexpr double PI_TWO_N38 = 1.142904749427469e-011
```

$\text{Pi} \cdot 2^{-38}$ .

Definition at line 117 of file MATH\_CONSTANTS.h.

**9.49.4.237 PI\_TWO\_N43**

```
constexpr double PI_TWO_N43 = 3.571577341960839e-013
```

Pi\*2<sup>-43</sup>.

Definition at line 115 of file MATH\_CONSTANTS.h.

**9.49.4.238 R2D**

```
constexpr double R2D = 180.0 / GNSS_PI
```

rad to deg

Definition at line 121 of file MATH\_CONSTANTS.h.

**9.49.4.239 SC2RAD**

```
constexpr double SC2RAD = GNSS_PI
```

semi-circle to radian (IS-GPS)

Definition at line 122 of file MATH\_CONSTANTS.h.

**9.49.4.240 SPEED\_OF\_LIGHT\_M\_MS**

```
constexpr double SPEED_OF_LIGHT_M_MS = 299792.4580
```

Speed of light in vacuum [m/ms].

Definition at line 28 of file MATH\_CONSTANTS.h.

**9.49.4.241 SPEED\_OF\_LIGHT\_M\_S**

```
constexpr double SPEED_OF_LIGHT_M_S = 299792458.0
```

Speed of light in vacuum [m/s].

Definition at line 27 of file MATH\_CONSTANTS.h.

**9.49.4.242 TWO\_N10**

```
constexpr double TWO_N10 = 0.0009765625
```

 $2^{-10}$ 

Definition at line 76 of file MATH\_CONSTANTS.h.

**9.49.4.243 TWO\_N11**

```
constexpr double TWO_N11 = 4.882812500000000e-004
```

 $2^{-11}$ 

Definition at line 77 of file MATH\_CONSTANTS.h.

**9.49.4.244 TWO\_N14**

```
constexpr double TWO_N14 = 0.00006103515625
```

 $2^{-14}$ 

Definition at line 78 of file MATH\_CONSTANTS.h.

**9.49.4.245 TWO\_N15**

```
constexpr double TWO_N15 = 0.00003051757813
```

 $2^{-15}$ 

Definition at line 79 of file MATH\_CONSTANTS.h.

**9.49.4.246 TWO\_N16**

```
constexpr double TWO_N16 = 0.0000152587890625
```

 $2^{-16}$ 

Definition at line 80 of file MATH\_CONSTANTS.h.

**9.49.4.247 TWO\_N17**

```
constexpr double TWO_N17 = 7.629394531250000e-006
```

$2^{-17}$

Definition at line 81 of file MATH\_CONSTANTS.h.

**9.49.4.248 TWO\_N18**

```
constexpr double TWO_N18 = 3.814697265625000e-006
```

$2^{-18}$

Definition at line 82 of file MATH\_CONSTANTS.h.

**9.49.4.249 TWO\_N19**

```
constexpr double TWO_N19 = 1.907348632812500e-006
```

$2^{-19}$

Definition at line 83 of file MATH\_CONSTANTS.h.

**9.49.4.250 TWO\_N2**

```
constexpr double TWO_N2 = 0.25
```

$2^{-2}$

Definition at line 71 of file MATH\_CONSTANTS.h.

**9.49.4.251 TWO\_N20**

```
constexpr double TWO_N20 = 9.536743164062500e-007
```

$2^{-20}$

Definition at line 84 of file MATH\_CONSTANTS.h.

**9.49.4.252 TWO\_N21**

```
constexpr double TWO_N21 = 4.768371582031250e-007
```

 $2^{-21}$ 

Definition at line 85 of file MATH\_CONSTANTS.h.

**9.49.4.253 TWO\_N23**

```
constexpr double TWO_N23 = 1.192092895507810e-007
```

 $2^{-23}$ 

Definition at line 86 of file MATH\_CONSTANTS.h.

**9.49.4.254 TWO\_N24**

```
constexpr double TWO_N24 = 5.960464477539063e-008
```

 $2^{-24}$ 

Definition at line 87 of file MATH\_CONSTANTS.h.

**9.49.4.255 TWO\_N25**

```
constexpr double TWO_N25 = 2.980232238769531e-008
```

 $2^{-25}$ 

Definition at line 88 of file MATH\_CONSTANTS.h.

**9.49.4.256 TWO\_N27**

```
constexpr double TWO_N27 = 7.450580596923828e-009
```

 $2^{-27}$ 

Definition at line 89 of file MATH\_CONSTANTS.h.

**9.49.4.257 TWO\_N29**

```
constexpr double TWO_N29 = 1.862645149230957e-009
```

$2^{-29}$

Definition at line 90 of file MATH\_CONSTANTS.h.

**9.49.4.258 TWO\_N30**

```
constexpr double TWO_N30 = 9.313225746154785e-010
```

$2^{-30}$

Definition at line 91 of file MATH\_CONSTANTS.h.

**9.49.4.259 TWO\_N31**

```
constexpr double TWO_N31 = 4.656612873077393e-010
```

$2^{-31}$

Definition at line 92 of file MATH\_CONSTANTS.h.

**9.49.4.260 TWO\_N32**

```
constexpr double TWO_N32 = 2.328306436538696e-010
```

$2^{-32}$

Definition at line 93 of file MATH\_CONSTANTS.h.

**9.49.4.261 TWO\_N33**

```
constexpr double TWO_N33 = 1.164153218269348e-010
```

$2^{-33}$

Definition at line 94 of file MATH\_CONSTANTS.h.

**9.49.4.262 TWO\_N34**

```
constexpr double TWO_N34 = 5.82076609134674e-011
```

 $2^{-34}$ 

Definition at line 95 of file MATH\_CONSTANTS.h.

**9.49.4.263 TWO\_N35**

```
constexpr double TWO_N35 = 2.91038304567337e-011
```

 $2^{-35}$ 

Definition at line 96 of file MATH\_CONSTANTS.h.

**9.49.4.264 TWO\_N38**

```
constexpr double TWO_N38 = 3.637978807091713e-012
```

 $2^{-38}$ 

Definition at line 97 of file MATH\_CONSTANTS.h.

**9.49.4.265 TWO\_N39**

```
constexpr double TWO_N39 = 1.818989403545856e-012
```

 $2^{-39}$ 

Definition at line 98 of file MATH\_CONSTANTS.h.

**9.49.4.266 TWO\_N40**

```
constexpr double TWO_N40 = 9.094947017729280e-013
```

 $2^{-40}$ 

Definition at line 99 of file MATH\_CONSTANTS.h.

**9.49.4.267 TWO\_N43**

```
constexpr double TWO_N43 = 1.136868377216160e-013
```

$2^{-43}$

Definition at line 100 of file MATH\_CONSTANTS.h.

**9.49.4.268 TWO\_N44**

```
constexpr double TWO_N44 = 5.684341886080802e-14
```

$2^{-44}$

Definition at line 101 of file MATH\_CONSTANTS.h.

**9.49.4.269 TWO\_N46**

```
constexpr double TWO_N46 = 1.4210854715202e-014
```

$2^{-46}$

Definition at line 102 of file MATH\_CONSTANTS.h.

**9.49.4.270 TWO\_N48**

```
constexpr double TWO_N48 = 3.552713678800501e-15
```

$2^{-46}$

Definition at line 103 of file MATH\_CONSTANTS.h.

**9.49.4.271 TWO\_N5**

```
constexpr double TWO_N5 = 0.03125
```

$2^{-5}$

Definition at line 72 of file MATH\_CONSTANTS.h.

**9.49.4.272 TWO\_N50**

```
constexpr double TWO_N50 = 8.881784197001252e-016
```

 $2^{-50}$ 

Definition at line 105 of file MATH\_CONSTANTS.h.

**9.49.4.273 TWO\_N51**

```
constexpr double TWO_N51 = 4.44089209850063e-016
```

 $2^{-51}$ 

Definition at line 106 of file MATH\_CONSTANTS.h.

**9.49.4.274 TWO\_N55**

```
constexpr double TWO_N55 = 2.775557561562891e-017
```

 $2^{-55}$ 

Definition at line 107 of file MATH\_CONSTANTS.h.

**9.49.4.275 TWO\_N57**

```
constexpr double TWO_N57 = 6.938893903907228e-18
```

 $2^{-57}$ 

Definition at line 108 of file MATH\_CONSTANTS.h.

**9.49.4.276 TWO\_N59**

```
constexpr double TWO_N59 = 1.73472347597681e-018
```

 $2^{-59}$ 

Definition at line 109 of file MATH\_CONSTANTS.h.

**9.49.4.277 TWO\_N6**

```
constexpr double TWO_N6 = 0.015625
```

$2^{-6}$

Definition at line 73 of file MATH\_CONSTANTS.h.

**9.49.4.278 TWO\_N60**

```
constexpr double TWO_N60 = 8.673617379884036e-19
```

$2^{-60}$

Definition at line 110 of file MATH\_CONSTANTS.h.

**9.49.4.279 TWO\_N66**

```
constexpr double TWO_N66 = 1.3552527156068805425093160010874271392822265625e-20
```

$2^{-66}$

Definition at line 111 of file MATH\_CONSTANTS.h.

**9.49.4.280 TWO\_N68**

```
constexpr double TWO_N68 = 3.388131789017201e-21
```

$2^{-68}$

Definition at line 112 of file MATH\_CONSTANTS.h.

**9.49.4.281 TWO\_N8**

```
constexpr double TWO_N8 = 0.00390625
```

$2^{-8}$

Definition at line 74 of file MATH\_CONSTANTS.h.

**9.49.4.282 TWO\_N9**

```
constexpr double TWO_N9 = 0.001953125
```

 $2^{-9}$ 

Definition at line 75 of file MATH\_CONSTANTS.h.

**9.49.4.283 TWO\_P11**

```
constexpr double TWO_P11 = 2048.0
```

 $2^{11}$ 

Definition at line 61 of file MATH\_CONSTANTS.h.

**9.49.4.284 TWO\_P12**

```
constexpr double TWO_P12 = 4096.0
```

 $2^{12}$ 

Definition at line 62 of file MATH\_CONSTANTS.h.

**9.49.4.285 TWO\_P14**

```
constexpr double TWO_P14 = 16384.0
```

 $2^{14}$ 

Definition at line 63 of file MATH\_CONSTANTS.h.

**9.49.4.286 TWO\_P16**

```
constexpr double TWO_P16 = 65536.0
```

 $2^{16}$ 

Definition at line 64 of file MATH\_CONSTANTS.h.

**9.49.4.287 TWO\_P19**

```
constexpr double TWO_P19 = 524288.0
```

 $2^{19}$ 

Definition at line 65 of file MATH\_CONSTANTS.h.

**9.49.4.288 TWO\_P3**

```
constexpr double TWO_P3 = 8.0
```

 $2^3$ 

Definition at line 59 of file MATH\_CONSTANTS.h.

**9.49.4.289 TWO\_P31**

```
constexpr double TWO_P31 = 2147483648.0
```

 $2^{31}$ 

Definition at line 66 of file MATH\_CONSTANTS.h.

**9.49.4.290 TWO\_P32**

```
constexpr double TWO_P32 = 4294967296.0
```

 $2^{32}$ 

Definition at line 67 of file MATH\_CONSTANTS.h.

**9.49.4.291 TWO\_P4**

```
constexpr double TWO_P4 = 16.0
```

 $2^4$ 

Definition at line 60 of file MATH\_CONSTANTS.h.

**9.49.4.292 TWO\_P56**

```
constexpr double TWO_P56 = 7.205759403792794e+016
```

$2^{56}$

Definition at line 68 of file MATH\_CONSTANTS.h.

**9.49.4.293 TWO\_P57**

```
constexpr double TWO_P57 = 1.441151880758559e+017
```

$2^{57}$

Definition at line 69 of file MATH\_CONSTANTS.h.

**9.49.4.294 TWO\_PI**

```
constexpr double TWO_PI = 2.0 * GNSS_PI
```

$2 * \pi$

Definition at line 49 of file MATH\_CONSTANTS.h.



# Chapter 10

## Class Documentation

### 10.1 Acq\_Conf Class Reference

#### Public Member Functions

- void **SetFromConfiguration** (const [ConfigurationInterface](#) \*configuration, const std::string &role, double chip\_rate, double opt\_freq)

#### Public Attributes

- std::string **item\_type**
- std::string **dump\_filename**
- int64\_t **fs\_in**
- int64\_t **resampled\_fs**
- size\_t **it\_size**
- float **doppler\_step**
- float **samples\_per\_ms**
- float **doppler\_step2**
- float **pfa**
- float **pfa2**
- float **samples\_per\_code**
- float **resampler\_ratio**
- uint32\_t **sampled\_ms**
- uint32\_t **ms\_per\_code**
- uint32\_t **samples\_per\_chip**
- uint32\_t **chips\_per\_second**
- uint32\_t **max\_dwells**
- uint32\_t **num\_doppler\_bins\_step2**
- uint32\_t **resampler\_latency\_samples**
- uint32\_t **dump\_channel**
- int32\_t **doppler\_max**
- int32\_t **doppler\_min**
- bool **bit\_transition\_flag**
- bool **use\_CFAR\_algorithm\_flag**
- bool **dump**
- bool **blocking**
- bool **blocking\_on\_standby**
- bool **make\_2\_steps**
- bool **use\_automatic\_resampler**
- bool **enable\_monitor\_output**

### 10.1.1 Detailed Description

Definition at line 32 of file `acq_conf.h`.

The documentation for this class was generated from the following file:

- [acq\\_conf.h](#)

## 10.2 Acquisition\_Dump\_Reader Class Reference

### Public Member Functions

- **Acquisition\_Dump\_Reader** (const std::string &basename, unsigned int sat, unsigned int doppler\_max, unsigned int doppler\_step, unsigned int samples\_per\_code, int channel=0, int execution=1)
- **Acquisition\_Dump\_Reader** (const std::string &basename, int channel=0, int execution=1)
- **Acquisition\_Dump\_Reader** (const [Acquisition\\_Dump\\_Reader](#) &other) noexcept  
*Copy constructor.*
- **Acquisition\_Dump\_Reader** & **operator=** (const [Acquisition\\_Dump\\_Reader](#) &)  
*Copy assignment operator.*
- **Acquisition\_Dump\_Reader** ([Acquisition\\_Dump\\_Reader](#) &&other) noexcept  
*Move constructor.*
- **Acquisition\_Dump\_Reader** & **operator=** ([Acquisition\\_Dump\\_Reader](#) &&other) noexcept  
*Move assignment operator.*
- bool **read\_binary\_acq** ()

### Public Attributes

- std::vector< int > **doppler**
- std::vector< unsigned int > **samples**
- std::vector< std::vector< float > > **mag**
- float **acq\_doppler\_hz** {}
- float **acq\_delay\_samples** {}
- float **test\_statistic** {}
- float **input\_power** {}
- float **threshold** {}
- int **positive\_acq** {}
- unsigned int **PRN** {}
- unsigned int **num\_dwells** {}
- uint64\_t **sample\_counter** {}

### 10.2.1 Detailed Description

Definition at line 25 of file `acquisition_dump_reader.h`.

### 10.2.2 Constructor & Destructor Documentation

## 10.2.2.1 Acquisition\_Dump\_Reader() [1/2]

```
Acquisition_Dump_Reader::Acquisition_Dump_Reader (
    const Acquisition_Dump_Reader & other ) [noexcept]
```

Copy constructor.

## 10.2.2.2 Acquisition\_Dump\_Reader() [2/2]

```
Acquisition_Dump_Reader::Acquisition_Dump_Reader (
    Acquisition_Dump_Reader && other ) [noexcept]
```

Move constructor.

## 10.2.3 Member Function Documentation

## 10.2.3.1 operator=() [1/2]

```
Acquisition_Dump_Reader& Acquisition_Dump_Reader::operator= (
    const Acquisition_Dump_Reader & )
```

Copy assignment operator.

## 10.2.3.2 operator=() [2/2]

```
Acquisition_Dump_Reader& Acquisition_Dump_Reader::operator= (
    Acquisition_Dump_Reader && other ) [noexcept]
```

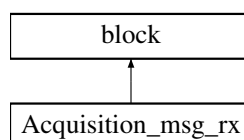
Move assignment operator.

The documentation for this class was generated from the following file:

- [acquisition\\_dump\\_reader.h](#)

## 10.3 Acquisition\_msg\_rx Class Reference

Inheritance diagram for Acquisition\_msg\_rx:



## Public Member Functions

- [~Acquisition\\_msg\\_rx\(\)](#)

*Default destructor.*

## Public Attributes

- int **rx\_message**
- gr::top\_block\_sptr **top\_block**

## Friends

- Acquisition\_msg\_rx\_sptr **Acquisition\_msg\_rx\_make()**

### 10.3.1 Detailed Description

Definition at line 36 of file acquisition\_msg\_rx.h.

### 10.3.2 Constructor & Destructor Documentation

#### 10.3.2.1 ~Acquisition\_msg\_rx()

```
Acquisition_msg_rx::~Acquisition_msg_rx ( )
```

Default destructor.

The documentation for this class was generated from the following file:

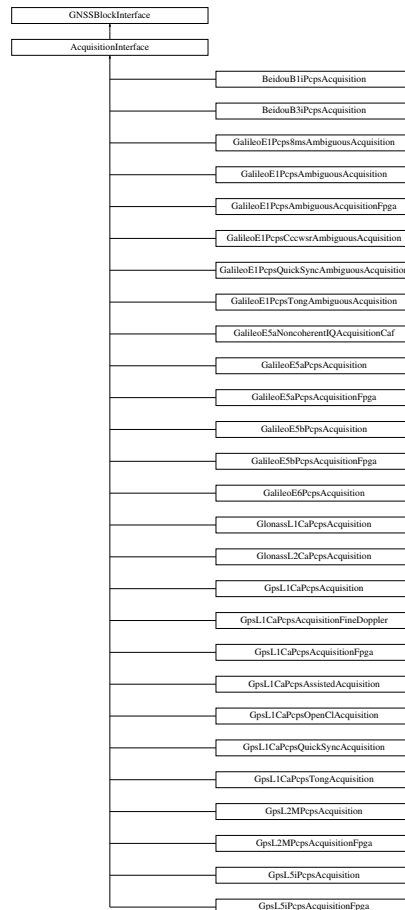
- [acquisition\\_msg\\_rx.h](#)

## 10.4 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\_threshold** (float threshold)=0
- virtual void **set\_doppler\_max** (unsigned int doppler\_max)=0
- virtual void **set\_doppler\_step** (unsigned int doppler\_step)=0
- virtual void **set\_doppler\_center** (int doppler\_center \_\_attribute\_\_((unused)))=0
- virtual void **init** ()=0
- virtual void **set\_local\_code** ()=0
- virtual void **set\_state** (int state)=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

### 10.4.1 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.

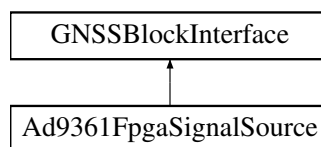
Definition at line 50 of file `acquisition_interface.h`.

The documentation for this class was generated from the following file:

- [acquisition\\_interface.h](#)

## 10.5 Ad9361FpgaSignalSource Class Reference

Inheritance diagram for Ad9361FpgaSignalSource:



### Public Member Functions

- **Ad9361FpgaSignalSource** (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.*
- std::string **role** () override
- std::string **implementation** () override  
*Returns "Ad9361\_Fpga\_Signal\_Source".*
- 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

### 10.5.1 Detailed Description

Definition at line 41 of file `ad9361_fpga_signal_source.h`.

### 10.5.2 Member Function Documentation

### 10.5.2.1 implementation()

```
std::string Ad9361FpgaSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "Ad9361\_Fpga\_Signal\_Source".

Implements [GNSSBlockInterface](#).

Definition at line 60 of file `ad9361_fpga_signal_source.h`.

### 10.5.2.2 start()

```
void Ad9361FpgaSignalSource::start ( ) [override], [virtual]
```

Start the flow of samples if needed.

Reimplemented from [GNSSBlockInterface](#).

The documentation for this class was generated from the following file:

- [ad9361\\_fpga\\_signal\\_source.h](#)

## 10.6 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** {}

### 10.6.1 Detailed Description

Interface of an Assisted GNSS REFERENCE LOCATION storage.

Definition at line 33 of file `agnss_ref_location.h`.

## 10.6.2 Constructor & Destructor Documentation

### 10.6.2.1 Agnss\_Ref\_Location()

```
Agnss_Ref_Location::Agnss_Ref_Location ( ) [default]
```

Default constructor

## 10.6.3 Member Function Documentation

### 10.6.3.1 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 51 of file `agnss_ref_location.h`.

The documentation for this class was generated from the following file:

- [agnss\\_ref\\_location.h](#)

## 10.7 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 **d\_TOW** {}
- double **d\_Week** {}
- double **d\_tv\_sec** {}
- double **d\_tv\_usec** {}
- bool **valid** {}

### 10.7.1 Detailed Description

Interface of an Assisted GNSS REFERENCE TIME storage.

Definition at line 33 of file agnss\_ref\_time.h.

### 10.7.2 Constructor & Destructor Documentation

#### 10.7.2.1 Agnss\_Ref\_Time()

```
Agnss_Ref_Time::Agnss_Ref_Time ( ) [default]
```

Default constructor

### 10.7.3 Member Function Documentation

#### 10.7.3.1 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 52 of file agnss\_ref\_time.h.

The documentation for this class was generated from the following file:

- [agnss\\_ref\\_time.h](#)

## 10.8 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**

### 10.8.1 Detailed Description

Definition at line 425 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.9 ambc\_t Struct Reference

### Public Attributes

- [gtime\\_t](#) **epoch** [4]
- int **n** [4]
- double **LC** [4]
- double **LCv** [4]
- int **fixcnt**
- char **flags** [MAXSAT]

### 10.9.1 Detailed Description

Definition at line 1056 of file rtklib.h.

The documentation for this struct was generated from the following file:

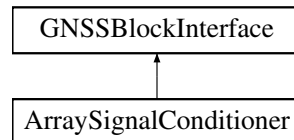
- [rtklib.h](#)

## 10.10 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_adapter`, 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** ()

### 10.10.1 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`.

### 10.10.2 Constructor & Destructor Documentation

### 10.10.2.1 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.

### 10.10.2.2 ~ArraySignalConditioner()

```
ArraySignalConditioner::~~ArraySignalConditioner ( ) [default]
```

Destructor.

## 10.10.3 Member Function Documentation

### 10.10.3.1 implementation()

```
std::string ArraySignalConditioner::implementation ( ) [inline], [override], [virtual]
```

Returns "Array\_Signal\_Conditioner".

Implements [GNSSBlockInterface](#).

Definition at line 60 of file array\_signal\_conditioner.h.

The documentation for this class was generated from the following file:

- [array\\_signal\\_conditioner.h](#)

## 10.11 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

### 10.11.1 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 {} and covariance {{C}}, which has a conjugate prior given by a normal-inverse-Wishart distribution with paramemters {{0}, {0}, {0}, and {}.

[1] TODO: Ref1

Definition at line 60 of file [bayesian\\_estimation.h](#).

The documentation for this class was generated from the following file:

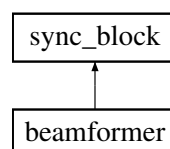
- [bayesian\\_estimation.h](#)

## 10.12 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** ()

### 10.12.1 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:

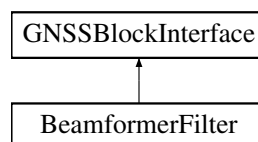
- [beamformer.h](#)

## 10.13 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

### 10.13.1 Detailed Description

Interface of an adapter of a digital beamformer block to a [GNSSBlockInterface](#).

Definition at line 38 of file beamformer\_filter.h.

### 10.13.2 Member Function Documentation

## 10.13.2.1 implementation()

```
std::string BeamformerFilter::implementation ( ) [inline], [override], [virtual]
```

returns "Beamformer\_Filte"

Implements [GNSSBlockInterface](#).

Definition at line 53 of file beamformer\_filter.h.

The documentation for this class was generated from the following file:

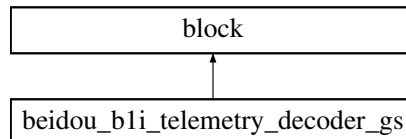
- [beamformer\\_filter.h](#)

## 10.14 beidou\_b1i\_telemetry\_decoder\_gs Class Reference

This class implements a block that decodes the BeiDou DNAV data.

```
#include <beidou_b1i_telemetry_decoder_gs.h>
```

Inheritance diagram for beidou\_b1i\_telemetry\_decoder\_gs:



## Public Member Functions

- [~beidou\\_b1i\\_telemetry\\_decoder\\_gs](#) ()  
*Class destructor.*
- void [set\\_satellite](#) (const [Gnss\\_Satellite](#) &satellite)  
*Set satellite PRN.*
- void [set\\_channel](#) (int channel)  
*Set receiver's channel.*
- void [reset](#) ()
- int [general\\_work](#) (int noutput\_items, gr\_vector\_int &ninput\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)  
*This is where all signal processing takes place.*

## Friends

- beidou\_b1i\_telemetry\_decoder\_gs\_sptr [beidou\\_b1i\\_make\\_telemetry\\_decoder\\_gs](#) (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)

### 10.14.1 Detailed Description

This class implements a block that decodes the BeiDou DNAV data.

#### Note

Code added as part of GSoC 2018 program

Definition at line 54 of file beidou\_b1i\_telemetry\_decoder\_gs.h.

### 10.14.2 Constructor & Destructor Documentation

#### 10.14.2.1 ~beidou\_b1i\_telemetry\_decoder\_gs()

```
beidou_b1i_telemetry_decoder_gs::~~beidou_b1i_telemetry_decoder_gs ( )
```

Class destructor.

### 10.14.3 Member Function Documentation

#### 10.14.3.1 general\_work()

```
int beidou_b1i_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 )
```

This is where all signal processing takes place.

#### 10.14.3.2 set\_channel()

```
void beidou_b1i_telemetry_decoder_gs::set_channel (
    int channel )
```

Set receiver's channel.

## 10.14.3.3 set\_satellite()

```
void beidou_b3i_telemetry_decoder_gs::set_satellite (
    const Gnss_Satellite & satellite )
```

Set satellite PRN.

The documentation for this class was generated from the following file:

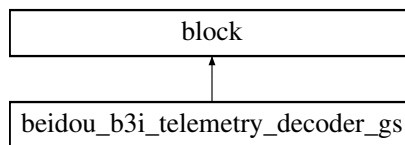
- [beidou\\_b3i\\_telemetry\\_decoder\\_gs.h](#)

## 10.15 beidou\_b3i\_telemetry\_decoder\_gs Class Reference

This class implements a block that decodes the BeiDou DNAV data.

```
#include <beidou_b3i_telemetry_decoder_gs.h>
```

Inheritance diagram for beidou\_b3i\_telemetry\_decoder\_gs:



## Public Member Functions

- [~beidou\\_b3i\\_telemetry\\_decoder\\_gs \(\)](#)  
*Class destructor.*
- void [set\\_satellite](#) (const [Gnss\\_Satellite](#) &satellite)  
*Set satellite PRN.*
- void [set\\_channel](#) (int channel)  
*Set receiver's channel.*
- void [reset](#) ()
- int [general\\_work](#) (int noutput\_items, gr\_vector\_int &ninput\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)  
*This is where all signal processing takes place.*

## Friends

- beidou\_b3i\_telemetry\_decoder\_gs\_sptr [beidou\\_b3i\\_make\\_telemetry\\_decoder\\_gs](#) (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)

## 10.15.1 Detailed Description

This class implements a block that decodes the BeiDou DNAV data.

Definition at line 51 of file [beidou\\_b3i\\_telemetry\\_decoder\\_gs.h](#).

## 10.15.2 Constructor & Destructor Documentation

### 10.15.2.1 `~beidou_b3i_telemetry_decoder_gs()`

```
beidou_b3i_telemetry_decoder_gs::~~beidou_b3i_telemetry_decoder_gs ( )
```

Class destructor.

## 10.15.3 Member Function Documentation

### 10.15.3.1 `general_work()`

```
int beidou_b3i_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 )
```

This is where all signal processing takes place.

### 10.15.3.2 `set_channel()`

```
void beidou_b3i_telemetry_decoder_gs::set_channel (
    int channel )
```

Set receiver's channel.

### 10.15.3.3 `set_satellite()`

```
void beidou_b3i_telemetry_decoder_gs::set_satellite (
    const Gnss\_Satellite & satellite )
```

Set satellite PRN.

The documentation for this class was generated from the following file:

- [beidou\\_b3i\\_telemetry\\_decoder\\_gs.h](#)

## 10.16 Beidou\_Dnav\_Almanac Class Reference

This class is a storage for the BeiDou D1 almanac.

```
#include <beidou_dnav_almanac.h>
```

### Public Member Functions

- [Beidou\\_Dnav\\_Almanac](#) ()=default
- `template<class Archive >`  
void **serialize** (Archive &ar, const unsigned int version)

### Public Attributes

- unsigned int [i\\_satellite\\_PRN](#) {}  
*SV PRN NUMBER.*
- double [d\\_Delta\\_i](#) {}
- double [d\\_Toa](#) {}  
*Almanac data reference time of week [s].*
- double [d\\_M\\_0](#) {}  
*Mean Anomaly at Reference Time [semi-circles].*
- double [d\\_e\\_eccentricity](#) {}  
*Eccentricity [dimensionless].*
- double [d\\_sqrt\\_A](#) {}  
*Square Root of the Semi-Major Axis [sqrt(m)].*
- double [d\\_OMEGA0](#) {}  
*Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles].*
- double [d\\_OMEGA](#) {}  
*Argument of Perigee [semi-circles].*
- double [d\\_OMEGA\\_DOT](#) {}  
*Rate of Right Ascension [semi-circles/s].*
- int [i\\_SV\\_health](#) {}  
*SV Health.*
- double [d\\_A\\_f0](#) {}  
*Coefficient 0 of code phase offset model [s].*
- double [d\\_A\\_f1](#) {}  
*Coefficient 1 of code phase offset model [s/s].*

### 10.16.1 Detailed Description

This class is a storage for the BeiDou D1 almanac.

Definition at line 32 of file `beidou_dnav_almanac.h`.

### 10.16.2 Constructor & Destructor Documentation

### 10.16.2.1 Beidou\_Dnav\_Almanac()

```
Beidou_Dnav_Almanac::Beidou_Dnav_Almanac ( ) [default]
```

Default constructor

## 10.16.3 Member Data Documentation

### 10.16.3.1 d\_A\_f0

```
double Beidou_Dnav_Almanac::d_A_f0 {}
```

Coefficient 0 of code phase offset model [s].

Definition at line 50 of file beidou\_dnav\_almanac.h.

### 10.16.3.2 d\_A\_f1

```
double Beidou_Dnav_Almanac::d_A_f1 {}
```

Coefficient 1 of code phase offset model [s/s].

Definition at line 51 of file beidou\_dnav\_almanac.h.

### 10.16.3.3 d\_e\_eccentricity

```
double Beidou_Dnav_Almanac::d_e_eccentricity {}
```

Eccentricity [dimensionless].

Definition at line 44 of file beidou\_dnav\_almanac.h.

### 10.16.3.4 d\_M\_0

```
double Beidou_Dnav_Almanac::d_M_0 {}
```

Mean Anomaly at Reference Time [semi-circles].

Definition at line 43 of file beidou\_dnav\_almanac.h.

### 10.16.3.5 d\_OMEGA

```
double Beidou_Dnav_Almanac::d_OMEGA {}
```

Argument of Perigee [semi-cicles].

Definition at line 47 of file beidou\_dnav\_almanac.h.

### 10.16.3.6 d\_OMEGA0

```
double Beidou_Dnav_Almanac::d_OMEGA0 {}
```

Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles].

Definition at line 46 of file beidou\_dnav\_almanac.h.

### 10.16.3.7 d\_OMEGA\_DOT

```
double Beidou_Dnav_Almanac::d_OMEGA_DOT {}
```

Rate of Right Ascension [semi-circles/s].

Definition at line 48 of file beidou\_dnav\_almanac.h.

### 10.16.3.8 d\_sqrt\_A

```
double Beidou_Dnav_Almanac::d_sqrt_A {}
```

Square Root of the Semi-Major Axis [sqrt(m)].

Definition at line 45 of file beidou\_dnav\_almanac.h.

### 10.16.3.9 d\_Toa

```
double Beidou_Dnav_Almanac::d_Toa {}
```

Almanac data reference time of week [s].

Definition at line 42 of file beidou\_dnav\_almanac.h.

### 10.16.3.10 i\_satellite\_PRN

```
unsigned int Beidou_Dnav_Almanac::i_satellite_PRN {}
```

SV PRN NUMBER.

Definition at line 40 of file beidou\_dnav\_almanac.h.

### 10.16.3.11 i\_SV\_health

```
int Beidou_Dnav_Almanac::i_SV_health {}
```

SV Health.

Definition at line 49 of file beidou\_dnav\_almanac.h.

The documentation for this class was generated from the following file:

- [beidou\\_dnav\\_almanac.h](#)

## 10.17 Beidou\_Dnav\_Ephemeris Class Reference

This class is a storage and orbital model functions for the GPS 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>
```

### Public Member Functions

- [Beidou\\_Dnav\\_Ephemeris](#) ()
- double [satellitePosition](#) (double transmitTime)  
*Compute the ECEF SV coordinates and ECEF velocity Implementation of Table 20-IV (IS-GPS-200K) and compute the clock bias term including relativistic effect (return value)*
- double [sv\\_clock\\_drift](#) (double transmitTime)  
*Sets (d\_satClkDrift) and returns the clock drift in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200K, 20.3.3.3.3.1)*
- double [sv\\_clock\\_relativistic\\_term](#) (double transmitTime)  
*Sets (d\_dtr) and returns the clock relativistic correction term in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200K, 20.3.3.3.3.1)*
- 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

- unsigned int [i\\_satellite\\_PRN](#) {}  
*SV PRN NUMBER.*
- double [d\\_TOW](#) {}  
*Time of BEIDOU Week of the ephemeris set (taken from subframes TOW) [s].*
- double [d\\_Crs](#) {}  
*Amplitude of the Sine Harmonic Correction Term to the Orbit Radius [m].*
- double [d\\_Delta\\_n](#) {}  
*Mean Motion Difference From Computed Value [semi-circles/s].*
- double [d\\_M\\_0](#) {}  
*Mean Anomaly at Reference Time [semi-circles].*
- double [d\\_Cuc](#) {}  
*Amplitude of the Cosine Harmonic Correction Term to the Argument of Latitude [rad].*
- double [d\\_eccentricity](#) {}  
*Eccentricity [dimensionless].*
- double [d\\_Cus](#) {}  
*Amplitude of the Sine Harmonic Correction Term to the Argument of Latitude [rad].*
- double [d\\_sqrt\\_A](#) {}  
*Square Root of the Semi-Major Axis [sqrt(m)].*
- double [d\\_Toe](#) {}  
*Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200K) [s].*
- double [d\\_Toc](#) {}  
*clock data reference time (Ref. 20.3.3.3.3.1 IS-GPS-200K) [s]*
- double [d\\_Cic](#) {}  
*Amplitude of the Cosine Harmonic Correction Term to the Angle of Inclination [rad].*
- double [d\\_OMEGA0](#) {}  
*Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles].*
- double [d\\_Cis](#) {}  
*Amplitude of the Sine Harmonic Correction Term to the Angle of Inclination [rad].*
- double [d\\_i\\_0](#) {}  
*Inclination Angle at Reference Time [semi-circles].*
- double [d\\_Crc](#) {}  
*Amplitude of the Cosine Harmonic Correction Term to the Orbit Radius [m].*
- double [d\\_OMEGA](#) {}  
*Argument of Perigee [semi-circles].*
- double [d\\_OMEGA\\_DOT](#) {}  
*Rate of Right Ascension [semi-circles/s].*
- double [d\\_IDOT](#) {}  
*Rate of Inclination Angle [semi-circles/s].*
- int [i\\_BEIDOU\\_week](#) {}  
*BEIDOU week number, aka WN [week].*
- int [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-200K)*
- int [i\\_SV\\_health](#) {}
- double [d\\_TGD1](#) {}  
*Estimated Group Delay Differential on B1I [s].*
- double [d\\_TGD2](#) {}  
*Estimated Group Delay Differential on B2I [s].*
- double [d\\_AODC](#) {}

- Age of Data, Clock.*

  - double [d\\_AODE](#) {}
- Age of Data, Ephemeris.*

  - int [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].*
- int [i\\_sig\\_type](#) {}

*BDS: data source (0:unknown,1:B1I,2:B1Q,3:B2I,4:B2Q,5:B3I,6:B3Q) \*/.*
- int [i\\_nav\\_type](#) {}

*BDS: nav type (0:unknown,1:IGSO/MEO,2:GEO) \*/.*
- 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.*
- double [d\\_spare1](#) {}
- double [d\\_spare2](#) {}
- double [d\\_A\\_f0](#) {}

*Coefficient 0 of code phase offset model [s].*
- double [d\\_A\\_f1](#) {}

*Coefficient 1 of code phase offset model [s/s].*
- double [d\\_A\\_f2](#) {}

*Coefficient 2 of code phase offset model [s/s<sup>2</sup>].*
- bool [b\\_integrity\\_status\\_flag](#) {}

*If true, enhanced level of integrity assurance.*
- 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.*
- bool [b\\_antispoofing\\_flag](#) {}

*If true, the AntiSpoofing mode is ON in that SV.*
- double [d\\_satClkDrift](#) {}

*GPS clock error.*
- double [d\\_dtr](#) {}

*relativistic clock correction term*
- double [d\\_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 [d\\_satpos\\_Y](#) {}

*Earth-fixed coordinate y of the satellite [m]. Completes a right-handed, Earth-Centered, Earth-Fixed orthogonal coordinate system.*
- double [d\\_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 [d\\_satvel\\_X](#) {}

*Earth-fixed velocity coordinate x of the satellite [m].*
- double [d\\_satvel\\_Y](#) {}

*Earth-fixed velocity coordinate y of the satellite [m].*
- double [d\\_satvel\\_Z](#) {}

*Earth-fixed velocity coordinate z of the satellite [m].*
- std::map< int, std::string > [satelliteBlock](#)

*Map that stores to which block the PRN belongs.*

### 10.17.1 Detailed Description

This class is a storage and orbital model functions for the GPS 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 38 of file beidou\_dnav\_ephemeris.h.

### 10.17.2 Constructor & Destructor Documentation

#### 10.17.2.1 Beidou\_Dnav\_Ephemeris()

```
Beidou_Dnav_Ephemeris::Beidou_Dnav_Ephemeris ( )
```

Default constructor

### 10.17.3 Member Function Documentation

#### 10.17.3.1 satellitePosition()

```
double Beidou_Dnav_Ephemeris::satellitePosition (
    double transmitTime )
```

Compute the ECEF SV coordinates and ECEF velocity Implementation of Table 20-IV (IS-GPS-200K) and compute the clock bias term including relativistic effect (return value)

### 10.17.3.2 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.

- < Time of GPS Week of the ephemeris set (taken from subframes TOW) [s]
- < Amplitude of the Sine Harmonic Correction Term to the Orbit Radius [m]
- < Mean Motion Difference From Computed Value [semi-circles/s]
- < Mean Anomaly at Reference Time [semi-circles]
- < Amplitude of the Cosine Harmonic Correction Term to the Argument of Latitude [rad]
- < Eccentricity [dimensionless]
- < Amplitude of the Sine Harmonic Correction Term to the Argument of Latitude [rad]
- < Square Root of the Semi-Major Axis [sqrt(m)]
- < Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200K) [s]
- < clock data reference time (Ref. 20.3.3.3.1 IS-GPS-200K) [s]
- < Amplitude of the Cosine Harmonic Correction Term to the Angle of Inclination [rad]
- < Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles]
- < Amplitude of the Sine Harmonic Correction Term to the Angle of Inclination [rad]
- < Inclination Angle at Reference Time [semi-circles]
- < Amplitude of the Cosine Harmonic Correction Term to the Orbit Radius [m]
- < Argument of Perigee [semi-circles]
- < Rate of Right Ascension [semi-circles/s]
- < Rate of Inclination Angle [semi-circles/s]
- < GPS week number, aka WN [week]
- < User Range Accuracy (URA) index of the SV (reference paragraph 6.2.1) for the standard positioning service user (Ref 20.3.3.1.3 IS-GPS-200K)
- < Issue of Data, Clock
- < Estimated Group Delay Differential: L1-L2 correction term only for the benefit of "L1 P(Y)" or "L2 P(Y)" s users [s]
- < Estimated Group Delay Differential: L1-L2 correction term only for the benefit of "L1 P(Y)" or "L2 P(Y)" s users [s]
- < 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]

< 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.

< Coefficient 0 of code phase offset model [s]

< Coefficient 1 of code phase offset model [s/s]

< Coefficient 2 of code phase offset model [s/s<sup>2</sup>]

< If true, indicates that the SV URA may be worse than indicated in d\_SV\_accuracy, use that SV at our own risk.

< If true, the AntiSpoofing mode is ON in that SV

Definition at line 140 of file beidou\_dnav\_ephemeris.h.

References b\_alert\_flag, b\_antispoofing\_flag, b\_fit\_interval\_flag, b\_integrity\_status\_flag, d\_A\_f0, d\_A\_f1, d\_A\_f2, d\_AODC, d\_AODE, d\_Cic, d\_Cis, d\_Crc, d\_Crs, d\_Cuc, d\_Cus, d\_Delta\_n, d\_eccentricity, d\_i\_0, d\_IDOT, d\_M\_0, d\_OMEGA, d\_OMEGA0, d\_OMEGA\_DOT, d\_sqrt\_A, d\_TGD1, d\_TGD2, d\_Toe, d\_TOW, i\_AODO, i\_BEIDOU\_↵ week, i\_satellite\_PRN, and i\_SV\_accuracy.

### 10.17.3.3 sv\_clock\_drift()

```
double Beidou_Dnav_Ephemeris::sv_clock_drift (
    double transmitTime )
```

Sets (*d\_satClkDrift*) and returns the clock drift in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200K, 20.3.3.3.3.1)

### 10.17.3.4 sv\_clock\_relativistic\_term()

```
double Beidou_Dnav_Ephemeris::sv_clock_relativistic_term (
    double transmitTime )
```

Sets (*d\_dtr*) and returns the clock relativistic correction term in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200K, 20.3.3.3.3.1)

## 10.17.4 Member Data Documentation

### 10.17.4.1 b\_alert\_flag

```
bool Beidou_Dnav_Ephemeris::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.

Definition at line 115 of file beidou\_dnav\_ephemeris.h.

Referenced by serialize().

#### 10.17.4.2 b\_antispoofing\_flag

```
bool Beidou_Dnav_Ephemeris::b_antispoofing_flag {}
```

If true, the AntiSpoofing mode is ON in that SV.

Definition at line 116 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.3 b\_fit\_interval\_flag

```
bool Beidou_Dnav_Ephemeris::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.

Definition at line 96 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.4 b\_integrity\_status\_flag

```
bool Beidou_Dnav_Ephemeris::b_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 114 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.5 d\_A\_f0

```
double Beidou_Dnav_Ephemeris::d_A_f0 {}
```

Coefficient 0 of code phase offset model [s].

Definition at line 100 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.6 d\_A\_f1

```
double Beidou_Dnav_Ephemeris::d_A_f1 {}
```

Coefficient 1 of code phase offset model [s/s].

Definition at line 101 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.7 d\_A\_f2

```
double Beidou_Dnav_Ephemeris::d_A_f2 {}
```

Coefficient 2 of code phase offset model [s/s<sup>2</sup>].

Definition at line 102 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.8 d\_AODC

```
double Beidou_Dnav_Ephemeris::d_AODC {}
```

Age of Data, Clock.

Definition at line 89 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.9 d\_AODE

```
double Beidou_Dnav_Ephemeris::d_AODE {}
```

Age of Data, Ephemeris.

Definition at line 90 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.10 d\_Cic

```
double Beidou_Dnav_Ephemeris::d_Cic {}
```

Amplitude of the Cosine Harmonic Correction Term to the Angle of Inclination [rad].

Definition at line 76 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.11 d\_Cis

```
double Beidou_Dnav_Ephemeris::d_Cis {}
```

Amplitude of the Sine Harmonic Correction Term to the Angle of Inclination [rad].

Definition at line 78 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.12 d\_Crc

```
double Beidou_Dnav_Ephemeris::d_Crc {}
```

Amplitude of the Cosine Harmonic Correction Term to the Orbit Radius [m].

Definition at line 80 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.13 d\_Crs

```
double Beidou_Dnav_Ephemeris::d_Crs {}
```

Amplitude of the Sine Harmonic Correction Term to the Orbit Radius [m].

Definition at line 67 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.14 d\_Cuc

```
double Beidou_Dnav_Ephemeris::d_Cuc {}
```

Amplitude of the Cosine Harmonic Correction Term to the Argument of Latitude [rad].

Definition at line 70 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.15 d\_Cus

```
double Beidou_Dnav_Ephemeris::d_Cus {}
```

Amplitude of the Sine Harmonic Correction Term to the Argument of Latitude [rad].

Definition at line 72 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.16 d\_Delta\_n

```
double Beidou_Dnav_Ephemeris::d_Delta_n {}
```

Mean Motion Difference From Computed Value [semi-circles/s].

Definition at line 68 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.17 d\_dtr

```
double Beidou_Dnav_Ephemeris::d_dtr {}
```

relativistic clock correction term

Definition at line 120 of file beidou\_dnav\_ephemeris.h.

#### 10.17.4.18 d\_eccentricity

```
double Beidou_Dnav_Ephemeris::d_eccentricity {}
```

Eccentricity [dimensionless].

Definition at line 71 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.19 d\_i\_0

```
double Beidou_Dnav_Ephemeris::d_i_0 {}
```

Inclination Angle at Reference Time [semi-circles].

Definition at line 79 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.20 d\_IDOT

```
double Beidou_Dnav_Ephemeris::d_IDOT {}
```

Rate of Inclination Angle [semi-circles/s].

Definition at line 83 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.21 d\_M\_0

```
double Beidou_Dnav_Ephemeris::d_M_0 {}
```

Mean Anomaly at Reference Time [semi-circles].

Definition at line 69 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.22 d\_OMEGA

```
double Beidou_Dnav_Ephemeris::d_OMEGA {}
```

Argument of Perigee [semi-circles].

Definition at line 81 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.23 d\_OMEGA0

```
double Beidou_Dnav_Ephemeris::d_OMEGA0 {}
```

Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles].

Definition at line 77 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.24 d\_OMEGA\_DOT

```
double Beidou_Dnav_Ephemeris::d_OMEGA_DOT {}
```

Rate of Right Ascension [semi-circles/s].

Definition at line 82 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.25 d\_satClkDrift

```
double Beidou_Dnav_Ephemeris::d_satClkDrift {}
```

GPS clock error.

Definition at line 119 of file beidou\_dnav\_ephemeris.h.

#### 10.17.4.26 d\_satpos\_X

```
double Beidou_Dnav_Ephemeris::d_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 123 of file beidou\_dnav\_ephemeris.h.

#### 10.17.4.27 d\_satpos\_Y

```
double Beidou_Dnav_Ephemeris::d_satpos_Y {}
```

Earth-fixed coordinate y of the satellite [m]. Completes a right-handed, Earth-Centered, Earth-Fixed orthogonal coordinate system.

Definition at line 124 of file beidou\_dnav\_ephemeris.h.

#### 10.17.4.28 d\_satpos\_Z

```
double Beidou_Dnav_Ephemeris::d_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 125 of file beidou\_dnav\_ephemeris.h.

#### 10.17.4.29 d\_satvel\_X

```
double Beidou_Dnav_Ephemeris::d_satvel_X {}
```

Earth-fixed velocity coordinate x of the satellite [m].

Definition at line 128 of file beidou\_dnav\_ephemeris.h.

#### 10.17.4.30 d\_satvel\_Y

```
double Beidou_Dnav_Ephemeris::d_satvel_Y {}
```

Earth-fixed velocity coordinate y of the satellite [m].

Definition at line 129 of file beidou\_dnav\_ephemeris.h.

**10.17.4.31 d\_satvel\_Z**

```
double Beidou_Dnav_Ephemeris::d_satvel_Z {}
```

Earth-fixed velocity coordinate z of the satellite [m].

Definition at line 130 of file beidou\_dnav\_ephemeris.h.

**10.17.4.32 d\_sqrt\_A**

```
double Beidou_Dnav_Ephemeris::d_sqrt_A {}
```

Square Root of the Semi-Major Axis [sqrt(m)].

Definition at line 73 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

**10.17.4.33 d\_TGD1**

```
double Beidou_Dnav_Ephemeris::d_TGD1 {}
```

Estimated Group Delay Differential on B1I [s].

Definition at line 87 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

**10.17.4.34 d\_TGD2**

```
double Beidou_Dnav_Ephemeris::d_TGD2 {}
```

Estimated Group Delay Differential on B2I [s].

Definition at line 88 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

**10.17.4.35 d\_Toc**

```
double Beidou_Dnav_Ephemeris::d_Toc {}
```

clock data reference time (Ref. 20.3.3.3.1 IS-GPS-200K) [s]

Definition at line 75 of file beidou\_dnav\_ephemeris.h.

#### 10.17.4.36 d\_Toe

```
double Beidou_Dnav_Ephemeris::d_Toe {}
```

Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200K) [s].

Definition at line 74 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.37 d\_TOW

```
double Beidou_Dnav_Ephemeris::d_TOW {}
```

Time of BEIDOU Week of the ephemeris set (taken from subframes TOW) [s].

Definition at line 66 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.38 i\_AODO

```
int Beidou_Dnav_Ephemeris::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].

Definition at line 91 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.39 i\_BEIDOU\_week

```
int Beidou_Dnav_Ephemeris::i_BEIDOU_week {}
```

BEIDOU week number, aka WN [week].

Definition at line 84 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.40 i\_nav\_type

```
int Beidou_Dnav_Ephemeris::i_nav_type {}
```

BDS: nav type (0:unknown,1:IGSO/MEO,2:GEO) \*/.

Definition at line 94 of file beidou\_dnav\_ephemeris.h.

#### 10.17.4.41 i\_satellite\_PRN

```
unsigned int Beidou_Dnav_Ephemeris::i_satellite_PRN {}
```

SV PRN NUMBER.

Definition at line 65 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.42 i\_sig\_type

```
int Beidou_Dnav_Ephemeris::i_sig_type {}
```

BDS: data source (0:unknown,1:B1I,2:B1Q,3:B2I,4:B2Q,5:B3I,6:B3Q) \*/.

Definition at line 93 of file beidou\_dnav\_ephemeris.h.

#### 10.17.4.43 i\_SV\_accuracy

```
int Beidou_Dnav_Ephemeris::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-200K)

Definition at line 85 of file beidou\_dnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.17.4.44 satelliteBlock

```
std::map<int, std::string> Beidou_Dnav_Ephemeris::satelliteBlock
```

Map that stores to which block the PRN belongs.

Definition at line 132 of file beidou\_dnav\_ephemeris.h.

The documentation for this class was generated from the following file:

- [beidou\\_dnav\\_ephemeris.h](#)

## 10.18 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>
```

### Public Member Functions

- [Beidou\\_Dnav\\_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 [d\\_alpha0](#) {}  
*Coefficient 0 of a cubic equation representing the amplitude of the vertical delay [s].*
- double [d\\_alpha1](#) {}  
*Coefficient 1 of a cubic equation representing the amplitude of the vertical delay [s/semi-circle].*
- double [d\\_alpha2](#) {}  
*Coefficient 2 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)<sup>2</sup>].*
- double [d\\_alpha3](#) {}  
*Coefficient 3 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)<sup>3</sup>].*
- double [d\\_beta0](#) {}  
*Coefficient 0 of a cubic equation representing the period of the model [s].*
- double [d\\_beta1](#) {}  
*Coefficient 1 of a cubic equation representing the period of the model [s/semi-circle].*
- double [d\\_beta2](#) {}  
*Coefficient 2 of a cubic equation representing the period of the model [s(semi-circle)<sup>2</sup>].*
- double [d\\_beta3](#) {}  
*Coefficient 3 of a cubic equation representing the period of the model [s(semi-circle)<sup>3</sup>].*
- bool [valid](#) {}  
*Valid flag.*

#### 10.18.1 Detailed Description

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

Definition at line 33 of file beidou\_dnav\_iono.h.

#### 10.18.2 Constructor & Destructor Documentation

### 10.18.2.1 Beidou\_Dnav\_Iono()

```
Beidou_Dnav_Iono::Beidou_Dnav_Iono ( ) [default]
```

Default constructor.

## 10.18.3 Member Function Documentation

### 10.18.3.1 serialize()

```
template<class Archive >
void Beidou_Dnav_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 55 of file beidou\_dnav\_iono.h.

References `d_alpha0`, `d_alpha1`, `d_alpha2`, `d_alpha3`, `d_beta0`, `d_beta1`, `d_beta2`, and `d_beta3`.

## 10.18.4 Member Data Documentation

### 10.18.4.1 d\_alpha0

```
double Beidou_Dnav_Iono::d_alpha0 {}
```

Coefficient 0 of a cubic equation representing the amplitude of the vertical delay [s].

Definition at line 39 of file beidou\_dnav\_iono.h.

Referenced by `serialize()`.

### 10.18.4.2 d\_alpha1

```
double Beidou_Dnav_Iono::d_alpha1 {}
```

Coefficient 1 of a cubic equation representing the amplitude of the vertical delay [s/semi-circle].

Definition at line 40 of file beidou\_dnav\_iono.h.

Referenced by `serialize()`.

#### 10.18.4.3 d\_alpha2

```
double Beidou_Dnav_Iono::d_alpha2 {}
```

Coefficient 2 of a cubic equation representing the amplitude of the vertical delay  $[s(\text{semi-circle})^2]$ .

Definition at line 41 of file beidou\_dnav\_iono.h.

Referenced by `serialize()`.

#### 10.18.4.4 d\_alpha3

```
double Beidou_Dnav_Iono::d_alpha3 {}
```

Coefficient 3 of a cubic equation representing the amplitude of the vertical delay  $[s(\text{semi-circle})^3]$ .

Definition at line 42 of file beidou\_dnav\_iono.h.

Referenced by `serialize()`.

#### 10.18.4.5 d\_beta0

```
double Beidou_Dnav_Iono::d_beta0 {}
```

Coefficient 0 of a cubic equation representing the period of the model [s].

Definition at line 43 of file beidou\_dnav\_iono.h.

Referenced by `serialize()`.

#### 10.18.4.6 d\_beta1

```
double Beidou_Dnav_Iono::d_beta1 {}
```

Coefficient 1 of a cubic equation representing the period of the model  $[s/\text{semi-circle}]$ .

Definition at line 44 of file beidou\_dnav\_iono.h.

Referenced by `serialize()`.

#### 10.18.4.7 d\_beta2

```
double Beidou_Dnav_Iono::d_beta2 {}
```

Coefficient 2 of a cubic equation representing the period of the model  $[s(\text{semi-circle})^2]$ .

Definition at line 45 of file beidou\_dnav\_iono.h.

Referenced by `serialize()`.

#### 10.18.4.8 d\_beta3

```
double Beidou_Dnav_Iono::d_beta3 {}
```

Coefficient 3 of a cubic equation representing the period of the model  $[s(\text{semi-circle})^3]$ .

Definition at line 46 of file beidou\_dnav\_iono.h.

Referenced by `serialize()`.

#### 10.18.4.9 valid

```
bool Beidou_Dnav_Iono::valid {}
```

Valid flag.

Definition at line 48 of file beidou\_dnav\_iono.h.

The documentation for this class was generated from the following file:

- [beidou\\_dnav\\_iono.h](#)

## 10.19 Beidou\_Dnav\_Navigation\_Message Class Reference

This class decodes a BeiDou D1 NAV Data message.

```
#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.*
- [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.*
- void [satellitePosition](#) (double transmitTime)  
*Computes the position of the satellite.*
- double [sv\\_clock\\_correction](#) (double transmitTime)  
*Sets (d\_satClkCorr) according to the User Algorithm for SV Clock Correction and returns the corrected clock.*
- double [utc\\_time](#) (const 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.*
- 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

### 10.19.1 Detailed Description

This class decodes a BeiDou D1 NAV Data message.

Definition at line 46 of file `beidou_dnav_navigation_message.h`.

### 10.19.2 Constructor & Destructor Documentation

#### 10.19.2.1 `Beidou_Dnav_Navigation_Message()`

```
Beidou_Dnav_Navigation_Message::Beidou_Dnav_Navigation_Message ( )
```

Default constructor

### 10.19.3 Member Function Documentation

#### 10.19.3.1 d1\_subframe\_decoder()

```
int32_t Beidou_Dnav_Navigation_Message::d1_subframe_decoder (
    std::string const & subframe )
```

Decodes the BDS D1 NAV message.

#### 10.19.3.2 d2\_subframe\_decoder()

```
int32_t Beidou_Dnav_Navigation_Message::d2_subframe_decoder (
    std::string const & subframe )
```

Decodes the BDS D2 NAV message.

#### 10.19.3.3 get\_ephemeris()

```
Beidou_Dnav_Ephemeris Beidou_Dnav_Navigation_Message::get_ephemeris ( ) const
```

Obtain a BDS SV Ephemeris class filled with current SV data.

#### 10.19.3.4 get\_iono()

```
Beidou_Dnav_Iono Beidou_Dnav_Navigation_Message::get_iono ( )
```

Obtain a BDS ionospheric correction parameters class filled with current SV data.

#### 10.19.3.5 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.

#### 10.19.3.6 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.

#### 10.19.3.7 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.

#### 10.19.3.8 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.

#### 10.19.3.9 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.

#### 10.19.3.10 satellitePosition()

```
void Beidou_Dnav_Navigation_Message::satellitePosition (
    double transmitTime )
```

Computes the position of the satellite.

#### 10.19.3.11 set\_satellite\_PRN()

```
void Beidou_Dnav_Navigation_Message::set_satellite_PRN (
    uint32_t prn ) [inline]
```

Sets satellite PRN number.

Definition at line 121 of file beidou\_dnav\_navigation\_message.h.

10.19.3.12 `sv_clock_correction()`

```
double Beidou_Dnav_Navigation_Message::sv_clock_correction (
    double transmitTime )
```

Sets (*d\_satClkCorr*) according to the User Algorithm for SV Clock Correction and returns the corrected clock.

10.19.3.13 `utc_time()`

```
double Beidou_Dnav_Navigation_Message::utc_time (
    const 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](#)

## 10.20 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 [d\\_A0\\_UTC](#) {}  
*BDT clock bias relative to UTC [s].*
- double [d\\_A1\\_UTC](#) {}  
*BDT clock rate relative to UTC [s/s].*
- int [i\\_DeltaT\\_LS](#) {}  
*Delta time due to leap seconds before the new leap second effective.*
- int [i\\_WN\\_LSF](#) {}  
*Week number of the new leap second.*
- int [i\\_DN](#) {}  
*Day number of week of the new leap second.*
- double [d\\_DeltaT\\_LSF](#) {}  
*Delta time due to leap seconds after the new leap second effective [s].*
- double [d\\_A0\\_GPS](#) {}  
*BDT clock bias relative to GPS time [s].*
- double [d\\_A1\\_GPS](#) {}  
*BDT clock rate relative to GPS time [s/s].*
- double [d\\_A0\\_GAL](#) {}  
*BDT clock bias relative to GAL time [s].*
- double [d\\_A1\\_GAL](#) {}  
*BDT clock rate relative to GAL time [s/s].*
- double [d\\_A0\\_GLO](#) {}  
*BDT clock bias relative to GLO time [s].*
- double [d\\_A1\\_GLO](#) {}  
*BDT clock rate relative to GLO time [s/s].*
- bool **valid** {}

### 10.20.1 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.

### 10.20.2 Member Data Documentation

#### 10.20.2.1 d\_A0\_GAL

```
double Beidou_Dnav_Utc_Model::d_A0_GAL {}
```

BDT clock bias relative to GAL time [s].

Definition at line 53 of file beidou\_dnav\_utc\_model.h.

#### 10.20.2.2 d\_A0\_GLO

```
double Beidou_Dnav_Utc_Model::d_A0_GLO {}
```

BDT clock bias relative to GLO time [s].

Definition at line 57 of file beidou\_dnav\_utc\_model.h.

#### 10.20.2.3 d\_A0\_GPS

```
double Beidou_Dnav_Utc_Model::d_A0_GPS {}
```

BDT clock bias relative to GPS time [s].

Definition at line 49 of file beidou\_dnav\_utc\_model.h.

#### 10.20.2.4 d\_A0\_UTC

```
double Beidou_Dnav_Utc_Model::d_A0_UTC {}
```

BDT clock bias relative to UTC [s].

Definition at line 41 of file beidou\_dnav\_utc\_model.h.

#### 10.20.2.5 d\_A1\_GAL

```
double Beidou_Dnav_Utc_Model::d_A1_GAL {}
```

BDT clock rate relative to GAL time [s/s].

Definition at line 54 of file beidou\_dnav\_utc\_model.h.

#### 10.20.2.6 d\_A1\_GLO

```
double Beidou_Dnav_Utc_Model::d_A1_GLO {}
```

BDT clock rate relative to GLO time [s/s].

Definition at line 58 of file beidou\_dnav\_utc\_model.h.

#### 10.20.2.7 d\_A1\_GPS

```
double Beidou_Dnav_Utc_Model::d_A1_GPS {}
```

BDT clock rate relative to GPS time [s/s].

Definition at line 50 of file beidou\_dnav\_utc\_model.h.

#### 10.20.2.8 d\_A1\_UTC

```
double Beidou_Dnav_Utc_Model::d_A1_UTC {}
```

BDT clock rate relative to UTC [s/s].

Definition at line 42 of file beidou\_dnav\_utc\_model.h.

#### 10.20.2.9 d\_DeltaT\_LSF

```
double Beidou_Dnav_Utc_Model::d_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.

### 10.20.2.10 i\_DeltaT\_LS

```
int Beidou_Dnav_Utc_Model::i_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.

### 10.20.2.11 i\_DN

```
int Beidou_Dnav_Utc_Model::i_DN {}
```

Day number of week of the new leap second.

Definition at line 45 of file beidou\_dnav\_utc\_model.h.

### 10.20.2.12 i\_WN\_LSF

```
int Beidou_Dnav_Utc_Model::i_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:

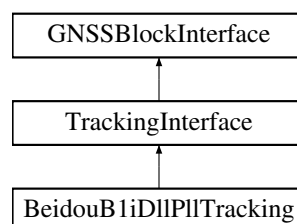
- [beidou\\_dnav\\_utc\\_model.h](#)

## 10.21 BeidouB1iDIPIITracking Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#include <beidou_b1i_dll_pll_tracking.h>
```

Inheritance diagram for BeidouB1iDIPIITracking:



## Public Member Functions

- **BeidouB1iDlIPllTracking** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- 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\_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.*

### 10.21.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 41 of file beidou\_b1i\_dll\_pll\_tracking.h.

### 10.21.2 Member Function Documentation

#### 10.21.2.1 set\_channel()

```
void BeidouB1iDlIPllTracking::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.21.2.2 set\_gnss\_synchro()

```
void BeidouB1iDlIPllTracking::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](#).

### 10.21.2.3 stop\_tracking()

```
void BeidouB1iDllPllTracking::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

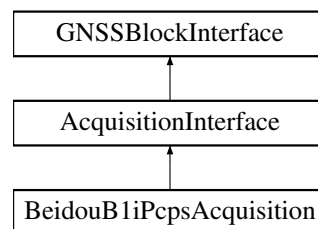
- [beidou\\_b1i\\_dll\\_pll\\_tracking.h](#)

## 10.22 BeidouB1iPcpsAcquisition Class Reference

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

```
#include <beidou_b1i_pcps_acquisition.h>
```

Inheritance diagram for BeidouB1iPcpsAcquisition:



### Public Member Functions

- **BeidouB1iPcpsAcquisition** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- std::string **role** () override
- std::string **implementation** () override  
*Returns "BEIDOU\_B1I\_PCPS\_Acquisition".*
- 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\_threshold** (float threshold) override  
*Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (uint32\_t doppler\_max) override

- *Set maximum Doppler off grid search.*
- void [set\\_doppler\\_step](#) (uint32\_t doppler\_step) override
- *Set Doppler steps for the grid search.*
- void [init](#) () override
- *Initializes acquisition algorithm.*
- void [set\\_local\\_code](#) () override
- *Sets local code for GPS L1/CA PCPS acquisition algorithm.*
- signed int [mag](#) () override
- *Returns the maximum peak of grid search.*
- void [reset](#) () override
- *Restart acquisition algorithm.*
- void [set\\_state](#) (int state) override
- *If state = 1, it forces the block to start acquiring from the first sample.*
- 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.*

### 10.22.1 Detailed Description

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

Definition at line 46 of file `beidou_b1i_pcps_acquisition.h`.

### 10.22.2 Member Function Documentation

#### 10.22.2.1 implementation()

```
std::string BeidouB1iPcpsAcquisition::implementation ( ) [inline], [override], [virtual]
```

Returns "BEIDOU\_B1I\_PCPS\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 63 of file `beidou_b1i_pcps_acquisition.h`.

#### 10.22.2.2 init()

```
void BeidouB1iPcpsAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.22.2.3 mag()

```
signed int BeidouBliPcpsAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

### 10.22.2.4 reset()

```
void BeidouBliPcpsAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.22.2.5 set\_channel()

```
void BeidouBliPcpsAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 88 of file beidou\_b1i\_pcps\_acquisition.h.

### 10.22.2.6 set\_channel\_fsm()

```
void BeidouBliPcpsAcquisition::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 97 of file beidou\_b1i\_pcps\_acquisition.h.

### 10.22.2.7 set\_doppler\_max()

```
void BeidouBliPcpsAcquisition::set_doppler_max (
    uint32_t doppler_max ) [override]
```

Set maximum Doppler off grid search.

### 10.22.2.8 set\_doppler\_step()

```
void BeidouBliPcpsAcquisition::set_doppler_step (
    uint32_t doppler_step ) [override]
```

Set Doppler steps for the grid search.

### 10.22.2.9 set\_gnss\_synchro()

```
void BeidouBliPcpsAcquisition::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](#).

### 10.22.2.10 set\_local\_code()

```
void BeidouBliPcpsAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local code for GPS L1/CA PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.22.2.11 set\_resampler\_latency()

```
void BeidouBliPcpsAcquisition::set_resampler_latency (
    uint32_t latency_samples ) [override], [virtual]
```

Sets the resampler latency to account it in the acquisition code delay estimation.

Implements [AcquisitionInterface](#).

### 10.22.2.12 set\_state()

```
void BeidouBliPcpsAcquisition::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

### 10.22.2.13 set\_threshold()

```
void BeidouBliPcpsAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

### 10.22.2.14 stop\_acquisition()

```
void BeidouBliPcpsAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

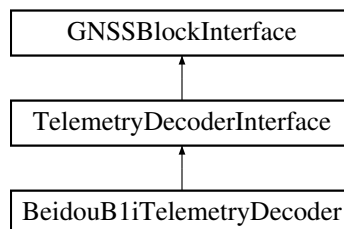
- [beidou\\_b1i\\_pcps\\_acquisition.h](#)

## 10.23 BeidouB1iTelemetryDecoder Class Reference

This class implements a NAV data decoder for BEIDOU B1I.

```
#include <beidou_b1i_telemetry_decoder.h>
```

Inheritance diagram for BeidouB1iTelemetryDecoder:



### Public Member Functions

- **BeidouB1iTelemetryDecoder** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- 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
- *Returns "BEIDOU\_B1I\_Telemetry\_Decoder".*
- void **set\_channel** (int channel) override
- void **reset** () override
- size\_t **item\_size** () override

### 10.23.1 Detailed Description

This class implements a NAV data decoder for BEIDOU B1I.

Definition at line 43 of file `beidou_b1i_telemetry_decoder.h`.

### 10.23.2 Member Function Documentation

#### 10.23.2.1 implementation()

```
std::string BeidouB1iTelemetryDecoder::implementation ( ) [inline], [override], [virtual]
```

Returns "BEIDOU\_B1I\_Telemetry\_Decoder".

Implements [GNSSBlockInterface](#).

Definition at line 67 of file `beidou_b1i_telemetry_decoder.h`.

The documentation for this class was generated from the following file:

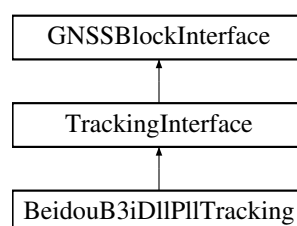
- [beidou\\_b1i\\_telemetry\\_decoder.h](#)

## 10.24 BeidouB3iDIIPITracking Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#include <beidou_b3i_dll_pll_tracking.h>
```

Inheritance diagram for BeidouB3iDIIPITracking:



## Public Member Functions

- **BeidouB3iDllPllTracking** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- 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\_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.*

### 10.24.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 41 of file beidou\_b3i\_dll\_pll\_tracking.h.

### 10.24.2 Member Function Documentation

#### 10.24.2.1 set\_channel()

```
void BeidouB3iDllPllTracking::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.24.2.2 set\_gnss\_synchro()

```
void BeidouB3iDllPllTracking::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](#).

## 10.24.2.3 stop\_tracking()

```
void BeidouB3iDllPllTracking::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

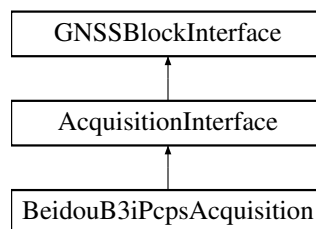
- [beidou\\_b3i\\_dll\\_pll\\_tracking.h](#)

## 10.25 BeidouB3iPcpsAcquisition Class Reference

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for BeiDou B3I signals.

```
#include <beidou_b3i_pcps_acquisition.h>
```

Inheritance diagram for BeidouB3iPcpsAcquisition:



### Public Member Functions

- **BeidouB3iPcpsAcquisition** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- std::string **role** () override
- std::string **implementation** () override
 

*Returns "BEIDOU\_B1I\_PCPS\_Acquisition".*
- 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\_threshold** (float threshold) override
 

*Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override

- *Set maximum Doppler off grid search.*
- void [set\\_doppler\\_step](#) (unsigned int doppler\_step) override
- *Set Doppler steps for the grid search.*
- void [init](#) () override
- *Initializes acquisition algorithm.*
- void [set\\_local\\_code](#) () override
- *Sets local code for GPS L1/CA PCPS acquisition algorithm.*
- signed int [mag](#) () override
- *Returns the maximum peak of grid search.*
- void [reset](#) () override
- *Restart acquisition algorithm.*
- void [set\\_state](#) (int state) override
- *If state = 1, it forces the block to start acquiring from the first sample.*
- 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.*

### 10.25.1 Detailed Description

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for BeiDou B3I signals.

Definition at line 45 of file `beidou_b3i_pcps_acquisition.h`.

### 10.25.2 Member Function Documentation

#### 10.25.2.1 `implementation()`

```
std::string BeidouB3iPcpsAcquisition::implementation ( ) [inline], [override], [virtual]
```

Returns "BEIDOU\_B1I\_PCPS\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 62 of file `beidou_b3i_pcps_acquisition.h`.

#### 10.25.2.2 `init()`

```
void BeidouB3iPcpsAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.25.2.3 mag()

```
signed int BeidouB3iPcpsAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

### 10.25.2.4 reset()

```
void BeidouB3iPcpsAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.25.2.5 set\_channel()

```
void BeidouB3iPcpsAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 87 of file beidou\_b3i\_pcps\_acquisition.h.

### 10.25.2.6 set\_channel\_fsm()

```
void BeidouB3iPcpsAcquisition::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 96 of file beidou\_b3i\_pcps\_acquisition.h.

#### 10.25.2.7 set\_doppler\_max()

```
void BeidouB3iPcpsAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.25.2.8 set\_doppler\_step()

```
void BeidouB3iPcpsAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.25.2.9 set\_gnss\_synchro()

```
void BeidouB3iPcpsAcquisition::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](#).

#### 10.25.2.10 set\_local\_code()

```
void BeidouB3iPcpsAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local code for GPS L1/CA PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.25.2.11 set\_resampler\_latency()

```
void BeidouB3iPcpsAcquisition::set_resampler_latency (
    uint32_t latency_samples ) [override], [virtual]
```

Sets the resampler latency to account it in the acquisition code delay estimation.

Implements [AcquisitionInterface](#).

**10.25.2.12 set\_state()**

```
void BeidouB3iPcpsAcquisition::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

**10.25.2.13 set\_threshold()**

```
void BeidouB3iPcpsAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

**10.25.2.14 stop\_acquisition()**

```
void BeidouB3iPcpsAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

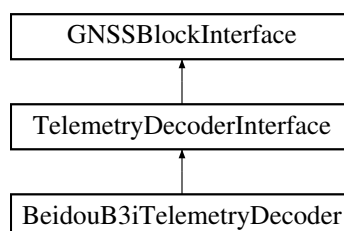
- [beidou\\_b3i\\_pcps\\_acquisition.h](#)

**10.26 BeidouB3iTelemetryDecoder Class Reference**

This class implements a NAV data decoder for BEIDOU B1I.

```
#include <beidou_b3i_telemetry_decoder.h>
```

Inheritance diagram for BeidouB3iTelemetryDecoder:



## Public Member Functions

- **BeidouB3iTelemetryDecoder** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- 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
- *Returns "BEIDOU\_B3I\_Telemetry\_Decoder".*
- void **set\_channel** (int channel) override
- void **reset** () override
- size\_t **item\_size** () override

### 10.26.1 Detailed Description

This class implements a NAV data decoder for BEIDOU B1I.

Definition at line 42 of file beidou\_b3i\_telemetry\_decoder.h.

### 10.26.2 Member Function Documentation

#### 10.26.2.1 implementation()

```
std::string BeidouB3iTelemetryDecoder::implementation ( ) [inline], [override], [virtual]
```

Returns "BEIDOU\_B3I\_Telemetry\_Decoder".

Implements [GNSSBlockInterface](#).

Definition at line 62 of file beidou\_b3i\_telemetry\_decoder.h.

The documentation for this class was generated from the following file:

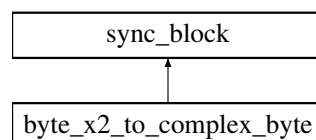
- [beidou\\_b3i\\_telemetry\\_decoder.h](#)

## 10.27 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 ()`

### 10.27.1 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:

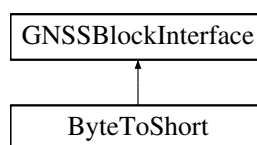
- [byte\\_x2\\_to\\_complex\\_byte.h](#)

## 10.28 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

### 10.28.1 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`.

### 10.28.2 Member Function Documentation

#### 10.28.2.1 `implementation()`

```
std::string ByteToShort::implementation ( ) [inline], [override], [virtual]
```

Returns "Byte\_To\_Short".

Implements [GNSSBlockInterface](#).

Definition at line 55 of file `byte_to_short.h`.

The documentation for this class was generated from the following file:

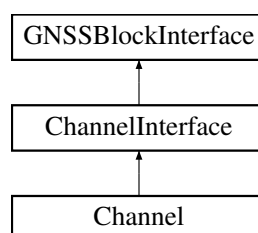
- [byte\\_to\\_short.h](#)

## 10.29 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](#) () const 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) override
- std::shared\_ptr< [AcquisitionInterface](#) > [acquisition](#) () const
- std::shared\_ptr< [TrackingInterface](#) > [tracking](#) () const
- std::shared\_ptr< [TelemetryDecoderInterface](#) > [telemetry](#) () const

### 10.29.1 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.

### 10.29.2 Constructor & Destructor Documentation

### 10.29.2.1 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.

### 10.29.2.2 ~Channel()

```
Channel::~~Channel ( ) [default]
```

Destructor.

## 10.29.3 Member Function Documentation

### 10.29.3.1 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](#).

### 10.29.3.2 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](#).

### 10.29.3.3 `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](#).

### 10.29.3.4 `get_right_block()`

```
gr::basic_block_sptr Channel::get_right_block ( ) [override], [virtual]
```

Gets the GNU Radio channel block output pointer.

Implements [ChannelInterface](#).

### 10.29.3.5 `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](#).

### 10.29.3.6 `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](#).

### 10.29.3.7 `implementation()`

```
std::string Channel::implementation ( ) [inline], [override], [virtual]
```

Returns "Channel".

Implements [GNSSBlockInterface](#).

Definition at line 85 of file channel.h.

#### 10.29.3.8 set\_signal()

```
void Channel::set_signal (
    const Gnss_Signal & gnss_signal_ ) [override], [virtual]
```

Sets the channel GNSS signal.

Implements [ChannelInterface](#).

#### 10.29.3.9 start\_acquisition()

```
void Channel::start_acquisition ( ) [override], [virtual]
```

Start the State Machine.

Implements [ChannelInterface](#).

#### 10.29.3.10 stop\_channel()

```
void Channel::stop_channel ( ) [override], [virtual]
```

Stop the State Machine.

Implements [ChannelInterface](#).

The documentation for this class was generated from the following file:

- [channel.h](#)

## 10.30 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)

### 10.30.1 Detailed Description

Definition at line 34 of file channel\_event.h.

The documentation for this class was generated from the following file:

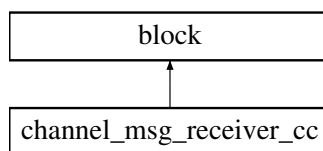
- [channel\\_event.h](#)

## 10.31 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)

### 10.31.1 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.

### 10.31.2 Constructor & Destructor Documentation

### 10.31.2.1 `~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:

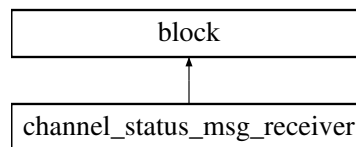
- [channel\\_msg\\_receiver\\_cc.h](#)

## 10.32 `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*
- `Monitor_Pvt get_current_status_pvt ()`  
*return the current receiver PVT*

### Friends

- `channel_status_msg_receiver_sptr channel_status_msg_receiver_make ()`

### 10.32.1 Detailed Description

GNU Radio block that receives asynchronous channel messages from tlm blocks.

Definition at line 44 of file `channel_status_msg_receiver.h`.

### 10.32.2 Constructor & Destructor Documentation

## 10.32.2.1 ~channel\_status\_msg\_receiver()

```
channel_status_msg_receiver::~~channel_status_msg_receiver ( ) [default]
```

Default destructor.

## 10.32.3 Member Function Documentation

## 10.32.3.1 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

## 10.32.3.2 get\_current\_status\_pvt()

```
Monitor_Pvt channel_status_msg_receiver::get_current_status_pvt ( )
```

return the current receiver PVT

The documentation for this class was generated from the following file:

- [channel\\_status\\_msg\\_receiver.h](#)

## 10.33 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** ()

### 10.33.1 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:

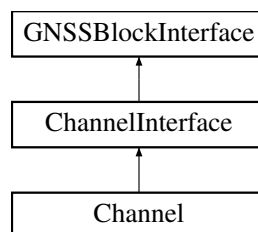
- [channel\\_fsm.h](#)

## 10.34 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** () const =0
- virtual void **start\_acquisition** ()=0
- virtual void **assist\_acquisition\_doppler** (double Carrier\_Doppler\_hz)=0
- virtual void **stop\_channel** ()=0
- virtual void **set\_signal** (const `Gnss_Signal` &)=0

### 10.34.1 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 43 of file `channel_interface.h`.

The documentation for this class was generated from the following file:

- [channel\\_interface.h](#)

## 10.35 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**

### 10.35.1 Detailed Description

Definition at line 45 of file [fft\\_internal.h](#).

The documentation for this struct was generated from the following file:

- [fft\\_internal.h](#)

## 10.36 clFFT\_Complex Struct Reference

### Public Attributes

- float **real**
- float **imag**

### 10.36.1 Detailed Description

Definition at line 65 of file [clFFT.h](#).

The documentation for this struct was generated from the following file:

- [clFFT.h](#)

## 10.37 cIFFT\_Dim3 Struct Reference

### Public Attributes

- unsigned int **x**
- unsigned int **y**
- unsigned int **z**

### 10.37.1 Detailed Description

Definition at line 52 of file cIFFT.h.

The documentation for this struct was generated from the following file:

- [cIFFT.h](#)

## 10.38 cIFFT\_SplitComplex Struct Reference

### Public Attributes

- float \* **real**
- float \* **imag**

### 10.38.1 Detailed Description

Definition at line 59 of file cIFFT.h.

The documentation for this struct was generated from the following file:

- [cIFFT.h](#)

## 10.39 cnav\_msg\_decoder\_t Struct Reference

```
#include <cnav_msg.h>
```

### Public Attributes

- [cnav\\_v27\\_part\\_t](#) part1
- [cnav\\_v27\\_part\\_t](#) part2

### 10.39.1 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.

### 10.39.2 Member Data Documentation

#### 10.39.2.1 part1

[cnav\\_v27\\_part\\_t](#) cnav\_msg\_decoder\_t::part1

Decoder for odd symbol pairs

Definition at line 97 of file cnav\_msg.h.

#### 10.39.2.2 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](#)

## 10.40 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](#)]

### 10.40.1 Detailed Description

GPS CNAV message container.

See also

`cnav_msg_decoder_add_symbol`

Definition at line 52 of file `cnav_msg.h`.

### 10.40.2 Member Data Documentation

#### 10.40.2.1 alert

```
bool cnav_msg_t::alert
```

CNAV message alert flag

Definition at line 57 of file `cnav_msg.h`.

#### 10.40.2.2 msg\_id

```
uint8_t cnav_msg_t::msg_id
```

Message id. 0..31

Definition at line 55 of file `cnav_msg.h`.

#### 10.40.2.3 prn

```
uint8_t cnav_msg_t::prn
```

SV PRN. 0..31

Definition at line 54 of file `cnav_msg.h`.

## 10.40.2.4 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.

## 10.40.2.5 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](#)

## 10.41 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](#)

## 10.41.1 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.

## 10.41.2 Member Data Documentation

### 10.41.2.1 `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`.

### 10.41.2.2 `dec`

```
v27_t cnav_v27_part_t::dec
```

Viterbi block decoder object

Definition at line 69 of file `cnav_msg.h`.

### 10.41.2.3 `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`.

### 10.41.2.4 `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`.

### 10.41.2.5 `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`.

#### 10.41.2.6 invert

```
bool cnav_v27_part_t::invert
```

When true, indicates the bits are inverted

Definition at line 80 of file cnav\_msg.h.

#### 10.41.2.7 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.

#### 10.41.2.8 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.

#### 10.41.2.9 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.

#### 10.41.2.10 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.

#### 10.41.2.11 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.

#### 10.41.2.12 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](#)

## 10.42 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)

#### 10.42.1 Detailed Description

Definition at line 34 of file command\_event.h.

The documentation for this class was generated from the following file:

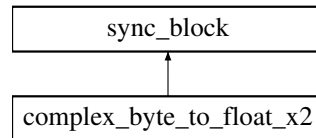
- [command\\_event.h](#)

## 10.43 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** ()

#### 10.43.1 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:

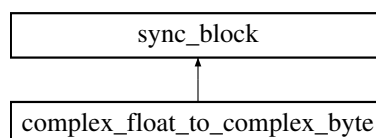
- [complex\\_byte\\_to\\_float\\_x2.h](#)

## 10.44 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** ()

### 10.44.1 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](#)

## 10.45 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** ()
- size\_t **size** ()
- bool **read** (int key, Data &p\_data)

### 10.45.1 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.

The documentation for this class was generated from the following file:

- [concurrent\\_map.h](#)

## 10.46 Concurrent\_Queue< Data > Class Template Reference

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

```
#include <concurrent_queue.h>
```

### Public Member Functions

- void **push** (Data const &data)
- bool **empty** () const
- bool **try\_pop** (Data &popped\_value)
- void **wait\_and\_pop** (Data &popped\_value)
- bool **timed\_wait\_and\_pop** (Data &popped\_value, int wait\_ms)

### 10.46.1 Detailed Description

```
template<typename Data>  
class Concurrent_Queue< Data >
```

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

Thread-safe object queue which uses the library boost\_thread to perform MUTEX based on the code available at <https://www.justsoftwaresolutions.co.uk/threading/implementing-a-thread-safe-queue-using-boost-mutex.html>

Definition at line 39 of file acquisition\_interface.h.

The documentation for this class was generated from the following files:

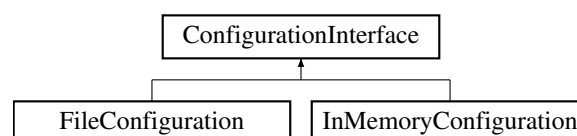
- [acquisition\\_interface.h](#)
- [concurrent\\_queue.h](#)

## 10.47 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

### 10.47.1 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:

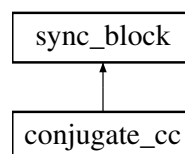
- [configuration\\_interface.h](#)

## 10.48 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** ()

### 10.48.1 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:

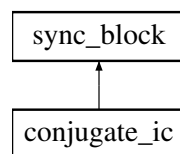
- [conjugate\\_cc.h](#)

## 10.49 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** ()

### 10.49.1 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:

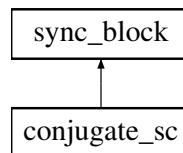
- [conjugate\\_ic.h](#)

## 10.50 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 ()`

#### 10.50.1 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](#)

## 10.51 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.*

### 10.51.1 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 63 of file `control_thread.h`.

### 10.51.2 Constructor & Destructor Documentation

#### 10.51.2.1 [ControlThread\(\)](#) [1/2]

```
ControlThread::ControlThread ( )
```

Default constructor.

#### 10.51.2.2 [ControlThread\(\)](#) [2/2]

```
ControlThread::ControlThread (
    std::shared_ptr< ConfigurationInterface > configuration ) [explicit]
```

Constructor that initializes the class with parameters.

#### Parameters

in	<i>configuration</i>	Pointer to a <a href="#">ConfigurationInterface</a>
----	----------------------	---

### 10.51.2.3 ~ControlThread()

```
ControlThread::~~ControlThread ( )
```

Destructor.

## 10.51.3 Member Function Documentation

### 10.51.3.1 flowgraph()

```
std::shared_ptr<GNSSFlowgraph> ControlThread::flowgraph ( ) [inline]
```

Instantiates a flowgraph.

#### Returns

Returns a smart pointer to a flowgraph object

Definition at line 119 of file control\_thread.h.

### 10.51.3.2 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; }

### 10.51.3.3 set\_control\_queue()

```
void ControlThread::set_control_queue (
    std::shared_ptr< Concurrent_Queue< pmt::pmt_t >> control_queue )
```

Sets the control\_queue.

## Parameters

in	<code>std::shared_ptr&lt;Concurrent_Queue&lt;pmt::pmt_t&gt;&gt;</code>	control_queue
----	--	---------------

The documentation for this class was generated from the following file:

- [control\\_thread.h](#)

## 10.52 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** ()

### 10.52.1 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](#)

## 10.53 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** ()

### 10.53.1 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](#)

## 10.54 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 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)
- **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 free** ()

### 10.54.1 Detailed Description

Class that implements carrier wipe-off and correlators.

Definition at line 37 of file `cpu_multicorrelator_real_codes.h`.

The documentation for this class was generated from the following file:

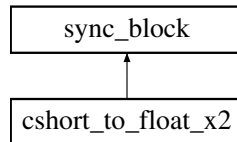
- [cpu\\_multicorrelator\\_real\\_codes.h](#)

## 10.55 cshort\_to\_float\_x2 Class Reference

This class adapts a `std::complex<short>` stream into two 32-bits (float) streams.

```
#include <csort_to_float_x2.h>
```

Inheritance diagram for `csort_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

- `csort_to_float_x2_sptr` **make\_csort\_to\_float\_x2** ()

#### 10.55.1 Detailed Description

This class adapts a `std::complex<short>` stream into two 32-bits (float) streams.

Definition at line 41 of file `csort_to_float_x2.h`.

The documentation for this class was generated from the following file:

- [csort\\_to\\_float\\_x2.h](#)

## 10.56 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

### 10.56.1 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](#)

## 10.57 `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)

### 10.57.1 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:

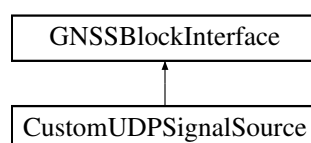
- [cuda\\_multicorrelator.h](#)

## 10.58 `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)
- std::string **role** () override
- std::string **implementation** () override  
*Returns "Custom\_UDP\_Signal\_Source".*
- 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

### 10.58.1 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.

### 10.58.2 Member Function Documentation

#### 10.58.2.1 implementation()

```
std::string CustomUDPSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "Custom\_UDP\_Signal\_Source".

Implements [GNSSBlockInterface](#).

Definition at line 60 of file custom\_udp\_signal\_source.h.

The documentation for this class was generated from the following file:

- [custom\\_udp\\_signal\\_source.h](#)

## 10.59 dgps\_t Struct Reference

### Public Attributes

- [gtime\\_t](#) t0
- double prc
- double rrc
- int iod
- double udre

### 10.59.1 Detailed Description

Definition at line 647 of file rtklib.h.

The documentation for this struct was generated from the following file:

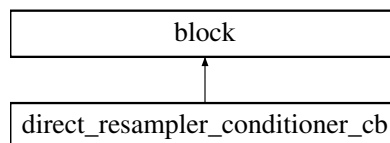
- [rtklib.h](#)

## 10.60 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)

### 10.60.1 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`.

The documentation for this class was generated from the following file:

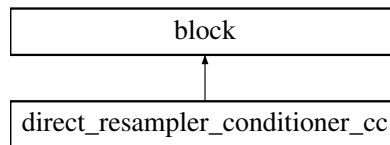
- [direct\\_resampler\\_conditioner\\_cb.h](#)

## 10.61 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)

#### 10.61.1 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.

The documentation for this class was generated from the following file:

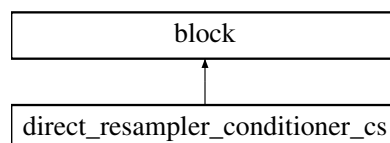
- [direct\\_resampler\\_conditioner\\_cc.h](#)

## 10.62 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)

### 10.62.1 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`.

The documentation for this class was generated from the following file:

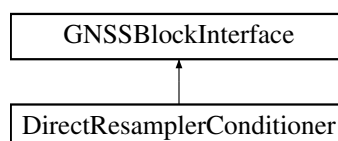
- [direct\\_resampler\\_conditioner\\_cs.h](#)

## 10.63 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

### 10.63.1 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`.

### 10.63.2 Member Function Documentation

#### 10.63.2.1 implementation()

```
std::string DirectResamplerConditioner::implementation ( ) [inline], [override], [virtual]
```

Returns "Direct\_Resampler".

Implements [GNSSBlockInterface](#).

Definition at line 53 of file `direct_resampler_conditioner.h`.

The documentation for this class was generated from the following file:

- [direct\\_resampler\\_conditioner.h](#)

## 10.64 Dll\_Pll\_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**
- float **pll\_pull\_in\_bw\_hz**
- float **dll\_pull\_in\_bw\_hz**
- float **fil\_bw\_hz**
- float **pll\_bw\_hz**
- float **dll\_bw\_hz**
- float **pll\_bw\_narrow\_hz**
- float **dll\_bw\_narrow\_hz**
- 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 **fil\_filter\_order**
- int32\_t **pll\_filter\_order**
- int32\_t **dll\_filter\_order**
- 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 **enable\_fil\_pull\_in**
- bool **enable\_fil\_steady\_state**
- bool **track\_pilot**
- bool **enable\_doppler\_correction**
- bool **carrier\_aiding**
- bool **high\_dyn**
- bool **dump**
- bool **dump\_mat**

### 10.64.1 Detailed Description

Definition at line 32 of file `dll_pll_conf.h`.

The documentation for this class was generated from the following file:

- [dll\\_pll\\_conf.h](#)

## 10.65 Dll\_Pll\_Conf\_Fpga Class Reference

### Public Member Functions

- void **SetFromConfiguration** (const [ConfigurationInterface](#) \*configuration, const std::string &role)

## Public Attributes

- std::string **device\_name**
- std::string **dump\_filename**
- double **fs\_in**
- double **carrier\_lock\_th**
- float **pll\_pull\_in\_bw\_hz**
- float **dll\_pull\_in\_bw\_hz**
- float **fil\_bw\_hz**
- float **pll\_bw\_hz**
- float **dll\_bw\_hz**
- float **pll\_bw\_narrow\_hz**
- float **dll\_bw\_narrow\_hz**
- 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**
- uint32\_t **code\_length\_chips**
- uint32\_t **code\_samples\_per\_chip**
- uint32\_t **extend\_fpga\_integration\_periods**
- uint32\_t **fpga\_integration\_period**
- int32\_t **fil\_filter\_order**
- int32\_t **pll\_filter\_order**
- int32\_t **dll\_filter\_order**
- int32\_t **extend\_correlation\_symbols**
- int32\_t **cn0\_samples**
- int32\_t **cn0\_min**
- int32\_t **max\_code\_lock\_fail**
- int32\_t **max\_carrier\_lock\_fail**
- int32\_t **cn0\_smoother\_samples**
- int32\_t **carrier\_lock\_test\_smoother\_samples**
- int32\_t \* **ca\_codes**
- int32\_t \* **data\_codes**
- char **signal** [3]
- char **system**
- bool **extended\_correlation\_in\_fpga**
- bool **track\_pilot**
- bool **enable\_doppler\_correction**
- bool **enable\_fil\_pull\_in**
- bool **enable\_fil\_steady\_state**
- bool **carrier\_aiding**
- bool **high\_dyn**
- bool **dump**
- bool **dump\_mat**

### 10.65.1 Detailed Description

Definition at line 34 of file `dll_pll_conf_fpga.h`.

The documentation for this class was generated from the following file:

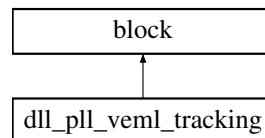
- [dll\\_pll\\_conf\\_fpga.h](#)

## 10.66 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)
- void **forecast** (int noutput\_items, gr\_vector\_int &ninput\_items\_required)

### Friends

- `dll_pll_veml_tracking_sptr` **dll\_pll\_veml\_make\_tracking** (const [Dll\\_Pll\\_Conf](#) &conf\_)

### 10.66.1 Detailed Description

This class implements a code DLL + carrier PLL tracking block.

Definition at line 57 of file `dll_pll_veml_tracking.h`.

The documentation for this class was generated from the following file:

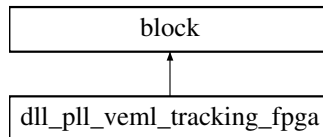
- [dll\\_pll\\_veml\\_tracking.h](#)

## 10.67 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, 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\_)

#### 10.67.1 Detailed Description

This class implements a code DLL + carrier PLL tracking block.

Definition at line 58 of file `dll_pll_veml_tracking_fpga.h`.

#### 10.67.2 Constructor & Destructor Documentation

### 10.67.2.1 ~dll\_pll\_veml\_tracking\_fpga()

```
dll_pll_veml_tracking_fpga::~dll_pll_veml_tracking_fpga ( )
```

Destructor.

## 10.67.3 Member Function Documentation

### 10.67.3.1 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.

### 10.67.3.2 reset()

```
void dll_pll_veml_tracking_fpga::reset ( )
```

This function disables the HW multicorrelator in the FPGA in order to stop the tracking process.

### 10.67.3.3 set\_channel()

```
void dll_pll_veml_tracking_fpga::set_channel (
    uint32_t channel,
    std::string device_io_name )
```

Set the channel number and configure some multicorrelator parameters.

### 10.67.3.4 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.

## 10.67.3.5 start\_tracking()

```
void dll_pll_veml_tracking_fpga::start_tracking ( )
```

This function starts the tracking process.

## 10.67.3.6 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](#)

## 10.68 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 **tgdc** [4]
- double **isc** [4]
- double **Adot**
- double **ndot**

### 10.68.1 Detailed Description

Definition at line 439 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.69 `erp_t` Struct Reference

### Public Attributes

- int **n**
- int **nmax**
- [erpd\\_t](#) \* **data**

### 10.69.1 Detailed Description

Definition at line 399 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.70 `erpd_t` Struct Reference

### Public Attributes

- double **mjd**
- double **xp**
- double **yp**
- double **xpr**
- double **ypr**
- double **ut1\_utc**
- double **lod**

### 10.70.1 Detailed Description

Definition at line 389 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.71 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)

### 10.71.1 Detailed Description

Class that implements a first-order exponential smoother.

$$\text{smoothed\_value}[k] = \alpha * \text{raw} + (1-\alpha) * \text{smoothed\_value}[k-1]$$

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.

### 10.71.2 Constructor & Destructor Documentation

#### 10.71.2.1 Exponential\_Smoother() [1/2]

```
Exponential_Smoother::Exponential_Smoother ( )
```

Constructor.

### 10.71.2.2 ~Exponential\_Smoother()

```
Exponential_Smoother::~~Exponential_Smoother ( ) [default]
```

Destructor.

### 10.71.2.3 Exponential\_Smoother() [2/2]

```
Exponential_Smoother::Exponential_Smoother (
    Exponential_Smoother && ) [default]
```

Move operator.

## 10.71.3 Member Function Documentation

### 10.71.3.1 operator=()

```
Exponential_Smoother& Exponential_Smoother::operator= (
    Exponential_Smoother && ) [default]
```

Move assignment operator.

### 10.71.3.2 set\_alpha()

```
void Exponential_Smoother::set_alpha (
    float alpha )
```

$0 < \alpha < 1$ . The higher, the most responsive, but more variance. Default value: 0.001

### 10.71.3.3 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](#)

## 10.72 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](#)]

### 10.72.1 Detailed Description

Definition at line 927 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.73 fcbd\_t Struct Reference

### Public Attributes

- [gtime\\_t](#) **ts**
- [gtime\\_t](#) **te**
- double **bias** [MAXSAT][3]
- double **std** [MAXSAT][3]

### 10.73.1 Detailed Description

Definition at line 559 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.74 file\_t Struct Reference

### Public Attributes

- FILE \* **fp**
- FILE \* **fp\_tag**
- FILE \* **fp\_tmp**
- FILE \* **fp\_tag\_tmp**
- char **path** [[MAXSTRPATH](#)]
- char **openpath** [[MAXSTRPATH](#)]
- int **mode**
- int **timetag**
- int **repmode**
- int **offset**
- [gtime\\_t](#) **time**
- [gtime\\_t](#) **wtime**
- unsigned int **tick**
- unsigned int **tick\_f**
- unsigned int **fpos**
- double **start**
- double **speed**
- double **swapintv**
- lock\_t **lock**

### 10.74.1 Detailed Description

Definition at line 1118 of file rtklib.h.

The documentation for this struct was generated from the following file:

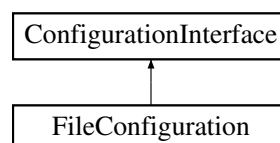
- [rtklib.h](#)

## 10.75 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

### 10.75.1 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](https://en.wikipedia.org/wiki/INI_file)

Definition at line 48 of file file\_configuration.h.

The documentation for this class was generated from the following file:

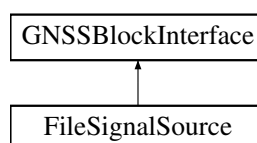
- [file\\_configuration.h](#)

## 10.76 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** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams, [Concurrent\\_Queue](#)< pmt::pmt\_t > \*queue)
- std::string **role** () override
- std::string **implementation** () override  
*Returns "File\_Signal\_Source".*
- 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

### 10.76.1 Detailed Description

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.

Definition at line 48 of file file\_signal\_source.h.

### 10.76.2 Member Function Documentation

#### 10.76.2.1 implementation()

```
std::string FileSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "File\_Signal\_Source".

Implements [GNSSBlockInterface](#).

Definition at line 65 of file file\_signal\_source.h.

The documentation for this class was generated from the following file:

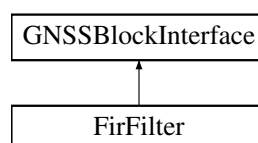
- [file\\_signal\\_source.h](#)

## 10.77 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

### 10.77.1 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](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`.

### 10.77.2 Constructor & Destructor Documentation

#### 10.77.2.1 FirFilter()

```
FirFilter::FirFilter (
    const ConfigurationInterface * configuration,
    std::string role,
    unsigned int in_streams,
    unsigned int out_streams )
```

Constructor.

#### 10.77.2.2 ~FirFilter()

```
FirFilter::~~FirFilter ( ) [default]
```

Destructor.

### 10.77.3 Member Function Documentation

#### 10.77.3.1 implementation()

```
std::string FirFilter::implementation ( ) [inline], [override], [virtual]
```

Returns "Fir\_Filter".

Implements [GNSSBlockInterface](#).

Definition at line 77 of file fir\_filter.h.

The documentation for this class was generated from the following file:

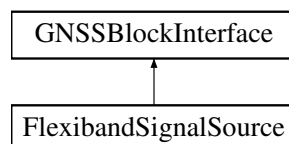
- [fir\\_filter.h](#)

## 10.78 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)
- std::string **role** () override
- std::string **implementation** () override  
Returns "Flexiband\_Signal\_Source".
- 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

### 10.78.1 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.

### 10.78.2 Member Function Documentation

#### 10.78.2.1 implementation()

```
std::string FlexibandSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "Flexiband\_Signal\_Source".

Implements [GNSSBlockInterface](#).

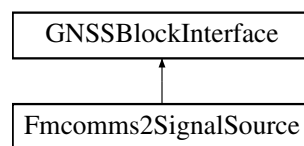
Definition at line 65 of file flexiband\_signal\_source.h.

The documentation for this class was generated from the following file:

- [flexiband\\_signal\\_source.h](#)

## 10.79 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)
- std::string **role** () override
- std::string **implementation** () override  
*Returns "Fmcomms2\_Signal\_Source".*
- 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

### 10.79.1 Detailed Description

Definition at line 44 of file fmcomms2\_signal\_source.h.

### 10.79.2 Member Function Documentation

#### 10.79.2.1 implementation()

```
std::string Fmcomms2SignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "Fmcomms2\_Signal\_Source".

Implements [GNSSBlockInterface](#).

Definition at line 61 of file fmcomms2\_signal\_source.h.

The documentation for this class was generated from the following file:

- [fmcomms2\\_signal\\_source.h](#)

## 10.80 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 nsamples, uint32\_t doppler\_max, uint32\_t nsamples\_↵ total, int64\_t fs\_in, uint32\_t select\_queue, uint32\_t \*all\_fft\_codes, uint32\_t excludelimit)  
*Constructor.*
- [~Fpga\\_Acquisition](#) ()=default  
*Destructor.*
- 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 [set\\_doppler\\_max](#) (uint32\_t doppler\_max)  
*Set maximum Doppler grid search.*
- void [set\\_doppler\\_step](#) (uint32\_t doppler\_step)

- *Set Doppler steps for the grid search.*
- void `reset_acquisition` ()
- *Reset the FPGA PL.*
- void `read_fpga_total_scale_factor` (uint32\_t \*total\_scale\_factor, uint32\_t \*fw\_scale\_factor)
- *Read the scaling factor that has been used by the FFT-IFFT.*
- 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.*

### 10.80.1 Detailed Description

Class that implements carrier wipe-off and correlators.

Definition at line 37 of file `fpga_acquisition.h`.

### 10.80.2 Constructor & Destructor Documentation

#### 10.80.2.1 Fpga\_Acquisition()

```
Fpga_Acquisition::Fpga_Acquisition (
    std::string device_name,
    uint32_t nsamples,
    uint32_t doppler_max,
    uint32_t nsamples_total,
    int64_t fs_in,
    uint32_t select_queue,
    uint32_t * all_fft_codes,
    uint32_t excludelimit )
```

Constructor.

#### 10.80.2.2 ~Fpga\_Acquisition()

```
Fpga_Acquisition::~~Fpga_Acquisition ( ) [default]
```

Destructor.

### 10.80.3 Member Function Documentation

#### 10.80.3.1 close\_device()

```
void Fpga_Acquisition::close_device ( )
```

Close the device driver.

#### 10.80.3.2 configure\_acquisition()

```
void Fpga_Acquisition::configure_acquisition (
    void )
```

Write the acquisition parameters into the FPGA.

#### 10.80.3.3 open\_device()

```
void Fpga_Acquisition::open_device ( )
```

Open the device driver.

#### 10.80.3.4 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.

#### 10.80.3.5 read\_fpga\_total\_scale\_factor()

```
void Fpga_Acquisition::read_fpga_total_scale_factor (
    uint32_t * total_scale_factor,
    uint32_t * fw_scale_factor )
```

Read the scaling factor that has been used by the FFT-IFFT.

### 10.80.3.6 reset\_acquisition()

```
void Fpga_Acquisition::reset_acquisition ( )
```

Reset the FPGA PL.

### 10.80.3.7 run\_acquisition()

```
void Fpga_Acquisition::run_acquisition ( )
```

Run the acquisition process in the FPGA.

### 10.80.3.8 set\_block\_exp()

```
void Fpga_Acquisition::set_block_exp (
    uint32_t total_block_exp )
```

Set the block exponent of the FFT in the FPGA.

### 10.80.3.9 set\_doppler\_max()

```
void Fpga_Acquisition::set_doppler_max (
    uint32_t doppler_max ) [inline]
```

Set maximum Doppler grid search.

#### Parameters

<i>doppler_max</i>	- Maximum Doppler shift considered in the grid search [Hz].
--------------------	---

Definition at line 89 of file fpga\_acquisition.h.

### 10.80.3.10 set\_doppler\_step()

```
void Fpga_Acquisition::set_doppler_step (
    uint32_t doppler_step ) [inline]
```

Set Doppler steps for the grid search.

## Parameters

<i>doppler_step</i>	- Frequency bin of the search grid [Hz].
---------------------	--

Definition at line 98 of file `fpga_acquisition.h`.

#### 10.80.3.11 `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.

#### 10.80.3.12 `set_local_code()`

```
bool Fpga_Acquisition::set_local_code (
    uint32_t PRN )
```

Select the code with the chosen PRN.

#### 10.80.3.13 `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](#)

## 10.81 `Fpga_dynamic_bit_selection` 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_dynamic_bit_selection.h>
```

## Public Member Functions

- [Fpga\\_dynamic\\_bit\\_selection](#) (const std::string &device\_name1, const std::string &device\_name2)  
*Constructor.*
- [~Fpga\\_dynamic\\_bit\\_selection](#) ()  
*Destructor.*
- void [bit\\_selection](#) (void)  
*This function configures the switch in the eFPGA.*

### 10.81.1 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_dynamic_bit_selection.h`.

### 10.81.2 Constructor & Destructor Documentation

#### 10.81.2.1 Fpga\_dynamic\_bit\_selection()

```
Fpga_dynamic_bit_selection::Fpga_dynamic_bit_selection (
    const std::string & device_name1,
    const std::string & device_name2 ) [explicit]
```

Constructor.

#### 10.81.2.2 ~Fpga\_dynamic\_bit\_selection()

```
Fpga_dynamic_bit_selection::~~Fpga_dynamic_bit_selection ( )
```

Destructor.

### 10.81.3 Member Function Documentation

#### 10.81.3.1 bit\_selection()

```
void Fpga_dynamic_bit_selection::bit_selection (
    void )
```

This function configures the switch in the eFPGA.

The documentation for this class was generated from the following file:

- [fpga\\_dynamic\\_bit\\_selection.h](#)

## 10.82 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](#) (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.*

### 10.82.1 Detailed Description

Class that implements carrier wipe-off and correlators.

Definition at line 40 of file `fpga_multicorrelator.h`.

### 10.82.2 Constructor & Destructor Documentation

#### 10.82.2.1 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.

#### 10.82.2.2 ~Fpga\_Multicorrelator\_8sc()

```
Fpga_Multicorrelator_8sc::~Fpga_Multicorrelator_8sc ( )
```

Destructor.

### 10.82.3 Member Function Documentation

#### 10.82.3.1 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.

#### 10.82.3.2 `disable_secondary_codes()`

```
void Fpga_Multicorrelator_8sc::disable_secondary_codes ( )
```

Disable the use of secondary codes in the FPGA.

#### 10.82.3.3 `enable_secondary_codes()`

```
void Fpga_Multicorrelator_8sc::enable_secondary_codes ( )
```

Enable the use of secondary codes in the FPGA.

#### 10.82.3.4 `free()`

```
bool Fpga_Multicorrelator_8sc::free ( )
```

Stop the correlation process in the FPGA and free code phase and code rate parameters.

#### 10.82.3.5 `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`.

#### 10.82.3.6 `lock_channel()`

```
void Fpga_Multicorrelator_8sc::lock_channel ( )
```

Start the tracking process in the FPGA.

#### 10.82.3.7 `open_channel()`

```
void Fpga_Multicorrelator_8sc::open_channel (
    std::string device_io_name,
    uint32_t channel )
```

Open the FPGA device driver.

### 10.82.3.8 read\_sample\_counter()

```
uint64_t Fpga_Multicorrelator_8sc::read_sample_counter ( )
```

Read the sample counter in the FPGA.

### 10.82.3.9 set\_initial\_sample()

```
void Fpga_Multicorrelator_8sc::set_initial_sample (
    uint64_t samples_offset )
```

Set the initial sample number where the tracking process begins.

### 10.82.3.10 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.

### 10.82.3.11 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.

### 10.82.3.12 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.

**10.82.3.13 unlock\_channel()**

```
void Fpga_Multicorrelator_8sc::unlock_channel ( )
```

finish the tracking process in the FPGA

**10.82.3.14 update\_local\_code()**

```
void Fpga_Multicorrelator_8sc::update_local_code ( )
```

Configure code phase and code rate parameters in the FPGA.

**10.82.3.15 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](#)

**10.83 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](#) (const std::string &device\_name)  
*Constructor.*
- [~Fpga\\_Switch](#) ()  
*Destructor.*
- void [set\\_switch\\_position](#) (int32\_t switch\_position)  
*This function configures the switch in the FPGA.*

### 10.83.1 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 38 of file `fpga_switch.h`.

### 10.83.2 Constructor & Destructor Documentation

#### 10.83.2.1 Fpga\_Switch()

```
Fpga_Switch::Fpga_Switch (
    const std::string & device_name ) [explicit]
```

Constructor.

#### 10.83.2.2 ~Fpga\_Switch()

```
Fpga_Switch::~~Fpga_Switch ( )
```

Destructor.

### 10.83.3 Member Function Documentation

#### 10.83.3.1 set\_switch\_position()

```
void Fpga_Switch::set_switch_position (
    int32_t switch_position )
```

This function configures the switch in the eFPGA.

The documentation for this class was generated from the following file:

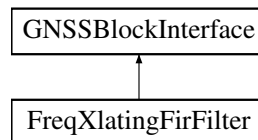
- [fpga\\_switch.h](#)

## 10.84 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

### 10.84.1 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](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`.

### 10.84.2 Member Function Documentation

## 10.84.2.1 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.

The documentation for this class was generated from the following file:

- [freq\\_xlating\\_fir\\_filter.h](#)

## 10.85 FrontEndCal Class Reference

### Public Member Functions

- void [set\\_configuration](#) (std::shared\_ptr< [ConfigurationInterface](#) > configuration)  
*Sets the configuration data required by get\_ephemeris function.*
- bool [get\\_ephemeris](#) ()  
*This function connects to a Secure User Location Protocol (SUPL) server to obtain the current GPS ephemeris and GPS assistance data. It requires the configuration parameters set by set\_configuration function.*
- double [estimate\\_doppler\\_from\\_eph](#) (unsigned int PRN, double tow, double lat, double lon, double height) noexcept(false)  
*This function estimates the GPS L1 satellite Doppler frequency [Hz] using the following data: 1- Orbital model from the ephemeris 2- Approximate GPS Time of Week (TOW) 3- Approximate receiver Latitude and Longitude (WGS-84)*
- void [GPS\\_L1\\_front\\_end\\_model\\_E4000](#) (double f\_bb\_true\_Hz, double f\_bb\_meas\_Hz, double fs\_nominal\_Hz, double \*estimated\_fs\_Hz, double \*estimated\_f\_if\_Hz, double \*f\_osc\_err\_ppm)  
*This function models the Elonics E4000 + RTL2832 front-end Inputs: f\_bb\_true\_Hz - Ideal output frequency in baseband [Hz] f\_in\_bb\_meas\_Hz - measured output frequency in baseband [Hz] Outputs: estimated\_fs\_Hz - Sampling frequency estimation based on the measurements and the front-end model estimated\_f\_if\_bb\_Hz - Equivalent bb if frequency estimation based on the measurements and the front-end model Front-end TUNER Elonics E4000 + RTL2832 sampler For GPS L1 1575.42 MHz.*

### 10.85.1 Detailed Description

Definition at line 27 of file front\_end\_cal.h.

### 10.85.2 Member Function Documentation

### 10.85.2.1 estimate\_doppler\_from\_eph()

```
double FrontEndCal::estimate_doppler_from_eph (
    unsigned int PRN,
    double tow,
    double lat,
    double lon,
    double height ) [noexcept]
```

This function estimates the GPS L1 satellite Doppler frequency [Hz] using the following data: 1- Orbital model from the ephemeris 2- Approximate GPS Time of Week (TOW) 3- Approximate receiver Latitude and Longitude (WGS-84)

### 10.85.2.2 get\_ephemeris()

```
bool FrontEndCal::get_ephemeris ( )
```

This function connects to a Secure User Location Protocol (SUPL) server to obtain the current GPS ephemeris and GPS assistance data. It requires the configuration parameters set by set\_configuration function.

### 10.85.2.3 GPS\_L1\_front\_end\_model\_E4000()

```
void FrontEndCal::GPS_L1_front_end_model_E4000 (
    double f_bb_true_Hz,
    double f_bb_meas_Hz,
    double fs_nominal_hz,
    double * estimated_fs_Hz,
    double * estimated_f_if_Hz,
    double * f_osc_err_ppm )
```

This function models the Elonics E4000 + RTL2832 front-end Inputs: f\_bb\_true\_Hz - Ideal output frequency in baseband [Hz] f\_in\_bb\_meas\_Hz - measured output frequency in baseband [Hz] Outputs: estimated\_fs\_Hz - Sampling frequency estimation based on the measurements and the front-end model estimated\_f\_if\_bb\_Hz - Equivalent bb if frequency estimation based on the measurements and the front-end model Front-end TUNER Elonics E4000 + RTL2832 sampler For GPS L1 1575.42 MHz.

### 10.85.2.4 set\_configuration()

```
void FrontEndCal::set_configuration (
    std::shared_ptr< ConfigurationInterface > configuration )
```

Sets the configuration data required by get\_ephemeris function.

The documentation for this class was generated from the following file:

- [front\\_end\\_cal.h](#)

## 10.86 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**

### 10.86.1 Detailed Description

Definition at line 1185 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.87 Galileo\_Almanac Class Reference

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

```
#include <galileo_almanac.h>
```

### Public Member Functions

- [Galileo\\_Almanac](#) ()=default
- [template<class Archive >](#)  
void **serialize** (Archive &ar, const unsigned int version)

## Public Attributes

- uint32\_t [i\\_satellite\\_PRN](#) {}  
*SV PRN NUMBER.*
- int32\_t [i\\_Toa](#) {}
- int32\_t [i\\_WNa](#) {}
- int32\_t [i\\_IODa](#) {}
- double [d\\_Delta\\_i](#) {}  
*Inclination at reference time relative to  $i_0 = 56^\circ$  [semi-circles].*
- double [d\\_M\\_0](#) {}  
*Mean Anomaly at Reference Time [semi-circles].*
- double [d\\_e\\_eccentricity](#) {}  
*Eccentricity [dimensionless].*
- double [d\\_Delta\\_sqrt\\_A](#) {}  
*Square Root of the Semi-Major Axis [sqrt(m)].*
- double [d\\_OMEGA0](#) {}  
*Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles].*
- double [d\\_OMEGA](#) {}  
*Argument of Perigee [semi-circles].*
- double [d\\_OMEGA\\_DOT](#) {}  
*Rate of Right Ascension [semi-circles/s].*
- double [d\\_A\\_f0](#) {}  
*Coefficient 0 of code phase offset model [s].*
- double [d\\_A\\_f1](#) {}  
*Coefficient 1 of code phase offset model [s/s].*
- int32\_t [E5b\\_HS](#) {}
- int32\_t [E1B\\_HS](#) {}
- int32\_t [E5a\\_HS](#) {}

### 10.87.1 Detailed Description

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

Definition at line 33 of file galileo\_almanac.h.

### 10.87.2 Constructor & Destructor Documentation

#### 10.87.2.1 Galileo\_Almanac()

```
Galileo_Almanac::Galileo_Almanac ( ) [default]
```

Default constructor

### 10.87.3 Member Data Documentation

### 10.87.3.1 d\_A\_f0

```
double Galileo_Almanac::d_A_f0 {}
```

Coefficient 0 of code phase offset model [s].

Definition at line 52 of file galileo\_almanac.h.

### 10.87.3.2 d\_A\_f1

```
double Galileo_Almanac::d_A_f1 {}
```

Coefficient 1 of code phase offset model [s/s].

Definition at line 53 of file galileo\_almanac.h.

### 10.87.3.3 d\_Delta\_i

```
double Galileo_Almanac::d_Delta_i {}
```

Inclination at reference time relative to  $i_0 = 56^\circ$  [semi-circles].

Definition at line 45 of file galileo\_almanac.h.

### 10.87.3.4 d\_Delta\_sqrt\_A

```
double Galileo_Almanac::d_Delta_sqrt_A {}
```

Square Root of the Semi-Major Axis [sqrt(m)].

Definition at line 48 of file galileo\_almanac.h.

### 10.87.3.5 d\_e\_eccentricity

```
double Galileo_Almanac::d_e_eccentricity {}
```

Eccentricity [dimensionless].

Definition at line 47 of file galileo\_almanac.h.

### 10.87.3.6 d\_M\_0

```
double Galileo_Almanac::d_M_0 {}
```

Mean Anomaly at Reference Time [semi-circles].

Definition at line 46 of file galileo\_almanac.h.

### 10.87.3.7 d\_OMEGA

```
double Galileo_Almanac::d_OMEGA {}
```

Argument of Perigee [semi-circles].

Definition at line 50 of file galileo\_almanac.h.

### 10.87.3.8 d\_OMEGA0

```
double Galileo_Almanac::d_OMEGA0 {}
```

Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles].

Definition at line 49 of file galileo\_almanac.h.

### 10.87.3.9 d\_OMEGA\_DOT

```
double Galileo_Almanac::d_OMEGA_DOT {}
```

Rate of Right Ascension [semi-circles/s].

Definition at line 51 of file galileo\_almanac.h.

### 10.87.3.10 i\_satellite\_PRN

```
uint32_t Galileo_Almanac::i_satellite_PRN {}
```

SV PRN NUMBER.

Definition at line 41 of file galileo\_almanac.h.

The documentation for this class was generated from the following file:

- [galileo\\_almanac.h](#)

## 10.88 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 {}`

### 10.88.1 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-O+S-SIS-ICD.pdf> paragraph 5.1.10

Definition at line 34 of file `galileo_almanac_helper.h`.

### 10.88.2 Constructor & Destructor Documentation

#### 10.88.2.1 Galileo\_Almanac\_Helper()

```
Galileo_Almanac_Helper::Galileo_Almanac_Helper ( ) [default]
```

Default constructor.

The documentation for this class was generated from the following file:

- [galileo\\_almanac\\_helper.h](#)

## 10.89 Galileo\_Cnav\_Message Class Reference

This class handles the Galileo CNAV Data message, as described in the Galileo High Accuracy Service E6-B Signal-In-Space Message Specification v1.2 (April 2020)

```
#include <galileo_cnav_message.h>
```

### Public Member Functions

- `void read_HAS_page (const std::string &page_string)`
- `bool have_new_HAS_message ()`
- `bool is_HAS_in_test_mode () const`
- `bool is_HAS_message_dummy () const`
- `Galileo_HAS_data get_HAS_data () const`

### 10.89.1 Detailed Description

This class handles the Galileo CNAV Data message, as described in the Galileo High Accuracy Service E6-B Signal-In-Space Message Specification v1.2 (April 2020)

Definition at line 40 of file `galileo_cnav_message.h`.

The documentation for this class was generated from the following file:

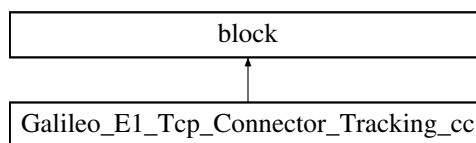
- [galileo\\_cnav\\_message.h](#)

## 10.90 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)

### 10.90.1 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:

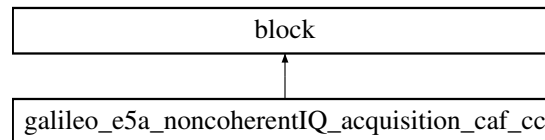
- [galileo\\_e1\\_tcp\\_connector\\_tracking\\_cc.h](#)

## 10.91 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)  
*Set acquisition/tracking common Gnss\_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.*
- unsigned int [mag](#) () const  
*Returns the maximum peak of grid search.*
- void [init](#) ()  
*Initializes acquisition algorithm.*
- void [set\\_local\\_code](#) (std::complex< float > \*code, std::complex< float > \*codeQ)  
*Sets local code for PCPS acquisition algorithm.*
- void [set\\_active](#) (bool active)  
*Starts acquisition algorithm, turning from standby mode to active mode.*
- void [set\\_state](#) (int state)  
*If set to 1, ensures that acquisition starts at the first available sample.*
- void [set\\_channel](#) (unsigned int 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\\_threshold](#) (float threshold)  
*Set statistics threshold of PCPS algorithm.*
- void [set\\_doppler\\_max](#) (unsigned int doppler\_max)  
*Set maximum Doppler grid search.*
- void [set\\_doppler\\_step](#) (unsigned int doppler\_step)  
*Set Doppler steps for the grid search.*
- int [general\\_work](#) (int noutput\_items, gr\_vector\_int &ninput\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)  
*Parallel Code Phase Search Acquisition signal processing.*

### Friends

- galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc\_sptr [galileo\\_e5a\\_noncoherentIQ\\_make\\_acquisition\\_caf\\_cc](#) (unsigned int sampled\_ms, unsigned int max\_dwells, unsigned int doppler\_max, int64\_t fs\_in, int samples\_per\_ms, int samples\_per\_code, bool bit\_transition\_flag, bool dump, const std::string &dump\_filename, bool both\_signal\_components\_, int CAF\_window\_hz\_, int Zero\_padding\_, bool enable\_monitor\_output)

### 10.91.1 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 67 of file `galileo_e5a_noncoherent_iq_acquisition_caf_cc.h`.

### 10.91.2 Constructor & Destructor Documentation

#### 10.91.2.1 `~galileo_e5a_noncoherentIQ_acquisition_caf_cc()`

```
galileo_e5a_noncoherentIQ_acquisition_caf_cc::~galileo_e5a_noncoherentIQ_acquisition_caf_cc (
)
```

Default destructor.

### 10.91.3 Member Function Documentation

#### 10.91.3.1 `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 )
```

Parallel Code Phase Search Acquisition signal processing.

#### 10.91.3.2 `init()`

```
void galileo_e5a_noncoherentIQ_acquisition_caf_cc::init ( )
```

Initializes acquisition algorithm.

#### 10.91.3.3 `mag()`

```
unsigned int galileo_e5a_noncoherentIQ_acquisition_caf_cc::mag ( ) const [inline]
```

Returns the maximum peak of grid search.

Definition at line 88 of file `galileo_e5a_noncoherent_iq_acquisition_caf_cc.h`.

#### 10.91.3.4 `set_active()`

```
void galileo_e5a_noncoherentIQ_acquisition_caf_cc::set_active (
    bool active ) [inline]
```

Starts acquisition algorithm, turning from standby mode to active mode.

## Parameters

<i>active</i>	- bool that activates/deactivates the block.
---------------	--

Definition at line 109 of file `galileo_e5a_noncoherent_iq_acquisition_caf_cc.h`.

10.91.3.5 `set_channel()`

```
void galileo_e5a_noncoherentIQ_acquisition_caf_cc::set_channel (
    unsigned int channel ) [inline]
```

Set acquisition channel unique ID.

## Parameters

<i>channel</i>	- receiver channel.
----------------	---------------------

Definition at line 125 of file `galileo_e5a_noncoherent_iq_acquisition_caf_cc.h`.

10.91.3.6 `set_channel_fsm()`

```
void galileo_e5a_noncoherentIQ_acquisition_caf_cc::set_channel_fsm (
    std::weak_ptr< ChannelFsm > channel_fsm ) [inline]
```

Set channel fsm associated to this acquisition instance.

Definition at line 133 of file `galileo_e5a_noncoherent_iq_acquisition_caf_cc.h`.

10.91.3.7 `set_doppler_max()`

```
void galileo_e5a_noncoherentIQ_acquisition_caf_cc::set_doppler_max (
    unsigned int doppler_max ) [inline]
```

Set maximum Doppler grid search.

## Parameters

<i>doppler_max</i>	- Maximum Doppler shift considered in the grid search [Hz].
--------------------	---

Definition at line 152 of file `galileo_e5a_noncoherent_iq_acquisition_caf_cc.h`.

## 10.91.3.8 set\_doppler\_step()

```
void galileo_e5a_noncoherentIQ_acquisition_caf_cc::set_doppler_step (
    unsigned int doppler_step ) [inline]
```

Set Doppler steps for the grid search.

## Parameters

<i>doppler_step</i>	- Frequency bin of the search grid [Hz].
---------------------	--

Definition at line 161 of file galileo\_e5a\_noncoherent\_iq\_acquisition\_caf\_cc.h.

## 10.91.3.9 set\_gnss\_synchro()

```
void galileo_e5a_noncoherentIQ_acquisition_caf_cc::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

<i>p_gnss_synchro</i>	Satellite information shared by the processing blocks.
-----------------------	--

Definition at line 80 of file galileo\_e5a\_noncoherent\_iq\_acquisition\_caf\_cc.h.

## 10.91.3.10 set\_local\_code()

```
void galileo_e5a_noncoherentIQ_acquisition_caf_cc::set_local_code (
    std::complex< float > * code,
    std::complex< float > * codeQ )
```

Sets local code for PCPS acquisition algorithm.

## Parameters

<i>code</i>	- Pointer to the PRN code.
-------------	----------------------------

## 10.91.3.11 set\_state()

```
void galileo_e5a_noncoherentIQ_acquisition_caf_cc::set_state (
    int state )
```

If set to 1, ensures that acquisition starts at the first available sample.

#### Parameters

<i>state</i>	- int=1 forces start of acquisition
--------------	-------------------------------------

#### 10.91.3.12 set\_threshold()

```
void galileo_e5a_noncoherentIQ_acquisition_caf_cc::set_threshold (
    float threshold ) [inline]
```

Set statistics threshold of PCPS algorithm.

#### Parameters

<i>threshold</i>	- Threshold for signal detection (check <a href="#">Navitec2012</a> , Algorithm 1, for a definition of this threshold).
------------------	---

Definition at line 143 of file galileo\_e5a\_noncoherent\_iq\_acquisition\_caf\_cc.h.

The documentation for this class was generated from the following file:

- [galileo\\_e5a\\_noncoherent\\_iq\\_acquisition\\_caf\\_cc.h](#)

## 10.92 Galileo\_Ephemeris Class Reference

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.pdf>)

```
#include <galileo_ephemeris.h>
```

### Public Member Functions

- void [satellitePosition](#) (double transmitTime)  
*Computes the ECEF SV coordinates and ECEF velocity.*
- double [Galileo\\_System\\_Time](#) (double WN, double TOW)  
*Galileo System Time (GST), ICD paragraph 5.1.2.*
- double [sv\\_clock\\_drift](#) (double transmitTime)  
*Satellite Time Correction Algorithm, ICD 5.1.4.*
- double [sv\\_clock\\_relativistic\\_term](#) (double transmitTime)  
*Satellite Time Correction Algorithm, ICD 5.1.4.*
- 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

- int32\_t **IOD\_ephemeris** {}
- int32\_t **IOD\_nav\_1** {}
- int32\_t **SV\_ID\_PRN\_4** {}
- double **M0\_1** {}  
*Mean anomaly at reference time [semi-circles].*
- double **delta\_n\_3** {}  
*Mean motion difference from computed value [semi-circles/sec].*
- double **e\_1** {}  
*Eccentricity.*
- double **A\_1** {}  
*Square root of the semi-major axis [meters<sup>1/2</sup>].*
- double **OMEGA\_0\_2** {}  
*Longitude of ascending node of orbital plane at weekly epoch [semi-circles].*
- double **i\_0\_2** {}  
*Inclination angle at reference time [semi-circles].*
- double **omega\_2** {}  
*Argument of perigee [semi-circles].*
- double **OMEGA\_dot\_3** {}  
*Rate of right ascension [semi-circles/sec].*
- double **iDot\_2** {}  
*Rate of inclination angle [semi-circles/sec].*
- double **C\_uc\_3** {}  
*Amplitude of the cosine harmonic correction term to the argument of latitude [radians].*
- double **C\_us\_3** {}  
*Amplitude of the sine harmonic correction term to the argument of latitude [radians].*
- double **C\_rc\_3** {}  
*Amplitude of the cosine harmonic correction term to the orbit radius [meters].*
- double **C\_rs\_3** {}  
*Amplitude of the sine harmonic correction term to the orbit radius [meters].*
- double **C\_ic\_4** {}  
*Amplitude of the cosine harmonic correction term to the angle of inclination [radians].*
- double **C\_is\_4** {}  
*Amplitude of the sine harmonic correction term to the angle of inclination [radians].*
- int32\_t **t0e\_1** {}  
*Ephemeris reference time [s].*
- int32\_t **t0c\_4** {}  
*Clock correction data reference Time of Week [sec].*
- double **af0\_4** {}  
*SV clock bias correction coefficient [s].*
- double **af1\_4** {}  
*SV clock drift correction coefficient [s/s].*
- double **af2\_4** {}  
*SV clock drift rate correction coefficient [s/s<sup>2</sup>].*
- int32\_t **WN\_5** {}  
*Week number.*
- int32\_t **TOW\_5** {}  
*Time of Week.*
- double **Galileo\_satClkDrift** {}
- double **Galileo\_dtr** {}

- relativistic clock correction term*
- int32\_t **SISA\_3** {}
- int32\_t **E5a\_HS** {}
  - E5a Signal Health Status.*
- int32\_t **E5b\_HS\_5** {}
  - E5b Signal Health Status.*
- int32\_t **E1B\_HS\_5** {}
  - E1B Signal Health Status.*
- bool **E5a\_DVS** {}
  - E5a Data Validity Status.*
- bool **E5b\_DVS\_5** {}
  - E5b Data Validity Status.*
- bool **E1B\_DVS\_5** {}
  - E1B Data Validity Status.*
- double **BGD\_E1E5a\_5** {}
  - E1-E5a Broadcast Group Delay [s].*
- double **BGD\_E1E5b\_5** {}
  - E1-E5b Broadcast Group Delay [s].*
- double **d\_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 **d\_satpos\_Y** {}
  - Earth-fixed coordinate y of the satellite [m]. Completes a right-handed, Earth-Centered, Earth-Fixed orthogonal coordinate system.*
- double **d\_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 **d\_satvel\_X** {}
  - Earth-fixed velocity coordinate x of the satellite [m].*
- double **d\_satvel\_Y** {}
  - Earth-fixed velocity coordinate y of the satellite [m].*
- double **d\_satvel\_Z** {}
  - Earth-fixed velocity coordinate z of the satellite [m].*
- uint32\_t **i\_satellite\_PRN** {}
  - SV PRN NUMBER.*
- bool **flag\_all\_ephemeris** {}

### 10.92.1 Detailed Description

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.pdf> )

Definition at line 36 of file galileo\_ephemeris.h.

### 10.92.2 Member Function Documentation

**10.92.2.1 Galileo\_System\_Time()**

```
double Galileo_Ephemeris::Galileo_System_Time (
    double WN,
    double TOW )
```

Galileo System Time (GST), ICD paragraph 5.1.2.

**10.92.2.2 satellitePosition()**

```
void Galileo_Ephemeris::satellitePosition (
    double transmitTime )
```

Computes the ECEF SV coordinates and ECEF velocity.

**10.92.2.3 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 111 of file galileo\_ephemeris.h.

References *A\_1*, *af0\_4*, *af1\_4*, *af2\_4*, *BGD\_E1E5a\_5*, *BGD\_E1E5b\_5*, *C\_ic\_4*, *C\_is\_4*, *C\_rc\_3*, *C\_rs\_3*, *C\_uc\_3*, *C\_us\_3*, *delta\_n\_3*, *E1B\_DVS\_5*, *E1B\_HS\_5*, *E5a\_DVS*, *E5a\_HS*, *E5b\_DVS\_5*, *E5b\_HS\_5*, *e\_1*, *Galileo\_dtr*, *i\_0\_2*, *i\_satellite\_PRN*, *iDot\_2*, *M0\_1*, *OMEGA\_0\_2*, *omega\_2*, *OMEGA\_dot\_3*, *t0c\_4*, *t0e\_1*, *TOW\_5*, and *WN\_5*.

**10.92.2.4 sv\_clock\_drift()**

```
double Galileo_Ephemeris::sv_clock_drift (
    double transmitTime )
```

Satellite Time Correction Algorithm, ICD 5.1.4.

**10.92.2.5 sv\_clock\_relativistic\_term()**

```
double Galileo_Ephemeris::sv_clock_relativistic_term (
    double transmitTime )
```

Satellite Time Correction Algorithm, ICD 5.1.4.

### 10.92.3 Member Data Documentation

#### 10.92.3.1 A\_1

```
double Galileo_Ephemeris::A_1 {}
```

Square root of the semi-major axis [meters<sup>1/2</sup>].

Definition at line 54 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

#### 10.92.3.2 af0\_4

```
double Galileo_Ephemeris::af0_4 {}
```

SV clock bias correction coefficient [s].

Definition at line 70 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

#### 10.92.3.3 af1\_4

```
double Galileo_Ephemeris::af1_4 {}
```

SV clock drift correction coefficient [s/s].

Definition at line 71 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

#### 10.92.3.4 af2\_4

```
double Galileo_Ephemeris::af2_4 {}
```

SV clock drift rate correction coefficient [s/s<sup>2</sup>].

Definition at line 72 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

### 10.92.3.5 BGD\_E1E5a\_5

```
double Galileo_Ephemeris::BGD_E1E5a_5 {}
```

E1-E5a Broadcast Group Delay [s].

Definition at line 89 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

### 10.92.3.6 BGD\_E1E5b\_5

```
double Galileo_Ephemeris::BGD_E1E5b_5 {}
```

E1-E5b Broadcast Group Delay [s].

Definition at line 90 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

### 10.92.3.7 C\_ic\_4

```
double Galileo_Ephemeris::C_ic_4 {}
```

Amplitude of the cosine harmonic correction term to the angle of inclination [radians].

Definition at line 64 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

### 10.92.3.8 C\_is\_4

```
double Galileo_Ephemeris::C_is_4 {}
```

Amplitude of the sine harmonic correction term to the angle of inclination [radians].

Definition at line 65 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

#### 10.92.3.9 C\_rc\_3

```
double Galileo_Ephemeris::C_rc_3 {}
```

Amplitude of the cosine harmonic correction term to the orbit radius [meters].

Definition at line 62 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

#### 10.92.3.10 C\_rs\_3

```
double Galileo_Ephemeris::C_rs_3 {}
```

Amplitude of the sine harmonic correction term to the orbit radius [meters].

Definition at line 63 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

#### 10.92.3.11 C\_uc\_3

```
double Galileo_Ephemeris::C_uc_3 {}
```

Amplitude of the cosine harmonic correction term to the argument of latitude [radians].

Definition at line 60 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

#### 10.92.3.12 C\_us\_3

```
double Galileo_Ephemeris::C_us_3 {}
```

Amplitude of the sine harmonic correction term to the argument of latitude [radians].

Definition at line 61 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

### 10.92.3.13 d\_satpos\_X

```
double Galileo_Ephemeris::d_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 93 of file galileo\_ephemeris.h.

### 10.92.3.14 d\_satpos\_Y

```
double Galileo_Ephemeris::d_satpos_Y {}
```

Earth-fixed coordinate y of the satellite [m]. Completes a right-handed, Earth-Centered, Earth-Fixed orthogonal coordinate system.

Definition at line 94 of file galileo\_ephemeris.h.

### 10.92.3.15 d\_satpos\_Z

```
double Galileo_Ephemeris::d_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 95 of file galileo\_ephemeris.h.

### 10.92.3.16 d\_satvel\_X

```
double Galileo_Ephemeris::d_satvel_X {}
```

Earth-fixed velocity coordinate x of the satellite [m].

Definition at line 98 of file galileo\_ephemeris.h.

### 10.92.3.17 d\_satvel\_Y

```
double Galileo_Ephemeris::d_satvel_Y {}
```

Earth-fixed velocity coordinate y of the satellite [m].

Definition at line 99 of file galileo\_ephemeris.h.

#### 10.92.3.18 d\_satvel\_Z

```
double Galileo_Ephemeris::d_satvel_Z {}
```

Earth-fixed velocity coordinate z of the satellite [m].

Definition at line 100 of file galileo\_ephemeris.h.

#### 10.92.3.19 delta\_n\_3

```
double Galileo_Ephemeris::delta_n_3 {}
```

Mean motion difference from computed value [semi-circles/sec].

Definition at line 52 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

#### 10.92.3.20 E1B\_DVS\_5

```
bool Galileo_Ephemeris::E1B_DVS_5 {}
```

E1B Data Validity Status.

Definition at line 88 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

#### 10.92.3.21 E1B\_HS\_5

```
int32_t Galileo_Ephemeris::E1B_HS_5 {}
```

E1B Signal Health Status.

Definition at line 85 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

### 10.92.3.22 E5a\_DVS

```
bool Galileo_Ephemeris::E5a_DVS {}
```

E5a Data Validity Status.

Definition at line 86 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

### 10.92.3.23 E5a\_HS

```
int32_t Galileo_Ephemeris::E5a_HS {}
```

E5a Signal Health Status.

Definition at line 83 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

### 10.92.3.24 E5b\_DVS\_5

```
bool Galileo_Ephemeris::E5b_DVS_5 {}
```

E5b Data Validity Status.

Definition at line 87 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

### 10.92.3.25 E5b\_HS\_5

```
int32_t Galileo_Ephemeris::E5b_HS_5 {}
```

E5b Signal Health Status.

Definition at line 84 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

**10.92.3.26 e\_1**

```
double Galileo_Ephemeris::e_1 {}
```

Eccentricity.

Definition at line 53 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

**10.92.3.27 Galileo\_dtr**

```
double Galileo_Ephemeris::Galileo_dtr {}
```

relativistic clock correction term

Definition at line 79 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

**10.92.3.28 i\_0\_2**

```
double Galileo_Ephemeris::i_0_2 {}
```

Inclination angle at reference time [semi-circles].

Definition at line 56 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

**10.92.3.29 i\_satellite\_PRN**

```
uint32_t Galileo_Ephemeris::i_satellite_PRN {}
```

SV PRN NUMBER.

Definition at line 102 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

### 10.92.3.30 iDot\_2

```
double Galileo_Ephemeris::iDot_2 {}
```

Rate of inclination angle [semi-circles/sec].

Definition at line 59 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

### 10.92.3.31 M0\_1

```
double Galileo_Ephemeris::M0_1 {}
```

Mean anomaly at reference time [semi-circles].

Definition at line 51 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

### 10.92.3.32 OMEGA\_0\_2

```
double Galileo_Ephemeris::OMEGA_0_2 {}
```

Longitude of ascending node of orbital plane at weekly epoch [semi-circles].

Definition at line 55 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

### 10.92.3.33 omega\_2

```
double Galileo_Ephemeris::omega_2 {}
```

Argument of perigee [semi-circles].

Definition at line 57 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

**10.92.3.34 OMEGA\_dot\_3**

```
double Galileo_Ephemeris::OMEGA_dot_3 {}
```

Rate of right ascension [semi-circles/sec].

Definition at line 58 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

**10.92.3.35 t0c\_4**

```
int32_t Galileo_Ephemeris::t0c_4 {}
```

Clock correction data reference Time of Week [sec].

Definition at line 69 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

**10.92.3.36 t0e\_1**

```
int32_t Galileo_Ephemeris::t0e_1 {}
```

Ephemeris reference time [s].

Definition at line 66 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

**10.92.3.37 TOW\_5**

```
int32_t Galileo_Ephemeris::TOW_5 {}
```

Time of Week.

Definition at line 77 of file galileo\_ephemeris.h.

Referenced by `serialize()`.

## 10.92.3.38 WN\_5

```
int32_t Galileo_Ephemeris::WN_5 {}
```

Week number.

Definition at line 76 of file `galileo_ephemeris.h`.

Referenced by `serialize()`.

The documentation for this class was generated from the following file:

- [galileo\\_ephemeris.h](#)

## 10.93 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 1.2 (Nov 2015). See <https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo-OS-SIS-ICD.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\_TOW2** () const
- int32\_t **get\_TOW3** () 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)

### 10.93.1 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 1.2 (Nov 2015). See <https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo-OS-SIS-ICD.pdf>.

Definition at line 50 of file `galileo_fnav_message.h`.

The documentation for this class was generated from the following file:

- [galileo\\_fnav\\_message.h](#)

## 10.94 Galileo\_HAS\_data Class Reference

This class is a storage for Galileo HAS message type 1, as defined in Galileo High Accuracy Service E6-B Signal-In-Space Message Specification v1.2 (April 2020).

```
#include <galileo_has_data.h>
```

### Public Attributes

- [mt1\\_header](#) **header**
- `uint8_t` **Nsys**
- `std::vector< uint8_t >` **gnss\_id\_mask**
- `std::vector< uint64_t >` **satellite\_mask**
- `std::vector< uint16_t >` **signal\_mask**
- `std::vector< bool >` **cell\_mask\_availability\_flag**
- `std::vector< std::vector< std::vector< bool > > >` **cell\_mask**
- `std::vector< uint8_t >` **nav\_message**
- `uint8_t` **validity\_interval\_index\_orbit\_corrections**
- `std::vector< uint16_t >` **gnss\_iod**
- `std::vector< int16_t >` **delta\_radial**
- `std::vector< int16_t >` **delta\_along\_track**
- `std::vector< int16_t >` **delta\_cross\_track**
- `uint8_t` **validity\_interval\_index\_clock\_fullset\_corrections**
- `std::vector< uint8_t >` **delta\_clock\_c0\_multiplier**
- `std::vector< bool >` **iod\_change\_flag**
- `std::vector< int16_t >` **delta\_clock\_c0**
- `uint8_t` **validity\_interval\_index\_clock\_subset\_corrections**
- `uint8_t` **Nsysprime**
- `std::vector< uint8_t >` **gnss\_id\_clock\_subset**
- `std::vector< uint8_t >` **delta\_clock\_c0\_multiplier\_clock\_subset**
- `std::vector< std::vector< uint64_t > >` **satellite\_submask**
- `std::vector< bool >` **iod\_change\_flag\_clock\_subset**
- `std::vector< int16_t >` **delta\_clock\_c0\_clock\_subset**
- `uint8_t` **validity\_interval\_index\_code\_bias\_corrections**
- `std::vector< std::vector< int16_t > >` **code\_bias**
- `uint8_t` **validity\_interval\_index\_phase\_bias\_corrections**
- `std::vector< std::vector< int16_t > >` **phase\_bias**
- `std::vector< std::vector< uint8_t > >` **phase\_discontinuity\_indicator**
- `uint8_t` **validity\_interval\_index\_ura\_corrections**
- `std::vector< uint8_t >` **ura**

### 10.94.1 Detailed Description

This class is a storage for Galileo HAS message type 1, as defined in Galileo High Accuracy Service E6-B Signal-In-Space Message Specification v1.2 (April 2020).

Definition at line 48 of file `galileo_has_data.h`.

The documentation for this class was generated from the following file:

- [galileo\\_has\\_data.h](#)

## 10.95 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 1.2 (Nov 2015). See <https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo-OS-SIS-ICD.pdf>.

```
#include <galileo_inav_message.h>
```

### Public Member Functions

- void **split\_page** (std::string page\_string, int32\_t flag\_even\_word)
- int32\_t **page\_jk\_decoder** (const char \*data\_jk)
- 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
- bool **get\_flag\_CRC\_test** () 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)
- bool **get\_flag\_GGTO** () const
- double **get\_A0G** () const
- double **get\_A1G** () const
- double **get\_t0G** () const
- double **get\_WN0G** () const

### 10.95.1 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 1.2 (Nov 2015). See <https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo-OS-SIS-ICD.pdf>.

Definition at line 44 of file `galileo_inav_message.h`.

The documentation for this class was generated from the following file:

- [galileo\\_inav\\_message.h](#)

## 10.96 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\\_5](#) {}  
*Effective Ionisation Level 1st order parameter [sfu].*
- double [ai1\\_5](#) {}  
*Effective Ionisation Level 2st order parameter [sfu/degree].*
- double [ai2\\_5](#) {}  
*Effective Ionisation Level 3st order parameter [sfu/degree].*
- int32\_t [TOW\\_5](#) {}  
*UTC data reference Time of Week [s].*
- int32\_t [WN\\_5](#) {}  
*UTC data reference Week number [week].*
- bool [Region1\\_flag\\_5](#) {}  
*Ionospheric Disturbance Flag for region 1.*
- bool [Region2\\_flag\\_5](#) {}  
*Ionospheric Disturbance Flag for region 2.*
- bool [Region3\\_flag\\_5](#) {}  
*Ionospheric Disturbance Flag for region 3.*
- bool [Region4\\_flag\\_5](#) {}  
*Ionospheric Disturbance Flag for region 4.*
- bool [Region5\\_flag\\_5](#) {}  
*Ionospheric Disturbance Flag for region 5.*

### 10.96.1 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-OIS-SIS-ICD.pdf>

Definition at line 36 of file galileo\_iono.h.

### 10.96.2 Constructor & Destructor Documentation

#### 10.96.2.1 Galileo\_Iono()

```
Galileo_Iono::Galileo_Iono ( ) [default]
```

Default constructor

### 10.96.3 Member Function Documentation

#### 10.96.3.1 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 66 of file galileo\_iono.h.

References `ai0_5`, `ai1_5`, `ai2_5`, `Region1_flag_5`, `Region2_flag_5`, `Region3_flag_5`, `Region4_flag_5`, `Region5_flag_5`, `TOW_5`, and `WN_5`.

### 10.96.4 Member Data Documentation

#### 10.96.4.1 ai0\_5

```
double Galileo_Iono::ai0_5 {}
```

Effective Ionisation Level 1st order parameter [sfu].

Definition at line 45 of file galileo\_iono.h.

Referenced by `serialize()`.

#### 10.96.4.2 ai1\_5

```
double Galileo_Iono::ai1_5 {}
```

Effective Ionisation Level 2st order parameter [sfu/degree].

Definition at line 46 of file galileo\_iono.h.

Referenced by `serialize()`.

#### 10.96.4.3 ai2\_5

```
double Galileo_Iono::ai2_5 {}
```

Effective Ionisation Level 3st order parameter [sfu/degree].

Definition at line 47 of file galileo\_iono.h.

Referenced by `serialize()`.

#### 10.96.4.4 Region1\_flag\_5

```
bool Galileo_Iono::Region1_flag_5 {}
```

Ionospheric Disturbance Flag for region 1.

Definition at line 54 of file galileo\_iono.h.

Referenced by `serialize()`.

#### 10.96.4.5 Region2\_flag\_5

```
bool Galileo_Iono::Region2_flag_5 {}
```

Ionospheric Disturbance Flag for region 2.

Definition at line 55 of file galileo\_iono.h.

Referenced by `serialize()`.

#### 10.96.4.6 Region3\_flag\_5

```
bool Galileo_Iono::Region3_flag_5 {}
```

Ionospheric Disturbance Flag for region 3.

Definition at line 56 of file galileo\_iono.h.

Referenced by `serialize()`.

#### 10.96.4.7 Region4\_flag\_5

```
bool Galileo_Iono::Region4_flag_5 {}
```

Ionospheric Disturbance Flag for region 4.

Definition at line 57 of file galileo\_iono.h.

Referenced by `serialize()`.

#### 10.96.4.8 Region5\_flag\_5

```
bool Galileo_Iono::Region5_flag_5 {}
```

Ionospheric Disturbance Flag for region 5.

Definition at line 58 of file galileo\_iono.h.

Referenced by `serialize()`.

#### 10.96.4.9 TOW\_5

```
int32_t Galileo_Iono::TOW_5 {}
```

UTC data reference Time of Week [s].

Definition at line 50 of file galileo\_iono.h.

Referenced by `serialize()`.

#### 10.96.4.10 WN\_5

```
int32_t Galileo_Iono::WN_5 {}
```

UTC data reference Week number [week].

Definition at line 51 of file galileo\_iono.h.

Referenced by `serialize()`.

The documentation for this class was generated from the following file:

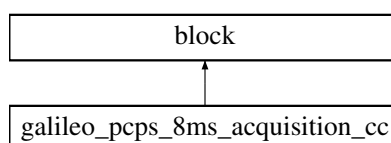
- [galileo\\_iono.h](#)

## 10.97 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)  
*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](#) (std::complex< float > \*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\\_state](#) ([uint32\\_t](#) state)  
*If set to 1, ensures that acquisition starts at the first available sample.*
- 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\\_threshold](#) (float threshold)  
*Set statistics threshold of PCPS algorithm.*
- void [set\\_doppler\\_max](#) ([uint32\\_t](#) doppler\_max)  
*Set maximum Doppler grid search.*
- void [set\\_doppler\\_step](#) ([uint32\\_t](#) doppler\_step)  
*Set Doppler steps for the grid search.*
- int [general\\_work](#) (int noutput\_items, gr\_vector\_int &ninput\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)  
*Parallel Code Phase Search Acquisition signal processing.*

## Friends

- [galileo\\_pcps\\_8ms\\_acquisition\\_cc\\_sptr](#) [galileo\\_pcps\\_8ms\\_make\\_acquisition\\_cc](#) ([uint32\\_t](#) sampled\_ms, [uint32\\_t](#) max\_dwells, [uint32\\_t](#) doppler\_max, [int64\\_t](#) fs\_in, [int32\\_t](#) samples\_per\_ms, [int32\\_t](#) samples\_per\_code, bool dump, const std::string &dump\_filename, bool enable\_monitor\_output)

### 10.97.1 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 57 of file `galileo_pcps_8ms_acquisition_cc.h`.

### 10.97.2 Constructor & Destructor Documentation

### 10.97.2.1 `~galileo_pcps_8ms_acquisition_cc()`

```
galileo_pcps_8ms_acquisition_cc::~~galileo_pcps_8ms_acquisition_cc ( )
```

Default destructor.

## 10.97.3 Member Function Documentation

### 10.97.3.1 `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 )
```

Parallel Code Phase Search Acquisition signal processing.

### 10.97.3.2 `init()`

```
void galileo_pcps_8ms_acquisition_cc::init ( )
```

Initializes acquisition algorithm.

### 10.97.3.3 `mag()`

```
uint32_t galileo_pcps_8ms_acquisition_cc::mag ( ) const [inline]
```

Returns the maximum peak of grid search.

Definition at line 78 of file `galileo_pcps_8ms_acquisition_cc.h`.

### 10.97.3.4 `set_active()`

```
void galileo_pcps_8ms_acquisition_cc::set_active (
    bool active ) [inline]
```

Starts acquisition algorithm, turning from standby mode to active mode.

## Parameters

<i>active</i>	- bool that activates/deactivates the block.
---------------	--

Definition at line 99 of file galileo\_pcps\_8ms\_acquisition\_cc.h.

### 10.97.3.5 set\_channel()

```
void galileo_pcps_8ms_acquisition_cc::set_channel (
    uint32_t channel ) [inline]
```

Set acquisition channel unique ID.

## Parameters

<i>channel</i>	- receiver channel.
----------------	---------------------

Definition at line 115 of file galileo\_pcps\_8ms\_acquisition\_cc.h.

### 10.97.3.6 set\_channel\_fsm()

```
void galileo_pcps_8ms_acquisition_cc::set_channel_fsm (
    std::weak_ptr< ChannelFsm > channel_fsm ) [inline]
```

Set channel fsm associated to this acquisition instance.

Definition at line 123 of file galileo\_pcps\_8ms\_acquisition\_cc.h.

### 10.97.3.7 set\_doppler\_max()

```
void galileo_pcps_8ms_acquisition_cc::set_doppler_max (
    uint32_t doppler_max ) [inline]
```

Set maximum Doppler grid search.

## Parameters

<i>doppler_max</i>	- Maximum Doppler shift considered in the grid search [Hz].
--------------------	---

Definition at line 142 of file galileo\_pcps\_8ms\_acquisition\_cc.h.

**10.97.3.8 set\_doppler\_step()**

```
void galileo_pcps_8ms_acquisition_cc::set_doppler_step (
    uint32_t doppler_step ) [inline]
```

Set Doppler steps for the grid search.

**Parameters**

<i>doppler_step</i>	- Frequency bin of the search grid [Hz].
---------------------	--

Definition at line 151 of file galileo\_pcps\_8ms\_acquisition\_cc.h.

**10.97.3.9 set\_gnss\_synchro()**

```
void galileo_pcps_8ms_acquisition_cc::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**

<i>p_gnss_synchro</i>	Satellite information shared by the processing blocks.
-----------------------	--

Definition at line 70 of file galileo\_pcps\_8ms\_acquisition\_cc.h.

**10.97.3.10 set\_local\_code()**

```
void galileo_pcps_8ms_acquisition_cc::set_local_code (
    std::complex< float > * code )
```

Sets local code for PCPS acquisition algorithm.

**Parameters**

<i>code</i>	- Pointer to the PRN code.
-------------	----------------------------

**10.97.3.11 set\_state()**

```
void galileo_pcps_8ms_acquisition_cc::set_state (
    int32_t state )
```

If set to 1, ensures that acquisition starts at the first available sample.

## Parameters

<i>state</i>	- int=1 forces start of acquisition
--------------	-------------------------------------

10.97.3.12 `set_threshold()`

```
void galileo_pcps_8ms_acquisition_cc::set_threshold (
    float threshold ) [inline]
```

Set statistics threshold of PCPS algorithm.

## Parameters

<i>threshold</i>	- Threshold for signal detection (check <a href="#">Navitec2012</a> , Algorithm 1, for a definition of this threshold).
------------------	---

Definition at line 133 of file `galileo_pcps_8ms_acquisition_cc.h`.

The documentation for this class was generated from the following file:

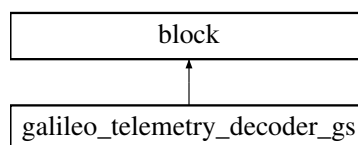
- [galileo\\_pcps\\_8ms\\_acquisition\\_cc.h](#)

10.98 `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)  
*Set satellite PRN.*
- void `set_channel` (int32\_t channel)  
*Set receiver's channel.*
- void `reset` ()
- int `general_work` (int noutput\_items, gr\_vector\_int &ninput\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)  
*This is where all signal processing takes place.*

## Public Attributes

- `int32_t flag_even_word_arrived`

## Friends

- `galileo_telemetry_decoder_gs_sptr galileo_make_telemetry_decoder_gs` (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf, int frame\_type)

### 10.98.1 Detailed Description

This class implements a block that decodes the INAV and FNAV data defined in Galileo ICD.

Definition at line 55 of file `galileo_telemetry_decoder_gs.h`.

### 10.98.2 Member Function Documentation

#### 10.98.2.1 `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 )
```

This is where all signal processing takes place.

#### 10.98.2.2 `set_channel()`

```
void galileo_telemetry_decoder_gs::set_channel (
    int32_t channel )
```

Set receiver's channel.

#### 10.98.2.3 `set_satellite()`

```
void galileo_telemetry_decoder_gs::set_satellite (
    const Gnss\_Satellite & satellite )
```

Set satellite PRN.

The documentation for this class was generated from the following file:

- [galileo\\_telemetry\\_decoder\\_gs.h](#)

## 10.99 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.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.*

### Public Attributes

- double **A0\_6** {}
- double **A1\_6** {}
- int32\_t **Delta\_tLS\_6** {}
- int32\_t [t0t\\_6](#) {}  
*UTC data reference Time of Week [s].*
- int32\_t [WNNot\\_6](#) {}  
*UTC data reference Week number [week].*
- int32\_t **WN\_LSF\_6** {}
- int32\_t **DN\_6** {}
- int32\_t **Delta\_tLSF\_6** {}
- double **A\_0G\_10** {}
- double **A\_1G\_10** {}
- int32\_t **t\_0G\_10** {}
- int32\_t **WN\_0G\_10** {}
- bool **flag\_utc\_model** {}

#### 10.99.1 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.pdf> paragraph 5.1.7.

Definition at line 36 of file galileo\_utc\_model.h.

#### 10.99.2 Constructor & Destructor Documentation

### 10.99.2.1 Galileo\_Utc\_Model()

```
Galileo_Utc_Model::Galileo_Utc_Model ( ) [default]
```

Default constructor

## 10.99.3 Member Function Documentation

### 10.99.3.1 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.

### 10.99.3.2 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 71 of file galileo\_utc\_model.h.

References t0t\_6, and WNot\_6.

## 10.99.4 Member Data Documentation

### 10.99.4.1 t0t\_6

```
int32_t Galileo_Utc_Model::t0t_6 {}
```

UTC data reference Time of Week [s].

Definition at line 51 of file galileo\_utc\_model.h.

Referenced by serialize().

### 10.99.4.2 WNot\_6

```
int32_t Galileo_Utc_Model::WNot_6 {}
```

UTC data reference Week number [week].

Definition at line 52 of file galileo\_utc\_model.h.

Referenced by `serialize()`.

The documentation for this class was generated from the following file:

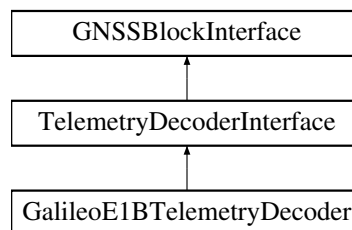
- [galileo\\_utc\\_model.h](#)

## 10.100 GalileoE1BTelemetryDecoder Class Reference

This class implements a NAV data decoder for Galileo INAV frames in E1B radio link.

```
#include <galileo_e1b_telemetry_decoder.h>
```

Inheritance diagram for GalileoE1BTelemetryDecoder:



### Public Member Functions

- **GalileoE1BTelemetryDecoder** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- 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
  - Returns "Galileo\_E1B\_Telemetry\_Decoder".*
- void **set\_channel** (int channel) override
- void **reset** () override
- size\_t **item\_size** () override

### 10.100.1 Detailed Description

This class implements a NAV data decoder for Galileo INAV frames in E1B radio link.

Definition at line 44 of file galileo\_e1b\_telemetry\_decoder.h.

## 10.100.2 Member Function Documentation

### 10.100.2.1 implementation()

```
std::string GalileoE1BTelemetryDecoder::implementation ( ) [inline], [override], [virtual]
```

Returns "Galileo\_E1B\_Telemetry\_Decoder".

Implements [GNSSBlockInterface](#).

Definition at line 70 of file galileo\_e1b\_telemetry\_decoder.h.

The documentation for this class was generated from the following file:

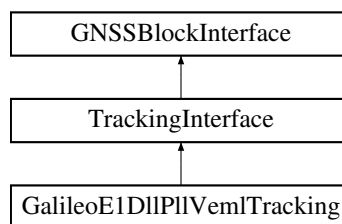
- [galileo\\_e1b\\_telemetry\\_decoder.h](#)

## 10.101 GalileoE1DIPIIVemlTracking Class Reference

This class Adapts a DLL+PLL VEML (Very Early Minus Late) tracking loop block to a [TrackingInterface](#) for Galileo E1 signals.

```
#include <galileo_e1_dll_pll_veml_tracking.h>
```

Inheritance diagram for GalileoE1DIPIIVemlTracking:



### Public Member Functions

- **GalileoE1DIPIIVemlTracking** (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\_DLL\_PLL\_VEML\_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.

### 10.101.1 Detailed Description

This class Adapts a DLL+PLL VEML (Very Early Minus Late) tracking loop block to a [TrackingInterface](#) for Galileo E1 signals.

Definition at line 42 of file `galileo_e1_dll_pll_veml_tracking.h`.

### 10.101.2 Member Function Documentation

#### 10.101.2.1 `implementation()`

```
std::string GalileoE1DllPllVemlTracking::implementation ( ) [inline], [override], [virtual]
```

Returns "Galileo\_E1\_DLL\_PLL\_VEML\_Tracking".

Implements [GNSSBlockInterface](#).

Definition at line 59 of file `galileo_e1_dll_pll_veml_tracking.h`.

#### 10.101.2.2 `set_channel()`

```
void GalileoE1DllPllVemlTracking::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.101.2.3 `set_gnss_synchro()`

```
void GalileoE1DllPllVemlTracking::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](#).

## 10.101.2.4 stop\_tracking()

```
void GalileoE1DllPllVemlTracking::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

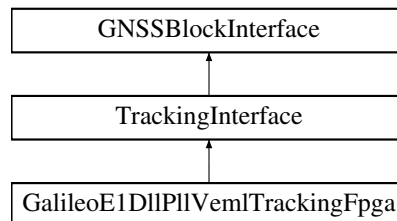
- [galileo\\_e1\\_dll\\_pll\\_veml\\_tracking.h](#)

## 10.102 GalileoE1DlIPllVemlTrackingFpga Class Reference

This class Adapts a DLL+PLL VEML (Very Early Minus Late) tracking loop block to a [TrackingInterface](#) for Galileo E1 signals.

```
#include <galileo_e1_dll_pll_veml_tracking_fpga.h>
```

Inheritance diagram for GalileoE1DlIPllVemlTrackingFpga:



## Public Member Functions

- [GalileoE1DlIPllVemlTrackingFpga](#) (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)  
*Constructor.*
- virtual [~GalileoE1DlIPllVemlTrackingFpga](#) ()  
*Destructor.*
- std::string [role](#) () override  
*Role.*
- std::string [implementation](#) () override  
*Returns "Galileo\_E1\_DLL\_PLL\_VEML\_Tracking\_Fpga".*
- size\_t [item\\_size](#) () override  
*Returns size of lv\_16sc\_t.*
- void [connect](#) (gr::top\_block\_sptr top\_block) override  
*Connect.*
- void [disconnect](#) (gr::top\_block\_sptr top\_block) override  
*Disconnect.*
- gr::basic\_block\_sptr [get\\_left\\_block](#) () override  
*Get left block.*
- gr::basic\_block\_sptr [get\\_right\\_block](#) () override  
*Get right block.*

- 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  
*Start the tracking process in the FPGA.*
- void [stop\\_tracking](#) () override  
*Stop the tracking process in the FPGA.*

### 10.102.1 Detailed Description

This class Adapts a DLL+PLL VEML (Very Early Minus Late) tracking loop block to a [TrackingInterface](#) for Galileo E1 signals.

Definition at line 42 of file `galileo_e1_dll_pll_veml_tracking_fpga.h`.

### 10.102.2 Constructor & Destructor Documentation

#### 10.102.2.1 GalileoE1DllPllVemlTrackingFpga()

```
GalileoE1DllPllVemlTrackingFpga::GalileoE1DllPllVemlTrackingFpga (
    const ConfigurationInterface * configuration,
    const std::string & role,
    unsigned int in_streams,
    unsigned int out_streams )
```

Constructor.

#### 10.102.2.2 ~GalileoE1DllPllVemlTrackingFpga()

```
virtual GalileoE1DllPllVemlTrackingFpga::~GalileoE1DllPllVemlTrackingFpga ( ) [virtual]
```

Destructor.

### 10.102.3 Member Function Documentation

### 10.102.3.1 connect()

```
void GalileoE1DllPllVemlTrackingFpga::connect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Connect.

Implements [GNSSBlockInterface](#).

### 10.102.3.2 disconnect()

```
void GalileoE1DllPllVemlTrackingFpga::disconnect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Disconnect.

Implements [GNSSBlockInterface](#).

### 10.102.3.3 get\_left\_block()

```
gr::basic_block_sptr GalileoE1DllPllVemlTrackingFpga::get_left_block ( ) [override], [virtual]
```

Get left block.

Implements [GNSSBlockInterface](#).

### 10.102.3.4 get\_right\_block()

```
gr::basic_block_sptr GalileoE1DllPllVemlTrackingFpga::get_right_block ( ) [override], [virtual]
```

Get right block.

Implements [GNSSBlockInterface](#).

### 10.102.3.5 implementation()

```
std::string GalileoE1DllPllVemlTrackingFpga::implementation ( ) [inline], [override], [virtual]
```

Returns "Galileo\_E1\_DLL\_PLL\_VEML\_Tracking\_Fpga".

Implements [GNSSBlockInterface](#).

Definition at line 70 of file `galileo_e1_dll_pll_veml_tracking_fpga.h`.

#### 10.102.3.6 item\_size()

```
size_t GalileoE1D1lP1lVemlTrackingFpga::item_size ( ) [inline], [override], [virtual]
```

Returns size of lv\_16sc\_t.

Implements [GNSSBlockInterface](#).

Definition at line 78 of file galileo\_e1\_dll\_pll\_veml\_tracking\_fpga.h.

#### 10.102.3.7 role()

```
std::string GalileoE1D1lP1lVemlTrackingFpga::role ( ) [inline], [override], [virtual]
```

Role.

Implements [GNSSBlockInterface](#).

Definition at line 62 of file galileo\_e1\_dll\_pll\_veml\_tracking\_fpga.h.

#### 10.102.3.8 set\_channel()

```
void GalileoE1D1lP1lVemlTrackingFpga::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.102.3.9 set\_gnss\_synchro()

```
void GalileoE1D1lP1lVemlTrackingFpga::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](#).

### 10.102.3.10 start\_tracking()

```
void GalileoE1DllPllVemlTrackingFpga::start_tracking ( ) [override], [virtual]
```

Start the tracking process in the FPGA.

Implements [TrackingInterface](#).

### 10.102.3.11 stop\_tracking()

```
void GalileoE1DllPllVemlTrackingFpga::stop_tracking ( ) [override], [virtual]
```

Stop the tracking process in the FPGA.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

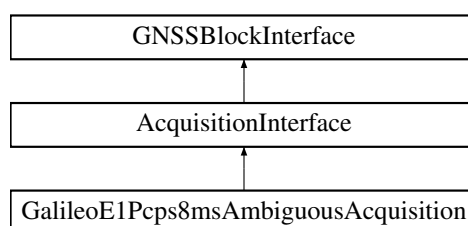
- [galileo\\_e1\\_dll\\_pll\\_veml\\_tracking\\_fpga.h](#)

## 10.103 GalileoE1Pcps8msAmbiguousAcquisition Class Reference

Adapts a PCPS 8ms acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

```
#include <galileo_e1_pcps_8ms_ambiguous_acquisition.h>
```

Inheritance diagram for GalileoE1Pcps8msAmbiguousAcquisition:



## Public Member Functions

- **GalileoE1Pcps8msAmbiguousAcquisition** (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\_PCPS\_8ms\_Ambiguous\_Acquisition".*
- 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\_threshold** (float threshold) override
  - Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override
  - Set maximum Doppler off grid search.*
- void **set\_doppler\_step** (unsigned int doppler\_step) override
  - Set Doppler steps for the grid search.*
- void **init** () override
  - Initializes acquisition algorithm.*
- void **set\_local\_code** () override
  - Sets local code for Galileo E1 PCPS acquisition algorithm.*
- 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\_state** (int state \_\_attribute\_\_((unused))) override
- void **set\_resampler\_latency** (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override

### 10.103.1 Detailed Description

Adapts a PCPS 8ms acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

Definition at line 41 of file `galileo_e1_pcps_8ms_ambiguous_acquisition.h`.

### 10.103.2 Member Function Documentation

### 10.103.2.1 implementation()

```
std::string GalileoE1Pcps8msAmbiguousAcquisition::implementation ( ) [inline], [override],  
[virtual]
```

Returns "Galileo\_E1\_PCPS\_8ms\_Ambiguous\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 59 of file `galileo_e1_pcps_8ms_ambiguous_acquisition.h`.

### 10.103.2.2 init()

```
void GalileoE1Pcps8msAmbiguousAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.103.2.3 mag()

```
signed int GalileoE1Pcps8msAmbiguousAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

### 10.103.2.4 reset()

```
void GalileoE1Pcps8msAmbiguousAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.103.2.5 set\_channel()

```
void GalileoE1Pcps8msAmbiguousAcquisition::set_channel (  
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 84 of file `galileo_e1_pcps_8ms_ambiguous_acquisition.h`.

#### 10.103.2.6 set\_channel\_fsm()

```
void GalileoE1Pcps8msAmbiguousAcquisition::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 93 of file galileo\_e1\_pcps\_8ms\_ambiguous\_acquisition.h.

#### 10.103.2.7 set\_doppler\_max()

```
void GalileoE1Pcps8msAmbiguousAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.103.2.8 set\_doppler\_step()

```
void GalileoE1Pcps8msAmbiguousAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.103.2.9 set\_gnss\_synchro()

```
void GalileoE1Pcps8msAmbiguousAcquisition::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](#).

**10.103.2.10 set\_local\_code()**

```
void GalileoE1Pcps8msAmbiguousAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local code for Galileo E1 PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

**10.103.2.11 set\_threshold()**

```
void GalileoE1Pcps8msAmbiguousAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

**10.103.2.12 stop\_acquisition()**

```
void GalileoE1Pcps8msAmbiguousAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

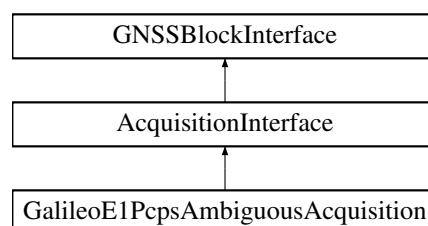
- [galileo\\_e1\\_pcps\\_8ms\\_ambiguous\\_acquisition.h](#)

**10.104 GalileoE1PcpsAmbiguousAcquisition Class Reference**

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

```
#include <galileo_e1_pcps_ambiguous_acquisition.h>
```

Inheritance diagram for GalileoE1PcpsAmbiguousAcquisition:



## Public Member Functions

- **GalileoE1PcpsAmbiguousAcquisition** (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\_PCPS\_Ambiguous\_Acquisition".*
- 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\_threshold** (float threshold) override
  - Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override
  - Set maximum Doppler off grid search.*
- void **set\_doppler\_step** (unsigned int doppler\_step) override
  - Set Doppler steps for the grid search.*
- void **set\_doppler\_center** (int doppler\_center) override
  - Set Doppler center for the grid search.*
- void **init** () override
  - Initializes acquisition algorithm.*
- void **set\_local\_code** () override
  - Sets local code for Galileo E1 PCPS acquisition algorithm.*
- signed int **mag** () override
  - Returns the maximum peak of grid search.*
- void **reset** () override
  - Restart acquisition algorithm.*
- void **set\_state** (int state) override
  - If state = 1, it forces the block to start acquiring from the first sample.*
- 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.*

### 10.104.1 Detailed Description

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

Definition at line 43 of file `galileo_e1_pcps_ambiguous_acquisition.h`.

### 10.104.2 Member Function Documentation

### 10.104.2.1 implementation()

```
std::string GalileoE1PcpsAmbiguousAcquisition::implementation ( ) [inline], [override], [virtual]
```

Returns "Galileo\_E1\_PCPS\_Ambiguous\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 62 of file `galileo_e1_pcps_ambiguous_acquisition.h`.

### 10.104.2.2 init()

```
void GalileoE1PcpsAmbiguousAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.104.2.3 mag()

```
signed int GalileoE1PcpsAmbiguousAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

### 10.104.2.4 reset()

```
void GalileoE1PcpsAmbiguousAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.104.2.5 set\_channel()

```
void GalileoE1PcpsAmbiguousAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 87 of file `galileo_e1_pcps_ambiguous_acquisition.h`.

#### 10.104.2.6 set\_channel\_fsm()

```
void GalileoElPcpsAmbiguousAcquisition::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 96 of file `galileo_e1_pcps_ambiguous_acquisition.h`.

#### 10.104.2.7 set\_doppler\_center()

```
void GalileoElPcpsAmbiguousAcquisition::set_doppler_center (
    int doppler_center ) [override]
```

Set Doppler center for the grid search.

#### 10.104.2.8 set\_doppler\_max()

```
void GalileoElPcpsAmbiguousAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.104.2.9 set\_doppler\_step()

```
void GalileoElPcpsAmbiguousAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.104.2.10 set\_gnss\_synchro()

```
void GalileoElPcpsAmbiguousAcquisition::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](#).

**10.104.2.11 set\_local\_code()**

```
void GalileoE1PcpsAmbiguousAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local code for Galileo E1 PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

**10.104.2.12 set\_resampler\_latency()**

```
void GalileoE1PcpsAmbiguousAcquisition::set_resampler_latency (
    uint32_t latency_samples ) [override], [virtual]
```

Sets the resampler latency to account it in the acquisition code delay estimation.

Implements [AcquisitionInterface](#).

**10.104.2.13 set\_state()**

```
void GalileoE1PcpsAmbiguousAcquisition::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

**10.104.2.14 set\_threshold()**

```
void GalileoE1PcpsAmbiguousAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

**10.104.2.15 stop\_acquisition()**

```
void GalileoE1PcpsAmbiguousAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

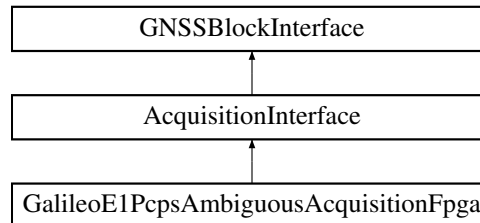
- [galileo\\_e1\\_pcps\\_ambiguous\\_acquisition.h](#)

## 10.105 GalileoE1PcpsAmbiguousAcquisitionFpga Class Reference

This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for Galileo E1 Signals.

```
#include <galileo_e1_pcps_ambiguous_acquisition_fpga.h>
```

Inheritance diagram for GalileoE1PcpsAmbiguousAcquisitionFpga:



### Public Member Functions

- [GalileoE1PcpsAmbiguousAcquisitionFpga](#) (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)  
*Constructor.*
- [~GalileoE1PcpsAmbiguousAcquisitionFpga](#) ()=default  
*Destructor.*
- std::string [role](#) () override  
*Role.*
- std::string [implementation](#) () override  
*Returns "Galileo\_E1\_PCPS\_Ambiguous\_Acquisition\_Fpga".*
- size\_t [item\\_size](#) () override  
*Returns size of lv\_16sc\_t.*
- void [connect](#) (gr::top\_block\_sptr top\_block) override  
*Connect.*
- void [disconnect](#) (gr::top\_block\_sptr top\_block) override  
*Disconnect.*
- gr::basic\_block\_sptr [get\\_left\\_block](#) () override  
*Get left block.*
- gr::basic\_block\_sptr [get\\_right\\_block](#) () override  
*Get right block.*
- 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\\_threshold](#) (float threshold) override  
*Set statistics threshold of PCPS algorithm.*
- void [set\\_doppler\\_max](#) (unsigned int doppler\_max) override  
*Set maximum Doppler off grid search.*
- void [set\\_doppler\\_step](#) (unsigned int doppler\_step) override  
*Set Doppler steps for the grid search.*

- void [set\\_doppler\\_center](#) (int doppler\_center) override  
*Set Doppler center for the grid search.*
- void [init](#) () override  
*Initializes acquisition algorithm.*
- void [set\\_local\\_code](#) () override  
*Sets local code for Galileo E1 PCPS acquisition algorithm.*
- signed int [mag](#) () override  
*Returns the maximum peak of grid search.*
- void [reset](#) () override  
*Restart acquisition algorithm.*
- void [set\\_state](#) (int state) override  
*If state = 1, it forces the block to start acquiring from the first sample.*
- void [stop\\_acquisition](#) () override  
*Stop running acquisition.*
- void [set\\_resampler\\_latency](#) (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override  
*Set resampler latency.*

### 10.105.1 Detailed Description

This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for Galileo E1 Signals.

Definition at line 41 of file `galileo_e1_pcps_ambiguous_acquisition_fpga.h`.

### 10.105.2 Constructor & Destructor Documentation

#### 10.105.2.1 GalileoE1PcpsAmbiguousAcquisitionFpga()

```
GalileoE1PcpsAmbiguousAcquisitionFpga::GalileoE1PcpsAmbiguousAcquisitionFpga (
    const ConfigurationInterface * configuration,
    const std::string & role,
    unsigned int in_streams,
    unsigned int out_streams )
```

Constructor.

#### 10.105.2.2 ~GalileoE1PcpsAmbiguousAcquisitionFpga()

```
GalileoE1PcpsAmbiguousAcquisitionFpga::~GalileoE1PcpsAmbiguousAcquisitionFpga ( ) [default]
```

Destructor.

### 10.105.3 Member Function Documentation

#### 10.105.3.1 connect()

```
void GalileoElPcpsAmbiguousAcquisitionFpga::connect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Connect.

Implements [GNSSBlockInterface](#).

#### 10.105.3.2 disconnect()

```
void GalileoElPcpsAmbiguousAcquisitionFpga::disconnect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Disconnect.

Implements [GNSSBlockInterface](#).

#### 10.105.3.3 get\_left\_block()

```
gr::basic_block_sptr GalileoElPcpsAmbiguousAcquisitionFpga::get_left_block ( ) [override],
[virtual]
```

Get left block.

Implements [GNSSBlockInterface](#).

#### 10.105.3.4 get\_right\_block()

```
gr::basic_block_sptr GalileoElPcpsAmbiguousAcquisitionFpga::get_right_block ( ) [override],
[virtual]
```

Get right block.

Implements [GNSSBlockInterface](#).

#### 10.105.3.5 implementation()

```
std::string GalileoElPcpsAmbiguousAcquisitionFpga::implementation ( ) [inline], [override],
[virtual]
```

Returns "Galileo\_E1\_PCPS\_Ambiguous\_Acquisition\_Fpga".

Implements [GNSSBlockInterface](#).

Definition at line 69 of file `galileo_e1_pcps_ambiguous_acquisition_fpga.h`.

### 10.105.3.6 init()

```
void GalileoE1PcpsAmbiguousAcquisitionFpga::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.105.3.7 item\_size()

```
size_t GalileoE1PcpsAmbiguousAcquisitionFpga::item_size ( ) [inline], [override], [virtual]
```

Returns size of lv\_16sc\_t.

Implements [GNSSBlockInterface](#).

Definition at line 77 of file galileo\_e1\_pcps\_ambiguous\_acquisition\_fpga.h.

### 10.105.3.8 mag()

```
signed int GalileoE1PcpsAmbiguousAcquisitionFpga::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

### 10.105.3.9 reset()

```
void GalileoE1PcpsAmbiguousAcquisitionFpga::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.105.3.10 role()

```
std::string GalileoE1PcpsAmbiguousAcquisitionFpga::role ( ) [inline], [override], [virtual]
```

Role.

Implements [GNSSBlockInterface](#).

Definition at line 61 of file galileo\_e1\_pcps\_ambiguous\_acquisition\_fpga.h.

#### 10.105.3.11 `set_channel()`

```
void GalileoElPcpsAmbiguousAcquisitionFpga::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 112 of file `galileo_e1_pcps_ambiguous_acquisition_fpga.h`.

#### 10.105.3.12 `set_channel_fsm()`

```
void GalileoElPcpsAmbiguousAcquisitionFpga::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 121 of file `galileo_e1_pcps_ambiguous_acquisition_fpga.h`.

#### 10.105.3.13 `set_doppler_center()`

```
void GalileoElPcpsAmbiguousAcquisitionFpga::set_doppler_center (
    int doppler_center ) [override]
```

Set Doppler center for the grid search.

#### 10.105.3.14 `set_doppler_max()`

```
void GalileoElPcpsAmbiguousAcquisitionFpga::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.105.3.15 `set_doppler_step()`

```
void GalileoElPcpsAmbiguousAcquisitionFpga::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

**10.105.3.16 set\_gnss\_synchro()**

```
void GalileoE1PcpsAmbiguousAcquisitionFpga::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](#).

**10.105.3.17 set\_local\_code()**

```
void GalileoE1PcpsAmbiguousAcquisitionFpga::set_local_code ( ) [override], [virtual]
```

Sets local code for Galileo E1 PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

**10.105.3.18 set\_resampler\_latency()**

```
void GalileoE1PcpsAmbiguousAcquisitionFpga::set_resampler_latency (
    uint32_t latency_samples __attribute__((unused)) ) [inline], [override]
```

Set resampler latency.

Definition at line 180 of file `galileo_e1_pcps_ambiguous_acquisition_fpga.h`.

**10.105.3.19 set\_state()**

```
void GalileoE1PcpsAmbiguousAcquisitionFpga::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

**10.105.3.20 set\_threshold()**

```
void GalileoE1PcpsAmbiguousAcquisitionFpga::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

### 10.105.3.21 stop\_acquisition()

```
void GalileoE1PcpsAmbiguousAcquisitionFpga::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

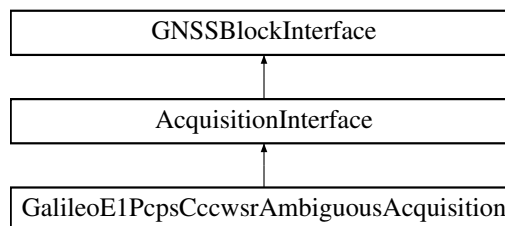
- [galileo\\_e1\\_pcps\\_ambiguous\\_acquisition\\_fpga.h](#)

## 10.106 GalileoE1PcpsCccwsrAmbiguousAcquisition Class Reference

Adapts a PCPS CCCWSR acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

```
#include <galileo_e1_pcps_cccwsr_ambiguous_acquisition.h>
```

Inheritance diagram for GalileoE1PcpsCccwsrAmbiguousAcquisition:



### Public Member Functions

- **GalileoE1PcpsCccwsrAmbiguousAcquisition** (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\_PCPS\_CCCWSR\_Ambiguous\_Acquisition".*
- 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\_threshold** (float threshold) override
 

*Set statistics threshold of CCCWSR algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override

- Set maximum Doppler off grid search.*
- void [set\\_doppler\\_step](#) (unsigned int doppler\_step) override
- Set Doppler steps for the grid search.*
- void [init](#) () override
- Initializes acquisition algorithm.*
- void [set\\_local\\_code](#) () override
- signed int [mag](#) () override
- Returns the maximum peak of grid search.*
- void [reset](#) () override
- Restart acquisition algorithm.*
- void [set\\_state](#) (int state) override
- If state = 1, it forces the block to start acquiring from the first sample.*
- void [stop\\_acquisition](#) () override
- Stop running acquisition.*
- void [set\\_resampler\\_latency](#) (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override

### 10.106.1 Detailed Description

Adapts a PCPS CCCWSR acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

Definition at line 41 of file `galileo_e1_pcps_cccwsr_ambiguous_acquisition.h`.

### 10.106.2 Member Function Documentation

#### 10.106.2.1 implementation()

```
std::string GalileoE1PcpsCccwsrAmbiguousAcquisition::implementation ( ) [inline], [override], [virtual]
```

Returns "Galileo\_E1\_PCPS\_CCCWSR\_Ambiguous\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 60 of file `galileo_e1_pcps_cccwsr_ambiguous_acquisition.h`.

#### 10.106.2.2 init()

```
void GalileoE1PcpsCccwsrAmbiguousAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.106.2.3 mag()

```
signed int GalileoElPcpsCccwsrAmbiguousAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

### 10.106.2.4 reset()

```
void GalileoElPcpsCccwsrAmbiguousAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.106.2.5 set\_channel()

```
void GalileoElPcpsCccwsrAmbiguousAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 85 of file `galileo_e1_pcps_cccwsr_ambiguous_acquisition.h`.

### 10.106.2.6 set\_channel\_fsm()

```
void GalileoElPcpsCccwsrAmbiguousAcquisition::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 94 of file `galileo_e1_pcps_cccwsr_ambiguous_acquisition.h`.

#### 10.106.2.7 set\_doppler\_max()

```
void GalileoE1PcpsCccwsrAmbiguousAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.106.2.8 set\_doppler\_step()

```
void GalileoE1PcpsCccwsrAmbiguousAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.106.2.9 set\_gnss\_synchro()

```
void GalileoE1PcpsCccwsrAmbiguousAcquisition::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](#).

#### 10.106.2.10 set\_state()

```
void GalileoE1PcpsCccwsrAmbiguousAcquisition::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

#### 10.106.2.11 set\_threshold()

```
void GalileoE1PcpsCccwsrAmbiguousAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of CCCWSR algorithm.

Implements [AcquisitionInterface](#).

### 10.106.2.12 stop\_acquisition()

```
void GalileoE1PcpsCccwsrAmbiguousAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

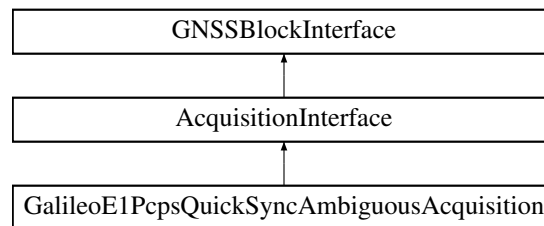
- [galileo\\_e1\\_pcps\\_cccwsr\\_ambiguous\\_acquisition.h](#)

## 10.107 GalileoE1PcpsQuickSyncAmbiguousAcquisition Class Reference

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

```
#include <galileo_e1_pcps_quicksync_ambiguous_acquisition.h>
```

Inheritance diagram for GalileoE1PcpsQuickSyncAmbiguousAcquisition:



### Public Member Functions

- **GalileoE1PcpsQuickSyncAmbiguousAcquisition** (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\_PCPS\_Ambiguous\_Acquisition".*
- 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\_threshold** (float threshold) override
 

*Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override

- *Set maximum Doppler off grid search.*  
void [set\\_doppler\\_step](#) (unsigned int doppler\_step) override
- *Set Doppler steps for the grid search.*  
void [init](#) () override
- *Initializes acquisition algorithm.*  
void [set\\_local\\_code](#) () override
- *Sets local code for Galileo E1 PCPS acquisition algorithm.*  
signed int [mag](#) () override
- *Returns the maximum peak of grid search.*  
void [reset](#) () override
- *Restart acquisition algorithm.*  
void [set\\_state](#) (int state) override
- *If state = 1, it forces the block to start acquiring from the first sample.*  
void [stop\\_acquisition](#) () override
- *Stop running acquisition.*  
void [set\\_resampler\\_latency](#) (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override

### 10.107.1 Detailed Description

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

Definition at line 41 of file `galileo_e1_pcps_quicksync_ambiguous_acquisition.h`.

### 10.107.2 Member Function Documentation

#### 10.107.2.1 implementation()

```
std::string GalileoE1PcpsQuickSyncAmbiguousAcquisition::implementation ( ) [inline], [override],
[virtual]
```

Returns "Galileo\_E1\_PCPS\_Ambiguous\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 60 of file `galileo_e1_pcps_quicksync_ambiguous_acquisition.h`.

#### 10.107.2.2 init()

```
void GalileoE1PcpsQuickSyncAmbiguousAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.107.2.3 mag()

```
signed int GalileoElPcpsQuickSyncAmbiguousAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

#### 10.107.2.4 reset()

```
void GalileoElPcpsQuickSyncAmbiguousAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.107.2.5 set\_channel()

```
void GalileoElPcpsQuickSyncAmbiguousAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 85 of file galileo\_e1\_pcps\_quicksync\_ambiguous\_acquisition.h.

#### 10.107.2.6 set\_channel\_fsm()

```
void GalileoElPcpsQuickSyncAmbiguousAcquisition::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 94 of file galileo\_e1\_pcps\_quicksync\_ambiguous\_acquisition.h.

#### 10.107.2.7 set\_doppler\_max()

```
void GalileoE1PcpsQuickSyncAmbiguousAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.107.2.8 set\_doppler\_step()

```
void GalileoE1PcpsQuickSyncAmbiguousAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.107.2.9 set\_gnss\_synchro()

```
void GalileoE1PcpsQuickSyncAmbiguousAcquisition::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](#).

#### 10.107.2.10 set\_local\_code()

```
void GalileoE1PcpsQuickSyncAmbiguousAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local code for Galileo E1 PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.107.2.11 set\_state()

```
void GalileoE1PcpsQuickSyncAmbiguousAcquisition::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

#### 10.107.2.12 `set_threshold()`

```
void GalileoE1PcpsQuickSyncAmbiguousAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

#### 10.107.2.13 `stop_acquisition()`

```
void GalileoE1PcpsQuickSyncAmbiguousAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

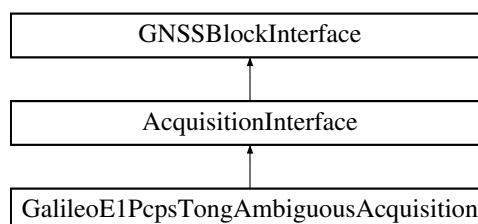
- [galileo\\_e1\\_pcps\\_quicksync\\_ambiguous\\_acquisition.h](#)

## 10.108 GalileoE1PcpsTongAmbiguousAcquisition Class Reference

Adapts a PCPS Tong acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

```
#include <galileo_e1_pcps_tong_ambiguous_acquisition.h>
```

Inheritance diagram for GalileoE1PcpsTongAmbiguousAcquisition:



## Public Member Functions

- **GalileoE1PcpsTongAmbiguousAcquisition** (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\_PCPS\_Tong\_Ambiguous\_Acquisition".*
- 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\_threshold** (float threshold) override
  - Set statistics threshold of TONG algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override
  - Set maximum Doppler off grid search.*
- void **set\_doppler\_step** (unsigned int doppler\_step) override
  - Set Doppler steps for the grid search.*
- void **init** () override
  - Initializes acquisition algorithm.*
- void **set\_local\_code** () override
  - Sets local code for Galileo E1 TONG acquisition algorithm.*
- signed int **mag** () override
  - Returns the maximum peak of grid search.*
- void **reset** () override
  - Restart acquisition algorithm.*
- void **set\_state** (int state) override
  - If state = 1, it forces the block to start acquiring from the first sample.*
- void **stop\_acquisition** () override
  - Stop running acquisition.*
- void **set\_resampler\_latency** (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override

### 10.108.1 Detailed Description

Adapts a PCPS Tong acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

Definition at line 41 of file `galileo_e1_pcps_tong_ambiguous_acquisition.h`.

### 10.108.2 Member Function Documentation

#### 10.108.2.1 implementation()

```
std::string GalileoE1PcpsTongAmbiguousAcquisition::implementation ( ) [inline], [override],  
[virtual]
```

Returns "Galileo\_E1\_PCPS\_Tong\_Ambiguous\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 60 of file `galileo_e1_pcps_tong_ambiguous_acquisition.h`.

#### 10.108.2.2 init()

```
void GalileoE1PcpsTongAmbiguousAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.108.2.3 mag()

```
signed int GalileoE1PcpsTongAmbiguousAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

#### 10.108.2.4 reset()

```
void GalileoE1PcpsTongAmbiguousAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.108.2.5 set\_channel()

```
void GalileoE1PcpsTongAmbiguousAcquisition::set_channel (   
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 85 of file `galileo_e1_pcps_tong_ambiguous_acquisition.h`.

#### 10.108.2.6 set\_channel\_fsm()

```
void GalileoE1PcpsTongAmbiguousAcquisition::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 94 of file galileo\_e1\_pcps\_tong\_ambiguous\_acquisition.h.

#### 10.108.2.7 set\_doppler\_max()

```
void GalileoE1PcpsTongAmbiguousAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.108.2.8 set\_doppler\_step()

```
void GalileoE1PcpsTongAmbiguousAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.108.2.9 set\_gnss\_synchro()

```
void GalileoE1PcpsTongAmbiguousAcquisition::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](#).

#### 10.108.2.10 `set_local_code()`

```
void GalileoE1PcpsTongAmbiguousAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local code for Galileo E1 TONG acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.108.2.11 `set_state()`

```
void GalileoE1PcpsTongAmbiguousAcquisition::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

#### 10.108.2.12 `set_threshold()`

```
void GalileoE1PcpsTongAmbiguousAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of TONG algorithm.

Implements [AcquisitionInterface](#).

#### 10.108.2.13 `stop_acquisition()`

```
void GalileoE1PcpsTongAmbiguousAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

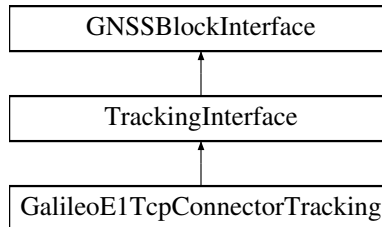
- [galileo\\_e1\\_pcps\\_tong\\_ambiguous\\_acquisition.h](#)

## 10.109 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.*

### 10.109.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 43 of file galileo\_e1\_tcp\_connector\_tracking.h.

### 10.109.2 Member Function Documentation

#### 10.109.2.1 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`.

#### 10.109.2.2 set\_channel()

```
void GalileoE1TcpConnectorTracking::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.109.2.3 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](#).

#### 10.109.2.4 stop\_tracking()

```
void GalileoE1TcpConnectorTracking::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

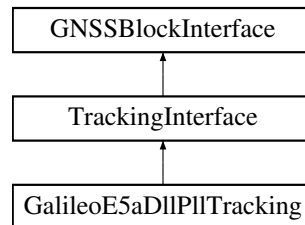
- [galileo\\_e1\\_tcp\\_connector\\_tracking.h](#)

## 10.110 GalileoE5aDIIPIITracking Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#include <galileo_e5a_dll_pll_tracking.h>
```

Inheritance diagram for GalileoE5aDIIPIITracking:



### Public Member Functions

- **GalileoE5aDIIPIITracking** (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\_E5a\_DLL\_PLL\_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.*

### 10.110.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 41 of file `galileo_e5a_dll_pll_tracking.h`.

### 10.110.2 Member Function Documentation

#### 10.110.2.1 implementation()

```
std::string GalileoE5aDllPllTracking::implementation ( ) [inline], [override], [virtual]
```

Returns "Galileo\_E5a\_DLL\_PLL\_Tracking".

Implements [GNSSBlockInterface](#).

Definition at line 58 of file `galileo_e5a_dll_pll_tracking.h`.

#### 10.110.2.2 set\_channel()

```
void GalileoE5aDllPllTracking::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.110.2.3 set\_gnss\_synchro()

```
void GalileoE5aDllPllTracking::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](#).

#### 10.110.2.4 stop\_tracking()

```
void GalileoE5aDllPllTracking::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

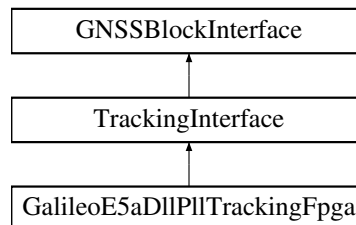
- [galileo\\_e5a\\_dll\\_pll\\_tracking.h](#)

## 10.111 GalileoE5aDIIPITrackingFpga Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#include <galileo_e5a_dll_pll_tracking_fpga.h>
```

Inheritance diagram for GalileoE5aDIIPITrackingFpga:



### Public Member Functions

- [GalileoE5aDIIPITrackingFpga](#) (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)  
*Constructor.*
- virtual [~GalileoE5aDIIPITrackingFpga](#) ()  
*Destructor.*
- std::string [role](#) () override  
*Role.*
- std::string [implementation](#) () override  
*Returns "Galileo\_E5a\_DLL\_PLL\_Tracking\_Fpga".*
- size\_t [item\\_size](#) () override  
*Returns size of lv\_16sc\_t.*
- void [connect](#) (gr::top\_block\_sptr top\_block) override  
*Connect.*
- void [disconnect](#) (gr::top\_block\_sptr top\_block) override  
*Disconnect.*
- gr::basic\_block\_sptr [get\\_left\\_block](#) () override  
*Get left block.*
- gr::basic\_block\_sptr [get\\_right\\_block](#) () override  
*Get right block.*
- 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  
*Start the tracking process in the FPGA.*
- void [stop\\_tracking](#) () override  
*Stop the tracking process in the FPGA.*

### 10.111.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 36 of file galileo\_e5a\_dll\_pll\_tracking\_fpga.h.

## 10.111.2 Constructor & Destructor Documentation

### 10.111.2.1 GalileoE5aDlIPllTrackingFpga()

```
GalileoE5aDlIPllTrackingFpga::GalileoE5aDlIPllTrackingFpga (
    const ConfigurationInterface * configuration,
    const std::string & role,
    unsigned int in_streams,
    unsigned int out_streams )
```

Constructor.

### 10.111.2.2 ~GalileoE5aDlIPllTrackingFpga()

```
virtual GalileoE5aDlIPllTrackingFpga::~~GalileoE5aDlIPllTrackingFpga ( ) [virtual]
```

Destructor.

## 10.111.3 Member Function Documentation

### 10.111.3.1 connect()

```
void GalileoE5aDlIPllTrackingFpga::connect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Connect.

Implements [GNSSBlockInterface](#).

### 10.111.3.2 disconnect()

```
void GalileoE5aDlIPllTrackingFpga::disconnect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Disconnect.

Implements [GNSSBlockInterface](#).

### 10.111.3.3 get\_left\_block()

```
gr::basic_block_sptr GalileoE5aDllPllTrackingFpga::get_left_block ( ) [override], [virtual]
```

Get left block.

Implements [GNSSBlockInterface](#).

### 10.111.3.4 get\_right\_block()

```
gr::basic_block_sptr GalileoE5aDllPllTrackingFpga::get_right_block ( ) [override], [virtual]
```

Get right block.

Implements [GNSSBlockInterface](#).

### 10.111.3.5 implementation()

```
std::string GalileoE5aDllPllTrackingFpga::implementation ( ) [inline], [override], [virtual]
```

Returns "Galileo\_E5a\_DLL\_PLL\_Tracking\_Fpga".

Implements [GNSSBlockInterface](#).

Definition at line 64 of file `galileo_e5a_dll_pll_tracking_fpga.h`.

### 10.111.3.6 item\_size()

```
size_t GalileoE5aDllPllTrackingFpga::item_size ( ) [inline], [override], [virtual]
```

Returns size of `lv_16sc_t`.

Implements [GNSSBlockInterface](#).

Definition at line 72 of file `galileo_e5a_dll_pll_tracking_fpga.h`.

### 10.111.3.7 role()

```
std::string GalileoE5aDllPllTrackingFpga::role ( ) [inline], [override], [virtual]
```

Role.

Implements [GNSSBlockInterface](#).

Definition at line 56 of file `galileo_e5a_dll_pll_tracking_fpga.h`.

#### 10.111.3.8 set\_channel()

```
void GalileoE5aDllPllTrackingFpga::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.111.3.9 set\_gnss\_synchro()

```
void GalileoE5aDllPllTrackingFpga::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](#).

#### 10.111.3.10 start\_tracking()

```
void GalileoE5aDllPllTrackingFpga::start_tracking ( ) [override], [virtual]
```

Start the tracking process in the FPGA.

Implements [TrackingInterface](#).

#### 10.111.3.11 stop\_tracking()

```
void GalileoE5aDllPllTrackingFpga::stop_tracking ( ) [override], [virtual]
```

Stop the tracking process in the FPGA.

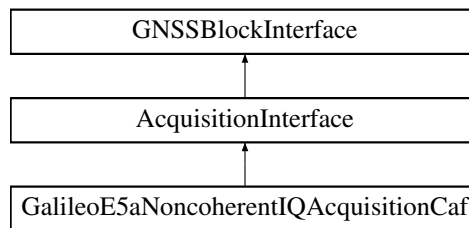
Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

- [galileo\\_e5a\\_dll\\_pll\\_tracking\\_fpga.h](#)

## 10.112 GalileoE5aNoncoherentIQAcquisitionCaf Class Reference

Inheritance diagram for GalileoE5aNoncoherentIQAcquisitionCaf:



### Public Member Functions

- **GalileoE5aNoncoherentIQAcquisitionCaf** (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\_E5a\_Noncoherent\_IQ\_Acquisition\_CAF".*
- 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\_threshold** (float threshold) override
  - Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override
  - Set maximum Doppler off grid search.*
- void **set\_doppler\_step** (unsigned int doppler\_step) override
  - Set Doppler steps for the grid search.*
- void **init** () override
  - Initializes acquisition algorithm.*
- void **set\_local\_code** () override
  - Sets local Galileo E5a code for PCPS acquisition algorithm.*
- signed int **mag** () override
  - Returns the maximum peak of grid search.*
- void **reset** () override
  - Restart acquisition algorithm.*
- void **set\_state** (int state) override
  - If set to 1, ensures that acquisition starts at the first available sample.*
- void **stop\_acquisition** () override
  - Stop running acquisition.*
- void **set\_resampler\_latency** (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override

### 10.112.1 Detailed Description

Definition at line 42 of file `galileo_e5a_noncoherent_iq_acquisition_caf.h`.

### 10.112.2 Member Function Documentation

#### 10.112.2.1 `implementation()`

```
std::string GalileoE5aNoncoherentIQAcquisitionCaf::implementation ( ) [inline], [override],  
[virtual]
```

Returns "Galileo\_E5a\_Noncoherent\_IQ\_Acquisition\_CAF".

Implements [GNSSBlockInterface](#).

Definition at line 60 of file `galileo_e5a_noncoherent_iq_acquisition_caf.h`.

#### 10.112.2.2 `init()`

```
void GalileoE5aNoncoherentIQAcquisitionCaf::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.112.2.3 `mag()`

```
signed int GalileoE5aNoncoherentIQAcquisitionCaf::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

#### 10.112.2.4 `reset()`

```
void GalileoE5aNoncoherentIQAcquisitionCaf::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.112.2.5 set\_channel()

```
void GalileoE5aNoncoherentIQAcquisitionCaf::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 85 of file `galileo_e5a_noncoherent_iq_acquisition_caf.h`.

#### 10.112.2.6 set\_channel\_fsm()

```
void GalileoE5aNoncoherentIQAcquisitionCaf::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 94 of file `galileo_e5a_noncoherent_iq_acquisition_caf.h`.

#### 10.112.2.7 set\_doppler\_max()

```
void GalileoE5aNoncoherentIQAcquisitionCaf::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.112.2.8 set\_doppler\_step()

```
void GalileoE5aNoncoherentIQAcquisitionCaf::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

**10.112.2.9 set\_gnss\_synchro()**

```
void GalileoE5aNoncoherentIQAcquisitionCaf::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](#).

**10.112.2.10 set\_local\_code()**

```
void GalileoE5aNoncoherentIQAcquisitionCaf::set_local_code ( ) [override], [virtual]
```

Sets local Galileo E5a code for PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

**10.112.2.11 set\_state()**

```
void GalileoE5aNoncoherentIQAcquisitionCaf::set_state (
    int state ) [override], [virtual]
```

If set to 1, ensures that acquisition starts at the first available sample.

**Parameters**

<i>state</i>	- int=1 forces start of acquisition
--------------	-------------------------------------

Implements [AcquisitionInterface](#).

**10.112.2.12 set\_threshold()**

```
void GalileoE5aNoncoherentIQAcquisitionCaf::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

## 10.112.2.13 stop\_acquisition()

```
void GalileoE5aNoncoherentIQAcquisitionCaf::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

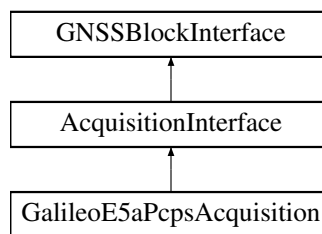
Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

- [galileo\\_e5a\\_noncoherent\\_iq\\_acquisition\\_caf.h](#)

## 10.113 GalileoE5aPcpsAcquisition Class Reference

Inheritance diagram for GalileoE5aPcpsAcquisition:



## Public Member Functions

- **GalileoE5aPcpsAcquisition** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- 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\_threshold** (float threshold) override  
*Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override  
*Set maximum Doppler off grid search.*
- void **set\_doppler\_step** (unsigned int doppler\_step) override  
*Set Doppler steps for the grid search.*
- void **set\_doppler\_center** (int doppler\_center) override  
*Set Doppler center for the grid search.*

- void [init](#) () override  
*Initializes acquisition algorithm.*
- void [set\\_local\\_code](#) () override  
*Sets local Galileo E5a code for PCPS acquisition algorithm.*
- signed int [mag](#) () override  
*Returns the maximum peak of grid search.*
- void [reset](#) () override  
*Restart acquisition algorithm.*
- void [set\\_state](#) (int state) override  
*If set to 1, ensures that acquisition starts at the first available sample.*
- 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.*

### 10.113.1 Detailed Description

Definition at line 37 of file `galileo_e5a_pcps_acquisition.h`.

### 10.113.2 Member Function Documentation

#### 10.113.2.1 `init()`

```
void GalileoE5aPcpsAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.113.2.2 `mag()`

```
signed int GalileoE5aPcpsAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

### 10.113.2.3 reset()

```
void GalileoE5aPcpsAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.113.2.4 set\_channel()

```
void GalileoE5aPcpsAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 78 of file `galileo_e5a_pcps_acquisition.h`.

### 10.113.2.5 set\_channel\_fsm()

```
void GalileoE5aPcpsAcquisition::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 87 of file `galileo_e5a_pcps_acquisition.h`.

### 10.113.2.6 set\_doppler\_center()

```
void GalileoE5aPcpsAcquisition::set_doppler_center (
    int doppler_center ) [override]
```

Set Doppler center for the grid search.

### 10.113.2.7 set\_doppler\_max()

```
void GalileoE5aPcpsAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.113.2.8 `set_doppler_step()`

```
void GalileoE5aPcpsAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.113.2.9 `set_gnss_synchro()`

```
void GalileoE5aPcpsAcquisition::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](#).

#### 10.113.2.10 `set_local_code()`

```
void GalileoE5aPcpsAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local Galileo E5a code for PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.113.2.11 `set_resampler_latency()`

```
void GalileoE5aPcpsAcquisition::set_resampler_latency (
    uint32_t latency_samples ) [override], [virtual]
```

Sets the resampler latency to account it in the acquisition code delay estimation.

Implements [AcquisitionInterface](#).

#### 10.113.2.12 `set_state()`

```
void GalileoE5aPcpsAcquisition::set_state (
    int state ) [override], [virtual]
```

If set to 1, ensures that acquisition starts at the first available sample.

## Parameters

<i>state</i>	- int=1 forces start of acquisition
--------------	-------------------------------------

Implements [AcquisitionInterface](#).

10.113.2.13 `set_threshold()`

```
void GalileoE5aPcpsAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

10.113.2.14 `stop_acquisition()`

```
void GalileoE5aPcpsAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

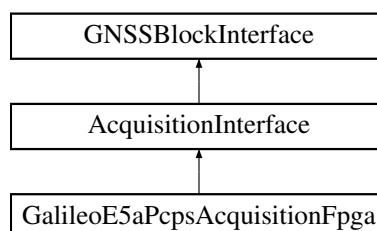
- [galileo\\_e5a\\_pcps\\_acquisition.h](#)

## 10.114 GalileoE5aPcpsAcquisitionFpga Class Reference

This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for Galileo E5a signals.

```
#include <galileo_e5a_pcps_acquisition_fpga.h>
```

Inheritance diagram for GalileoE5aPcpsAcquisitionFpga:



## Public Member Functions

- [GalileoE5aPcpsAcquisitionFpga](#) (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)  
*Constructor.*
- [~GalileoE5aPcpsAcquisitionFpga](#) ()=default  
*Destructor.*
- std::string [role](#) () override  
*Role.*
- std::string [implementation](#) () override  
*Returns "Galileo\_E5a\_Pcps\_Acquisition\_Fpga".*
- size\_t [item\\_size](#) () override  
*Returns size of lv\_16sc\_t.*
- void [connect](#) (gr::top\_block\_sptr top\_block) override  
*Connect.*
- void [disconnect](#) (gr::top\_block\_sptr top\_block) override  
*Disconnect.*
- gr::basic\_block\_sptr [get\\_left\\_block](#) () override  
*Get left block.*
- gr::basic\_block\_sptr [get\\_right\\_block](#) () override  
*Get right block.*
- 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\\_threshold](#) (float threshold) override  
*Set statistics threshold of PCPS algorithm.*
- void [set\\_doppler\\_max](#) (unsigned int doppler\_max) override  
*Set maximum Doppler off grid search.*
- void [set\\_doppler\\_step](#) (unsigned int doppler\_step) override  
*Set Doppler steps for the grid search.*
- void [set\\_doppler\\_center](#) (int doppler\_center) override  
*Set Doppler center for the grid search.*
- void [init](#) () override  
*Initializes acquisition algorithm.*
- void [set\\_local\\_code](#) () override  
*Sets local Galileo E5a code for PCPS acquisition algorithm.*
- signed int [mag](#) () override  
*Returns the maximum peak of grid search.*
- void [reset](#) () override  
*Restart acquisition algorithm.*
- void [set\\_state](#) (int state) override  
*If set to 1, ensures that acquisition starts at the first available sample.*
- void [set\\_single\\_doppler\\_flag](#) (unsigned int single\_doppler\_flag)  
*This function is only used in the unit tests.*
- void [stop\\_acquisition](#) () override  
*Stop running acquisition.*
- void [set\\_resampler\\_latency](#) (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override  
*Set resampler latency.*

### 10.114.1 Detailed Description

This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for Galileo E5a signals.

Definition at line 42 of file `galileo_e5a_pcps_acquisition_fpga.h`.

### 10.114.2 Constructor & Destructor Documentation

#### 10.114.2.1 GalileoE5aPcpsAcquisitionFpga()

```
GalileoE5aPcpsAcquisitionFpga::GalileoE5aPcpsAcquisitionFpga (
    const ConfigurationInterface * configuration,
    const std::string & role,
    unsigned int in_streams,
    unsigned int out_streams )
```

Constructor.

#### 10.114.2.2 ~GalileoE5aPcpsAcquisitionFpga()

```
GalileoE5aPcpsAcquisitionFpga::~GalileoE5aPcpsAcquisitionFpga ( ) [default]
```

Destructor.

### 10.114.3 Member Function Documentation

#### 10.114.3.1 connect()

```
void GalileoE5aPcpsAcquisitionFpga::connect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Connect.

Implements [GNSSBlockInterface](#).

#### 10.114.3.2 disconnect()

```
void GalileoE5aPcpsAcquisitionFpga::disconnect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Disconnect.

Implements [GNSSBlockInterface](#).

#### 10.114.3.3 get\_left\_block()

```
gr::basic_block_sptr GalileoE5aPcpsAcquisitionFpga::get_left_block ( ) [override], [virtual]
```

Get left block.

Implements [GNSSBlockInterface](#).

#### 10.114.3.4 get\_right\_block()

```
gr::basic_block_sptr GalileoE5aPcpsAcquisitionFpga::get_right_block ( ) [override], [virtual]
```

Get right block.

Implements [GNSSBlockInterface](#).

#### 10.114.3.5 implementation()

```
std::string GalileoE5aPcpsAcquisitionFpga::implementation ( ) [inline], [override], [virtual]
```

Returns "Galileo\_E5a\_Pcps\_Acquisition\_Fpga".

Implements [GNSSBlockInterface](#).

Definition at line 70 of file `galileo_e5a_pcps_acquisition_fpga.h`.

#### 10.114.3.6 init()

```
void GalileoE5aPcpsAcquisitionFpga::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.114.3.7 item\_size()

```
size_t GalileoE5aPcpsAcquisitionFpga::item_size ( ) [inline], [override], [virtual]
```

Returns size of lv\_16sc\_t.

Implements [GNSSBlockInterface](#).

Definition at line 78 of file galileo\_e5a\_pcps\_acquisition\_fpga.h.

### 10.114.3.8 mag()

```
signed int GalileoE5aPcpsAcquisitionFpga::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

### 10.114.3.9 reset()

```
void GalileoE5aPcpsAcquisitionFpga::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.114.3.10 role()

```
std::string GalileoE5aPcpsAcquisitionFpga::role ( ) [inline], [override], [virtual]
```

Role.

Implements [GNSSBlockInterface](#).

Definition at line 62 of file galileo\_e5a\_pcps\_acquisition\_fpga.h.

### 10.114.3.11 set\_channel()

```
void GalileoE5aPcpsAcquisitionFpga::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 113 of file galileo\_e5a\_pcps\_acquisition\_fpga.h.

#### 10.114.3.12 `set_channel_fsm()`

```
void GalileoE5aPcpsAcquisitionFpga::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 122 of file `galileo_e5a_pcps_acquisition_fpga.h`.

#### 10.114.3.13 `set_doppler_center()`

```
void GalileoE5aPcpsAcquisitionFpga::set_doppler_center (
    int doppler_center ) [override]
```

Set Doppler center for the grid search.

#### 10.114.3.14 `set_doppler_max()`

```
void GalileoE5aPcpsAcquisitionFpga::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.114.3.15 `set_doppler_step()`

```
void GalileoE5aPcpsAcquisitionFpga::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.114.3.16 `set_gnss_synchro()`

```
void GalileoE5aPcpsAcquisitionFpga::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](#).

**10.114.3.17 set\_local\_code()**

```
void GalileoE5aPcpsAcquisitionFpga::set_local_code ( ) [override], [virtual]
```

Sets local Galileo E5a code for PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

**10.114.3.18 set\_resampler\_latency()**

```
void GalileoE5aPcpsAcquisitionFpga::set_resampler_latency (
    uint32_t latency_samples __attribute__((unused)) ) [inline], [override]
```

Set resampler latency.

Definition at line 188 of file galileo\_e5a\_pcps\_acquisition\_fpga.h.

**10.114.3.19 set\_single\_doppler\_flag()**

```
void GalileoE5aPcpsAcquisitionFpga::set_single_doppler_flag (
    unsigned int single_doppler_flag )
```

This function is only used in the unit tests.

**10.114.3.20 set\_state()**

```
void GalileoE5aPcpsAcquisitionFpga::set_state (
    int state ) [override], [virtual]
```

If set to 1, ensures that acquisition starts at the first available sample.

**Parameters**

<i>state</i>	- int=1 forces start of acquisition
--------------	-------------------------------------

Implements [AcquisitionInterface](#).

**10.114.3.21 set\_threshold()**

```
void GalileoE5aPcpsAcquisitionFpga::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

#### 10.114.3.22 stop\_acquisition()

```
void GalileoE5aPcpsAcquisitionFpga::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

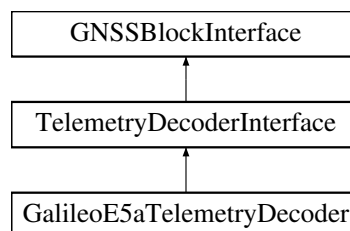
- [galileo\\_e5a\\_pcps\\_acquisition\\_fpga.h](#)

## 10.115 GalileoE5aTelemetryDecoder Class Reference

This class implements a NAV data decoder for Galileo INAV frames in E1B radio link.

```
#include <galileo_e5a_telemetry_decoder.h>
```

Inheritance diagram for GalileoE5aTelemetryDecoder:



### Public Member Functions

- **GalileoE5aTelemetryDecoder** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- 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
  - Returns "Galileo\_E5a\_Telemetry\_Decoder".*
- void **set\_channel** (int channel) override
- void **reset** () override
- size\_t **item\_size** () override

### 10.115.1 Detailed Description

This class implements a NAV data decoder for Galileo INAV frames in E1B radio link.

Definition at line 46 of file `galileo_e5a_telemetry_decoder.h`.

### 10.115.2 Member Function Documentation

#### 10.115.2.1 `implementation()`

```
std::string GalileoE5aTelemetryDecoder::implementation ( ) [inline], [override], [virtual]
```

Returns "Galileo\_E5a\_Telemetry\_Decoder".

Implements [GNSSBlockInterface](#).

Definition at line 72 of file `galileo_e5a_telemetry_decoder.h`.

The documentation for this class was generated from the following file:

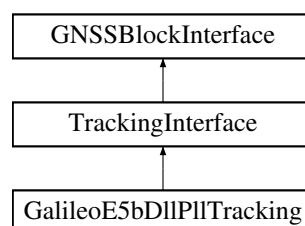
- [galileo\\_e5a\\_telemetry\\_decoder.h](#)

## 10.116 GalileoE5bDIIPIITracking Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#include <galileo_e5b_dll_pll_tracking.h>
```

Inheritance diagram for GalileoE5bDIIPIITracking:



## Public Member Functions

- **GalileoE5bDl1Pl1Tracking** (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\_E5b\_DLL\_PLL\_Tracking".*
- size\_t **item\_size** () override
- void **connect** (gr::top\_block\_sptr top\_block) override
 

*Connect.*
- void **disconnect** (gr::top\_block\_sptr top\_block) override
 

*Disconnect.*
- gr::basic\_block\_sptr **get\_left\_block** () override
 

*Get left block.*
- gr::basic\_block\_sptr **get\_right\_block** () override
 

*Get right block.*
- 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.*

### 10.116.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 42 of file `galileo_e5b_dll_pll_tracking.h`.

### 10.116.2 Member Function Documentation

#### 10.116.2.1 connect()

```
void GalileoE5bDl1Pl1Tracking::connect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Connect.

Implements [GNSSBlockInterface](#).

### 10.116.2.2 disconnect()

```
void GalileoE5bDlIPllTracking::disconnect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Disconnect.

Implements [GNSSBlockInterface](#).

### 10.116.2.3 get\_left\_block()

```
gr::basic_block_sptr GalileoE5bDlIPllTracking::get_left_block ( ) [override], [virtual]
```

Get left block.

Implements [GNSSBlockInterface](#).

### 10.116.2.4 get\_right\_block()

```
gr::basic_block_sptr GalileoE5bDlIPllTracking::get_right_block ( ) [override], [virtual]
```

Get right block.

Implements [GNSSBlockInterface](#).

### 10.116.2.5 implementation()

```
std::string GalileoE5bDlIPllTracking::implementation ( ) [inline], [override], [virtual]
```

Returns "Galileo\_E5b\_DLL\_PLL\_Tracking".

Implements [GNSSBlockInterface](#).

Definition at line 59 of file `galileo_e5b_dll_pll_tracking.h`.

### 10.116.2.6 set\_channel()

```
void GalileoE5bDlIPllTracking::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

### 10.116.2.7 set\_gnss\_synchro()

```
void GalileoE5bDllPllTracking::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](#).

### 10.116.2.8 stop\_tracking()

```
void GalileoE5bDllPllTracking::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

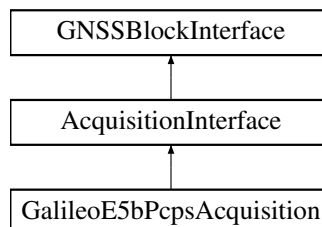
Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

- [galileo\\_e5b\\_dll\\_pll\\_tracking.h](#)

## 10.117 GalileoE5bPcpsAcquisition Class Reference

Inheritance diagram for GalileoE5bPcpsAcquisition:



### Public Member Functions

- [GalileoE5bPcpsAcquisition](#) (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)  
*Constructor.*
- [~GalileoE5bPcpsAcquisition](#) ()=default  
*Destructor.*
- std::string [role](#) () override  
*Role.*
- std::string [implementation](#) () override  
*Returns "GALILEO\_E5b\_PCPS\_Acquisition".*
- size\_t [item\\_size](#) () override  
*Returns size of lv\_16sc\_t.*

- void [connect](#) (gr::top\_block\_sptr top\_block) override  
*Connect.*
- void [disconnect](#) (gr::top\_block\_sptr top\_block) override  
*Disconnect.*
- gr::basic\_block\_sptr [get\\_left\\_block](#) () override  
*Get left block.*
- gr::basic\_block\_sptr [get\\_right\\_block](#) () override  
*Get right block.*
- 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\\_threshold](#) (float threshold) override  
*Set statistics threshold of PCPS algorithm.*
- void [set\\_doppler\\_max](#) (unsigned int doppler\_max) override  
*Set maximum Doppler off grid search.*
- void [set\\_doppler\\_step](#) (unsigned int doppler\_step) override  
*Set Doppler steps for the grid search.*
- void [set\\_doppler\\_center](#) (int doppler\_center) override  
*Set Doppler center for the grid search.*
- void [init](#) () override  
*Initializes acquisition algorithm.*
- void [set\\_local\\_code](#) () override  
*Sets local Galileo E5b code for PCPS acquisition algorithm.*
- signed int [mag](#) () override  
*Returns the maximum peak of grid search.*
- void [reset](#) () override  
*Restart acquisition algorithm.*
- void [set\\_state](#) (int state) override  
*If set to 1, ensures that acquisition starts at the first available sample.*
- 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.*

### 10.117.1 Detailed Description

Definition at line 38 of file `galileo_e5b_pcps_acquisition.h`.

### 10.117.2 Constructor & Destructor Documentation

### 10.117.2.1 GalileoE5bPcpsAcquisition()

```
GalileoE5bPcpsAcquisition::GalileoE5bPcpsAcquisition (
    const ConfigurationInterface * configuration,
    const std::string & role,
    unsigned int in_streams,
    unsigned int out_streams )
```

Constructor.

### 10.117.2.2 ~GalileoE5bPcpsAcquisition()

```
GalileoE5bPcpsAcquisition::~GalileoE5bPcpsAcquisition ( ) [default]
```

Destructor.

## 10.117.3 Member Function Documentation

### 10.117.3.1 connect()

```
void GalileoE5bPcpsAcquisition::connect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Connect.

Implements [GNSSBlockInterface](#).

### 10.117.3.2 disconnect()

```
void GalileoE5bPcpsAcquisition::disconnect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Disconnect.

Implements [GNSSBlockInterface](#).

### 10.117.3.3 get\_left\_block()

```
gr::basic_block_sptr GalileoE5bPcpsAcquisition::get_left_block ( ) [override], [virtual]
```

Get left block.

Implements [GNSSBlockInterface](#).

#### 10.117.3.4 get\_right\_block()

```
gr::basic_block_sptr GalileoE5bPcpsAcquisition::get_right_block ( ) [override], [virtual]
```

Get right block.

Implements [GNSSBlockInterface](#).

#### 10.117.3.5 implementation()

```
std::string GalileoE5bPcpsAcquisition::implementation ( ) [inline], [override], [virtual]
```

Returns "GALILEO\_E5b\_PCPS\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 66 of file galileo\_e5b\_pcps\_acquisition.h.

#### 10.117.3.6 init()

```
void GalileoE5bPcpsAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.117.3.7 item\_size()

```
size_t GalileoE5bPcpsAcquisition::item_size ( ) [inline], [override], [virtual]
```

Returns size of lv\_16sc\_t.

Implements [GNSSBlockInterface](#).

Definition at line 74 of file galileo\_e5b\_pcps\_acquisition.h.

#### 10.117.3.8 mag()

```
signed int GalileoE5bPcpsAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

#### 10.117.3.9 reset()

```
void GalileoE5bPcpsAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.117.3.10 role()

```
std::string GalileoE5bPcpsAcquisition::role ( ) [inline], [override], [virtual]
```

Role.

Implements [GNSSBlockInterface](#).

Definition at line 57 of file `galileo_e5b_pcps_acquisition.h`.

#### 10.117.3.11 set\_channel()

```
void GalileoE5bPcpsAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 109 of file `galileo_e5b_pcps_acquisition.h`.

#### 10.117.3.12 set\_channel\_fsm()

```
void GalileoE5bPcpsAcquisition::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 118 of file `galileo_e5b_pcps_acquisition.h`.

**10.117.3.13 set\_doppler\_center()**

```
void GalileoE5bPcpsAcquisition::set_doppler_center (
    int doppler_center ) [override]
```

Set Doppler center for the grid search.

**10.117.3.14 set\_doppler\_max()**

```
void GalileoE5bPcpsAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

**10.117.3.15 set\_doppler\_step()**

```
void GalileoE5bPcpsAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

**10.117.3.16 set\_gnss\_synchro()**

```
void GalileoE5bPcpsAcquisition::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](#).

**10.117.3.17 set\_local\_code()**

```
void GalileoE5bPcpsAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local Galileo E5b code for PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.117.3.18 `set_resampler_latency()`

```
void GalileoE5bPcpsAcquisition::set_resampler_latency (
    uint32_t latency_samples ) [override], [virtual]
```

Sets the resampler latency to account it in the acquisition code delay estimation.

Implements [AcquisitionInterface](#).

#### 10.117.3.19 `set_state()`

```
void GalileoE5bPcpsAcquisition::set_state (
    int state ) [override], [virtual]
```

If set to 1, ensures that acquisition starts at the first available sample.

##### Parameters

<i>state</i>	- int=1 forces start of acquisition
--------------	-------------------------------------

Implements [AcquisitionInterface](#).

#### 10.117.3.20 `set_threshold()`

```
void GalileoE5bPcpsAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

#### 10.117.3.21 `stop_acquisition()`

```
void GalileoE5bPcpsAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

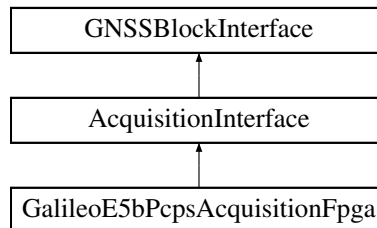
- [galileo\\_e5b\\_pcps\\_acquisition.h](#)

## 10.118 GalileoE5bPcpsAcquisitionFpga Class Reference

This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for Galileo E5b signals.

```
#include <galileo_e5b_pcps_acquisition_fpga.h>
```

Inheritance diagram for GalileoE5bPcpsAcquisitionFpga:



### Public Member Functions

- [GalileoE5bPcpsAcquisitionFpga](#) (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)  
*Constructor.*
- [~GalileoE5bPcpsAcquisitionFpga](#) ()=default  
*Destructor.*
- std::string [role](#) () override  
*Role.*
- std::string [implementation](#) () override  
*Returns "Galileo\_E5b\_Pcps\_Acquisition\_Fpga".*
- size\_t [item\\_size](#) () override  
*Returns size of lv\_16sc\_t.*
- void [connect](#) (gr::top\_block\_sptr top\_block) override  
*Connect.*
- void [disconnect](#) (gr::top\_block\_sptr top\_block) override  
*Disconnect.*
- gr::basic\_block\_sptr [get\\_left\\_block](#) () override  
*Get left block.*
- gr::basic\_block\_sptr [get\\_right\\_block](#) () override  
*Get right block.*
- 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\\_threshold](#) (float threshold) override  
*Set statistics threshold of PCPS algorithm.*
- void [set\\_doppler\\_max](#) (unsigned int doppler\_max) override  
*Set maximum Doppler off grid search.*
- void [set\\_doppler\\_step](#) (unsigned int doppler\_step) override  
*Set Doppler steps for the grid search.*

- void [set\\_doppler\\_center](#) (int doppler\_center) override  
*Set Doppler center for the grid search.*
- void [init](#) () override  
*Initializes acquisition algorithm.*
- void [set\\_local\\_code](#) () override  
*Sets local Galileo E5b code for PCPS acquisition algorithm.*
- signed int [mag](#) () override  
*Returns the maximum peak of grid search.*
- void [reset](#) () override  
*Restart acquisition algorithm.*
- void [set\\_state](#) (int state) override  
*If set to 1, ensures that acquisition starts at the first available sample.*
- void [set\\_single\\_doppler\\_flag](#) (unsigned int single\_doppler\_flag)  
*This function is only used in the unit tests.*
- void [stop\\_acquisition](#) () override  
*Stop running acquisition.*
- void [set\\_resampler\\_latency](#) (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override  
*Set resampler latency.*

### 10.118.1 Detailed Description

This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for Galileo E5b signals.

Definition at line 42 of file `galileo_e5b_pcps_acquisition_fpga.h`.

### 10.118.2 Constructor & Destructor Documentation

#### 10.118.2.1 GalileoE5bPcpsAcquisitionFpga()

```
GalileoE5bPcpsAcquisitionFpga::GalileoE5bPcpsAcquisitionFpga (
    const ConfigurationInterface * configuration,
    const std::string & role,
    unsigned int in_streams,
    unsigned int out_streams )
```

Constructor.

#### 10.118.2.2 ~GalileoE5bPcpsAcquisitionFpga()

```
GalileoE5bPcpsAcquisitionFpga::~GalileoE5bPcpsAcquisitionFpga ( ) [default]
```

Destructor.

### 10.118.3 Member Function Documentation

#### 10.118.3.1 connect()

```
void GalileoE5bPcpsAcquisitionFpga::connect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Connect.

Implements [GNSSBlockInterface](#).

#### 10.118.3.2 disconnect()

```
void GalileoE5bPcpsAcquisitionFpga::disconnect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Disconnect.

Implements [GNSSBlockInterface](#).

#### 10.118.3.3 get\_left\_block()

```
gr::basic_block_sptr GalileoE5bPcpsAcquisitionFpga::get_left_block ( ) [override], [virtual]
```

Get left block.

Implements [GNSSBlockInterface](#).

#### 10.118.3.4 get\_right\_block()

```
gr::basic_block_sptr GalileoE5bPcpsAcquisitionFpga::get_right_block ( ) [override], [virtual]
```

Get right block.

Implements [GNSSBlockInterface](#).

#### 10.118.3.5 implementation()

```
std::string GalileoE5bPcpsAcquisitionFpga::implementation ( ) [inline], [override], [virtual]
```

Returns "Galileo\_E5b\_Pcps\_Acquisition\_Fpga".

Implements [GNSSBlockInterface](#).

Definition at line 69 of file galileo\_e5b\_pcps\_acquisition\_fpga.h.

#### 10.118.3.6 init()

```
void GalileoE5bPcpsAcquisitionFpga::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.118.3.7 item\_size()

```
size_t GalileoE5bPcpsAcquisitionFpga::item_size ( ) [inline], [override], [virtual]
```

Returns size of lv\_16sc\_t.

Implements [GNSSBlockInterface](#).

Definition at line 77 of file galileo\_e5b\_pcps\_acquisition\_fpga.h.

#### 10.118.3.8 mag()

```
signed int GalileoE5bPcpsAcquisitionFpga::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

#### 10.118.3.9 reset()

```
void GalileoE5bPcpsAcquisitionFpga::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.118.3.10 `role()`

```
std::string GalileoE5bPcpsAcquisitionFpga::role ( ) [inline], [override], [virtual]
```

Role.

Implements [GNSSBlockInterface](#).

Definition at line 61 of file `galileo_e5b_pcps_acquisition_fpga.h`.

### 10.118.3.11 `set_channel()`

```
void GalileoE5bPcpsAcquisitionFpga::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 112 of file `galileo_e5b_pcps_acquisition_fpga.h`.

### 10.118.3.12 `set_channel_fsm()`

```
void GalileoE5bPcpsAcquisitionFpga::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 121 of file `galileo_e5b_pcps_acquisition_fpga.h`.

### 10.118.3.13 `set_doppler_center()`

```
void GalileoE5bPcpsAcquisitionFpga::set_doppler_center (
    int doppler_center ) [override]
```

Set Doppler center for the grid search.

#### 10.118.3.14 `set_doppler_max()`

```
void GalileoE5bPcpsAcquisitionFpga::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.118.3.15 `set_doppler_step()`

```
void GalileoE5bPcpsAcquisitionFpga::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.118.3.16 `set_gnss_synchro()`

```
void GalileoE5bPcpsAcquisitionFpga::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](#).

#### 10.118.3.17 `set_local_code()`

```
void GalileoE5bPcpsAcquisitionFpga::set_local_code ( ) [override], [virtual]
```

Sets local Galileo E5b code for PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.118.3.18 `set_resampler_latency()`

```
void GalileoE5bPcpsAcquisitionFpga::set_resampler_latency (
    uint32_t latency_samples \_\_attribute\_\_\(\(unused\)\) ) [inline], [override]
```

Set resampler latency.

Definition at line 187 of file `galileo_e5b_pcps_acquisition_fpga.h`.

### 10.118.3.19 set\_single\_doppler\_flag()

```
void GalileoE5bPcpsAcquisitionFpga::set_single_doppler_flag (
    unsigned int single_doppler_flag )
```

This function is only used in the unit tests.

### 10.118.3.20 set\_state()

```
void GalileoE5bPcpsAcquisitionFpga::set_state (
    int state ) [override], [virtual]
```

If set to 1, ensures that acquisition starts at the first available sample.

#### Parameters

<i>state</i>	- int=1 forces start of acquisition
--------------	-------------------------------------

Implements [AcquisitionInterface](#).

### 10.118.3.21 set\_threshold()

```
void GalileoE5bPcpsAcquisitionFpga::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

### 10.118.3.22 stop\_acquisition()

```
void GalileoE5bPcpsAcquisitionFpga::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

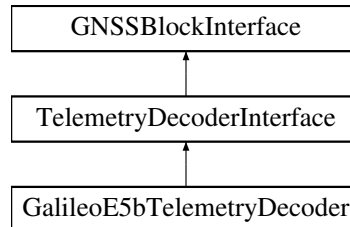
- [galileo\\_e5b\\_pcps\\_acquisition\\_fpga.h](#)

## 10.119 GalileoE5bTelemetryDecoder Class Reference

This class implements a NAV data decoder for Galileo INAV frames in E5b radio link.

```
#include <galileo_e5b_telemetry_decoder.h>
```

Inheritance diagram for GalileoE5bTelemetryDecoder:



### Public Member Functions

- **GalileoE5bTelemetryDecoder** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- std::string [implementation](#) () override  
*Returns "Galileo\_E5b\_Telemetry\_Decoder".*
- void [connect](#) (gr::top\_block\_sptr top\_block) override  
*Connect.*
- void [disconnect](#) (gr::top\_block\_sptr top\_block) override  
*Disconnect.*
- gr::basic\_block\_sptr [get\\_left\\_block](#) () override  
*Get left block.*
- gr::basic\_block\_sptr [get\\_right\\_block](#) () override  
*Get right block.*
- void [set\\_satellite](#) (const [Gnss\\_Satellite](#) &satellite) override
- std::string [role](#) () override
- void [set\\_channel](#) (int channel) override
- void [reset](#) () override
- size\_t [item\\_size](#) () override

### 10.119.1 Detailed Description

This class implements a NAV data decoder for Galileo INAV frames in E5b radio link.

Definition at line 44 of file `galileo_e5b_telemetry_decoder.h`.

### 10.119.2 Member Function Documentation

### 10.119.2.1 connect()

```
void GalileoE5bTelemetryDecoder::connect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Connect.

Implements [GNSSBlockInterface](#).

### 10.119.2.2 disconnect()

```
void GalileoE5bTelemetryDecoder::disconnect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Disconnect.

Implements [GNSSBlockInterface](#).

### 10.119.2.3 get\_left\_block()

```
gr::basic_block_sptr GalileoE5bTelemetryDecoder::get_left_block ( ) [override], [virtual]
```

Get left block.

Implements [GNSSBlockInterface](#).

### 10.119.2.4 get\_right\_block()

```
gr::basic_block_sptr GalileoE5bTelemetryDecoder::get_right_block ( ) [override], [virtual]
```

Get right block.

Implements [GNSSBlockInterface](#).

### 10.119.2.5 implementation()

```
std::string GalileoE5bTelemetryDecoder::implementation ( ) [inline], [override], [virtual]
```

Returns "Galileo\_E5b\_Telemetry\_Decoder".

Implements [GNSSBlockInterface](#).

Definition at line 58 of file `galileo_e5b_telemetry_decoder.h`.

The documentation for this class was generated from the following file:

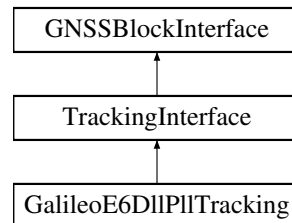
- [galileo\\_e5b\\_telemetry\\_decoder.h](#)

## 10.120 GalileoE6DIPIITracking Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#include <galileo_e6_dll_pll_tracking.h>
```

Inheritance diagram for GalileoE6DIPIITracking:



### Public Member Functions

- **GalileoE6DIPIITracking** (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\_E6\_DLL\_PLL\_Tracking".*
- size\_t **item\_size** () override
- void **connect** (gr::top\_block\_sptr top\_block) override  
*Connect.*
- void **disconnect** (gr::top\_block\_sptr top\_block) override  
*Disconnect.*
- gr::basic\_block\_sptr **get\_left\_block** () override  
*Get left block.*
- gr::basic\_block\_sptr **get\_right\_block** () override  
*Get right block.*
- 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.*

### 10.120.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 37 of file galileo\_e6\_dll\_pll\_tracking.h.

### 10.120.2 Member Function Documentation

### 10.120.2.1 connect()

```
void GalileoE6DllPllTracking::connect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Connect.

Implements [GNSSBlockInterface](#).

### 10.120.2.2 disconnect()

```
void GalileoE6DllPllTracking::disconnect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Disconnect.

Implements [GNSSBlockInterface](#).

### 10.120.2.3 get\_left\_block()

```
gr::basic_block_sptr GalileoE6DllPllTracking::get_left_block ( ) [override], [virtual]
```

Get left block.

Implements [GNSSBlockInterface](#).

### 10.120.2.4 get\_right\_block()

```
gr::basic_block_sptr GalileoE6DllPllTracking::get_right_block ( ) [override], [virtual]
```

Get right block.

Implements [GNSSBlockInterface](#).

### 10.120.2.5 implementation()

```
std::string GalileoE6DllPllTracking::implementation ( ) [inline], [override], [virtual]
```

Returns "Galileo\_E6\_DLL\_PLL\_Tracking".

Implements [GNSSBlockInterface](#).

Definition at line 54 of file `galileo_e6_dll_pll_tracking.h`.

#### 10.120.2.6 set\_channel()

```
void GalileoE6DllPllTracking::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.120.2.7 set\_gnss\_synchro()

```
void GalileoE6DllPllTracking::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](#).

#### 10.120.2.8 stop\_tracking()

```
void GalileoE6DllPllTracking::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

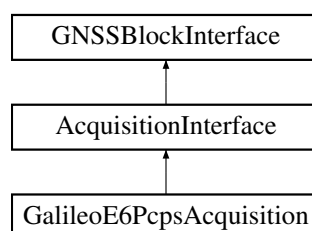
- [galileo\\_e6\\_dll\\_pll\\_tracking.h](#)

## 10.121 GalileoE6PcpsAcquisition Class Reference

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E6 Signals.

```
#include <galileo_e6_pcps_acquisition.h>
```

Inheritance diagram for GalileoE6PcpsAcquisition:



## Public Member Functions

- **GalileoE6PcpsAcquisition** (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\_E6\_PCPS\_Acquisition".*
- 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\_threshold** (float threshold) override
  - Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override
  - Set maximum Doppler off grid search.*
- void **set\_doppler\_step** (unsigned int doppler\_step) override
  - Set Doppler steps for the grid search.*
- void **set\_doppler\_center** (int doppler\_center) override
  - Set Doppler center for the grid search.*
- void **init** () override
  - Initializes acquisition algorithm.*
- void **set\_local\_code** () override
  - Sets local code for Galileo E1 PCPS acquisition algorithm.*
- signed int **mag** () override
  - Returns the maximum peak of grid search.*
- void **reset** () override
  - Restart acquisition algorithm.*
- void **set\_state** (int state) override
  - If state = 1, it forces the block to start acquiring from the first sample.*
- 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.*

### 10.121.1 Detailed Description

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E6 Signals.

Definition at line 43 of file `galileo_e6_pcps_acquisition.h`.

### 10.121.2 Member Function Documentation

#### 10.121.2.1 implementation()

```
std::string GalileoE6PcpsAcquisition::implementation ( ) [inline], [override], [virtual]
```

Returns "Galileo\_E6\_PCPS\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 62 of file `galileo_e6_pcps_acquisition.h`.

#### 10.121.2.2 init()

```
void GalileoE6PcpsAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.121.2.3 mag()

```
signed int GalileoE6PcpsAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

#### 10.121.2.4 reset()

```
void GalileoE6PcpsAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.121.2.5 set\_channel()

```
void GalileoE6PcpsAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 87 of file `galileo_e6_pcps_acquisition.h`.

### 10.121.2.6 set\_channel\_fsm()

```
void GalileoE6PcpsAcquisition::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 96 of file galileo\_e6\_pcps\_acquisition.h.

### 10.121.2.7 set\_doppler\_center()

```
void GalileoE6PcpsAcquisition::set_doppler_center (
    int doppler_center ) [override]
```

Set Doppler center for the grid search.

### 10.121.2.8 set\_doppler\_max()

```
void GalileoE6PcpsAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

### 10.121.2.9 set\_doppler\_step()

```
void GalileoE6PcpsAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

### 10.121.2.10 set\_gnss\_synchro()

```
void GalileoE6PcpsAcquisition::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](#).

#### 10.121.2.11 `set_local_code()`

```
void GalileoE6PcpsAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local code for Galileo E1 PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.121.2.12 `set_resampler_latency()`

```
void GalileoE6PcpsAcquisition::set_resampler_latency (
    uint32_t latency_samples ) [override], [virtual]
```

Sets the resampler latency to account it in the acquisition code delay estimation.

Implements [AcquisitionInterface](#).

#### 10.121.2.13 `set_state()`

```
void GalileoE6PcpsAcquisition::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

#### 10.121.2.14 `set_threshold()`

```
void GalileoE6PcpsAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

#### 10.121.2.15 `stop_acquisition()`

```
void GalileoE6PcpsAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

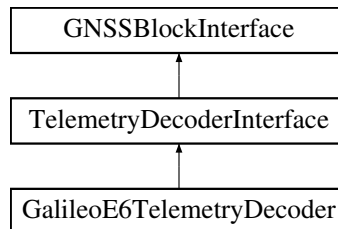
- [galileo\\_e6\\_pcps\\_acquisition.h](#)

## 10.122 GalileoE6TelemetryDecoder Class Reference

This class implements a NAV data decoder for Galileo CNAV frames in E6 radio link.

```
#include <galileo_e6_telemetry_decoder.h>
```

Inheritance diagram for GalileoE6TelemetryDecoder:



### Public Member Functions

- **GalileoE6TelemetryDecoder** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- std::string [implementation](#) () override  
*Returns "Galileo\_E6\_Telemetry\_Decoder".*
- void [connect](#) (gr::top\_block\_sptr top\_block) override  
*Connect.*
- void [disconnect](#) (gr::top\_block\_sptr top\_block) override  
*Disconnect.*
- gr::basic\_block\_sptr [get\\_left\\_block](#) () override  
*Get left block.*
- gr::basic\_block\_sptr [get\\_right\\_block](#) () override  
*Get right block.*
- void [set\\_satellite](#) (const [Gnss\\_Satellite](#) &satellite) override
- std::string [role](#) () override
- void [set\\_channel](#) (int channel) override
- void [reset](#) () override
- size\_t [item\\_size](#) () override

### 10.122.1 Detailed Description

This class implements a NAV data decoder for Galileo CNAV frames in E6 radio link.

Definition at line 43 of file `galileo_e6_telemetry_decoder.h`.

### 10.122.2 Member Function Documentation

#### 10.122.2.1 connect()

```
void GalileoE6TelemetryDecoder::connect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Connect.

Implements [GNSSBlockInterface](#).

#### 10.122.2.2 disconnect()

```
void GalileoE6TelemetryDecoder::disconnect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Disconnect.

Implements [GNSSBlockInterface](#).

#### 10.122.2.3 get\_left\_block()

```
gr::basic_block_sptr GalileoE6TelemetryDecoder::get_left_block ( ) [override], [virtual]
```

Get left block.

Implements [GNSSBlockInterface](#).

#### 10.122.2.4 get\_right\_block()

```
gr::basic_block_sptr GalileoE6TelemetryDecoder::get_right_block ( ) [override], [virtual]
```

Get right block.

Implements [GNSSBlockInterface](#).

#### 10.122.2.5 implementation()

```
std::string GalileoE6TelemetryDecoder::implementation ( ) [inline], [override], [virtual]
```

Returns "Galileo\_E6\_Telemetry\_Decoder".

Implements [GNSSBlockInterface](#).

Definition at line 57 of file `galileo_e6_telemetry_decoder.h`.

The documentation for this class was generated from the following file:

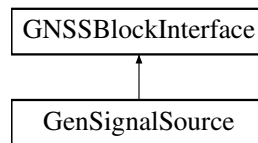
- [galileo\\_e6\\_telemetry\\_decoder.h](#)

## 10.123 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.*
- virtual [~GenSignalSource](#) ()=default  
*Virtual 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 Source".*
- size\_t **item\_size** () override
- std::shared\_ptr< [GNSSBlockInterface](#) > **signal\_generator** () const

### 10.123.1 Detailed Description

This class wraps blocks that generates synthesized GNSS signal and filters the signal.

Definition at line 40 of file gen\_signal\_source.h.

### 10.123.2 Constructor & Destructor Documentation

#### 10.123.2.1 GenSignalSource()

```

GenSignalSource::GenSignalSource (
    std::shared_ptr< GNSSBlockInterface > signal_generator,
    std::shared_ptr< GNSSBlockInterface > filter,
    std::string role,
    Concurrent\_Queue< pmt::pmt_t > * queue )

```

Constructor.

#### 10.123.2.2 ~GenSignalSource()

```
virtual GenSignalSource::~~GenSignalSource ( ) [virtual], [default]
```

Virtual destructor.

### 10.123.3 Member Function Documentation

#### 10.123.3.1 implementation()

```
std::string GenSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "Signal Source".

Implements [GNSSBlockInterface](#).

Definition at line 57 of file `gen_signal_source.h`.

The documentation for this class was generated from the following file:

- [gen\\_signal\\_source.h](#)

## 10.124 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 print\_average\_values)
- bool **close\_file** ()

#### 10.124.1 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](#)

## 10.125 `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`

### 10.125.1 Detailed Description

Definition at line 464 of file `rtklib.h`.

The documentation for this struct was generated from the following file:

- [`rtklib.h`](#)

## 10.126 `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
- `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

- 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<sup>2</sup>].*
- 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\_lambda\_n\_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\_lambda\_n\_A[s].*
- bool [d\\_C\\_n](#) {}  
*Generalized "unhealthy flag" of n\_A satellite at instant of almanac upload [dimensionless].*
- bool [d\\_I\\_n](#) {}  
*Health flag for nth satellite; In = 0 indicates the n-th satellite is healthy, In = 1 indicates malfunction of this nth satellite [dimensionless].*
- int32\_t [i\\_satellite\\_freq\\_channel](#) {}  
*SV Frequency [Channel](#) Number.*
- uint32\_t [i\\_satellite\\_PRN](#) {}  
*SV PRN Number, equivalent to slot number for compatibility with GPS.*
- uint32\_t [i\\_satellite\\_slot\\_number](#) {}  
*SV Slot Number.*

### 10.126.1 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 37 of file glonass\_gnav\_almanac.h.

## 10.126.2 Constructor & Destructor Documentation

### 10.126.2.1 Glonass\_Gnav\_Almanac()

```
Glonass_Gnav_Almanac::Glonass_Gnav_Almanac ( ) [default]
```

Default constructor

## 10.126.3 Member Function Documentation

### 10.126.3.1 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 69 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_I_n`, `d_lambda_n_A`, `d_M_n_A`, `d_n_A`, `d_omega_n_A`, `d_t_lambda_n_A`, `d_tau_n_A`, `i_satellite_freq_channel`, `i_satellite_PRN`, and `i_satellite_slot_number`.

## 10.126.4 Member Data Documentation

### 10.126.4.1 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 57 of file glonass\_gnav\_almanac.h.

Referenced by `serialize()`.

#### 10.126.4.2 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 49 of file glonass\_gnav\_almanac.h.

Referenced by serialize().

#### 10.126.4.3 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 50 of file glonass\_gnav\_almanac.h.

Referenced by serialize().

#### 10.126.4.4 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<sup>2</sup>].

Definition at line 51 of file glonass\_gnav\_almanac.h.

Referenced by serialize().

#### 10.126.4.5 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 52 of file glonass\_gnav\_almanac.h.

Referenced by serialize().

#### 10.126.4.6 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 46 of file glonass\_gnav\_almanac.h.

Referenced by `serialize()`.

#### 10.126.4.7 d\_KP

```
double Glonass_Gnav_Almanac::d_KP {}
```

Notification on forthcoming leap second correction of UTC [dimensionless].

Definition at line 55 of file glonass\_gnav\_almanac.h.

Referenced by `serialize()`.

#### 10.126.4.8 d\_l\_n

```
bool Glonass_Gnav_Almanac::d_l_n {}
```

Health flag for nth satellite; l<sub>n</sub> = 0 indicates the n-th satellite is healthy, l<sub>n</sub> = 1 indicates malfunction of this nth satellite [dimensionless].

Definition at line 58 of file glonass\_gnav\_almanac.h.

Referenced by `serialize()`.

#### 10.126.4.9 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 47 of file glonass\_gnav\_almanac.h.

Referenced by `serialize()`.

#### 10.126.4.10 d\_M\_n\_A

```
double Glonass_Gnav_Almanac::d_M_n_A {}
```

Type of satellite n\_A [dimensionless].

Definition at line 54 of file glonass\_gnav\_almanac.h.

Referenced by `serialize()`.

#### 10.126.4.11 d\_n\_A

```
double Glonass_Gnav_Almanac::d_n_A {}
```

Conventional number of satellite within GLONASS space segment [dimensionless].

Definition at line 45 of file glonass\_gnav\_almanac.h.

Referenced by `serialize()`.

#### 10.126.4.12 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 53 of file glonass\_gnav\_almanac.h.

Referenced by `serialize()`.

#### 10.126.4.13 d\_t\_lambda\_n\_A

```
double Glonass_Gnav_Almanac::d_t_lambda_n_A {}
```

Time of first ascending node passage [s].

Definition at line 48 of file glonass\_gnav\_almanac.h.

Referenced by `serialize()`.

#### 10.126.4.14 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 56 of file glonass\_gnav\_almanac.h.

Referenced by `serialize()`.

#### 10.126.4.15 i\_satellite\_freq\_channel

```
int32_t Glonass_Gnav_Almanac::i_satellite_freq_channel {}
```

SV Frequency [Channel](#) Number.

Definition at line 61 of file glonass\_gnav\_almanac.h.

Referenced by `serialize()`.

#### 10.126.4.16 i\_satellite\_PRN

```
uint32_t Glonass_Gnav_Almanac::i_satellite_PRN {}
```

SV PRN Number, equivalent to slot number for compatibility with GPS.

Definition at line 62 of file glonass\_gnav\_almanac.h.

Referenced by `serialize()`.

#### 10.126.4.17 i\_satellite\_slot\_number

```
uint32_t Glonass_Gnav_Almanac::i_satellite_slot_number {}
```

SV Slot Number.

Definition at line 63 of file glonass\_gnav\_almanac.h.

Referenced by `serialize()`.

The documentation for this class was generated from the following file:

- [glonass\\_gnav\\_almanac.h](#)

## 10.127 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 (IS-GPS-200K, 20.3.3.3.3.1)*
- 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](#) (const double offset\_time, const 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 \*T<sub>OW</sub>) 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<sup>2</sup>].*

  - double [d\\_AYn](#) {}
- Earth-fixed acceleration coordinate y of the satellite in PZ-90.02 coordinate system [km/s<sup>2</sup>].*

  - double [d\\_AZn](#) {}
- Earth-fixed acceleration coordinate z of the satellite in PZ-90.02 coordinate system [km/s<sup>2</sup>].*

  - 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; In = 0 indicates the n-th satellite is healthy, In = 1 indicates malfunction of this nth satellite [dimensionless].*

  - bool [d\\_l5th\\_n](#) {}
- Health flag for nth satellite; In = 0 indicates the n-th satellite is healthy, In = 1 indicates malfunction of this nth satellite [dimensionless].*

  - int32\_t [i\\_satellite\\_freq\\_channel](#) {}
- SV Frequency [Channel](#) Number.*

  - uint32\_t [i\\_satellite\\_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\\_iodo](#) {}  
*Issue of data, ephemeris (Bit 0-6 of tb)*
- double [d\\_tau\\_c](#) {}  
*GLONASS 2 UTC correction (todo) may be eliminated.*
- double [d\\_TOW](#) {}  
*GLONASS IN GPST seconds of week.*
- int32\_t [d\\_WN](#) {}  
*GLONASS IN GPST week number of the start of frame.*
- double [d\\_tod](#) {}  
*Time of Day since ephemeris where decoded.*

### 10.127.1 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`.

### 10.127.2 Constructor & Destructor Documentation

#### 10.127.2.1 Glonass\_Gnav\_Ephemeris()

```

Glonass_Gnav_Ephemeris::Glonass_Gnav_Ephemeris ( ) [default]

```

Default constructor

### 10.127.3 Member Function Documentation

## 10.127.3.1 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.

## 10.127.3.2 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	<i>tod_offset</i>	Is the start of day offset
in	<i>glot2utc_corr</i>	Correction from GLONASST to UTC
in	<i>glot2gpst_corr</i>	Correction from GLONASST to GPST
out	<i>WN</i>	Week Number, not in mod(1024) format
out	<i>TOW</i>	Time of Week in seconds of week

## 10.127.3.3 glot\_to\_utc()

```
boost::posix_time::ptime Glonass_Gnav_Ephemeris::glot_to_utc (
    const double offset_time,
    const double glot2utc_corr ) const
```

Converts from GLONASST to UTC.

The function simply adjust for the 6 hrs offset between GLONASST and UTC

## Parameters

in	<i>offset_time</i>	Is the start of day offset
in	<i>glot2utc_corr</i>	Correction from GLONASST to UTC

**Returns**

UTC time as a `boost::posix_time::ptime` object

**10.127.3.4 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  $t_b$  [dimensionless]

< Correction to the nth satellite time ( $t_n$ ) relative to GLONASS time ( $t_e$ )

< 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<sup>2</sup>]

< Earth-fixed acceleration coordinate y of the satellite in PZ-90.02 coordinate system [km/s<sup>2</sup>]

< Earth-fixed acceleration coordinate z of the satellite in PZ-90.02 coordinate system [km/s<sup>2</sup>]

< 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  $t_b$  [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; In = 0 indicates the n-th satellite is helthy, In = 1 indicates malfunction of this nth satellite [dimensionless]

< Health flag for nth satellite; In = 0 indicates the n-th satellite is helthy, In = 1 indicates malfunction of this nth satellite [dimensionless]

Definition at line 128 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_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_PRN`, and `i_satellite_slot_number`.

#### 10.127.3.5 `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 (IS-GPS-200K, 20.3.3.3.3.1)

### 10.127.4 Member Data Documentation

#### 10.127.4.1 `d_AXn`

```
double Glonass_Gnav_Ephemeris::d_AXn {}
```

Earth-fixed acceleration coordinate x of the satellite in PZ-90.02 coordinate system [km/s<sup>2</sup>].

Definition at line 60 of file glonass\_gnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.127.4.2 d\_AYn

```
double Glonass_Gnav_Ephemeris::d_AYn {}
```

Earth-fixed acceleration coordinate y of the satellite in PZ-90.02 coordinate system [km/s<sup>2</sup>].

Definition at line 61 of file glonass\_gnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.127.4.3 d\_AZn

```
double Glonass_Gnav_Ephemeris::d_AZn {}
```

Earth-fixed acceleration coordinate z of the satellite in PZ-90.02 coordinate system [km/s<sup>2</sup>].

Definition at line 62 of file glonass\_gnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.127.4.4 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()`.

#### 10.127.4.5 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()`.

#### 10.127.4.6 d\_dtr

```
double Glonass_Gnav_Ephemeris::d_dtr {}
```

relativistic clock correction term

Definition at line 84 of file glonass\_gnav\_ephemeris.h.

#### 10.127.4.7 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()`.

#### 10.127.4.8 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()`.

#### 10.127.4.9 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()`.

#### 10.127.4.10 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.

#### 10.127.4.11 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().

#### 10.127.4.12 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().

#### 10.127.4.13 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().

#### 10.127.4.14 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()`.

#### 10.127.4.15 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()`.

#### 10.127.4.16 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()`.

#### 10.127.4.17 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()`.

**10.127.4.18 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()`.

**10.127.4.19 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()`.

**10.127.4.20 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()`.

**10.127.4.21 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()`.

**10.127.4.22 d\_satClkDrift**

```
double Glonass_Gnav_Ephemeris::d_satClkDrift {}
```

GLONASS clock error.

Definition at line 83 of file glonass\_gnav\_ephemeris.h.

**10.127.4.23 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()`.

**10.127.4.24 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()`.

**10.127.4.25 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.

**10.127.4.26 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()`.

**10.127.4.27 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.

**10.127.4.28 d\_TOW**

```
double Glonass_Gnav_Ephemeris::d_TOW {}
```

GLONASST IN GPST seconds of week.

Definition at line 87 of file glonass\_gnav\_ephemeris.h.

**10.127.4.29 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()`.

**10.127.4.30 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()`.

**10.127.4.31 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()`.

**10.127.4.32 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.

**10.127.4.33 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()`.

**10.127.4.34 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()`.

**10.127.4.35 d\_yr**

```
double Glonass_Gnav_Ephemeris::d_yr = 1972.0
```

Current year.

Definition at line 82 of file glonass\_gnav\_ephemeris.h.

**10.127.4.36 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()`.

#### 10.127.4.37 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()`.

#### 10.127.4.38 i\_satellite\_PRN

```
uint32_t Glonass_Gnav_Ephemeris::i_satellite_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()`.

#### 10.127.4.39 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()`.

The documentation for this class was generated from the following file:

- [glonass\\_gnav\\_ephemeris.h](#)

## 10.128 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](#) ()
- 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\\_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)

### 10.128.1 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`.

### 10.128.2 Constructor & Destructor Documentation

### 10.128.2.1 Glonass\_Gnav\_Navigation\_Message()

```
Glonass_Gnav_Navigation_Message::Glonass_Gnav_Navigation_Message ( )
```

Default constructor

## 10.128.3 Member Function Documentation

### 10.128.3.1 CRC\_test()

```
bool Glonass_Gnav_Navigation_Message::CRC_test (
    std::bitset< GLONASS_GNAV_STRING_BITS > bits ) const
```

Compute CRC for GLONASS GNAV strings.

#### Parameters

<i>bits</i>	Bits of the string message where to compute CRC
-------------	---

### 10.128.3.2 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

<i>satellite_slot_number</i>	Slot number identifier for the satellite
------------------------------	--

#### Returns

Returns the [GlONASS\\_GNAV\\_Almanac](#) object for the input slot number

### 10.128.3.3 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`.

#### 10.128.3.4 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

<i>satellite_slot_number</i>	[in] Satellite slot number identifier
------------------------------	---------------------------------------

##### Returns

Frame number being decoded, 0 if operation was not successful.

#### 10.128.3.5 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`.

#### 10.128.3.6 have\_new\_almanac()

```
bool Glonass_Gnav_Navigation_Message::have_new_almanac ( )
```

Returns true if new [Glonass\\_Gnav\\_Almanac](#) object has arrived.

#### 10.128.3.7 have\_new\_ephemeris()

```
bool Glonass_Gnav_Navigation_Message::have_new_ephemeris ( )
```

Returns true if a new [Glonass\\_Gnav\\_Ephemeris](#) object has arrived.

#### 10.128.3.8 have\_new\_utc\_model()

```
bool Glonass_Gnav_Navigation_Message::have_new_utc_model ( )
```

Returns true if new [Glonass\\_Gnav\\_Utc\\_Model](#) object has arrived.

#### 10.128.3.9 string\_decoder()

```
int32_t Glonass_Gnav_Navigation_Message::string_decoder (
    const std::string & frame_string )
```

Decodes the GLONASS GNAV string.

## Parameters

<i>frame_string</i>	[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](#)

## 10.129 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*
- 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].*

### 10.129.1 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.

### 10.129.2 Constructor & Destructor Documentation

#### 10.129.2.1 Glonass\_Gnav\_Utc\_Model()

```
Glonass_Gnav_Utc_Model::Glonass_Gnav_Utc_Model ( ) [default]
```

Default constructor

### 10.129.3 Member Function Documentation

#### 10.129.3.1 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 64 of file glonass\_gnav\_utc\_model.h.

References `d_B1`, `d_B2`, `d_N_4`, `d_N_A`, `d_tau_c`, and `d_tau_gps`.

### 10.129.3.2 `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](#)

## 10.129.4 Member Data Documentation

### 10.129.4.1 `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()`.

### 10.129.4.2 `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()`.

### 10.129.4.3 `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()`.

**10.129.4.4 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()`.

**10.129.4.5 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()`.

**10.129.4.6 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()`.

The documentation for this class was generated from the following file:

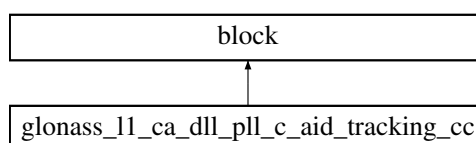
- [glonass\\_gnav\\_utc\\_model.h](#)

**10.130 glonass\_l1\_ca\_dll\_pll\_c\_aid\_tracking\_cc Class Reference**

This class implements a DLL + PLL tracking loop block.

```
#include <glonass_l1_ca_dll_pll_c_aid_tracking_cc.h>
```

Inheritance diagram for `glonass_l1_ca_dll_pll_c_aid_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

- glonass\_l1\_ca\_dll\_pll\_c\_aid\_tracking\_cc\_sptr **glonass\_l1\_ca\_dll\_pll\_c\_aid\_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 pll\_bw\_narrow\_hz, float dll\_bw\_narrow\_hz, int32\_t extend\_correlation\_ms, float early\_late\_space\_chips)

### 10.130.1 Detailed Description

This class implements a DLL + PLL tracking loop block.

Definition at line 68 of file `glonass_l1_ca_dll_pll_c_aid_tracking_cc.h`.

The documentation for this class was generated from the following file:

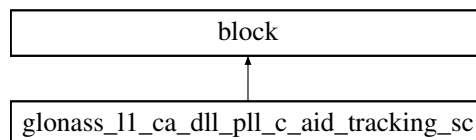
- [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_cc.h](#)

## 10.131 glonass\_l1\_ca\_dll\_pll\_c\_aid\_tracking\_sc Class Reference

This class implements a DLL + PLL tracking loop block.

```
#include <glonass_l1_ca_dll_pll_c_aid_tracking_sc.h>
```

Inheritance diagram for `glonass_l1_ca_dll_pll_c_aid_tracking_sc`:



## 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

- `glonass_l1_ca_dll_pll_c_aid_tracking_sc_sptr glonass_l1_ca_dll_pll_c_aid_make_tracking_sc` (`int64_t fs_in`, `uint32_t vector_length`, `bool dump`, `const std::string &dump_filename`, `float pll_bw_hz`, `float dll_bw_hz`, `float pll_bw_narrow_hz`, `float dll_bw_narrow_hz`, `int32_t extend_correlation_ms`, `float early_late_space_chips`)

### 10.131.1 Detailed Description

This class implements a DLL + PLL tracking loop block.

Definition at line 69 of file `glonass_l1_ca_dll_pll_c_aid_tracking_sc.h`.

The documentation for this class was generated from the following file:

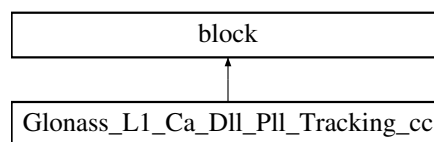
- [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_sc.h](#)

## 10.132 Glonass\_L1\_Ca\_Dll\_Pll\_Tracking\_cc Class Reference

This class implements a DLL + PLL tracking loop block.

```
#include <glonass_l1_ca_dll_pll_tracking_cc.h>
```

Inheritance diagram for `Glonass_L1_Ca_Dll_Pll_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

- `glonass_l1_ca_dll_pll_tracking_cc_sptr glonass_l1_ca_dll_pll_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`)

### 10.132.1 Detailed Description

This class implements a DLL + PLL tracking loop block.

Definition at line 63 of file `glonass_l1_ca_dll_pll_tracking_cc.h`.

The documentation for this class was generated from the following file:

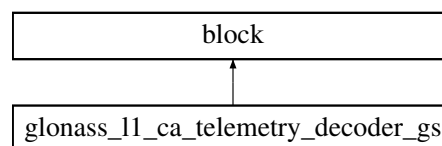
- [glonass\\_l1\\_ca\\_dll\\_pll\\_tracking\\_cc.h](#)

## 10.133 glonass\_l1\_ca\_telemetry\_decoder\_gs Class Reference

This class implements a block that decodes the GNAV data defined in GLONASS ICD v5.1.

```
#include <glonass_l1_ca_telemetry_decoder_gs.h>
```

Inheritance diagram for `glonass_l1_ca_telemetry_decoder_gs`:



### Public Member Functions

- [~glonass\\_l1\\_ca\\_telemetry\\_decoder\\_gs](#) ()  
*Class destructor.*
- void [set\\_satellite](#) (const [Gnss\\_Satellite](#) &satellite)  
*Set satellite PRN.*
- void [set\\_channel](#) (int32\_t channel)  
*Set receiver's channel.*
- void **reset** ()
- int [general\\_work](#) (int noutput\_items, gr\_vector\_int &ninput\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)  
*This is where all signal processing takes place.*

### Friends

- `glonass_l1_ca_telemetry_decoder_gs_sptr` **glonass\_l1\_ca\_make\_telemetry\_decoder\_gs** (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)

### 10.133.1 Detailed Description

This class implements a block that decodes the GNAV data defined in GLONASS ICD v5.1.

#### Note

Code added as part of GSoC 2017 program

#### See also

[GLONASS ICD](#)

Definition at line 56 of file `glonass_l1_ca_telemetry_decoder_gs.h`.

### 10.133.2 Constructor & Destructor Documentation

#### 10.133.2.1 `~glonass_l1_ca_telemetry_decoder_gs()`

```
glonass_l1_ca_telemetry_decoder_gs::~~glonass_l1_ca_telemetry_decoder_gs ( )
```

Class destructor.

### 10.133.3 Member Function Documentation

#### 10.133.3.1 `general_work()`

```
int glonass_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 )
```

This is where all signal processing takes place.

#### 10.133.3.2 `set_channel()`

```
void glonass_l1_ca_telemetry_decoder_gs::set_channel (
    int32_t channel )
```

Set receiver's channel.

### 10.133.3.3 set\_satellite()

```
void glonass_l1_ca_telemetry_decoder_gs::set_satellite (
    const Gnss_Satellite & satellite )
```

Set satellite PRN.

The documentation for this class was generated from the following file:

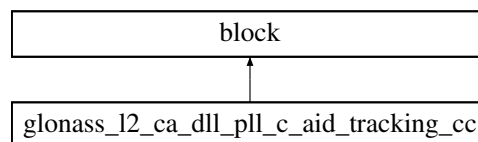
- [glonass\\_l1\\_ca\\_telemetry\\_decoder\\_gs.h](#)

## 10.134 glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking\_cc Class Reference

This class implements a DLL + PLL tracking loop block.

```
#include <glonass_l2_ca_dll_pll_c_aid_tracking_cc.h>
```

Inheritance diagram for `glonass_l2_ca_dll_pll_c_aid_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

- `glonass_l2_ca_dll_pll_c_aid_tracking_cc_sptr` **glonass\_l2\_ca\_dll\_pll\_c\_aid\_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 pll\_bw\_narrow\_hz, float dll\_bw\_narrow\_hz, int32\_t extend\_correlation\_ms, float early\_late\_space\_chips)

### 10.134.1 Detailed Description

This class implements a DLL + PLL tracking loop block.

Definition at line 65 of file `glonass_l2_ca_dll_pll_c_aid_tracking_cc.h`.

The documentation for this class was generated from the following file:

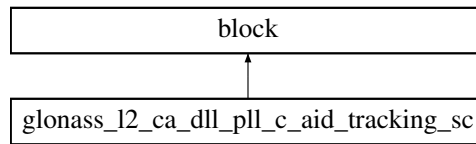
- [glonass\\_l2\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_cc.h](#)

## 10.135 glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking\_sc Class Reference

This class implements a DLL + PLL tracking loop block.

```
#include <glonass_l2_ca_dll_pll_c_aid_tracking_sc.h>
```

Inheritance diagram for glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking\_sc:



### 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

- glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking\_sc\_sptr **glonass\_l2\_ca\_dll\_pll\_c\_aid\_make\_tracking\_sc** (int64\_t fs\_in, uint32\_t vector\_length, bool dump, const std::string &dump\_filename, float pll\_bw\_hz, float dll\_bw\_hz, float pll\_bw\_narrow\_hz, float dll\_bw\_narrow\_hz, int32\_t extend\_correlation\_ms, float early\_late\_space\_chips)

### 10.135.1 Detailed Description

This class implements a DLL + PLL tracking loop block.

Definition at line 65 of file glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking\_sc.h.

The documentation for this class was generated from the following file:

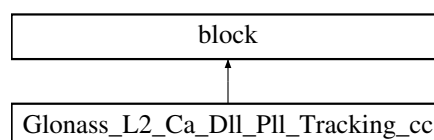
- [glonass\\_l2\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_sc.h](#)

## 10.136 Glonass\_L2\_Ca\_Dll\_Pll\_Tracking\_cc Class Reference

This class implements a DLL + PLL tracking loop block.

```
#include <glonass_l2_ca_dll_pll_tracking_cc.h>
```

Inheritance diagram for Glonass\_L2\_Ca\_Dll\_Pll\_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

- glonass\_l2\_ca\_dll\_pll\_tracking\_cc\_sptr **glonass\_l2\_ca\_dll\_pll\_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)

### 10.136.1 Detailed Description

This class implements a DLL + PLL tracking loop block.

Definition at line 60 of file `glonass_l2_ca_dll_pll_tracking_cc.h`.

The documentation for this class was generated from the following file:

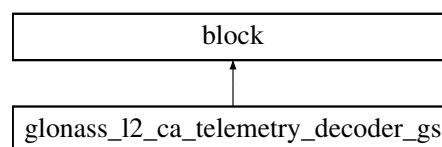
- [glonass\\_l2\\_ca\\_dll\\_pll\\_tracking\\_cc.h](#)

## 10.137 glonass\_l2\_ca\_telemetry\_decoder\_gs Class Reference

This class implements a block that decodes the GNAV data defined in GLONASS ICD v5.1.

```
#include <glonass_l2_ca_telemetry_decoder_gs.h>
```

Inheritance diagram for `glonass_l2_ca_telemetry_decoder_gs`:



## Public Member Functions

- [~glonass\\_l2\\_ca\\_telemetry\\_decoder\\_gs](#) ()  
*Class destructor.*
- void **set\_satellite** (const [Gnss\\_Satellite](#) &satellite)  
*Set satellite PRN.*
- void **set\_channel** (int32\_t channel)  
*Set receiver's channel.*
- void **reset** ()
- int **general\_work** (int noutput\_items, gr\_vector\_int &ninput\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)  
*This is where all signal processing takes place.*

## Friends

- `glonass_l2_ca_telemetry_decoder_gs_sptr` **`glonass_l2_ca_make_telemetry_decoder_gs`** (const [Gnss\\_Satellite](#) &satellite, const [TIm\\_Conf](#) &conf)

### 10.137.1 Detailed Description

This class implements a block that decodes the GNAV data defined in GLONASS ICD v5.1.

#### See also

[GLONASS ICD](#)

Definition at line 54 of file `glonass_l2_ca_telemetry_decoder_gs.h`.

### 10.137.2 Constructor & Destructor Documentation

#### 10.137.2.1 `~glonass_l2_ca_telemetry_decoder_gs()`

```
glonass_l2_ca_telemetry_decoder_gs::~glonass_l2_ca_telemetry_decoder_gs ( )
```

Class destructor.

### 10.137.3 Member Function Documentation

#### 10.137.3.1 `general_work()`

```
int glonass_l2_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 )
```

This is where all signal processing takes place.

#### 10.137.3.2 `set_channel()`

```
void glonass_l2_ca_telemetry_decoder_gs::set_channel (
    int32_t channel )
```

Set receiver's channel.

### 10.137.3.3 set\_satellite()

```
void glonass_l2_ca_telemetry_decoder_gs::set_satellite (
    const Gnss_Satellite & satellite )
```

Set satellite PRN.

The documentation for this class was generated from the following file:

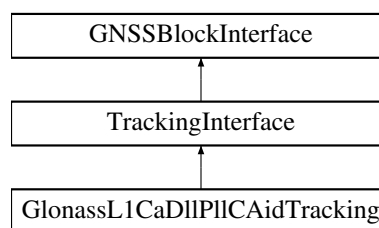
- [glonass\\_l2\\_ca\\_telemetry\\_decoder\\_gs.h](#)

## 10.138 GlonassL1CaDllPllCAidTracking Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#include <glonass_l1_ca_dll_pll_c_aid_tracking.h>
```

Inheritance diagram for GlonassL1CaDllPllCAidTracking:



### Public Member Functions

- **GlonassL1CaDllPllCAidTracking** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- std::string **role** () override
- std::string **implementation** () override
  - Returns "GLONASS\_L1\_CA\_DLL\_PLL\_C\_Aid\_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.*

### 10.138.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 45 of file `glonass_l1_ca_dll_pll_c_aid_tracking.h`.

### 10.138.2 Member Function Documentation

#### 10.138.2.1 `implementation()`

```
std::string GlonassL1CaDllPllCAidTracking::implementation ( ) [inline], [override], [virtual]
```

Returns "GLONASS\_L1\_CA\_DLL\_PLL\_C\_Aid\_Tracking".

Implements [GNSSBlockInterface](#).

Definition at line 62 of file `glonass_l1_ca_dll_pll_c_aid_tracking.h`.

#### 10.138.2.2 `set_channel()`

```
void GlonassL1CaDllPllCAidTracking::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.138.2.3 `set_gnss_synchro()`

```
void GlonassL1CaDllPllCAidTracking::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](#).

#### 10.138.2.4 stop\_tracking()

```
void GlonassL1CaDllPllCAidTracking::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

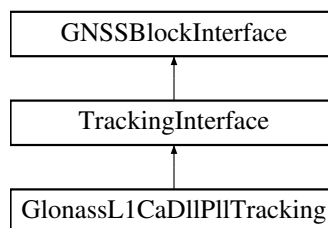
- [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking.h](#)

### 10.139 GlonassL1CaDllPllTracking Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#include <glonass_l1_ca_dll_pll_tracking.h>
```

Inheritance diagram for GlonassL1CaDllPllTracking:



#### Public Member Functions

- **GlonassL1CaDllPllTracking** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- std::string **role** () override
- std::string **implementation** () override  
*Returns "GLONASS\_L1\_CA\_DLL\_PLL\_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.*

### 10.139.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 43 of file `glonass_l1_ca_dll_pll_tracking.h`.

### 10.139.2 Member Function Documentation

#### 10.139.2.1 `implementation()`

```
std::string GlonassL1CaDllPllTracking::implementation ( ) [inline], [override], [virtual]
```

Returns "GLONASS\_L1\_CA\_DLL\_PLL\_Tracking".

Implements [GNSSBlockInterface](#).

Definition at line 60 of file `glonass_l1_ca_dll_pll_tracking.h`.

#### 10.139.2.2 `set_channel()`

```
void GlonassL1CaDllPllTracking::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.139.2.3 `set_gnss_synchro()`

```
void GlonassL1CaDllPllTracking::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](#).

### 10.139.2.4 stop\_tracking()

```
void GlonassL1CaDllPllTracking::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

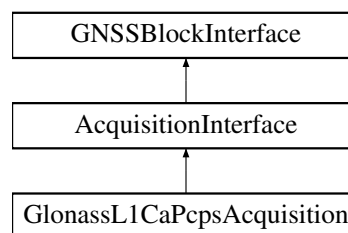
- [glonass\\_l1\\_ca\\_dll\\_pll\\_tracking.h](#)

## 10.140 GlonassL1CaPcpsAcquisition Class Reference

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

```
#include <glonass_l1_ca_pcps_acquisition.h>
```

Inheritance diagram for GlonassL1CaPcpsAcquisition:



### Public Member Functions

- **GlonassL1CaPcpsAcquisition** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- std::string **role** () override
- std::string **implementation** () override  
*Returns "GLONASS\_L1\_CA\_PCPS\_Acquisition".*
- 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\_threshold** (float threshold) override  
*Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override

- *Set maximum Doppler off grid search.*
- void [set\\_doppler\\_step](#) (unsigned int doppler\_step) override
- *Set Doppler steps for the grid search.*
- void [init](#) () override
- *Initializes acquisition algorithm.*
- void [set\\_local\\_code](#) () override
- *Sets local code for GPS L1/CA PCPS acquisition algorithm.*
- signed int [mag](#) () override
- *Returns the maximum peak of grid search.*
- void [reset](#) () override
- *Restart acquisition algorithm.*
- void [set\\_state](#) (int state) override
- *If state = 1, it forces the block to start acquiring from the first sample.*
- void [stop\\_acquisition](#) () override
- *Stop running acquisition.*
- void [set\\_resampler\\_latency](#) (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override

### 10.140.1 Detailed Description

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

Definition at line 45 of file `glonass_l1_ca_pcps_acquisition.h`.

### 10.140.2 Member Function Documentation

#### 10.140.2.1 implementation()

```
std::string GlonassL1CaPcpsAcquisition::implementation ( ) [inline], [override], [virtual]
```

Returns "GLONASS\_L1\_CA\_PCPS\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 64 of file `glonass_l1_ca_pcps_acquisition.h`.

#### 10.140.2.2 init()

```
void GlonassL1CaPcpsAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.140.2.3 mag()

```
signed int GlonassL1CaPcpsAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

#### 10.140.2.4 reset()

```
void GlonassL1CaPcpsAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.140.2.5 set\_channel()

```
void GlonassL1CaPcpsAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 89 of file glonass\_l1\_ca\_pcps\_acquisition.h.

#### 10.140.2.6 set\_channel\_fsm()

```
void GlonassL1CaPcpsAcquisition::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 98 of file glonass\_l1\_ca\_pcps\_acquisition.h.

#### 10.140.2.7 set\_doppler\_max()

```
void GlonassL1CaPcpsAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.140.2.8 set\_doppler\_step()

```
void GlonassL1CaPcpsAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.140.2.9 set\_gnss\_synchro()

```
void GlonassL1CaPcpsAcquisition::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](#).

#### 10.140.2.10 set\_local\_code()

```
void GlonassL1CaPcpsAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local code for GPS L1/CA PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.140.2.11 set\_state()

```
void GlonassL1CaPcpsAcquisition::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

#### 10.140.2.12 `set_threshold()`

```
void GlonassL1CaPcpsAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

#### 10.140.2.13 `stop_acquisition()`

```
void GlonassL1CaPcpsAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

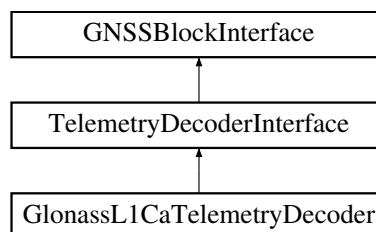
- [glonass\\_l1\\_ca\\_pcps\\_acquisition.h](#)

## 10.141 GlonassL1CaTelemetryDecoder Class Reference

This class implements a NAV data decoder for GLONASS L1 C/A.

```
#include <glonass_l1_ca_telemetry_decoder.h>
```

Inheritance diagram for GlonassL1CaTelemetryDecoder:



### Public Member Functions

- **GlonassL1CaTelemetryDecoder** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- 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
- void **set\_channel** (int channel) override
- std::string **role** () override
- std::string **implementation** () override
- *Returns "GLONASS\_L1\_CA\_Telemetry\_Decoder".*
- void **reset** () override
- size\_t **item\_size** () override

### 10.141.1 Detailed Description

This class implements a NAV data decoder for GLONASS L1 C/A.

Definition at line 43 of file `glonass_l1_ca_telemetry_decoder.h`.

### 10.141.2 Member Function Documentation

#### 10.141.2.1 `implementation()`

```
std::string GlonassL1CaTelemetryDecoder::implementation ( ) [inline], [override], [virtual]
```

Returns "GLONASS\_L1\_CA\_Telemetry\_Decoder".

Implements [GNSSBlockInterface](#).

Definition at line 68 of file `glonass_l1_ca_telemetry_decoder.h`.

The documentation for this class was generated from the following file:

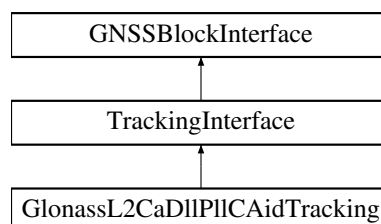
- [glonass\\_l1\\_ca\\_telemetry\\_decoder.h](#)

## 10.142 GlonassL2CaDIIPICAidTracking Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#include <glonass_l2_ca_dll_pll_c_aid_tracking.h>
```

Inheritance diagram for `GlonassL2CaDIIPICAidTracking`:



## Public Member Functions

- **GlonassL2CaDllPllCAidTracking** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- std::string **role** () override
- std::string **implementation** () override  
*Returns "GLONASS\_L2\_CA\_DLL\_PLL\_C\_Aid\_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.*

### 10.142.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 43 of file `glonass_l2_ca_dll_pll_c_aid_tracking.h`.

### 10.142.2 Member Function Documentation

#### 10.142.2.1 implementation()

```
std::string GlonassL2CaDllPllCAidTracking::implementation ( ) [inline], [override], [virtual]
```

Returns "GLONASS\_L2\_CA\_DLL\_PLL\_C\_Aid\_Tracking".

Implements [GNSSBlockInterface](#).

Definition at line 60 of file `glonass_l2_ca_dll_pll_c_aid_tracking.h`.

#### 10.142.2.2 set\_channel()

```
void GlonassL2CaDllPllCAidTracking::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

10.142.2.3 `set_gnss_synchro()`

```
void GlonassL2CaDllPllCAidTracking::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](#).

10.142.2.4 `stop_tracking()`

```
void GlonassL2CaDllPllCAidTracking::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

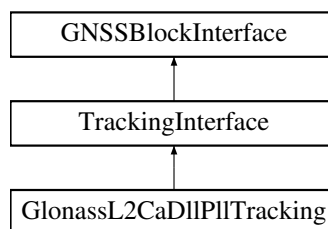
- [glonass\\_l2\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking.h](#)

## 10.143 GlonassL2CaDllPllCAidTracking Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#include <glonass_l2_ca_dll_pll_tracking.h>
```

Inheritance diagram for GlonassL2CaDllPllCAidTracking:



## Public Member Functions

- **GlonassL2CaDllPllCAidTracking** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- std::string **role** () override
- std::string **implementation** () override
  - Returns "GLONASS\_L1\_CA\_DLL\_PLL\_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.

### 10.143.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 42 of file `glonass_l2_ca_dll_pll_tracking.h`.

### 10.143.2 Member Function Documentation

#### 10.143.2.1 `implementation()`

```
std::string GlonassL2CaDllPllTracking::implementation ( ) [inline], [override], [virtual]
```

Returns "GLONASS\_L1\_CA\_DLL\_PLL\_Tracking".

Implements [GNSSBlockInterface](#).

Definition at line 59 of file `glonass_l2_ca_dll_pll_tracking.h`.

#### 10.143.2.2 `set_channel()`

```
void GlonassL2CaDllPllTracking::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.143.2.3 `set_gnss_synchro()`

```
void GlonassL2CaDllPllTracking::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](#).

## 10.143.2.4 stop\_tracking()

```
void GlonassL2CaDllPllTracking::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

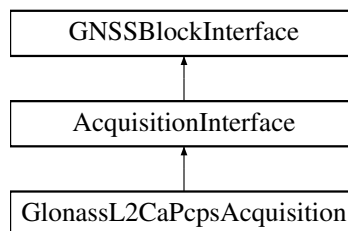
- [glonass\\_l2\\_ca\\_dll\\_pll\\_tracking.h](#)

## 10.144 GlonassL2CaPcpsAcquisition Class Reference

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GLONASS L2 C/A signals.

```
#include <glonass_l2_ca_pcps_acquisition.h>
```

Inheritance diagram for GlonassL2CaPcpsAcquisition:



## Public Member Functions

- **GlonassL2CaPcpsAcquisition** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- std::string **role** () override
- std::string **implementation** () override  
*Returns "GLONASS\_L2\_CA\_PCPS\_Acquisition".*
- 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\_threshold** (float threshold) override  
*Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override

- *Set maximum Doppler off grid search.*  
• void [set\\_doppler\\_step](#) (unsigned int doppler\_step) override
- *Set Doppler steps for the grid search.*  
• void [init](#) () override
- *Initializes acquisition algorithm.*  
• void [set\\_local\\_code](#) () override
- *Sets local code for GLONASS L2/CA PCPS acquisition algorithm.*  
• signed int [mag](#) () override
- *Returns the maximum peak of grid search.*  
• void [reset](#) () override
- *Restart acquisition algorithm.*  
• void [set\\_state](#) (int state) override
- *If state = 1, it forces the block to start acquiring from the first sample.*  
• void [stop\\_acquisition](#) () override
- *Stop running acquisition.*  
• void [set\\_resampler\\_latency](#) (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override

### 10.144.1 Detailed Description

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GLONASS L2 C/A signals.

Definition at line 44 of file `glonass_l2_ca_pcps_acquisition.h`.

### 10.144.2 Member Function Documentation

#### 10.144.2.1 implementation()

```
std::string GlonassL2CaPcpsAcquisition::implementation ( ) [inline], [override], [virtual]
```

Returns "GLONASS\_L2\_CA\_PCPS\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 63 of file `glonass_l2_ca_pcps_acquisition.h`.

#### 10.144.2.2 init()

```
void GlonassL2CaPcpsAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.144.2.3 mag()

```
signed int GlonassL2CaPcpsAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

### 10.144.2.4 reset()

```
void GlonassL2CaPcpsAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.144.2.5 set\_channel()

```
void GlonassL2CaPcpsAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 88 of file `glonass_l2_ca_pcps_acquisition.h`.

### 10.144.2.6 set\_channel\_fsm()

```
void GlonassL2CaPcpsAcquisition::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 97 of file `glonass_l2_ca_pcps_acquisition.h`.

#### 10.144.2.7 set\_doppler\_max()

```
void GlonassL2CaPcpsAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.144.2.8 set\_doppler\_step()

```
void GlonassL2CaPcpsAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.144.2.9 set\_gnss\_synchro()

```
void GlonassL2CaPcpsAcquisition::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](#).

#### 10.144.2.10 set\_local\_code()

```
void GlonassL2CaPcpsAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local code for GLONASS L2/CA PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.144.2.11 set\_state()

```
void GlonassL2CaPcpsAcquisition::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

10.144.2.12 `set_threshold()`

```
void GlonassL2CaPcpsAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

10.144.2.13 `stop_acquisition()`

```
void GlonassL2CaPcpsAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

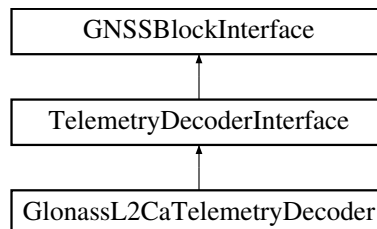
- [glonass\\_l2\\_ca\\_pcps\\_acquisition.h](#)

## 10.145 GlonassL2CaTelemetryDecoder Class Reference

This class implements a NAV data decoder for GLONASS L2 C/A.

```
#include <glonass_l2_ca_telemetry_decoder.h>
```

Inheritance diagram for GlonassL2CaTelemetryDecoder:



## Public Member Functions

- **GlonassL2CaTelemetryDecoder** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- 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
- void **set\_channel** (int channel) override
- std::string **role** () override
- std::string **implementation** () override
- *Returns "GLONASS\_L2\_CA\_Telemetry\_Decoder".*
- void **reset** () override
- size\_t **item\_size** () override

### 10.145.1 Detailed Description

This class implements a NAV data decoder for GLONASS L2 C/A.

Definition at line 42 of file `glonass_l2_ca_telemetry_decoder.h`.

### 10.145.2 Member Function Documentation

#### 10.145.2.1 `implementation()`

```
std::string GlonassL2CaTelemetryDecoder::implementation ( ) [inline], [override], [virtual]
```

Returns "GLONASS\_L2\_CA\_Telemetry\_Decoder".

Implements [GNSSBlockInterface](#).

Definition at line 67 of file `glonass_l2_ca_telemetry_decoder.h`.

The documentation for this class was generated from the following file:

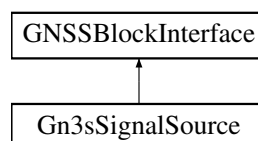
- [glonass\\_l2\\_ca\\_telemetry\\_decoder.h](#)

## 10.146 Gn3sSignalSource Class Reference

This class reads samples from a GN3S USB dongle, a RF front-end signal sampler.

```
#include <gn3s_signal_source.h>
```

Inheritance diagram for Gn3sSignalSource:



### Public Member Functions

- **Gn3sSignalSource** (const [ConfigurationInterface](#) \*configuration, 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 "Gn3s\_Signal\_Source".*
- 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

### 10.146.1 Detailed Description

This class reads samples from a GN3S USB dongle, a RF front-end signal sampler.

Definition at line 41 of file gn3s\_signal\_source.h.

### 10.146.2 Member Function Documentation

#### 10.146.2.1 implementation()

```
std::string Gn3sSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "Gn3s\_Signal\_Source".

Implements [GNSSBlockInterface](#).

Definition at line 58 of file gn3s\_signal\_source.h.

The documentation for this class was generated from the following file:

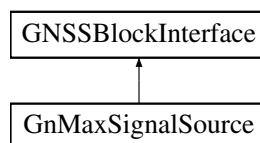
- [gn3s\\_signal\\_source.h](#)

## 10.147 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, 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 "GNMAX\_Signal\_Source".*
- 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

### 10.147.1 Detailed Description

This class reads samples from a gnMAX2769 USB dongle, a RF front-end signal sampler.

Definition at line 38 of file gnmax\_signal\_source.h.

### 10.147.2 Member Function Documentation

#### 10.147.2.1 implementation()

```
std::string GnMaxSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "GNMAX\_Signal\_Source".

Implements [GNSSBlockInterface](#).

Definition at line 55 of file gnmax\_signal\_source.h.

The documentation for this class was generated from the following file:

- [gnmax\\_signal\\_source.h](#)

## 10.148 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)*

### 10.148.1 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](#)

## 10.149 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](#) &)  
*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 when information 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"}.*
- std::string [get\\_system\\_short](#) () const  
*Gets the satellite system {"G", "R", "SBAS", "E", "C"}.*
- 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*

### 10.149.1 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`.

### 10.149.2 Constructor & Destructor Documentation

#### 10.149.2.1 `Gnss_Satellite()` [1/4]

```
Gnss_Satellite::Gnss_Satellite ( ) [default]
```

Default Constructor.

#### 10.149.2.2 `Gnss_Satellite()` [2/4]

```
Gnss_Satellite::Gnss_Satellite (
    const std::string & system_,
    uint32_t PRN_ )
```

Concrete GNSS satellite Constructor.

#### 10.149.2.3 `~Gnss_Satellite()`

```
Gnss_Satellite::~Gnss_Satellite ( ) [default]
```

Default Destructor.

#### 10.149.2.4 Gnss\_Satellite() [3/4]

```
Gnss_Satellite::Gnss_Satellite (
    const Gnss_Satellite & other ) [noexcept]
```

Copy constructor.

#### 10.149.2.5 Gnss\_Satellite() [4/4]

```
Gnss_Satellite::Gnss_Satellite (
    Gnss_Satellite && other ) [noexcept]
```

Move constructor.

### 10.149.3 Member Function Documentation

#### 10.149.3.1 get\_block()

```
std::string Gnss_Satellite::get_block ( ) const
```

Gets the satellite block. If GPS, returns {"IIA", "IIR", "IIR-M", "IIF"}.

#### 10.149.3.2 get\_PRN()

```
uint32_t Gnss_Satellite::get_PRN ( ) const
```

Gets satellite's PRN.

#### 10.149.3.3 get\_rf\_link()

```
int32_t Gnss_Satellite::get_rf_link ( ) const
```

Gets the satellite's rf link.

#### 10.149.3.4 get\_system()

```
std::string Gnss_Satellite::get_system ( ) const
```

Gets the satellite system {"GPS", "GLONASS", "SBAS", "Galileo", "Beidou"}.

#### 10.149.3.5 get\_system\_short()

```
std::string Gnss_Satellite::get_system_short ( ) const
```

Gets the satellite system {"G", "R", "SBAS", "E", "C"}.

#### 10.149.3.6 operator=() [1/2]

```
Gnss_Satellite& Gnss_Satellite::operator= (
    const Gnss_Satellite & )
```

Copy assignment operator.

#### 10.149.3.7 operator=() [2/2]

```
Gnss_Satellite& Gnss_Satellite::operator= (
    Gnss_Satellite && other ) [noexcept]
```

Move assignment operator.

#### 10.149.3.8 update\_PRN()

```
void Gnss_Satellite::update_PRN (
    uint32_t PRN )
```

Updates the PRN Number when information is decoded, only applies to GLONASS GNAV messages.

#### 10.149.3.9 what\_block()

```
std::string Gnss_Satellite::what_block (
    const std::string & system_,
    uint32_t PRN_ )
```

Gets the block of a given satellite.

### 10.149.4 Friends And Related Function Documentation

#### 10.149.4.1 operator<<

```
std::ostream& operator<< (
    std::ostream & ,
    const Gnss_Satellite & ) [friend]
```

operator<< for pretty printing

#### 10.149.4.2 operator==

```
bool operator== (
    const Gnss_Satellite & ,
    const Gnss_Satellite & ) [friend]
```

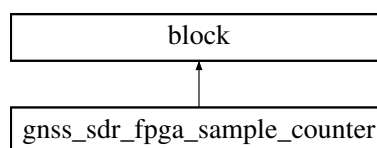
operator== for comparison

The documentation for this class was generated from the following file:

- [gnss\\_satellite.h](#)

## 10.150 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)

### 10.150.1 Detailed Description

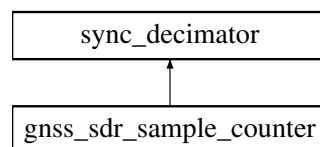
Definition at line 40 of file `gnss_sdr_fpga_sample_counter.h`.

The documentation for this class was generated from the following file:

- [gnss\\_sdr\\_fpga\\_sample\\_counter.h](#)

## 10.151 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)

### 10.151.1 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](#)

## 10.152 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**

### 10.152.1 Detailed Description

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

Definition at line 55 of file gnss\_sdr\_supl\_client.h.

### 10.152.2 Member Function Documentation

**10.152.2.1 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.

**10.152.2.2 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.

**10.152.2.3 load\_ephemeris\_xml()**

```
bool Gnss_Sdr_Supl_Client::load_ephemeris_xml (
    const std::string & file_name )
```

Read GPS NAV ephemeris map from XML file.

**10.152.2.4 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.

**10.152.2.5 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.

**10.152.2.6 load\_gal\_iono\_xml()**

```
bool Gnss_Sdr_Supl_Client::load_gal_iono_xml (
    const std::string & file_name )
```

Read Galileo iono from XML file.

#### 10.152.2.7 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.

#### 10.152.2.8 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.

#### 10.152.2.9 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.

#### 10.152.2.10 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.

#### 10.152.2.11 load\_iono\_xml()

```
bool Gnss_Sdr_Supl_Client::load_iono_xml (
    const std::string & file_name )
```

Read iono from XML file.

#### 10.152.2.12 load\_ref\_location\_xml()

```
bool Gnss_Sdr_Supl_Client::load_ref_location_xml (
    const std::string & file_name )
```

Read ref location from XML file.

#### 10.152.2.13 load\_ref\_time\_xml()

```
bool Gnss_Sdr_Supl_Client::load_ref_time_xml (
    const std::string & file_name )
```

Read ref time from XML file.

#### 10.152.2.14 load\_utc\_xml()

```
bool Gnss_Sdr_Supl_Client::load_utc_xml (
    const std::string & file_name )
```

Read GPS utc model from XML file.

#### 10.152.2.15 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.

#### 10.152.2.16 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.

#### 10.152.2.17 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.

#### 10.152.2.18 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.

#### 10.152.2.19 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.

#### 10.152.2.20 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.

#### 10.152.2.21 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.

#### 10.152.2.22 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.

**10.152.2.23 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.

**10.152.2.24 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.

**10.152.2.25 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.

**10.152.2.26 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.

**10.152.2.27 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.

### 10.152.2.28 save\_utc\_xml()

```
bool Gnss_Sdr_Supl_Client::save_utc_xml (
    const std::string & file_name,
    Gps_Utc_Model & utc )
```

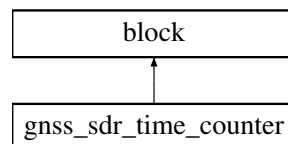
Save UTC model map to XML file.

The documentation for this class was generated from the following file:

- [gnss\\_sdr\\_supl\\_client.h](#)

## 10.153 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** ()

### 10.153.1 Detailed Description

Definition at line 38 of file gnss\_sdr\_time\_counter.h.

The documentation for this class was generated from the following file:

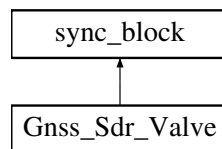
- [gnss\\_sdr\\_time\\_counter.h](#)

## 10.154 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)

### 10.154.1 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 54 of file gnss\_sdr\_valve.h.

The documentation for this class was generated from the following file:

- [gnss\\_sdr\\_valve.h](#)

## 10.155 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  
*Get the satellite signal {"1C" for GPS L1 C/A, "2S" for GPS L2C (M), "L5" for GPS L5, "1G" for GLONASS L1 C/A, "1B" for Galileo E1B, "5X" for Galileo E5a.*
- [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*

### 10.155.1 Detailed Description

This class represents a GNSS signal.

It contains information about the space vehicle and the specific signal.

Definition at line 37 of file gnss\_signal.h.

### 10.155.2 Member Function Documentation

#### 10.155.2.1 [get\\_satellite\(\)](#)

```
Gnss\_Satellite Gnss_Signal::get_satellite ( ) const
```

Get the [Gnss\\_Satellite](#) associated to the signal.

#### 10.155.2.2 [get\\_signal\\_str\(\)](#)

```
std::string Gnss_Signal::get_signal_str ( ) const
```

Get the satellite signal {"1C" for GPS L1 C/A, "2S" for GPS L2C (M), "L5" for GPS L5, "1G" for GLONASS L1 C/A, "1B" for Galileo E1B, "5X" for Galileo E5a.

### 10.155.3 Friends And Related Function Documentation

## 10.155.3.1 operator&lt;&lt;

```
std::ostream& operator<< (
    std::ostream & ,
    const Gnss_Signal & ) [friend]
```

operator<< for pretty printing

## 10.155.3.2 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](#)

## 10.156 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  
*Copy constructor.*
- [Gnss\\_Synchro](#) & operator= (const [Gnss\\_Synchro](#) &rhs) noexcept  
*Copy assignment operator.*
- [Gnss\\_Synchro](#) ([Gnss\\_Synchro](#) &&other) noexcept  
*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.*

### 10.156.1 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.

### 10.156.2 Constructor & Destructor Documentation

#### 10.156.2.1 Gnss\_Synchro() [1/3]

```
Gnss_Synchro::Gnss_Synchro ( ) [default]
```

Default constructor.

#### 10.156.2.2 ~Gnss\_Synchro()

```
Gnss_Synchro::~~Gnss_Synchro ( ) [default]
```

Default destructor.

#### 10.156.2.3 Gnss\_Synchro() [2/3]

```
Gnss_Synchro::Gnss_Synchro (
    const Gnss_Synchro & other ) [inline], [noexcept]
```

Copy constructor.

Definition at line 83 of file gnss\_synchro.h.

#### 10.156.2.4 Gnss\_Synchro() [3/3]

```
Gnss_Synchro::Gnss_Synchro (
    Gnss_Synchro && other ) [inline], [noexcept]
```

Move constructor.

Definition at line 126 of file gnss\_synchro.h.

### 10.156.3 Member Function Documentation

#### 10.156.3.1 `operator=()` [1/2]

```
Gnss_Synchro& Gnss_Synchro::operator= (
    const Gnss_Synchro & rhs ) [inline], [noexcept]
```

Copy assignment operator.

Definition at line 89 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_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`.

#### 10.156.3.2 `operator=()` [2/2]

```
Gnss_Synchro& Gnss_Synchro::operator= (
    Gnss_Synchro && other ) [inline], [noexcept]
```

Move assignment operator.

Definition at line 132 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_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`.

#### 10.156.3.3 `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 173 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_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`.

## 10.156.4 Member Data Documentation

### 10.156.4.1 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

### 10.156.4.2 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

### 10.156.4.3 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

### 10.156.4.4 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.5 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.6 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 operator=(), and serialize().

#### 10.156.4.7 Channel\_ID

```
int32_t Gnss_Synchro::Channel_ID {}
```

Set by [Channel](#) constructor.

Definition at line 49 of file gnss\_synchro.h.

Referenced by operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.8 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.9 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 operator=(), and serialize().

#### 10.156.4.10 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.11 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.12 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.13 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.14 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.15 fs

```
int64_t Gnss_Synchro::fs {}
```

Set by Tracking processing block.

Definition at line 58 of file gnss\_synchro.h.

Referenced by operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.16 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.17 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.18 Prompt\_I

```
double Gnss_Synchro::Prompt_I {}
```

Set by Tracking processing block.

Definition at line 59 of file gnss\_synchro.h.

Referenced by operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.19 Prompt\_Q

```
double Gnss_Synchro::Prompt_Q {}
```

Set by Tracking processing block.

Definition at line 60 of file gnss\_synchro.h.

Referenced by operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.20 Pseudorange\_m

```
double Gnss_Synchro::Pseudorange_m {}
```

Set by Observables processing block.

Definition at line 72 of file gnss\_synchro.h.

Referenced by operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.21 RX\_time

```
double Gnss_Synchro::RX_time {}
```

Set by Observables processing block.

Definition at line 73 of file gnss\_synchro.h.

Referenced by operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.22 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.23 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

#### 10.156.4.24 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

## 10.156.4.25 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 operator=(), Serdes\_Gnss\_Synchro::readProtobuffer(), and serialize().

The documentation for this class was generated from the following file:

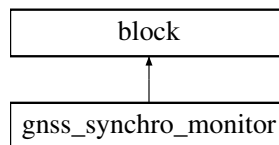
- [gnss\\_synchro.h](#)

## 10.157 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, int udp\_port, const std::vector< std::string > &udp\_addresses, bool enable\_protobuf)

## 10.157.1 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.

## 10.157.2 Constructor & Destructor Documentation

### 10.157.2.1 ~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](#)

## 10.158 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 uint16\_t &port, bool enable\_protobuf)
- bool **write\_gnss\_synchro** (const std::vector< [Gnss\\_Synchro](#) > &stocks)

### 10.158.1 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](#)

## 10.159 GNSSBlockFactory Class Reference

Class that produces all kinds of GNSS blocks.

```
#include <gnss_block_factory.h>
```

## Public Member Functions

- `std::unique_ptr< GNSSBlockInterface > 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::unique_ptr< 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.*

### 10.159.1 Detailed Description

Class that produces all kinds of GNSS blocks.

Definition at line 48 of file `gnss_block_factory.h`.

### 10.159.2 Member Function Documentation

#### 10.159.2.1 GetBlock()

```
std::unique_ptr<GNSSBlockInterface> GNSSBlockFactory::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.

The documentation for this class was generated from the following file:

- [gnss\\_block\\_factory.h](#)



### 10.160.1 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`.

### 10.160.2 Member Function Documentation

#### 10.160.2.1 `start()`

```
virtual void GNSSBlockInterface::start ( ) [inline], [virtual]
```

Start the flow of samples if needed.

Reimplemented in [Ad9361FpgaSignalSource](#).

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](#)

## 10.161 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 configuration.*
- 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)  
*Priorize visible satellites in the specified vector.*

### 10.161.1 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 62 of file gnss\_flowgraph.h.

### 10.161.2 Constructor & Destructor Documentation

#### 10.161.2.1 GNSSFlowgraph()

```
GNSSFlowgraph::GNSSFlowgraph (
    std::shared_ptr< ConfigurationInterface > configuration,
    std::shared_ptr< Concurrent\_Queue< pmt::pmt_t >> queue )
```

Constructor that initializes the receiver flow graph.

10.161.2.2 `~GNSSFlowgraph()`

```
GNSSFlowgraph::~~GNSSFlowgraph ( )
```

Destructor.

## 10.161.3 Member Function Documentation

10.161.3.1 `acquisition_manager()`

```
void GNSSFlowgraph::acquisition_manager (
    unsigned int who )
```

Manage satellite acquisition.

## Parameters

in	<i>who</i>	Channel ID
----	------------	------------

10.161.3.2 `apply_action()`

```
void GNSSFlowgraph::apply_action (
    unsigned int who,
    unsigned int what )
```

Applies an action to the flow graph.

## Parameters

in	<i>who</i>	Who generated the action
in	<i>what</i>	What is the action. 0: acquisition failed; 1: acquisition success; 2: tracking lost

10.161.3.3 `connect()`

```
void GNSSFlowgraph::connect ( )
```

Connects the defined blocks in the flow graph.

Signal Source > Signal conditioner > Channels >> Observables >> PVT > Output filter

#### 10.161.3.4 disconnect()

```
void GNSSFlowgraph::disconnect ( )
```

Disconnect the blocks in the flow graph.

#### 10.161.3.5 get\_pvt()

```
std::shared_ptr<PvtInterface> GNSSFlowgraph::get_pvt ( ) [inline]
```

Returns a smart pointer to the PVT object.

Definition at line 146 of file gnss\_flowgraph.h.

#### 10.161.3.6 prioritize\_satellites()

```
void GNSSFlowgraph::prioritize_satellites (
    const std::vector< std::pair< int, Gnss_Satellite >> & visible_satellites )
```

Priorize visible satellites in the specified vector.

#### 10.161.3.7 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

#### 10.161.3.8 set\_configuration()

```
void GNSSFlowgraph::set_configuration (
    const std::shared_ptr< ConfigurationInterface > & configuration )
```

Set flow graph configuration.

#### 10.161.3.9 start()

```
void GNSSFlowgraph::start ( )
```

Start the flow graph.

## 10.161.3.10 stop()

```
void GNSSFlowgraph::stop ( )
```

Stop the flow graph.

## 10.161.3.11 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](#)

## 10.162 Gnuplot Class Reference

## Public Member Functions

- [Gnuplot](#) (const std::string &style="points")  
*set a style during construction*
- **Gnuplot** (const std::vector< double > &x, const std::string &title="", const std::string &style="points", const std::string &labelx="x", const std::string &labely="y")
- **Gnuplot** (const std::vector< double > &x, const std::vector< double > &y, const std::string &title="", const std::string &style="points", const std::string &labelx="x", const std::string &labely="y")
- **Gnuplot** (const std::vector< double > &x, const std::vector< double > &y, const std::vector< double > &z, const std::string &title="", const std::string &style="points", const std::string &labelx="x", const std::string &labely="y", const std::string &labelz="z")
- [Gnuplot](#) & **cmd** (const std::string &cmdstr)
- [Gnuplot](#) & **operator<<** (const std::string &cmdstr)  
*Sends a command to an active gnuplot session, identical to cmd()*
- [Gnuplot](#) & **showonscreen** ()
- [Gnuplot](#) & **disablescreen** ()
- [Gnuplot](#) & **savetops** (const std::string &filename="gnuplot\_output")
- [Gnuplot](#) & **savetopdf** (const std::string &filename="gnuplot\_output", unsigned int font\_size=12)
- [Gnuplot](#) & **set\_style** (const std::string &stylestr="points")
- [Gnuplot](#) & **set\_smooth** (const std::string &stylestr="csplines")
- [Gnuplot](#) & **unset\_smooth** ()
- [Gnuplot](#) & **set\_pointsize** (const double pointsize=1.0)
- [Gnuplot](#) & **set\_grid** ()
- [Gnuplot](#) & **unset\_grid** ()
- [Gnuplot](#) & **set\_multiplot** (int rows, int cols)
- [Gnuplot](#) & **unset\_multiplot** ()
- [Gnuplot](#) & **set\_samples** (const int samples=100)
- [Gnuplot](#) & **set\_isosamples** (const int isolines=10)

- [Gnuplot](#) & [set\\_hidden3d](#) ()
- [Gnuplot](#) & [unset\\_hidden3d](#) ()
- [Gnuplot](#) & [set\\_contour](#) (const std::string &position="base")
- [Gnuplot](#) & [unset\\_contour](#) ()
- [Gnuplot](#) & [set\\_surface](#) ()
- [Gnuplot](#) & [unset\\_surface](#) ()
- [Gnuplot](#) & [set\\_legend](#) (const std::string &position="default")
- [Gnuplot](#) & [unset\\_legend](#) ()
- [Gnuplot](#) & [set\\_title](#) (const std::string &title="")
- [Gnuplot](#) & [unset\\_title](#) ()
- *Clears the title of a gnuplot session.*
- [Gnuplot](#) & [set\\_ylabel](#) (const std::string &label="x")
- [Gnuplot](#) & [set\\_xlabel](#) (const std::string &label="y")
- [Gnuplot](#) & [set\\_zlabel](#) (const std::string &label="z")
- [Gnuplot](#) & [set\\_xrange](#) (const double iFrom, const double iTo)
- [Gnuplot](#) & [set\\_yrange](#) (const double iFrom, const double iTo)
- [Gnuplot](#) & [set\\_zrange](#) (const double iFrom, const double iTo)
- [Gnuplot](#) & [set\\_xautoscale](#) ()
- [Gnuplot](#) & [set\\_yautoscale](#) ()
- [Gnuplot](#) & [set\\_zautoscale](#) ()
- [Gnuplot](#) & [set\\_xlogscale](#) (const double base=10)
- [Gnuplot](#) & [set\\_ylogscale](#) (const double base=10)
- [Gnuplot](#) & [set\\_zlogscale](#) (const double base=10)
- [Gnuplot](#) & [unset\\_xlogscale](#) ()
- [Gnuplot](#) & [unset\\_ylogscale](#) ()
- [Gnuplot](#) & [unset\\_zlogscale](#) ()
- [Gnuplot](#) & [set\\_cbrange](#) (const double iFrom, const double iTo)
- [Gnuplot](#) & [plotfile\\_x](#) (const std::string &filename, const unsigned int column=1, const std::string &title="")
- [template<typename X >](#)  
[Gnuplot](#) & [plot\\_x](#) (const X &x, const std::string &title="")
- [Gnuplot](#) & [plotfile\\_xy](#) (const std::string &filename, const unsigned int column\_x=1, const unsigned int column\_y=2, const std::string &title="", const unsigned int decimate=1)
- [template<typename X , typename Y >](#)  
[Gnuplot](#) & [plot\\_xy](#) (const X &x, const Y &y, const std::string &title="", const unsigned int decimate=1)
- [Gnuplot](#) & [plotfile\\_xy\\_err](#) (const std::string &filename, const unsigned int column\_x=1, const unsigned int column\_y=2, const unsigned int column\_dy=3, const std::string &title="")
- [template<typename X , typename Y , typename E >](#)  
[Gnuplot](#) & [plot\\_xy\\_err](#) (const X &x, const Y &y, const E &dy, const std::string &title="")
- [template<typename X , typename Y , typename E >](#)  
[Gnuplot](#) & [plot\\_grid3d](#) (const X &x, const Y &y, const E &mag, const std::string &title="")
- [Gnuplot](#) & [plotfile\\_xyz](#) (const std::string &filename, const unsigned int column\_x=1, const unsigned int column\_y=2, const unsigned int column\_z=3, const std::string &title="")
- [template<typename X , typename Y , typename Z >](#)  
[Gnuplot](#) & [plot\\_xyz](#) (const X &x, const Y &y, const Z &z, const std::string &title="")
- [Gnuplot](#) & [plot\\_slope](#) (const double a, const double b, const std::string &title="")
- [Gnuplot](#) & [plot\\_equation](#) (const std::string &equation, const std::string &title="")
- [Gnuplot](#) & [plot\\_equation3d](#) (const std::string &equation, const std::string &title="")
- [Gnuplot](#) & [plot\\_image](#) (const unsigned char \*ucPicBuf, const unsigned int iWidth, const unsigned int iHeight, const std::string &title="")
- [Gnuplot](#) & [plot\\_circle](#) (double east, double north, double radius, const std::string &label="")
- [Gnuplot](#) & [replot](#) (void)
- *replot repeats the last plot or splot command.*
- [Gnuplot](#) & [reset\\_plot](#) ()
- [Gnuplot](#) & [reset\\_all](#) ()
- void [remove\\_tmpfiles](#) ()
- bool [is\\_valid](#) ()

## Static Public Member Functions

- static bool **set\_GNUPlotPath** (const std::string &path)
- static void **set\_terminal\_std** (const std::string &type)

### 10.162.1 Detailed Description

Definition at line 75 of file `gnuplot_i.h`.

### 10.162.2 Constructor & Destructor Documentation

#### 10.162.2.1 Gnuplot()

```
Gnuplot::Gnuplot (  
    const std::string & style = "points" ) [inline], [explicit]
```

set a style during construction

Definition at line 695 of file `gnuplot_i.h`.

### 10.162.3 Member Function Documentation

#### 10.162.3.1 operator<<()

```
Gnuplot& Gnuplot::operator<< (  
    const std::string & cmdstr ) [inline]
```

Sends a command to an active gnuplot session, identical to `cmd()`

Definition at line 225 of file `gnuplot_i.h`.

#### 10.162.3.2 replot()

```
Gnuplot& Gnuplot::replot (  
    void ) [inline]
```

`replot` repeats the last plot or `splot` command.

Definition at line 639 of file `gnuplot_i.h`.

### 10.162.3.3 unset\_title()

```
Gnuplot& Gnuplot::unset_title ( ) [inline]
```

Clears the title of a gnuplot session.

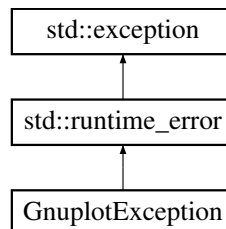
Definition at line 434 of file gnuplot\_i.h.

The documentation for this class was generated from the following file:

- [gnuplot\\_i.h](#)

## 10.163 GnuplotException Class Reference

Inheritance diagram for GnuplotException:



### Public Member Functions

- **GnuplotException** (const std::string &msg)

### 10.163.1 Detailed Description

Definition at line 68 of file gnuplot\_i.h.

The documentation for this class was generated from the following file:

- [gnuplot\\_i.h](#)

## 10.164 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 [i\\_satellite\\_PRN](#) {}  
*SV PRN NUMBER.*
- double [d\\_TOW](#) {}  
*Time Of Week assigned to the acquisition data.*
- double [d\\_Doppler0](#) {}  
*Doppler (0 order term) [Hz].*
- double [d\\_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].*

### 10.164.1 Detailed Description

This class is a storage for the GPS GSM RRLC 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`.

### 10.164.2 Constructor & Destructor Documentation

#### 10.164.2.1 Gps\_Acq\_Assist()

```
Gps_Acq_Assist::Gps_Acq_Assist ( ) [default]
```

Default constructor

### 10.164.3 Member Data Documentation

#### 10.164.3.1 Azimuth

```
double Gps_Acq_Assist::Azimuth {}
```

Satellite Azimuth [deg].

Definition at line 54 of file `gps_acq_assist.h`.

#### 10.164.3.2 Code\_Phase

```
double Gps_Acq_Assist::Code_Phase {}
```

Code phase [chips].

Definition at line 50 of file `gps_acq_assist.h`.

#### 10.164.3.3 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`.

#### 10.164.3.4 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`.

#### 10.164.3.5 d\_Doppler0

```
double Gps_Acq_Assist::d_Doppler0 {}
```

Doppler (0 order term) [Hz].

Definition at line 47 of file `gps_acq_assist.h`.

#### 10.164.3.6 d\_Doppler1

```
double Gps_Acq_Assist::d_Doppler1 {}
```

Doppler (1 order term) [Hz].

Definition at line 48 of file gps\_acq\_assist.h.

#### 10.164.3.7 d\_TOW

```
double Gps_Acq_Assist::d_TOW {}
```

Time Of Week assigned to the acquisition data.

Definition at line 46 of file gps\_acq\_assist.h.

#### 10.164.3.8 dopplerUncertainty

```
double Gps_Acq_Assist::dopplerUncertainty {}
```

Doppler Uncertainty [Hz].

Definition at line 49 of file gps\_acq\_assist.h.

#### 10.164.3.9 Elevation

```
double Gps_Acq_Assist::Elevation {}
```

Satellite Elevation [deg].

Definition at line 55 of file gps\_acq\_assist.h.

#### 10.164.3.10 GPS\_Bit\_Number

```
double Gps_Acq_Assist::GPS_Bit_Number {}
```

GPS Bit Number.

Definition at line 52 of file gps\_acq\_assist.h.

### 10.164.3.11 i\_satellite\_PRN

```
uint32_t Gps_Acq_Assist::i_satellite_PRN {}
```

SV PRN NUMBER.

Definition at line 45 of file `gps_acq_assist.h`.

The documentation for this class was generated from the following file:

- [gps\\_acq\\_assist.h](#)

## 10.165 Gps\_Almanac Class Reference

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

```
#include <gps_almanac.h>
```

### Public Member Functions

- [Gps\\_Almanac](#) ()=default
- `template<class Archive >`  
void **serialize** (Archive &ar, const unsigned int version)

### Public Attributes

- `uint32_t` [i\\_satellite\\_PRN](#) {}  
*SV PRN NUMBER.*
- `double` [d\\_Delta\\_i](#) {}  
*Inclination Angle at Reference Time (relative to  $i_0 = 0.30$  semi-circles)*
- `int32_t` [i\\_Toa](#) {}  
*Almanac data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200K) [s].*
- `int32_t` [i\\_WNa](#) {}  
*Almanac week number.*
- `double` [d\\_M\\_0](#) {}  
*Mean Anomaly at Reference Time [semi-circles].*
- `double` [d\\_e\\_eccentricity](#) {}  
*Eccentricity [dimensionless].*
- `double` [d\\_sqrt\\_A](#) {}  
*Square Root of the Semi-Major Axis [sqrt(m)].*
- `double` [d\\_OMEGA0](#) {}  
*Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles].*
- `double` [d\\_OMEGA](#) {}  
*Argument of Perigee [semi-circles].*
- `double` [d\\_OMEGA\\_DOT](#) {}  
*Rate of Right Ascension [semi-circles/s].*
- `int32_t` [i\\_SV\\_health](#) {}  
*SV Health.*
- `int32_t` [i\\_AS\\_status](#) {}  
*Anti-Spoofing Flags and SV Configuration.*
- `double` [d\\_A\\_f0](#) {}  
*Coefficient 0 of code phase offset model [s].*
- `double` [d\\_A\\_f1](#) {}  
*Coefficient 1 of code phase offset model [s/s].*

### 10.165.1 Detailed Description

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

See <https://www.gps.gov/technical/icwg/IS-GPS-200K.pdf> Appendix II

Definition at line 35 of file gps\_almanac.h.

### 10.165.2 Constructor & Destructor Documentation

#### 10.165.2.1 Gps\_Almanac()

```
Gps_Almanac::Gps_Almanac ( ) [default]
```

Default constructor

### 10.165.3 Member Data Documentation

#### 10.165.3.1 d\_A\_f0

```
double Gps_Almanac::d_A_f0 {}
```

Coefficient 0 of code phase offset model [s].

Definition at line 55 of file gps\_almanac.h.

#### 10.165.3.2 d\_A\_f1

```
double Gps_Almanac::d_A_f1 {}
```

Coefficient 1 of code phase offset model [s/s].

Definition at line 56 of file gps\_almanac.h.

#### 10.165.3.3 d\_Delta\_i

```
double Gps_Almanac::d_Delta_i {}
```

Inclination Angle at Reference Time (relative to  $i_0 = 0.30$  semi-circles)

Definition at line 44 of file gps\_almanac.h.

#### 10.165.3.4 d\_e\_eccentricity

```
double Gps_Almanac::d_e_eccentricity {}
```

Eccentricity [dimensionless].

Definition at line 48 of file gps\_almanac.h.

#### 10.165.3.5 d\_M\_0

```
double Gps_Almanac::d_M_0 {}
```

Mean Anomaly at Reference Time [semi-circles].

Definition at line 47 of file gps\_almanac.h.

#### 10.165.3.6 d\_OMEGA

```
double Gps_Almanac::d_OMEGA {}
```

Argument of Perigee [semi-circles].

Definition at line 51 of file gps\_almanac.h.

#### 10.165.3.7 d\_OMEGA0

```
double Gps_Almanac::d_OMEGA0 {}
```

Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles].

Definition at line 50 of file gps\_almanac.h.

#### 10.165.3.8 d\_OMEGA\_DOT

```
double Gps_Almanac::d_OMEGA_DOT {}
```

Rate of Right Ascension [semi-circles/s].

Definition at line 52 of file gps\_almanac.h.

#### 10.165.3.9 d\_sqrt\_A

```
double Gps_Almanac::d_sqrt_A {}
```

Square Root of the Semi-Major Axis [sqrt(m)].

Definition at line 49 of file gps\_almanac.h.

#### 10.165.3.10 i\_AS\_status

```
int32_t Gps_Almanac::i_AS_status {}
```

Anti-Spoofing Flags and SV Configuration.

Definition at line 54 of file gps\_almanac.h.

#### 10.165.3.11 i\_satellite\_PRN

```
uint32_t Gps_Almanac::i_satellite_PRN {}
```

SV PRN NUMBER.

Definition at line 43 of file gps\_almanac.h.

#### 10.165.3.12 i\_SV\_health

```
int32_t Gps_Almanac::i_SV_health {}
```

SV Health.

Definition at line 53 of file gps\_almanac.h.

#### 10.165.3.13 i\_Toa

```
int32_t Gps_Almanac::i_Toa {}
```

Almanac data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200K) [s].

Definition at line 45 of file gps\_almanac.h.

### 10.165.3.14 i\_WNa

```
int32_t Gps_Almanac::i_WNa {}
```

Almanac week number.

Definition at line 46 of file `gps_almanac.h`.

The documentation for this class was generated from the following file:

- [gps\\_almanac.h](#)

## 10.166 Gps\_CNAV\_Ephemeris Class Reference

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

```
#include <gps_cnav_ephemeris.h>
```

### Public Member Functions

- [Gps\\_CNAV\\_Ephemeris](#) ()=default
- double [satellitePosition](#) (double transmitTime)  
*Compute the ECEF SV coordinates and ECEF velocity Implementation of Table 20-IV (IS-GPS-200K)*
- double [sv\\_clock\\_drift](#) (double transmitTime)  
*Sets (d\_satClkDrift) and returns the clock drift in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200K, 20.3.3.3.3.1)*
- double [sv\\_clock\\_relativistic\\_term](#) (double transmitTime)  
*Sets (d\_dtr) and returns the clock relativistic correction term in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200K, 20.3.3.3.3.1)*
- 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

- uint32\_t [i\\_satellite\\_PRN](#) {}
- int32\_t [i\\_GPS\\_week](#) {}  
*GPS week number, aka WN [week].*
- int32\_t [i\\_URA](#) {}  
*ED Accuracy Index.*
- int32\_t [i\\_signal\\_health](#) {}  
*Signal health (L1/L2/L5)*
- int32\_t [d\\_Top](#) {}  
*Data predict time of week.*
- double [d\\_DELTA\\_A](#) {}  
*Semi-major axis difference at reference time.*
- double [d\\_A\\_DOT](#) {}  
*Change rate in semi-major axis.*

- double [d\\_Delta\\_n](#) {}  
*Mean Motion Difference From Computed Value [semi-circles/s].*
- double [d\\_DELTA\\_DOT\\_N](#) {}  
*Rate of mean motion difference from computed value.*
- double [d\\_M\\_0](#) {}  
*Mean Anomaly at Reference Time [semi-circles].*
- double [d\\_e\\_eccentricity](#) {}  
*Eccentricity.*
- double [d\\_OMEGA](#) {}  
*Argument of Perigee [semi-circles].*
- double [d\\_OMEGA0](#) {}  
*Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles].*
- int32\_t [d\\_Toe1](#) {}  
*Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200K) [s].*
- int32\_t [d\\_Toe2](#) {}  
*Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200K) [s].*
- double [d\\_DELTA\\_OMEGA\\_DOT](#) {}  
*Rate of Right Ascension difference [semi-circles/s].*
- double [d\\_i\\_0](#) {}  
*Inclination Angle at Reference Time [semi-circles].*
- double [d\\_IDOT](#) {}  
*Rate of Inclination Angle [semi-circles/s].*
- double [d\\_Cis](#) {}  
*Amplitude of the Sine Harmonic Correction Term to the Angle of Inclination [rad].*
- double [d\\_Cic](#) {}  
*Amplitude of the Cosine Harmonic Correction Term to the Angle of Inclination [rad].*
- double [d\\_Crs](#) {}  
*Amplitude of the Sine Harmonic Correction Term to the Orbit Radius [m].*
- double [d\\_Crc](#) {}  
*Amplitude of the Cosine Harmonic Correction Term to the Orbit Radius [m].*
- double [d\\_Cus](#) {}  
*Amplitude of the Sine Harmonic Correction Term to the Argument of Latitude [rad].*
- double [d\\_Cuc](#) {}  
*Amplitude of the Cosine Harmonic Correction Term to the Argument of Latitude [rad].*
- int32\_t [d\\_Toc](#) {}  
*clock data reference time (Ref. 20.3.3.3.3.1 IS-GPS-200K) [s]*
- double [d\\_A\\_f0](#) {}  
*Coefficient 0 of code phase offset model [s].*
- double [d\\_A\\_f1](#) {}  
*Coefficient 1 of code phase offset model [s/s].*
- double [d\\_A\\_f2](#) {}  
*Coefficient 2 of code phase offset model [s/s<sup>2</sup>].*
- double [d\\_URA0](#) {}  
*NED Accuracy Index.*
- double [d\\_URA1](#) {}  
*NED Accuracy Change Index.*
- double [d\\_URA2](#) {}  
*NED Accuracy Change Rate Index.*
- 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].*
- double [d\\_ISCL1](#) {}

- double **d\_ISCL2** {}
- double **d\_ISCL5I** {}
- double **d\_ISCL5Q** {}
- int32\_t **d\_TOW** {}  
*Time of GPS Week of the ephemeris set (taken from subframes TOW) [s].*
- double **d\_satClkDrift** {}  
*GPS clock error.*
- double **d\_dtr** {}  
*relativistic clock correction term*
- double **d\_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 **d\_satpos\_Y** {}  
*Earth-fixed coordinate y of the satellite [m]. Completes a right-handed, Earth-Centered, Earth-Fixed orthogonal coordinate system.*
- double **d\_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 **d\_satvel\_X** {}  
*Earth-fixed velocity coordinate x of the satellite [m].*
- double **d\_satvel\_Y** {}  
*Earth-fixed velocity coordinate y of the satellite [m].*
- double **d\_satvel\_Z** {}  
*Earth-fixed velocity coordinate z of the satellite [m].*
- bool **b\_integrity\_status\_flag** {}  
*If true, enhanced level of integrity assurance.*
- bool **b\_l2c\_phasing\_flag** {}
- 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.*
- bool **b\_antispoofing\_flag** {}  
*If true, the AntiSpoofing mode is ON in that SV.*

### 10.166.1 Detailed Description

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

See <https://www.gps.gov/technical/icwg/IS-GPS-200K.pdf> Appendix III

Definition at line 35 of file gps\_cnav\_ephemeris.h.

### 10.166.2 Constructor & Destructor Documentation

#### 10.166.2.1 Gps\_CNAV\_Ephemeris()

```
Gps_CNAV_Ephemeris::Gps_CNAV_Ephemeris ( ) [default]
```

Default constructor

### 10.166.3 Member Function Documentation

#### 10.166.3.1 satellitePosition()

```
double Gps_CNAV_Ephemeris::satellitePosition (
    double transmitTime )
```

Compute the ECEF SV coordinates and ECEF velocity Implementation of Table 20-IV (IS-GPS-200K)

#### 10.166.3.2 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.

< Time of GPS Week of the ephemeris set (taken from subframes TOW) [s]

< Amplitude of the Sine Harmonic Correction Term to the Orbit Radius [m]

< Mean Anomaly at Reference Time [semi-circles]

< Amplitude of the Cosine Harmonic Correction Term to the Argument of Latitude [rad]

< Eccentricity [dimensionless]

< Amplitude of the Sine Harmonic Correction Term to the Argument of Latitude [rad]

< Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200K) [s]

< Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200K) [s]

< clock data reference time (Ref. 20.3.3.3.1 IS-GPS-200K) [s]

< Amplitude of the Cosine Harmonic Correction Term to the Angle of Inclination [rad]

< Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles]

< Amplitude of the Sine Harmonic Correction Term to the Angle of Inclination [rad]

< Inclination Angle at Reference Time [semi-circles]

< Amplitude of the Cosine Harmonic Correction Term to the Orbit Radius [m]

< Argument of Perigee [semi-circles]

< Rate of Inclination Angle [semi-circles/s]

- < GPS week number, aka WN [week]
- < Estimated Group Delay Differential: L1-L2 correction term only for the benefit of "L1 P(Y)" or "L2 P(Y)" s users [s]
- < Estimated Group Delay Differential: L1P(Y)-L1C/A correction term only for the benefit of "L1 P(Y)" or "L2 P(Y)" s users [s]
- < Estimated Group Delay Differential: L1P(Y)-L2C correction term only for the benefit of "L1 P(Y)" or "L2 P(Y)" s users [s]
- < Estimated Group Delay Differential: L1P(Y)-L5i correction term only for the benefit of "L1 P(Y)" or "L2 P(Y)" s users [s]
- < Estimated Group Delay Differential: L1P(Y)-L5q correction term only for the benefit of "L1 P(Y)" or "L2 P(Y)" s users [s]
- < Semi-major axis difference at reference time [m]
- < Change rate in semi-major axis [m/s]
- < Rate of Right Ascension difference [semi-circles/s]
- < Coefficient 0 of code phase offset model [s]
- < Coefficient 1 of code phase offset model [s/s]
- < Coefficient 2 of code phase offset model [s/s^2]
- < If true, indicates that the SV URA may be worse than indicated in d\_SV\_accuracy, use that SV at our own risk.
- < If true, the AntiSpoofing mode is ON in that SV

Definition at line 141 of file gps\_cnav\_ephemeris.h.

References b\_alert\_flag, b\_antispoofing\_flag, b\_integrity\_status\_flag, d\_A\_DOT, d\_A\_f0, d\_A\_f1, d\_A\_f2, d\_Cic, d\_Cis, d\_Crc, d\_Crs, d\_Cuc, d\_Cus, d\_DELTA\_A, d\_DELTA\_OMEGA\_DOT, d\_e\_eccentricity, d\_i\_0, d\_IDOT, d\_M\_0, d\_OMEGA, d\_OMEGA0, d\_TGD, d\_Toc, d\_Toe1, d\_Toe2, d\_TOW, and i\_GPS\_week.

### 10.166.3.3 sv\_clock\_drift()

```
double Gps_CNAV_Ephemeris::sv_clock_drift (
    double transmitTime )
```

Sets (*d\_satClkDrift*) and returns the clock drift in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200K, 20.3.3.3.3.1)

### 10.166.3.4 sv\_clock\_relativistic\_term()

```
double Gps_CNAV_Ephemeris::sv_clock_relativistic_term (
    double transmitTime )
```

Sets (*d\_dtr*) and returns the clock relativistic correction term in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200K, 20.3.3.3.3.1)

## 10.166.4 Member Data Documentation

### 10.166.4.1 b\_alert\_flag

```
bool Gps_CNAV_Ephemeris::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.

Definition at line 133 of file gps\_cnav\_ephemeris.h.

Referenced by serialize().

### 10.166.4.2 b\_antispoofing\_flag

```
bool Gps_CNAV_Ephemeris::b_antispoofing_flag {}
```

If true, the AntiSpoofing mode is ON in that SV.

Definition at line 134 of file gps\_cnav\_ephemeris.h.

Referenced by serialize().

### 10.166.4.3 b\_integrity\_status\_flag

```
bool Gps_CNAV_Ephemeris::b_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 131 of file gps\_cnav\_ephemeris.h.

Referenced by serialize().

#### 10.166.4.4 d\_A\_DOT

```
double Gps_CNAV_Ephemeris::d_A_DOT {}
```

Change rate in semi-major axis.

Definition at line 69 of file gps\_cnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.166.4.5 d\_A\_f0

```
double Gps_CNAV_Ephemeris::d_A_f0 {}
```

Coefficient 0 of code phase offset model [s].

Definition at line 90 of file gps\_cnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.166.4.6 d\_A\_f1

```
double Gps_CNAV_Ephemeris::d_A_f1 {}
```

Coefficient 1 of code phase offset model [s/s].

Definition at line 91 of file gps\_cnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.166.4.7 d\_A\_f2

```
double Gps_CNAV_Ephemeris::d_A_f2 {}
```

Coefficient 2 of code phase offset model [s/s<sup>2</sup>].

Definition at line 92 of file gps\_cnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.166.4.8 d\_Cic

```
double Gps_CNAV_Ephemeris::d_Cic {}
```

Amplitude of the Cosine Harmonic Correction Term to the Angle of Inclination [rad].

Definition at line 82 of file gps\_cnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.166.4.9 d\_Cis

```
double Gps_CNAV_Ephemeris::d_Cis {}
```

Amplitude of the Sine Harmonic Correction Term to the Angle of Inclination [rad].

Definition at line 81 of file gps\_cnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.166.4.10 d\_Crc

```
double Gps_CNAV_Ephemeris::d_Crc {}
```

Amplitude of the Cosine Harmonic Correction Term to the Orbit Radius [m].

Definition at line 84 of file gps\_cnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.166.4.11 d\_Crs

```
double Gps_CNAV_Ephemeris::d_Crs {}
```

Amplitude of the Sine Harmonic Correction Term to the Orbit Radius [m].

Definition at line 83 of file gps\_cnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.166.4.12 d\_Cuc

```
double Gps_CNAV_Ephemeris::d_Cuc {}
```

Amplitude of the Cosine Harmonic Correction Term to the Argument of Latitude [rad].

Definition at line 86 of file `gps_cnav_ephemeris.h`.

Referenced by `serialize()`.

#### 10.166.4.13 d\_Cus

```
double Gps_CNAV_Ephemeris::d_Cus {}
```

Amplitude of the Sine Harmonic Correction Term to the Argument of Latitude [rad].

Definition at line 85 of file `gps_cnav_ephemeris.h`.

Referenced by `serialize()`.

#### 10.166.4.14 d\_DELTA\_A

```
double Gps_CNAV_Ephemeris::d_DELTA_A {}
```

Semi-major axis difference at reference time.

Definition at line 68 of file `gps_cnav_ephemeris.h`.

Referenced by `serialize()`.

#### 10.166.4.15 d\_DELTA\_DOT\_N

```
double Gps_CNAV_Ephemeris::d_DELTA_DOT_N {}
```

Rate of mean motion difference from computed value.

Definition at line 71 of file `gps_cnav_ephemeris.h`.

#### 10.166.4.16 d\_Delta\_n

```
double Gps_CNAV_Ephemeris::d_Delta_n {}
```

Mean Motion Difference From Computed Value [semi-circles/s].

Definition at line 70 of file `gps_cnav_ephemeris.h`.

**10.166.4.17 d\_DELTA\_OMEGA\_DOT**

```
double Gps_CNAV_Ephemeris::d_DELTA_OMEGA_DOT {}
```

Rate of Right Ascension difference [semi-circles/s].

Definition at line 78 of file gps\_cnav\_ephemeris.h.

Referenced by `serialize()`.

**10.166.4.18 d\_dtr**

```
double Gps_CNAV_Ephemeris::d_dtr {}
```

relativistic clock correction term

Definition at line 109 of file gps\_cnav\_ephemeris.h.

**10.166.4.19 d\_e\_eccentricity**

```
double Gps_CNAV_Ephemeris::d_e_eccentricity {}
```

Eccentricity.

Definition at line 73 of file gps\_cnav\_ephemeris.h.

Referenced by `serialize()`.

**10.166.4.20 d\_i\_0**

```
double Gps_CNAV_Ephemeris::d_i_0 {}
```

Inclination Angle at Reference Time [semi-circles].

Definition at line 79 of file gps\_cnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.166.4.21 d\_IDOT

```
double Gps_CNAV_Ephemeris::d_IDOT {}
```

Rate of Inclination Angle [semi-circles/s].

Definition at line 80 of file gps\_cnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.166.4.22 d\_M\_0

```
double Gps_CNAV_Ephemeris::d_M_0 {}
```

Mean Anomaly at Reference Time [semi-circles].

Definition at line 72 of file gps\_cnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.166.4.23 d\_OMEGA

```
double Gps_CNAV_Ephemeris::d_OMEGA {}
```

Argument of Perigee [semi-cicles].

Definition at line 74 of file gps\_cnav\_ephemeris.h.

Referenced by `serialize()`.

#### 10.166.4.24 d\_OMEGA0

```
double Gps_CNAV_Ephemeris::d_OMEGA0 {}
```

Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-cicles].

Definition at line 75 of file gps\_cnav\_ephemeris.h.

Referenced by `serialize()`.

**10.166.4.25 d\_satClkDrift**

```
double Gps_CNAV_Ephemeris::d_satClkDrift {}
```

GPS clock error.

Definition at line 108 of file gps\_cnav\_ephemeris.h.

**10.166.4.26 d\_satpos\_X**

```
double Gps_CNAV_Ephemeris::d_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 112 of file gps\_cnav\_ephemeris.h.

**10.166.4.27 d\_satpos\_Y**

```
double Gps_CNAV_Ephemeris::d_satpos_Y {}
```

Earth-fixed coordinate y of the satellite [m]. Completes a right-handed, Earth-Centered, Earth-Fixed orthogonal coordinate system.

Definition at line 113 of file gps\_cnav\_ephemeris.h.

**10.166.4.28 d\_satpos\_Z**

```
double Gps_CNAV_Ephemeris::d_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 114 of file gps\_cnav\_ephemeris.h.

**10.166.4.29 d\_satvel\_X**

```
double Gps_CNAV_Ephemeris::d_satvel_X {}
```

Earth-fixed velocity coordinate x of the satellite [m].

Definition at line 117 of file gps\_cnav\_ephemeris.h.

**10.166.4.30 d\_satvel\_Y**

```
double Gps_CNAV_Ephemeris::d_satvel_Y {}
```

Earth-fixed velocity coordinate y of the satellite [m].

Definition at line 118 of file `gps_cnav_ephemeris.h`.

**10.166.4.31 d\_satvel\_Z**

```
double Gps_CNAV_Ephemeris::d_satvel_Z {}
```

Earth-fixed velocity coordinate z of the satellite [m].

Definition at line 119 of file `gps_cnav_ephemeris.h`.

**10.166.4.32 d\_TGD**

```
double Gps_CNAV_Ephemeris::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].

Definition at line 99 of file `gps_cnav_ephemeris.h`.

Referenced by `serialize()`.

**10.166.4.33 d\_Toc**

```
int32_t Gps_CNAV_Ephemeris::d_Toc {}
```

clock data reference time (Ref. 20.3.3.3.1 IS-GPS-200K) [s]

Definition at line 89 of file `gps_cnav_ephemeris.h`.

Referenced by `serialize()`.

**10.166.4.34 d\_Toe1**

```
int32_t Gps_CNAV_Ephemeris::d_Toe1 {}
```

Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200K) [s].

Definition at line 76 of file `gps_cnav_ephemeris.h`.

Referenced by `serialize()`.

**10.166.4.35 d\_Toe2**

```
int32_t Gps_CNAV_Ephemeris::d_Toe2 {}
```

Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200K) [s].

Definition at line 77 of file `gps_cnav_ephemeris.h`.

Referenced by `serialize()`.

**10.166.4.36 d\_Top**

```
int32_t Gps_CNAV_Ephemeris::d_Top {}
```

Data predict time of week.

Definition at line 67 of file `gps_cnav_ephemeris.h`.

**10.166.4.37 d\_TOW**

```
int32_t Gps_CNAV_Ephemeris::d_TOW {}
```

Time of GPS Week of the ephemeris set (taken from subframes TOW) [s].

Definition at line 105 of file `gps_cnav_ephemeris.h`.

Referenced by `serialize()`.

**10.166.4.38 d\_URA0**

```
double Gps_CNAV_Ephemeris::d_URA0 {}
```

NED Accuracy Index.

Definition at line 94 of file `gps_cnav_ephemeris.h`.

**10.166.4.39 d\_URA1**

```
double Gps_CNAV_Ephemeris::d_URA1 {}
```

NED Accuracy Change Index.

Definition at line 95 of file `gps_cnav_ephemeris.h`.

#### 10.166.4.40 d\_URA2

```
double Gps_CNAV_Ephemeris::d_URA2 {}
```

NED Accuracy Change Rate Index.

Definition at line 96 of file `gps_cnav_ephemeris.h`.

#### 10.166.4.41 i\_GPS\_week

```
int32_t Gps_CNAV_Ephemeris::i_GPS_week {}
```

GPS week number, aka WN [week].

Definition at line 64 of file `gps_cnav_ephemeris.h`.

Referenced by `serialize()`.

#### 10.166.4.42 i\_signal\_health

```
int32_t Gps_CNAV_Ephemeris::i_signal_health {}
```

Signal health (L1/L2/L5)

Definition at line 66 of file `gps_cnav_ephemeris.h`.

#### 10.166.4.43 i\_URA

```
int32_t Gps_CNAV_Ephemeris::i_URA {}
```

ED Accuracy Index.

Definition at line 65 of file `gps_cnav_ephemeris.h`.

The documentation for this class was generated from the following file:

- [gps\\_cnav\\_ephemeris.h](#)

## 10.167 Gps\_CNAV\_Iono Class Reference

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

```
#include <gps_cnav_iono.h>
```

## Public Member Functions

- [Gps\\_CNAV\\_Iono](#) ()=default  
*Default constructor.*
- `template<class Archive >`  
`void serialize (Archive &archive, const unsigned int version) const`  
*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\_alpha0 {}`  
*Coefficient 0 of a cubic equation representing the amplitude of the vertical delay [s].*
- `double d\_alpha1 {}`  
*Coefficient 1 of a cubic equation representing the amplitude of the vertical delay [s/semi-circle].*
- `double d\_alpha2 {}`  
*Coefficient 2 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)^2].*
- `double d\_alpha3 {}`  
*Coefficient 3 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)^3].*
- `double d\_beta0 {}`  
*Coefficient 0 of a cubic equation representing the period of the model [s].*
- `double d\_beta1 {}`  
*Coefficient 1 of a cubic equation representing the period of the model [s/semi-circle].*
- `double d\_beta2 {}`  
*Coefficient 2 of a cubic equation representing the period of the model [s(semi-circle)^2].*
- `double d\_beta3 {}`  
*Coefficient 3 of a cubic equation representing the period of the model [s(semi-circle)^3].*
- `bool valid {}`  
*Valid flag.*

### 10.167.1 Detailed Description

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

See <https://www.gps.gov/technical/icwg/IS-GPS-200K.pdf> Appendix III

Definition at line 35 of file `gps_cnav_iono.h`.

### 10.167.2 Constructor & Destructor Documentation

#### 10.167.2.1 Gps\_CNAV\_Iono()

```
Gps_CNAV_Iono::Gps_CNAV_Iono ( ) [default]
```

Default constructor.

### 10.167.3 Member Function Documentation

#### 10.167.3.1 `serialize()`

```
template<class Archive >
void Gps_CNAV_Iono::serialize (
    Archive & archive,
    const unsigned int version ) const [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 56 of file `gps_cnav_iono.h`.

References `d_alpha0`, `d_alpha1`, `d_alpha2`, `d_alpha3`, `d_beta0`, `d_beta1`, `d_beta2`, and `d_beta3`.

### 10.167.4 Member Data Documentation

#### 10.167.4.1 `d_alpha0`

```
double Gps_CNAV_Iono::d_alpha0 {}
```

Coefficient 0 of a cubic equation representing the amplitude of the vertical delay [s].

Definition at line 41 of file `gps_cnav_iono.h`.

Referenced by `serialize()`.

#### 10.167.4.2 `d_alpha1`

```
double Gps_CNAV_Iono::d_alpha1 {}
```

Coefficient 1 of a cubic equation representing the amplitude of the vertical delay [s/semi-circle].

Definition at line 42 of file `gps_cnav_iono.h`.

Referenced by `serialize()`.

#### 10.167.4.3 d\_alpha2

```
double Gps_CNAV_Iono::d_alpha2 {}
```

Coefficient 2 of a cubic equation representing the amplitude of the vertical delay  $[s(\text{semi-circle})^2]$ .

Definition at line 43 of file `gps_cnav_iono.h`.

Referenced by `serialize()`.

#### 10.167.4.4 d\_alpha3

```
double Gps_CNAV_Iono::d_alpha3 {}
```

Coefficient 3 of a cubic equation representing the amplitude of the vertical delay  $[s(\text{semi-circle})^3]$ .

Definition at line 44 of file `gps_cnav_iono.h`.

Referenced by `serialize()`.

#### 10.167.4.5 d\_beta0

```
double Gps_CNAV_Iono::d_beta0 {}
```

Coefficient 0 of a cubic equation representing the period of the model [s].

Definition at line 45 of file `gps_cnav_iono.h`.

Referenced by `serialize()`.

#### 10.167.4.6 d\_beta1

```
double Gps_CNAV_Iono::d_beta1 {}
```

Coefficient 1 of a cubic equation representing the period of the model  $[s/\text{semi-circle}]$ .

Definition at line 46 of file `gps_cnav_iono.h`.

Referenced by `serialize()`.

#### 10.167.4.7 d\_beta2

```
double Gps_CNAV_Iono::d_beta2 {}
```

Coefficient 2 of a cubic equation representing the period of the model [s(semi-circle)<sup>2</sup>].

Definition at line 47 of file `gps_cnav_iono.h`.

Referenced by `serialize()`.

#### 10.167.4.8 d\_beta3

```
double Gps_CNAV_Iono::d_beta3 {}
```

Coefficient 3 of a cubic equation representing the period of the model [s(semi-circle)<sup>3</sup>].

Definition at line 48 of file `gps_cnav_iono.h`.

Referenced by `serialize()`.

#### 10.167.4.9 valid

```
bool Gps_CNAV_Iono::valid {}
```

Valid flag.

Definition at line 49 of file `gps_cnav_iono.h`.

The documentation for this class was generated from the following file:

- [gps\\_cnav\\_iono.h](#)

## 10.168 Gps\_CNAV\_Navigation\_Message Class Reference

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

```
#include <gps_cnav_navigation_message.h>
```

## Public Member Functions

- [Gps\\_CNAV\\_Navigation\\_Message](#) ()
- void **decode\_page** (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\\_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.*

### 10.168.1 Detailed Description

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

See <https://www.gps.gov/technical/icwg/IS-GPS-200K.pdf> Appendix III

Definition at line 44 of file `gps_cnav_navigation_message.h`.

### 10.168.2 Constructor & Destructor Documentation

#### 10.168.2.1 Gps\_CNAV\_Navigation\_Message()

```
Gps_CNAV_Navigation_Message::Gps_CNAV_Navigation_Message ( )
```

Default constructor

### 10.168.3 Member Function Documentation

#### 10.168.3.1 get\_ephemeris()

```
Gps\_CNAV\_Ephemeris Gps_CNAV_Navigation_Message::get_ephemeris ( ) const
```

Obtain a GPS SV Ephemeris class filled with current SV data.

**10.168.3.2 get\_iono()**

```
Gps_CNAV_Iono Gps_CNAV_Navigation_Message::get_iono ( ) const
```

Obtain a GPS ionospheric correction parameters class filled with current SV data.

**10.168.3.3 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.

**10.168.3.4 have\_new\_ephemeris()**

```
bool Gps_CNAV_Navigation_Message::have_new_ephemeris ( )
```

Check if we have a new ephemeris stored in the GPS ephemeris class.

**10.168.3.5 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.

**10.168.3.6 have\_new\_utc\_model()**

```
bool Gps_CNAV_Navigation_Message::have_new_utc_model ( )
```

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](#)

**10.169 Gps\_CNAV\_Utc\_Model Class Reference**

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

```
#include <gps_cnav_utc_model.h>
```

## Public Member Functions

- [Gps\\_CNAV\\_Utc\\_Model](#) ()=default
- template<class Archive >  
void **serialize** (Archive &archive, const uint32\_t version)

## Public Attributes

- double [d\\_A2](#) {}  
*2nd order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200K) [s/s]*
- double [d\\_A1](#) {}  
*1st order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200K) [s/s]*
- double [d\\_A0](#) {}  
*Constant of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200K) [s].*
- int32\_t [d\\_t\\_OT](#) {}  
*Reference time for UTC data (reference 20.3.4.5 and 20.3.3.5.2.4 IS-GPS-200K) [s].*
- int32\_t [i\\_WN\\_T](#) {}  
*UTC reference week number [weeks].*
- 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.*
- int32\_t [i\\_WN\\_LSF](#) {}  
*Week number at the end of which the leap second becomes effective [weeks].*
- int32\_t [i\\_DN](#) {}  
*Day number (DN) at the end of which the leap second becomes effective [days].*
- 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].*
- bool **valid** {}

### 10.169.1 Detailed Description

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

See <https://www.gps.gov/technical/icwg/IS-GPS-200K.pdf> Appendix III

Definition at line 35 of file `gps_cnav_utc_model.h`.

### 10.169.2 Constructor & Destructor Documentation

#### 10.169.2.1 Gps\_CNAV\_Utc\_Model()

```
Gps_CNAV_Utc_Model::Gps_CNAV_Utc_Model ( ) [default]
```

Default constructor

### 10.169.3 Member Data Documentation

#### 10.169.3.1 d\_A0

```
double Gps_CNAV_Utc_Model::d_A0 {}
```

Constant of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200K) [s].

Definition at line 46 of file `gps_cnav_utc_model.h`.

#### 10.169.3.2 d\_A1

```
double Gps_CNAV_Utc_Model::d_A1 {}
```

1st order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200K) [s/s]

Definition at line 45 of file `gps_cnav_utc_model.h`.

#### 10.169.3.3 d\_A2

```
double Gps_CNAV_Utc_Model::d_A2 {}
```

2nd order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200K) [s/s]

Definition at line 44 of file `gps_cnav_utc_model.h`.

#### 10.169.3.4 d\_DeltaT\_LS

```
int32_t Gps_CNAV_Utc_Model::d_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_cnav_utc_model.h`.

#### 10.169.3.5 d\_DeltaT\_LSF

```
int32_t Gps_CNAV_Utc_Model::d_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_cnav_utc_model.h`.

### 10.169.3.6 d\_t\_OT

```
int32_t Gps_CNAV_Utc_Model::d_t_OT {}
```

Reference time for UTC data (reference 20.3.4.5 and 20.3.3.5.2.4 IS-GPS-200K) [s].

Definition at line 47 of file `gps_cnav_utc_model.h`.

### 10.169.3.7 i\_DN

```
int32_t Gps_CNAV_Utc_Model::i_DN {}
```

Day number (DN) at the end of which the leap second becomes effective [days].

Definition at line 51 of file `gps_cnav_utc_model.h`.

### 10.169.3.8 i\_WN\_LSF

```
int32_t Gps_CNAV_Utc_Model::i_WN_LSF {}
```

Week number at the end of which the leap second becomes effective [weeks].

Definition at line 50 of file `gps_cnav_utc_model.h`.

### 10.169.3.9 i\_WN\_T

```
int32_t Gps_CNAV_Utc_Model::i_WN_T {}
```

UTC reference week number [weeks].

Definition at line 48 of file `gps_cnav_utc_model.h`.

The documentation for this class was generated from the following file:

- [gps\\_cnav\\_utc\\_model.h](#)

## 10.170 Gps\_Ephemeris Class Reference

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

```
#include <gps_ephemeris.h>
```

## Public Member Functions

- [Gps\\_Ephemeris](#) ()
- double [satellitePosition](#) (double transmitTime)  
*Compute the ECEF SV coordinates and ECEF velocity Implementation of Table 20-IV (IS-GPS-200K) and compute the clock bias term including relativistic effect (return value)*
- double [sv\\_clock\\_drift](#) (double transmitTime)  
*Sets (d\_satClkDrift)and returns the clock drift in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200K, 20.3.3.3.3.1)*
- double [sv\\_clock\\_relativistic\\_term](#) (double transmitTime)  
*Sets (d\_dtr) and returns the clock relativistic correction term in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200K, 20.3.3.3.3.1)*
- 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

- uint32\_t [i\\_satellite\\_PRN](#) {}
- int32\_t [d\\_TOW](#) {}  
*Time of GPS Week of the ephemeris set (taken from subframes TOW) [s].*
- double [d\\_Crs](#) {}  
*Amplitude of the Sine Harmonic Correction Term to the Orbit Radius [m].*
- double [d\\_Delta\\_n](#) {}  
*Mean Motion Difference From Computed Value [semi-circles/s].*
- double [d\\_M\\_0](#) {}  
*Mean Anomaly at Reference Time [semi-circles].*
- double [d\\_Cuc](#) {}  
*Amplitude of the Cosine Harmonic Correction Term to the Argument of Latitude [rad].*
- double [d\\_e\\_eccentricity](#) {}  
*Eccentricity [dimensionless].*
- double [d\\_Cus](#) {}  
*Amplitude of the Sine Harmonic Correction Term to the Argument of Latitude [rad].*
- double [d\\_sqrt\\_A](#) {}  
*Square Root of the Semi-Major Axis [sqrt(m)].*
- int32\_t [d\\_Toe](#) {}  
*Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200K) [s].*
- int32\_t [d\\_Toc](#) {}  
*clock data reference time (Ref. 20.3.3.3.3.1 IS-GPS-200K) [s]*
- double [d\\_Cic](#) {}  
*Amplitude of the Cosine Harmonic Correction Term to the Angle of Inclination [rad].*
- double [d\\_OMEGA0](#) {}  
*Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles].*
- double [d\\_Cis](#) {}  
*Amplitude of the Sine Harmonic Correction Term to the Angle of Inclination [rad].*
- double [d\\_i\\_0](#) {}  
*Inclination Angle at Reference Time [semi-circles].*
- double [d\\_Crc](#) {}  
*Amplitude of the Cosine Harmonic Correction Term to the Orbit Radius [m].*
- double [d\\_OMEGA](#) {}

- Argument of Perigee [semi-cycles].*
- double [d\\_OMEGA\\_DOT](#) {}
  - Rate of Right Ascension [semi-circles/s].*
- double [d\\_IDOT](#) {}
  - Rate of Inclination Angle [semi-circles/s].*
- int32\_t [i\\_code\\_on\\_L2](#) {}
  - If 1, P code ON in L2; if 2, C/A code ON in L2;.*
- int32\_t [i\\_GPS\\_week](#) {}
  - GPS week number, aka WN [week].*
- 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.*
- 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-200K)*
- int32\_t [i\\_SV\\_health](#) {}
- 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].*
- int32\_t [d\\_IODC](#) {}
  - Issue of Data, Clock.*
- int32\_t [d\\_IODE\\_SF2](#) {}
  - Issue of Data, Ephemeris (IODE), subframe 2.*
- int32\_t [d\\_IODE\\_SF3](#) {}
  - Issue of Data, Ephemeris(IODE), subframe 3.*
- 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].*
- 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.*
- double [d\\_spare1](#) {}
- double [d\\_spare2](#) {}
- double [d\\_A\\_f0](#) {}
  - Coefficient 0 of code phase offset model [s].*
- double [d\\_A\\_f1](#) {}
  - Coefficient 1 of code phase offset model [s/s].*
- double [d\\_A\\_f2](#) {}
  - Coefficient 2 of code phase offset model [s/s^2].*
- bool [b\\_integrity\\_status\\_flag](#) {}
  - If true, enhanced level of integrity assurance.*
- 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.*
- bool [b\\_antispoofing\\_flag](#) {}
  - If true, the AntiSpoofing mode is ON in that SV.*
- double [d\\_satClkDrift](#) {}
  - GPS clock error.*
- double [d\\_dtr](#) {}
  - relativistic clock correction term*
- double [d\\_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 [d\\_satpos\\_Y](#) {}

*Earth-fixed coordinate y of the satellite [m]. Completes a right-handed, Earth-Centered, Earth-Fixed orthogonal coordinate system.*

- double [d\\_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 [d\\_satvel\\_X](#) {}

*Earth-fixed velocity coordinate x of the satellite [m].*

- double [d\\_satvel\\_Y](#) {}

*Earth-fixed velocity coordinate y of the satellite [m].*

- double [d\\_satvel\\_Z](#) {}

*Earth-fixed velocity coordinate z of the satellite [m].*

- std::map< int, std::string > [satelliteBlock](#)

*Map that stores to which block the PRN belongs <https://www.navcen.uscg.gov/?Do=constellation&Status>.*

### 10.170.1 Detailed Description

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

See <https://www.gps.gov/technical/icwg/IS-GPS-200K.pdf> Appendix II

Definition at line 38 of file `gps_ephemeris.h`.

### 10.170.2 Constructor & Destructor Documentation

#### 10.170.2.1 Gps\_Ephemeris()

```
Gps_Ephemeris::Gps_Ephemeris ( )
```

Default constructor

### 10.170.3 Member Function Documentation

#### 10.170.3.1 satellitePosition()

```
double Gps_Ephemeris::satellitePosition (
    double transmitTime )
```

Compute the ECEF SV coordinates and ECEF velocity Implementation of Table 20-IV (IS-GPS-200K) and compute the clock bias term including relativistic effect (return value)

10.170.3.2 `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.

- < Time of GPS Week of the ephemeris set (taken from subframes TOW) [s]
- < Amplitude of the Sine Harmonic Correction Term to the Orbit Radius [m]
- < Mean Motion Difference From Computed Value [semi-circles/s]
- < Mean Anomaly at Reference Time [semi-circles]
- < Amplitude of the Cosine Harmonic Correction Term to the Argument of Latitude [rad]
- < Eccentricity [dimensionless]
- < Amplitude of the Sine Harmonic Correction Term to the Argument of Latitude [rad]
- < Square Root of the Semi-Major Axis [sqrt(m)]
- < Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200K) [s]
- < clock data reference time (Ref. 20.3.3.3.1 IS-GPS-200K) [s]
- < Amplitude of the Cosine Harmonic Correction Term to the Angle of Inclination [rad]
- < Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles]
- < Amplitude of the Sine Harmonic Correction Term to the Angle of Inclination [rad]
- < Inclination Angle at Reference Time [semi-circles]
- < Amplitude of the Cosine Harmonic Correction Term to the Orbit Radius [m]
- < Argument of Perigee [semi-circles]
- < Rate of Right Ascension [semi-circles/s]
- < Rate of Inclination Angle [semi-circles/s]
- < If 1, P code ON in L2; if 2, C/A code ON in L2;
- < GPS week number, aka WN [week]
- < When true, indicates that the NAV data stream was commanded OFF on the P-code of the L2 channel
- < 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-200K)
- < Estimated Group Delay Differential: L1-L2 correction term only for the benefit of "L1 P(Y)" or "L2 P(Y)" s users [s]
- < Issue of Data, Clock

< 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]

< 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.

< Coefficient 0 of code phase offset model [s]

< Coefficient 1 of code phase offset model [s/s]

< Coefficient 2 of code phase offset model [s/s<sup>2</sup>]

< If true, indicates that the SV URA may be worse than indicated in d\_SV\_accuracy, use that SV at our own risk.

< If true, the AntiSpoofing mode is ON in that SV

Definition at line 140 of file gps\_ephemeris.h.

References b\_alert\_flag, b\_antispoofing\_flag, b\_fit\_interval\_flag, b\_integrity\_status\_flag, b\_L2\_P\_data\_flag, d\_A\_f0, d\_A\_f1, d\_A\_f2, d\_Cic, d\_Cis, d\_Crc, d\_Crs, d\_Cuc, d\_Cus, d\_Delta\_n, d\_e\_eccentricity, d\_i\_0, d\_IDOT, d\_IODC, d\_IODE\_SF2, d\_IODE\_SF3, d\_M\_0, d\_OMEGA, d\_OMEGA0, d\_OMEGA\_DOT, d\_sqrt\_A, d\_TGD, d\_Toc, d\_Toe, d\_TOW, i\_AODO, i\_code\_on\_L2, i\_GPS\_week, and i\_SV\_accuracy.

#### 10.170.3.3 sv\_clock\_drift()

```
double Gps_Ephemeris::sv_clock_drift (
    double transmitTime )
```

Sets (*d\_satClkDrift*) and returns the clock drift in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200K, 20.3.3.3.3.1)

#### 10.170.3.4 sv\_clock\_relativistic\_term()

```
double Gps_Ephemeris::sv_clock_relativistic_term (
    double transmitTime )
```

Sets (*d\_dtr*) and returns the clock relativistic correction term in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200K, 20.3.3.3.3.1)

### 10.170.4 Member Data Documentation

#### 10.170.4.1 b\_alert\_flag

```
bool Gps_Ephemeris::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.

Definition at line 116 of file gps\_ephemeris.h.

Referenced by serialize().

#### 10.170.4.2 b\_antispoofing\_flag

```
bool Gps_Ephemeris::b_antispoofing_flag {}
```

If true, the AntiSpoofing mode is ON in that SV.

Definition at line 117 of file gps\_ephemeris.h.

Referenced by serialize().

#### 10.170.4.3 b\_fit\_interval\_flag

```
bool Gps_Ephemeris::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.

Definition at line 95 of file gps\_ephemeris.h.

Referenced by serialize().

#### 10.170.4.4 b\_integrity\_status\_flag

```
bool Gps_Ephemeris::b_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 115 of file gps\_ephemeris.h.

Referenced by serialize().

#### 10.170.4.5 b\_L2\_P\_data\_flag

```
bool Gps_Ephemeris::b_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 86 of file `gps_ephemeris.h`.

Referenced by `serialize()`.

#### 10.170.4.6 d\_A\_f0

```
double Gps_Ephemeris::d_A_f0 {}
```

Coefficient 0 of code phase offset model [s].

Definition at line 99 of file `gps_ephemeris.h`.

Referenced by `serialize()`.

#### 10.170.4.7 d\_A\_f1

```
double Gps_Ephemeris::d_A_f1 {}
```

Coefficient 1 of code phase offset model [s/s].

Definition at line 100 of file `gps_ephemeris.h`.

Referenced by `serialize()`.

#### 10.170.4.8 d\_A\_f2

```
double Gps_Ephemeris::d_A_f2 {}
```

Coefficient 2 of code phase offset model [s/s<sup>2</sup>].

Definition at line 101 of file `gps_ephemeris.h`.

Referenced by `serialize()`.

#### 10.170.4.9 d\_Cic

```
double Gps_Ephemeris::d_Cic {}
```

Amplitude of the Cosine Harmonic Correction Term to the Angle of Inclination [rad].

Definition at line 76 of file gps\_ephemeris.h.

Referenced by `serialize()`.

#### 10.170.4.10 d\_Cis

```
double Gps_Ephemeris::d_Cis {}
```

Amplitude of the Sine Harmonic Correction Term to the Angle of Inclination [rad].

Definition at line 78 of file gps\_ephemeris.h.

Referenced by `serialize()`.

#### 10.170.4.11 d\_Crc

```
double Gps_Ephemeris::d_Crc {}
```

Amplitude of the Cosine Harmonic Correction Term to the Orbit Radius [m].

Definition at line 80 of file gps\_ephemeris.h.

Referenced by `serialize()`.

#### 10.170.4.12 d\_Crs

```
double Gps_Ephemeris::d_Crs {}
```

Amplitude of the Sine Harmonic Correction Term to the Orbit Radius [m].

Definition at line 67 of file gps\_ephemeris.h.

Referenced by `serialize()`.

#### 10.170.4.13 d\_Cuc

```
double Gps_Ephemeris::d_Cuc {}
```

Amplitude of the Cosine Harmonic Correction Term to the Argument of Latitude [rad].

Definition at line 70 of file gps\_ephemeris.h.

Referenced by `serialize()`.

#### 10.170.4.14 d\_Cus

```
double Gps_Ephemeris::d_Cus {}
```

Amplitude of the Sine Harmonic Correction Term to the Argument of Latitude [rad].

Definition at line 72 of file gps\_ephemeris.h.

Referenced by `serialize()`.

#### 10.170.4.15 d\_Delta\_n

```
double Gps_Ephemeris::d_Delta_n {}
```

Mean Motion Difference From Computed Value [semi-circles/s].

Definition at line 68 of file gps\_ephemeris.h.

Referenced by `serialize()`.

#### 10.170.4.16 d\_dtr

```
double Gps_Ephemeris::d_dtr {}
```

relativistic clock correction term

Definition at line 121 of file gps\_ephemeris.h.

**10.170.4.17 d\_e\_eccentricity**

```
double Gps_Ephemeris::d_e_eccentricity {}
```

Eccentricity [dimensionless].

Definition at line 71 of file gps\_ephemeris.h.

Referenced by `serialize()`.

**10.170.4.18 d\_i\_0**

```
double Gps_Ephemeris::d_i_0 {}
```

Inclination Angle at Reference Time [semi-circles].

Definition at line 79 of file gps\_ephemeris.h.

Referenced by `serialize()`.

**10.170.4.19 d\_IDOT**

```
double Gps_Ephemeris::d_IDOT {}
```

Rate of Inclination Angle [semi-circles/s].

Definition at line 83 of file gps\_ephemeris.h.

Referenced by `serialize()`.

**10.170.4.20 d\_IODC**

```
int32_t Gps_Ephemeris::d_IODC {}
```

Issue of Data, Clock.

Definition at line 90 of file gps\_ephemeris.h.

Referenced by `serialize()`.

#### 10.170.4.21 d\_IODE\_SF2

```
int32_t Gps_Ephemeris::d_IODE_SF2 {}
```

Issue of Data, Ephemeris (IODE), subframe 2.

Definition at line 91 of file gps\_ephemeris.h.

Referenced by `serialize()`.

#### 10.170.4.22 d\_IODE\_SF3

```
int32_t Gps_Ephemeris::d_IODE_SF3 {}
```

Issue of Data, Ephemeris (IODE), subframe 3.

Definition at line 92 of file gps\_ephemeris.h.

Referenced by `serialize()`.

#### 10.170.4.23 d\_M\_0

```
double Gps_Ephemeris::d_M_0 {}
```

Mean Anomaly at Reference Time [semi-circles].

Definition at line 69 of file gps\_ephemeris.h.

Referenced by `serialize()`.

#### 10.170.4.24 d\_OMEGA

```
double Gps_Ephemeris::d_OMEGA {}
```

Argument of Perigee [semi-circles].

Definition at line 81 of file gps\_ephemeris.h.

Referenced by `serialize()`.

**10.170.4.25 d\_OMEGA0**

```
double Gps_Ephemeris::d_OMEGA0 {}
```

Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles].

Definition at line 77 of file gps\_ephemeris.h.

Referenced by `serialize()`.

**10.170.4.26 d\_OMEGA\_DOT**

```
double Gps_Ephemeris::d_OMEGA_DOT {}
```

Rate of Right Ascension [semi-circles/s].

Definition at line 82 of file gps\_ephemeris.h.

Referenced by `serialize()`.

**10.170.4.27 d\_satClkDrift**

```
double Gps_Ephemeris::d_satClkDrift {}
```

GPS clock error.

Definition at line 120 of file gps\_ephemeris.h.

**10.170.4.28 d\_satpos\_X**

```
double Gps_Ephemeris::d_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 124 of file gps\_ephemeris.h.

**10.170.4.29 d\_satpos\_Y**

```
double Gps_Ephemeris::d_satpos_Y {}
```

Earth-fixed coordinate y of the satellite [m]. Completes a right-handed, Earth-Centered, Earth-Fixed orthogonal coordinate system.

Definition at line 125 of file gps\_ephemeris.h.

**10.170.4.30 d\_satpos\_Z**

```
double Gps_Ephemeris::d_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 126 of file gps\_ephemeris.h.

**10.170.4.31 d\_satvel\_X**

```
double Gps_Ephemeris::d_satvel_X {}
```

Earth-fixed velocity coordinate x of the satellite [m].

Definition at line 129 of file gps\_ephemeris.h.

**10.170.4.32 d\_satvel\_Y**

```
double Gps_Ephemeris::d_satvel_Y {}
```

Earth-fixed velocity coordinate y of the satellite [m].

Definition at line 130 of file gps\_ephemeris.h.

**10.170.4.33 d\_satvel\_Z**

```
double Gps_Ephemeris::d_satvel_Z {}
```

Earth-fixed velocity coordinate z of the satellite [m].

Definition at line 131 of file gps\_ephemeris.h.

**10.170.4.34 d\_sqrt\_A**

```
double Gps_Ephemeris::d_sqrt_A {}
```

Square Root of the Semi-Major Axis [sqrt(m)].

Definition at line 73 of file gps\_ephemeris.h.

Referenced by `serialize()`.

**10.170.4.35 d\_TGD**

```
double Gps_Ephemeris::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].

Definition at line 89 of file gps\_ephemeris.h.

Referenced by `serialize()`.

**10.170.4.36 d\_Toc**

```
int32_t Gps_Ephemeris::d_Toc {}
```

clock data reference time (Ref. 20.3.3.3.1 IS-GPS-200K) [s]

Definition at line 75 of file gps\_ephemeris.h.

Referenced by `serialize()`.

**10.170.4.37 d\_Toe**

```
int32_t Gps_Ephemeris::d_Toe {}
```

Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200K) [s].

Definition at line 74 of file gps\_ephemeris.h.

Referenced by `serialize()`.

**10.170.4.38 d\_TOW**

```
int32_t Gps_Ephemeris::d_TOW {}
```

Time of GPS Week of the ephemeris set (taken from subframes TOW) [s].

Definition at line 66 of file gps\_ephemeris.h.

Referenced by `serialize()`.

#### 10.170.4.39 i\_AODO

```
int32_t Gps_Ephemeris::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].

Definition at line 93 of file `gps_ephemeris.h`.

Referenced by `serialize()`.

#### 10.170.4.40 i\_code\_on\_L2

```
int32_t Gps_Ephemeris::i_code_on_L2 {}
```

If 1, P code ON in L2; if 2, C/A code ON in L2;.

Definition at line 84 of file `gps_ephemeris.h`.

Referenced by `serialize()`.

#### 10.170.4.41 i\_GPS\_week

```
int32_t Gps_Ephemeris::i_GPS_week {}
```

GPS week number, aka WN [week].

Definition at line 85 of file `gps_ephemeris.h`.

Referenced by `serialize()`.

#### 10.170.4.42 i\_SV\_accuracy

```
int32_t Gps_Ephemeris::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-200K)

Definition at line 87 of file `gps_ephemeris.h`.

Referenced by `serialize()`.

## 10.170.4.43 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 133 of file gps\_ephemeris.h.

The documentation for this class was generated from the following file:

- [gps\\_ephemeris.h](#)

## 10.171 Gps\_Iono Class Reference

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

```
#include <gps_iono.h>
```

### 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

- `bool valid {}`  
*Valid flag.*
- `double d\_alpha0 {}`  
*Coefficient 0 of a cubic equation representing the amplitude of the vertical delay [s].*
- `double d\_alpha1 {}`  
*Coefficient 1 of a cubic equation representing the amplitude of the vertical delay [s/semi-circle].*
- `double d\_alpha2 {}`  
*Coefficient 2 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)<sup>2</sup>].*
- `double d\_alpha3 {}`  
*Coefficient 3 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)<sup>3</sup>].*
- `double d\_beta0 {}`  
*Coefficient 0 of a cubic equation representing the period of the model [s].*
- `double d\_beta1 {}`  
*Coefficient 1 of a cubic equation representing the period of the model [s/semi-circle].*
- `double d\_beta2 {}`  
*Coefficient 2 of a cubic equation representing the period of the model [s(semi-circle)<sup>2</sup>].*
- `double d\_beta3 {}`  
*Coefficient 3 of a cubic equation representing the period of the model [s(semi-circle)<sup>3</sup>].*

### 10.171.1 Detailed Description

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

See <https://www.gps.gov/technical/icwg/IS-GPS-200K.pdf> Appendix II

Definition at line 35 of file `gps_iono.h`.

### 10.171.2 Constructor & Destructor Documentation

#### 10.171.2.1 Gps\_Iono()

```
Gps_Iono::Gps_Iono ( ) [default]
```

Default constructor.

### 10.171.3 Member Function Documentation

#### 10.171.3.1 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 56 of file `gps_iono.h`.

References `d_alpha0`, `d_alpha1`, `d_alpha2`, `d_alpha3`, `d_beta0`, `d_beta1`, `d_beta2`, and `d_beta3`.

### 10.171.4 Member Data Documentation

#### 10.171.4.1 d\_alpha0

```
double Gps_Iono::d_alpha0 {}
```

Coefficient 0 of a cubic equation representing the amplitude of the vertical delay [s].

Definition at line 40 of file `gps_iono.h`.

Referenced by `serialize()`.

#### 10.171.4.2 d\_alpha1

```
double Gps_Iono::d_alpha1 {}
```

Coefficient 1 of a cubic equation representing the amplitude of the vertical delay [s/semi-circle].

Definition at line 41 of file gps\_iono.h.

Referenced by `serialize()`.

#### 10.171.4.3 d\_alpha2

```
double Gps_Iono::d_alpha2 {}
```

Coefficient 2 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)<sup>2</sup>].

Definition at line 42 of file gps\_iono.h.

Referenced by `serialize()`.

#### 10.171.4.4 d\_alpha3

```
double Gps_Iono::d_alpha3 {}
```

Coefficient 3 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)<sup>3</sup>].

Definition at line 43 of file gps\_iono.h.

Referenced by `serialize()`.

#### 10.171.4.5 d\_beta0

```
double Gps_Iono::d_beta0 {}
```

Coefficient 0 of a cubic equation representing the period of the model [s].

Definition at line 44 of file gps\_iono.h.

Referenced by `serialize()`.

#### 10.171.4.6 d\_beta1

```
double Gps_Iono::d_beta1 {}
```

Coefficient 1 of a cubic equation representing the period of the model [s/semi-circle].

Definition at line 45 of file `gps_iono.h`.

Referenced by `serialize()`.

#### 10.171.4.7 d\_beta2

```
double Gps_Iono::d_beta2 {}
```

Coefficient 2 of a cubic equation representing the period of the model [s(semi-circle)<sup>2</sup>].

Definition at line 46 of file `gps_iono.h`.

Referenced by `serialize()`.

#### 10.171.4.8 d\_beta3

```
double Gps_Iono::d_beta3 {}
```

Coefficient 3 of a cubic equation representing the period of the model [s(semi-circle)<sup>3</sup>].

Definition at line 47 of file `gps_iono.h`.

Referenced by `serialize()`.

#### 10.171.4.9 valid

```
bool Gps_Iono::valid {}
```

Valid flag.

Definition at line 38 of file `gps_iono.h`.

The documentation for this class was generated from the following file:

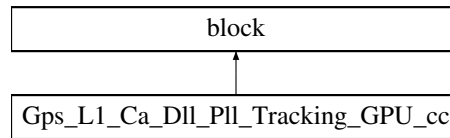
- [gps\\_iono.h](#)

## 10.172 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)

### 10.172.1 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:

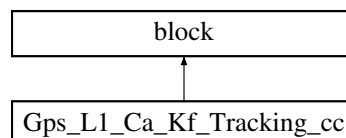
- [gps\\_l1\\_ca\\_dll\\_pll\\_tracking\\_gpu\\_cc.h](#)

## 10.173 Gps\_L1\_Ca\_Kf\_Tracking\_cc Class Reference

This class implements a DLL + PLL tracking loop block.

```
#include <gps_l1_ca_kf_tracking_cc.h>
```

Inheritance diagram for Gps\_L1\_Ca\_Kf\_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\\_kf\\_tracking\\_cc\\_sptr](#) **gps\_l1\_ca\_kf\_make\_tracking\_cc** (uint32\_t order, int64\_t if\_freq, 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)

### 10.173.1 Detailed Description

This class implements a DLL + PLL tracking loop block.

Definition at line 74 of file [gps\\_l1\\_ca\\_kf\\_tracking\\_cc.h](#).

The documentation for this class was generated from the following file:

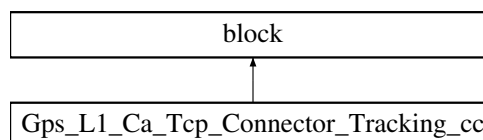
- [gps\\_l1\\_ca\\_kf\\_tracking\\_cc.h](#)

## 10.174 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)

### 10.174.1 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:

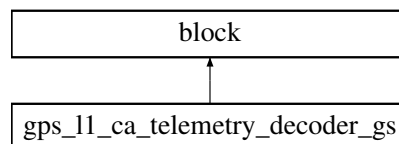
- [gps\\_l1\\_ca\\_tcp\\_connector\\_tracking\\_cc.h](#)

## 10.175 gps\_l1\_ca\_telemetry\_decoder\_gs Class Reference

This class implements a block that decodes the NAV data defined in IS-GPS-200K.

```
#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)  
*Set satellite PRN.*
- void **set\_channel** (int channel)  
*Set receiver's channel.*
- void **reset** ()
- int **general\_work** (int noutput\_items, gr\_vector\_int &ninput\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)  
*This is where all signal processing takes place.*

## Friends

- `gps_l1_ca_telemetry_decoder_gs_sptr` **gps\_l1\_ca\_make\_telemetry\_decoder\_gs** (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)

### 10.175.1 Detailed Description

This class implements a block that decodes the NAV data defined in IS-GPS-200K.

Definition at line 52 of file `gps_l1_ca_telemetry_decoder_gs.h`.

### 10.175.2 Member Function Documentation

#### 10.175.2.1 `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 )
```

This is where all signal processing takes place.

#### 10.175.2.2 `set_channel()`

```
void gps_l1_ca_telemetry_decoder_gs::set_channel (
    int channel )
```

Set receiver's channel.

#### 10.175.2.3 `set_satellite()`

```
void gps_l1_ca_telemetry_decoder_gs::set_satellite (
    const Gnss_Satellite & satellite )
```

Set satellite PRN.

The documentation for this class was generated from the following file:

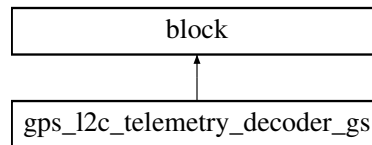
- [gps\\_l1\\_ca\\_telemetry\\_decoder\\_gs.h](#)

## 10.176 gps\_l2c\_telemetry\_decoder\_gs Class Reference

This class implements a block that decodes CNAV data defined in IS-GPS-200K.

```
#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)  
*Set satellite PRN.*
- void [set\\_channel](#) (int32\_t channel)  
*Set receiver's channel.*
- void [reset](#) ()
- int [general\\_work](#) (int noutput\_items, gr\_vector\_int &ninput\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)  
*This is where all signal processing takes place.*

### Friends

- [gps\\_l2c\\_telemetry\\_decoder\\_gs\\_sptr](#) [gps\\_l2c\\_make\\_telemetry\\_decoder\\_gs](#) (const [Gnss\\_Satellite](#) &satellite, const [Tim\\_Conf](#) &conf)

### 10.176.1 Detailed Description

This class implements a block that decodes CNAV data defined in IS-GPS-200K.

Definition at line 53 of file `gps_l2c_telemetry_decoder_gs.h`.

### 10.176.2 Member Function Documentation

#### 10.176.2.1 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 )
```

This is where all signal processing takes place.

### 10.176.2.2 set\_channel()

```
void gps_l2c_telemetry_decoder_gs::set_channel (
    int32_t channel )
```

Set receiver's channel.

### 10.176.2.3 set\_satellite()

```
void gps_l2c_telemetry_decoder_gs::set_satellite (
    const Gnss_Satellite & satellite )
```

Set satellite PRN.

The documentation for this class was generated from the following file:

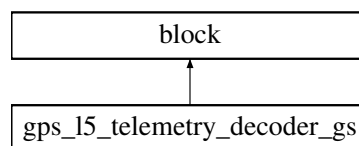
- [gps\\_l2c\\_telemetry\\_decoder\\_gs.h](#)

## 10.177 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)  
*Set satellite PRN.*
- void [set\\_channel](#) (int32\_t channel)  
*Set receiver's channel.*
- void **reset** ()
- 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

- `gps_l5_telemetry_decoder_gs_sptr` **gps\_l5\_make\_telemetry\_decoder\_gs** (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)

### 10.177.1 Detailed Description

This class implements a GPS L5 Telemetry decoder.

Definition at line 55 of file `gps_l5_telemetry_decoder_gs.h`.

### 10.177.2 Member Function Documentation

#### 10.177.2.1 `set_channel()`

```
void gps_l5_telemetry_decoder_gs::set_channel (
    int32_t channel )
```

Set receiver's channel.

#### 10.177.2.2 `set_satellite()`

```
void gps_l5_telemetry_decoder_gs::set_satellite (
    const Gnss_Satellite & satellite )
```

Set satellite PRN.

The documentation for this class was generated from the following file:

- [gps\\_l5\\_telemetry\\_decoder\\_gs.h](#)

## 10.178 Gps\_Navigation\_Message Class Reference

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

```
#include <gps_navigation_message.h>
```

## Public Member Functions

- [Gps\\_Navigation\\_Message](#) ()
- [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\\_Utc\\_Model](#) [get\\_utc\\_model](#) ()  
*Obtain a GPS UTC model parameters class filled with current SV data.*
- `int32_t` [subframe\\_decoder](#) (`char *subframe`)  
*Decodes the GPS NAV message.*
- `double` [utc\\_time](#) (`const double gpstime_corrected`) const  
*Computes the Coordinated Universal Time (UTC) and returns it in [s](#)*
- `int32_t` [get\\_TOW](#) () const  
*Gets Time of Week, in seconds.*
- `int32_t` [get\\_GPS\\_week](#) () const  
*Sets Time of Week, in seconds.*
- `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.*
- `bool` [satellite\\_validation](#) ()

### 10.178.1 Detailed Description

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

See <https://www.gps.gov/technical/icwg/IS-GPS-200K.pdf> Appendix II

Definition at line 45 of file `gps_navigation_message.h`.

### 10.178.2 Constructor & Destructor Documentation

#### 10.178.2.1 [Gps\\_Navigation\\_Message](#)()

```
Gps_Navigation_Message::Gps_Navigation_Message ( )
```

Default constructor

### 10.178.3 Member Function Documentation

#### 10.178.3.1 get\_ephemeris()

`Gps_Ephemeris Gps_Navigation_Message::get_ephemeris ( ) const`

Obtain a GPS SV Ephemeris class filled with current SV data.

#### 10.178.3.2 get\_flag\_iono\_valid()

`bool Gps_Navigation_Message::get_flag_iono_valid ( ) const [inline]`

Gets flag\_iono\_valid.

Definition at line 122 of file gps\_navigation\_message.h.

#### 10.178.3.3 get\_flag\_utc\_model\_valid()

`bool Gps_Navigation_Message::get_flag_utc_model_valid ( ) const [inline]`

Gets flag\_utc\_model\_valid.

Definition at line 130 of file gps\_navigation\_message.h.

#### 10.178.3.4 get\_GPS\_week()

`int32_t Gps_Navigation_Message::get_GPS_week ( ) const [inline]`

Sets Time of Week, in seconds.

Definition at line 90 of file gps\_navigation\_message.h.

#### 10.178.3.5 get\_iono()

`Gps_Iono Gps_Navigation_Message::get_iono ( )`

Obtain a GPS ionospheric correction parameters class filled with current SV data.

#### 10.178.3.6 `get_satellite_PRN()`

```
uint32_t Gps_Navigation_Message::get_satellite_PRN ( ) const [inline]
```

Gets satellite PRN number.

Definition at line 106 of file `gps_navigation_message.h`.

#### 10.178.3.7 `get_TOW()`

```
int32_t Gps_Navigation_Message::get_TOW ( ) const [inline]
```

Gets Time of Week, in seconds.

Definition at line 82 of file `gps_navigation_message.h`.

#### 10.178.3.8 `get_utc_model()`

```
Gps_Utc_Model Gps_Navigation_Message::get_utc_model ( )
```

Obtain a GPS UTC model parameters class filled with current SV data.

#### 10.178.3.9 `set_channel()`

```
void Gps_Navigation_Message::set_channel (
    int32_t channel_id ) [inline]
```

Sets channel ID.

Definition at line 114 of file `gps_navigation_message.h`.

#### 10.178.3.10 `set_satellite_PRN()`

```
void Gps_Navigation_Message::set_satellite_PRN (
    uint32_t prn ) [inline]
```

Sets satellite PRN number.

Definition at line 98 of file `gps_navigation_message.h`.

## 10.178.3.11 subframe\_decoder()

```
int32_t Gps_Navigation_Message::subframe_decoder (
    char * subframe )
```

Decodes the GPS NAV message.

## 10.178.3.12 utc\_time()

```
double Gps_Navigation_Message::utc_time (
    const double gpstime_corrected ) const
```

Computes the Coordinated Universal Time (UTC) and returns it in [s](#)

The documentation for this class was generated from the following file:

- [gps\\_navigation\\_message.h](#)

## 10.179 Gps\_Utc\_Model Class Reference

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

```
#include <gps_utc_model.h>
```

## Public Member Functions

- [Gps\\_Utc\\_Model](#) ()=default
- `template<class Archive >`  
void **serialize** (Archive &archive, const uint32\_t version)

## Public Attributes

- double [d\\_A0](#) {}  
*Constant of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200K) [s].*
- double [d\\_A1](#) {}  
*1st order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200K) [s/s]*
- double [d\\_A2](#) {}  
*2nd order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200K) [s/s]*
- int32\_t [d\\_t\\_OT](#) {}  
*Reference time for UTC data (reference 20.3.4.5 and 20.3.3.5.2.4 IS-GPS-200K) [s].*
- int32\_t [i\\_WN\\_T](#) {}  
*UTC reference week number [weeks].*
- 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.*
- int32\_t [i\\_WN\\_LSF](#) {}  
*Week number at the end of which the leap second becomes effective [weeks].*
- int32\_t [i\\_DN](#) {}  
*Day number (DN) at the end of which the leap second becomes effective [days].*
- 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].*
- bool **valid** {}

### 10.179.1 Detailed Description

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

See <https://www.gps.gov/technical/icwg/IS-GPS-200K.pdf> Appendix II

Definition at line 35 of file gps\_utc\_model.h.

### 10.179.2 Constructor & Destructor Documentation

#### 10.179.2.1 Gps\_Utc\_Model()

```
Gps_Utc_Model::Gps_Utc_Model ( ) [default]
```

Default constructor

### 10.179.3 Member Data Documentation

#### 10.179.3.1 d\_A0

```
double Gps_Utc_Model::d_A0 {}
```

Constant of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200K) [s].

Definition at line 44 of file gps\_utc\_model.h.

#### 10.179.3.2 d\_A1

```
double Gps_Utc_Model::d_A1 {}
```

1st order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200K) [s/s]

Definition at line 45 of file gps\_utc\_model.h.

#### 10.179.3.3 d\_A2

```
double Gps_Utc_Model::d_A2 {}
```

2nd order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200K) [s/s]

Definition at line 46 of file gps\_utc\_model.h.

#### 10.179.3.4 d\_DeltaT\_LS

```
int32_t Gps_Utc_Model::d_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`.

#### 10.179.3.5 d\_DeltaT\_LSF

```
int32_t Gps_Utc_Model::d_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`.

#### 10.179.3.6 d\_t\_OT

```
int32_t Gps_Utc_Model::d_t_OT {}
```

Reference time for UTC data (reference 20.3.4.5 and 20.3.3.5.2.4 IS-GPS-200K) [s].

Definition at line 47 of file `gps_utc_model.h`.

#### 10.179.3.7 i\_DN

```
int32_t Gps_Utc_Model::i_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`.

#### 10.179.3.8 i\_WN\_LSF

```
int32_t Gps_Utc_Model::i_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`.

## 10.179.3.9 i\_WN\_T

```
int32_t Gps_Utc_Model::i_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:

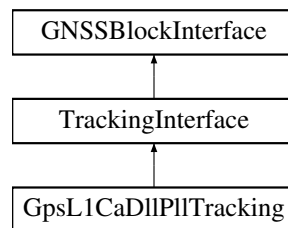
- [gps\\_utc\\_model.h](#)

## 10.180 GpsL1CaDIPIITracking Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#include <gps_l1_ca_dll_pll_tracking.h>
```

Inheritance diagram for GpsL1CaDIPIITracking:



## Public Member Functions

- **GpsL1CaDIPIITracking** (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\_DLL\_PLL\_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.*

### 10.180.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 44 of file `gps_l1_ca_dll_pll_tracking.h`.

### 10.180.2 Member Function Documentation

#### 10.180.2.1 implementation()

```
std::string GpsL1CaDllPllTracking::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L1\_CA\_DLL\_PLL\_Tracking".

Implements [GNSSBlockInterface](#).

Definition at line 61 of file `gps_l1_ca_dll_pll_tracking.h`.

#### 10.180.2.2 set\_channel()

```
void GpsL1CaDllPllTracking::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.180.2.3 set\_gnss\_synchro()

```
void GpsL1CaDllPllTracking::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](#).

### 10.180.2.4 stop\_tracking()

```
void GpsL1CaDllPllTracking::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

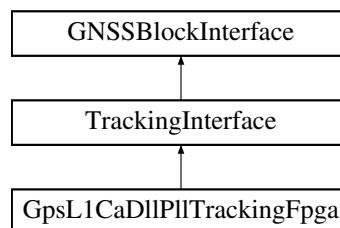
- [gps\\_l1\\_ca\\_dll\\_pll\\_tracking.h](#)

## 10.181 GpsL1CaDllPllTrackingFpga Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#include <gps_l1_ca_dll_pll_tracking_fpga.h>
```

Inheritance diagram for GpsL1CaDllPllTrackingFpga:



### Public Member Functions

- [GpsL1CaDllPllTrackingFpga](#) (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)  
*Constructor.*
- virtual [~GpsL1CaDllPllTrackingFpga](#) ()  
*Destructor.*
- std::string [role](#) () override  
*Role.*
- std::string [implementation](#) () override  
*Returns "GPS\_L1\_CA\_DLL\_PLL\_Tracking\_Fpga".*
- size\_t [item\\_size](#) () override  
*Returns size of lv\_16sc\_t.*
- void [connect](#) (gr::top\_block\_sptr top\_block) override  
*Connect.*
- void [disconnect](#) (gr::top\_block\_sptr top\_block) override  
*Disconnect.*
- gr::basic\_block\_sptr [get\\_left\\_block](#) () override  
*Get left block.*
- gr::basic\_block\_sptr [get\\_right\\_block](#) () override  
*Get right block.*

- 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  
*Start the tracking process in the FPGA.*
- void [stop\\_tracking](#) () override  
*Stop the tracking process in the FPGA.*

### 10.181.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 41 of file `gps_l1_ca_dll_pll_tracking_fpga.h`.

### 10.181.2 Constructor & Destructor Documentation

#### 10.181.2.1 GpsL1CaDllPllTrackingFpga()

```
GpsL1CaDllPllTrackingFpga::GpsL1CaDllPllTrackingFpga (
    const ConfigurationInterface * configuration,
    const std::string & role,
    unsigned int in_streams,
    unsigned int out_streams )
```

Constructor.

#### 10.181.2.2 ~GpsL1CaDllPllTrackingFpga()

```
virtual GpsL1CaDllPllTrackingFpga::~~GpsL1CaDllPllTrackingFpga ( ) [virtual]
```

Destructor.

### 10.181.3 Member Function Documentation

#### 10.181.3.1 connect()

```
void GpsL1CaDllPllTrackingFpga::connect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Connect.

Implements [GNSSBlockInterface](#).

#### 10.181.3.2 disconnect()

```
void GpsL1CaDllPllTrackingFpga::disconnect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Disconnect.

Implements [GNSSBlockInterface](#).

#### 10.181.3.3 get\_left\_block()

```
gr::basic_block_sptr GpsL1CaDllPllTrackingFpga::get_left_block ( ) [override], [virtual]
```

Get left block.

Implements [GNSSBlockInterface](#).

#### 10.181.3.4 get\_right\_block()

```
gr::basic_block_sptr GpsL1CaDllPllTrackingFpga::get_right_block ( ) [override], [virtual]
```

Get right block.

Implements [GNSSBlockInterface](#).

#### 10.181.3.5 implementation()

```
std::string GpsL1CaDllPllTrackingFpga::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L1\_CA\_DLL\_PLL\_Tracking\_Fpga".

Implements [GNSSBlockInterface](#).

Definition at line 69 of file `gps_l1_ca_dll_pll_tracking_fpga.h`.

### 10.181.3.6 item\_size()

```
size_t GpsL1CaDllPllTrackingFpga::item_size ( ) [inline], [override], [virtual]
```

Returns size of `lv_16sc_t`.

Implements [GNSSBlockInterface](#).

Definition at line 77 of file `gps_l1_ca_dll_pll_tracking_fpga.h`.

### 10.181.3.7 role()

```
std::string GpsL1CaDllPllTrackingFpga::role ( ) [inline], [override], [virtual]
```

Role.

Implements [GNSSBlockInterface](#).

Definition at line 61 of file `gps_l1_ca_dll_pll_tracking_fpga.h`.

### 10.181.3.8 set\_channel()

```
void GpsL1CaDllPllTrackingFpga::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

### 10.181.3.9 set\_gnss\_synchro()

```
void GpsL1CaDllPllTrackingFpga::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](#).

### 10.181.3.10 start\_tracking()

```
void GpsL1CaDllPllTrackingFpga::start_tracking ( ) [override], [virtual]
```

Start the tracking process in the FPGA.

Implements [TrackingInterface](#).

### 10.181.3.11 stop\_tracking()

```
void GpsL1CaDllPllTrackingFpga::stop_tracking ( ) [override], [virtual]
```

Stop the tracking process in the FPGA.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

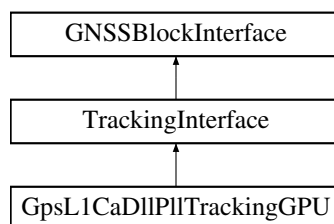
- [gps\\_l1\\_ca\\_dll\\_pll\\_tracking\\_fpga.h](#)

## 10.182 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.

### 10.182.1 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`.

### 10.182.2 Member Function Documentation

#### 10.182.2.1 `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`.

#### 10.182.2.2 `set_channel()`

```
void GpsL1CaDllPllTrackingGPU::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.182.2.3 `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](#).

#### 10.182.2.4 stop\_tracking()

```
void GpsL1CaDllPllTrackingGPU::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

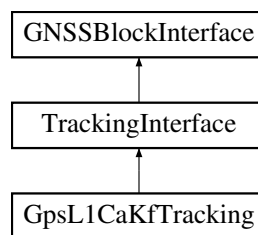
- [gps\\_l1\\_ca\\_dll\\_pll\\_tracking\\_gpu.h](#)

### 10.183 GpsL1CaKfTracking Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#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.*

### 10.183.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 44 of file `gps_l1_ca_kf_tracking.h`.

### 10.183.2 Member Function Documentation

#### 10.183.2.1 implementation()

```
std::string GpsL1CaKfTracking::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L1\_CA\_KF\_Tracking".

Implements [GNSSBlockInterface](#).

Definition at line 61 of file `gps_l1_ca_kf_tracking.h`.

#### 10.183.2.2 set\_channel()

```
void GpsL1CaKfTracking::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.183.2.3 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](#).

### 10.183.2.4 stop\_tracking()

```
void GpsL1CaKfTracking::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

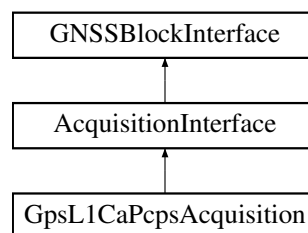
- [gps\\_l1\\_ca\\_kf\\_tracking.h](#)

## 10.184 GpsL1CaPcpsAcquisition Class Reference

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

```
#include <gps_l1_ca_pcps_acquisition.h>
```

Inheritance diagram for GpsL1CaPcpsAcquisition:



### Public Member Functions

- **GpsL1CaPcpsAcquisition** (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\_PCPS\_Acquisition".*
- 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\_threshold** (float threshold) override  
*Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override

- *Set maximum Doppler off grid search.*  
void [set\\_doppler\\_step](#) (unsigned int doppler\_step) override
- *Set Doppler steps for the grid search.*  
void [set\\_doppler\\_center](#) (int doppler\_center) override
- *Set Doppler center for the grid search.*  
void [init](#) () override
- *Initializes acquisition algorithm.*  
void [set\\_local\\_code](#) () override
- *Sets local code for GPS L1/CA PCPS acquisition algorithm.*  
signed int [mag](#) () override
- *Returns the maximum peak of grid search.*  
void [reset](#) () override
- *Restart acquisition algorithm.*  
void [set\\_state](#) (int state) override
- *If state = 1, it forces the block to start acquiring from the first sample.*  
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.*

### 10.184.1 Detailed Description

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

Definition at line 49 of file `gps_l1_ca_pcps_acquisition.h`.

### 10.184.2 Member Function Documentation

#### 10.184.2.1 implementation()

```
std::string GpsL1CaPcpsAcquisition::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L1\_CA\_PCPS\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 68 of file `gps_l1_ca_pcps_acquisition.h`.

#### 10.184.2.2 init()

```
void GpsL1CaPcpsAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.184.2.3 mag()

```
signed int GpsL1CaPcpsAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

#### 10.184.2.4 reset()

```
void GpsL1CaPcpsAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.184.2.5 set\_channel()

```
void GpsL1CaPcpsAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 93 of file `gps_l1_ca_pcps_acquisition.h`.

#### 10.184.2.6 set\_channel\_fsm()

```
void GpsL1CaPcpsAcquisition::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 102 of file `gps_l1_ca_pcps_acquisition.h`.

#### 10.184.2.7 set\_doppler\_center()

```
void GpsL1CaPcpsAcquisition::set_doppler_center (
    int doppler_center ) [override]
```

Set Doppler center for the grid search.

**10.184.2.8 set\_doppler\_max()**

```
void GpsL1CaPcpsAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

**10.184.2.9 set\_doppler\_step()**

```
void GpsL1CaPcpsAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

**10.184.2.10 set\_gnss\_synchro()**

```
void GpsL1CaPcpsAcquisition::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](#).

**10.184.2.11 set\_local\_code()**

```
void GpsL1CaPcpsAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local code for GPS L1/CA PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

**10.184.2.12 set\_resampler\_latency()**

```
void GpsL1CaPcpsAcquisition::set_resampler_latency (
    uint32_t latency_samples ) [override], [virtual]
```

Sets the resampler latency to account it in the acquisition code delay estimation.

Implements [AcquisitionInterface](#).

**10.184.2.13 set\_state()**

```
void GpsL1CaPcpsAcquisition::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

**10.184.2.14 set\_threshold()**

```
void GpsL1CaPcpsAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

**10.184.2.15 stop\_acquisition()**

```
void GpsL1CaPcpsAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

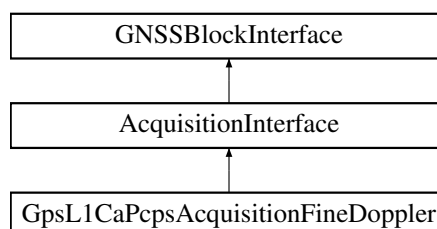
- [gps\\_l1\\_ca\\_pcps\\_acquisition.h](#)

**10.185 GpsL1CaPcpsAcquisitionFineDoppler Class Reference**

This class Adapts a PCPS acquisition block with fine Doppler estimation to an [AcquisitionInterface](#) for GPS L1 C/A signals.

```
#include <gps_l1_ca_pcps_acquisition_fine_doppler.h>
```

Inheritance diagram for GpsL1CaPcpsAcquisitionFineDoppler:



## Public Member Functions

- **GpsL1CaPcpsAcquisitionFineDoppler** (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\_PCPS\_Acquisition\_Fine\_Doppler".*
- size\_t **item\_size** () override
- void **connect** (gnss\_shared\_ptr< gr::top\_block > top\_block) override
- void **disconnect** (gnss\_shared\_ptr< gr::top\_block > top\_block) override
- gnss\_shared\_ptr< gr::basic\_block > **get\_left\_block** () override
- gnss\_shared\_ptr< gr::basic\_block > **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\_threshold** (float threshold) override
  - Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override
  - Set maximum Doppler off grid search.*
- void **set\_doppler\_step** (unsigned int doppler\_step) override
  - Set Doppler steps for the grid search.*
- void **init** () override
  - Initializes acquisition algorithm.*
- void **set\_local\_code** () override
- signed int **mag** () override
  - Returns the maximum peak of grid search.*
- void **reset** () override
  - Restart acquisition algorithm.*
- void **set\_state** (int state) override
  - If state = 1, it forces the block to start acquiring from the first sample.*
- void **stop\_acquisition** () override
  - Stop running acquisition.*
- void **set\_resampler\_latency** (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override

### 10.185.1 Detailed Description

This class Adapts a PCPS acquisition block with fine Doppler estimation to an [AcquisitionInterface](#) for GPS L1 C/A signals.

Definition at line 44 of file `gps_l1_ca_pcps_acquisition_fine_doppler.h`.

### 10.185.2 Member Function Documentation

#### 10.185.2.1 implementation()

```
std::string GpsL1CaPcpsAcquisitionFineDoppler::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L1\_CA\_PCPS\_Acquisition\_Fine\_Doppler".

Implements [GNSSBlockInterface](#).

Definition at line 62 of file `gps_l1_ca_pcps_acquisition_fine_doppler.h`.

#### 10.185.2.2 init()

```
void GpsL1CaPcpsAcquisitionFineDoppler::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.185.2.3 mag()

```
signed int GpsL1CaPcpsAcquisitionFineDoppler::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

#### 10.185.2.4 reset()

```
void GpsL1CaPcpsAcquisitionFineDoppler::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.185.2.5 set\_channel()

```
void GpsL1CaPcpsAcquisitionFineDoppler::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 87 of file `gps_l1_ca_pcps_acquisition_fine_doppler.h`.

#### 10.185.2.6 set\_channel\_fsm()

```
void GpsL1CaPcpsAcquisitionFineDoppler::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 96 of file `gps_l1_ca_pcps_acquisition_fine_doppler.h`.

#### 10.185.2.7 set\_doppler\_max()

```
void GpsL1CaPcpsAcquisitionFineDoppler::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.185.2.8 set\_doppler\_step()

```
void GpsL1CaPcpsAcquisitionFineDoppler::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.185.2.9 set\_gnss\_synchro()

```
void GpsL1CaPcpsAcquisitionFineDoppler::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](#).

### 10.185.2.10 `set_state()`

```
void GpsL1CaPcpsAcquisitionFineDoppler::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

### 10.185.2.11 `set_threshold()`

```
void GpsL1CaPcpsAcquisitionFineDoppler::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

### 10.185.2.12 `stop_acquisition()`

```
void GpsL1CaPcpsAcquisitionFineDoppler::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

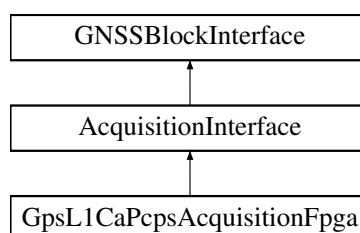
- [gps\\_l1\\_ca\\_pcps\\_acquisition\\_fine\\_doppler.h](#)

## 10.186 GpsL1CaPcpsAcquisitionFpga Class Reference

This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for GPS L1 C/A signals.

```
#include <gps_l1_ca_pcps_acquisition_fpga.h>
```

Inheritance diagram for GpsL1CaPcpsAcquisitionFpga:



## Public Member Functions

- [GpsL1CaPcpsAcquisitionFpga](#) (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)  
*Constructor.*
- [~GpsL1CaPcpsAcquisitionFpga](#) ()=default  
*Destructor.*
- std::string [role](#) () override  
*Role.*
- std::string [implementation](#) () override  
*Returns "GPS\_L1\_CA\_PCPS\_Acquisition\_Fpga".*
- size\_t [item\\_size](#) () override  
*Returns size of lv\_16sc\_t.*
- void [connect](#) (gr::top\_block\_sptr top\_block) override  
*Connect.*
- void [disconnect](#) (gr::top\_block\_sptr top\_block) override  
*Disconnect.*
- gr::basic\_block\_sptr [get\\_left\\_block](#) () override  
*Get left block.*
- gr::basic\_block\_sptr [get\\_right\\_block](#) () override  
*Get right block.*
- 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\\_threshold](#) (float threshold) override  
*Set statistics threshold of PCPS algorithm.*
- void [set\\_doppler\\_max](#) (unsigned int doppler\_max) override  
*Set maximum Doppler off grid search.*
- void [set\\_doppler\\_step](#) (unsigned int doppler\_step) override  
*Set Doppler steps for the grid search.*
- void [set\\_doppler\\_center](#) (int doppler\_center) override  
*Set Doppler center for the grid search.*
- void [init](#) () override  
*Initializes acquisition algorithm.*
- void [set\\_local\\_code](#) () override  
*Sets local code for GPS L1/CA PCPS acquisition algorithm.*
- signed int [mag](#) () override  
*Returns the maximum peak of grid search.*
- void [reset](#) () override  
*Restart acquisition algorithm.*
- void [set\\_state](#) (int state) override  
*If state = 1, it forces the block to start acquiring from the first sample.*
- void [stop\\_acquisition](#) () override  
*Stop running acquisition.*
- void [set\\_resampler\\_latency](#) (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override  
*Set Resampler Latency.*

### 10.186.1 Detailed Description

This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for GPS L1 C/A signals.

Definition at line 44 of file `gps_l1_ca_pcps_acquisition_fpga.h`.

### 10.186.2 Constructor & Destructor Documentation

#### 10.186.2.1 GpsL1CaPcpsAcquisitionFpga()

```
GpsL1CaPcpsAcquisitionFpga::GpsL1CaPcpsAcquisitionFpga (
    const ConfigurationInterface * configuration,
    const std::string & role,
    unsigned int in_streams,
    unsigned int out_streams )
```

Constructor.

#### 10.186.2.2 ~GpsL1CaPcpsAcquisitionFpga()

```
GpsL1CaPcpsAcquisitionFpga::~GpsL1CaPcpsAcquisitionFpga ( ) [default]
```

Destructor.

### 10.186.3 Member Function Documentation

#### 10.186.3.1 connect()

```
void GpsL1CaPcpsAcquisitionFpga::connect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Connect.

Implements [GNSSBlockInterface](#).

### 10.186.3.2 disconnect()

```
void GpsL1CaPcpsAcquisitionFpga::disconnect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Disconnect.

Implements [GNSSBlockInterface](#).

### 10.186.3.3 get\_left\_block()

```
gr::basic_block_sptr GpsL1CaPcpsAcquisitionFpga::get_left_block ( ) [override], [virtual]
```

Get left block.

Implements [GNSSBlockInterface](#).

### 10.186.3.4 get\_right\_block()

```
gr::basic_block_sptr GpsL1CaPcpsAcquisitionFpga::get_right_block ( ) [override], [virtual]
```

Get right block.

Implements [GNSSBlockInterface](#).

### 10.186.3.5 implementation()

```
std::string GpsL1CaPcpsAcquisitionFpga::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L1\_CA\_PCPS\_Acquisition\_Fpga".

Implements [GNSSBlockInterface](#).

Definition at line 71 of file `gps_l1_ca_pcps_acquisition_fpga.h`.

### 10.186.3.6 init()

```
void GpsL1CaPcpsAcquisitionFpga::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.186.3.7 item\_size()

```
size_t GpsL1CaPcpsAcquisitionFpga::item_size ( ) [inline], [override], [virtual]
```

Returns size of lv\_16sc\_t.

Implements [GNSSBlockInterface](#).

Definition at line 79 of file gps\_l1\_ca\_pcps\_acquisition\_fpga.h.

### 10.186.3.8 mag()

```
signed int GpsL1CaPcpsAcquisitionFpga::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

### 10.186.3.9 reset()

```
void GpsL1CaPcpsAcquisitionFpga::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.186.3.10 role()

```
std::string GpsL1CaPcpsAcquisitionFpga::role ( ) [inline], [override], [virtual]
```

Role.

Implements [GNSSBlockInterface](#).

Definition at line 63 of file gps\_l1\_ca\_pcps\_acquisition\_fpga.h.

### 10.186.3.11 set\_channel()

```
void GpsL1CaPcpsAcquisitionFpga::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 114 of file gps\_l1\_ca\_pcps\_acquisition\_fpga.h.

**10.186.3.12 set\_channel\_fsm()**

```
void GpsL1CaPcpsAcquisitionFpga::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 123 of file `gps_l1_ca_pcps_acquisition_fpga.h`.

**10.186.3.13 set\_doppler\_center()**

```
void GpsL1CaPcpsAcquisitionFpga::set_doppler_center (
    int doppler_center ) [override]
```

Set Doppler center for the grid search.

**10.186.3.14 set\_doppler\_max()**

```
void GpsL1CaPcpsAcquisitionFpga::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

**10.186.3.15 set\_doppler\_step()**

```
void GpsL1CaPcpsAcquisitionFpga::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

**10.186.3.16 set\_gnss\_synchro()**

```
void GpsL1CaPcpsAcquisitionFpga::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](#).

#### 10.186.3.17 `set_local_code()`

```
void GpsL1CaPcpsAcquisitionFpga::set_local_code ( ) [override], [virtual]
```

Sets local code for GPS L1/CA PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.186.3.18 `set_resampler_latency()`

```
void GpsL1CaPcpsAcquisitionFpga::set_resampler_latency (
    uint32_t latency_samples __attribute__((unused)) ) [inline], [override]
```

Set Resampler Latency.

Definition at line 182 of file `gps_l1_ca_pcps_acquisition_fpga.h`.

#### 10.186.3.19 `set_state()`

```
void GpsL1CaPcpsAcquisitionFpga::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

#### 10.186.3.20 `set_threshold()`

```
void GpsL1CaPcpsAcquisitionFpga::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

#### 10.186.3.21 `stop_acquisition()`

```
void GpsL1CaPcpsAcquisitionFpga::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

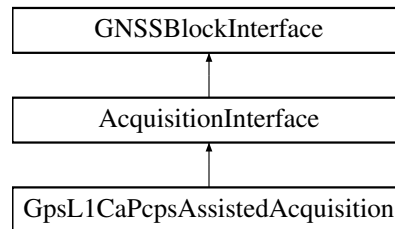
- [gps\\_l1\\_ca\\_pcps\\_acquisition\\_fpga.h](#)

## 10.187 GpsL1CaPcpsAssistedAcquisition Class Reference

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

```
#include <gps_l1_ca_pcps_assisted_acquisition.h>
```

Inheritance diagram for GpsL1CaPcpsAssistedAcquisition:



### Public Member Functions

- **GpsL1CaPcpsAssistedAcquisition** (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\_PCPS\_Assisted\_Acquisition".*
- 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\_threshold** (float threshold) override
  - Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override
  - Set maximum Doppler off grid search.*
- void **set\_doppler\_step** (unsigned int doppler\_step) override
  - Set Doppler steps for the grid search.*
- void **init** () override
  - Initializes acquisition algorithm.*
- void **set\_local\_code** () override
- signed int **mag** () override
  - Returns the maximum peak of grid search.*
- void **reset** () override
  - Restart acquisition algorithm.*
- void **set\_state** (int state \_\_attribute\_\_((unused))) override
- void **stop\_acquisition** () override
  - Stop running acquisition.*
- void **set\_resampler\_latency** (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override

### 10.187.1 Detailed Description

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

Definition at line 42 of file `gps_l1_ca_pcps_assisted_acquisition.h`.

### 10.187.2 Member Function Documentation

#### 10.187.2.1 `implementation()`

```
std::string GpsL1CaPcpsAssistedAcquisition::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L1\_CA\_PCPS\_Assisted\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 61 of file `gps_l1_ca_pcps_assisted_acquisition.h`.

#### 10.187.2.2 `init()`

```
void GpsL1CaPcpsAssistedAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.187.2.3 `mag()`

```
signed int GpsL1CaPcpsAssistedAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

#### 10.187.2.4 `reset()`

```
void GpsL1CaPcpsAssistedAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.187.2.5 set\_channel()

```
void GpsL1CaPcpsAssistedAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 86 of file `gps_l1_ca_pcps_assisted_acquisition.h`.

### 10.187.2.6 set\_channel\_fsm()

```
void GpsL1CaPcpsAssistedAcquisition::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 `gps_l1_ca_pcps_assisted_acquisition.h`.

### 10.187.2.7 set\_doppler\_max()

```
void GpsL1CaPcpsAssistedAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

### 10.187.2.8 set\_doppler\_step()

```
void GpsL1CaPcpsAssistedAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

**10.187.2.9 set\_gnss\_synchro()**

```
void GpsL1CaPcpsAssistedAcquisition::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](#).

**10.187.2.10 set\_threshold()**

```
void GpsL1CaPcpsAssistedAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

**10.187.2.11 stop\_acquisition()**

```
void GpsL1CaPcpsAssistedAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

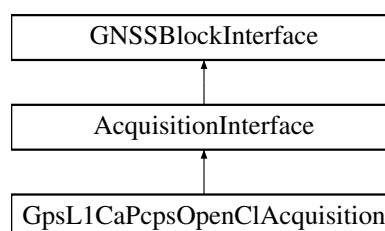
- [gps\\_l1\\_ca\\_pcps\\_assisted\\_acquisition.h](#)

**10.188 GpsL1CaPcpsOpenCLAcquisition Class Reference**

This class adapts an OpenCL PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

```
#include <gps_l1_ca_pcps_openc1_acquisition.h>
```

Inheritance diagram for GpsL1CaPcpsOpenCLAcquisition:



## Public Member Functions

- **GpsL1CaPcpsOpenClAcquisition** (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\_PCPS\_OpenCl\_Acquisition".*
- 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\_threshold** (float threshold) override
  - Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override
  - Set maximum Doppler off grid search.*
- void **set\_doppler\_step** (unsigned int doppler\_step) override
  - Set Doppler steps for the grid search.*
- void **init** () override
  - Initializes acquisition algorithm.*
- void **set\_local\_code** () override
  - Sets local code for GPS L1/CA PCPS acquisition algorithm.*
- signed int **mag** () override
  - Returns the maximum peak of grid search.*
- void **reset** () override
  - Restart acquisition algorithm.*
- void **set\_state** (int state \_\_attribute\_\_((unused))) override
- void **stop\_acquisition** () override
  - Stop running acquisition.*
- void **set\_resampler\_latency** (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override
- bool **opencl\_ready** () const

### 10.188.1 Detailed Description

This class adapts an OpenCL PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

Definition at line 41 of file `gps_l1_ca_pcps_openc1_acquisition.h`.

### 10.188.2 Member Function Documentation

#### 10.188.2.1 implementation()

```
std::string GpsL1CaPcpsOpenClAcquisition::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L1\_CA\_PCPS\_OpenCl\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 59 of file `gps_l1_ca_pcps_openc1_acquisition.h`.

#### 10.188.2.2 init()

```
void GpsL1CaPcpsOpenClAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.188.2.3 mag()

```
signed int GpsL1CaPcpsOpenClAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

#### 10.188.2.4 reset()

```
void GpsL1CaPcpsOpenClAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.188.2.5 set\_channel()

```
void GpsL1CaPcpsOpenClAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 84 of file `gps_l1_ca_pcps_openc1_acquisition.h`.

#### 10.188.2.6 set\_channel\_fsm()

```
void GpsL1CaPcpsOpenClAcquisition::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 93 of file `gps_l1_ca_pcps_openc1_acquisition.h`.

#### 10.188.2.7 set\_doppler\_max()

```
void GpsL1CaPcpsOpenClAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.188.2.8 set\_doppler\_step()

```
void GpsL1CaPcpsOpenClAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.188.2.9 set\_gnss\_synchro()

```
void GpsL1CaPcpsOpenClAcquisition::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](#).

**10.188.2.10 set\_local\_code()**

```
void GpsL1CaPcpsOpenClAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local code for GPS L1/CA PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

**10.188.2.11 set\_threshold()**

```
void GpsL1CaPcpsOpenClAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

**10.188.2.12 stop\_acquisition()**

```
void GpsL1CaPcpsOpenClAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

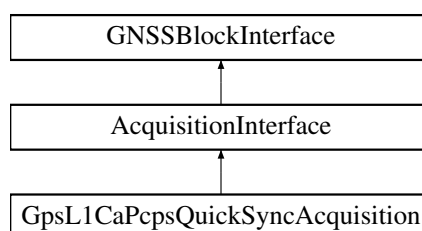
- [gps\\_l1\\_ca\\_pcps\\_openc1\\_acquisition.h](#)

**10.189 GpsL1CaPcpsQuickSyncAcquisition Class Reference**

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

```
#include <gps_l1_ca_pcps_quicksync_acquisition.h>
```

Inheritance diagram for GpsL1CaPcpsQuickSyncAcquisition:



## Public Member Functions

- **GpsL1CaPcpsQuickSyncAcquisition** (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\_PCPS\_QuickSync\_Acquisition".*
- 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\_threshold** (float threshold) override
  - Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override
  - Set maximum Doppler off grid search.*
- void **set\_doppler\_step** (unsigned int doppler\_step) override
  - Set Doppler steps for the grid search.*
- void **init** () override
  - Initializes acquisition algorithm.*
- void **set\_local\_code** () override
  - Sets local code for GPS L1/CA PCPS acquisition algorithm.*
- signed int **mag** () override
  - Returns the maximum peak of grid search.*
- void **reset** () override
  - Restart acquisition algorithm.*
- void **set\_state** (int state) override
  - If state = 1, it forces the block to start acquiring from the first sample.*
- void **stop\_acquisition** () override
  - Stop running acquisition.*
- void **set\_resampler\_latency** (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override

### 10.189.1 Detailed Description

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

Definition at line 43 of file `gps_l1_ca_pcps_quicksync_acquisition.h`.

### 10.189.2 Member Function Documentation

#### 10.189.2.1 implementation()

```
std::string GpsL1CaPcpsQuickSyncAcquisition::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L1\_CA\_PCPS\_QuickSync\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 62 of file `gps_l1_ca_pcps_quicksync_acquisition.h`.

#### 10.189.2.2 init()

```
void GpsL1CaPcpsQuickSyncAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.189.2.3 mag()

```
signed int GpsL1CaPcpsQuickSyncAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

#### 10.189.2.4 reset()

```
void GpsL1CaPcpsQuickSyncAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.189.2.5 set\_channel()

```
void GpsL1CaPcpsQuickSyncAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 87 of file `gps_l1_ca_pcps_quicksync_acquisition.h`.

#### 10.189.2.6 set\_channel\_fsm()

```
void GpsL1CaPcpsQuickSyncAcquisition::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 96 of file `gps_l1_ca_pcps_quicksync_acquisition.h`.

#### 10.189.2.7 set\_doppler\_max()

```
void GpsL1CaPcpsQuickSyncAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.189.2.8 set\_doppler\_step()

```
void GpsL1CaPcpsQuickSyncAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.189.2.9 set\_gnss\_synchro()

```
void GpsL1CaPcpsQuickSyncAcquisition::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](#).

#### 10.189.2.10 `set_local_code()`

```
void GpsL1CaPcpsQuickSyncAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local code for GPS L1/CA PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.189.2.11 `set_state()`

```
void GpsL1CaPcpsQuickSyncAcquisition::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

#### 10.189.2.12 `set_threshold()`

```
void GpsL1CaPcpsQuickSyncAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

#### 10.189.2.13 `stop_acquisition()`

```
void GpsL1CaPcpsQuickSyncAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

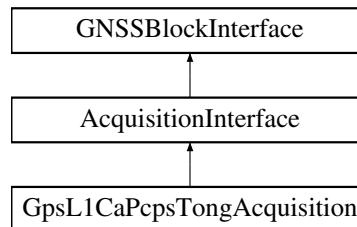
- [gps\\_l1\\_ca\\_pcps\\_quicksync\\_acquisition.h](#)

## 10.190 GpsL1CaPcpsTongAcquisition Class Reference

This class adapts a PCPS Tong acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

```
#include <gps_l1_ca_pcps_tong_acquisition.h>
```

Inheritance diagram for GpsL1CaPcpsTongAcquisition:



### Public Member Functions

- **GpsL1CaPcpsTongAcquisition** (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\_PCPS\_Tong\_Acquisition".*
- 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\_threshold** (float threshold) override
  - Set statistics threshold of TONG algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override
  - Set maximum Doppler off grid search.*
- void **set\_doppler\_step** (unsigned int doppler\_step) override
  - Set Doppler steps for the grid search.*
- void **init** () override
  - Initializes acquisition algorithm.*
- void **set\_local\_code** () override
  - Sets local code for GPS L1/CA TONG acquisition algorithm.*
- signed int **mag** () override
  - Returns the maximum peak of grid search.*
- void **reset** () override
  - Restart acquisition algorithm.*
- void **set\_state** (int state) override
  - If state = 1, it forces the block to start acquiring from the first sample.*
- void **stop\_acquisition** () override
  - Stop running acquisition.*
- void **set\_resampler\_latency** (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override

### 10.190.1 Detailed Description

This class adapts a PCPS Tong acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

Definition at line 42 of file `gps_l1_ca_pcps_tong_acquisition.h`.

### 10.190.2 Member Function Documentation

#### 10.190.2.1 `implementation()`

```
std::string GpsL1CaPcpsTongAcquisition::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L1\_CA\_PCPS\_Tong\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 60 of file `gps_l1_ca_pcps_tong_acquisition.h`.

#### 10.190.2.2 `init()`

```
void GpsL1CaPcpsTongAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.190.2.3 `mag()`

```
signed int GpsL1CaPcpsTongAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

#### 10.190.2.4 `reset()`

```
void GpsL1CaPcpsTongAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.190.2.5 set\_channel()

```
void GpsL1CaPcpsTongAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 85 of file `gps_l1_ca_pcps_tong_acquisition.h`.

#### 10.190.2.6 set\_channel\_fsm()

```
void GpsL1CaPcpsTongAcquisition::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 94 of file `gps_l1_ca_pcps_tong_acquisition.h`.

#### 10.190.2.7 set\_doppler\_max()

```
void GpsL1CaPcpsTongAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.190.2.8 set\_doppler\_step()

```
void GpsL1CaPcpsTongAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.190.2.9 `set_gnss_synchro()`

```
void GpsL1CaPcpsTongAcquisition::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](#).

#### 10.190.2.10 `set_local_code()`

```
void GpsL1CaPcpsTongAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local code for GPS L1/CA TONG acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.190.2.11 `set_state()`

```
void GpsL1CaPcpsTongAcquisition::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

#### 10.190.2.12 `set_threshold()`

```
void GpsL1CaPcpsTongAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of TONG algorithm.

Implements [AcquisitionInterface](#).

#### 10.190.2.13 `stop_acquisition()`

```
void GpsL1CaPcpsTongAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

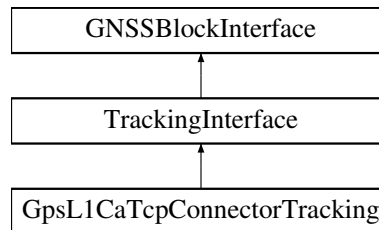
- [gps\\_l1\\_ca\\_pcps\\_tong\\_acquisition.h](#)

## 10.191 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.*

### 10.191.1 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`.

### 10.191.2 Member Function Documentation

#### 10.191.2.1 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`.

#### 10.191.2.2 set\_channel()

```
void GpsL1CaTcpConnectorTracking::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.191.2.3 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](#).

#### 10.191.2.4 stop\_tracking()

```
void GpsL1CaTcpConnectorTracking::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

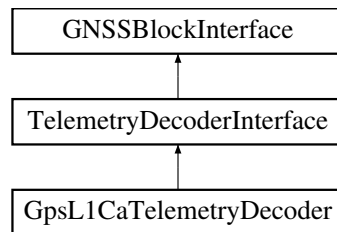
- [gps\\_l1\\_ca\\_tcp\\_connector\\_tracking.h](#)

## 10.192 GpsL1CaTelemetryDecoder Class Reference

This class implements a NAV data decoder for GPS L1 C/A.

```
#include <gps_l1_ca_telemetry_decoder.h>
```

Inheritance diagram for GpsL1CaTelemetryDecoder:



### Public Member Functions

- **GpsL1CaTelemetryDecoder** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- 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
  - Returns "GPS\_L1\_CA\_Telemetry\_Decoder".*
- void **set\_channel** (int channel) override
- void **reset** () override
- size\_t **item\_size** () override

### 10.192.1 Detailed Description

This class implements a NAV data decoder for GPS L1 C/A.

Definition at line 45 of file `gps_l1_ca_telemetry_decoder.h`.

### 10.192.2 Member Function Documentation

#### 10.192.2.1 implementation()

```
std::string GpsL1CaTelemetryDecoder::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L1\_CA\_Telemetry\_Decoder".

Implements [GNSSBlockInterface](#).

Definition at line 69 of file `gps_l1_ca_telemetry_decoder.h`.

The documentation for this class was generated from the following file:

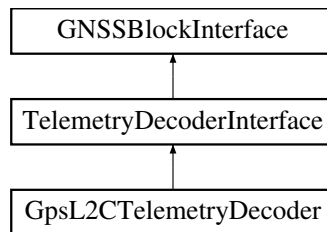
- [gps\\_l1\\_ca\\_telemetry\\_decoder.h](#)

## 10.193 GpsL2CTelemetryDecoder Class Reference

This class implements a NAV data decoder for GPS L2 M.

```
#include <gps_l2c_telemetry_decoder.h>
```

Inheritance diagram for GpsL2CTelemetryDecoder:



### Public Member Functions

- **GpsL2CTelemetryDecoder** (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\_L2C\_Telemetry\_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
- void **set\_channel** (int channel) override
- void **reset** () override
- size\_t **item\_size** () override

### 10.193.1 Detailed Description

This class implements a NAV data decoder for GPS L2 M.

Definition at line 43 of file `gps_l2c_telemetry_decoder.h`.

### 10.193.2 Member Function Documentation

#### 10.193.2.1 implementation()

```
std::string GpsL2CTelemetryDecoder::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L2C\_Telemetry\_Decoder".

Implements [GNSSBlockInterface](#).

Definition at line 60 of file `gps_l2c_telemetry_decoder.h`.

The documentation for this class was generated from the following file:

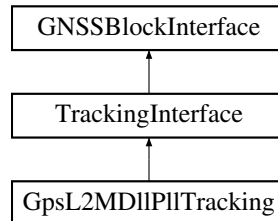
- [gps\\_l2c\\_telemetry\\_decoder.h](#)

## 10.194 GpsL2MDIIPITracking Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#include <gps_l2_m_dll_pll_tracking.h>
```

Inheritance diagram for GpsL2MDIIPITracking:



### Public Member Functions

- **GpsL2MDIIPITracking** (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\_L2\_M\_DLL\_PLL\_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.*

### 10.194.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 42 of file `gps_l2_m_dll_pll_tracking.h`.

### 10.194.2 Member Function Documentation

#### 10.194.2.1 implementation()

```
std::string GpsL2MD11P11Tracking::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L2\_M\_DLL\_PLL\_Tracking".

Implements [GNSSBlockInterface](#).

Definition at line 59 of file `gps_l2_m_dll_pll_tracking.h`.

#### 10.194.2.2 set\_channel()

```
void GpsL2MD11P11Tracking::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.194.2.3 set\_gnss\_synchro()

```
void GpsL2MD11P11Tracking::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](#).

#### 10.194.2.4 stop\_tracking()

```
void GpsL2MD11P11Tracking::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

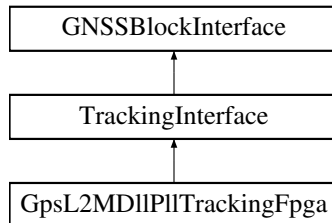
- [gps\\_l2\\_m\\_dll\\_pll\\_tracking.h](#)

## 10.195 GpsL2MDIIPITrackingFpga Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#include <gps_l2_m_dll_pll_tracking_fpga.h>
```

Inheritance diagram for GpsL2MDIIPITrackingFpga:



### Public Member Functions

- **GpsL2MDIIPITrackingFpga** (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\_L2\_M\_DLL\_PLL\_Tracking\_Fpga".*
- 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.*

### 10.195.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 44 of file `gps_l2_m_dll_pll_tracking_fpga.h`.

### 10.195.2 Member Function Documentation

#### 10.195.2.1 implementation()

```
std::string GpsL2MD11P11TrackingFpga::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L2\_M\_DLL\_PLL\_Tracking\_Fpga".

Implements [GNSSBlockInterface](#).

Definition at line 61 of file `gps_l2_m_dll_pll_tracking_fpga.h`.

#### 10.195.2.2 set\_channel()

```
void GpsL2MD11P11TrackingFpga::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

#### 10.195.2.3 set\_gnss\_synchro()

```
void GpsL2MD11P11TrackingFpga::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](#).

#### 10.195.2.4 stop\_tracking()

```
void GpsL2MD11P11TrackingFpga::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

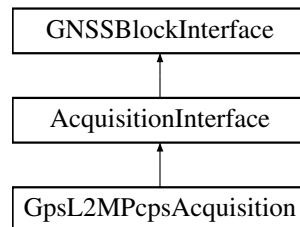
- [gps\\_l2\\_m\\_dll\\_pll\\_tracking\\_fpga.h](#)

## 10.196 GpsL2MPcpsAcquisition Class Reference

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L2 M signals.

```
#include <gps_l2_m_pcps_acquisition.h>
```

Inheritance diagram for GpsL2MPcpsAcquisition:



### Public Member Functions

- **GpsL2MPcpsAcquisition** (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\_L2\_M\_PCPS\_Acquisition".*
- 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\_threshold** (float threshold) override
  - Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override
  - Set maximum Doppler off grid search.*
- void **set\_doppler\_step** (unsigned int doppler\_step) override
  - Set Doppler steps for the grid search.*
- void **set\_doppler\_center** (int doppler\_center) override
  - Set Doppler center for the grid search.*
- void **init** () override
  - Initializes acquisition algorithm.*
- void **set\_local\_code** () override
  - Sets local code for GPS L2/M PCPS acquisition algorithm.*
- signed int **mag** () override
  - Returns the maximum peak of grid search.*
- void **reset** () override
  - Restart acquisition algorithm.*

- void [set\\_state](#) (int state) override  
*If state = 1, it forces the block to start acquiring from the first sample.*
- 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.*

### 10.196.1 Detailed Description

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L2 M signals.

Definition at line 44 of file `gps_l2_m_pcps_acquisition.h`.

### 10.196.2 Member Function Documentation

#### 10.196.2.1 implementation()

```
std::string GpsL2MPcpsAcquisition::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L2\_M\_PCPS\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 63 of file `gps_l2_m_pcps_acquisition.h`.

#### 10.196.2.2 init()

```
void GpsL2MPcpsAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.196.2.3 mag()

```
signed int GpsL2MPcpsAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

#### 10.196.2.4 reset()

```
void GpsL2MPcpsAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.196.2.5 set\_channel()

```
void GpsL2MPcpsAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 88 of file `gps_l2_m_pcps_acquisition.h`.

#### 10.196.2.6 set\_channel\_fsm()

```
void GpsL2MPcpsAcquisition::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 97 of file `gps_l2_m_pcps_acquisition.h`.

#### 10.196.2.7 set\_doppler\_center()

```
void GpsL2MPcpsAcquisition::set_doppler_center (
    int doppler_center ) [override]
```

Set Doppler center for the grid search.

#### 10.196.2.8 set\_doppler\_max()

```
void GpsL2MPcpsAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.196.2.9 set\_doppler\_step()

```
void GpsL2MPcpsAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.196.2.10 set\_gnss\_synchro()

```
void GpsL2MPcpsAcquisition::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](#).

#### 10.196.2.11 set\_local\_code()

```
void GpsL2MPcpsAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local code for GPS L2/M PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.196.2.12 set\_resampler\_latency()

```
void GpsL2MPcpsAcquisition::set_resampler_latency (
    uint32_t latency_samples ) [override], [virtual]
```

Sets the resampler latency to account it in the acquisition code delay estimation.

Implements [AcquisitionInterface](#).

#### 10.196.2.13 set\_state()

```
void GpsL2MPcpsAcquisition::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

#### 10.196.2.14 set\_threshold()

```
void GpsL2MPcpsAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

#### 10.196.2.15 stop\_acquisition()

```
void GpsL2MPcpsAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

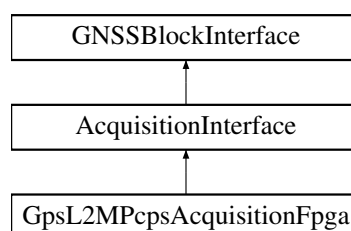
- [gps\\_l2\\_m\\_pcps\\_acquisition.h](#)

## 10.197 GpsL2MPcpsAcquisitionFpga Class Reference

This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for GPS L2 M signals.

```
#include <gps_l2_m_pcps_acquisition_fpga.h>
```

Inheritance diagram for GpsL2MPcpsAcquisitionFpga:



## Public Member Functions

- **GpsL2MPcpsAcquisitionFpga** (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\_L2\_M\_PCPS\_Acquisition\_Fpga".*
- 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\_threshold** (float threshold) override
  - Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override
  - Set maximum Doppler off grid search.*
- void **set\_doppler\_step** (unsigned int doppler\_step) override
  - Set Doppler steps for the grid search.*
- void **init** () override
  - Initializes acquisition algorithm.*
- void **set\_local\_code** () override
  - Sets local code for GPS L2/M PCPS acquisition algorithm.*
- signed int **mag** () override
  - Returns the maximum peak of grid search.*
- void **reset** () override
  - Restart acquisition algorithm.*
- void **set\_state** (int state) override
  - If state = 1, it forces the block to start acquiring from the first sample.*
- void **stop\_acquisition** () override
  - Stop running acquisition.*
- void **set\_resampler\_latency** (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override

### 10.197.1 Detailed Description

This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for GPS L2 M signals.

Definition at line 44 of file `gps_l2_m_pcps_acquisition_fpga.h`.

### 10.197.2 Member Function Documentation

### 10.197.2.1 implementation()

```
std::string GpsL2MPcpsAcquisitionFpga::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L2\_M\_PCPS\_Acquisition\_Fpga".

Implements [GNSSBlockInterface](#).

Definition at line 63 of file `gps_l2_m_pcps_acquisition_fpga.h`.

### 10.197.2.2 init()

```
void GpsL2MPcpsAcquisitionFpga::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.197.2.3 mag()

```
signed int GpsL2MPcpsAcquisitionFpga::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

### 10.197.2.4 reset()

```
void GpsL2MPcpsAcquisitionFpga::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.197.2.5 set\_channel()

```
void GpsL2MPcpsAcquisitionFpga::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 88 of file `gps_l2_m_pcps_acquisition_fpga.h`.

#### 10.197.2.6 set\_channel\_fsm()

```
void GpsL2MPcpsAcquisitionFpga::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 97 of file `gps_l2_m_pcps_acquisition_fpga.h`.

#### 10.197.2.7 set\_doppler\_max()

```
void GpsL2MPcpsAcquisitionFpga::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.197.2.8 set\_doppler\_step()

```
void GpsL2MPcpsAcquisitionFpga::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.197.2.9 set\_gnss\_synchro()

```
void GpsL2MPcpsAcquisitionFpga::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](#).

#### 10.197.2.10 `set_local_code()`

```
void GpsL2MPcpsAcquisitionFpga::set_local_code ( ) [override], [virtual]
```

Sets local code for GPS L2/M PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.197.2.11 `set_state()`

```
void GpsL2MPcpsAcquisitionFpga::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

#### 10.197.2.12 `set_threshold()`

```
void GpsL2MPcpsAcquisitionFpga::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

#### 10.197.2.13 `stop_acquisition()`

```
void GpsL2MPcpsAcquisitionFpga::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

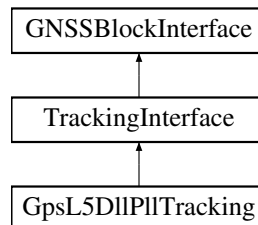
- [gps\\_l2\\_m\\_pcps\\_acquisition\\_fpga.h](#)

## 10.198 GpsL5DlPIITracking Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#include <gps_l5_dll_pll_tracking.h>
```

Inheritance diagram for GpsL5DlPIITracking:



### Public Member Functions

- **GpsL5DlPIITracking** (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\_L5\_DLL\_PLL\_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.*

### 10.198.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 41 of file `gps_l5_dll_pll_tracking.h`.

### 10.198.2 Member Function Documentation

### 10.198.2.1 implementation()

```
std::string GpsL5D11P11Tracking::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L5\_DLL\_PLL\_Tracking".

Implements [GNSSBlockInterface](#).

Definition at line 58 of file `gps_l5_dll_pll_tracking.h`.

### 10.198.2.2 set\_channel()

```
void GpsL5D11P11Tracking::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

### 10.198.2.3 set\_gnss\_synchro()

```
void GpsL5D11P11Tracking::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](#).

### 10.198.2.4 stop\_tracking()

```
void GpsL5D11P11Tracking::stop_tracking ( ) [override], [virtual]
```

Stop running tracking.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

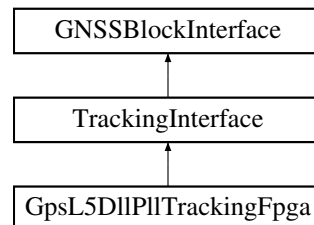
- [gps\\_l5\\_dll\\_pll\\_tracking.h](#)

## 10.199 GpsL5DIIPITrackingFpga Class Reference

This class implements a code DLL + carrier PLL tracking loop.

```
#include <gps_l5_dll_pll_tracking_fpga.h>
```

Inheritance diagram for GpsL5DIIPITrackingFpga:



### Public Member Functions

- [GpsL5DIIPITrackingFpga](#) (const [ConfigurationInterface](#) \*configuration, const std::string &[role](#), unsigned int in\_streams, unsigned int out\_streams)  
*Constructor.*
- virtual [~GpsL5DIIPITrackingFpga](#) ()  
*Destructor.*
- std::string [role](#) () override  
*Role.*
- std::string [implementation](#) () override  
*Returns "GPS\_L5\_DLL\_PLL\_Tracking\_Fpga".*
- size\_t [item\\_size](#) () override  
*Returns size of lv\_16sc\_t.*
- void [connect](#) (gr::top\_block\_sptr top\_block) override  
*Connect.*
- void [disconnect](#) (gr::top\_block\_sptr top\_block) override  
*Disconnect.*
- gr::basic\_block\_sptr [get\\_left\\_block](#) () override  
*Get left block.*
- gr::basic\_block\_sptr [get\\_right\\_block](#) () override  
*Get right block.*
- 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  
*Start the tracking process in the FPGA.*
- void [stop\\_tracking](#) () override  
*Stop the tracking process in the FPGA.*

### 10.199.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.

Definition at line 42 of file `gps_l5_dll_pll_tracking_fpga.h`.

## 10.199.2 Constructor & Destructor Documentation

### 10.199.2.1 GpsL5D11P11TrackingFpga()

```
GpsL5D11P11TrackingFpga::GpsL5D11P11TrackingFpga (
    const ConfigurationInterface * configuration,
    const std::string & role,
    unsigned int in_streams,
    unsigned int out_streams )
```

Constructor.

### 10.199.2.2 ~GpsL5D11P11TrackingFpga()

```
virtual GpsL5D11P11TrackingFpga::~~GpsL5D11P11TrackingFpga ( ) [virtual]
```

Destructor.

## 10.199.3 Member Function Documentation

### 10.199.3.1 connect()

```
void GpsL5D11P11TrackingFpga::connect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Connect.

Implements [GNSSBlockInterface](#).

### 10.199.3.2 disconnect()

```
void GpsL5D11P11TrackingFpga::disconnect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Disconnect.

Implements [GNSSBlockInterface](#).

### 10.199.3.3 `get_left_block()`

```
gr::basic_block_sptr GpsL5D11P11TrackingFpga::get_left_block ( ) [override], [virtual]
```

Get left block.

Implements [GNSSBlockInterface](#).

### 10.199.3.4 `get_right_block()`

```
gr::basic_block_sptr GpsL5D11P11TrackingFpga::get_right_block ( ) [override], [virtual]
```

Get right block.

Implements [GNSSBlockInterface](#).

### 10.199.3.5 `implementation()`

```
std::string GpsL5D11P11TrackingFpga::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L5\_DLL\_PLL\_Tracking\_Fpga".

Implements [GNSSBlockInterface](#).

Definition at line 70 of file `gps_l5_dll_pll_tracking_fpga.h`.

### 10.199.3.6 `item_size()`

```
size_t GpsL5D11P11TrackingFpga::item_size ( ) [inline], [override], [virtual]
```

Returns size of `lv_16sc_t`.

Implements [GNSSBlockInterface](#).

Definition at line 78 of file `gps_l5_dll_pll_tracking_fpga.h`.

### 10.199.3.7 `role()`

```
std::string GpsL5D11P11TrackingFpga::role ( ) [inline], [override], [virtual]
```

Role.

Implements [GNSSBlockInterface](#).

Definition at line 62 of file `gps_l5_dll_pll_tracking_fpga.h`.

### 10.199.3.8 set\_channel()

```
void GpsL5D11P11TrackingFpga::set_channel (
    unsigned int channel ) [override], [virtual]
```

Set tracking channel unique ID.

Implements [TrackingInterface](#).

### 10.199.3.9 set\_gnss\_synchro()

```
void GpsL5D11P11TrackingFpga::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](#).

### 10.199.3.10 start\_tracking()

```
void GpsL5D11P11TrackingFpga::start_tracking ( ) [override], [virtual]
```

Start the tracking process in the FPGA.

Implements [TrackingInterface](#).

### 10.199.3.11 stop\_tracking()

```
void GpsL5D11P11TrackingFpga::stop_tracking ( ) [override], [virtual]
```

Stop the tracking process in the FPGA.

Implements [TrackingInterface](#).

The documentation for this class was generated from the following file:

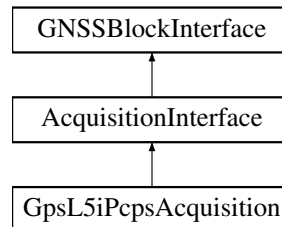
- [gps\\_l5\\_dll\\_pll\\_tracking\\_fpga.h](#)

## 10.200 GpsL5iPcpsAcquisition Class Reference

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L5i signals.

```
#include <gps_l5i_pcps_acquisition.h>
```

Inheritance diagram for GpsL5iPcpsAcquisition:



### Public Member Functions

- **GpsL5iPcpsAcquisition** (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\_L5i\_PCPS\_Acquisition".*
- 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\_threshold** (float threshold) override  
*Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (unsigned int doppler\_max) override  
*Set maximum Doppler off grid search.*
- void **set\_doppler\_step** (unsigned int doppler\_step) override  
*Set Doppler steps for the grid search.*
- void **set\_doppler\_center** (int doppler\_center) override  
*Set Doppler center for the grid search.*
- void **init** () override  
*Initializes acquisition algorithm.*
- void **set\_local\_code** () override  
*Sets local code for GPS L2/M PCPS acquisition algorithm.*
- signed int **mag** () override  
*Returns the maximum peak of grid search.*
- void **reset** () override  
*Restart acquisition algorithm.*

- void [set\\_state](#) (int state) override  
*If state = 1, it forces the block to start acquiring from the first sample.*
- 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.*

### 10.200.1 Detailed Description

This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L5i signals.

Definition at line 44 of file `gps_l5i_pcps_acquisition.h`.

### 10.200.2 Member Function Documentation

#### 10.200.2.1 implementation()

```
std::string GpsL5iPcpsAcquisition::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L5i\_PCPS\_Acquisition".

Implements [GNSSBlockInterface](#).

Definition at line 63 of file `gps_l5i_pcps_acquisition.h`.

#### 10.200.2.2 init()

```
void GpsL5iPcpsAcquisition::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.200.2.3 mag()

```
signed int GpsL5iPcpsAcquisition::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

#### 10.200.2.4 reset()

```
void GpsL5iPcpsAcquisition::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

#### 10.200.2.5 set\_channel()

```
void GpsL5iPcpsAcquisition::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 88 of file `gps_l5i_pcps_acquisition.h`.

#### 10.200.2.6 set\_channel\_fsm()

```
void GpsL5iPcpsAcquisition::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 97 of file `gps_l5i_pcps_acquisition.h`.

#### 10.200.2.7 set\_doppler\_center()

```
void GpsL5iPcpsAcquisition::set_doppler_center (
    int doppler_center ) [override]
```

Set Doppler center for the grid search.

#### 10.200.2.8 set\_doppler\_max()

```
void GpsL5iPcpsAcquisition::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

### 10.200.2.9 set\_doppler\_step()

```
void GpsL5iPcpsAcquisition::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

### 10.200.2.10 set\_gnss\_synchro()

```
void GpsL5iPcpsAcquisition::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](#).

### 10.200.2.11 set\_local\_code()

```
void GpsL5iPcpsAcquisition::set_local_code ( ) [override], [virtual]
```

Sets local code for GPS L2/M PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.200.2.12 set\_resampler\_latency()

```
void GpsL5iPcpsAcquisition::set_resampler_latency (
    uint32_t latency_samples ) [override], [virtual]
```

Sets the resampler latency to account it in the acquisition code delay estimation.

Implements [AcquisitionInterface](#).

### 10.200.2.13 set\_state()

```
void GpsL5iPcpsAcquisition::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

#### 10.200.2.14 `set_threshold()`

```
void GpsL5iPcpsAcquisition::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

#### 10.200.2.15 `stop_acquisition()`

```
void GpsL5iPcpsAcquisition::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

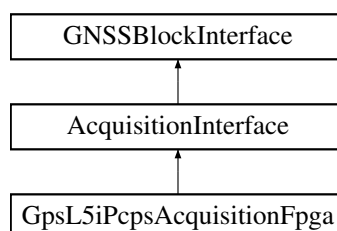
- [gps\\_l5i\\_pcps\\_acquisition.h](#)

## 10.201 `GpsL5iPcpsAcquisitionFpga` Class Reference

This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for GPS L5i signals.

```
#include <gps_l5i_pcps_acquisition_fpga.h>
```

Inheritance diagram for `GpsL5iPcpsAcquisitionFpga`:



## Public Member Functions

- [GpsL5iPcpsAcquisitionFpga](#) (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)  
*Constructor.*
- [~GpsL5iPcpsAcquisitionFpga](#) ()=default  
*Destructor.*
- std::string [role](#) () override  
*Role.*
- std::string [implementation](#) () override  
*Returns "GPS\_L5i\_PCPS\_Acquisition\_Fpga".*
- size\_t [item\\_size](#) () override  
*Returns size of lv\_16sc\_t.*
- void [connect](#) (gr::top\_block\_sptr top\_block) override  
*Connect.*
- void [disconnect](#) (gr::top\_block\_sptr top\_block) override  
*Disconnect.*
- gr::basic\_block\_sptr [get\\_left\\_block](#) () override  
*Get left block.*
- gr::basic\_block\_sptr [get\\_right\\_block](#) () override  
*Get right block.*
- 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\\_threshold](#) (float threshold) override  
*Set statistics threshold of PCPS algorithm.*
- void [set\\_doppler\\_max](#) (unsigned int doppler\_max) override  
*Set maximum Doppler off grid search.*
- void [set\\_doppler\\_step](#) (unsigned int doppler\_step) override  
*Set Doppler steps for the grid search.*
- void [set\\_doppler\\_center](#) (int doppler\_center) override  
*Set Doppler center for the grid search.*
- void [init](#) () override  
*Initializes acquisition algorithm.*
- void [set\\_local\\_code](#) () override  
*Sets local code for GPS L5 PCPS acquisition algorithm.*
- signed int [mag](#) () override  
*Returns the maximum peak of grid search.*
- void [reset](#) () override  
*Restart acquisition algorithm.*
- void [set\\_state](#) (int state) override  
*If state = 1, it forces the block to start acquiring from the first sample.*
- void [stop\\_acquisition](#) () override  
*Stop running acquisition.*
- void [set\\_resampler\\_latency](#) (uint32\_t latency\_samples \_\_attribute\_\_((unused))) override  
*Set resampler latency.*

### 10.201.1 Detailed Description

This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for GPS L5i signals.

Definition at line 43 of file `gps_l5i_pcps_acquisition_fpga.h`.

### 10.201.2 Constructor & Destructor Documentation

#### 10.201.2.1 GpsL5iPcpsAcquisitionFpga()

```
GpsL5iPcpsAcquisitionFpga::GpsL5iPcpsAcquisitionFpga (
    const ConfigurationInterface * configuration,
    const std::string & role,
    unsigned int in_streams,
    unsigned int out_streams )
```

Constructor.

#### 10.201.2.2 ~GpsL5iPcpsAcquisitionFpga()

```
GpsL5iPcpsAcquisitionFpga::~GpsL5iPcpsAcquisitionFpga ( ) [default]
```

Destructor.

### 10.201.3 Member Function Documentation

#### 10.201.3.1 connect()

```
void GpsL5iPcpsAcquisitionFpga::connect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Connect.

Implements [GNSSBlockInterface](#).

### 10.201.3.2 disconnect()

```
void GpsL5iPcpsAcquisitionFpga::disconnect (
    gr::top_block_sptr top_block ) [override], [virtual]
```

Disconnect.

Implements [GNSSBlockInterface](#).

### 10.201.3.3 get\_left\_block()

```
gr::basic_block_sptr GpsL5iPcpsAcquisitionFpga::get_left_block ( ) [override], [virtual]
```

Get left block.

Implements [GNSSBlockInterface](#).

### 10.201.3.4 get\_right\_block()

```
gr::basic_block_sptr GpsL5iPcpsAcquisitionFpga::get_right_block ( ) [override], [virtual]
```

Get right block.

Implements [GNSSBlockInterface](#).

### 10.201.3.5 implementation()

```
std::string GpsL5iPcpsAcquisitionFpga::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L5i\_PCPS\_Acquisition\_Fpga".

Implements [GNSSBlockInterface](#).

Definition at line 71 of file `gps_l5i_pcps_acquisition_fpga.h`.

### 10.201.3.6 init()

```
void GpsL5iPcpsAcquisitionFpga::init ( ) [override], [virtual]
```

Initializes acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.201.3.7 item\_size()

```
size_t GpsL5iPcpsAcquisitionFpga::item_size ( ) [inline], [override], [virtual]
```

Returns size of lv\_16sc\_t.

Implements [GNSSBlockInterface](#).

Definition at line 79 of file gps\_l5i\_pcps\_acquisition\_fpga.h.

### 10.201.3.8 mag()

```
signed int GpsL5iPcpsAcquisitionFpga::mag ( ) [override], [virtual]
```

Returns the maximum peak of grid search.

Implements [AcquisitionInterface](#).

### 10.201.3.9 reset()

```
void GpsL5iPcpsAcquisitionFpga::reset ( ) [override], [virtual]
```

Restart acquisition algorithm.

Implements [AcquisitionInterface](#).

### 10.201.3.10 role()

```
std::string GpsL5iPcpsAcquisitionFpga::role ( ) [inline], [override], [virtual]
```

Role.

Implements [GNSSBlockInterface](#).

Definition at line 63 of file gps\_l5i\_pcps\_acquisition\_fpga.h.

### 10.201.3.11 set\_channel()

```
void GpsL5iPcpsAcquisitionFpga::set_channel (
    unsigned int channel ) [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Implements [AcquisitionInterface](#).

Definition at line 114 of file gps\_l5i\_pcps\_acquisition\_fpga.h.

#### 10.201.3.12 `set_channel_fsm()`

```
void GpsL5iPcpsAcquisitionFpga::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 123 of file `gps_l5i_pcps_acquisition_fpga.h`.

#### 10.201.3.13 `set_doppler_center()`

```
void GpsL5iPcpsAcquisitionFpga::set_doppler_center (
    int doppler_center ) [override]
```

Set Doppler center for the grid search.

#### 10.201.3.14 `set_doppler_max()`

```
void GpsL5iPcpsAcquisitionFpga::set_doppler_max (
    unsigned int doppler_max ) [override], [virtual]
```

Set maximum Doppler off grid search.

Implements [AcquisitionInterface](#).

#### 10.201.3.15 `set_doppler_step()`

```
void GpsL5iPcpsAcquisitionFpga::set_doppler_step (
    unsigned int doppler_step ) [override], [virtual]
```

Set Doppler steps for the grid search.

Implements [AcquisitionInterface](#).

#### 10.201.3.16 `set_gnss_synchro()`

```
void GpsL5iPcpsAcquisitionFpga::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](#).

**10.201.3.17 set\_local\_code()**

```
void GpsL5iPcpsAcquisitionFpga::set_local_code ( ) [override], [virtual]
```

Sets local code for GPS L5 PCPS acquisition algorithm.

Implements [AcquisitionInterface](#).

**10.201.3.18 set\_resampler\_latency()**

```
void GpsL5iPcpsAcquisitionFpga::set_resampler_latency (
    uint32_t latency_samples __attribute__((unused)) ) [inline], [override]
```

Set resampler latency.

Definition at line 182 of file `gps_l5i_pcps_acquisition_fpga.h`.

**10.201.3.19 set\_state()**

```
void GpsL5iPcpsAcquisitionFpga::set_state (
    int state ) [override], [virtual]
```

If state = 1, it forces the block to start acquiring from the first sample.

Implements [AcquisitionInterface](#).

**10.201.3.20 set\_threshold()**

```
void GpsL5iPcpsAcquisitionFpga::set_threshold (
    float threshold ) [override], [virtual]
```

Set statistics threshold of PCPS algorithm.

Implements [AcquisitionInterface](#).

**10.201.3.21 stop\_acquisition()**

```
void GpsL5iPcpsAcquisitionFpga::stop_acquisition ( ) [override], [virtual]
```

Stop running acquisition.

Implements [AcquisitionInterface](#).

The documentation for this class was generated from the following file:

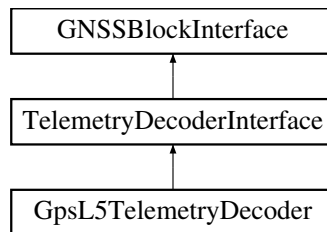
- [gps\\_l5i\\_pcps\\_acquisition\\_fpga.h](#)

## 10.202 GpsL5TelemetryDecoder Class Reference

This class implements a NAV data decoder for GPS L5.

```
#include <gps_l5_telemetry_decoder.h>
```

Inheritance diagram for GpsL5TelemetryDecoder:



### Public Member Functions

- **GpsL5TelemetryDecoder** (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\_L5\_Telemetry\_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
- void **set\_channel** (int channel) override
- void **reset** () override
- size\_t **item\_size** () override

### 10.202.1 Detailed Description

This class implements a NAV data decoder for GPS L5.

Definition at line 43 of file `gps_l5_telemetry_decoder.h`.

### 10.202.2 Member Function Documentation

#### 10.202.2.1 implementation()

```
std::string GpsL5TelemetryDecoder::implementation ( ) [inline], [override], [virtual]
```

Returns "GPS\_L5\_Telemetry\_Decoder".

Implements [GNSSBlockInterface](#).

Definition at line 60 of file `gps_l5_telemetry_decoder.h`.

The documentation for this class was generated from the following file:

- [gps\\_l5\\_telemetry\\_decoder.h](#)

## 10.203 GPU\_Complex Struct Reference

### Public Member Functions

- CUDA\_CALLABLE\_MEMBER\_DEVICE **GPU\_Complex** (float a, float b)
- CUDA\_CALLABLE\_MEMBER\_DEVICE float **magnitude2** (void)
- CUDA\_CALLABLE\_MEMBER\_DEVICE **GPU\_Complex operator\*** (const [GPU\\_Complex](#) &a)
- CUDA\_CALLABLE\_MEMBER\_DEVICE **GPU\_Complex operator+** (const [GPU\\_Complex](#) &a)
- 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**

#### 10.203.1 Detailed Description

Definition at line 45 of file `cuda_multicorrelator.h`.

The documentation for this struct was generated from the following file:

- [cuda\\_multicorrelator.h](#)

## 10.204 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)
- CUDA\_CALLABLE\_MEMBER\_DEVICE **GPU\_Complex\_Short operator\*** (const [GPU\\_Complex\\_Short](#) &a)
- CUDA\_CALLABLE\_MEMBER\_DEVICE **GPU\_Complex\_Short operator+** (const [GPU\\_Complex\\_Short](#) &a)

### Public Attributes

- float **r**
- float **i**

#### 10.204.1 Detailed Description

Definition at line 88 of file `cuda_multicorrelator.h`.

The documentation for this struct was generated from the following file:

- [cuda\\_multicorrelator.h](#)

## 10.205 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 print\_average\_values)
- bool **close\_file** ()

### 10.205.1 Detailed Description

Prints PVT information to GPX format file.

See <https://www.topografix.com/gpx.asp>

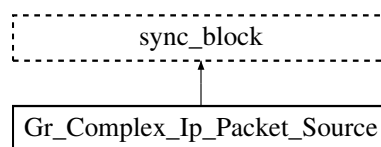
Definition at line 39 of file gpx\_printer.h.

The documentation for this class was generated from the following file:

- [gpx\\_printer.h](#)

## 10.206 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\_)

### 10.206.1 Detailed Description

Definition at line 41 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](#)

## 10.207 gtime\_t Struct Reference

### Public Attributes

- time\_t **time**
- double **sec**

### 10.207.1 Detailed Description

Definition at line 362 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.208 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**

### 10.208.1 Detailed Description

Definition at line 1084 of file rtklib.h.

The documentation for this struct was generated from the following file:

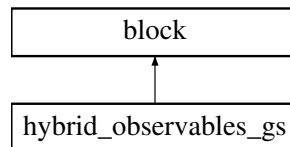
- [rtklib.h](#)

## 10.209 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\_)

#### 10.209.1 Detailed Description

This class implements a block that computes observables.

Definition at line 57 of file hybrid\_observables\_gs.h.

The documentation for this class was generated from the following file:

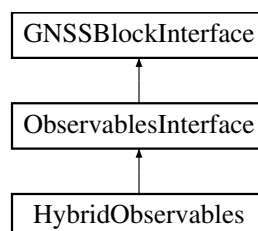
- [hybrid\\_observables\\_gs.h](#)

## 10.210 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.*

### 10.210.1 Detailed Description

This class implements an [ObservablesInterface](#) for observables of all kind of GNSS signals.

Definition at line 43 of file `hybrid_observables.h`.

### 10.210.2 Member Function Documentation

#### 10.210.2.1 implementation()

```
std::string HybridObservables::implementation ( ) [inline], [override], [virtual]
```

Returns "Hybrid\_Observables".

Implements [GNSSBlockInterface](#).

Definition at line 59 of file `hybrid_observables.h`.

#### 10.210.2.2 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`.

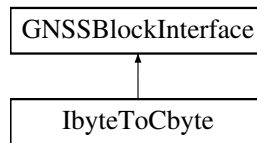
The documentation for this class was generated from the following file:

- [hybrid\\_observables.h](#)

## 10.211 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

### 10.211.1 Detailed Description

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.

### 10.211.2 Member Function Documentation

#### 10.211.2.1 implementation()

```
std::string IbyteToCbyte::implementation ( ) [inline], [override], [virtual]
```

Returns "Ibyte\_To\_Cbyte".

Implements [GNSSBlockInterface](#).

Definition at line 55 of file ibyte\_to\_cbyte.h.

The documentation for this class was generated from the following file:

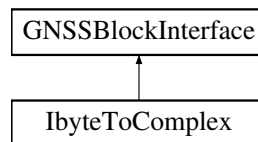
- [ibyte\\_to\\_cbyte.h](#)

## 10.212 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

### 10.212.1 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.

### 10.212.2 Member Function Documentation

#### 10.212.2.1 implementation()

```
std::string IbyteToComplex::implementation ( ) [inline], [override], [virtual]
```

Returns "Ibyte\_To\_Complex".

Implements [GNSSBlockInterface](#).

Definition at line 54 of file ibyte\_to\_complex.h.

The documentation for this class was generated from the following file:

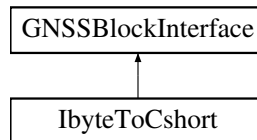
- [ibyte\\_to\\_complex.h](#)

## 10.213 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

### 10.213.1 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`.

### 10.213.2 Member Function Documentation

#### 10.213.2.1 implementation()

```
std::string IbyteToCshort::implementation ( ) [inline], [override], [virtual]
```

Returns "Ibyte\_To\_Cshort".

Implements [GNSSBlockInterface](#).

Definition at line 54 of file `ibyte_to_cshort.h`.

The documentation for this class was generated from the following file:

- [ibyte\\_to\\_cshort.h](#)

## 10.214 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.*

### 10.214.1 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.

### 10.214.2 Constructor & Destructor Documentation

#### 10.214.2.1 INIReader()

```
INIReader::INIReader (  
    const std::string & filename ) [explicit]
```

Construct **INIReader** and parse given filename. See [ini.h](#) for more info about the parsing.

### 10.214.3 Member Function Documentation

#### 10.214.3.1 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.

## 10.214.3.2 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.

## 10.214.3.3 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:

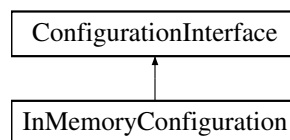
- [INIReader.h](#)

## 10.215 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

### 10.215.1 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`.

The documentation for this class was generated from the following file:

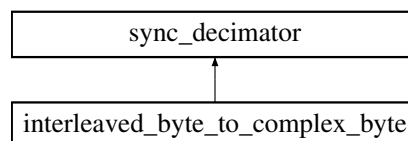
- [in\\_memory\\_configuration.h](#)

## 10.216 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** ()

### 10.216.1 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:

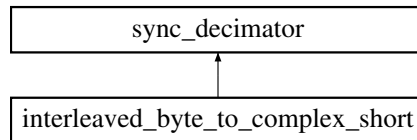
- [interleaved\\_byte\\_to\\_complex\\_byte.h](#)

## 10.217 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** ()

#### 10.217.1 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:

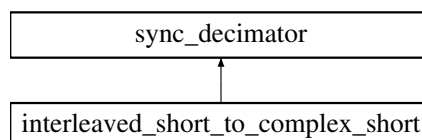
- [interleaved\\_byte\\_to\\_complex\\_short.h](#)

## 10.218 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** ()

### 10.218.1 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:

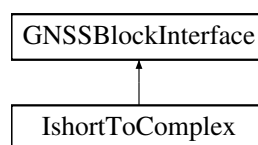
- [interleaved\\_short\\_to\\_complex\\_short.h](#)

## 10.219 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

### 10.219.1 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`.

### 10.219.2 Member Function Documentation

#### 10.219.2.1 implementation()

```
std::string IshortToComplex::implementation ( ) [inline], [override], [virtual]
```

Returns "Ishort\_To\_Complex".

Implements [GNSSBlockInterface](#).

Definition at line 54 of file `ishort_to_complex.h`.

The documentation for this class was generated from the following file:

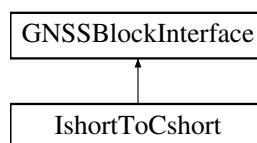
- [ishort\\_to\\_complex.h](#)

## 10.220 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

### 10.220.1 Detailed Description

Adapts a short integer (16 bits) interleaved sample stream into a `std::complex<short>` stream.

Definition at line 39 of file `ishort_to_cshort.h`.

### 10.220.2 Member Function Documentation

#### 10.220.2.1 `implementation()`

```
std::string IshortToCshort::implementation ( ) [inline], [override], [virtual]
```

Returns "Ishort\_To\_Cshort".

Implements [GNSSBlockInterface](#).

Definition at line 54 of file `ishort_to_cshort.h`.

The documentation for this class was generated from the following file:

- [ishort\\_to\\_cshort.h](#)

## 10.221 `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**

### 10.221.1 Detailed Description

Definition at line 32 of file `fft_internal.h`.

The documentation for this struct was generated from the following file:

- [fft\\_internal.h](#)

## 10.222 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 print\_average\_values)
- bool **close\_file** ()

### 10.222.1 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:

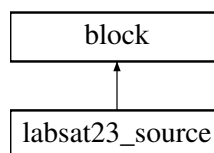
- [kml\\_printer.h](#)

## 10.223 labsat23\_source Class Reference

This class implements conversion between Labsat2 and 3 format byte packet samples 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, int channel\_selector, [Concurrent\\_Queue](#)< pmt::pmt\_t > \*queue)

### 10.223.1 Detailed Description

This class implements conversion between Labsat2 and 3 format byte packet samples to `gr_complex`.

Definition at line 47 of file `labsat23_source.h`.

The documentation for this class was generated from the following file:

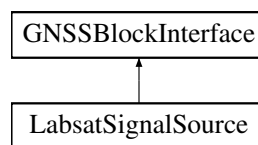
- [labsat23\\_source.h](#)

## 10.224 LabsatSignalSource Class Reference

This class reads samples stored by a LabSat 2 or LabSat 3 device.

```
#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)
- std::string **role** () override
- std::string **implementation** () override  
*Returns "Labsat\_Signal\_Source".*
- 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

### 10.224.1 Detailed Description

This class reads samples stored by a LabSat 2 or LabSat 3 device.

Definition at line 41 of file `labsat_signal_source.h`.

### 10.224.2 Member Function Documentation

10.224.2.1 `implementation()`

```
std::string LabsatSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "Labsat\_Signal\_Source".

Implements [GNSSBlockInterface](#).

Definition at line 58 of file `labsat_signal_source.h`.

The documentation for this class was generated from the following file:

- [labsat\\_signal\\_source.h](#)

10.225 `lex_t` Struct Reference

## Public Attributes

- `int n`
- `int nmax`
- `lexmsg_t * msgs`

## 10.225.1 Detailed Description

Definition at line 690 of file `rtklib.h`.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

10.226 `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]

### 10.226.1 Detailed Description

Definition at line 697 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.227 lexion\_t Struct Reference

### Public Attributes

- [gtime\\_t](#) **t0**
- double **tspan**
- double **pos0** [2]
- double **coef** [3][2]

### 10.227.1 Detailed Description

Definition at line 714 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.228 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]

### 10.228.1 Detailed Description

Definition at line 678 of file rtklib.h.

The documentation for this struct was generated from the following file:

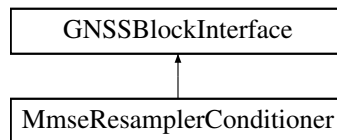
- [rtklib.h](#)

## 10.229 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

### 10.229.1 Detailed Description

Interface of a MMSE resampler block adapter to a SignalConditionerInterface.

Definition at line 48 of file mmse\_resampler\_conditioner.h.

The documentation for this class was generated from the following file:

- [mmse\\_resampler\\_conditioner.h](#)

## 10.230 ModelFunction Class Reference

### Public Member Functions

- virtual arma::vec **operator()** (const arma::vec &input)=0

### 10.230.1 Detailed Description

Definition at line 46 of file nonlinear\_tracking.h.

The documentation for this class was generated from the following file:

- [nonlinear\\_tracking.h](#)

## 10.231 Monitor\_Pvt Class Reference

This class contains parameters and outputs of the PVT block.

```
#include <monitor_pvt.h>
```

### 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

- `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**
- `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**

### 10.231.1 Detailed Description

This class contains parameters and outputs of the PVT block.

Definition at line 32 of file `monitor_pvt.h`.

## 10.231.2 Member Function Documentation

### 10.231.2.1 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 93 of file `monitor_pvt.h`.

The documentation for this class was generated from the following file:

- [monitor\\_pvt.h](#)

## 10.232 Monitor\_Pvt\_Udp\_Sink Class Reference

### Public Member Functions

- **Monitor\_Pvt\_Udp\_Sink** (const std::vector< std::string > &addresses, const uint16\_t &port, bool protobuf←\_enabled)
- bool **write\_monitor\_pvt** (const [Monitor\\_Pvt](#) \*const monitor\_pvt)

### 10.232.1 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](#)

## 10.233 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]

### 10.233.1 Detailed Description

Definition at line 1277 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.234 mt1\_header Struct Reference

### Public Attributes

- uint16\_t **toh**
- uint8\_t **mask\_id**
- uint8\_t **iod\_id**
- bool **mask\_flag**
- bool **orbit\_correction\_flag**
- bool **clock\_fullset\_flag**
- bool **clock\_subset\_flag**
- bool **code\_bias\_flag**
- bool **phase\_bias\_flag**
- bool **ura\_flag**

### 10.234.1 Detailed Description

Definition at line 29 of file galileo\_has\_data.h.

The documentation for this struct was generated from the following file:

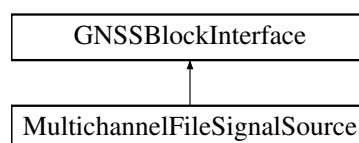
- [galileo\\_has\\_data.h](#)

## 10.235 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)
- std::string **role** () override
- std::string **implementation** () override
 

*Returns "Multichannel\_File\_Signal\_Source".*
- 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

### 10.235.1 Detailed Description

Class that reads signals samples from files at different frequency bands and adapts it to a [SignalSourceInterface](#).

Definition at line 48 of file `multichannel_file_signal_source.h`.

### 10.235.2 Member Function Documentation

#### 10.235.2.1 implementation()

```
std::string MultichannelFileSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "Multichannel\_File\_Signal\_Source".

Implements [GNSSBlockInterface](#).

Definition at line 65 of file `multichannel_file_signal_source.h`.

The documentation for this class was generated from the following file:

- [multichannel\\_file\\_signal\\_source.h](#)

## 10.236 `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 ncmx`
- `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_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`

### 10.236.1 Detailed Description

Definition at line 754 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.237 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](#) ()  
*Default destructor.*
- bool [Print\\_Nmea\\_Line](#) (const [Rtklib\\_Solver](#) \*const pvt\_data, bool print\_average\_values)  
*Print NMEA PVT and satellite info to the initialized device.*

### 10.237.1 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](https://en.wikipedia.org/wiki/NMEA_0183)

Definition at line 44 of file nmea\_printer.h.

### 10.237.2 Constructor & Destructor Documentation

#### 10.237.2.1 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.

### 10.237.2.2 ~Nmea\_Printer()

```
Nmea_Printer::~Nmea_Printer ( )
```

Default destructor.

## 10.237.3 Member Function Documentation

### 10.237.3.1 Print\_Nmea\_Line()

```
bool Nmea_Printer::Print_Nmea_Line (
    const Rtklib_Solver *const pvt_data,
    bool print_average_values )
```

Print NMEA PVT and satellite info to the initialized device.

The documentation for this class was generated from the following file:

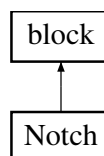
- [nmea\\_printer.h](#)

## 10.238 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)

### 10.238.1 Detailed Description

This class implements a real-time software-defined multi state notch filter.

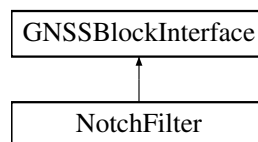
Definition at line 48 of file notch\_cc.h.

The documentation for this class was generated from the following file:

- [notch\\_cc.h](#)

## 10.239 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** ()

### 10.239.1 Detailed Description

Definition at line 35 of file notch\_filter.h.

### 10.239.2 Member Function Documentation

### 10.239.2.1 implementation()

```
std::string NotchFilter::implementation ( ) [inline], [virtual]
```

Returns "Notch\_Filter".

Implements [GNSSBlockInterface](#).

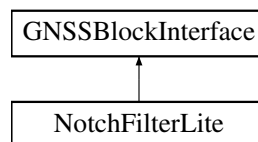
Definition at line 50 of file notch\_filter.h.

The documentation for this class was generated from the following file:

- [notch\\_filter.h](#)

## 10.240 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** ()

### 10.240.1 Detailed Description

Definition at line 35 of file notch\_filter\_lite.h.

### 10.240.2 Member Function Documentation

## 10.240.2.1 implementation()

```
std::string NotchFilterLite::implementation ( ) [inline], [virtual]
```

Returns "Notch\_Filter\_Lite".

Implements [GNSSBlockInterface](#).

Definition at line 50 of file notch\_filter\_lite.h.

The documentation for this class was generated from the following file:

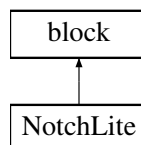
- [notch\\_filter\\_lite.h](#)

## 10.241 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)

### 10.241.1 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:

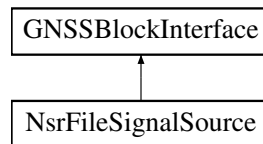
- [notch\\_lite\\_cc.h](#)

## 10.242 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)
- std::string **role** () override
- std::string **implementation** () override  
Returns "`Nsr_File_Signal_Source`".
- 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

### 10.242.1 Detailed Description

Class that reads signals samples from a file and adapts it to a `SignalSourceInterface`.

Definition at line 45 of file `nsr_file_signal_source.h`.

### 10.242.2 Member Function Documentation

#### 10.242.2.1 implementation()

```
std::string NsrFileSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "`Nsr_File_Signal_Source`".

Implements [GNSSBlockInterface](#).

Definition at line 61 of file `nsr_file_signal_source.h`.

The documentation for this class was generated from the following file:

- [nsr\\_file\\_signal\\_source.h](#)

## 10.243 ntrip\_t Struct Reference

### Public Attributes

- int **state**
- int **type**
- int **nb**
- char **url** [256]
- char **mntpnt** [256]
- char **user** [256]
- char **passwd** [256]
- char **str** [NTRIP\_MAXSTR]
- unsigned char **buff** [NTRIP\_MAXRSP]
- [tcpcli\\_t](#) \* **tcp**

### 10.243.1 Detailed Description

Definition at line 1170 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.244 Obs\_Conf Class Reference

### Public Attributes

- std::string **dump\_filename**
- int32\_t **smoothing\_factor**
- uint32\_t **nchannels\_in**
- uint32\_t **nchannels\_out**
- bool **enable\_carrier\_smoothing**
- bool **dump**
- bool **dump\_mat**

### 10.244.1 Detailed Description

Definition at line 30 of file obs\_conf.h.

The documentation for this class was generated from the following file:

- [obs\\_conf.h](#)

## 10.245 `obs_t` Struct Reference

### Public Attributes

- int **n**
- int **nmax**
- [obsd\\_t](#) \* **data**

### 10.245.1 Detailed Description

Definition at line 382 of file `rtklib.h`.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.246 `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](#)]

### 10.246.1 Detailed Description

Definition at line 369 of file `rtklib.h`.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.247 `Observables_Dump_Reader` Class Reference

### Public Member Functions

- **Observables\_Dump\_Reader** (int n\_channels)
- bool **read\_binary\_obs** ()
- bool **restart** ()
- int64\_t **num\_epochs** ()
- bool **open\_obs\_file** (std::string out\_file)
- void **close\_obs\_file** ()

## Public Attributes

- `std::vector< double > RX_time`
- `std::vector< double > TOW_at_current_symbol_s`
- `std::vector< double > Carrier_Doppler_hz`
- `std::vector< double > Acc_carrier_phase_hz`
- `std::vector< double > Pseudorange_m`
- `std::vector< double > PRN`
- `std::vector< double > valid`

### 10.247.1 Detailed Description

Definition at line 25 of file `observables_dump_reader.h`.

The documentation for this class was generated from the following file:

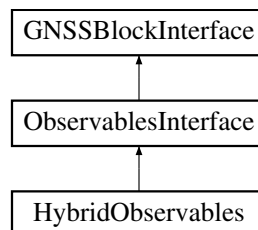
- [observables\\_dump\\_reader.h](#)

## 10.248 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

### 10.248.1 Detailed Description

This abstract class represents an interface to an observables block.

Abstract class for pseudorange\_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 43 of file `observables_interface.h`.

The documentation for this class was generated from the following file:

- [observables\\_interface.h](#)

## 10.249 opt\_t Struct Reference

### Public Attributes

- const char \* **name**
- int **format**
- void \* **var**
- const char \* **comment**

### 10.249.1 Detailed Description

Definition at line 918 of file rtklib.h.

The documentation for this struct was generated from the following file:

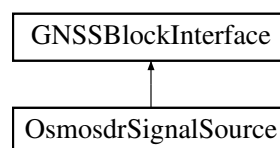
- [rtklib.h](#)

## 10.250 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)
- std::string **role** () override
- std::string **implementation** () override
  - Returns "Osmosdr\_Signal\_Source".*
- 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

### 10.250.1 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`.

### 10.250.2 Member Function Documentation

#### 10.250.2.1 implementation()

```
std::string OsmosdrSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "Osmosdr\_Signal\_Source".

Implements [GNSSBlockInterface](#).

Definition at line 62 of file `osmosdr_signal_source.h`.

The documentation for this class was generated from the following file:

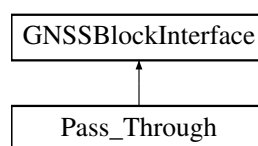
- [osmosdr\\_signal\\_source.h](#)

## 10.251 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

### 10.251.1 Detailed Description

This class implements a block that connects input and output (does nothing)

Definition at line 42 of file `pass_through.h`.

### 10.251.2 Member Function Documentation

#### 10.251.2.1 `implementation()`

```
std::string Pass_Through::implementation ( ) [inline], [override], [virtual]
```

Returns "Pass\_Through".

Implements [GNSSBlockInterface](#).

Definition at line 58 of file `pass_through.h`.

The documentation for this class was generated from the following file:

- [pass\\_through.h](#)

## 10.252 pclk\_t Struct Reference

### Public Attributes

- [gtime\\_t](#) **time**
- int **index**
- double **clk** [MAXSAT][1]
- float **std** [MAXSAT][1]

#### 10.252.1 Detailed Description

Definition at line 493 of file `rtklib.h`.

The documentation for this struct was generated from the following file:

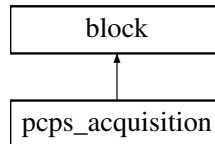
- [rtklib.h](#)

## 10.253 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 **init** ()  
*Initializes acquisition algorithm and reserves memory.*
- 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.*
- void **set\_local\_code** (std::complex< float > \*code)  
*Sets local code for PCPS acquisition algorithm.*
- void **set\_state** (int32\_t state)  
*If set to 1, ensures that acquisition starts at the first available sample.*
- void **set\_resampler\_latency** (uint32\_t latency\_samples)
- uint32\_t **mag** () const  
*Returns the maximum peak of grid search.*
- 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\_threshold** (float threshold)  
*Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (uint32\_t doppler\_max)  
*Set maximum Doppler grid search.*
- void **set\_doppler\_step** (uint32\_t doppler\_step)  
*Set Doppler steps for the grid search.*
- void **set\_doppler\_center** (int32\_t doppler\_center)  
*Set Doppler center frequency for the grid search. It will refresh the Doppler grid.*
- int **general\_work** (int noutput\_items, gr\_vector\_int &input\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)  
*Parallel Code Phase Search Acquisition signal processing.*

### Friends

- pcps\_acquisition\_sptr **pcps\_make\_acquisition** (const Acq\_Conf &conf\_)

### 10.253.1 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 92 of file `pcps_acquisition.h`.

### 10.253.2 Member Function Documentation

#### 10.253.2.1 `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 )
```

Parallel Code Phase Search Acquisition signal processing.

#### 10.253.2.2 `init()`

```
void pcps_acquisition::init ( )
```

Initializes acquisition algorithm and reserves memory.

#### 10.253.2.3 `mag()`

```
uint32_t pcps_acquisition::mag ( ) const [inline]
```

Returns the maximum peak of grid search.

Definition at line 131 of file `pcps_acquisition.h`.

#### 10.253.2.4 `set_active()`

```
void pcps_acquisition::set_active (
    bool active ) [inline]
```

Starts acquisition algorithm, turning from standby mode to active mode.

## Parameters

<i>active</i>	- bool that activates/deactivates the block.
---------------	--

Definition at line 141 of file pcps\_acquisition.h.

### 10.253.2.5 set\_channel()

```
void pcps_acquisition::set_channel (
    uint32_t channel ) [inline]
```

Set acquisition channel unique ID.

## Parameters

<i>channel</i>	- receiver channel.
----------------	---------------------

Definition at line 151 of file pcps\_acquisition.h.

### 10.253.2.6 set\_channel\_fsm()

```
void pcps_acquisition::set_channel_fsm (
    std::weak_ptr< ChannelFsm > channel_fsm ) [inline]
```

Set channel fsm associated to this acquisition instance.

Definition at line 159 of file pcps\_acquisition.h.

### 10.253.2.7 set\_doppler\_center()

```
void pcps_acquisition::set_doppler_center (
    int32_t doppler_center ) [inline]
```

Set Doppler center frequency for the grid search. It will refresh the Doppler grid.

## Parameters

<i>doppler_center</i>	- Frequency center of the search grid [Hz].
-----------------------	---

Definition at line 199 of file pcps\_acquisition.h.

**10.253.2.8 set\_doppler\_max()**

```
void pcps_acquisition::set_doppler_max (
    uint32_t doppler_max ) [inline]
```

Set maximum Doppler grid search.

**Parameters**

<i>doppler_max</i>	- Maximum Doppler shift considered in the grid search [Hz].
--------------------	---

Definition at line 179 of file pcps\_acquisition.h.

**10.253.2.9 set\_doppler\_step()**

```
void pcps_acquisition::set_doppler_step (
    uint32_t doppler_step ) [inline]
```

Set Doppler steps for the grid search.

**Parameters**

<i>doppler_step</i>	- Frequency bin of the search grid [Hz].
---------------------	--

Definition at line 189 of file pcps\_acquisition.h.

**10.253.2.10 set\_gnss\_synchro()**

```
void pcps_acquisition::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**

<i>p_gnss_synchro</i>	Satellite information shared by the processing blocks.
-----------------------	--

Definition at line 107 of file pcps\_acquisition.h.

**10.253.2.11 set\_local\_code()**

```
void pcps_acquisition::set_local_code (
    std::complex< float > * code )
```

Sets local code for PCPS acquisition algorithm.

#### Parameters

<i>code</i>	- Pointer to the PRN code.
-------------	----------------------------

#### 10.253.2.12 set\_state()

```
void pcps_acquisition::set_state (
    int32_t state )
```

If set to 1, ensures that acquisition starts at the first available sample.

#### Parameters

<i>state</i>	- int=1 forces start of acquisition
--------------	-------------------------------------

#### 10.253.2.13 set\_threshold()

```
void pcps_acquisition::set_threshold (
    float threshold ) [inline]
```

Set statistics threshold of PCPS algorithm.

#### Parameters

<i>threshold</i>	- Threshold for signal detection (check <a href="#">Navitec2012</a> , Algorithm 1, for a definition of this threshold).
------------------	---

Definition at line 169 of file pcps\_acquisition.h.

The documentation for this class was generated from the following file:

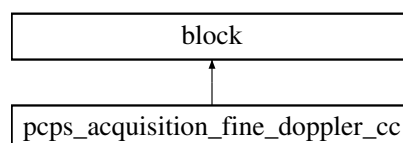
- [pcps\\_acquisition.h](#)

## 10.254 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)  
*Set acquisition/tracking common [Gnss\\_Synchro](#) object pointer to exchange synchronization data between acquisition and tracking blocks.*
- unsigned int [mag](#) () const  
*Returns the maximum peak of grid search.*
- void [init](#) ()  
*Initializes acquisition algorithm.*
- void [set\\_local\\_code](#) (std::complex< float > \*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](#) (unsigned int 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\\_threshold](#) (float threshold)  
*Set statistics threshold of PCPS algorithm.*
- void [set\\_doppler\\_max](#) (unsigned int doppler\_max)  
*Set maximum Doppler grid search.*
- void [set\\_doppler\\_step](#) (unsigned int doppler\_step)  
*Set Doppler steps for the grid search.*
- void [set\\_state](#) (int state)  
*If set to 1, ensures that acquisition starts at the first available sample.*
- unsigned int [nextPowerOf2](#) (unsigned int n)  
*Obtains the next power of 2 greater or equal to the input parameter.*
- void [dump\\_results](#) (int effective\_fft\_size)
- 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)  
*Parallel Code Phase Search Acquisition signal processing.*

## Friends

- [pcps\\_acquisition\\_fine\\_doppler\\_cc\\_sptr](#) [pcps\\_make\\_acquisition\\_fine\\_doppler\\_cc](#) (const [Acq\\_Conf](#) &conf\_)

### 10.254.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition.

Definition at line 73 of file [pcps\\_acquisition\\_fine\\_doppler\\_cc.h](#).

### 10.254.2 Constructor & Destructor Documentation

### 10.254.2.1 `~pcps_acquisition_fine_doppler_cc()`

```
pcps_acquisition_fine_doppler_cc::~pcps_acquisition_fine_doppler_cc ( ) [default]
```

Default destructor.

## 10.254.3 Member Function Documentation

### 10.254.3.1 `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 )
```

Parallel Code Phase Search Acquisition signal processing.

### 10.254.3.2 `init()`

```
void pcps_acquisition_fine_doppler_cc::init ( )
```

Initializes acquisition algorithm.

### 10.254.3.3 `mag()`

```
unsigned int pcps_acquisition_fine_doppler_cc::mag ( ) const [inline]
```

Returns the maximum peak of grid search.

Definition at line 94 of file `pcps_acquisition_fine_doppler_cc.h`.

### 10.254.3.4 `nextPowerOf2()`

```
unsigned int pcps_acquisition_fine_doppler_cc::nextPowerOf2 (
    unsigned int n )
```

Obtains the next power of 2 greater or equal to the input parameter.

**Parameters**

<i>n</i>	- Integer value to obtain the next power of 2.
----------	--

**10.254.3.5 set\_active()**

```
void pcps_acquisition_fine_doppler_cc::set_active (
    bool active ) [inline]
```

Starts acquisition algorithm, turning from standby mode to active mode.

**Parameters**

<i>active</i>	- bool that activates/deactivates the block.
---------------	--

Definition at line 115 of file pcps\_acquisition\_fine\_doppler\_cc.h.

**10.254.3.6 set\_channel()**

```
void pcps_acquisition_fine_doppler_cc::set_channel (
    unsigned int channel ) [inline]
```

Set acquisition channel unique ID.

**Parameters**

<i>channel</i>	- receiver channel.
----------------	---------------------

Definition at line 124 of file pcps\_acquisition\_fine\_doppler\_cc.h.

**10.254.3.7 set\_channel\_fsm()**

```
void pcps_acquisition_fine_doppler_cc::set_channel_fsm (
    std::weak_ptr< ChannelFsm > channel_fsm ) [inline]
```

Set channel fsm associated to this acquisition instance.

Definition at line 133 of file pcps\_acquisition\_fine\_doppler\_cc.h.

### 10.254.3.8 set\_doppler\_max()

```
void pcps_acquisition_fine_doppler_cc::set_doppler_max (
    unsigned int doppler_max ) [inline]
```

Set maximum Doppler grid search.

## Parameters

<i>doppler_max</i>	- Maximum Doppler shift considered in the grid search [Hz].
--------------------	---

Definition at line 152 of file pcps\_acquisition\_fine\_doppler\_cc.h.

10.254.3.9 `set_doppler_step()`

```
void pcps_acquisition_fine_doppler_cc::set_doppler_step (
    unsigned int doppler_step )
```

Set Doppler steps for the grid search.

## Parameters

<i>doppler_step</i>	- Frequency bin of the search grid [Hz].
---------------------	--

10.254.3.10 `set_gnss_synchro()`

```
void pcps_acquisition_fine_doppler_cc::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

<i>p_gnss_synchro</i>	Satellite information shared by the processing blocks.
-----------------------	--

Definition at line 86 of file pcps\_acquisition\_fine\_doppler\_cc.h.

10.254.3.11 `set_local_code()`

```
void pcps_acquisition_fine_doppler_cc::set_local_code (
    std::complex< float > * code )
```

Sets local code for PCPS acquisition algorithm.

## Parameters

<i>code</i>	- Pointer to the PRN code.
-------------	----------------------------

**10.254.3.12 set\_state()**

```
void pcps_acquisition_fine_doppler_cc::set_state (
    int state )
```

If set to 1, ensures that acquisition starts at the first available sample.

**Parameters**

<i>state</i>	- int=1 forces start of acquisition
--------------	-------------------------------------

**10.254.3.13 set\_threshold()**

```
void pcps_acquisition_fine_doppler_cc::set_threshold (
    float threshold ) [inline]
```

Set statistics threshold of PCPS algorithm.

**Parameters**

<i>threshold</i>	- Threshold for signal detection (check <a href="#">Navitec2012</a> , Algorithm 1, for a definition of this threshold).
------------------	---

Definition at line 143 of file pcps\_acquisition\_fine\_doppler\_cc.h.

The documentation for this class was generated from the following file:

- [pcps\\_acquisition\\_fine\\_doppler\\_cc.h](#)

**10.255 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_state (int32_t state)`  
*If set to 1, ensures that acquisition starts at the first available sample.*
- 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_threshold (float threshold)`  
*Set statistics threshold of PCPS algorithm.*
- void `set_doppler_max (uint32_t doppler_max)`  
*Set maximum Doppler grid search.*
- void `set_doppler_step (uint32_t doppler_step)`  
*Set Doppler steps for the grid search.*
- 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.*

## Friends

- `pcps_acquisition_fpga_sptr pcps_make_acquisition_fpga (pcpsconf_fpga_t conf_)`

### 10.255.1 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 77 of file `pcps_acquisition_fpga.h`.

### 10.255.2 Constructor & Destructor Documentation

#### 10.255.2.1 `~pcps_acquisition_fpga()`

```
pcps_acquisition_fpga::~pcps_acquisition_fpga ( ) [default]
```

Destructor.

### 10.255.3 Member Function Documentation

#### 10.255.3.1 init()

```
void pcps_acquisition_fpga::init ( )
```

Initializes acquisition algorithm.

#### 10.255.3.2 mag()

```
uint32_t pcps_acquisition_fpga::mag ( ) const [inline]
```

Returns the maximum peak of grid search.

Definition at line 98 of file pcps\_acquisition\_fpga.h.

#### 10.255.3.3 reset\_acquisition()

```
void pcps_acquisition_fpga::reset_acquisition ( )
```

This function triggers a HW reset of the FPGA PL.

#### 10.255.3.4 set\_active()

```
void pcps_acquisition_fpga::set_active (
    bool active )
```

Starts acquisition algorithm, turning from standby mode to active mode.

##### Parameters

<i>active</i>	- bool that activates/deactivates the block.
---------------	--

#### 10.255.3.5 set\_channel()

```
void pcps_acquisition_fpga::set_channel (
    uint32_t channel ) [inline]
```

Set acquisition channel unique ID.

## Parameters

<i>channel</i>	- receiver channel.
----------------	---------------------

Definition at line 131 of file pcps\_acquisition\_fpga.h.

**10.255.3.6 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 139 of file pcps\_acquisition\_fpga.h.

**10.255.3.7 set\_doppler\_center()**

```
void pcps_acquisition_fpga::set_doppler_center (
    int32_t doppler_center ) [inline]
```

Set Doppler center frequency for the grid search. It will refresh the Doppler grid.

## Parameters

<i>doppler_center</i>	- Frequency center of the search grid [Hz].
-----------------------	---

Definition at line 178 of file pcps\_acquisition\_fpga.h.

**10.255.3.8 set\_doppler\_max()**

```
void pcps_acquisition_fpga::set_doppler_max (
    uint32_t doppler_max ) [inline]
```

Set maximum Doppler grid search.

## Parameters

<i>doppler_max</i>	- Maximum Doppler shift considered in the grid search [Hz].
--------------------	---

Definition at line 158 of file pcps\_acquisition\_fpga.h.

**10.255.3.9 set\_doppler\_step()**

```
void pcps_acquisition_fpga::set_doppler_step (
    uint32_t doppler_step ) [inline]
```

Set Doppler steps for the grid search.

**Parameters**

<i>doppler_step</i>	- Frequency bin of the search grid [Hz].
---------------------	--

Definition at line 168 of file pcps\_acquisition\_fpga.h.

**10.255.3.10 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**

<i>p_gnss_synchro</i>	Satellite information shared by the processing blocks.
-----------------------	--

Definition at line 90 of file pcps\_acquisition\_fpga.h.

**10.255.3.11 set\_local\_code()**

```
void pcps_acquisition_fpga::set_local_code ( )
```

Sets local code for PCPS acquisition algorithm.

**10.255.3.12 set\_state()**

```
void pcps_acquisition_fpga::set_state (
    int32_t state )
```

If set to 1, ensures that acquisition starts at the first available sample.

**Parameters**

<i>state</i>	- int=1 forces start of acquisition
--------------	-------------------------------------

## 10.255.3.13 set\_threshold()

```
void pcps_acquisition_fpga::set_threshold (
    float threshold ) [inline]
```

Set statistics threshold of PCPS algorithm.

## Parameters

<i>threshold</i>	- Threshold for signal detection (check <a href="#">Navitec2012</a> , Algorithm 1, for a definition of this threshold).
------------------	---

Definition at line 149 of file pcps\_acquisition\_fpga.h.

The documentation for this class was generated from the following file:

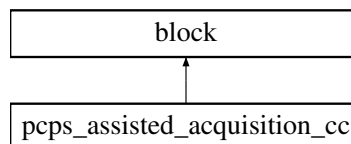
- [pcps\\_acquisition\\_fpga.h](#)

## 10.256 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)  
*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](#) (std::complex< float > \*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\\_threshold](#) (float threshold)  
*Set statistics threshold of PCPS algorithm.*
- void [set\\_state](#) (int32\_t state)
- void [set\\_doppler\\_max](#) (uint32\_t doppler\_max)  
*Set maximum Doppler grid search.*
- void [set\\_doppler\\_step](#) (uint32\_t doppler\_step)  
*Set Doppler steps for the grid search.*
- int [general\\_work](#) (int noutput\_items, gr\_vector\_int &ninput\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)  
*Parallel Code Phase Search Acquisition signal processing.*
- void [forecast](#) (int noutput\_items, gr\_vector\_int &ninput\_items\_required)

## Friends

- pcps\_assisted\_acquisition\_cc\_sptr [pcps\\_make\\_assisted\\_acquisition\\_cc](#) (int32\_t max\_dwells, uint32\_t sampled\_ms, int32\_t doppler\_max, int32\_t doppler\_min, int64\_t fs\_in, int32\_t samples\_per\_ms, bool dump, const std::string &dump\_filename, bool enable\_monitor\_output)

### 10.256.1 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\_assisted\_acquisition\_cc.h.

### 10.256.2 Constructor & Destructor Documentation

#### 10.256.2.1 ~pcps\_assisted\_acquisition\_cc()

```
pcps_assisted_acquisition_cc::~pcps_assisted_acquisition_cc ( )
```

Default destructor.

### 10.256.3 Member Function Documentation

**10.256.3.1 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 )
```

Parallel Code Phase Search Acquisition signal processing.

**10.256.3.2 init()**

```
void pcps_assisted_acquisition_cc::init ( )
```

Initializes acquisition algorithm.

**10.256.3.3 mag()**

```
uint32_t pcps_assisted_acquisition_cc::mag ( ) const [inline]
```

Returns the maximum peak of grid search.

Definition at line 96 of file pcps\_assisted\_acquisition\_cc.h.

**10.256.3.4 set\_active()**

```
void pcps_assisted_acquisition_cc::set_active (
    bool active ) [inline]
```

Starts acquisition algorithm, turning from standby mode to active mode.

**Parameters**

<i>active</i>	- bool that activates/deactivates the block.
---------------	--

Definition at line 117 of file pcps\_assisted\_acquisition\_cc.h.

**10.256.3.5 set\_channel()**

```
void pcps_assisted_acquisition_cc::set_channel (
    uint32_t channel ) [inline]
```

Set acquisition channel unique ID.

**Parameters**

<i>channel</i>	- receiver channel.
----------------	---------------------

Definition at line 126 of file pcps\_assisted\_acquisition\_cc.h.

**10.256.3.6 set\_channel\_fsm()**

```
void pcps_assisted_acquisition_cc::set_channel_fsm (
    std::weak_ptr< ChannelFsm > channel_fsm ) [inline]
```

Set channel fsm associated to this acquisition instance.

Definition at line 134 of file pcps\_assisted\_acquisition\_cc.h.

**10.256.3.7 set\_doppler\_max()**

```
void pcps_assisted_acquisition_cc::set_doppler_max (
    uint32_t doppler_max ) [inline]
```

Set maximum Doppler grid search.

**Parameters**

<i>doppler_max</i>	- Maximum Doppler shift considered in the grid search [Hz].
--------------------	---

Definition at line 158 of file pcps\_assisted\_acquisition\_cc.h.

**10.256.3.8 set\_doppler\_step()**

```
void pcps_assisted_acquisition_cc::set_doppler_step (
    uint32_t doppler_step )
```

Set Doppler steps for the grid search.

**Parameters**

<i>doppler_step</i>	- Frequency bin of the search grid [Hz].
---------------------	--

## 10.256.3.9 set\_gnss\_synchro()

```
void pcps_assisted_acquisition_cc::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

<i>p_gnss_synchro</i>	Satellite information shared by the processing blocks.
-----------------------	--

Definition at line 88 of file pcps\_assisted\_acquisition\_cc.h.

## 10.256.3.10 set\_local\_code()

```
void pcps_assisted_acquisition_cc::set_local_code (
    std::complex< float > * code )
```

Sets local code for PCPS acquisition algorithm.

## Parameters

<i>code</i>	- Pointer to the PRN code.
-------------	----------------------------

## 10.256.3.11 set\_threshold()

```
void pcps_assisted_acquisition_cc::set_threshold (
    float threshold ) [inline]
```

Set statistics threshold of PCPS algorithm.

## Parameters

<i>threshold</i>	- Threshold for signal detection (check <a href="#">Navitec2012</a> , Algorithm 1, for a definition of this threshold).
------------------	---

Definition at line 144 of file pcps\_assisted\_acquisition\_cc.h.

The documentation for this class was generated from the following file:

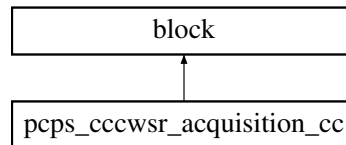
- [pcps\\_assisted\\_acquisition\\_cc.h](#)

## 10.257 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)  
*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](#) (std::complex< float > \*code\_data, std::complex< float > \*code\_pilot)  
*Sets local code for CCCWSR acquisition algorithm.*
- void [set\\_active](#) (bool active)  
*Starts acquisition algorithm, turning from standby mode to active mode.*
- void [set\\_state](#) (int32\_t state)  
*If set to 1, ensures that acquisition starts at the first available sample.*
- 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\\_threshold](#) (float threshold)  
*Set statistics threshold of CCCWSR algorithm.*
- void [set\\_doppler\\_max](#) (uint32\_t doppler\_max)  
*Set maximum Doppler grid search.*
- void [set\\_doppler\\_step](#) (uint32\_t doppler\_step)  
*Set Doppler steps for the grid search.*
- int [general\\_work](#) (int noutput\_items, gr\_vector\_int &ninput\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)  
*Coherent [Channel](#) Combining With Sign Recovery Acquisition signal processing.*

### Friends

- pcps\_cccwsr\_acquisition\_cc\_sptr [pcps\\_cccwsr\\_make\\_acquisition\\_cc](#) (uint32\_t sampled\_ms, uint32\_t max\_dwells, uint32\_t doppler\_max, int64\_t fs\_in, int32\_t samples\_per\_ms, int32\_t samples\_per\_code, bool dump, const std::string &dump\_filename, bool enable\_monitor\_output)

### 10.257.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition with Coherent [Channel](#) Combining With Sign Recovery scheme.

Definition at line 62 of file pcps\_cccwsr\_acquisition\_cc.h.

### 10.257.2 Constructor & Destructor Documentation

#### 10.257.2.1 `~pcps_cccwsr_acquisition_cc()`

```
pcps_cccwsr_acquisition_cc::~~pcps_cccwsr_acquisition_cc ( )
```

Default destructor.

### 10.257.3 Member Function Documentation

#### 10.257.3.1 `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 )
```

Coherent [Channel](#) Combining With Sign Recovery Acquisition signal processing.

#### 10.257.3.2 `init()`

```
void pcps_cccwsr_acquisition_cc::init ( )
```

Initializes acquisition algorithm.

#### 10.257.3.3 `mag()`

```
uint32_t pcps_cccwsr_acquisition_cc::mag ( ) const [inline]
```

Returns the maximum peak of grid search.

Definition at line 83 of file pcps\_cccwsr\_acquisition\_cc.h.

#### 10.257.3.4 `set_active()`

```
void pcps_cccwsr_acquisition_cc::set_active (
    bool active ) [inline]
```

Starts acquisition algorithm, turning from standby mode to active mode.

**Parameters**

<i>active</i>	- bool that activates/deactivates the block.
---------------	--

Definition at line 105 of file pcps\_cccwsr\_acquisition\_cc.h.

**10.257.3.5 set\_channel()**

```
void pcps_cccwsr_acquisition_cc::set_channel (
    uint32_t channel ) [inline]
```

Set acquisition channel unique ID.

**Parameters**

<i>channel</i>	- receiver channel.
----------------	---------------------

Definition at line 121 of file pcps\_cccwsr\_acquisition\_cc.h.

**10.257.3.6 set\_channel\_fsm()**

```
void pcps_cccwsr_acquisition_cc::set_channel_fsm (
    std::weak_ptr< ChannelFsm > channel_fsm ) [inline]
```

Set channel fsm associated to this acquisition instance.

Definition at line 129 of file pcps\_cccwsr\_acquisition\_cc.h.

**10.257.3.7 set\_doppler\_max()**

```
void pcps_cccwsr_acquisition_cc::set_doppler_max (
    uint32_t doppler_max ) [inline]
```

Set maximum Doppler grid search.

**Parameters**

<i>doppler_max</i>	- Maximum Doppler shift considered in the grid search [Hz].
--------------------	---

Definition at line 148 of file pcps\_cccwsr\_acquisition\_cc.h.

**10.257.3.8 set\_doppler\_step()**

```
void pcps_cccwsr_acquisition_cc::set_doppler_step (
    uint32_t doppler_step ) [inline]
```

Set Doppler steps for the grid search.

**Parameters**

<i>doppler_step</i>	- Frequency bin of the search grid [Hz].
---------------------	--

Definition at line 157 of file pcps\_cccwsr\_acquisition\_cc.h.

**10.257.3.9 set\_gnss\_synchro()**

```
void pcps_cccwsr_acquisition_cc::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**

<i>p_gnss_synchro</i>	Satellite information shared by the processing blocks.
-----------------------	--

Definition at line 75 of file pcps\_cccwsr\_acquisition\_cc.h.

**10.257.3.10 set\_local\_code()**

```
void pcps_cccwsr_acquisition_cc::set_local_code (
    std::complex< float > * code_data,
    std::complex< float > * code_pilot )
```

Sets local code for CCCWSR acquisition algorithm.

**Parameters**

<i>data_code</i>	- Pointer to the data PRN code.
<i>pilot_code</i>	- Pointer to the pilot PRN code.

**10.257.3.11 set\_state()**

```
void pcps_cccwsr_acquisition_cc::set_state (
```

```
int32_t state )
```

If set to 1, ensures that acquisition starts at the first available sample.

#### Parameters

<i>state</i>	- int=1 forces start of acquisition
--------------	-------------------------------------

### 10.257.3.12 set\_threshold()

```
void pcps_cccwsr_acquisition_cc::set_threshold (
    float threshold ) [inline]
```

Set statistics threshold of CCCWSR algorithm.

#### Parameters

<i>threshold</i>	- Threshold for signal detection (check <a href="#">Navitec2012</a> , Algorithm 1, for a definition of this threshold).
------------------	---

Definition at line 139 of file pcps\_cccwsr\_acquisition\_cc.h.

The documentation for this class was generated from the following file:

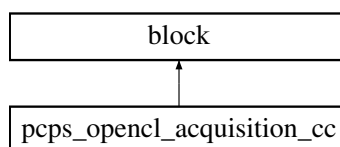
- [pcps\\_cccwsr\\_acquisition\\_cc.h](#)

## 10.258 pcps\_openc1\_acquisition\_cc Class Reference

This class implements a Parallel Code Phase Search Acquisition.

```
#include <pcps_openc1_acquisition_cc.h>
```

Inheritance diagram for pcps\_openc1\_acquisition\_cc:



### Public Member Functions

- [~pcps\\_openc1\\_acquisition\\_cc](#) ()  
*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** (std::complex< float > \*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\_state** (int state)  
*If set to 1, ensures that acquisition starts at the first available sample.*
- 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\_threshold** (float threshold)  
*Set statistics threshold of PCPS algorithm.*
- void **set\_doppler\_max** (uint32\_t doppler\_max)  
*Set maximum Doppler grid search.*
- void **set\_doppler\_step** (uint32\_t doppler\_step)  
*Set Doppler steps for the grid search.*
- bool **openc1\_ready** () const
- void **acquisition\_core\_volk** ()
- void **acquisition\_core\_openc1** ()
- int **general\_work** (int noutput\_items, gr\_vector\_int &ninput\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)  
*Parallel Code Phase Search Acquisition signal processing.*

## Friends

- pcps\_openc1\_acquisition\_cc\_sptr **pcps\_make\_openc1\_acquisition\_cc** (uint32\_t sampled\_ms, uint32\_t max\_dwells, uint32\_t doppler\_max, int64\_t fs\_in, int samples\_per\_ms, int samples\_per\_code, bool bit\_transition\_flag, bool dump, const std::string &dump\_filename, bool enable\_monitor\_output)

### 10.258.1 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 83 of file pcps\_openc1\_acquisition\_cc.h.

### 10.258.2 Constructor & Destructor Documentation

### 10.258.2.1 `~pcps_openc1_acquisition_cc()`

```
pcps_openc1_acquisition_cc::~~pcps_openc1_acquisition_cc ( )
```

Default destructor.

## 10.258.3 Member Function Documentation

### 10.258.3.1 `general_work()`

```
int pcps_openc1_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 )
```

Parallel Code Phase Search Acquisition signal processing.

### 10.258.3.2 `init()`

```
void pcps_openc1_acquisition_cc::init ( )
```

Initializes acquisition algorithm.

### 10.258.3.3 `mag()`

```
uint32_t pcps_openc1_acquisition_cc::mag ( ) const [inline]
```

Returns the maximum peak of grid search.

Definition at line 104 of file `pcps_openc1_acquisition_cc.h`.

### 10.258.3.4 `set_active()`

```
void pcps_openc1_acquisition_cc::set_active (
    bool active ) [inline]
```

Starts acquisition algorithm, turning from standby mode to active mode.

## Parameters

<i>active</i>	- bool that activates/deactivates the block.
---------------	--

Definition at line 125 of file pcps\_openc1\_acquisition\_cc.h.

**10.258.3.5 set\_channel()**

```
void pcps_openc1_acquisition_cc::set_channel (
    uint32_t channel ) [inline]
```

Set acquisition channel unique ID.

## Parameters

<i>channel</i>	- receiver channel.
----------------	---------------------

Definition at line 141 of file pcps\_openc1\_acquisition\_cc.h.

**10.258.3.6 set\_channel\_fsm()**

```
void pcps_openc1_acquisition_cc::set_channel_fsm (
    std::weak_ptr< ChannelFsm > channel_fsm ) [inline]
```

Set channel fsm associated to this acquisition instance.

Definition at line 149 of file pcps\_openc1\_acquisition\_cc.h.

**10.258.3.7 set\_doppler\_max()**

```
void pcps_openc1_acquisition_cc::set_doppler_max (
    uint32_t doppler_max ) [inline]
```

Set maximum Doppler grid search.

## Parameters

<i>doppler_max</i>	- Maximum Doppler shift considered in the grid search [Hz].
--------------------	---

Definition at line 168 of file pcps\_openc1\_acquisition\_cc.h.

**10.258.3.8 set\_doppler\_step()**

```
void pcps_openc1_acquisition_cc::set_doppler_step (
    uint32_t doppler_step ) [inline]
```

Set Doppler steps for the grid search.

**Parameters**

<i>doppler_step</i>	- Frequency bin of the search grid [Hz].
---------------------	--

Definition at line 177 of file pcps\_openc1\_acquisition\_cc.h.

**10.258.3.9 set\_gnss\_synchro()**

```
void pcps_openc1_acquisition_cc::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**

<i>p_gnss_synchro</i>	Satellite information shared by the processing blocks.
-----------------------	--

Definition at line 96 of file pcps\_openc1\_acquisition\_cc.h.

**10.258.3.10 set\_local\_code()**

```
void pcps_openc1_acquisition_cc::set_local_code (
    std::complex< float > * code )
```

Sets local code for PCPS acquisition algorithm.

**Parameters**

<i>code</i>	- Pointer to the PRN code.
-------------	----------------------------

**10.258.3.11 set\_state()**

```
void pcps_openc1_acquisition_cc::set_state (
    int state )
```

If set to 1, ensures that acquisition starts at the first available sample.

## Parameters

<i>state</i>	- int=1 forces start of acquisition
--------------	-------------------------------------

10.258.3.12 `set_threshold()`

```
void pcps_openc1_acquisition_cc::set_threshold (
    float threshold ) [inline]
```

Set statistics threshold of PCPS algorithm.

## Parameters

<i>threshold</i>	- Threshold for signal detection (check <a href="#">Navitec2012</a> , Algorithm 1, for a definition of this threshold).
------------------	---

Definition at line 159 of file `pcps_openc1_acquisition_cc.h`.

The documentation for this class was generated from the following file:

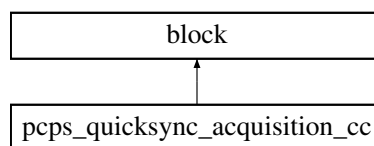
- [pcps\\_openc1\\_acquisition\\_cc.h](#)

10.259 `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)  
*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` (std::complex< float > \*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_state` (int32\_t state)  
*If set to 1, ensures that acquisition starts at the first available sample.*
- 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_threshold` (float threshold)  
*Set statistics threshold of PCPS algorithm.*
- void `set_doppler_max` (uint32\_t doppler\_max)  
*Set maximum Doppler grid search.*
- void `set_doppler_step` (uint32\_t doppler\_step)  
*Set Doppler steps for the grid search.*
- int `general_work` (int noutput\_items, gr\_vector\_int &ninput\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)  
*Parallel Code Phase Search Acquisition signal processing.*

## Friends

- pcps\_quicksync\_acquisition\_cc\_sptr **pcps\_quicksync\_make\_acquisition\_cc** (uint32\_t folding\_factor, uint32\_t sampled\_ms, uint32\_t max\_dwells, uint32\_t doppler\_max, int64\_t fs\_in, int32\_t samples\_per\_ms, int32\_t samples\_per\_code, bool bit\_transition\_flag, bool dump, const std::string &dump\_filename, bool enable\_monitor\_output)

## 10.259.1 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 84 of file pcps\_quicksync\_acquisition\_cc.h.

## 10.259.2 Constructor & Destructor Documentation

### 10.259.2.1 ~pcps\_quicksync\_acquisition\_cc()

```
pcps_quicksync_acquisition_cc::~pcps_quicksync_acquisition_cc ( )
```

Default destructor.

### 10.259.3 Member Function Documentation

#### 10.259.3.1 `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 )
```

Parallel Code Phase Search Acquisition signal processing.

#### 10.259.3.2 `init()`

```
void pcps_quicksync_acquisition_cc::init ( )
```

Initializes acquisition algorithm.

#### 10.259.3.3 `mag()`

```
uint32_t pcps_quicksync_acquisition_cc::mag ( ) const [inline]
```

Returns the maximum peak of grid search.

Definition at line 105 of file `pcps_quicksync_acquisition_cc.h`.

#### 10.259.3.4 `set_active()`

```
void pcps_quicksync_acquisition_cc::set_active (
    bool active ) [inline]
```

Starts acquisition algorithm, turning from standby mode to active mode.

##### Parameters

<i>active</i>	- bool that activates/deactivates the block.
---------------	--

Definition at line 126 of file `pcps_quicksync_acquisition_cc.h`.

### 10.259.3.5 set\_channel()

```
void pcps_quicksync_acquisition_cc::set_channel (
    uint32_t channel ) [inline]
```

Set acquisition channel unique ID.

#### Parameters

<i>channel</i>	- receiver channel.
----------------	---------------------

Definition at line 142 of file pcps\_quicksync\_acquisition\_cc.h.

### 10.259.3.6 set\_channel\_fsm()

```
void pcps_quicksync_acquisition_cc::set_channel_fsm (
    std::weak_ptr< ChannelFsm > channel_fsm ) [inline]
```

Set channel fsm associated to this acquisition instance.

Definition at line 150 of file pcps\_quicksync\_acquisition\_cc.h.

### 10.259.3.7 set\_doppler\_max()

```
void pcps_quicksync_acquisition_cc::set_doppler_max (
    uint32_t doppler_max ) [inline]
```

Set maximum Doppler grid search.

#### Parameters

<i>doppler_max</i>	- Maximum Doppler shift considered in the grid search [Hz].
--------------------	---

Definition at line 169 of file pcps\_quicksync\_acquisition\_cc.h.

### 10.259.3.8 set\_doppler\_step()

```
void pcps_quicksync_acquisition_cc::set_doppler_step (
    uint32_t doppler_step ) [inline]
```

Set Doppler steps for the grid search.

## Parameters

<i>doppler_step</i>	- Frequency bin of the search grid [Hz].
---------------------	--

Definition at line 178 of file pcps\_quicksync\_acquisition\_cc.h.

10.259.3.9 `set_gnss_synchro()`

```
void pcps_quicksync_acquisition_cc::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

<i>p_gnss_synchro</i>	Satellite information shared by the processing blocks.
-----------------------	--

Definition at line 97 of file pcps\_quicksync\_acquisition\_cc.h.

10.259.3.10 `set_local_code()`

```
void pcps_quicksync_acquisition_cc::set_local_code (
    std::complex< float > * code )
```

Sets local code for PCPS acquisition algorithm.

## Parameters

<i>code</i>	- Pointer to the PRN code.
-------------	----------------------------

10.259.3.11 `set_state()`

```
void pcps_quicksync_acquisition_cc::set_state (
    int32_t state )
```

If set to 1, ensures that acquisition starts at the first available sample.

## Parameters

<i>state</i>	- int=1 forces start of acquisition
--------------	-------------------------------------

## 10.259.3.12 set\_threshold()

```
void pcps_quicksync_acquisition_cc::set_threshold (
    float threshold ) [inline]
```

Set statistics threshold of PCPS algorithm.

## Parameters

<i>threshold</i>	- Threshold for signal detection (check <a href="#">Navitec2012</a> , Algorithm 1, for a definition of this threshold).
------------------	---

Definition at line 160 of file pcps\_quicksync\_acquisition\_cc.h.

The documentation for this class was generated from the following file:

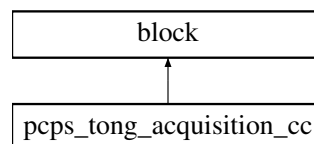
- [pcps\\_quicksync\\_acquisition\\_cc.h](#)

## 10.260 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)  
*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](#) (std::complex< float > \*code)  
*Sets local code for TONG acquisition algorithm.*
- void [set\\_active](#) (bool active)  
*Starts acquisition algorithm, turning from standby mode to active mode.*

- void [set\\_state](#) (int32\_t state)  
*If set to 1, ensures that acquisition starts at the first available sample.*
- 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\\_threshold](#) (float threshold)  
*Set statistics threshold of TONG algorithm.*
- void [set\\_doppler\\_max](#) (uint32\_t doppler\_max)  
*Set maximum Doppler grid search.*
- void [set\\_doppler\\_step](#) (uint32\_t doppler\_step)  
*Set Doppler steps for the grid search.*
- int [general\\_work](#) (int noutput\_items, gr\_vector\_int &ninput\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)  
*Parallel Code Phase Search Acquisition signal processing.*

## Friends

- pcps\_tong\_acquisition\_cc\_sptr **pcps\_tong\_make\_acquisition\_cc** (uint32\_t sampled\_ms, uint32\_t doppler\_max, int64\_t fs\_in, int32\_t samples\_per\_ms, int32\_t samples\_per\_code, uint32\_t tong\_init\_val, uint32\_t tong\_max\_val, uint32\_t tong\_max\_dwells, bool dump, const std::string &dump\_filename, bool enable\_monitor\_output)

### 10.260.1 Detailed Description

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

Definition at line 78 of file pcps\_tong\_acquisition\_cc.h.

### 10.260.2 Constructor & Destructor Documentation

#### 10.260.2.1 ~pcps\_tong\_acquisition\_cc()

```
pcps_tong_acquisition_cc::~pcps_tong_acquisition_cc ( )
```

Default destructor.

### 10.260.3 Member Function Documentation

**10.260.3.1 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 )
```

Parallel Code Phase Search Acquisition signal processing.

**10.260.3.2 init()**

```
void pcps_tong_acquisition_cc::init ( )
```

Initializes acquisition algorithm.

**10.260.3.3 mag()**

```
uint32_t pcps_tong_acquisition_cc::mag ( ) const [inline]
```

Returns the maximum peak of grid search.

Definition at line 99 of file `pcps_tong_acquisition_cc.h`.

**10.260.3.4 set\_active()**

```
void pcps_tong_acquisition_cc::set_active (
    bool active ) [inline]
```

Starts acquisition algorithm, turning from standby mode to active mode.

**Parameters**

<i>active</i>	- bool that activates/deactivates the block.
---------------	--

Definition at line 120 of file `pcps_tong_acquisition_cc.h`.

**10.260.3.5 set\_channel()**

```
void pcps_tong_acquisition_cc::set_channel (
    uint32_t channel ) [inline]
```

Set acquisition channel unique ID.

**Parameters**

<i>channel</i>	- receiver channel.
----------------	---------------------

Definition at line 136 of file pcps\_tong\_acquisition\_cc.h.

**10.260.3.6 set\_channel\_fsm()**

```
void pcps_tong_acquisition_cc::set_channel_fsm (
    std::weak_ptr< ChannelFsm > channel_fsm ) [inline]
```

Set channel fsm associated to this acquisition instance.

Definition at line 144 of file pcps\_tong\_acquisition\_cc.h.

**10.260.3.7 set\_doppler\_max()**

```
void pcps_tong_acquisition_cc::set_doppler_max (
    uint32_t doppler_max ) [inline]
```

Set maximum Doppler grid search.

**Parameters**

<i>doppler_max</i>	- Maximum Doppler shift considered in the grid search [Hz].
--------------------	---

Definition at line 163 of file pcps\_tong\_acquisition\_cc.h.

**10.260.3.8 set\_doppler\_step()**

```
void pcps_tong_acquisition_cc::set_doppler_step (
    uint32_t doppler_step ) [inline]
```

Set Doppler steps for the grid search.

**Parameters**

<i>doppler_step</i>	- Frequency bin of the search grid [Hz].
---------------------	--

Definition at line 172 of file pcps\_tong\_acquisition\_cc.h.

10.260.3.9 `set_gnss_synchro()`

```
void pcps_tong_acquisition_cc::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

<i>p_gnss_synchro</i>	Satellite information shared by the processing blocks.
-----------------------	--

Definition at line 91 of file `pcps_tong_acquisition_cc.h`.

10.260.3.10 `set_local_code()`

```
void pcps_tong_acquisition_cc::set_local_code (
    std::complex< float > * code )
```

Sets local code for TONG acquisition algorithm.

## Parameters

<i>code</i>	- Pointer to the PRN code.
-------------	----------------------------

10.260.3.11 `set_state()`

```
void pcps_tong_acquisition_cc::set_state (
    int32_t state )
```

If set to 1, ensures that acquisition starts at the first available sample.

## Parameters

<i>state</i>	- int=1 forces start of acquisition
--------------	-------------------------------------

10.260.3.12 `set_threshold()`

```
void pcps_tong_acquisition_cc::set_threshold (
    float threshold ) [inline]
```

Set statistics threshold of TONG algorithm.

## Parameters

<i>threshold</i>	- Threshold for signal detection (check <a href="#">Navitec2012</a> , Algorithm 1, for a definition of this threshold).
------------------	---

Definition at line 154 of file pcps\_tong\_acquisition\_cc.h.

The documentation for this class was generated from the following file:

- [pcps\\_tong\\_acquisition\\_cc.h](#)

## 10.261 pcpsconf\_fpga\_t Struct Reference

### Public Attributes

- std::string **device\_name**
- int64\_t **fs\_in**
- float **doppler\_step2**
- uint32\_t \* **all\_fft\_codes**
- uint32\_t **doppler\_max**
- uint32\_t **select\_queue\_Fpga**
- uint32\_t **downsampling\_factor**
- uint32\_t **total\_block\_exp**
- uint32\_t **excludelimit**
- uint32\_t **num\_doppler\_bins\_step2**
- uint32\_t **max\_num\_acqs**
- int32\_t **samples\_per\_code**
- int32\_t **code\_length**
- bool **make\_2\_steps**
- bool **repeat\_satellite**

### 10.261.1 Detailed Description

Definition at line 45 of file pcps\_acquisition\_fpga.h.

The documentation for this struct was generated from the following file:

- [pcps\\_acquisition\\_fpga.h](#)

## 10.262 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]

### 10.262.1 Detailed Description

Definition at line 406 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.263 pcvs\_t Struct Reference

### Public Attributes

- int **n**
- int **nmax**
- [pcv\\_t](#) \* **pcv**

### 10.263.1 Detailed Description

Definition at line 418 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.264 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]

### 10.264.1 Detailed Description

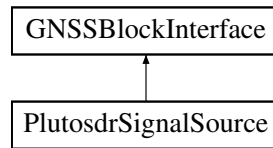
Definition at line 480 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.265 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)
- std::string **role** () override
- std::string **implementation** () override  
Returns "Plutosdr\_Signal\_Source".
- 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

### 10.265.1 Detailed Description

Definition at line 45 of file plutosdr\_signal\_source.h.

### 10.265.2 Member Function Documentation

#### 10.265.2.1 implementation()

```
std::string PlutosdrSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "Plutosdr\_Signal\_Source".

Implements [GNSSBlockInterface](#).

Definition at line 62 of file plutosdr\_signal\_source.h.

The documentation for this class was generated from the following file:

- [plutosdr\\_signal\\_source.h](#)

## 10.266 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]

### 10.266.1 Detailed Description

Definition at line 742 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.267 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 **sbscorr**
- int **sbassatsel**
- int **rovpos**

- int **refpos**
- double **eratio** [NFREQ]
- double **err** [5]
- double **std** [3]
- double **prn** [6]
- double **sclkstab**
- double **thresar** [8]
- double **elmaskar**
- double **elmaskhold**
- double **thresslip**
- double **maxtdiff**
- double **maxinno**
- 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 **rxopt** [2][256]
- int **posopt** [6]
- int **syncsol**
- double **odisp** [2][6 \* 11]
- [exterr\\_t](#) **exterr**
- int **freqopt**
- char **pppopt** [256]

### 10.267.1 Detailed Description

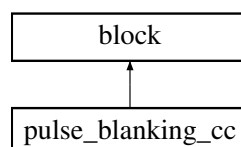
Definition at line 944 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.268 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)

### 10.268.1 Detailed Description

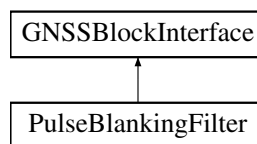
Definition at line 44 of file pulse\_blanking\_cc.h.

The documentation for this class was generated from the following file:

- [pulse\\_blanking\\_cc.h](#)

## 10.269 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

### 10.269.1 Detailed Description

Definition at line 39 of file pulse\_blanking\_filter.h.

## 10.269.2 Member Function Documentation

### 10.269.2.1 implementation()

```
std::string PulseBlankingFilter::implementation ( ) [inline], [override], [virtual]
```

Returns "Pulse\_Blanking\_Filter".

Implements [GNSSBlockInterface](#).

Definition at line 54 of file pulse\_blanking\_filter.h.

The documentation for this class was generated from the following file:

- [pulse\\_blanking\\_filter.h](#)

## 10.270 Pvt\_Conf Class Reference

### Public Attributes

- std::map< int, int > **rtcm\_msg\_rate\_ms**
- std::string **rinex\_name**
- std::string **dump\_filename**
- std::string **nmea\_dump\_filename**
- std::string **nmea\_dump\_devname**
- std::string **rtcm\_dump\_devname**
- std::string **output\_path**
- std::string **rinex\_output\_path**
- std::string **gpx\_output\_path**
- std::string **geojson\_output\_path**
- std::string **nmea\_output\_file\_path**
- std::string **kml\_output\_path**
- std::string **xml\_output\_path**
- std::string **rtcm\_output\_file\_path**
- std::string **udp\_addresses**
- uint32\_t **type\_of\_receiver**
- int32\_t **output\_rate\_ms**
- int32\_t **display\_rate\_ms**
- int32\_t **kml\_rate\_ms**
- int32\_t **gpx\_rate\_ms**
- int32\_t **geojson\_rate\_ms**
- int32\_t **nmea\_rate\_ms**
- int32\_t **rinex\_version**
- int32\_t **rinexobs\_rate\_ms**
- int32\_t **max\_obs\_block\_rx\_clock\_offset\_ms**
- int **udp\_port**
- uint16\_t **rtcm\_tcp\_port**
- uint16\_t **rtcm\_station\_id**
- bool **flag\_nmea\_tty\_port**

- bool **flag\_rtcn\_server**
- bool **flag\_rtcn\_tty\_port**
- bool **output\_enabled**
- bool **rinex\_output\_enabled**
- bool **gpx\_output\_enabled**
- bool **geojson\_output\_enabled**
- bool **nmea\_output\_file\_enabled**
- bool **kml\_output\_enabled**
- bool **xml\_output\_enabled**
- bool **rtcn\_output\_file\_enabled**
- bool **monitor\_enabled**
- bool **protobuf\_enabled**
- bool **enable\_rx\_clock\_correction**
- bool **show\_local\_time\_zone**
- bool **pre\_2009\_file**
- bool **dump**
- bool **dump\_mat**

### 10.270.1 Detailed Description

Definition at line 30 of file `pvt_conf.h`.

The documentation for this class was generated from the following file:

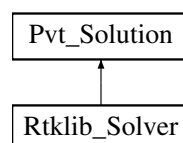
- [pvt\\_conf.h](#)

## 10.271 Pvt\_Solution Class Reference

Base class for a PVT solution.

```
#include <pvt_solution.h>
```

Inheritance diagram for Pvt\_Solution:



## Public Member Functions

- 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\_pre\_2009\_file** (bool pre\_2009\_file)  
*Flag for the week rollover computation in post processing mode for signals older than 2009.*
- void **set\_averaging\_depth** (int depth)  
*Set length of averaging window.*
- void **set\_averaging\_flag** (bool flag)
- void **perform\_pos\_averaging** ()
- 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].*
- double **get\_avg\_latitude** () const  
*Get RX position averaged Latitude WGS84 [deg].*
- double **get\_avg\_longitude** () const  
*Get RX position averaged Longitude WGS84 [deg].*
- double **get\_avg\_height** () const  
*Get RX position averaged height WGS84 [m].*
- int **get\_num\_valid\_observations** () const  
*Get the number of valid pseudorange observations (valid satellites)*
- bool **is\_pre\_2009** () const
- bool **is\_valid\_position** () const
- bool **is\_averaging** () const
- virtual double **get\_hdop** () const =0
- virtual double **get\_vdop** () const =0
- virtual double **get\_pdop** () const =0
- virtual double **get\_gdop** () const =0

### 10.271.1 Detailed Description

Base class for a PVT solution.

Definition at line 35 of file pvt\_solution.h.

### 10.271.2 Member Function Documentation

#### 10.271.2.1 get\_avg\_height()

```
double Pvt_Solution::get_avg_height ( ) const
```

Get RX position averaged height WGS84 [m].

#### 10.271.2.2 get\_avg\_latitude()

```
double Pvt_Solution::get_avg_latitude ( ) const
```

Get RX position averaged Latitude WGS84 [deg].

#### 10.271.2.3 get\_avg\_longitude()

```
double Pvt_Solution::get_avg_longitude ( ) const
```

Get RX position averaged Longitude WGS84 [deg].

#### 10.271.2.4 get\_clock\_drift\_ppm()

```
double Pvt_Solution::get_clock_drift_ppm ( ) const
```

Get the Rx clock drift [ppm].

#### 10.271.2.5 get\_course\_over\_ground()

```
double Pvt_Solution::get_course_over_ground ( ) const
```

Get RX course over ground [deg].

**10.271.2.6 get\_height()**

```
double Pvt_Solution::get_height ( ) const
```

Get RX position height WGS84 [m].

**10.271.2.7 get\_latitude()**

```
double Pvt_Solution::get_latitude ( ) const
```

Get RX position Latitude WGS84 [deg].

**10.271.2.8 get\_longitude()**

```
double Pvt_Solution::get_longitude ( ) const
```

Get RX position Longitude WGS84 [deg].

**10.271.2.9 get\_num\_valid\_observations()**

```
int Pvt_Solution::get_num_valid_observations ( ) const
```

Get the number of valid pseudorange observations (valid satellites)

**10.271.2.10 get\_speed\_over\_ground()**

```
double Pvt_Solution::get_speed_over_ground ( ) const
```

Get RX speed over ground [m/s].

**10.271.2.11 get\_time\_offset\_s()**

```
double Pvt_Solution::get_time_offset_s ( ) const
```

Get RX time offset [s].

**10.271.2.12 set\_averaging\_depth()**

```
void Pvt_Solution::set_averaging_depth (
    int depth )
```

Set length of averaging window.

**10.271.2.13 set\_clock\_drift\_ppm()**

```
void Pvt_Solution::set_clock_drift_ppm (
    double clock_drift_ppm )
```

Set the Rx clock drift [ppm].

**10.271.2.14 set\_course\_over\_ground()**

```
void Pvt_Solution::set_course_over_ground (
    double cog_deg )
```

Set RX course over ground [deg].

**10.271.2.15 set\_num\_valid\_observations()**

```
void Pvt_Solution::set_num_valid_observations (
    int num )
```

Set the number of valid pseudorange observations (valid satellites)

**10.271.2.16 set\_pre\_2009\_file()**

```
void Pvt_Solution::set_pre_2009_file (
    bool pre_2009_file )
```

Flag for the week rollover computation in post processing mode for signals older than 2009.

**10.271.2.17 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].

#### 10.271.2.18 `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].

#### 10.271.2.19 `set_speed_over_ground()`

```
void Pvt_Solution::set_speed_over_ground (
    double speed_m_s )
```

Set RX speed over ground [m/s].

#### 10.271.2.20 `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:

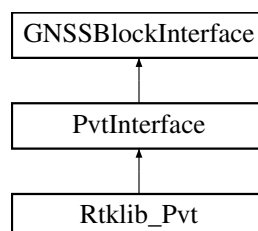
- [pvt\\_solution.h](#)

## 10.272 `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\\_Ephemeris](#) > **get\_gps\_ephemeris** () const =0
- virtual std::map< int, [Galileo\\_Ephemeris](#) > **get\_galileo\_ephemeris** () const =0
- virtual std::map< int, [Gps\\_Almanac](#) > **get\_gps\_almanac** () const =0
- virtual std::map< int, [Galileo\\_Almanac](#) > **get\_galileo\_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

### 10.272.1 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 48 of file pvt\_interface.h.

The documentation for this class was generated from the following file:

- [pvt\\_interface.h](#)

## 10.273 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]

### 10.273.1 Detailed Description

Definition at line 1202 of file rtklib.h.

The documentation for this struct was generated from the following file:

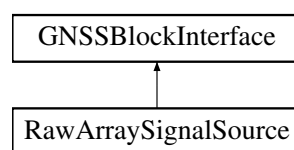
- [rtklib.h](#)

## 10.274 RawArraySignalSource Class Reference

This class reads samples from a GN3S USB dongle, a 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)
- std::string **role** () override
- std::string **implementation** () override
  - Returns "RawArraySignalSource".*
- 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

### 10.274.1 Detailed Description

This class reads samples from a GN3S USB dongle, a RF front-end signal sampler.

Definition at line 42 of file `raw_array_signal_source.h`.

### 10.274.2 Member Function Documentation

#### 10.274.2.1 implementation()

```
std::string RawArraySignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "RawArraySignalSource".

Implements [GNSSBlockInterface](#).

Definition at line 59 of file `raw_array_signal_source.h`.

The documentation for this class was generated from the following file:

- [raw\\_array\\_signal\\_source.h](#)

## 10.275 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](#) (int version=0, const std::string &base\_path=".", const std::string &base\_name="-")  
*Constructor. Creates GNSS Navigation and Observables RINEX files.*
- [~Rinex\\_Printer](#) ()  
*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, int type\_of\_rx, 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. The meaning of type\_of\_rx is as follows:*
- void [log\\_rinex\\_nav\\_gps\\_nav](#) (int type\_of\_rx, const std::map< int32\_t, [Gps\\_Ephemeris](#) > &new\_eph)  
*Print RINEX annotation for GPS NAV message.*
- void [log\\_rinex\\_nav\\_gps\\_cnav](#) (int type\_of\_rx, const std::map< int32\_t, [Gps\\_CNAV\\_Ephemeris](#) > &new\_↵  
cnav\_eph)  
*Print RINEX annotation for GPS CNAV message.*
- void [log\\_rinex\\_nav\\_gal\\_nav](#) (int type\_of\_rx, const std::map< int32\_t, [Galileo\\_Ephemeris](#) > &new\_gal\_eph)  
*Print RINEX annotation for Galileo NAV message.*

- void [log\\_rinex\\_nav\\_glo\\_gnav](#) (int type\_of\_rx, const std::map< int32\_t, [Glonass\\_Gnav\\_Ephemeris](#) > &new←\_glo\_eph)  
*Print RINEX annotation for Glonass GNAV message.*
- void [log\\_rinex\\_nav\\_bds\\_dnav](#) (int type\_of\_rx, const std::map< int32\_t, [Beidou\\_Dnav\\_Ephemeris](#) > &new←\_bds\_eph)  
*Print RINEX annotation for BeiDou DNAV message.*
- void [set\\_pre\\_2009\\_file](#) (bool pre\_2009\_file)  
*Set processing for signals older than 2009.*
- 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.*

### 10.275.1 Detailed Description

Class that handles the generation of Receiver INdependent EXchange format (RINEX) files.

Definition at line 82 of file rinex\_printer.h.

### 10.275.2 Constructor & Destructor Documentation

#### 10.275.2.1 Rinex\_Printer()

```
Rinex_Printer::Rinex_Printer (
    int version = 0,
    const std::string & base_path = ".",
    const std::string & base_name = "-" ) [explicit]
```

Constructor. Creates GNSS Navigation and Observables RINEX files.

#### 10.275.2.2 ~Rinex\_Printer()

```
Rinex_Printer::~~Rinex_Printer ( )
```

Destructor. Removes created files if empty.

### 10.275.3 Member Function Documentation

### 10.275.3.1 get\_navfilename()

```
std::vector<std::string> Rinex_Printer::get_navfilename ( ) const [inline]
```

Returns name of RINEX navigation file(s)

Definition at line 219 of file rinex\_printer.h.

### 10.275.3.2 get\_obsfilename()

```
std::string Rinex_Printer::get_obsfilename ( ) const [inline]
```

Returns name of RINEX observation file.

Definition at line 227 of file rinex\_printer.h.

### 10.275.3.3 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 211 of file rinex\_printer.h.

### 10.275.3.4 log\_rinex\_nav\_bds\_dnav()

```
void Rinex_Printer::log_rinex_nav_bds_dnav (
    int type_of_rx,
    const std::map< int32_t, Beidou_Dnav_Ephemeris > & new_bds_eph )
```

Print RINEX annotation for BeiDou DNAV message.

### 10.275.3.5 log\_rinex\_nav\_gal\_nav()

```
void Rinex_Printer::log_rinex_nav_gal_nav (
    int type_of_rx,
    const std::map< int32_t, Galileo_Ephemeris > & new_gal_eph )
```

Print RINEX annotation for Galileo NAV message.

**10.275.3.6 log\_rinex\_nav\_glo\_gnav()**

```
void Rinex_Printer::log_rinex_nav_glo_gnav (
    int type_of_rx,
    const std::map< int32_t, Glonass_Gnav_Ephemeris > & new_glo_eph )
```

Print RINEX annotation for Glonass GNAV message.

**10.275.3.7 log\_rinex\_nav\_gps\_cnav()**

```
void Rinex_Printer::log_rinex_nav_gps_cnav (
    int type_of_rx,
    const std::map< int32_t, Gps_CNAV_Ephemeris > & new_cnav_eph )
```

Print RINEX annotation for GPS CNAV message.

**10.275.3.8 log\_rinex\_nav\_gps\_nav()**

```
void Rinex_Printer::log_rinex_nav_gps_nav (
    int type_of_rx,
    const std::map< int32_t, Gps_Ephemeris > & new_eph )
```

Print RINEX annotation for GPS NAV message.

**10.275.3.9 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,
    int type_of_rx,
    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. The meaning of type\_of\_rx is as follows:

type_of_rx	Signals ----
0	Unknown
1	GPS L1 C/A
2	GPS L2C
3	GPS L5
4	Galileo E1B
5	Galileo E5a
6	Galileo E5b
7	GPS L1 C/A + GPS L2C

type_of_rx	Signals ----
8	GPS L1 C/A + GPS L5
9	GPS L1 C/A + Galileo E1B
10	GPS L1 C/A + Galileo E5a
11	GPS L1 C/A + Galileo E5b
12	Galileo E1B + GPS L2C
13	Galileo E5a + GPS L5
14	Galileo E1B + Galileo E5a
15	Galileo E1B + Galileo E5b
16	GPS L2C + GPS L5
17	GPS L2C + Galileo E5a
20	GPS L5 + Galileo E5b
21	GPS L1 C/A + Galileo E1B + GPS L2C
22	GPS L1 C/A + Galileo E1B + GPS L5
23	GLONASS L1 C/A
24	GLONASS L2 C/A
25	GLONASS L1 C/A + GLONASS L2 C/A
26	GPS L1 C/A + GLONASS L1 C/A
27	Galileo E1B + GLONASS L1 C/A
28	GPS L2C + GLONASS L1 C/A
29	GPS L1 C/A + GLONASS L2 C/A
30	Galileo E1B + GLONASS L2 C/A
31	GPS L2C + GLONASS L2 C/A
32	GPS L1 C/A + Galileo E1B + GPS L5 + Galileo E5a
33	GPS L1 C/A + Galileo E1B + Galileo E5a
100	Galileo E6B
101	Galileo E1B + Galileo E6B
102	Galileo E5a + Galileo E6B
103	Galileo E5b + Galileo E6B
104	Galileo E1B + Galileo E5a + Galileo E6B
105	Galileo E1B + Galileo E5b + Galileo E6B
106	GPS L1 C/A + Galileo E1B + Galileo E6B
500	BeiDou B1I
501	BeiDou B1I + GPS L1 C/A
502	BeiDou B1I + Galileo E1B
503	BeiDou B1I + GLONASS L1 C/A
504	BeiDou B1I + GPS L1 C/A + Galileo E1B
505	BeiDou B1I + GPS L1 C/A + GLONASS L1 C/A + Galileo E1B
506	BeiDou B1I + BeiDou B3I
600	BeiDou B3I
601	BeiDou B3I + GPS L2C
602	BeiDou B3I + GLONASS L2 C/A
603	BeiDou B3I + GPS L2C + GLONASS L2 C/A
604	BeiDou B3I + GPS L1 C/A
605	BeiDou B3I + Galileo E1B
606	BeiDou B3I + GLONASS L1 C/A
607	BeiDou B3I + GPS L1 C/A + Galileo E1B
608	BeiDou B3I + GPS L1 C/A + Galileo E1B + BeiDou B1I
609	BeiDou B3I + GPS L1 C/A + Galileo E1B + GLONASS L1 C/A
610	BeiDou B3I + GPS L1 C/A + Galileo E1B + GLONASS L1 C/A + BeiDou B1I
1000	GPS L1 C/A + GPS L2C + GPS L5
1001	GPS L1 C/A + Galileo E1B + GPS L2C + GPS L5 + Galileo E5a

### 10.275.3.10 set\_pre\_2009\_file()

```
void Rinex_Printer::set_pre_2009_file (
    bool pre_2009_file )
```

Set processing for signals older than 2009.

The documentation for this class was generated from the following file:

- [rinex\\_printer.h](#)

## 10.276 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)*

- `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 an 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.*

### 10.276.1 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

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):

length	data message	parity	preamble	000000
8	24	8	6	10

(C) Carles Fernandez-Prades, 2015. cfernandez(at)cttc.es

Definition at line 90 of file rtcM.h.

### 10.276.2 Constructor & Destructor Documentation

#### 10.276.2.1 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>.

### 10.276.3 Member Function Documentation

#### 10.276.3.1 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.

**10.276.3.2 bin\_to\_double()**

```
double RtcM::bin_to_double (
    const std::string & s ) const
```

Returns double from a string of binary symbols.

**10.276.3.3 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.

**10.276.3.4 bin\_to\_uint()**

```
uint32_t RtcM::bin_to_uint (
    const std::string & s ) const
```

Returns an uint32\_t from a string of binary symbols.

**10.276.3.5 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.

**10.276.3.6 check\_CRC()**

```
bool RtcM::check_CRC (
    const std::string & message ) const
```

Checks that the CRC of a RTCM package is correct.

**10.276.3.7 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.

**10.276.3.8 hex\_to\_int()**

```
int64_t RtcM::hex_to_int (
    const std::string & s ) const
```

Returns a `int64_t` from a string of hexadecimal symbols.

**10.276.3.9 hex\_to\_uint()**

```
uint64_t RtcM::hex_to_uint (
    const std::string & s ) const
```

Returns an `uint64_t` from a string of hexadecimal symbols.

**10.276.3.10 is\_server\_running()**

```
bool RtcM::is_server_running ( ) const
```

Returns true if the server is running, false otherwise.

**10.276.3.11 lock\_time()** [1/4]

```
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.

**10.276.3.12 lock\_time()** [2/4]

```
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.

**10.276.3.13 lock\_time()** [3/4]

```
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.

**10.276.3.14 lock\_time()** [4/4]

```
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**

<i>eph</i>	GLONASS GNAV Broadcast Ephemeris
<i>obs_time</i>	Time of observation at the moment of printing
<i>observables</i>	Set of observables as defined by the platform

**Returns**

Returns the time period in which GLONASS signals have been continually tracked.

**10.276.3.15 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)

### 10.276.3.16 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)

### 10.276.3.17 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)

### 10.276.3.18 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)

### 10.276.3.19 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)

### 10.276.3.20 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)

### 10.276.3.21 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)

### 10.276.3.22 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)

### 10.276.3.23 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)

### 10.276.3.24 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)

### 10.276.3.25 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)

### 10.276.3.26 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)

### 10.276.3.27 print\_MT1005\_test()

```
std::string RtcM::print_MT1005_test ( )
```

For testing purposes.

### 10.276.3.28 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)

### 10.276.3.29 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)

### 10.276.3.30 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

<i>glonass_gnav_eph</i>	GLONASS GNAV Broadcast Ephemeris
<i>obs_time</i>	Time of observation at the moment of printing
<i>observables</i>	Set of observables as defined by the platform

#### Returns

string with message contents

### 10.276.3.31 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

<i>glonass_gnav_eph</i>	GLONASS GNAV Broadcast Ephemeris
<i>obs_time</i>	Time of observation at the moment of printing
<i>observables</i>	Set of observables as defined by the platform

**Returns**

string with message contents

**10.276.3.32 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**

<i>glonass_gnav_eph</i>	GLONASS GNAV Broadcast Ephemeris
<i>obs_time</i>	Time of observation at the moment of printing
<i>observables</i>	Set of observables as defined by the platform

**Returns**

string with message contents

**10.276.3.33 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

<i>glonass_gnav_eph</i>	GLONASS GNAV Broadcast Ephemeris
<i>obs_time</i>	Time of observation at the moment of printing
<i>observables</i>	Set of observables as defined by the platform

## Returns

string with message contents

## 10.276.3.34 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.

## 10.276.3.35 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

<i>glonass_gnav_eph</i>	GLONASS GNAV Broadcast Ephemeris
<i>glonass_gnav_utc_model</i>	GLONASS GNAV Clock Information

## Returns

Returns message type as a string type

## 10.276.3.36 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)

#### 10.276.3.37 print\_MT1045()

```
std::string RtcM::print_MT1045 (
    const Galileo_Ephemeris & gal_eph )
```

Prints message type 1045 (Galileo Ephemeris), should be broadcast every 2 minutes.

#### 10.276.3.38 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.

#### 10.276.3.39 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.

#### 10.276.3.40 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

<i>message</i>	Message to read as a string type
<i>glonass_gnav_eph</i>	GLONASS GNAV Broadcast Ephemeris
<i>glonass_gnav_utc_model</i>	GLONASS GNAV Clock Information

## Returns

Returns 1 if anything goes wrong, 0 otherwise.

**10.276.3.41 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.

**10.276.3.42 run\_server()**

```
void RtcM::run_server ( )
```

Starts running the server.

**10.276.3.43 send\_message()**

```
void RtcM::send_message (
    const std::string & msg )
```

Sends a message through the server to all connected clients.

**10.276.3.44 stop\_server()**

```
void RtcM::stop_server ( )
```

Stops the server.

The documentation for this class was generated from the following file:

- [rtcm.h](#)

## 10.277 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, bool time\_tag\_name=true, const std::string &base\_path=".")  
*Default constructor.*
- [~RtcM\\_Printer](#) ()  
*Default destructor.*
- void [Print\\_RtcM\\_Messages](#) (const [Rtklib\\_Solver](#) \*pvt\_solver, const std::map< int, [Gnss\\_Synchro](#) > &gnss\_observables\_map, double rx\_time, int32\_t type\_of\_rx, int32\_t rtcM\_MSM\_rate\_ms, int32\_t rtcM\_MT1019\_rate\_ms, int32\_t rtcM\_MT1020\_rate\_ms, int32\_t rtcM\_MT1045\_rate\_ms, int32\_t rtcM\_MT1077\_rate\_ms, int32\_t rtcM\_MT1097\_rate\_ms, 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.*
- std::string [print\\_MT1005\\_test](#) ()  
*For testing purposes.*

### 10.277.1 Detailed Description

This class provides a implementation of a subset of the RTCM Standard 10403.2 messages.

Definition at line 47 of file `rtcm_printer.h`.

### 10.277.2 Constructor & Destructor Documentation

#### 10.277.2.1 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,
    bool time_tag_name = true,
    const std::string & base_path = "." )
```

Default constructor.

### 10.277.2.2 ~Rtcn\_Printer()

```
Rtcn_Printer::~~Rtcn_Printer ( )
```

Default destructor.

## 10.277.3 Member Function Documentation

### 10.277.3.1 lock\_time()

```
uint32_t Rtcn_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 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

locked time during logging process

### 10.277.3.2 print\_MT1005\_test()

```
std::string Rtcn_Printer::print_MT1005_test ( )
```

For testing purposes.

### 10.277.3.3 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,
    int32_t type_of_rx,
    int32_t rtcm_MSM_rate_ms,
    int32_t rtcm_MT1019_rate_ms,
    int32_t rtcm_MT1020_rate_ms,
    int32_t rtcm_MT1045_rate_ms,
    int32_t rtcm_MT1077_rate_ms,
    int32_t rtcm_MT1097_rate_ms,
    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.

The documentation for this class was generated from the following file:

- [rtcm\\_printer.h](#)

## 10.278 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]

### 10.278.1 Detailed Description

Definition at line 876 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.279 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**
- [ambc\\_t](#) **ambc** [MAXSAT]
- [ssat\\_t](#) **ssat** [MAXSAT]
- int **neb**
- char **errbuf** [MAXERRMSG]
- [prcopt\\_t](#) **opt**

### 10.279.1 Detailed Description

Definition at line 1067 of file rtklib.h.

The documentation for this struct was generated from the following file:

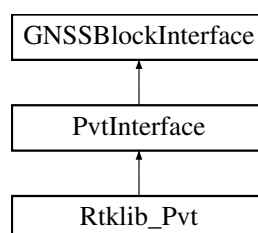
- [rtklib.h](#)

## 10.280 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\\_Ephemeris](#) > **get\_gps\_ephemeris** () const override
- std::map< int, [Galileo\\_Ephemeris](#) > **get\_galileo\_ephemeris** () const override
- std::map< int, [Gps\\_Almanac](#) > **get\_gps\_almanac** () const override
- std::map< int, [Galileo\\_Almanac](#) > **get\_galileo\_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

### 10.280.1 Detailed Description

This class implements a [PvtInterface](#) for the RTKLIB PVT block.

Definition at line 48 of file rtklib\_pvt.h.

### 10.280.2 Member Function Documentation

#### 10.280.2.1 implementation()

```
std::string Rtklib_Pvt::implementation ( ) [inline], [override], [virtual]
```

Returns "RTKLIB\_PVT".

Implements [GNSSBlockInterface](#).

Definition at line 64 of file rtklib\_pvt.h.

## 10.280.2.2 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 86 of file rtklib\_pvt.h.

The documentation for this class was generated from the following file:

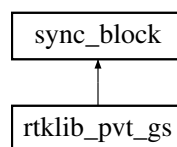
- [rtklib\\_pvt.h](#)

## 10.281 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](#) ()  
*Default destructor.*
- `std::map< int, Gps\_Ephemeris > get\_gps\_ephemeris\_map () const`  
*Get latest set of GPS ephemeris from PVT block.*
- `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\_Dnav\_Almanac > get\_beidou\_dnav\_almanac\_map () const`  
*Get latest set of BeiDou DNAV almanac from PVT block.*
- `void clear\_ephemeris ()`  
*Clear all ephemeris information and the almanacs for GPS and Galileo.*
- `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)`  
*PVT Signal Processing.*

## Friends

- `rtklib_pvt_gs_sptr rtklib_make_pvt_gs (uint32_t nchannels, const Pvt\_Conf &conf_, const rtk\_t &rtk)`

### 10.281.1 Detailed Description

This class implements a block that computes the PVT solution using the RTKLIB integrated library.

Definition at line 71 of file `rtklib_pvt_gs.h`.

### 10.281.2 Constructor & Destructor Documentation

#### 10.281.2.1 `~rtklib_pvt_gs()`

```
rtklib_pvt_gs::~~rtklib_pvt_gs ( )
```

Default destructor.

### 10.281.3 Member Function Documentation

#### 10.281.3.1 `clear_ephemeris()`

```
void rtklib_pvt_gs::clear_ephemeris ( )
```

Clear all ephemeris information and the almanacs for GPS and Galileo.

#### 10.281.3.2 `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.

#### 10.281.3.3 `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.

#### 10.281.3.4 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.

#### 10.281.3.5 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.

#### 10.281.3.6 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.

#### 10.281.3.7 get\_gps\_ephemeris\_map()

```
std::map<int, Gps_Ephemeris> rtklib_pvt_gs::get_gps_ephemeris_map ( ) const
```

Get latest set of GPS ephemeris from PVT block.

#### 10.281.3.8 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.

### 10.281.3.9 work()

```
int rtklib_pvt_gs::work (
    int noutput_items,
    gr_vector_const_void_star & input_items,
    gr_vector_void_star & output_items )
```

PVT Signal Processing.

The documentation for this class was generated from the following file:

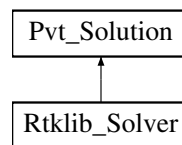
- [rtklib\\_pvt\\_gs.h](#)

## 10.282 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, int nchannels, const std::string &dump\_filename, bool flag\_dump\_to\_file, bool flag\_dump\_to\_mat)
- bool **get\_PVT** (const std::map< int, [Gnss\\_Synchro](#) > &gnss\_observables\_map, bool flag\_averaging)
- 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

### Public Attributes

- [sol\\_t](#) **pvt\_sol** {}
- std::array< [ssat\\_t](#), MAXSAT > **pvt\_ssat** {}
- std::map< int, [Galileo\\_Ephemeris](#) > [galileo\\_ephemeris\\_map](#)  
*Map storing new Galileo\_Ephemeris.*
- std::map< int, [Gps\\_Ephemeris](#) > [gps\\_ephemeris\\_map](#)  
*Map storing new GPS\_Ephemeris.*
- 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 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\_Iono gps_cnav_iono`
- `Gps\_CNAV\_Utc\_Model gps_cnav_utc_model`
- `Glonass\_Gnav\_Utc\_Model glonass_gnav_utc_model`
- *Map storing GLONASS GNAV UTC Model.*
- `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`
- `std::map< int, Beidou\_Dnav\_Almanac > beidou_dnav_almanac_map`

### 10.282.1 Detailed Description

This class implements a PVT solution based on RTKLIB.

Definition at line 75 of file `rtklib_solver.h`.

### 10.282.2 Member Data Documentation

#### 10.282.2.1 `beidou_dnav_ephemeris_map`

```
std::map<int, Beidou\_Dnav\_Ephemeris> Rtklib_Solver::beidou_dnav_ephemeris_map
```

Map storing new BeiDou DNAV Ephemeris.

Definition at line 96 of file `rtklib_solver.h`.

#### 10.282.2.2 `galileo_ephemeris_map`

```
std::map<int, Galileo\_Ephemeris> Rtklib_Solver::galileo_ephemeris_map
```

Map storing new [Galileo\\_Ephemeris](#).

Definition at line 92 of file `rtklib_solver.h`.

### 10.282.2.3 glonass\_gnav\_almanac

`Glonass_Gnav_Almanac` `Rtklib_Solver::glonass_gnav_almanac`

Map storing GLONASS GNAV Almanac Model.

Definition at line 110 of file `rtklib_solver.h`.

### 10.282.2.4 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 95 of file `rtklib_solver.h`.

### 10.282.2.5 glonass\_gnav\_utc\_model

`Glonass_Gnav_Utc_Model` `Rtklib_Solver::glonass_gnav_utc_model`

Map storing GLONASS GNAV UTC Model.

Definition at line 109 of file `rtklib_solver.h`.

### 10.282.2.6 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 94 of file `rtklib_solver.h`.

### 10.282.2.7 gps\_ephemeris\_map

`std::map<int, Gps_Ephemeris>` `Rtklib_Solver::gps_ephemeris_map`

Map storing new GPS\_Ephemeris.

Definition at line 93 of file `rtklib_solver.h`.

The documentation for this class was generated from the following file:

- [rtklib\\_solver.h](#)

## 10.283 Rtklib\_Solver\_Dump\_Reader Class Reference

### Public Member Functions

- bool **read\_binary\_obs** ()
- bool **restart** ()
- int64\_t **num\_epochs** ()
- bool **open\_obs\_file** (std::string out\_file)

### Public Attributes

- uint32\_t **TOW\_at\_current\_symbol\_ms**
- uint32\_t **week**
- double **RX\_time**
- double **clk\_offset\_s**
- double **rr** [6]
- double **qr** [6]
- double **latitude**
- double **longitude**
- double **height**
- uint8\_t **ns**
- uint8\_t **status**
- uint8\_t **type**
- float **AR\_ratio**
- float **AR\_thres**
- double **dop** [4]

### 10.283.1 Detailed Description

Definition at line 25 of file rtklib\_solver\_dump\_reader.h.

The documentation for this class was generated from the following file:

- [rtklib\\_solver\\_dump\\_reader.h](#)

## 10.284 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**

### 10.284.1 Detailed Description

Definition at line 1239 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.285 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

### 10.285.1 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.

The documentation for this class was generated from the following file:

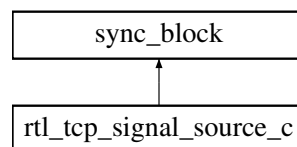
- [rtl\\_tcp\\_dongle\\_info.h](#)

## 10.286 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)

### 10.286.1 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:

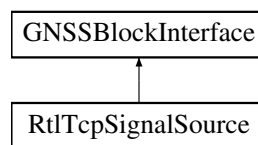
- [rtl\\_tcp\\_signal\\_source.c.h](#)

## 10.287 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)
- std::string **role** () override
- std::string **implementation** () override
 

*Returns "RtlTcp\_Signal\_Source".*
- 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

### 10.287.1 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.

### 10.287.2 Member Function Documentation

## 10.287.2.1 implementation()

```
std::string RtlTcpSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "RtlTcp\_Signal\_Source".

Implements [GNSSBlockInterface](#).

Definition at line 64 of file `rtl_tcp_signal_source.h`.

The documentation for this class was generated from the following file:

- [rtl\\_tcp\\_signal\\_source.h](#)

## 10.288 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<sup>2</sup>) (ECEF)*
- double [d\\_af0](#) {}  
*Satellite clock-offset (s)*
- double [d\\_af1](#) {}  
*Satellite drift (s/s)*

### 10.288.1 Detailed Description

This class stores SBAS SV ephemeris data.

Definition at line 33 of file sbas\_ephemeris.h.

### 10.288.2 Member Data Documentation

#### 10.288.2.1 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.

#### 10.288.2.2 d\_acc

```
double Sbas_Ephemeris::d_acc[3] {}
```

Satellite acceleration (m/s<sup>2</sup>) (ECEF)

Definition at line 45 of file sbas\_ephemeris.h.

#### 10.288.2.3 d\_af0

```
double Sbas_Ephemeris::d_af0 {}
```

Satellite clock-offset (s)

Definition at line 46 of file sbas\_ephemeris.h.

#### 10.288.2.4 d\_af1

```
double Sbas_Ephemeris::d_af1 {}
```

Satellite drift (s/s)

Definition at line 47 of file sbas\_ephemeris.h.

#### 10.288.2.5 d\_pos

```
double Sbas_Ephemeris::d_pos[3] {}
```

Satellite position (m) (ECEF)

Definition at line 43 of file sbas\_ephemeris.h.

#### 10.288.2.6 d\_tof

```
double Sbas_Ephemeris::d_tof {}
```

Time of message frame (GPST)

Definition at line 40 of file sbas\_ephemeris.h.

#### 10.288.2.7 d\_vel

```
double Sbas_Ephemeris::d_vel[3] {}
```

Satellite velocity (m/s) (ECEF)

Definition at line 44 of file sbas\_ephemeris.h.

#### 10.288.2.8 i\_prn

```
int Sbas_Ephemeris::i_prn {}
```

PRN number.

Definition at line 38 of file sbas\_ephemeris.h.

#### 10.288.2.9 i\_sv\_ura

```
int Sbas_Ephemeris::i_sv_ura {}
```

SV accuracy (URA index), not standardized.

Definition at line 41 of file sbas\_ephemeris.h.

### 10.288.2.10 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:

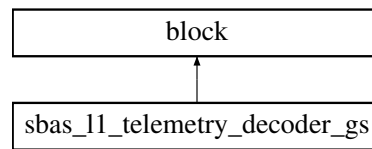
- [sbas\\_ephemeris.h](#)

## 10.289 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)  
*Set satellite PRN.*
- void [set\\_channel](#) (int32\_t channel)  
*Set receiver's channel.*
- void [reset](#) ()
- int [general\\_work](#) (int noutput\_items, gr\_vector\_int &ninput\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)  
*This is where all signal processing takes place.*

### Friends

- `sbas_l1_telemetry_decoder_gs_sptr` [sbas\\_l1\\_make\\_telemetry\\_decoder\\_gs](#) (const [Gnss\\_Satellite](#) &satellite, bool dump)

### 10.289.1 Detailed Description

This class implements a block that decodes the SBAS integrity and corrections data defined in RTCA MOPS DO-229.

Definition at line 54 of file `sbas_l1_telemetry_decoder_gs.h`.

## 10.289.2 Member Function Documentation

### 10.289.2.1 `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 )
```

This is where all signal processing takes place.

### 10.289.2.2 `set_channel()`

```
void sbas_l1_telemetry_decoder_gs::set_channel (
    int32_t channel )
```

Set receiver's channel.

### 10.289.2.3 `set_satellite()`

```
void sbas_l1_telemetry_decoder_gs::set_satellite (
    const Gnss_Satellite & satellite )
```

Set satellite PRN.

The documentation for this class was generated from the following file:

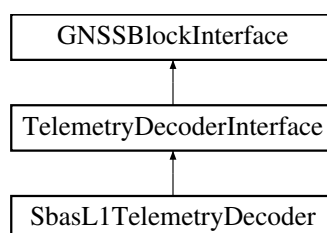
- [sbas\\_l1\\_telemetry\\_decoder\\_gs.h](#)

## 10.290 SbasL1TelemetryDecoder Class Reference

This class implements a NAV data decoder for SBAS frames in L1 radio link.

```
#include <sbas_l1_telemetry_decoder.h>
```

Inheritance diagram for SbasL1TelemetryDecoder:



## Public Member Functions

- **SbasL1TelemetryDecoder** (const [ConfigurationInterface](#) \*configuration, const std::string &role, unsigned int in\_streams, unsigned int out\_streams)
- std::string **role** () override
- std::string **implementation** () override  
*Returns "SBAS\_L1\_Telemetry\_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
- void **set\_channel** (int channel) override
- void **reset** () override
- size\_t **item\_size** () override

### 10.290.1 Detailed Description

This class implements a NAV data decoder for SBAS frames in L1 radio link.

Definition at line 41 of file `sbas_l1_telemetry_decoder.h`.

### 10.290.2 Member Function Documentation

#### 10.290.2.1 implementation()

```
std::string SbasL1TelemetryDecoder::implementation ( ) [inline], [override], [virtual]
```

Returns "SBAS\_L1\_Telemetry\_Decoder".

Implements [GNSSBlockInterface](#).

Definition at line 60 of file `sbas_l1_telemetry_decoder.h`.

The documentation for this class was generated from the following file:

- [sbas\\_l1\\_telemetry\\_decoder.h](#)

## 10.291 sbs\_t Struct Reference

### Public Attributes

- int **n**
- int **nmax**
- [sbsmsg\\_t](#) \* **msgs**

### 10.291.1 Detailed Description

Definition at line 575 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.292 sbsfcrr\_t Struct Reference

### Public Attributes

- [gtime\\_t](#) **t0**
- double **prc**
- double **rrc**
- double **dt**
- int **iodf**
- short **udre**
- short **ai**

### 10.292.1 Detailed Description

Definition at line 582 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.293 sbsignp\_t Struct Reference

### Public Attributes

- [gtime\\_t](#) **t0**
- short **lat**
- short **lon**
- short **give**
- float **delay**

### 10.293.1 Detailed Description

Definition at line 621 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.294 sbsigband\_t Struct Reference

### Public Attributes

- short **x**
- const short \* **y**
- unsigned char **bits**
- unsigned char **bite**

### 10.294.1 Detailed Description

Definition at line 630 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.295 sbsion\_t Struct Reference

### Public Attributes

- int **iodi**
- int **nigp**
- [sbsigp\\_t](#) **igp** [MAXNIGP]

### 10.295.1 Detailed Description

Definition at line 639 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.296 sbslcorr\_t Struct Reference

### Public Attributes

- [gtime\\_t](#) **t0**
- int **iode**
- double **dpos** [3]
- double **dvel** [3]
- double **daf0**
- double **daf1**

### 10.296.1 Detailed Description

Definition at line 594 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.297 sbsmsg\_t Struct Reference

### Public Attributes

- int **week**
- int **tow**
- int **prn**
- unsigned char **msg** [29]

### 10.297.1 Detailed Description

Definition at line 567 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.298 sbssat\_t Struct Reference

### Public Attributes

- int **iodp**
- int **nsat**
- int **tlat**
- [sbssatp\\_t](#) **sat** [MAXSAT]

### 10.298.1 Detailed Description

Definition at line 612 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.299 sbssatp\_t Struct Reference

### Public Attributes

- int **sat**
- [sbsfcorr\\_t](#) **fcorr**
- [sbslcorr\\_t](#) **lcorr**

### 10.299.1 Detailed Description

Definition at line 604 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.300 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**

### 10.300.1 Detailed Description

Definition at line 502 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.301 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*

### 10.301.1 Detailed Description

This class implements serialization and deserialization of [Gnss\\_Synchro](#) objects using Protocol Buffers.

Definition at line 32 of file `serdes_gnss_synchro.h`.

### 10.301.2 Constructor & Destructor Documentation

#### 10.301.2.1 Serdes\_Gnss\_Synchro() [1/2]

```
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`.

#### 10.301.2.2 Serdes\_Gnss\_Synchro() [2/2]

```
Serdes_Gnss_Synchro::Serdes_Gnss_Synchro (
    Serdes\_Gnss\_Synchro && other ) [inline], [noexcept]
```

*< Move constructor*

Definition at line 58 of file `serdes_gnss_synchro.h`.

### 10.301.3 Member Function Documentation

#### 10.301.3.1 createProtobuffer()

```
std::string Serdes_Gnss_Synchro::createProtobuffer (
    const std::vector< Gnss\_Synchro > & vgs ) [inline]
```

## Parameters

<code>vgs</code>	Serialization into a string
------------------	-----------------------------

Definition at line 72 of file `serdes_gnss_synchro.h`.

### 10.301.3.2 `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`.

### 10.301.3.3 `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`.

### 10.301.3.4 `readProtobuffer()`

```
std::vector<Gnss_Synchro> Serdes_Gnss_Synchro::readProtobuffer (
    const gnss_sdr::Observables & obs ) const [inline]
```

< Deserialization

Definition at line 120 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::Channel_ID`, `Gnss_Synchro::CN0_dB_hz`, `Gnss_Synchro::correlation_length_ms`, `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::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](#)

## 10.302 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  
    < Copy constructor
- [Serdes\\_Monitor\\_Pvt](#) & operator= (const [Serdes\\_Monitor\\_Pvt](#) &rhs) noexcept  
    < Copy assignment operator
- [Serdes\\_Monitor\\_Pvt](#) ([Serdes\\_Monitor\\_Pvt](#) &&other) noexcept  
    < Move constructor
- [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

### 10.302.1 Detailed Description

This class implements serialization and deserialization of [Monitor\\_Pvt](#) objects using Protocol Buffers.

Definition at line 37 of file `serdes_monitor_pvt.h`.

### 10.302.2 Constructor & Destructor Documentation

#### 10.302.2.1 Serdes\_Monitor\_Pvt() [1/2]

```
Serdes_Monitor_Pvt::Serdes_Monitor_Pvt (
    const Serdes\_Monitor\_Pvt & other ) [inline], [noexcept]
```

< Copy constructor

Definition at line 52 of file `serdes_monitor_pvt.h`.

#### 10.302.2.2 Serdes\_Monitor\_Pvt() [2/2]

```
Serdes_Monitor_Pvt::Serdes_Monitor_Pvt (
    Serdes\_Monitor\_Pvt && other ) [inline], [noexcept]
```

< Move constructor

Definition at line 63 of file `serdes_monitor_pvt.h`.

### 10.302.3 Member Function Documentation

#### 10.302.3.1 createProtobuffer()

```
std::string Serdes_Monitor_Pvt::createProtobuffer (  
    const Monitor_Pvt *const monitor ) [inline]
```

## Parameters

<i>monitor</i>	Serialization into a string
----------------	-----------------------------

Definition at line 77 of file serdes\_monitor\_pvt.h.

## 10.302.3.2 operator=() [1/2]

```
Serdes_Monitor_Pvt& Serdes_Monitor_Pvt::operator= (
    const Serdes_Monitor_Pvt & rhs ) [inline], [noexcept]
```

< Copy assignment operator

Definition at line 57 of file serdes\_monitor\_pvt.h.

## 10.302.3.3 operator=() [2/2]

```
Serdes_Monitor_Pvt& Serdes_Monitor_Pvt::operator= (
    Serdes_Monitor_Pvt && other ) [inline], [noexcept]
```

< Move assignment operator

Definition at line 68 of file serdes\_monitor\_pvt.h.

## 10.302.3.4 readProtobuffer()

```
Monitor_Pvt Serdes_Monitor_Pvt::readProtobuffer (
    const gnss_sdr::MonitorPvt & mon ) const [inline]
```

< Deserialization

Definition at line 117 of file serdes\_monitor\_pvt.h.

The documentation for this class was generated from the following file:

- [serdes\\_monitor\\_pvt.h](#)

## 10.303 serial\_t Struct Reference

## Public Attributes

- dev\_t **dev**
- int **error**

### 10.303.1 Detailed Description

Definition at line 1111 of file rtklib.h.

The documentation for this struct was generated from the following file:

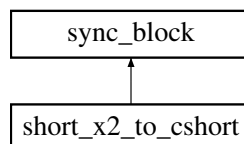
- [rtklib.h](#)

## 10.304 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 ()`

### 10.304.1 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:

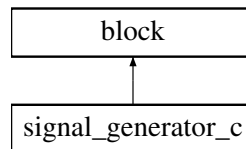
- [short\\_x2\\_to\\_cshort.h](#)

## 10.305 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 > &C/N0\_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.*

### 10.305.1 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.

### 10.305.2 Friends And Related Function Documentation

### 10.305.2.1 signal\_make\_generator\_c

```
signal_generator_c_sptr signal_make_generator_c (
    const std::vector< std::string > & signall,
    const std::vector< std::string > & system,
    const std::vector< unsigned int > & PRN,
    const std::vector< float > & CNO_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:

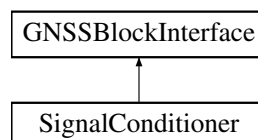
- [signal\\_generator\\_c.h](#)

## 10.306 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** ()

### 10.306.1 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`.

### 10.306.2 Constructor & Destructor Documentation

#### 10.306.2.1 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.

#### 10.306.2.2 ~SignalConditioner()

```
SignalConditioner::~SignalConditioner ( ) [default]
```

Destructor.

### 10.306.3 Member Function Documentation

#### 10.306.3.1 implementation()

```
std::string SignalConditioner::implementation ( ) [inline], [override], [virtual]
```

Returns "Signal\_Conditioner".

Implements [GNSSBlockInterface](#).

Definition at line 58 of file `signal_conditioner.h`.

The documentation for this class was generated from the following file:

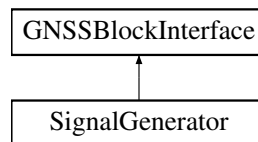
- [signal\\_conditioner.h](#)

## 10.307 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

### 10.307.1 Detailed Description

This class generates synthesized GNSS signal.

Definition at line 39 of file signal\_generator.h.

### 10.307.2 Member Function Documentation

#### 10.307.2.1 implementation()

```
std::string SignalGenerator::implementation ( ) [inline], [override], [virtual]
```

Returns "GNSSSignalGenerator".

Implements [GNSSBlockInterface](#).

Definition at line 56 of file signal\_generator.h.

The documentation for this class was generated from the following file:

- [signal\\_generator.h](#)

## 10.308 snrmask\_t Struct Reference

### Public Attributes

- int **ena** [2]
- double **mask** [[NFREQ](#)][9]

### 10.308.1 Detailed Description

Definition at line 937 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.309 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**

### 10.309.1 Detailed Description

Definition at line 821 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.310 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**

### 10.310.1 Detailed Description

Definition at line 839 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.311 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**

### 10.311.1 Detailed Description

Definition at line 1008 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.312 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**

### 10.312.1 Detailed Description

Definition at line 852 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.313 solstatbuf\_t Struct Reference

### Public Attributes

- int **n**
- int **nmax**
- [solstat\\_t](#) \* **data**

### 10.313.1 Detailed Description

Definition at line 869 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.314 Spirent\_Motion\_Csv\_Dump\_Reader Class Reference

### Public Member Functions

- bool **read\_csv\_obs** ()
- bool **restart** ()
- int64\_t **num\_epochs** ()
- bool **open\_obs\_file** (std::string out\_file)
- void **close\_obs\_file** ()

## Public Attributes

- int **header\_lines**
- double **TOW\_ms**
- double **Pos\_X**
- double **Pos\_Y**
- double **Pos\_Z**
- double **Vel\_X**
- double **Vel\_Y**
- double **Vel\_Z**
- double **Acc\_X**
- double **Acc\_Y**
- double **Acc\_Z**
- double **Jerk\_X**
- double **Jerk\_Y**
- double **Jerk\_Z**
- double **Lat**
- double **Long**
- double **Height**
- double **Heading**
- double **Elevation**
- double **Bank**
- double **Ang\_vel\_X**
- double **Ang\_vel\_Y**
- double **Ang\_vel\_Z**
- double **Ang\_acc\_X**
- double **Ang\_acc\_Y**
- double **Ang\_acc\_Z**
- double **Ant1\_Pos\_X**
- double **Ant1\_Pos\_Y**
- double **Ant1\_Pos\_Z**
- double **Ant1\_Vel\_X**
- double **Ant1\_Vel\_Y**
- double **Ant1\_Vel\_Z**
- double **Ant1\_Acc\_X**
- double **Ant1\_Acc\_Y**
- double **Ant1\_Acc\_Z**
- double **Ant1\_Lat**
- double **Ant1\_Long**
- double **Ant1\_Height**
- double **Ant1\_DOP**

### 10.314.1 Detailed Description

Definition at line 25 of file `spirent_motion_csv_dump_reader.h`.

The documentation for this class was generated from the following file:

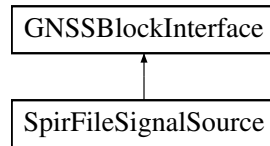
- [spirent\\_motion\\_csv\\_dump\\_reader.h](#)

## 10.315 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)
- std::string **role** () override
- std::string **implementation** () override  
Returns "*Spir\_File\_Signal\_Source*".
- 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

### 10.315.1 Detailed Description

Class that reads signals samples from a file and adapts it to a `SignalSourceInterface`.

Definition at line 48 of file `spir_file_signal_source.h`.

### 10.315.2 Member Function Documentation

#### 10.315.2.1 implementation()

```
std::string SpirFileSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "*Spir\_File\_Signal\_Source*".

Implements [GNSSBlockInterface](#).

Definition at line 65 of file `spir_file_signal_source.h`.

The documentation for this class was generated from the following file:

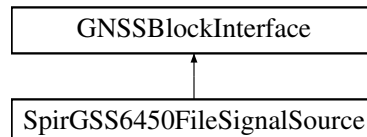
- [spir\\_file\\_signal\\_source.h](#)

## 10.316 SpirGSS6450FileSignalSource Class Reference

Class that reads signals samples from a file and adapts it to a `SignalSourceInterface`.

```
#include <spir_gss6450_file_signal_source.h>
```

Inheritance diagram for `SpirGSS6450FileSignalSource`:



### Public Member Functions

- **SpirGSS6450FileSignalSource** (const [ConfigurationInterface](#) \*configuration, const std::string &role, uint32\_t in\_streams, uint32\_t out\_streams, [Concurrent\\_Queue](#)< pmt::pmt\_t > \*queue)
- 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** (int RF\_channel) 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

### 10.316.1 Detailed Description

Class that reads signals samples from a file and adapts it to a `SignalSourceInterface`.

Definition at line 53 of file `spir_gss6450_file_signal_source.h`.

The documentation for this class was generated from the following file:

- [spir\\_gss6450\\_file\\_signal\\_source.h](#)

## 10.317 **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 **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]

### 10.317.1 Detailed Description

Definition at line 1031 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.318 **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**

### 10.318.1 Detailed Description

Definition at line 657 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.319 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 **itr**
- int **deltype**
- double **pos** [3]
- double **del** [3]
- double **hgt**

### 10.319.1 Detailed Description

Definition at line 803 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.320 stec\_t Struct Reference

### Public Attributes

- [gtime\\_t](#) **time**
- unsigned char **sat**
- double **ion**
- float **std**
- float **azel** [2]
- unsigned char **flag**

### 10.320.1 Detailed Description

Definition at line 723 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.321 stream\_cfg Struct Reference

### Public Attributes

- int64\_t **bw\_hz**
- int64\_t **fs\_hz**
- int64\_t **lo\_hz**
- const char \* **rfport**

### 10.321.1 Detailed Description

Definition at line 43 of file ad9361\_manager.h.

The documentation for this struct was generated from the following file:

- [ad9361\\_manager.h](#)

## 10.322 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](#)]

### 10.322.1 Detailed Description

Definition at line 1095 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.323 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)

### 10.323.1 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](#)

## 10.324 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\_)

### 10.324.1 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](#)

## 10.325 Tcp\_Packet\_Data Class Reference

Class that implements a TCP data packet.

```
#include <tcp_packet_data.h>
```

## Public Attributes

- float **proc\_pack\_code\_error**
- float **proc\_pack\_carr\_error**
- float **proc\_pack\_carrier\_doppler\_hz**

### 10.325.1 Detailed Description

Class that implements a TCP data packet.

Definition at line 30 of file tcp\_packet\_data.h.

The documentation for this class was generated from the following file:

- [tcp\\_packet\\_data.h](#)

## 10.326 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**

### 10.326.1 Detailed Description

Definition at line 1142 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.327 tcpcli\_t Struct Reference

### Public Attributes

- [tcp\\_t](#) **svr**
- int **toinact**
- int **tirecon**

### 10.327.1 Detailed Description

Definition at line 1162 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.328 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)

### 10.328.1 Detailed Description

Definition at line 41 of file tcp\_cmd\_interface.h.

### 10.328.2 Member Function Documentation

#### 10.328.2.1 get\_LLH()

```
std::array<float, 3> TcpCmdInterface::get_LLH ( ) const
```

gets the Latitude, Longitude and Altitude vector from the last TC command issued

#### 10.328.2.2 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](#)

## 10.329 tcpsvr\_t Struct Reference

### Public Attributes

- [tcp\\_t svr](#)
- [tcp\\_t cli](#) [MAXCLI]

### 10.329.1 Detailed Description

Definition at line 1155 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.330 `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**

### 10.330.1 Detailed Description

Definition at line 546 of file `rtklib.h`.

The documentation for this struct was generated from the following file:

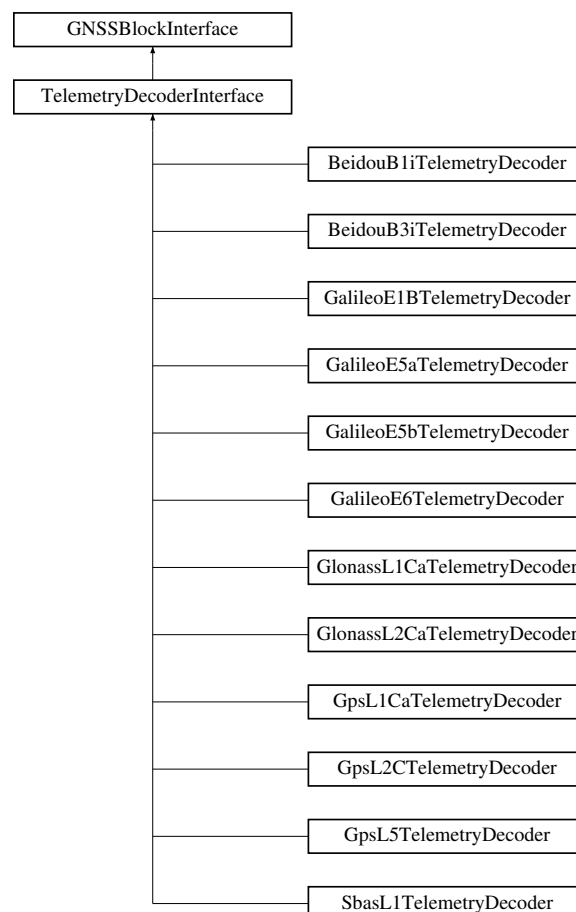
- [rtklib.h](#)

## 10.331 `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

### 10.331.1 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](#)

## 10.332 tle\_t Struct Reference

### Public Attributes

- int **n**
- int **nmax**
- [tled\\_t](#) \* **data**

### 10.332.1 Detailed Description

Definition at line 539 of file `rtklib.h`.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.333 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**

### 10.333.1 Detailed Description

Definition at line 516 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.334 Tlm\_Conf Class Reference

### Public Member Functions

- void **SetFromConfiguration** (const [ConfigurationInterface](#) \*configuration, const std::string &role)

### Public Attributes

- std::string **dump\_filename**
- bool **dump**
- bool **dump\_mat**
- bool **remove\_dat**

### 10.334.1 Detailed Description

Definition at line 30 of file tlm\_conf.h.

The documentation for this class was generated from the following file:

- [tlm\\_conf.h](#)

## 10.335 Tlm\_Dump\_Reader Class Reference

### Public Member Functions

- bool **read\_binary\_obs** ()
- bool **restart** ()
- int64\_t **num\_epochs** ()
- bool **open\_obs\_file** (std::string out\_file)

### Public Attributes

- double **TOW\_at\_current\_symbol**
- uint64\_t **Tracking\_sample\_counter**
- double **d\_TOW\_at\_Preamble**
- int32\_t **nav\_symbol**
- int32\_t **prn**

### 10.335.1 Detailed Description

Definition at line 25 of file `tlm_dump_reader.h`.

The documentation for this class was generated from the following file:

- [tlm\\_dump\\_reader.h](#)

## 10.336 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.*

### 10.336.1 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`.

### 10.336.2 Member Function Documentation

#### 10.336.2.1 [get\\_code\\_nco\(\)](#)

```
float Tracking_2nd_DLL_filter::get_code_nco (
    float DLL_discriminator )
```

Numerically controlled oscillator.

### 10.336.2.2 initialize()

```
void Tracking_2nd_DLL_filter::initialize ( )
```

Start tracking with acquisition information.

### 10.336.2.3 set\_DLL\_BW()

```
void Tracking_2nd_DLL_filter::set_DLL_BW (
    float dll_bw_hz )
```

Set DLL filter bandwidth [Hz].

### 10.336.2.4 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](#)

## 10.337 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)

### 10.337.1 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.

### 10.337.2 Member Function Documentation

#### 10.337.2.1 set\_pdi()

```
void Tracking_2nd_PLL_filter::set_pdi (
    float pdi_carr )
```

Set Summation interval for code [s].

#### 10.337.2.2 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](#)

## 10.338 Tracking\_Dump\_Reader Class Reference

### Public Member Functions

- bool **read\_binary\_obs** ()
- bool **restart** ()
- int64\_t **num\_epochs** ()
- bool **open\_obs\_file** (std::string out\_file)

## Public Attributes

- float **abs\_VE**
- float **abs\_E**
- float **abs\_P**
- float **abs\_L**
- float **abs\_VL**
- float **prompt\_I**
- float **prompt\_Q**
- uint64\_t **PRN\_start\_sample\_count**
- float **acc\_carrier\_phase\_rad**
- float **carrier\_doppler\_hz**
- float **carrier\_doppler\_rate\_hz\_s**
- float **code\_freq\_chips**
- float **code\_freq\_rate\_chips**
- float **carr\_error\_hz**
- float **carr\_error\_filt\_hz**
- float **code\_error\_chips**
- float **code\_error\_filt\_chips**
- float **CN0\_SNV\_dB\_Hz**
- float **carrier\_lock\_test**
- float **aux1**
- double **aux2**
- unsigned int **PRN**

### 10.338.1 Detailed Description

Definition at line 25 of file `tracking_dump_reader.h`.

The documentation for this class was generated from the following file:

- [tracking\\_dump\\_reader.h](#)

## 10.339 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)

### 10.339.1 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](#)

## 10.340 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)

### 10.340.1 Detailed Description

This class implements a generic 1st, 2nd or 3rd order loop filter.

Definition at line 35 of file tracking\_loop\_filter.h.

### 10.340.2 Constructor & Destructor Documentation

### 10.340.2.1 Tracking\_loop\_filter()

```
Tracking_loop_filter::Tracking_loop_filter (
    Tracking_loop_filter && ) [default]
```

Move operator.

## 10.340.3 Member Function Documentation

### 10.340.3.1 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](#)

## 10.341 Tracking\_True\_Obs\_Reader Class Reference

### Public Member Functions

- bool **read\_binary\_obs** ()
- bool **restart** ()
- int64\_t **num\_epochs** ()
- bool **open\_obs\_file** (std::string out\_file)
- void **close\_obs\_file** ()

### Public Attributes

- bool **d\_dump**
- double **signal\_timestamp\_s**
- double **acc\_carrier\_phase\_cycles**
- double **doppler\_l1\_hz**
- double **prn\_delay\_chips**
- double **tow**

### 10.341.1 Detailed Description

Definition at line 25 of file tracking\_true\_obs\_reader.h.

The documentation for this class was generated from the following file:

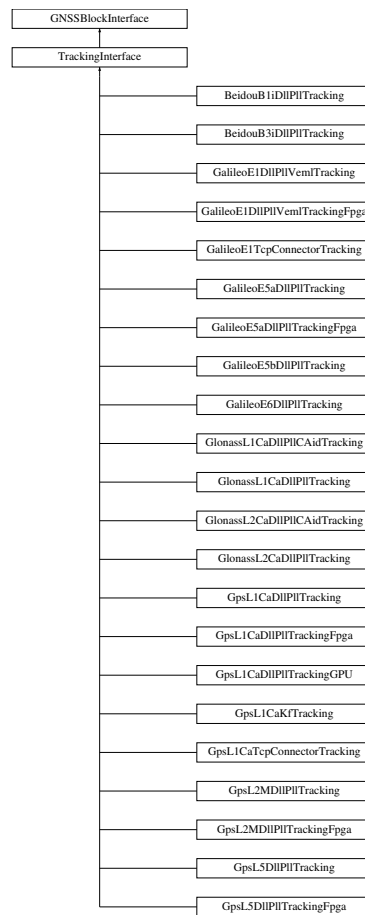
- [tracking\\_true\\_obs\\_reader.h](#)

## 10.342 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

### 10.342.1 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`.

The documentation for this class was generated from the following file:

- [tracking\\_interface.h](#)

## 10.343 trop\_t Struct Reference

### Public Attributes

- [gtime\\_t](#) **time**
- double **trp** [3]
- float **std** [3]

### 10.343.1 Detailed Description

Definition at line 734 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.344 True\_Observables\_Reader Class Reference

### Public Member Functions

- bool **read\_binary\_obs** ()
- bool **restart** ()
- int64\_t **num\_epochs** ()
- bool **open\_obs\_file** (std::string out\_file)

### Public Attributes

- double **gps\_time\_sec** [12]
- double **doppler\_l1\_hz** [12]
- double **acc\_carrier\_phase\_l1\_cycles** [12]
- double **dist\_m** [12]
- double **true\_dist\_m** [12]
- double **carrier\_phase\_l1\_cycles** [12]
- double **prn** [12]

### 10.344.1 Detailed Description

Definition at line 25 of file true\_observables\_reader.h.

The documentation for this class was generated from the following file:

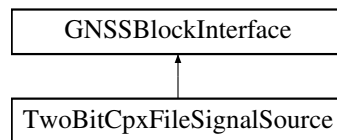
- [true\\_observables\\_reader.h](#)

## 10.345 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)
- std::string **role** () override
- std::string **implementation** () override  
Returns `"Two_Bit_Cpx_File_Signal_Source"`.
- 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

### 10.345.1 Detailed Description

Class that reads signals samples from a file and adapts it to a `SignalSourceInterface`.

Definition at line 47 of file `two_bit_cpx_file_signal_source.h`.

### 10.345.2 Member Function Documentation

#### 10.345.2.1 implementation()

```
std::string TwoBitCpxFileSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns `"Two_Bit_Cpx_File_Signal_Source"`.

Implements [GNSSBlockInterface](#).

Definition at line 65 of file `two_bit_cpx_file_signal_source.h`.

The documentation for this class was generated from the following file:

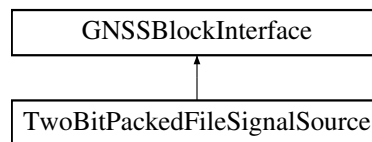
- [two\\_bit\\_cpx\\_file\\_signal\\_source.h](#)

## 10.346 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)
- std::string **role** () override
- std::string **implementation** () override  
Returns "Two\_Bit\_Packed\_File\_Signal\_Source".
- 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
- bool **big\_endian\_items** () const
- bool **big\_endian\_bytes** () const
- bool **is\_complex** () const
- bool **reverse\_interleaving** () const

### 10.346.1 Detailed Description

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.

Definition at line 49 of file two\_bit\_packed\_file\_signal\_source.h.

### 10.346.2 Member Function Documentation

## 10.346.2.1 implementation()

```
std::string TwoBitPackedFileSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "Two\_Bit\_Packed\_File\_Signal\_Source".

Implements [GNSSBlockInterface](#).

Definition at line 65 of file `two_bit_packed_file_signal_source.h`.

The documentation for this class was generated from the following file:

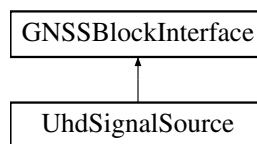
- [two\\_bit\\_packed\\_file\\_signal\\_source.h](#)

## 10.347 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)
- std::string **role** () override
- std::string **implementation** () override  
Returns "UHD\_Signal\_Source".
- 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

### 10.347.1 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`.

## 10.347.2 Member Function Documentation

### 10.347.2.1 implementation()

```
std::string UhdSignalSource::implementation ( ) [inline], [override], [virtual]
```

Returns "UHD\_Signal\_Source".

Implements [GNSSBlockInterface](#).

Definition at line 58 of file uhd\_signal\_source.h.

The documentation for this class was generated from the following file:

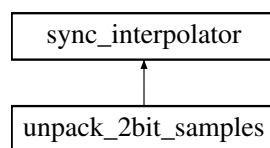
- [uhd\\_signal\\_source.h](#)

## 10.348 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)

### 10.348.1 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:

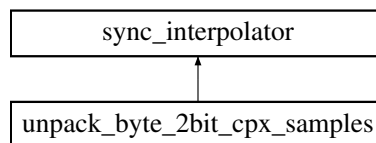
- [unpack\\_2bit\\_samples.h](#)

## 10.349 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 ()`

### 10.349.1 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:

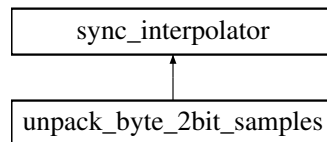
- [unpack\\_byte\\_2bit\\_cpx\\_samples.h](#)

## 10.350 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** ()

### 10.350.1 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:

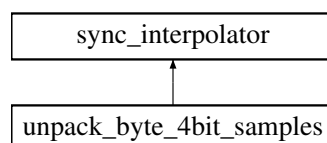
- [unpack\\_byte\\_2bit\\_samples.h](#)

## 10.351 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** ()

### 10.351.1 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 41 of file unpack\_byte\_4bit\_samples.h.

The documentation for this class was generated from the following file:

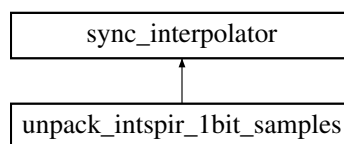
- [unpack\\_byte\\_4bit\\_samples.h](#)

## 10.352 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** ()

### 10.352.1 Detailed Description

This class implements conversion between byte packet samples to 2bit samples 1 byte = 4 2bit samples.

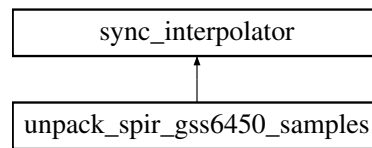
Definition at line 41 of file unpack\_intspir\_1bit\_samples.h.

The documentation for this class was generated from the following file:

- [unpack\\_intspir\\_1bit\\_samples.h](#)

## 10.353 unpack\_spir\_gss6450\_samples Class Reference

Inheritance diagram for unpack\_spir\_gss6450\_samples:



### Public Member Functions

- **unpack\_spir\_gss6450\_samples** (int adc\_nbit)
- void **decode\_4bits\_word** (uint32\_t input\_uint32, gr\_complex \*out, int adc\_bits\_)
- int **work** (int noutput\_items, gr\_vector\_const\_void\_star &input\_items, gr\_vector\_void\_star &output\_items)

### Friends

- unpack\_spir\_gss6450\_samples\_sptr **make\_unpack\_spir\_gss6450\_samples\_sptr** (int adc\_nbit)

### 10.353.1 Detailed Description

Definition at line 37 of file unpack\_spir\_gss6450\_samples.h.

The documentation for this class was generated from the following file:

- [unpack\\_spir\\_gss6450\\_samples.h](#)

## 10.354 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

### 10.354.1 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](#)

## 10.355 url\_t Struct Reference

### Public Attributes

- char **type** [32]
- char **path** [1024]
- char **dir** [1024]
- double **tint**

### 10.355.1 Detailed Description

Definition at line 909 of file rtklib.h.

The documentation for this struct was generated from the following file:

- [rtklib.h](#)

## 10.356 v27\_decision\_t Struct Reference

### Public Attributes

- unsigned int **w** [2]

### 10.356.1 Detailed Description

Definition at line 38 of file fec.h.

The documentation for this struct was generated from the following file:

- [fec.h](#)

## 10.357 v27\_poly\_t Struct Reference

### Public Attributes

- unsigned char **c0** [32]
- unsigned char **c1** [32]

### 10.357.1 Detailed Description

Definition at line 32 of file fec.h.

The documentation for this struct was generated from the following file:

- [fec.h](#)

## 10.358 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**

### 10.358.1 Detailed Description

Definition at line 45 of file fec.h.

The documentation for this struct was generated from the following file:

- [fec.h](#)

## 10.359 Viterbi\_Decoder Class Reference

Class that implements a Viterbi decoder.

```
#include <viterbi_decoder.h>
```

### Public Member Functions

- **Viterbi\_Decoder** (const int g\_encoder[], const int KK, const int nn)
- void **reset** ()
- float **decode\_block** (const double input\_c[], int \*output\_u\_int, const int LL)  
*Uses the Viterbi algorithm to perform hard-decision decoding of a convolutional code.*
- float **decode\_continuous** (const double sym[], const int traceback\_depth, int bits[], const int nbits\_requested, int &nbits\_decoded)

### 10.359.1 Detailed Description

Class that implements a Viterbi decoder.

Definition at line 34 of file viterbi\_decoder.h.

### 10.359.2 Member Function Documentation

#### 10.359.2.1 decode\_block()

```
float Viterbi_Decoder::decode_block (
    const double input_c[],
    int * output_u_int,
    const int LL )
```

Uses the Viterbi algorithm to perform hard-decision decoding of a convolutional code.

#### Parameters

in	<i>input_c[]</i>	The received signal in LLR-form. For BPSK, must be in form $r = 2*a*y/(\sigma^2)$ .
in	<i>LL</i>	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.h](#)



# Chapter 11

## File Documentation

### 11.1 `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 <cstdint>
#include <string>
```

#### Classes

- class [Acq\\_Conf](#)

#### 11.1.1 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](mailto: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

### 11.2 `acquisition_dump_reader.h` File Reference

Helper file for unit testing.

```
#include <cstdint>
#include <string>
#include <vector>
```

## Classes

- class [Acquisition\\_Dump\\_Reader](#)

### 11.2.1 Detailed Description

Helper file for unit testing.

#### Authors

Carles Fernandez-Prades, 2017. cfernandez(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

## 11.3 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 [Concurrent\\_Queue< Data >](#)  
*This class implements a thread-safe std::queue.*
- class [AcquisitionInterface](#)  
*This abstract class represents an interface to an acquisition GNSS block.*

### 11.3.1 Detailed Description

Header file of the interface to an acquisition GNSS block.

#### Author

Carlos Aviles, 2010. carlos.avilesr(at)gmail.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

## 11.4 acquisition\_msg\_rx.h File Reference

This is a helper class to catch the asynchronous messages emitted by an acquisition block.

```
#include "gnss_block_interface.h"
#include <gnuradio/block.h>
#include <gnuradio/top_block.h>
#include <pmt/pmt.h>
```

### Classes

- class [Acquisition\\_msg\\_rx](#)

### Typedefs

- using **Acquisition\_msg\_rx\_sptr** = gnss\_shared\_ptr< [Acquisition\\_msg\\_rx](#) >

### Functions

- Acquisition\_msg\_rx\_sptr **Acquisition\_msg\_rx\_make** ()

#### 11.4.1 Detailed Description

This is a helper class to catch the asynchronous messages emitted by an acquisition block.

#### Author

Carles Fernandez-Prades, 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) 2012-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

## 11.5 ad9361\_fpga\_signal\_source.h File Reference

signal source for Analog Devices front-end AD9361 connected directly to 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_dynamic_bit_selection.h"
#include "fpga_switch.h"
#include "gnss_block_interface.h"
#include <pmt/pmt.h>
#include <cstdint>
#include <memory>
#include <mutex>
#include <string>
#include <thread>
```

## Classes

- class [Ad9361FpgaSignalSource](#)

### 11.5.1 Detailed Description

signal source for Analog Devices front-end AD9361 connected directly to 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) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

## 11.6 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 <stdint>
#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 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\_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)

### 11.6.1 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

## 11.7 agnss\_ref\_location.h File Reference

Interface of an Assisted GNSS REFERENCE LOCATION storage.

```
#include <boost/serialization/nvp.hpp>
```

## Classes

- class [Agnss\\_Ref\\_Location](#)

*Interface of an Assisted GNSS REFERENCE LOCATION storage.*

### 11.7.1 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

## 11.8 agnss\_ref\_time.h File Reference

Interface of an Assisted GNSS REFERENCE TIME storage.

```
#include <boost/serialization/nvp.hpp>
```

## Classes

- class [Agnss\\_Ref\\_Time](#)

*Interface of an Assisted GNSS REFERENCE TIME storage.*

### 11.8.1 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

## 11.9 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 <cstdint>
#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.*

### 11.9.1 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

## 11.10 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)*

### 11.10.1 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

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

## 11.11 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

#### 11.11.1 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

## 11.12 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](#).*

### 11.12.1 Detailed Description

Interface of an adapter of a digital beamformer.

#### Author

Javier Arribas jarribas (at) ctte.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

## 11.13 Beidou\_B1I.h File Reference

Defines system parameters for BeiDou B1I signal and DNAV data.

```
#include "gnss_frequencies.h"
#include <stdint>
```

## 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 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] = "1111110000001100001100"
- constexpr char [BEIDOU\\_B1I\\_D2\\_SECONDARY\\_CODE\\_STR](#) [3] = "00"

### 11.13.1 Detailed Description

Defines system parameters for BeiDou B1I signal and DNAV data.

#### Author

Sergi Segura, 2018. [sergi.segura.munoz\(at\)gmail.com](mailto:sergi.segura.munoz(at)gmail.com)  
Damian Miralles, 2018. [dmiralles2009@gmail.com](mailto: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

## 11.14 beidou\_b1i\_dll\_pll\_tracking.h File Reference

Interface of an adapter of a DLL+PLL tracking loop block for Beidou B1I to a [TrackingInterface](#).

```
#include "dll_pll_veml_tracking.h"  
#include "tracking_interface.h"  
#include <string>
```

### Classes

- class [BeidouB1iDllPlTracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*

### 11.14.1 Detailed Description

Interface of an adapter of a DLL+PLL tracking loop block for Beidou B1I to a [TrackingInterface](#).

#### Author

Sergi Segura, 2018. [sergi.segura.munoz\(at\)gmail.com](mailto:sergi.segura.munoz(at)gmail.com)

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

## 11.15 beidou\_b1i\_pcps\_acquisition.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Beidou B1I signals.

```
#include "channel_fsm.h"
#include "complex_byte_to_float_x2.h"
#include "gnss_synchro.h"
#include "pcps_acquisition.h"
#include <gnuradio/blocks/float_to_complex.h>
#include <gnuradio/blocks/stream_to_vector.h>
#include <stdint>
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [BeidouB1IPcpsAcquisition](#)  
*This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.*

### 11.15.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Beidou B1I signals.

#### Authors

- Sergi Segura, 2018. [sergi.segura.munoz\(at\)gmail.com](mailto: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

## 11.16 beidou\_b1i\_signal\_replica.h File Reference

This file implements various functions for BeiDou B1I signal replica generation.

```
#include <complex>
#include <stdint>
#include <gsl/gsl>
```

### 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.*

### 11.16.1 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

## 11.17 beidou\_b1i\_telemetry\_decoder.h File Reference

Interface of an adapter of a Beidou B1I NAV data decoder block to a [TelemetryDecoderInterface](#).

```
#include "beidou_b1i_telemetry_decoder_gs.h"
#include "gnss_satellite.h"
#include "gnss_synchro.h"
#include "telemetry_decoder_interface.h"
#include "tlm_conf.h"
#include <gnuradio/runtime_types.h>
#include <cstdio>
#include <string>
```

### Classes

- class [BeidouB1iTelemetryDecoder](#)

*This class implements a NAV data decoder for BEIDOU B1I.*

### 11.17.1 Detailed Description

Interface of an adapter of a Beidou B1I NAV data decoder block to a [TelemetryDecoderInterface](#).

#### Author

Damian Miralles, 2018. [dmiralles2009@gmail.com](mailto: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

## 11.18 beidou\_b1i\_telemetry\_decoder\_gs.h File Reference

Implementation of a BEIDOU B1I DNAV data decoder block.

```
#include "beidou_dnav_navigation_message.h"
#include "gnss_block_interface.h"
#include "gnss_satellite.h"
#include "tlm_conf.h"
#include <boost/circular_buffer.hpp>
#include <gnuradio/block.h>
#include <gnuradio/types.h>
#include <array>
#include <cstdint>
#include <fstream>
#include <string>
```

### Classes

- class [beidou\\_b1i\\_telemetry\\_decoder\\_gs](#)

*This class implements a block that decodes the BeiDou DNAV data.*

### Typedefs

- using **beidou\_b1i\_telemetry\_decoder\_gs\_sptr** = gnss\_shared\_ptr< [beidou\\_b1i\\_telemetry\\_decoder\\_gs](#) >

### Functions

- beidou\_b1i\_telemetry\_decoder\_gs\_sptr **beidou\_b1i\_make\_telemetry\_decoder\_gs** (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)

#### 11.18.1 Detailed Description

Implementation of a BEIDOU B1I DNAV data decoder block.

Code added as part of GSoC 2018 program.

#### Author

Damian Miralles, 2018. [dmiralles2009\(at\)gmail.com](mailto:dmiralles2009(at)gmail.com)

Sergi Segura, 2018. [sergi.segura.munoz\(at\)gmail.es](mailto:sergi.segura.munoz(at)gmail.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

## 11.19 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 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\_SECOND** \* **BEIDOU\_B3I\_TELEMETRY\_SYMBOLS\_PER\_BIT**
- constexpr char **BEIDOU\_B3I\_SECONDARY\_CODE\_STR** [21] = "00000100110101001110"
- constexpr char **BEIDOU\_B3I\_GEO\_PREAMBLE\_SYMBOLS\_STR** [23] = "1111110000001100001100"
- constexpr char **BEIDOU\_B3I\_D2\_SECONDARY\_CODE\_STR** [3] = "00"

### 11.19.1 Detailed Description

Defines system parameters for BeiDou B3I signal and DNAV data.

#### Author

Damian Miralles, 2019. [dmiralles2009@gmail.com](mailto: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

## 11.20 beidou\_b3i\_dll\_pll\_tracking.h File Reference

Interface of an adapter of a DLL+PLL tracking loop block for Beidou B3I to a [TrackingInterface](#).

```
#include "dll_pll_veml_tracking.h"
#include "tracking_interface.h"
#include <string>
```

### Classes

- class [BeidouB3iDllPllTracking](#)

*This class implements a code DLL + carrier PLL tracking loop.*

### 11.20.1 Detailed Description

Interface of an adapter of a DLL+PLL tracking loop block for Beidou B3I to a [TrackingInterface](#).

#### Author

Damian Miralles, 2019. dmiralles2009(at)gmail.com

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

## 11.21 beidou\_b3i\_pcps\_acquisition.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Beidou B3I signals.

```
#include "acq_conf.h"
#include "channel_fsm.h"
#include "complex_byte_to_float_x2.h"
#include "gnss_synchro.h"
#include "pcps_acquisition.h"
#include <gnuradio/blocks/float_to_complex.h>
#include <gnuradio/blocks/stream_to_vector.h>
#include <stdint>
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [BeidouB3iPcpsAcquisition](#)

*This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for BeiDou B3I signals.*

### 11.21.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Beidou B3I signals.

#### Author

Damian Miralles, 2019. [dmiralles2009@gmail.com](mailto: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

## 11.22 beidou\_b3i\_signal\_replica.h File Reference

This file implements various functions for BeiDou B3I signal replica generation.

```
#include <complex>
#include <cstdint>
#include <gsl/gsl>
```

### 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.*

### 11.22.1 Detailed Description

This file implements various functions for BeiDou B3I signal replica generation.

#### Author

Damian Miralles, 2019. [dmiralles2009@gmail.com](mailto: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

## 11.23 beidou\_b3i\_telemetry\_decoder.h File Reference

Interface of an adapter of a Beidou B3I NAV data decoder block to a [TelemetryDecoderInterface](#).

```
#include "beidou_b3i_telemetry_decoder_gs.h"
#include "gnss_satellite.h"
#include "gnss_synchro.h"
#include "telemetry_decoder_interface.h"
#include "tlm_conf.h"
#include <gnuradio/runtime_types.h>
#include <cstdint>
#include <string>
```

### Classes

- class [BeidouB3iTelemetryDecoder](#)  
*This class implements a NAV data decoder for BEIDOU B3I.*

#### 11.23.1 Detailed Description

Interface of an adapter of a Beidou B3I NAV data decoder block to a [TelemetryDecoderInterface](#).

#### Author

Damian Miralles, 2019. [dmiralles2009@gmail.com](mailto: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

## 11.24 beidou\_b3i\_telemetry\_decoder\_gs.h File Reference

Implementation of a BEIDOU B3I DNAV data decoder block.

```
#include "beidou_dnav_navigation_message.h"
#include "gnss_block_interface.h"
#include "gnss_satellite.h"
#include "tlm_conf.h"
#include <boost/circular_buffer.hpp>
#include <gnuradio/block.h>
#include <gnuradio/types.h>
#include <array>
#include <stdint>
#include <fstream>
#include <string>
```

## Classes

- class [beidou\\_b3i\\_telemetry\\_decoder\\_gs](#)

*This class implements a block that decodes the BeiDou DNAV data.*

## Typedefs

- using **beidou\_b3i\_telemetry\_decoder\_gs\_sptr** = gnss\_shared\_ptr< [beidou\\_b3i\\_telemetry\\_decoder\\_gs](#) >

## Functions

- beidou\_b3i\_telemetry\_decoder\_gs\_sptr **beidou\_b3i\_make\_telemetry\_decoder\_gs** (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)

### 11.24.1 Detailed Description

Implementation of a BEIDOU B3I DNAV data decoder block.

#### Author

Damian Miralles, 2019. 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

## 11.25 Beidou\_DNAV.h File Reference

Defines system parameters for BeiDou DNAV data processing.

```
#include "MATH_CONSTANTS.h"
#include <cstdint>
#include <utility>
#include <vector>
```

## Functions

- `const std::vector< std::pair< int32_t, int32_t > > D1_PRE {{{1, 1}}}`
- `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_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_PNUM {{{43, 4}}}`
- `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})

## 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 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 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"

### 11.25.1 Detailed Description

Defines system parameters for BeiDou DNAV data processing.

Author

Damian Miralles, 2018. [dmiralles2009@gmail.com](mailto: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

## 11.26 beidou\_dnav\_almanac.h File Reference

Interface of a Beidou DNAV Almanac storage.

```
#include <boost/serialization/nvp.hpp>
```

### Classes

- class [Beidou\\_Dnav\\_Almanac](#)  
*This class is a storage for the BeiDou D1 almanac.*

#### 11.26.1 Detailed Description

Interface of a Beidou DNAV Almanac storage.

##### Author

Sergi Segura, 2018. [sergi.segura.munoz\(at\)gmail.com](mailto: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

## 11.27 beidou\_dnav\_ephemeris.h File Reference

Interface of a BEIDOU EPHEMERIS storage.

```
#include <boost/serialization/nvp.hpp>
#include <map>
#include <string>
```

### Classes

- class [Beidou\\_Dnav\\_Ephemeris](#)  
*This class is a storage and orbital model functions for the GPS SV ephemeris data as described in BeiDou Navigation Satellite System Signal In Space Interface Control Document Open Service Signal B1I (Version 3.0)*

#### 11.27.1 Detailed Description

Interface of a BEIDOU EPHEMERIS storage.

##### Author

Sergi Segura, 2018. [sergi.segura.munoz\(at\)gmail.com](mailto: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

## 11.28 beidou\_dnav\_iono.h File Reference

Interface of a BEIDOU IONOSPHERIC MODEL storage.

```
#include <boost/serialization/nvp.hpp>
```

### Classes

- class [Beidou\\_Dnav\\_Iono](#)

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

### 11.28.1 Detailed Description

Interface of a BEIDOU IONOSPHERIC MODEL storage.

#### Author

Sergi Segura, 2018. [sergi.segura.munoz\(at\)gmail.com](mailto: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

## 11.29 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_ephemeris.h"
#include "beidou_dnav_iono.h"
#include "beidou_dnav_utc_model.h"
#include <bitset>
#include <cstdint>
#include <map>
#include <string>
#include <utility>
#include <vector>
```

### Classes

- class [Beidou\\_Dnav\\_Navigation\\_Message](#)

*This class decodes a BeiDou D1 NAV Data message.*

### 11.29.1 Detailed Description

Interface of a BeiDou DNAV Data message decoder.

#### Author

Sergi Segura, 2018. [sergi.segura.munoz\(at\)gmail.com](mailto:sergi.segura.munoz(at)gmail.com)  
Damian Miralles, 2018. [dmiralles2009@gmail.com](mailto: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

## 11.30 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.*

### 11.30.1 Detailed Description

Interface of a BeiDou UTC MODEL storage.

#### Author

Damian Miralles, 2018. [dmiralles2009@gmail.com](mailto:dmiralles2009@gmail.com)  
Sergi Segura, 2018. [sergi.segura.munoz\(at\)gmail.com](mailto: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

## 11.31 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)`

### 11.31.1 Detailed Description

Utilities for bit manipulation of the libswiftnav library.

#### Author

Fergus Noble [fergus@swift-nav.com](mailto: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](mailto:fergus@swift-nav.com)

SPDX-License-Identifier: LGPL-3.0-only

## 11.32 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 <stdint>
#include <string>
```

## Classes

- class [ByteToShort](#)  
*Adapts an 8-bits sample stream (IF) to a short int stream (IF)*

### 11.32.1 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

## 11.33 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 ()`

### 11.33.1 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

## 11.34 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 <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.*

### 11.34.1 Detailed Description

Interface of a GNSS channel.

#### Author

Carlos Aviles, 2010. carlos.avilesr(at)gmail.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

## 11.35 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)

### 11.35.1 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

## 11.36 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.*

### 11.36.1 Detailed Description

Interface of the State Machine for channel.

#### Authors

Javier Arribas, 2019. [javiarribas@gmail.com](mailto:javiarribas@gmail.com) Antonio Ramos, 2017. [antonio.ramos\(at\)cttc.es](mailto:antonio.ramos(at)cttc.es) Luis Esteve, 2011. [luis\(at\)epsilon-formacion.com](mailto: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

## 11.37 channel\_interface.h File Reference

This class represents an interface to a channel GNSS block.

```
#include "gnss_block_interface.h"
#include "gnss_signal.h"
```

### Classes

- class [ChannelInterface](#)

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

### 11.37.1 Detailed Description

This class represents an interface to a channel GNSS block.

#### Author

Carlos Aviles, 2010. carlos.avilesr(at)gmail.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

## 11.38 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)

### 11.38.1 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

## 11.39 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** ()

### 11.39.1 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

## 11.40 cIFFT.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\\_DataFormat](#) 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 numRows, 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)

### 11.40.1 Detailed Description

FFT in OpenCL.

Version: <1.0>

Copyright ( C ) 2008 Apple Inc. All Rights Reserved. SPDX-License-Identifier: LicenseRef-Apple-Permissive

## 11.41 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)

### 11.41.1 Detailed Description

Utilities for CNAV message manipulation of the libswiftnav library.

#### Author

Valeri Atamaniouk [valeri@swift-nav.com](mailto: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](mailto:valeri@swift-nav.com)

SPDX-License-Identifier: LGPL-3.0-only

## 11.42 `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)`

### 11.42.1 Detailed Description

Class that defines a receiver command event.

#### Author

Javier Arribas, 2019. [jarribas\(at\)cttc.es](mailto: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

## 11.43 `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** ()

### 11.43.1 Detailed Description

Adapts a `std::complex<signed char>` stream into two 16-bits (short) streams.

#### Author

Carles Fernandez Prades, [cfernandez\(at\)cttc.es](mailto: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

## 11.44 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** ()

### 11.44.1 Detailed Description

Adapts a `gr_complex` stream into a `std::complex<signed char>` stream.

#### Author

Carles Fernandez Prades, [cfernandez\(at\)cttc.es](mailto: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

## 11.45 `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`.*

### 11.45.1 Detailed Description

Interface of a thread-safe `std::map`.

#### Author

Javier Arribas, 2011. [jarribas\(at\)cttc.es](mailto: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

## 11.46 `concurrent_queue.h` File Reference

Interface of a thread-safe `std::queue`.

```
#include <chrono>
#include <condition_variable>
#include <mutex>
#include <queue>
#include <thread>
```

## Classes

- class [Concurrent\\_Queue< Data >](#)

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

### 11.46.1 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

## 11.47 configuration\_interface.h File Reference

This class represents an interface to configuration parameters.

```
#include <stdint>
#include <string>
```

## Classes

- class [ConfigurationInterface](#)

*This abstract class represents an interface to configuration parameters.*

### 11.47.1 Detailed Description

This class represents an interface to configuration parameters.

#### Author

Carlos Aviles, 2010. carlos.avilesr(at)gmail.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

## 11.48 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 ()`

#### 11.48.1 Detailed Description

Conjugate a stream of `gr_complex`.

#### Author

Carles Fernandez Prades, [cfernandez\(at\)cttc.es](mailto: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

## 11.49 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** ()

### 11.49.1 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

## 11.50 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** ()

### 11.50.1 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

## 11.51 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 <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.*

### 11.51.1 Detailed Description

Interface of the receiver control plane.

#### Author

Carlos Aviles, 2010. carlos.avilesr(at)gmail.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

## 11.52 convolutional.h File Reference

General functions used to implement convolutional encoding.

```
#include <volk_gnssssdr/volk_gnssssdr.h>
#include <vector>
```

### Functions

- int [parity\\_counter](#) (int symbol, int length)  
*Determines if a symbol has odd (1) or even (0) parity Output parameters:*
- int [nsc\\_enc\\_bit](#) (int state\_out\_p[], int input, int state\_in, const int g[], int KK, int nn)  
*Convolutionally encodes a single bit using a rate 1/n encoder. Takes in one input bit at a time, and produces a n-bit output.*
- void [nsc\\_transit](#) (int output\_p[], int trans\_p[], int input, int g[], int KK, int nn)  
*Function that creates the transit and output vectors.*
- float [Gamma](#) (const float rec\_array[], int symbol, int nn)  
*Computes the branch metric used for decoding.*
- void [Viterbi](#) (int output\_u\_int[], const int out0[], const int state0[], const int out1[], const int state1[], const float input\_c[], int KK, int nn, int LL)  
*Uses the Viterbi algorithm to perform hard-decision decoding of a convolutional code.*

### Variables

- const float **MAXLOG** = 1e7

#### 11.52.1 Detailed Description

General functions used to implement convolutional encoding.

#### Author

Matthew C. Valenti, 2006-2008.  
C. Fernandez-Prades, 2019.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2006-2008 Matthew C. Valenti Copyright (C) 2019 C. Fernandez-Prades SPDX-License-Identifier: GPL-3.0-or-later

This file is a derived work of the original file, which had this note:

The functions in this file are part of the Iterative Solutions Coded Modulation Library. The Iterative Solutions Coded Modulation Library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

## 11.53 `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.*

### 11.53.1 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

## 11.54 `cpu_multicorrelator_16sc.h` File Reference

Highly optimized CPU vector multiTAP correlator class for `lv_16sc_t` (short int complex)

```
#include <volk_gnssdr/volk_gnssdr.h>
```

### Classes

- class [Cpu\\_Multicorrelator\\_16sc](#)  
*Class that implements carrier wipe-off and correlators.*

### 11.54.1 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

## 11.55 `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.*

### 11.55.1 Detailed Description

Highly optimized CPU vector multiTAP correlator class using real-valued local codes.

### Authors

- Javier Arribas, 2015. [jarribas\(at\)cttc.es](mailto:jarribas(at)cttc.es)
- Cillian O'Driscoll, 2017, [cillian.odriscoll\(at\)gmail.com](mailto:cillian.odriscoll(at)gmail.com)

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

## 11.56 `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** ()

### 11.56.1 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

## 11.57 `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.*

### 11.57.1 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

## 11.58 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 "gnss_block_interface.h"
#include "gr_complex_ip_packet_source.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.*

### 11.58.1 Detailed Description

Receives ip frames containing samples in UDP frame encapsulation using a high performance packet capture library (libpcap)

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

## 11.59 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.*

### 11.59.1 Detailed Description

Interface of an adapter of a direct resampler conditioner block to a `SignalConditionerInterface`.

#### Author

Carlos Aviles, 2010. [carlos.avilesr\(at\)gmail.com](mailto:carlos.avilesr(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

## 11.60 `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)

### 11.60.1 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](mailto: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

## 11.61 `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 <stdint>
```

### 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`)

#### 11.61.1 Detailed Description

Nearest neighborhood resampler with `gr_complex` input and `gr_complex` output.

#### Author

Luis Esteve, 2011. [luis\(at\)epsilon-formacion.com](mailto: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

## 11.62 `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 <stdint>
```

## 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)

### 11.62.1 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

## 11.63 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"`

### 11.63.1 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

## 11.64 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 <stdint>
#include <string>
```

### Classes

- class [Dll\\_Pll\\_Conf](#)

### 11.64.1 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

## 11.65 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 <stdint>
#include <string>
```

## Classes

- class [Dll\\_Pll\\_Conf\\_Fpga](#)

### 11.65.1 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

## 11.66 dll\_pll\_veml\_tracking.h File Reference

Implementation of a code DLL + carrier PLL tracking block.

```
#include "cpu_multicorrelator_real_codes.h"
#include "dll_pll_conf.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_gnssssdr/volk_gnssssdr_alloc.h>
#include <cstdint>
#include <cstdint>
#include <fstream>
#include <string>
#include <typeinfo>
#include <utility>
```

## 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\_)

### 11.66.1 Detailed Description

Implementation of a code DLL + carrier PLL tracking block.

#### Author

Javier Arribas, 2018. jarribas(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) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

## 11.67 dll\_pll\_veml\_tracking\_fpga.h File Reference

Implementation of a code DLL + carrier PLL tracking block using an FPGA.

```
#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_gnssssdr/volk_gnssssdr_alloc.h>
#include <cstdint>
#include <cstring>
#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\_)

### 11.67.1 Detailed Description

Implementation of a code DLL + carrier PLL tracking block using an FPGA.

#### Author

Marc Majoral, 2019. [marc.majoral@cttc.es](mailto:marc.majoral@cttc.es)  
 Javier Arribas, 2019. [jarribas@cttc.es](mailto:jarribas@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

## 11.68 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)

### 11.68.1 Detailed Description

Utilities for CRC computation of the libswiftnav library.

#### Author

Fergus Noble [fergus@swift-nav.com](mailto: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](mailto:fergus@swift-nav.com)

SPDX-License-Identifier: LGPL-3.0-only

## 11.69 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.*

### 11.69.1 Detailed Description

Class that implements an exponential smoother.

#### Authors

Carles Fernandez, 2019 [cfernandez@cttc.es](mailto: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

## 11.70 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)

### 11.70.1 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

## 11.71 `fft_base_kernels.h` File Reference

FFT base kernels for OpenCL.

```
#include <string>
```

### 11.71.1 Detailed Description

FFT base kernels for OpenCL.

Version: <1.0>

Copyright ( C ) 2008 Apple Inc. All Rights Reserved. SPDX-License-Identifier: LicenseRef-Apple-Permissive

## 11.72 `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)

### 11.72.1 Detailed Description

Internals of FFT for OpenCL.

Version: <1.0>

Copyright ( C ) 2008 Apple Inc. All Rights Reserved. SPDX-License-Identifier: LicenseRef-Apple-Permissive

## 11.73 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](#).*

### 11.73.1 Detailed Description

A [ConfigurationInterface](#) that reads the configuration from a file.

#### Author

Carlos Aviles, 2010. carlos.avilesr(at)gmail.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](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

## 11.74 file\_signal\_source.h File Reference

Interface of a class that reads signals samples from a file and adapts it to a `SignalSourceInterface`.

```
#include "concurrent_queue.h"
#include "gnss_block_interface.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>
```

### Classes

- class [FileSignalSource](#)

*Class that reads signals samples from a file and adapts it to a `SignalSourceInterface`.*

### 11.74.1 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)gmail.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

## 11.75 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.*

### 11.75.1 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

## 11.76 flexiband\_signal\_source.h File Reference

Signal 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 "gnss_block_interface.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).*

### 11.76.1 Detailed Description

Signal 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

## 11.77 fmcomms2\_signal\_source.h File Reference

Interface to use SDR hardware based in FCOMMS2 driver from analog devices, for example FCOMMS4 and ADALM-PLUTO (PlutoSdr)

```
#include "gnss_block_interface.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](#)

### 11.77.1 Detailed Description

Interface to use SDR hardware based in FCOMMS2 driver from analog devices, for example FCOMMS4 and ADALM-PLUTO (PlutoSdr)

#### Author

Rodrigo Muñoz, 2017. [rmunozl@inacap.cl](mailto:rmunozl@inacap.cl), [rodrigo.munoz@proteinlab.cl](mailto:rodrigo.munoz@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

## 11.78 fpga\_acquisition.h File Reference

Highly optimized FPGA vector correlator class.

```
#include <cstdint>
#include <string>
```

### Classes

- class [Fpga\\_Acquisition](#)  
*Class that implements carrier wipe-off and correlators.*

### 11.78.1 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

## 11.79 fpga\_dynamic\_bit\_selection.h File Reference

Dynamic bit selection in the received signal.

```
#include <stddef>
#include <stdint>
#include <string>
```

### Classes

- class [Fpga\\_dynamic\\_bit\\_selection](#)

*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.*

### 11.79.1 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) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

## 11.80 fpga\_multicorrelator.h File Reference

FPGA vector correlator class.

```
#include <gnuradio/block.h>
#include <volk_gnssdr/volk_gnssdr_alloc.h>
#include <stdint>
#include <string>
```

## Classes

- class [Fpga\\_Multicorrelator\\_8sc](#)  
*Class that implements carrier wipe-off and correlators.*

### 11.80.1 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

## 11.81 fpga\_switch.h File Reference

Switch that connects the HW accelerator queues to the analog front end or the DMA.

```
#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.*

### 11.81.1 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

## 11.82 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.*

### 11.82.1 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

## 11.83 front\_end\_cal.h File Reference

Interface of the Front-end calibration program.

```
#include <armadillo>
#include <memory>
#include <string>
```

### Classes

- class [FrontEndCal](#)

### 11.83.1 Detailed Description

Interface of the Front-end calibration program.

#### 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

## 11.84 galileo\_almanac.h File Reference

Interface of a Galileo ALMANAC storage.

```
#include <boost/serialization/nvp.hpp>
#include <cstdint>
```

### Classes

- class [Galileo\\_Almanac](#)

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

### 11.84.1 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

## 11.85 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.*

### 11.85.1 Detailed Description

Interface of a Galileo ALMANAC storage helper.

#### 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

## 11.86 Galileo\_CNAV.h File Reference

Galileo CNAV message constants. Data from: Galileo High Accuracy Service E6-B Signal-In-Space Message Specification v1.2 (April 2020).

```
#include <cstdint>
#include <cstdint>
#include <utility>
```

## Functions

- const std::pair< int32\_t, int32\_t > **GALILEO\_HAS\_STATUS** ({1, 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\_URA\_FLAG** ({19, 1})
- 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\_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\_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** = 14
  - constexpr size\_t **HAS\_MSG\_DELTA\_ALONG\_TRACK\_LENGTH** = 12
  - constexpr size\_t **HAS\_MSG\_DELTA\_CROSS\_TRACK\_LENGTH** = 12
  - constexpr size\_t **HAS\_MSG\_DELTA\_CLOCK\_C0\_MULTIPLIER\_LENGTH** = 2
  - constexpr size\_t **HAS\_MSG\_DELTA\_CLOCK\_C0\_LENGTH** = 14
  - constexpr size\_t **HAS\_MSG\_NSYSPRIME\_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\_C0\_SUBSET\_LENGTH** = 14
  - 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 size\_t **HAS\_MSG\_URA\_LENGTH** = 2
  - constexpr int32\_t **GALILEO\_CNAV\_SYMBOLS\_PER\_PAGE** = 1000
- Total number of symbols per HAS page including the sync pattern.*
- constexpr int32\_t **GALILEO\_CNAV\_PREAMBLE\_PERIOD\_SYMBOLS** = 1000
  - constexpr int32\_t **GALILEO\_CNAV\_PAGE\_MS** = 1
- Duration in ms of a CNAV page.*
- 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\_BITS\_FOR\_CRC** = GALILEO\_CNAV\_HAS\_PAGE\_DATA\_BITS + GALILEO\_CNAV\_PAGE\_RESERVED\_BITS
  - constexpr int32\_t **GALILEO\_CNAV\_BYTES\_FOR\_CRC** = 60
  - 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\_ENCODED\_BLOCKS** = 255
  - constexpr int32\_t **GALILEO\_CNAV\_MT1\_HEADER\_BITS** = 32
  - constexpr int32\_t **HAS\_MSG\_MAX\_SATS** = 40
  - constexpr int32\_t **HAS\_MSG\_MAX\_SIGNALS** = 16
  - constexpr uint8\_t **HAS\_MSG\_GPS\_SYSTEM** = 0
  - constexpr uint8\_t **HAS\_MSG\_GALILEO\_SYSTEM** = 2
  - constexpr char **GALILEO\_CNAV\_PREAMBLE** [17] = "1011011101110000"

### 11.86.1 Detailed Description

Galileo CNAV message constants. Data from: Galileo High Accuracy Service E6-B Signal-In-Space Message Specification v1.2 (April 2020).

#### 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

## 11.87 galileo\_cnav\_message.h File Reference

Implementation of a Galileo CNAV Data message as described in Galileo High Accuracy Service E6-B Signal-In-Space Message Specification v1.2 (April 2020)

```
#include "Galileo_CNAV.h"
#include "galileo_has_data.h"
#include <bitset>
#include <cstdint>
#include <list>
#include <string>
```

### Classes

- class [Galileo\\_Cnav\\_Message](#)

*This class handles the Galileo CNAV Data message, as described in the Galileo High Accuracy Service E6-B Signal-In-Space Message Specification v1.2 (April 2020)*

### 11.87.1 Detailed Description

Implementation of a Galileo CNAV Data message as described in Galileo High Accuracy Service E6-B Signal-In-Space Message Specification v1.2 (April 2020)

#### 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

## 11.88 Galileo\_E1.h File Reference

Defines system parameters for Galileo E1 signal and NAV data.

```
#include "MATH_CONSTANTS.h"
#include "gnss_frequencies.h"
#include <cstdint>
#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 interpolation.*
- constexpr char **GALILEO\_E1\_C\_SECONDARY\_CODE** [26] = "0011100000001010110110010"
- 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]

### 11.88.1 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

## 11.89 galileo\_e1\_dll\_pll\_veml\_tracking.h File Reference

Adapts a DLL+PLL VEML (Very Early Minus Late) tracking loop block to a [TrackingInterface](#) for Galileo E1 signals.

```
#include "dll_pll_veml_tracking.h"
#include "tracking_interface.h"
#include <string>
```

### Classes

- class [GalileoE1DIPIIVemlTracking](#)

*This class Adapts a DLL+PLL VEML (Very Early Minus Late) tracking loop block to a [TrackingInterface](#) for Galileo E1 signals.*

### 11.89.1 Detailed Description

Adapts a DLL+PLL VEML (Very Early Minus Late) tracking loop block to a [TrackingInterface](#) for Galileo E1 signals.

#### Author

Luis Esteve, 2012. luis(at)epsilon-formacion.com

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

## 11.90 galileo\_e1\_dll\_pll\_veml\_tracking\_fpga.h File Reference

Adapts a DLL+PLL VEML (Very Early Minus Late) tracking loop block to a [TrackingInterface](#) for Galileo E1 signals for the FPGA.

```
#include "dll_pll_veml_tracking_fpga.h"
#include "tracking_interface.h"
#include <string>
```

### Classes

- class [GalileoE1DIPIIVemlTrackingFpga](#)

*This class Adapts a DLL+PLL VEML (Very Early Minus Late) tracking loop block to a [TrackingInterface](#) for Galileo E1 signals.*

### 11.90.1 Detailed Description

Adapts a DLL+PLL VEML (Very Early Minus Late) tracking loop block to a [TrackingInterface](#) for Galileo E1 signals for the FPGA.

#### Author

Marc Majoral, 2019. mmajoral(at)cttc.cat

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

## 11.91 galileo\_e1\_pcps\_8ms\_ambiguous\_acquisition.h File Reference

Adapts a PCPS 8ms acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

```
#include "acquisition_interface.h"
#include "galileo_pcps_8ms_acquisition_cc.h"
#include "gnss_synchro.h"
#include <gnuradio/blocks/stream_to_vector.h>
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [GalileoE1Pcps8msAmbiguousAcquisition](#)

*Adapts a PCPS 8ms acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.*

### 11.91.1 Detailed Description

Adapts a PCPS 8ms acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

#### 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

## 11.92 galileo\_e1\_pcps\_ambiguous\_acquisition.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

```
#include "acq_conf.h"
#include "channel_fsm.h"
#include "complex_byte_to_float_x2.h"
#include "gnss_synchro.h"
#include "pcps_acquisition.h"
#include <gnuradio/blocks/float_to_complex.h>
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [GalileoE1PcpsAmbiguousAcquisition](#)

*This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.*

### 11.92.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

#### 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

## 11.93 galileo\_e1\_pcps\_ambiguous\_acquisition\_fpga.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals for the FPGA.

```
#include "acq_conf.h"
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include "pcps_acquisition_fpga.h"
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [GalileoE1PcpsAmbiguousAcquisitionFpga](#)

*This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for Galileo E1 Signals.*

### 11.93.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals for the FPGA.

#### Author

Marc Majoral, 2019. 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

## 11.94 galileo\_e1\_pcps\_cccwsr\_ambiguous\_acquisition.h File Reference

Adapts a PCPS CCCWSR acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

```
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include "pcps_cccwsr_acquisition_cc.h"
#include <gnuradio/blocks/stream_to_vector.h>
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [GalileoE1PcpsCccwsrAmbiguousAcquisition](#)  
*Adapts a PCPS CCCWSR acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.*

### 11.94.1 Detailed Description

Adapts a PCPS CCCWSR acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

#### 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

## 11.95 galileo\_e1\_pcps\_quicksync\_ambiguous\_acquisition.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

```
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include "pcps_quicksync_acquisition_cc.h"
#include <gnuradio/blocks/stream_to_vector.h>
#include <memory>
#include <string>
#include <vector>
```

## Classes

- class [GalileoE1PcpsQuickSyncAmbiguousAcquisition](#)

*This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.*

### 11.95.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

#### Date

June, 2014

#### Author

Damian Miralles Sanchez. [dmiralles2009@gmail.com](mailto: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

## 11.96 galileo\_e1\_pcps\_tong\_ambiguous\_acquisition.h File Reference

Adapts a PCPS Tong acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

```
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include "pcps_tong_acquisition_cc.h"
#include <gnuradio/blocks/stream_to_vector.h>
#include <memory>
#include <string>
#include <vector>
```

## Classes

- class [GalileoE1PcpsTongAmbiguousAcquisition](#)

*Adapts a PCPS Tong acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.*

### 11.96.1 Detailed Description

Adapts a PCPS Tong acquisition block to an [AcquisitionInterface](#) for Galileo E1 Signals.

#### Author

Marc Molina, 2013. [marc.molina.pena\(at\)gmail.com](mailto: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

## 11.97 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/gsl>
```

### 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.*

### 11.97.1 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

## 11.98 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.*

### 11.98.1 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

## 11.99 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_gnssssdr/volk_gnssssdr_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`)

### 11.99.1 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

## 11.100 galileo\_e1b\_telemetry\_decoder.h File Reference

Interface of an adapter of a GALILEO E1B NAV data decoder block to a [TelemetryDecoderInterface](#).

```
#include "galileo_telemetry_decoder_gs.h"
#include "gnss_satellite.h"
#include "gnss_synchro.h"
#include "telemetry_decoder_interface.h"
#include "tlm_conf.h"
#include <gnuradio/runtime_types.h>
#include <cstring>
#include <string>
```

## Classes

- class [GalileoE1BTelemetryDecoder](#)  
*This class implements a NAV data decoder for Galileo INAV frames in E1B radio link.*

### 11.100.1 Detailed Description

Interface of an adapter of a GALILEO E1B NAV data decoder block to a [TelemetryDecoderInterface](#).

#### 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

## 11.101 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/gsl>
```

### 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.*

### 11.101.1 Detailed Description

This library implements various functions for Galileo E5 signal replica generation.

#### Author

Marc Sales, 2014. [marcsales92\(at\)gmail.com](mailto:marcsales92(at)gmail.com)  
Piyush Gupta, 2020. [piyush04111999@gmail.com](mailto: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

## 11.102 Galileo\_E5a.h File Reference

Defines system parameters for Galileo E5a signal and NAV data.

```
#include "MATH_CONSTANTS.h"
#include "gnss_frequencies.h"
#include <cstdint>
#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]

### 11.102.1 Detailed Description

Defines system parameters for Galileo E5a signal and NAV data.

#### Author

Marc Sales, 2014. [marcsales92@gmail.com](mailto: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

## 11.103 galileo\_e5a\_dll\_pll\_tracking.h File Reference

Adapts a code DLL + carrier PLL tracking block to a [TrackingInterface](#) for Galileo E5a signals.

```
#include "dll_pll_veml_tracking.h"
#include "tracking_interface.h"
#include <string>
```

### Classes

- class [GalileoE5aDllPllTracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*

### 11.103.1 Detailed Description

Adapts a code DLL + carrier PLL tracking block to a [TrackingInterface](#) for Galileo E5a signals.

#### Author

Marc Sales, 2014. [marcsales92\(at\)gmail.com](mailto:marcsales92(at)gmail.com) on work from:

- Javier Arribas, 2011. [jarribas\(at\)cttc.es](mailto:jarribas(at)cttc.es)
- Luis Esteve, 2012. [luis\(at\)epsilon-formacion.com](mailto: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

## 11.104 galileo\_e5a\_dll\_pll\_tracking\_fpga.h File Reference

Adapts a code DLL + carrier PLL tracking block to a [TrackingInterface](#) for Galileo E5a signals for the FPGA.

```
#include "dll_pll_veml_tracking_fpga.h"
#include "tracking_interface.h"
#include <string>
```

## Classes

- class [GalileoE5aDIIPITrackingFpga](#)

*This class implements a code DLL + carrier PLL tracking loop.*

### 11.104.1 Detailed Description

Adapts a code DLL + carrier PLL tracking block to a [TrackingInterface](#) for Galileo E5a signals for the FPGA.

#### Author

Marc Majoral, 2019. mmajoral(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

## 11.105 galileo\_e5a\_noncoherent\_iq\_acquisition\_caf.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E5a data and pilot Signals.

```
#include "channel_fsm.h"
#include "galileo_e5a_noncoherent_iq_acquisition_caf_cc.h"
#include "gnss_synchro.h"
#include <memory>
#include <string>
#include <vector>
```

## Classes

- class [GalileoE5aNoncoherentIQAcquisitionCaf](#)

### 11.105.1 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 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

## 11.106 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 "channel_fsm.h"
#include "gnss_synchro.h"
#include <gnuradio/block.h>
#include <gnuradio/fft/fft.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\\_cc](#) >

### Functions

- [galileo\\_e5a\\_noncoherentIQ\\_acquisition\\_caf\\_cc\\_sptr](#) [galileo\\_e5a\\_noncoherentIQ\\_make\\_acquisition\\_caf\\_cc](#) (unsigned int sampled\_ms, unsigned int max\_dwells, unsigned int doppler\_max, int64\_t fs\_in, int samples\_per\_ms, int samples\_per\_code, bool bit\_transition\_flag, bool dump, const std::string &dump\_filename, bool both\_signal\_components\_, int CAF\_window\_hz\_, int Zero\_padding\_, bool enable\_monitor\_output)

#### 11.106.1 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 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](mailto: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

## 11.107 galileo\_e5a\_pcps\_acquisition.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E5a data and pilot Signals.

```
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include "pcps_acquisition.h"
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [GalileoE5aPcpsAcquisition](#)

### 11.107.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E5a data and pilot Signals.

#### 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

## 11.108 galileo\_e5a\_pcps\_acquisition\_fpga.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E5a data and pilot Signals for the FPGA.

```
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include "pcps_acquisition_fpga.h"
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [GalileoE5aPcpsAcquisitionFpga](#)

*This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for Galileo E5a signals.*

### 11.108.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E5a data and pilot Signals for the FPGA.

#### Author

Marc Majoral, 2019. 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

## 11.109 galileo\_e5a\_telemetry\_decoder.h File Reference

Interface of an adapter of a GALILEO E5a FNAV data decoder block to a [TelemetryDecoderInterface](#).

```
#include "galileo_telemetry_decoder_gs.h"
#include "gnss_satellite.h"
#include "gnss_synchro.h"
#include "telemetry_decoder_interface.h"
#include "tlm_conf.h"
#include <gnuradio/runtime_types.h>
#include <cstring>
#include <string>
```

### Classes

- class [GalileoE5aTelemetryDecoder](#)

*This class implements a NAV data decoder for Galileo INAV frames in E1B radio link.*

### 11.109.1 Detailed Description

Interface of an adapter of a GALILEO E5a FNAV data decoder block to a [TelemetryDecoderInterface](#).

#### Author

Marc Sales, 2014. marcsales92(at)gmail.com 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

## 11.110 Galileo\_E5b.h File Reference

Defines system parameters for Galileo E5b signal and NAV data.

```
#include "MATH_CONSTANTS.h"
#include "gnss_frequencies.h"
#include <stdint.h>
#include <stdint.h>
```

### 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]

### 11.110.1 Detailed Description

Defines system parameters for Galileo E5b signal and NAV data.

#### Author

Piyush Gupta, 2020. [piyush04111999@gmail.com](mailto: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

## 11.111 galileo\_e5b\_dll\_pll\_tracking.h File Reference

Adapts a code DLL + carrier PLL tracking block to a [TrackingInterface](#) for Galileo E5b signals.

```
#include "dll_pll_veml_tracking.h"
#include "tracking_interface.h"
#include <string>
```

### Classes

- class [GalileoE5bDllPllTracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*

### 11.111.1 Detailed Description

Adapts a code DLL + carrier PLL tracking block to a [TrackingInterface](#) for Galileo E5b signals.

#### Author

Piyush Gupta, 2020. [piyush04111999@gmail.com](mailto:piyush04111999@gmail.com) on work from:

- Javier Arribas, 2011. [jarribas@cttc.es](mailto:jarribas@cttc.es)
- Luis Esteve, 2012. [luis@epsilon-formacion.com](mailto:luis@epsilon-formacion.com)
- Marc Sales, 2014. [marcsales92@gmail.com](mailto: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

## 11.112 galileo\_e5b\_pcps\_acquisition.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E5b data and pilot Signals.

```
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include "pcps_acquisition.h"
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [GalileoE5bPcpsAcquisition](#)

### 11.112.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E5b data and pilot Signals.

#### Author

Piyush Gupta, 2020. [piyush04111999@gmail.com](mailto: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

## 11.113 galileo\_e5b\_pcps\_acquisition\_fpga.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E5b data and pilot Signals for the FPGA.

```
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include "pcps_acquisition_fpga.h"
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [GalileoE5bPcpsAcquisitionFpga](#)

*This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for Galileo E5b signals.*

### 11.113.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E5b data and pilot Signals for the FPGA.

#### Author

Piyush Gupta, 2020. [piyush04111999@gmail.com](mailto: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

## 11.114 galileo\_e5b\_telemetry\_decoder.h File Reference

Interface of an adapter of a GALILEO E5B NAV data decoder block to a [TelemetryDecoderInterface](#).

```
#include "galileo_telemetry_decoder_gs.h"
#include "gnss_satellite.h"
#include "gnss_synchro.h"
#include "telemetry_decoder_interface.h"
#include "tlm_conf.h"
#include <gnuradio/runtime_types.h>
#include <cstdio>
#include <string>
```

### Classes

- class [GalileoE5bTelemetryDecoder](#)

*This class implements a NAV data decoder for Galileo INAV frames in E5b radio link.*

### 11.114.1 Detailed Description

Interface of an adapter of a GALILEO E5B NAV data decoder block to a [TelemetryDecoderInterface](#).

#### Author

Piyush Gupta 2020 [piyush04111999@gmail.com](mailto: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

## 11.115 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 <cstdint>
#include <stdint>
```

### 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]

### 11.115.1 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](mailto: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

## 11.116 galileo\_e6\_dll\_pll\_tracking.h File Reference

Adapts a code DLL + carrier PLL tracking block to a [TrackingInterface](#) for Galileo E6 signals.

```
#include "dll_pll_veml_tracking.h"
#include "tracking_interface.h"
#include <string>
```

### Classes

- class [GalileoE6DllPllTracking](#)

*This class implements a code DLL + carrier PLL tracking loop.*

#### 11.116.1 Detailed Description

Adapts a code DLL + carrier PLL tracking block to a [TrackingInterface](#) for Galileo E6 signals.

#### 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

## 11.117 galileo\_e6\_pcps\_acquisition.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E6 B/C Signals.

```
#include "acq_conf.h"
#include "channel_fsm.h"
#include "complex_byte_to_float_x2.h"
#include "gnss_synchro.h"
#include "pcps_acquisition.h"
#include <gnuradio/blocks/float_to_complex.h>
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [GalileoE6PcpsAcquisition](#)

*This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E6 Signals.*

### 11.117.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Galileo E6 B/C Signals.

#### 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

## 11.118 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/gsl>
```

### 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.*

### 11.118.1 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

## 11.119 galileo\_e6\_telemetry\_decoder.h File Reference

Interface of an adapter of a GALILEO E6 CNAV data decoder block to a [TelemetryDecoderInterface](#).

```
#include "galileo_telemetry_decoder_gs.h"
#include "gnss_satellite.h"
#include "gnss_synchro.h"
#include "telemetry_decoder_interface.h"
#include "tlm_conf.h"
#include <gnuradio/runtime_types.h>
#include <cstdio>
#include <string>
```

### Classes

- class [GalileoE6TelemetryDecoder](#)

*This class implements a NAV data decoder for Galileo CNAV frames in E6 radio link.*

### 11.119.1 Detailed Description

Interface of an adapter of a GALILEO E6 CNAV data decoder block to a [TelemetryDecoderInterface](#).

#### 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

## 11.120 galileo\_ephemeris.h File Reference

Interface of a Galileo EPHEMERIS storage.

```
#include <boost/serialization/nvp.hpp>
#include <cstdint>
```

## Classes

- class [Galileo\\_Ephemeris](#)

*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.pdf>)*

### 11.120.1 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

## 11.121 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\_DELTA12\_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](#)

### 11.121.1 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

## 11.122 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 1.2 (Nov 2015). See <https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo-OS-SIS-ICD.pdf>.*

### 11.122.1 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 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

## 11.123 galileo\_has\_data.h File Reference

Class for Galileo HAS message type 1 data storage.

```
#include <cstdint>
#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 E6-B Signal-In-Space Message Specification v1.2 (April 2020).*

### 11.123.1 Detailed Description

Class for Galileo HAS message type 1 data storage.

#### 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

## 11.124 Galileo\_INAV.h File Reference

Galileo INAV message constants.

```
#include "MATH_CONSTANTS.h"
#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 > > 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 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

### 11.124.1 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

### 11.125 galileo\_inav\_message.h File Reference

Implementation of a Galileo I/NAV Data message as described in Galileo OS SIS ICD Issue 1.2 (Nov. 2015)

```
#include "Galileo_INAV.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\\_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 1.2 (Nov 2015). See <https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo-OS-SIS-ICD.pdf>.*

### 11.125.1 Detailed Description

Implementation of a Galileo I/NAV Data message as described in Galileo OS SIS ICD Issue 1.2 (Nov. 2015)

#### 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

## 11.126 galileo\_iono.h File Reference

Interface of a Galileo Ionospheric Model storage.

```
#include <boost/serialization/nvp.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.*

### 11.126.1 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

## 11.127 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 "channel_fsm.h"
#include "gnss_synchro.h"
#include <gnuradio/block.h>
#include <gnuradio/fft/fft.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** (uint32\_t sampled\_ms, uint32\_t max\_dwells, uint32\_t doppler\_max, int64\_t fs\_in, int32\_t samples\_per\_ms, int32\_t samples\_per\_code, bool dump, const std::string &dump\_filename, bool enable\_monitor\_output)

#### 11.127.1 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

## 11.128 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_block_interface.h"
#include "gnss_satellite.h"
#include "tlm_conf.h"
#include <boost/circular_buffer.hpp>
#include <gnuradio/block.h>
#include <gnuradio/types.h>
#include <cstdint>
#include <fstream>
#include <string>
#include <vector>
```

### 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 [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf, int frame\_type)

#### 11.128.1 Detailed Description

Implementation of a Galileo unified INAV and FNAV message demodulator block.

#### 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

## 11.129 galileo\_utc\_model.h File Reference

Interface of a Galileo UTC MODEL storage.

```
#include <boost/serialization/nvp.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.pdf> paragraph 5.1.7.*

### 11.129.1 Detailed Description

Interface of a Galileo UTC MODEL storage.

#### Author

Javier Arribas, 2013. [jarribas\(at\)cttc.es](mailto: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

### 11.130 `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 <pmt/pmt.h>
#include <memory>
#include <string>
```

## Classes

- class [GenSignalSource](#)

*This class wraps blocks that generates synthesized GNSS signal and filters the signal.*

### 11.130.1 Detailed Description

It wraps blocks that generates synthesized GNSS signal and filters it.

#### Author

Marc Molina, 2013. [marc.molina.pena@gmail.com](mailto: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

## 11.131 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 ellipsoid\_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.*

### 11.131.1 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

## 11.132 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.*

### 11.132.1 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

## 11.133 glonass\_gnav\_almanac.h File Reference

Interface of a GLONASS GNAV ALMANAC storage.

```
#include <boost/serialization/nvp.hpp>
#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)*

### 11.133.1 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

## 11.134 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)*

### 11.134.1 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

## 11.135 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 <bitset>
#include <cstdint>
#include <map>
#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)*

### 11.135.1 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

## 11.136 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)*

### 11.136.1 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

## 11.137 glonass\_l1\_ca\_dll\_pll\_c\_aid\_tracking.h File Reference

Interface of an adapter of a DLL+PLL tracking loop block for Glonass L1 C/A to a [TrackingInterface](#).

```
#include "glonass_l1_ca_dll_pll_c_aid_tracking_cc.h"
#include "glonass_l1_ca_dll_pll_c_aid_tracking_sc.h"
#include "tracking_interface.h"
#include <string>
```

### Classes

- class [GlonassL1CaDllPllCAidTracking](#)

*This class implements a code DLL + carrier PLL tracking loop.*

### 11.137.1 Detailed Description

Interface of an adapter of a DLL+PLL tracking loop block for Glonass L1 C/A to a [TrackingInterface](#).

#### Author

Gabriel Araujo, 2017. gabriel.araujo.5000(at)gmail.com  
 Luis Esteve, 2017. luis(at)epsilon-formacion.com  
 Damian Miralles, 2017. dmiralles2009(at)gmail.com

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, Birkha user, 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

## 11.138 glonass\_l1\_ca\_dll\_pll\_c\_aid\_tracking\_cc.h File Reference

Implementation of a code DLL + carrier PLL tracking block.

```
#include "gnss_synchro.h"
#include "tracking_2nd_DLL_filter.h"
#include "tracking_FLL_PLL_filter.h"
#include "cpu_multicorrelator.h"
#include "gnss_block_interface.h"
#include <gnuradio/block.h>
#include <pmt/pmt.h>
#include <volk_gnssssdr/volk_gnssssdr_alloc.h>
#include <deque>
#include <fstream>
#include <map>
#include <string>
```

### Classes

- class [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_cc](#)  
*This class implements a DLL + PLL tracking loop block.*

### Typedefs

- using [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_cc\\_sptr](#) = gnss\_shared\_ptr< [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_cc](#) >

### Functions

- [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_cc\\_sptr](#) [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_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 pll\_bw\_narrow\_hz, float dll\_bw\_narrow\_hz, int32\_t extend\_correlation\_ms, float early\_late\_space\_chips)

### 11.138.1 Detailed Description

Implementation of a code DLL + carrier PLL tracking block.

#### Author

Gabriel Araujo, 2017. gabriel.araujo.5000(at)gmail.com  
 Luis Esteve, 2017. luis(at)epsilon-formacion.com  
 Damian Miralles, 2017. dmiralles2009(at)gmail.com

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, Birkha user, 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

## 11.139 glonass\_l1\_ca\_dll\_pll\_c\_aid\_tracking\_sc.h File Reference

Implementation of a code DLL + carrier PLL tracking block.

```
#include "cpu_multicorrelator_16sc.h"
#include "glonass_l1_signal_replica.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 <volk_gnssssdr/volk_gnssssdr.h>
#include <volk_gnssssdr/volk_gnssssdr_alloc.h>
#include <deque>
#include <fstream>
#include <map>
#include <string>
```

### Classes

- class [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_sc](#)  
*This class implements a DLL + PLL tracking loop block.*

### Typedefs

- using [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_sc\\_sptr](#) = gnss\_shared\_ptr<[glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_sc](#)>

### Functions

- [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_sc\\_sptr](#) [glonass\\_l1\\_ca\\_dll\\_pll\\_c\\_aid\\_make\\_tracking\\_sc](#) (int64\_t fs\_in, uint32\_t vector\_length, bool dump, const std::string &dump\_filename, float pll\_bw\_hz, float dll\_bw\_hz, float pll\_bw\_narrow\_hz, float dll\_bw\_narrow\_hz, int32\_t extend\_correlation\_ms, float early\_late\_space\_chips)

### 11.139.1 Detailed Description

Implementation of a code DLL + carrier PLL tracking block.

#### Author

Gabriel Araujo, 2017. gabriel.araujo.5000(at)gmail.com

Luis Esteve, 2017. luis(at)epsilon-formacion.com

Damian Miralles, 2017. dmiralles2009(at)gmail.com

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, Birkha user, 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

## 11.140 glonass\_l1\_ca\_dll\_pll\_tracking.h File Reference

Interface of an adapter of a DLL+PLL tracking loop block for Glonass L1 C/A to a [TrackingInterface](#).

```
#include "glonass_l1_ca_dll_pll_tracking_cc.h"
#include "tracking_interface.h"
#include <string>
```

### Classes

- class [GlonassL1CaDllPllTracking](#)

*This class implements a code DLL + carrier PLL tracking loop.*

### 11.140.1 Detailed Description

Interface of an adapter of a DLL+PLL tracking loop block for Glonass L1 C/A to a [TrackingInterface](#).

#### Author

Gabriel Araujo, 2017. gabriel.araujo.5000(at)gmail.com

Luis Esteve, 2017. luis(at)epsilon-formacion.com

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, Birkha user, 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

## 11.141 glonass\_l1\_ca\_dll\_pll\_tracking\_cc.h File Reference

Implementation of a code DLL + carrier PLL tracking block.

```
#include "cpu_multicorrelator.h"
#include "gnss_block_interface.h"
#include "gnss_synchro.h"
#include "tracking_2nd_DLL_filter.h"
#include "tracking_2nd_PLL_filter.h"
#include <gnuradio/block.h>
#include <volk_gnssssdr/volk_gnssssdr_alloc.h>
#include <fstream>
#include <map>
#include <string>
```

### Classes

- class [Glonass\\_L1\\_Ca\\_DLL\\_Pll\\_Tracking\\_cc](#)  
*This class implements a DLL + PLL tracking loop block.*

### Typedefs

- using [glonass\\_l1\\_ca\\_dll\\_pll\\_tracking\\_cc\\_sptr](#) = gnss\_shared\_ptr< [Glonass\\_L1\\_Ca\\_DLL\\_Pll\\_Tracking\\_cc](#) >

### Functions

- glonass\_l1\_ca\_dll\_pll\_tracking\_cc\_sptr [glonass\\_l1\\_ca\\_dll\\_pll\\_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)

#### 11.141.1 Detailed Description

Implementation of a code DLL + carrier PLL tracking block.

#### Author

Gabriel Araujo, 2017. gabriel.araujo.5000(at)gmail.com  
Luis Esteve, 2017. luis(at)epsilon-formacion.com  
Damian Miralles, 2017. dmiralles2009(at)gmail.com

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, Birkha user, 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

## 11.142 glonass\_l1\_ca\_pcps\_acquisition.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Glonass L1 C/A signals.

```
#include "acq_conf.h"
#include "channel_fsm.h"
#include "complex_byte_to_float_x2.h"
#include "gnss_synchro.h"
#include "pcps_acquisition.h"
#include <gnuradio/blocks/float_to_complex.h>
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [GlonassL1CaPcpsAcquisition](#)

*This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.*

### 11.142.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Glonass L1 C/A signals.

#### Author

Gabriel Araujo, 2017. gabriel.araujo.5000(at)gmail.com  
Luis Esteve, 2017. 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

## 11.143 glonass\_l1\_ca\_telemetry\_decoder.h File Reference

Interface of an adapter of a GLONASS L1 C/A NAV data decoder block to a [TelemetryDecoderInterface](#).

```
#include "glonass_l1_ca_telemetry_decoder_gs.h"
#include "gnss_satellite.h"
#include "gnss_synchro.h"
#include "telemetry_decoder_interface.h"
#include "tlm_conf.h"
#include <gnuradio/runtime_types.h>
#include <stddef>
#include <string>
```

### Classes

- class [GlonassL1CaTelemetryDecoder](#)

*This class implements a NAV data decoder for GLONASS L1 C/A.*

### 11.143.1 Detailed Description

Interface of an adapter of a GLONASS L1 C/A NAV data decoder block to a [TelemetryDecoderInterface](#).

#### Note

Code added as part of GSoC 2017 program

#### 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

## 11.144 glonass\_l1\_ca\_telemetry\_decoder\_gs.h File Reference

Implementation of a GLONASS L1 C/A NAV data decoder block.

```
#include "GLONASS_L1_L2_CA.h"
#include "glonass_gnav_navigation_message.h"
#include "gnss_block_interface.h"
#include "gnss_satellite.h"
#include "gnss_synchro.h"
#include "tlm_conf.h"
#include <boost/circular_buffer.hpp>
#include <gnuradio/block.h>
#include <gnuradio/types.h>
#include <array>
#include <cstdint>
#include <fstream>
#include <string>
```

### Classes

- class [glonass\\_l1\\_ca\\_telemetry\\_decoder\\_gs](#)

*This class implements a block that decodes the GNAV data defined in GLONASS ICD v5.1.*

### Typedefs

- using [glonass\\_l1\\_ca\\_telemetry\\_decoder\\_gs\\_sptr](#) = gnss\_shared\_ptr< [glonass\\_l1\\_ca\\_telemetry\\_decoder\\_gs](#) >

### Functions

- [glonass\\_l1\\_ca\\_telemetry\\_decoder\\_gs\\_sptr glonass\\_l1\\_ca\\_make\\_telemetry\\_decoder\\_gs](#) (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)

### 11.144.1 Detailed Description

Implementation of a GLONASS L1 C/A NAV data decoder block.

#### Note

Code added as part of GSoC 2017 program

#### 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

## 11.145 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`

### 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}}

## 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` = `DFREQ2_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\\_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\\_CHIP\\_PERIOD\\_S](#) = 1.9569e-06  
*GLONASS L1 C/A chip period [seconds].*
- constexpr double [GLONASS\\_L1\\_CA\\_SYMBOL\\_RATE\\_BPS](#) = 1000.0
- constexpr int32\_t [GLONASS\\_CA\\_NBR\\_SATS](#) = 24
- constexpr int32\_t [GLONASS\\_L1\\_CA\\_HISTORY\\_DEEP](#) = 100
- constexpr double [GLONASS\\_GNAV\\_PREAMBLE\\_DURATION\\_S](#) = 0.300
- 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\\_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\\_BITS\\_SECOND](#) \* [GLONASS\\_GNAV\\_TELEMETRY\\_SYMBOLS\\_PER\\_BIT](#)  
*NAV message bit rate [symbols/s].*
- constexpr int32\_t [GLONASS\\_GNAV\\_STRING\\_SYMBOLS](#) = 2000  
*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}`

### 11.145.1 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@gmail.com](mailto: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

## 11.146 glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking.h File Reference

Interface of an adapter of a DLL+PLL tracking loop block for Glonass L2 C/A to a [TrackingInterface](#).

```
#include "glonass_l2_ca_dll_pll_c_aid_tracking_cc.h"
#include "glonass_l2_ca_dll_pll_c_aid_tracking_sc.h"
#include "tracking_interface.h"
#include <string>
```

### Classes

- class [GlonassL2CaDllPllCAidTracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*

### 11.146.1 Detailed Description

Interface of an adapter of a DLL+PLL tracking loop block for Glonass L2 C/A to a [TrackingInterface](#).

#### Author

Damian Miralles, 2018. [dmiralles2009@gmail.com](mailto:dmiralles2009@gmail.com)

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, Birkha user, 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

## 11.147 glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking\_cc.h File Reference

Implementation of a code DLL + carrier PLL tracking block.

```
#include "cpu_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 <pmt/pmt.h>
#include <volk_gnssssdr/volk_gnssssdr_alloc.h>
#include <deque>
#include <fstream>
#include <map>
#include <string>
```

### Classes

- class [glonass\\_l2\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_cc](#)  
*This class implements a DLL + PLL tracking loop block.*

### Typedefs

- using **glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking\_cc\_sptr** = gnss\_shared\_ptr< [glonass\\_l2\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_cc](#) >

### Functions

- glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking\_cc\_sptr **glonass\_l2\_ca\_dll\_pll\_c\_aid\_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 pll\_bw\_narrow\_hz, float dll\_bw\_narrow\_hz, int32\_t extend\_correlation\_ms, float early\_late\_space\_chips)

#### 11.147.1 Detailed Description

Implementation of a code DLL + carrier PLL tracking block.

#### Author

Damian Miralles, 2018. dmiralles2009(at)gmail.com

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, Birkha user, 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

## 11.148 glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking\_sc.h File Reference

Implementation of a code DLL + carrier PLL tracking block.

```
#include "cpu_multicorrelator_16sc.h"
#include "glonass_l2_signal_replica.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 <volk_gnssdr/volk_gnssdr_alloc.h>
#include <deque>
#include <fstream>
#include <map>
#include <string>
```

### Classes

- class [glonass\\_l2\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_sc](#)  
*This class implements a DLL + PLL tracking loop block.*

### Typedefs

- using **glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking\_sc\_sptr** = gnss\_shared\_ptr< [glonass\\_l2\\_ca\\_dll\\_pll\\_c\\_aid\\_tracking\\_sc](#) >

### Functions

- glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking\_sc\_sptr **glonass\_l2\_ca\_dll\_pll\_c\_aid\_make\_tracking\_sc** (int64\_t fs\_in, uint32\_t vector\_length, bool dump, const std::string &dump\_filename, float pll\_bw\_hz, float dll\_bw\_hz, float pll\_bw\_narrow\_hz, float dll\_bw\_narrow\_hz, int32\_t extend\_correlation\_ms, float early\_late\_space\_chips)

#### 11.148.1 Detailed Description

Implementation of a code DLL + carrier PLL tracking block.

#### Author

Damian Miralles, 2018. dmiralles2009(at)gmail.com

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, Birkha user, 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

## 11.149 glonass\_l2\_ca\_dll\_pll\_tracking.h File Reference

Interface of an adapter of a DLL+PLL tracking loop block for Glonass L2 C/A to a [TrackingInterface](#).

```
#include "glonass_l2_ca_dll_pll_tracking_cc.h"
#include "tracking_interface.h"
#include <string>
```

### Classes

- class [GlonassL2CaDllPllTracking](#)

*This class implements a code DLL + carrier PLL tracking loop.*

### 11.149.1 Detailed Description

Interface of an adapter of a DLL+PLL tracking loop block for Glonass L2 C/A to a [TrackingInterface](#).

#### Author

Damian Miralles, 2018, dmiralles2009(at)gmail.com

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, Birkha user, 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

## 11.150 glonass\_l2\_ca\_dll\_pll\_tracking\_cc.h File Reference

Implementation of a code DLL + carrier PLL tracking block.

```
#include "cpu_multicorrelator.h"
#include "gnss_block_interface.h"
#include "gnss_synchro.h"
#include "tracking_2nd_DLL_filter.h"
#include "tracking_2nd_PLL_filter.h"
#include <gnuradio/block.h>
#include <volk_gnssssdr/volk_gnssssdr_alloc.h>
#include <fstream>
#include <map>
#include <string>
```

### Classes

- class [Glonass\\_L2\\_Ca\\_Dll\\_Pll\\_Tracking\\_cc](#)

*This class implements a DLL + PLL tracking loop block.*

## Typedefs

- using **glonass\_l2\_ca\_dll\_pll\_tracking\_cc\_sptr** = gnss\_shared\_ptr< [Glonass\\_L2\\_Ca\\_Dll\\_Pll\\_Tracking\\_cc](#) >

## Functions

- glonass\_l2\_ca\_dll\_pll\_tracking\_cc\_sptr **glonass\_l2\_ca\_dll\_pll\_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)

### 11.150.1 Detailed Description

Implementation of a code DLL + carrier PLL tracking block.

#### Author

Damian Miralles, 2018. dmiralles2009(at)gmail.com

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, Birkha user, 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

### 11.151 glonass\_l2\_ca\_pcps\_acquisition.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Glonass L2 C/A signals.

```
#include "acq_conf.h"
#include "channel_fsm.h"
#include "complex_byte_to_float_x2.h"
#include "gnss_synchro.h"
#include "pcps_acquisition.h"
#include <gnuradio/blocks/float_to_complex.h>
#include <memory>
#include <string>
#include <vector>
```

## Classes

- class [GlonassL2CaPcpsAcquisition](#)

*This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GLONASS L2 C/A signals.*

### 11.151.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for Glonass L2 C/A signals.

#### Author

Damian Miralles, 2018, [dmiralles2009@gmail.com](mailto: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

## 11.152 glonass\_l2\_ca\_telemetry\_decoder.h File Reference

Interface of an adapter of a GLONASS L2 C/A NAV data decoder block to a [TelemetryDecoderInterface](#).

```
#include "glonass_l2_ca_telemetry_decoder_gs.h"
#include "gnss_satellite.h"
#include "gnss_synchro.h"
#include "telemetry_decoder_interface.h"
#include "tlm_conf.h"
#include <gnuradio/runtime_types.h>
#include <cstdio>
#include <string>
```

### Classes

- class [GlonassL2CaTelemetryDecoder](#)

*This class implements a NAV data decoder for GLONASS L2 C/A.*

### 11.152.1 Detailed Description

Interface of an adapter of a GLONASS L2 C/A NAV data decoder block to a [TelemetryDecoderInterface](#).

#### Author

Damian Miralles, 2018. [dmiralles2009\(at\)gmail.com](mailto: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

## 11.153 glonass\_l2\_ca\_telemetry\_decoder\_gs.h File Reference

Implementation of a GLONASS L2 C/A NAV data decoder block.

```
#include "GLONASS_L1_L2_CA.h"
#include "glonass_gnav_navigation_message.h"
#include "gnss_block_interface.h"
#include "gnss_satellite.h"
#include "gnss_synchro.h"
#include "tlm_conf.h"
#include <boost/circular_buffer.hpp>
#include <gnuradio/block.h>
#include <gnuradio/types.h>
#include <array>
#include <cstdint>
#include <fstream>
#include <string>
```

### Classes

- class [glonass\\_l2\\_ca\\_telemetry\\_decoder\\_gs](#)

*This class implements a block that decodes the GNAV data defined in GLONASS ICD v5.1.*

### Typedefs

- using [glonass\\_l2\\_ca\\_telemetry\\_decoder\\_gs\\_sptr](#) = gnss\_shared\_ptr<[glonass\\_l2\\_ca\\_telemetry\\_decoder\\_gs](#)>

### Functions

- [glonass\\_l2\\_ca\\_telemetry\\_decoder\\_gs\\_sptr glonass\\_l2\\_ca\\_make\\_telemetry\\_decoder\\_gs](#) (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)

#### 11.153.1 Detailed Description

Implementation of a GLONASS L2 C/A NAV data decoder block.

#### Author

Damian Miralles, 2018. [dmiralles2009\(at\)gmail.com](mailto: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

## 11.154 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/gsl>
```

### 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\\_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.*

### 11.154.1 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

## 11.155 gn3s\_signal\_source.h File Reference

GN3S USB dongle GPS RF front-end signal sampler driver.

```
#include "concurrent_queue.h"
#include "gnss_block_interface.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/hier_block2.h>
#include <pmt/pmt.h>
#include <cstdint>
#include <memory>
#include <string>
```

### Classes

- class [Gn3sSignalSource](#)  
*This class reads samples from a GN3S USB dongle, a RF front-end signal sampler.*

### 11.155.1 Detailed Description

GN3S USB dongle GPS RF front-end signal sampler driver.

#### 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

## 11.156 gnmax\_signal\_source.h File Reference

gnMAX2769 USB dongle GPS RF front-end signal sampler driver

```
#include "concurrent_queue.h"
#include "gnss_block_interface.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/hier_block2.h>
#include <pmt/pmt.h>
#include <string>
```

### Classes

- class [GnMaxSignalSource](#)

*This class reads samples from a gnMAX2769 USB dongle, a RF front-end signal sampler.*

### 11.156.1 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

## 11.157 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>
```

### Classes

- class [GNSSBlockFactory](#)  
*Class that produces all kinds of GNSS blocks.*

### 11.157.1 Detailed Description

Interface of a factory that returns smart pointers to GNSS blocks.

#### Author

Carlos Aviles, 2010. carlos.avilesr(at)gmail.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) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

## 11.158 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`)

### 11.158.1 Detailed Description

This interface represents a GNSS block.

#### Author

Carlos Aviles, 2010. [carlos.avilesr\(at\)gmail.com](mailto:carlos.avilesr(at)gmail.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

## 11.159 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 >](#)

### 11.159.1 Detailed Description

This class implements a circular deque for [Gnss\\_Synchro](#).

#### Author

Antonio Ramos, 2018. [antonio.ramosdet\(at\)gmail.com](mailto: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

## 11.160 gnss\_flowgraph.h File Reference

Interface of a GNSS receiver flow graph.

```
#include "channel_status_msg_receiver.h"
#include "concurrent_queue.h"
#include "gnss_sdr_sample_counter.h"
#include "gnss_signal.h"
#include "pvt_interface.h"
#include <gnuradio/blocks/null_sink.h>
#include <gnuradio/runtime_types.h>
#include <pmt/pmt.h>
#include <list>
#include <map>
#include <memory>
#include <mutex>
#include <string>
#include <utility>
#include <vector>
```

### Classes

- class [GNSSFlowgraph](#)

*This class represents a GNSS flow graph.*

### 11.160.1 Detailed Description

Interface of a GNSS receiver flow graph.

#### Author

Carlos Aviles, 2010. [carlos.avilesr\(at\)gmail.com](mailto:carlos.avilesr(at)gmail.com) Luis Esteve, 2011. [luis\(at\)epsilon-formacion.com](mailto:luis(at)epsilon-formacion.com) Carles Fernandez-Prades, 2014-2020. [cfernandez\(at\)cttc.es](mailto:cfernandez(at)cttc.es) Álvaro Cebrián Juan, 2018. [acebrianjuan\(at\)gmail.com](mailto: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

## 11.161 gnss\_frequencies.h File Reference

GNSS Frequencies.

## 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)*

### 11.161.1 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

### 11.162 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 int32\_t [MAXCODE](#) = 55  
*max number of obs code*

### 11.162.1 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

## 11.163 gnss\_satellite.h File Reference

Interface of the [Gnss\\_Satellite](#) class.

```
#include <stdint>
#include <map>
#include <ostream>
#include <set>
#include <string>
```

### Classes

- class [Gnss\\_Satellite](#)  
*This class represents a GNSS satellite.*

### 11.163.1 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

## 11.164 gnss\_sdr\_create\_directory.h File Reference

Create a directory.

```
#include <string>
```

### Functions

- bool **gnss\_sdr\_create\_directory** (const std::string &foldername)

### 11.164.1 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

## 11.165 gnss\_sdr\_flags.h File Reference

Helper file for gnss-sdr commandline flags.

```
#include <gflags/gflags.h>
#include <cstdint>
```

## Functions

- [DECLARE\\_string](#) (c)  
*Path to the configuration file.*
- [DECLARE\\_string](#) (config\_file)  
*Path to the configuration file.*
- [DECLARE\\_string](#) (log\_dir)  
*Path to the folder in which logging will be stored.*
- [DECLARE\\_string](#) (s)  
*Path to the file containing the signal samples.*
- [DECLARE\\_string](#) (signal\_source)  
*Path to the file containing the signal samples.*
- [DECLARE\\_bool](#) (rf\_shutdown)  
*Shutdown RF when program exits.*
- [DECLARE\\_int32](#) (doppler\_max)  
*If defined, maximum Doppler value in the search grid, in Hz (overrides the configuration file).*
- [DECLARE\\_int32](#) (doppler\_step)  
*If defined, sets the frequency step in the search grid, in Hz, in Hz (overrides the configuration file).*
- [DECLARE\\_int32](#) (cn0\_samples)  
*Number of correlator outputs used for CN0 estimation.*
- [DECLARE\\_int32](#) (cn0\_min)  
*Minimum valid CN0 (in dB-Hz).*
- [DECLARE\\_int32](#) (max\_lock\_fail)  
*Maximum number of code lock failures before dropping a satellite.*
- [DECLARE\\_int32](#) (max\_carrier\_lock\_fail)  
*Maximum number of carrier lock failures before dropping a satellite.*
- [DECLARE\\_double](#) (carrier\_lock\_th)  
*Carrier lock threshold (in rad).*
- [DECLARE\\_double](#) (dll\_bw\_hz)  
*Bandwidth of the DLL low pass filter, in Hz (overrides the configuration file).*
- [DECLARE\\_double](#) (pll\_bw\_hz)  
*Bandwidth of the PLL low pass filter, in Hz (overrides the configuration file).*
- [DECLARE\\_int32](#) (carrier\_smoothing\_factor)  
*Sets carrier smoothing factor M (overrides the configuration file).*
- [DECLARE\\_string](#) (RINEX\_version)  
*If defined, specifies the RINEX version (2.11 or 3.02). Overrides the configuration file.*
- [DECLARE\\_string](#) (RINEX\_name)  
*If defined, specifies the RINEX files base name.*

## Variables

- `const int32_t DEFAULT_CARRIER_SMOOTHING_FACTOR = 200`

### 11.165.1 Detailed Description

Helper file for gnss-sdr commandline flags.

#### 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

## 11.166 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)

#### 11.166.1 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

## 11.167 gnss\_sdr\_make\_unique.h File Reference

This file implements std::make\_unique for C++11.

#### 11.167.1 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

## 11.168 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 <cstdio>
#include <stdint>
```

### 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 \_t\_size)

#### 11.168.1 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

## 11.169 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 <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.*

### 11.169.1 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

## 11.170 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** ()

### 11.170.1 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

## 11.171 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 <cstdio>
#include <stdint>
```

## 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)

### 11.171.1 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)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

## 11.172 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.*

### 11.172.1 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

## 11.173 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/gsl>
```

## 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.*

### 11.173.1 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

## 11.174 gnss\_synchro.h File Reference

Interface of the [Gnss\\_Synchro](#) class.

```
#include <boost/serialization/nvp.hpp>
#include <stdint>
#include <utility>
```

## Classes

- class [Gnss\\_Synchro](#)  
*This is the class that contains the information that is shared by the processing blocks.*

### 11.174.1 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

## 11.175 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, int udp\_port, const std::vector< std::string > &udp\_addresses, bool enable\_protobuf)

### 11.175.1 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

## 11.176 gnss\_synchro\_udp\_sink.h File Reference

Interface of a class that sends serialized [Gnss\\_Synchro](#) objects over udp to one or multiple endpoints.

```
#include "gnss_synchro.h"
#include "serdes_gnss_synchro.h"
#include <boost/asio.hpp>
#include <boost/system/error_code.hpp>
#include <stdint>
#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

#### 11.176.1 Detailed Description

Interface of a class that sends serialized [Gnss\\_Synchro](#) objects over udp to one or multiple endpoints.

#### Author

Álvaro Cebrián Juan, 2018. [acebrianjuan\(at\)gmail.com](mailto: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

## 11.177 gnuplot\_i.h File Reference

A C++ interface to gnuplot.

```
#include <gflags/gflags.h>
#include <cmath>
#include <cstdlib>
#include <cstring>
#include <fstream>
#include <iostream>
#include <list>
#include <sstream>
#include <stdexcept>
#include <string>
#include <sys/stat.h>
#include <vector>
```

## Classes

- class [GnuplotException](#)
- class [Gnuplot](#)

## Functions

- **DEFINE\_bool** (show\_plots, true, "Show plots on screen. Disable for non-interactive testing.")
- `template<typename Container >`  
void **stringtok** (Container &container, std::string const &in, const char \*const delimiters=" \")

### 11.177.1 Detailed Description

A C++ interface to gnuplot.

#### Author

Carles Fernandez-Prades, 2017. cfernandez(at)cttc.es

Original source code found at <https://code.google.com/archive/p/gnuplot-cpp/> by Jeremy Conlin jeremit0(at)gmail.com

Version history: 0. C interface by N. Devillard (27/01/03)

1. C++ interface: direct translation from the C interface by Rajarshi Guha (07/03/03)
2. corrections for Win32 compatibility by V. Chyzhdzenka (20/05/03)
3. some member functions added, corrections for Win32 and Linux compatibility by M. Burgis (10/03/08)
4. Some fixes and improvements for Linux and macOS by C. Fernandez (22/10/17)

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

### 11.178 `gps_acq_assist.h` File Reference

Interface of a GPS RRL ACQUISITION ASSISTANCE storage.

```
#include <cstdint>
```

## Classes

- class [Gps\\_Acq\\_Assist](#)

*This class is a storage for the GPS GSM RRL ACQUISITION ASSISTANCE data as described in Digital cellular telecommunications system (Phase 2+); Location Services (LCS); Mobile Station (MS) - Serving Mobile Location Centre (SM-LC) Radio Resource LCS Protocol (RRLP) (3GPP TS 44.031 version 5.12.0 Release 5)*

### 11.178.1 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

## 11.179 `gps_almanac.h` File Reference

Interface of a GPS ALMANAC storage.

```
#include <boost/serialization/nvp.hpp>
#include <cstdint>
```

### Classes

- class [Gps\\_Almanac](#)

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

### 11.179.1 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

## 11.180 `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_NO {{{133, 17}}}`
- `const std::vector< std::pair< int32_t, int32_t > > CNAV_DELTA_NO_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_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 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\_TLSE\_LSB** = 1

### 11.180.1 Detailed Description

Defines parameters for GPS CNAV.

#### Author

Antonio Ramos, 2017. [antonio.ramos\(at\)cttc.es](mailto: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

## 11.181 `gps_cnav_ephemeris.h` File Reference

Interface of a GPS CNAV EPHEMERIS storage.

```
#include <boost/serialization/nvp.hpp>
#include <cstdint>
```

### Classes

- class [Gps\\_CNAV\\_Ephemeris](#)

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

### 11.181.1 Detailed Description

Interface of a GPS CNAV EPHEMERIS storage.

#### Author

Javier Arribas, 2015. [jarribas\(at\)cttc.es](mailto: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

## 11.182 `gps_cnav_iono.h` File Reference

Interface of a GPS CNAV IONOSPHERIC MODEL storage.

```
#include <boost/serialization/nvp.hpp>
```

## Classes

- class [Gps\\_CNAV\\_Iono](#)

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

### 11.182.1 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

## 11.183 gps\_cnav\_navigation\_message.h File Reference

Interface of a GPS CNAV Data message decoder.

```
#include "GPS_CNAV.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-200K.*

### 11.183.1 Detailed Description

Interface of a GPS CNAV Data message decoder.

#### 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

## 11.184 `gps_cnav_utc_model.h` File Reference

Interface of a GPS CNAV UTC MODEL storage.

```
#include <boost/serialization/nvp.hpp>
#include <cstdint>
```

### Classes

- class [Gps\\_CNAV\\_Utc\\_Model](#)

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

### 11.184.1 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

## 11.185 `gps_ephemeris.h` File Reference

Interface of a GPS EPHEMERIS storage.

```
#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-200K.*

### 11.185.1 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

## 11.186 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-200K.*

### 11.186.1 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

## 11.187 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 {{{271, 1}}}`
- `const std::vector< std::pair< int32_t, int32_t > > AODO {{{272, 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 > > 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}}}

## 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`



### 11.187.1 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

## 11.188 gps\_l1\_ca\_dll\_pll\_tracking.h File Reference

Interface of an adapter of a DLL+PLL tracking loop block for GPS L1 C/A to a [TrackingInterface](#).

```
#include "dll_pll_veml_tracking.h"
#include "tracking_interface.h"
#include <string>
```

### Classes

- class [GpsL1CaDllPllTracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*

### 11.188.1 Detailed Description

Interface of an adapter of a DLL+PLL tracking loop block for GPS L1 C/A to a [TrackingInterface](#).

#### Author

Carlos Aviles, 2010. carlos.avilesr(at)gmail.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

## 11.189 gps\_l1\_ca\_dll\_pll\_tracking\_fpga.h File Reference

Interface of an adapter of a DLL+PLL tracking loop block for GPS L1 C/A to a [TrackingInterface](#) for the FPGA.

```
#include "dll_pll_veml_tracking_fpga.h"
#include "tracking_interface.h"
#include <string>
```

## Classes

- class [GpsL1CaDllPllTrackingFpga](#)

*This class implements a code DLL + carrier PLL tracking loop.*

### 11.189.1 Detailed Description

Interface of an adapter of a DLL+PLL tracking loop block for GPS L1 C/A to a [TrackingInterface](#) for the FPGA.

#### Author

Marc Majoral, 2019, mmajoral(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

## 11.190 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.*

### 11.190.1 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

## 11.191 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)

#### 11.191.1 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

## 11.192 `gps_l1_ca_kf_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_kf_tracking_cc.h"
#include "tracking_interface.h"
#include <string>
```

### Classes

- class [GpsL1CaKfTracking](#)

*This class implements a code DLL + carrier PLL tracking loop.*

### 11.192.1 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](mailto:jarribas(at)cttc.es)

Jordi Vila-Valls 2018. [jvila\(at\)cttc.es](mailto:jvila(at)cttc.es)

Carles Fernandez-Prades 2018. [cfernandez\(at\)cttc.es](mailto: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

## 11.193 `gps_l1_ca_kf_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_gnssssdr/volk_gnssssdr_alloc.h>
#include <fstream>
#include <map>
#include <string>
```

## Classes

- class [Gps\\_L1\\_Ca\\_Kf\\_Tracking\\_cc](#)  
*This class implements a DLL + PLL tracking loop block.*

## Typedefs

- using **gps\_l1\_ca\_kf\_tracking\_cc\_sptr** = gnss\_shared\_ptr< [Gps\\_L1\\_Ca\\_Kf\\_Tracking\\_cc](#) >

## Functions

- `gps_l1_ca_kf_tracking_cc_sptr gps_l1_ca_kf_make_tracking_cc (uint32_t order, int64_t if_freq, 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)`

### 11.193.1 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

## 11.194 gps\_l1\_ca\_pcps\_acquisition.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

```
#include "acq_conf.h"
#include "channel_fsm.h"
#include "complex_byte_to_float_x2.h"
#include "gnss_synchro.h"
#include "pcps_acquisition.h"
#include <gnuradio/blocks/float_to_complex.h>
#include <memory>
#include <string>
#include <vector>
```

## Classes

- class [GpsL1CaPcpsAcquisition](#)

*This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.*

### 11.194.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

#### Authors

- Javier Arribas, 2011. jarribas(at)cttc.es
- Luis Esteve, 2012. luis(at)epsilon-formacion.com
- 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

## 11.195 gps\_l1\_ca\_pcps\_acquisition\_fine\_doppler.h File Reference

Adapts a PCPS acquisition block with fine Doppler estimation to an [AcquisitionInterface](#) for GPS L1 C/A signals.

```
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include "pcps_acquisition_fine_doppler_cc.h"
#include <memory>
#include <string>
#include <vector>
```

## Classes

- class [GpsL1CaPcpsAcquisitionFineDoppler](#)

*This class Adapts a PCPS acquisition block with fine Doppler estimation to an [AcquisitionInterface](#) for GPS L1 C/A signals.*

## Typedefs

- using [pcps\\_acquisition\\_fine\\_doppler\\_cc\\_sptr](#) = gnss\_shared\_ptr< [pcps\\_acquisition\\_fine\\_doppler\\_cc](#) >

### 11.195.1 Detailed Description

Adapts a PCPS acquisition block with fine Doppler estimation to an [AcquisitionInterface](#) for GPS L1 C/A signals.

#### 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

## 11.196 gps\_l1\_ca\_pcps\_acquisition\_fpga.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals for the FPGA.

```
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include "pcps_acquisition_fpga.h"
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [GpsL1CaPcpsAcquisitionFpga](#)

*This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for GPS L1 C/A signals.*

### 11.196.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals for the FPGA.

### 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) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

## 11.197 gps\_l1\_ca\_pcps\_assisted\_acquisition.h File Reference

Adapts a PCPS Assisted acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

```
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include "pcps_assisted_acquisition_cc.h"
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [GpsL1CaPcpsAssistedAcquisition](#)

*This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.*

### 11.197.1 Detailed Description

Adapts a PCPS Assisted acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

#### Authors

- 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

## 11.198 gps\_l1\_ca\_pcps\_opengl\_acquisition.h File Reference

Adapts an OpenGL PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

```
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include "pcps_opengl_acquisition_cc.h"
#include <gnuradio/blocks/stream_to_vector.h>
#include <memory>
#include <string>
#include <vector>
```

#### Classes

- class [GpsL1CaPcpsOpenCLAcquisition](#)

*This class adapts an OpenGL PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.*

### 11.198.1 Detailed Description

Adapts an OpenGL PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

#### 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

## 11.199 gps\_l1\_ca\_pcps\_quicksync\_acquisition.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals implementing the QuickSync Algorithm.

```
#include "channel_fsm.h"
#include "configuration_interface.h"
#include "gnss_synchro.h"
#include "pcps_quicksync_acquisition_cc.h"
#include <gnuradio/blocks/stream_to_vector.h>
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [GpsL1CaPcpsQuickSyncAcquisition](#)

*This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.*

### 11.199.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals implementing the QuickSync Algorithm.

#### Date

June, 2014

#### Author

Damian Miralles Sanchez. [dmiralles2009@gmail.com](mailto: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

## 11.200 gps\_l1\_ca\_pcps\_tong\_acquisition.h File Reference

Adapts a PCPS Tong acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

```
#include "channel_fsm.h"
#include "configuration_interface.h"
#include "gnss_synchro.h"
#include "pcps_tong_acquisition_cc.h"
#include <gnuradio/blocks/stream_to_vector.h>
#include <memory>
#include <string>
#include <vector>
```

## Classes

- class [GpsL1CaPcpsTongAcquisition](#)

*This class adapts a PCPS Tong acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.*

### 11.200.1 Detailed Description

Adapts a PCPS Tong acquisition block to an [AcquisitionInterface](#) for GPS L1 C/A signals.

#### 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

## 11.201 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.*

### 11.201.1 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

## 11.202 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_gnssssdr/volk_gnssssdr_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)

#### 11.202.1 Detailed Description

Interface of a TCP connector block based on code DLL + carrier PLL.

#### Author

David Pubill, 2012. [dpubill\(at\)cttc.es](mailto:dpubill@cttc.es) Javier Arribas, 2011. [jarribas\(at\)cttc.es](mailto: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

## 11.203 `gps_l1_ca_telemetry_decoder.h` File Reference

Interface of an adapter of a GPS L1 C/A NAV data decoder block to a [TelemetryDecoderInterface](#).

```
#include "gnss_satellite.h"
#include "gnss_synchro.h"
#include "gps_l1_ca_telemetry_decoder_gs.h"
#include "telemetry_decoder_interface.h"
#include "tlm_conf.h"
#include <gnuradio/runtime_types.h>
#include <cstdint>
#include <string>
```

### Classes

- class [GpsL1CaTelemetryDecoder](#)

*This class implements a NAV data decoder for GPS L1 C/A.*

### 11.203.1 Detailed Description

Interface of an adapter of a GPS L1 C/A NAV data decoder block to a [TelemetryDecoderInterface](#).

#### 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

## 11.204 `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_L1_CA.h"
#include "gnss_block_interface.h"
#include "gnss_satellite.h"
#include "gnss_synchro.h"
#include "gps_navigation_message.h"
#include "tlm_conf.h"
#include <boost/circular_buffer.hpp>
#include <gnuradio/block.h>
#include <gnuradio/types.h>
#include <array>
#include <stdint>
#include <fstream>
#include <string>
```

## Classes

- class [gps\\_l1\\_ca\\_telemetry\\_decoder\\_gs](#)  
*This class implements a block that decodes the NAV data defined in IS-GPS-200K.*

## Typedefs

- using [gps\\_l1\\_ca\\_telemetry\\_decoder\\_gs\\_sptr](#) = gnss\_shared\_ptr< [gps\\_l1\\_ca\\_telemetry\\_decoder\\_gs](#) >

## Functions

- [gps\\_l1\\_ca\\_telemetry\\_decoder\\_gs\\_sptr](#) [gps\\_l1\\_ca\\_make\\_telemetry\\_decoder\\_gs](#) (const [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)

### 11.204.1 Detailed Description

Interface of a NAV message demodulator block based on Kay Borre book MATLAB-based GPS receiver.

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

## 11.205 gps\_l2\_m\_dll\_pll\_tracking.h File Reference

Interface of an adapter of a DLL+PLL tracking loop block for GPS L1 C/A to a [TrackingInterface](#).

```
#include "dll_pll_veml_tracking.h"
#include "tracking_interface.h"
#include <string>
```

## Classes

- class [GpsL2MDIIPITracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*

### 11.205.1 Detailed Description

Interface of an adapter of a DLL+PLL tracking loop block for GPS L1 C/A to a [TrackingInterface](#).

#### Author

Carlos Aviles, 2010. carlos.avilesr(at)gmail.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

## 11.206 gps\_l2\_m\_dll\_pll\_tracking\_fpga.h File Reference

Interface of an adapter of a DLL+PLL tracking loop block for GPS L2C to a [TrackingInterface](#) for the FPGA.

```
#include "dll_pll_veml_tracking_fpga.h"
#include "tracking_interface.h"
#include <gnuradio/runtime_types.h>
#include <cstring>
#include <string>
```

### Classes

- class [GpsL2MDIIPITrackingFpga](#)  
*This class implements a code DLL + carrier PLL tracking loop.*

### 11.206.1 Detailed Description

Interface of an adapter of a DLL+PLL tracking loop block for GPS L2C to a [TrackingInterface](#) for the FPGA.

#### Author

Marc Majoral, 2019, mmajoral(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

## 11.207 gps\_l2\_m\_pcps\_acquisition.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L2 M signals.

```
#include "channel_fsm.h"
#include "complex_byte_to_float_x2.h"
#include "gnss_synchro.h"
#include "pcps_acquisition.h"
#include <gnuradio/blocks/float_to_complex.h>
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [GpsL2MPcpsAcquisition](#)  
*This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L2 M signals.*

### 11.207.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L2 M signals.

#### Authors

- 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

## 11.208 gps\_l2\_m\_pcps\_acquisition\_fpga.h File Reference

Adapts an FPGA-offloaded PCPS acquisition block to an [AcquisitionInterface](#) for GPS L2 M signals.

```
#include "channel_fsm.h"
#include "pcps_acquisition_fpga.h"
#include <gnuradio/runtime_types.h>
#include <cstdio>
#include <memory>
#include <string>
#include <vector>
```

#### Classes

- class [GpsL2MPcpsAcquisitionFpga](#)

*This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for GPS L2 M signals.*

### 11.208.1 Detailed Description

Adapts an FPGA-offloaded PCPS acquisition block to an [AcquisitionInterface](#) for GPS L2 M signals.

#### Authors

- 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

## 11.209 GPS\_L2C.h File Reference

Defines system parameters for GPS L2C signal.

```
#include "GPS_CNAV.h"
#include "gnss_frequencies.h"
#include <stdint>
#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_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 int32\_t `GPS_L2_M_CODE_LENGTH_CHIPS` = 10230  
*GPS L2 M code length [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]

### 11.209.1 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

### 11.210 `gps_l2c_signal_replica.h` File Reference

This file implements signal generators for GPS L2C signals.

```
#include <complex>
#include <cstdint>
#include <gsl/gsl>
```

## 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\\_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.*

### 11.210.1 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

## 11.211 gps\_l2c\_telemetry\_decoder.h File Reference

Interface of an adapter of a GPS L2C (CNAV) data decoder block to a [TelemetryDecoderInterface](#).

```
#include "gnss_satellite.h"
#include "gnss_synchro.h"
#include "gps_l2c_telemetry_decoder_gs.h"
#include "telemetry_decoder_interface.h"
#include "tlm_conf.h"
#include <gnuradio/runtime_types.h>
#include <cstring>
#include <string>
```

## Classes

- class [GpsL2CTelemetryDecoder](#)  
*This class implements a NAV data decoder for GPS L2 M.*

### 11.211.1 Detailed Description

Interface of an adapter of a GPS L2C (CNAV) data decoder block to a [TelemetryDecoderInterface](#).

#### 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

## 11.212 `gps_l2c_telemetry_decoder_gs.h` File Reference

Interface of a CNAV message demodulator block.

```
#include "gnss_block_interface.h"
#include "gnss_satellite.h"
#include "gps_cnav_navigation_message.h"
#include "tlm_conf.h"
#include <gnuradio/block.h>
#include <gnuradio/types.h>
#include <cstdint>
#include <fstream>
#include <string>
#include "cnav_msg.h"
```

### Classes

- class [gps\\_l2c\\_telemetry\\_decoder\\_gs](#)

*This class implements a block that decodes CNAV data defined in IS-GPS-200K.*

### 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 [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)

#### 11.212.1 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

## 11.213 `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, 1, 1, 1, 0}
- constexpr char `GPS_L5I_NH_CODE_STR` [11] = "0000110101"
- constexpr char `GPS_L5Q_NH_CODE_STR` [21] = "00000100110101001110"

### 11.213.1 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

## 11.214 gps\_l5\_dll\_pll\_tracking.h File Reference

Interface of an adapter of a DLL+PLL tracking loop block for GPS L5 to a [TrackingInterface](#).

```
#include "dll_pll_veml_tracking.h"  
#include "tracking_interface.h"  
#include <string>
```

### Classes

- class [GpsL5DIIPITracking](#)  
*This class implements a code DLL + carrier PLL tracking loop.*

### 11.214.1 Detailed Description

Interface of an adapter of a DLL+PLL tracking loop block for GPS L5 to a [TrackingInterface](#).

#### Author

Javier Arribas, 2017. 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

## 11.215 gps\_l5\_dll\_pll\_tracking\_fpga.h File Reference

Interface of an adapter of a DLL+PLL tracking loop block for GPS L5 to a [TrackingInterface](#) for the FPGA.

```
#include "dll_pll_veml_tracking_fpga.h"  
#include "tracking_interface.h"  
#include <string>
```

### Classes

- class [GpsL5DIIPITrackingFpga](#)  
*This class implements a code DLL + carrier PLL tracking loop.*

### 11.215.1 Detailed Description

Interface of an adapter of a DLL+PLL tracking loop block for GPS L5 to a [TrackingInterface](#) for the FPGA.

#### Author

Marc Majoral, 2019. mmajoral(at)cttc.cat Javier Arribas, 2019. 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

## 11.216 gps\_l5\_signal\_replica.h File Reference

This file implements signal generators for GPS L5 signals.

```
#include <complex>
#include <cstdint>
#include <gsl/gsl>
```

### 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.*

### 11.216.1 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

## 11.217 `gps_l5_telemetry_decoder.h` File Reference

Interface of an adapter of a GPS L5 (CNAV) data decoder block to a [TelemetryDecoderInterface](#).

```
#include "gnss_satellite.h"
#include "gnss_synchro.h"
#include "gps_l5_telemetry_decoder_gs.h"
#include "telemetry_decoder_interface.h"
#include "tlm_conf.h"
#include <gnuradio/runtime_types.h>
#include <cstdint>
#include <string>
```

### Classes

- class [GpsL5TelemetryDecoder](#)

*This class implements a NAV data decoder for GPS L5.*

### 11.217.1 Detailed Description

Interface of an adapter of a GPS L5 (CNAV) data decoder block to a [TelemetryDecoderInterface](#).

#### 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

## 11.218 `gps_l5_telemetry_decoder_gs.h` File Reference

Interface of a CNAV message demodulator block.

```
#include "GPS_L5.h"
#include "gnss_block_interface.h"
#include "gnss_satellite.h"
#include "gps_cnav_navigation_message.h"
#include "tlm_conf.h"
#include <boost/circular_buffer.hpp>
#include <gnuradio/block.h>
#include <gnuradio/types.h>
#include <cstdint>
#include <fstream>
#include <string>
#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 [Gnss\\_Satellite](#) &satellite, const [Tlm\\_Conf](#) &conf)

### 11.218.1 Detailed Description

Interface of a CNAV message demodulator 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

## 11.219 gps\_l5i\_pcps\_acquisition.h File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L5i signals.

```
#include "channel_fsm.h"
#include "complex_byte_to_float_x2.h"
#include "gnss_synchro.h"
#include "pcps_acquisition.h"
#include <gnuradio/blocks/float_to_complex.h>
#include <memory>
#include <string>
#include <vector>
```

## Classes

- class [GpsL5iPcpsAcquisition](#)  
*This class adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L5i signals.*

### 11.219.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L5i signals.

## Authors

- 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

## 11.220 `gps_l5i_pcps_acquisition_fpga.h` File Reference

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L5i signals for the FPGA.

```
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include "pcps_acquisition_fpga.h"
#include <memory>
#include <string>
#include <vector>
```

### Classes

- class [GpsL5iPcpsAcquisitionFpga](#)

*This class adapts a PCPS acquisition block off-loaded on an FPGA to an [AcquisitionInterface](#) for GPS L5i signals.*

### 11.220.1 Detailed Description

Adapts a PCPS acquisition block to an [AcquisitionInterface](#) for GPS L5i signals for the FPGA.

#### 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) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

## 11.221 `gps_navigation_message.h` File Reference

Interface of a GPS NAV Data message decoder.

```
#include "GPS_L1_CA.h"
#include "gps_ephemeris.h"
#include "gps_iono.h"
#include "gps_utc_model.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-200K.*

### 11.221.1 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

## 11.222 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/gsl>
```

### 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.*

### 11.222.1 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

## 11.223 `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-200K.*

### 11.223.1 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

## 11.224 `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.*

### 11.224.1 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

## 11.225 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](#)

#### 11.225.1 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

## 11.226 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 <cstdlib>
#include <string>
```

### Classes

- class [HybridObservables](#)

*This class implements an [ObservablesInterface](#) for observables of all kind of GNSS signals.*

### 11.226.1 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

## 11.227 `hybrid_observables_gs.h` File Reference

Interface of the observables computation block.

```
#include "gnss_block_interface.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 <map>
#include <memory>
#include <string>
#include <typeinfo>
#include <vector>
```

### 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\_)

### 11.227.1 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

## 11.228 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 [lbyteToCbyte](#)

### 11.228.1 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

## 11.229 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 [lbyteToComplex](#)

*Adapts an I/Q interleaved byte integer sample stream to a gr\_complex (float) stream.*

### 11.229.1 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

## 11.230 lbyte\_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 [lbyteToCshort](#)

*Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream.*

### 11.230.1 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

## 11.231 in\_memory\_configuration.h File Reference

A [ConfigurationInterface](#) for testing purposes.

```
#include "configuration_interface.h"
#include "string_converter.h"
#include <cstdlib>
#include <map>
#include <memory>
#include <string>
```

### Classes

- class [InMemoryConfiguration](#)

*This class is an implementation of the interface [ConfigurationInterface](#).*

### 11.231.1 Detailed Description

A [ConfigurationInterface](#) for testing purposes.

#### Author

Carlos Aviles, 2010. carlos.avilesr(at)gmail.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

## 11.232 ini.h File Reference

This function parses an INI file into easy-to-access name/value pairs.

### 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 pairs (whitespace stripped), and comments starting with ';' (semicolon). Section is "" if name=value pair parsed before any section heading.*

### 11.232.1 Detailed Description

This function parses 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. Parse given INI-style file. May have [section]s, name=value pairs (whitespace stripped), and comments starting with ';' (semicolon). Section is "" if name=value pair parsed before any section heading. For each name=value pair 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 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:

## 11.233 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.)*

### 11.233.1 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:

## 11.234 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** ()

#### 11.234.1 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

## 11.235 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** ()

### 11.235.1 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

## 11.236 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** ()

### 11.236.1 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

## 11.237 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 <stdint>
#include <string>
```

### Classes

- class [lshortToComplex](#)

*Adapts an I/Q interleaved short integer sample stream to a `gr_complex` (float) stream.*

### 11.237.1 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

## 11.238 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 <stdint>
#include <string>
```

## Classes

- class [lshortToCshort](#)

*Adapts a short integer (16 bits) interleaved sample stream into a `std::complex<short>` stream.*

### 11.238.1 Detailed Description

Adapts a short interleaved sample stream into a `std::complex<short>` stream.

#### Author

Carles Fernandez-Prades, [cfernandez\(at\)cttc.es](mailto: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

## 11.239 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.*

### 11.239.1 Detailed Description

Utility functions for converting between item types.

#### Authors

- Cillian O'Driscoll, 2019. [cillian.odriscoll\(at\)gmail.com](mailto: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

## 11.240 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)*

### 11.240.1 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

## 11.241 labsat23\_source.h File Reference

Unpacks the Labsat 2 (ls2) and (ls3) capture files.

```
#include "concurrent_queue.h"
#include "gnss_block_interface.h"
#include <gnuradio/block.h>
#include <pmt/pmt.h>
#include <stdint>
#include <fstream>
#include <string>
```

### Classes

- class [labsat23\\_source](#)

*This class implements conversion between Labsat2 and 3 format byte packet samples 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, int channel\_selector, [Concurrent\\_Queue](#)< pmt::pmt\_t > \*queue)

### 11.241.1 Detailed Description

Unpacks the Labsat 2 (ls2) and (ls3) capture files.

#### 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

## 11.242 labsat\_signal\_source.h File Reference

Labsat 2 and 3 front-end signal sampler driver.

```
#include "concurrent_queue.h"
#include "gnss_block_interface.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>
```

## Classes

- class [LabsatSignalSource](#)

*This class reads samples stored by a LabSat 2 or LabSat 3 device.*

### 11.242.1 Detailed Description

Labsat 2 and 3 front-end signal sampler driver.

#### 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

## 11.243 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](#) (gr\_complex \*Prompt\_buffer, int length)  
*A carrier lock detector.*

### 11.243.1 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

## 11.244 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-200K, pag 92.*
- constexpr double `GPS_F` = -4.442807633e-10  
*Constant, [ $s/(m)^{1/2}$ ], IS-GPS-200K, pag. 92.*
- constexpr double `GALILEO_GM` = 3.986004418e14  
*Geocentric gravitational constant [ $m^3/s^2$ ], OS SIS ICD v1.3, pag. 44.*
- constexpr double `GALILEO_F` = -4.442807309e-10  
*Constant, [ $s/(m)^{1/2}$ ], OS SIS ICD v1.3, 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_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](#) = 0.00003051757813
- $2^{-15}$ 
  - constexpr double [TWO\\_N16](#) = 0.0000152587890625
- $2^{-16}$ 
  - 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\\_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\\_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.*

### 11.244.1 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

## 11.245 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`.*

### 11.245.1 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

## 11.246 monitor\_pvt.h File Reference

Interface of the [Monitor\\_Pvt](#) class.

```
#include <boost/serialization/nvp.hpp>
#include <cstdint>
```

## Classes

- class [Monitor\\_Pvt](#)

*This class contains parameters and outputs of the PVT block.*

### 11.246.1 Detailed Description

Interface of the [Monitor\\_Pvt](#) class.

#### Author

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

## 11.247 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

### 11.247.1 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](mailto: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

## 11.248 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 <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`.*

### 11.248.1 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

## 11.249 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).*

### 11.249.1 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

## 11.250 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](#)

### 11.250.1 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

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

## 11.251 notch\_cc.h File Reference

Implements a notch filter algorithm.

```
#include "gnss_block_interface.h"
#include <gnuradio/block.h>
#include <gnuradio/fft/fft.h>
#include <volk_gnssssdr/volk_gnssssdr_alloc.h>
#include <stdint>
#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)

#### 11.251.1 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

## 11.252 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](#)

### 11.252.1 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

## 11.253 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](#)

### 11.253.1 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

## 11.254 notch\_lite\_cc.h File Reference

Implements a notch filter light algorithm.

```
#include "gnss_block_interface.h"
#include <gnuradio/block.h>
#include <gnuradio/fft/fft.h>
#include <volk_gnssssdr/volk_gnssssdr_alloc.h>
#include <stdint>
#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)

#### 11.254.1 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

## 11.255 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.ifenc.com/products/sx-scientific-gnss-solutions/nsr-software-receiver.html>.

```
#include "concurrent_queue.h"
#include "gnss_block_interface.h"
#include "unpack_byte_2bit_samples.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 <string>
```

## Classes

- class [NsrFileSignalSource](#)

*Class that reads signals samples from a file and adapts it to a `SignalSourceInterface`.*

### 11.255.1 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.fr/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) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

## 11.256 obs\_conf.h File Reference

Class that contains all the configuration parameters for generic observables block.

```
#include <cstdint>
#include <string>
```

## Classes

- class [Obs\\_Conf](#)

### 11.256.1 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

## 11.257 obsdiff\_flags.h File Reference

Helper file for unit testing.

```
#include <gflags/gflags.h>
```

### Functions

- **DEFINE\_double** (skip\_obs\_transitory\_s, 30.0, "Skip the initial observable outputs to avoid transitory results [s]")
- **DEFINE\_double** (skip\_obs\_ends\_s, 5.0, "Skip the lasts observable outputs to avoid transitory results [s]")
- **DEFINE\_bool** (single\_diffs, false, "Compute also the single difference errors for Accumulated Carrier Phase and Carrier Doppler (requires LO synchronization between receivers)")
- **DEFINE\_bool** (compare\_with\_5X, false, "Compare the E5a Doppler and Carrier Phases with the E5 full bw in RINEX (expect discrepancy due to the center frequencies difference)")
- **DEFINE\_bool** (dupli\_sat, false, "Enable special observable test mode where the scenario contains duplicated satellite orbits")
- **DEFINE\_bool** (single\_diff, false, "Enable special observable test mode using only rover observables")
- **DEFINE\_string** (dupli\_sat\_prns, "1,2,3,4", "List of duplicated satellites PRN pairs (i.e. 1,2,3,4 indicates that the PRNs 1,2 share the same orbit. The same applies for PRNs 3,4)")
- **DEFINE\_string** (base\_rinex\_obs, "base.obs", "Filename of reference RINEX observation file")
- **DEFINE\_string** (rinex\_nav, "base.nav", "Filename of reference RINEX navigation file")
- **DEFINE\_string** (rover\_rinex\_obs, "base.obs", "Filename of test RINEX observation file")
- **DEFINE\_string** (system, "G", "GNSS satellite system: G for GPS, E for Galileo")
- **DEFINE\_string** (signal, "1C", "GNSS signal: 1C for GPS L1 CA, 1B for Galileo E1")
- **DEFINE\_bool** (remove\_rx\_clock\_error, false, "Compute and remove the receivers clock error prior to compute observable differences (requires a valid RINEX nav file for both receivers)")

### 11.257.1 Detailed Description

Helper file for unit testing.

#### 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

## 11.258 observable\_tests\_flags.h File Reference

Helper file for unit testing.

```
#include <gflags/gflags.h>
#include <limits>
```

## Functions

- **DEFINE\_double** (skip\_obs\_transitory\_s, 30.0, "Skip the initial observable outputs to avoid transitory results [s]")
- **DEFINE\_bool** (compute\_single\_diffs, false, "Compute also the single difference errors for Accumulated Carrier Phase and Carrier Doppler (requires LO synchronization between receivers)")
- **DEFINE\_bool** (compare\_with\_5X, false, "Compare the E5a Doppler and Carrier Phases with the E5 full bw in RINEX (expect discrepancy due to the center frequencies differences)")
- **DEFINE\_bool** (duplicated\_satellites\_test, false, "Enable special observable test mode where the scenario contains duplicated satellite orbits")
- **DEFINE\_string** (duplicated\_satellites\_prns, "1,2,3,4", "List of duplicated satellites PRN pairs (i.e. 1,2,3,4 indicates that the PRNs 1,2 share the same orbit. The same applies for PRNs 3,4)")

### 11.258.1 Detailed Description

Helper file for unit testing.

#### 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

## 11.259 observables\_dump\_reader.h File Reference

Helper file for unit testing.

```
#include <cstdint>
#include <fstream>
#include <string>
#include <vector>
```

## Classes

- class [Observables\\_Dump\\_Reader](#)

### 11.259.1 Detailed Description

Helper file for unit testing.

#### 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

## 11.260 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.*

### 11.260.1 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

## 11.261 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 "gnss_block_interface.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>)*

### 11.261.1 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

## 11.262 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 <cstdio>
#include <string>
```

### Classes

- class [Pass\\_Through](#)

*This class implements a block that connects input and output (does nothing)*

### 11.262.1 Detailed Description

Interface of a block that just puts its input in its output.

#### Author

Carlos Aviles, 2010. carlos.avilesr(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

## 11.263 pcps\_acquisition.h File Reference

This class implements a Parallel Code Phase Search Acquisition.

```
#include "acq_conf.h"
#include "channel_fsm.h"
#include <armadillo>
#include <glog/logging.h>
#include <gnuradio/block.h>
#include <gnuradio/fft/fft.h>
#include <gnuradio/gr_complex.h>
#include <gnuradio/thread/thread.h>
#include <gnuradio/types.h>
#include <volk/volk_complex.h>
#include <volk_gnssssdr/volk_gnssssdr_alloc.h>
#include <complex>
#include <cstdint>
#include <memory>
#include <queue>
#include <string>
#include <utility>
#include <gsl/gsl>
```

### 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\_)

#### 11.263.1 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](mailto:jarribas(at)cttc.es)
- Luis Esteve, 2012. [luis\(at\)epsilon-formacion.com](mailto:luis(at)epsilon-formacion.com)
- Marc Molina, 2013. [marc.molina.pena@gmail.com](mailto:marc.molina.pena@gmail.com)
- Cillian O'Driscoll, 2017. [cillian\(at\)ieee.org](mailto:cillian(at)ieee.org)
- Antonio Ramos, 2017. [antonio.ramos@cttc.es](mailto: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

## 11.264 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 "acq_conf.h"
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include <armadillo>
#include <gnuradio/block.h>
#include <gnuradio/fft/fft.h>
#include <gnuradio/gr_complex.h>
#include <volk_gnssssdr/volk_gnssssdr_alloc.h>
#include <cstdlib>
#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\_)

### 11.264.1 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

## 11.265 pcps\_acquisition\_fpga.h File Reference

This class implements a Parallel Code Phase Search Acquisition for the FPGA.

```
#include "channel_fsm.h"
#include "fpga_acquisition.h"
#include <glog/logging.h>
#include <stdint>
#include <memory>
#include <string>
```

#### Classes

- struct [pcpsconf\\_fpga\\_t](#)
- 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** (`pcpsconf_fpga_t` conf\_)

### 11.265.1 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) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

## 11.266 pcps\_assisted\_acquisition\_cc.h File Reference

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

```
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include <gnuradio/block.h>
#include <gnuradio/fft/fft.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** (`int32_t` max\_dwells, `uint32_t` sampled\_ms, `int32_t` doppler\_max, `int32_t` doppler\_min, `int64_t` fs\_in, `int32_t` samples\_per\_ms, `bool` dump, `const std::string` &dump\_filename, `bool` enable\_monitor\_output)

### 11.266.1 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

## 11.267 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 "channel_fsm.h"
#include "gnss_synchro.h"
#include <gnuradio/block.h>
#include <gnuradio/fft/fft.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 (uint32_t sampled_ms, uint32_t max_dwells, uint32_t doppler_max, int64_t fs_in, int32_t samples_per_ms, int32_t samples_per_code, bool dump, const std::string &dump_filename, bool enable_monitor_output)`

### 11.267.1 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

### 11.268 pcps\_openccl\_acquisition\_cc.h File Reference

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

```
#include "channel_fsm.h"
#include "gnss_block_interface.h"
#include "gnss_synchro.h"
#include "openccl/fft_internal.h"
#include <gnuradio/block.h>
#include <gnuradio/fft/fft.h>
#include <gnuradio/gr_complex.h>
#include "openccl/cl.hpp"
#include <cstdint>
#include <fstream>
#include <memory>
#include <string>
#include <vector>
```

## Classes

- class [pcps\\_openccl\\_acquisition\\_cc](#)

*This class implements a Parallel Code Phase Search Acquisition.*

## Typedefs

- using **pcps\_openc1\_acquisition\_cc\_sptr** = gnss\_shared\_ptr< [pcps\\_openc1\\_acquisition\\_cc](#) >

## Functions

- pcps\_openc1\_acquisition\_cc\_sptr **pcps\_make\_openc1\_acquisition\_cc** (uint32\_t sampled\_ms, uint32\_t max\_dwells, uint32\_t doppler\_max, int64\_t fs\_in, int samples\_per\_ms, int samples\_per\_code, bool bit\_transition\_flag, bool dump, const std::string &dump\_filename, bool enable\_monitor\_output)

### 11.268.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition using OpenGL 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](mailto:jarribas(at)cttc.es)
- Luis Esteve, 2012. [luis\(at\)epsilon-formacion.com](mailto:luis(at)epsilon-formacion.com)
- Marc Molina, 2013. [marc.molina.pena@gmail.com](mailto: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

## 11.269 pcps\_quicksync\_acquisition\_cc.h File Reference

This class implements a Parallel Code Phase Search Acquisition with the QuickSync Algorithm.

```
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include <gnuradio/block.h>
#include <gnuradio/fft/fft.h>
#include <gnuradio/gr_complex.h>
#include <algorithm>
#include <cassert>
#include <fstream>
#include <functional>
#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 (uint32_t folding_factor, uint32_t sampled_ms, uint32_t max_dwells, uint32_t doppler_max, int64_t fs_in, int32_t samples_per_ms, int32_t samples_per_code, bool bit_transition_flag, bool dump, const std::string &dump_filename, bool enable_monitor_output)`

### 11.269.1 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](mailto: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

## 11.270 pcps\_tong\_acquisition\_cc.h File Reference

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

```
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include <gnuradio/block.h>
#include <gnuradio/fft/fft.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** (uint32\_t sampled\_ms, uint32\_t doppler\_max, int64\_t fs\_in, int32\_t samples\_per\_ms, int32\_t samples\_per\_code, uint32\_t tong\_init\_val, uint32\_t tong\_max\_val, uint32\_t tong\_max\_dwells, bool dump, const std::string &dump\_filename, bool enable\_monitor\_output)

#### 11.270.1 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 equal 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

## 11.271 plutosdr\_signal\_source.h File Reference

Signal source for PlutoSDR.

```
#include "gnss_block_interface.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](#)

### 11.271.1 Detailed Description

Signal source for PlutoSDR.

#### Author

Rodrigo Muñoz, 2017, [rmunozl@inacap.cl](mailto:rmunozl@inacap.cl), [rodrigo.munoz@proteinlab.cl](mailto:rodrigo.munoz@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

## 11.272 position\_test\_flags.h File Reference

Helper file for unit testing.

```
#include <gflags/gflags.h>
#include <limits>
#include <string>
```

### Functions

- **DEFINE\_string** (config\_file\_ptest, std::string(""), "File containing the configuration parameters for the position test.")
- **DEFINE\_bool** (plot\_position\_test, false, "Plots results of with gnuplot")
- **DEFINE\_bool** (static\_scenario, true, "Compute figures of merit for static user position (DRMS, CEP, etc..)")
- **DEFINE\_bool** (use\_ref\_motion\_file, false, "Enable or disable the use of a reference file containing the true receiver position, velocity and acceleration.")
- **DEFINE\_int32** (ref\_motion\_file\_type, 1, "Type of reference motion file: 1- Spirent CSV motion file")
- **DEFINE\_string** (ref\_motion\_filename, std::string("motion.csv"), "Path and filename for the reference motion file")
- **DEFINE\_string** (pvt\_solver\_dump\_filename, std::string("PVT.dat"), "Path and filename for the PVT solver binary dump file")
- **DEFINE\_double** (static\_2D\_error\_m, 2.0, "Static scenario 2D (East, North) positioning error threshold [meters]")
- **DEFINE\_double** (static\_3D\_error\_m, 5.0, "Static scenario 3D (East, North, Up) positioning error threshold [meters]")
- **DEFINE\_double** (accuracy CEP, 2.0, "Static scenario 2D (East, North) accuracy Circular Error Position (CEP) threshold [meters]")
- **DEFINE\_double** (precision\_SEP, 10.0, "Static scenario 3D (East, North, Up) precision Spherical Error Position (SEP) threshold [meters]")
- **DEFINE\_double** (dynamic\_3D\_position\_RMSE, 10.0, "Dynamic scenario 3D (ECEF) accuracy RMS↵E threshold [meters]")
- **DEFINE\_double** (dynamic\_3D\_velocity\_RMSE, 5.0, "Dynamic scenario 3D (ECEF) velocity accuracy RM↵SE threshold [meters/second]")
- **DEFINE\_bool** (enable\_carrier\_smoothing, false, "Activates carrier smoothing of pseudoranges")

### 11.272.1 Detailed Description

Helper file for unit testing.

#### 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

## 11.273 pulse\_blanking\_cc.h File Reference

Implements a pulse blanking algorithm.

```
#include "gnss_block_interface.h"
#include <gnuradio/block.h>
#include <volk_gnssssdr/volk_gnssssdr_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)

### 11.273.1 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

## 11.274 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](#)

### 11.274.1 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

## 11.275 pvt\_conf.h File Reference

Class that contains all the configuration parameters for the PVT block.

```
#include <cstdint>
#include <map>
#include <string>
```

## Classes

- class [Pvt\\_Conf](#)

### 11.275.1 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

## 11.276 pvt\_interface.h File Reference

This class represents an interface to a PVT block.

```
#include "galileo_almanac.h"
#include "galileo_ephemeris.h"
#include "gnss_block_interface.h"
#include "gps_almanac.h"
#include "gps_ephemeris.h"
#include <map>
```

## Classes

- class [PvtInterface](#)

*This class represents an interface to a PVT block.*

### 11.276.1 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

## 11.277 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 <deque>
```

## Classes

- class [Pvt\\_Solution](#)

*Base class for a PVT solution.*

### 11.277.1 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

## 11.278 raw\_array\_signal\_source.h File Reference

CTTC Experimental GNSS 8 channels array signal source.

```
#include "concurrent_queue.h"
#include "gnss_block_interface.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 a GN3S USB dongle, a RF front-end signal sampler.*

### 11.278.1 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

## 11.279 rinex\_printer.h File Reference

Interface of a RINEX 2.11 / 3.01 printer See <ftp://igs.org/pub/data/format/rinex301.pdf>.

```
#include <boost/date_time/posix_time/posix_time.hpp>
#include <cstdint>
#include <cstdlib>
#include <fstream>
#include <iomanip>
#include <map>
#include <sstream>
#include <string>
#include <vector>
```

### Classes

- class [Rinex\\_Printer](#)

*Class that handles the generation of Receiver INdependent EXchange format (RINEX) files.*

## Functions

- `std::string asString` (long double x, `std::string::size_type` precision)
- `int64_t asInt` (const `std::string` &s)

### 11.279.1 Detailed Description

Interface of a RINEX 2.11 / 3.01 printer See <ftp://igs.org/pub/data/format/rinex301.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://kb.igs.org/hc/en-us/articles/201096516-IGS-Formats>

#### Author

Carles Fernandez Prades, 2011. [cfernandez\(at\)cttc.es](mailto: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

### 11.280 rtcm.h File Reference

Interface for the RTCM 3.2 Standard.

```
#include "concurrent_queue.h"
#include "galileo_ephemeris.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 <glog/logging.h>
#include <algorithm>
#include <array>
#include <bitset>
#include <cstdint>
#include <cstring>
#include <deque>
#include <list>
#include <map>
#include <memory>
#include <set>
#include <string>
#include <thread>
#include <utility>
#include <vector>
```

## 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

### 11.280.1 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

## 11.281 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 <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.*

### 11.281.1 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

## 11.282 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 [sbsfcrr\\_t](#)
- struct [sbsslcorr\\_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) pthread\_mutex\_init(f, NULL)
- #define **rtk\_lock**(f) pthread\_mutex\_lock(f)
- #define **rtk\_unlock**(f) pthread\_mutex\_unlock(f)
- #define **VER\_RTKLIB** "2.4.2"
- #define **NTRIP\_AGENT** "RTKLIB/" VER\_RTKLIB
- #define **NTRIP\_CLI\_PORT** 2101 /\* default ntrip-client connection port \*/
- #define **NTRIP\_SVR\_PORT** 80 /\* default ntrip-server connection port \*/
- #define **NTRIP\_MAXRSP** 32768 /\* max size of ntrip response \*/
- #define **NTRIP\_MAXSTR** 256 /\* max length of mountpoint string \*/
- #define **NTRIP\_RSP\_OK\_CLI** "ICY 200 OK\r\n" /\* ntrip response: client \*/
- #define **NTRIP\_RSP\_OK\_SVR** "OK\r\n" /\* ntrip response: server \*/
- #define **NTRIP\_RSP\_SRCTBL** "SOURCETABLE 200 OK\r\n" /\* ntrip response: source table \*/
- #define **NTRIP\_RSP\_TBLEND** "ENDSOURCETABLE"
- #define **NTRIP\_RSP\_HTTP** "HTTP/" /\* ntrip response: http \*/
- #define **NTRIP\_RSP\_ERROR** "ERROR" /\* ntrip response: error \*/
- #define **FTP\_CMD** "wget" /\* ftp/http command \*/
- #define **ENAGLO**
- #define **ENABDS**
- #define **STR\_MODE\_R** 0x1 /\* stream mode: read \*/

- `#define STR_MODE_W 0x2 /* stream mode: write */`
- `#define STR_MODE_RW 0x3 /* stream mode: read/write */`
- `#define STR_NONE 0 /* stream type: none */`
- `#define STR_SERIAL 1 /* stream type: serial */`
- `#define STR_FILE 2 /* stream type: file */`
- `#define STR_TCPSVR 3 /* stream type: TCP server */`
- `#define STR_TCPCLI 4 /* stream type: TCP client */`
- `#define STR_UDP 5 /* stream type: UDP stream */`
- `#define STR_NTRIPSVR 6 /* stream type: NTRIP server */`
- `#define STR_NTRIPCLI 7 /* stream type: NTRIP client */`
- `#define STR_FTP 8 /* stream type: ftp */`
- `#define STR_HTTP 9 /* stream type: http */`
- `#define NP_PPP(opt) ((opt)->dynamics ? 9 : 3) /* number of pos solution */`
- `#define IC_PPP(s, opt) (NP_PPP(opt) + (s)) /* state index of clocks (s=0:gps,1:glo) */`
- `#define IT_PPP(opt) (IC_PPP(0, opt) + NSYS) /* state index of tropos */`
- `#define NR_PPP(opt) (IT_PPP(opt) + ((opt)->tropopt < TROPOPT_EST ? 0 : ((opt)->tropopt == TROPOPT_EST ? 1 : 3))) /* number of solutions */`
- `#define IB_PPP(s, opt) (NR_PPP(opt) + (s)-1) /* state index of phase bias */`
- `#define NX_PPP(opt) (IB_PPP(MAXSAT, opt) + 1) /* number of estimated states */`
- `#define NF_RTK(opt) ((opt)->ionoopt == IONOOPT_IFLC ? 1 : (opt)->nf)`
- `#define NP_RTK(opt) ((opt)->dynamics == 0 ? 3 : 9)`
- `#define NI_RTK(opt) ((opt)->ionoopt != IONOOPT_EST ? 0 : MAXSAT)`
- `#define NT_RTK(opt) ((opt)->tropopt < TROPOPT_EST ? 0 : ((opt)->tropopt < TROPOPT_ESTG ? 2 : 6))`
- `#define NL_RTK(opt) ((opt)->glomodear != 2 ? 0 : NFREQGLO)`
- `#define NB_RTK(opt) ((opt)->mode <= PMODE_DGPS ? 0 : MAXSAT * NF_RTK(opt))`
- `#define NR_RTK(opt) (NP_RTK(opt) + NI_RTK(opt) + NT_RTK(opt) + NL_RTK(opt))`
- `#define NX_RTK(opt) (NR_RTK(opt) + NB_RTK(opt))`

## Typedefs

- using `fatalfunc_t` = `void(const char *)`  
*fatal callback function type*
- typedef struct `half_cyc_tag` `half_cyc_t`

## 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-kinematic*
- 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 **MINPRNQZS** = 0
- const int **MAXPRNQZS** = 0
- const int **MINPRNQZS\_S** = 0
- const int **MAXPRNQZS\_S** = 0
- const int **NSATQZS** = 0
- const int **NSYSQZS** = 0
- const int **MINPRNBDS** = 1  
*min satellite sat number of BeiDou*
- const int **MAXPRNBDS** = 37  
*max satellite sat number of BeiDou*
- const int **NSATBDS** = (**MAXPRNBDS** - **MINPRNBDS** + 1)  
*number of BeiDou satellites*

- const int **NSYSBDS** = 1
- const int **MINPRNIRN** = 0
- const int **MAXPRNIRN** = 0
- const int **NSATIRN** = 0
- const int **NSYSIRN** = 0
- const int **MINPRNLEO** = 0
- const int **MAXPRNLEO** = 0
- 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 **MAXOBS****TYPE** = 64  
*max number of obs type in RINEX*
- const double **MAXD****TOE** = 7200.0  
*max time difference to GPS Toe (s)*
- const double **MAXD****TOE****\_QZS** = 7200.0  
*max time difference to QZSS Toe (s)*
- const double **MAXD****TOE****\_GAL** = 10800.0  
*max time difference to Galileo Toe (s)*
- const double **MAXD****TOE****\_BDS** = 21600.0  
*max time difference to BeiDou Toe (s)*
- const double **MAXD****TOE****\_GLO** = 1800.0  
*max time difference to GLONASS Toe (s)*
- const double **MAXD****TOE****\_SBS** = 360.0  
*max time difference to SBAS Toe (s)*
- const double **MAXD****TOE****\_S** = 86400.0  
*max time difference to ephemeris toe (s) for other*
- const double **MAXG****DOP** = 300.0  
*max GDOP*
- const int **MAXS****BS****URA** = 8  
*max URA of SBAS satellite*
- const int **MAXB****BAND** = 10  
*max SBAS band of IGP*
- const int **MAXN****IGP** = 201  
*max number of IGP in SBAS band*
- const int **MAXN****GEO** = 4  
*max number of GEO satellites*
- const int **MAXS****OLMSG** = 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 ionosphere*
- 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*
- const double [EFACT\\_GPS](#) = 1.0
- error factor: GPS*
- const double [EFACT\\_GLO](#) = 1.5
- error factor: GLONASS*

- const double **EFACT\_GAL** = 1.0  
*error factor: Galileo*
- const double **EFACT\_QZS** = 1.0  
*error factor: QZSS*
- const double **EFACT\_BDS** = 1.0  
*error factor: BeiDou*
- const double **EFACT\_IRN** = 1.5  
*error factor: IRNSS*
- const 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

### 11.282.1 Detailed Description

main header file for the rtklib library

#### 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

## 11.283 rtklib\_conversions.h File Reference

GNSS-SDR to RTKLIB data structures conversion functions.

```
#include "rtklib.h"
```

#### Functions

- **eph\_t eph\_to\_rtklib** (const [Galileo\\_Ephemeris](#) &gal\_eph)
- **eph\_t eph\_to\_rtklib** (const [Gps\\_Ephemeris](#) &gps\_eph, bool pre\_2009\_file)
- **eph\_t eph\_to\_rtklib** (const [Gps\\_CNAV\\_Ephemeris](#) &gps\_cnav\_eph)
- **eph\_t eph\_to\_rtklib** (const [Beidou\\_Dnav\\_Ephemeris](#) &bei\_eph)
- **alm\_t alm\_to\_rtklib** (const [Gps\\_Almanac](#) &gps\_alm)
- **alm\_t alm\_to\_rtklib** (const [Galileo\\_Almanac](#) &gal\_alm)
- **geph\_t eph\_to\_rtklib** (const [Glonass\\_Gnav\\_Ephemeris](#) &glonass\_gnav\_eph, const [Glonass\\_Gnav\\_Utc\\_Model](#) &gnav\_clock\_model)

*Transforms a [Glonass\\_Gnav\\_Ephemeris](#) to its RTKLIB counterpart.*

- **obsd\_t insert\_obs\_to\_rtklib** ([obsd\\_t](#) &rtklib\_obs, const [Gnss\\_Synchro](#) &gnss\_synchro, int week, int band, bool pre\_2009\_file=false)

### 11.283.1 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

## 11.284 rtklib\_ephemeris.h File Reference

satellite ephemeris and clock functions

```
#include "rtklib.h"
```

### Functions

- double **var\_uraeph** (int ura)
- 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)
- 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 **ephpos** (gtime\_t time, gtime\_t teph, int sat, const nav\_t \*nav, int iode, double \*rs, double \*dts, double \*var, int \*svh)
- int **satpos\_sbass** (gtime\_t time, gtime\_t teph, int sat, const nav\_t \*nav, double \*rs, double \*dts, double \*var, int \*svh)
- int **satpos\_ssrr** (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)
- 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)

### 11.284.1 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

## 11.285 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 **combttec** ([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

### 11.285.1 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

### 11.286 rtklib\_lambda.h File Reference

Integer ambiguity resolution.

```
#include "rtklib.h"
```

#### Macros

- #define **SGN\_LAMBDA**(x) ((x) <= 0.0 ? -1.0 : 1.0)
- #define **ROUND\_LAMBDA**(x) (floor((x) + 0.5))
- #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

### 11.286.1 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

### 11.286.2 Macro Definition Documentation

## 11.286.2.1 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.

## 11.287 rtklib\_pntpos.h File Reference

standard code-based positioning

```
#include "rtklib.h"
#include "rtklib_rtkcmn.h"
```

## Functions

- double **varerr** (const [prcopt\\_t](#) \*opt, double el, int sys)
- double **gettgd** (int sat, const [nav\\_t](#) \*nav)
- double **getisc1** (int sat, const [nav\\_t](#) \*nav)
- double **getisc2** (int sat, const [nav\\_t](#) \*nav)
- double **getisc5i** (int sat, const [nav\\_t](#) \*nav)
- double **getisc5q** (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)
- int **ionocorr** ([gtime\\_t](#) time, const [nav\\_t](#) \*nav, int sat, const double \*pos, const double \*azel, int ionopt, double \*ion, double \*var)
- 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 + 3

### 11.287.1 of estimated parameters

- const int `MAXITR` = 10  
*max number of iteration for point pos*
- const double `ERR_ION` = 5.0  
*ionospheric delay std (m)*
- const double `ERR_TROP` = 3.0  
*tropospheric delay std (m)*

### 11.287.2 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

### 11.287.3 Function Documentation

### 11.287.3.1 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)

## 11.287.4 Variable Documentation

### 11.287.4.1 ERR\_ION

```
const double ERR_ION = 5.0
```

ionospheric delay std (m)

Definition at line 42 of file `rtklib_pntpos.h`.

### 11.287.4.2 ERR\_TROP

```
const double ERR_TROP = 3.0
```

tropspheric delay std (m)

Definition at line 43 of file `rtklib_pntpos.h`.

### 11.287.4.3 MAXITR

```
const int MAXITR = 10
```

max number of iteration for point pos

Definition at line 41 of file `rtklib_pntpos.h`.

## 11.287.4.4 NX

```
const int NX = 4 + 3
```

## 11.287.5 of estimated parameters

Definition at line 40 of file rtklib\_pntpos.h.

## 11.288 rtklib\_ppp.h File Reference

Precise Point Positioning.

```
#include "rtklib.h"
```

## Macros

- #define **MIN\_PPP**(x, y) ((x) <= (y) ? (x) : (y))
- #define **ROUND\_PPP**(x) static\_cast<int>(floor((x) + 0.5))
- #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)
- int **corr\_ion** (itime\_t time, const nav\_t \*nav, int sat, const double \*pos, const double \*azel, int ionopt, 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)

### 11.288.1 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

## 11.288.2 Macro Definition Documentation

### 11.288.2.1 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.

### 11.288.2.2 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.

## 11.289 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 **interppl** (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

### 11.289.1 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

## 11.290 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 <cstdio>
#include <ctime>
#include <map>
#include <string>
```

### Classes

- class [Rtklib\\_Pvt](#)

*This class implements a [PvtInterface](#) for the RTKLIB PVT block.*

### 11.290.1 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

## 11.291 rtklib\_pvt\_gs.h File Reference

Interface of a Position Velocity and Time computation block.

```
#include "gnss_block_interface.h"
#include "gnss_synchro.h"
#include "rtklib.h"
#include <boost/date_time/gregorian/gregorian.hpp>
#include <boost/date_time/posix_time/posix_time.hpp>
#include <gnuradio/sync_block.h>
#include <gnuradio/types.h>
#include <pmt/pmt.h>
#include <chrono>
#include <cstdio>
#include <stdint>
#include <ctime>
#include <map>
#include <memory>
#include <string>
#include <sys/types.h>
#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)

### 11.291.1 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

### 11.292 rtklib\_rtcn.h File Reference

RTCM functions headers.

```
#include "rtklib.h"
#include "rtklib_rtcn2.h"
#include "rtklib_rtcn3.h"
```

## Macros

- `#define RTCM2PREAMB 0x66 /* rtcn ver.2 frame preamble */`
- `#define RTCM3PREAMB 0xD3 /* rtcn ver.3 frame preamble */`

## Functions

- int [init\\_rtcn](#) ([rtcn\\_t](#) \*rtcn)
- void [free\\_rtcn](#) ([rtcn\\_t](#) \*rtcn)
- int [input\\_rtcn2](#) ([rtcn\\_t](#) \*rtcn, unsigned char data)
- int [input\\_rtcn3](#) ([rtcn\\_t](#) \*rtcn, unsigned char data)
- int [input\\_rtcn2f](#) ([rtcn\\_t](#) \*rtcn, FILE \*fp)
- int [input\\_rtcn3f](#) ([rtcn\\_t](#) \*rtcn, FILE \*fp)
- int [gen\\_rtcn2](#) ([rtcn\\_t](#) \*rtcn, int type, int sync)

### 11.292.1 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/tomokitakasu/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

## 11.293 rtklib\_rtc2.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, bool pre\_2009\_file=false)
- int **decode\_type16** ([rtcm\\_t](#) \*rtcm)
- int **decode\_type17** ([rtcm\\_t](#) \*rtcm, bool pre\_2009\_file=false)
- 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\_rtc2** ([rtcm\\_t](#) \*rtcm)

### 11.293.1 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

### 11.294 rtklib\_rtc3.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, bool pre\_2009\_file=false)
- 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, bool pre\_2009\_file=false)
- 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\_rtc3** ([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** []

### 11.294.1 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

### 11.294.2 Variable Documentation

#### 11.294.2.1 CODES\_BDS

```
const int CODES_BDS[ ]
```

##### Initial value:

```
= {
    CODE_L1I, CODE_L1Q, CODE_L1X, CODE_L7I,
    CODE_L7Q, CODE_L7X, CODE_L6I, CODE_L6Q,
    CODE_L6X}
```

Definition at line 73 of file rtklib\_rtc3.h.

### 11.294.2.2 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\_rtc3.h.

### 11.294.2.3 CODES\_GLO

```
const int CODES_GLO[]
```

**Initial value:**

```
= {  
    CODE_L1C, CODE_L1P, CODE_L2C, CODE_L2P}
```

Definition at line 58 of file rtklib\_rtc3.h.

### 11.294.2.4 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\_rtc3.h.

### 11.294.2.5 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\_rtc3.h.

### 11.294.2.6 CODES\_SBS

```
const int CODES_SBS[ ]
```

**Initial value:**

```
= {  
    CODE_L1C, CODE_L5I, CODE_L5Q, CODE_L5X}
```

Definition at line 78 of file rtklib\_rtc3.h.

### 11.294.2.7 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\_rtc3.h.

## 11.295 rtklib\_rtkcmn.h File Reference

rtklib common functions

```
#include "rtklib.h"  
#include <string>
```

## Macros

- `#define Rx(t, X)`
- `#define Ry(t, X)`
- `#define Rz(t, X)`

## Functions

- void **fatalerr** (const char \*format,...)
- int **satno** (int sys, int prn)
- int **satsys** (int sat, int \*prn)
- int **satid2no** (const char \*id)
- void **satno2id** (int sat, char \*id)
- 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)
- 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](#) **bd2time** (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, bool pre\_2009\_file=false)
- unsigned int **ticketget** ()
- 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 **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 **unignav** ([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** (const char \*path)
- int **repstr** (char \*str, const char \*pat, const char \*rep)
- int **reppath** (const char \*path, char \*rpath, [gtime\\_t](#) time, const char \*rov, const char \*base)
- int **reppaths** (const char \*path, char \*rpath[], int nmax, [gtime\\_t](#) ts, [gtime\\_t](#) te, 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 **interp** (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)

### 11.295.1 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-200K, Navstar GPS Space Segment/Navigation User Interfaces, 7 March, 2006 [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

## 11.295.2 Macro Definition Documentation

### 11.295.2.1 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 66 of file rtklib\_rtkcmn.h.

## 11.295.2.2 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 77 of file rtklib\_rtkcmn.h.

## 11.295.2.3 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 88 of file rtklib\_rtkcmn.h.

## 11.296 rtklib\_rtkpos.h File Reference

## rtklib ppp-related functions

```
#include "rtklib.h"
#include "rtklib_rtkcmn.h"
```

**Macros**

- **#define IL\_RTK**(s, opt) (NP\_RTK(opt) + (s)-1) /\* ionos (s:satellite no) \*/
- **#define IT\_RTK**(r, opt) (NP\_RTK(opt) + NI\_RTK(opt) + NT\_RTK(opt) / 2 \* (r)) /\* tropos (r:0=rov,1:ref) \*/
- **#define IB\_RTK**(f, opt) (NP\_RTK(opt) + NI\_RTK(opt) + NT\_RTK(opt) + (f)) /\* receiver h/w bias \*/
- **#define IB\_RTK**(s, f, opt) (NR\_RTK(opt) + MAXSAT \* (f) + (s)-1) /\* phase bias (s:satno,f:freq) \*/

## 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 **globs\_L1L2** (const [obsd\\_t](#) \*obs, int i, int j, const double \*lam)
- double **globs\_L1L5** (const [obsd\\_t](#) \*obs, int i, int j, const double \*lam)
- double **varerr** (int sat, int sys, double el, double bl, double dt, int f, const [prcopt\\_t](#) \*opt)
- 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, int i, 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 [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 **ddmat** ([rtk\\_t](#) \*rtk, double \*D)
- 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 **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](#))

### 11.296.1 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.

## 11.297 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 bufsize, 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**

## 11.297.1 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



## 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]

### 11.298.1 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

## 11.298.2 Variable Documentation

### 11.298.2.1 IGPBAND1

```
const sbsigband_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\_sbass.h.

### 11.298.2.2 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.

## 11.299 rtklib\_solution.h File Reference

solution functions headers

```
#include "rtklib.h"
```

### Macros

- **#define COMMENTH** "%" /\* comment line indicator for solution \*/
- **#define MSG\_DISCONNECT** "\$\_DISCONNECT\r\n" /\* disconnect message \*/

### 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 **outsollexs** (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 **outsollex** (FILE \*fp, const [sol\\_t](#) \*sol, const [ssat\\_t](#) \*ssat, const [solopt\\_t](#) \*opt)

### 11.299.1 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.

## 11.300 rtklib\_solver.h File Reference

PVT solver based on rtklib library functions adapted to the GNSS-SDR data flow and structures.

```
#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_iono.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_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_solution.h"
#include "rtklib.h"
#include <array>
#include <fstream>
#include <map>
#include <string>
```

### Classes

- class [Rtklib\\_Solver](#)

*This class implements a PVT solution based on RTKLIB.*

### 11.300.1 Detailed Description

PVT solver based on rtklib library functions adapted to the GNSS-SDR data flow and structures.

#### Authors

- 2017, Javier Arribas
- 2017, 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) 2017-2019, Carles Fernandez All rights reserved.

SPDX-License-Identifier: BSD-2-Clause

## 11.301 rtklib\_solver\_dump\_reader.h File Reference

Helper file for unit testing.

```
#include <cstdint>
#include <fstream>
#include <string>
#include <vector>
```

### Classes

- class [Rtklib\\_Solver\\_Dump\\_Reader](#)

### 11.301.1 Detailed Description

Helper file for unit testing.

#### 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

## 11.302 rtklib\_stream.h File Reference

streaming functions

```
#include "rtklib.h"
```

### Macros

- **#define TINTACT** 200 /\* period for stream active (ms) \*/
- **#define SERIBUFFSIZE** 4096 /\* serial buffer size (bytes) \*/
- **#define TIMETAGH\_LEN** 64 /\* time tag file header length \*/
- **#define MAXCLI** 32 /\* max client connection for tcp svr \*/
- **#define MAXSTATMSG** 32 /\* max length of status message \*/
- **#define VER\_RTKLIB** "2.4.2"
- **#define NTRIP\_AGENT** "RTKLIB/" VER\_RTKLIB
- **#define NTRIP\_CLI\_PORT** 2101 /\* default ntrip-client connection port \*/
- **#define NTRIP\_SVR\_PORT** 80 /\* default ntrip-server connection port \*/
- **#define NTRIP\_MAXRSP** 32768 /\* max size of ntrip response \*/
- **#define NTRIP\_MAXSTR** 256 /\* max length of mountpoint string \*/
- **#define NTRIP\_RSP\_OK\_CLI** "ICY 200 OK\r\n" /\* ntrip response: client \*/
- **#define NTRIP\_RSP\_OK\_SVR** "OK\r\n" /\* ntrip response: server \*/
- **#define NTRIP\_RSP\_SRCTBL** "SOURCETABLE 200 OK\r\n" /\* ntrip response: source table \*/
- **#define NTRIP\_RSP\_TBLEND** "ENDSOURCETABLE"
- **#define NTRIP\_RSP\_HTTP** "HTTP/" /\* ntrip response: http \*/
- **#define NTRIP\_RSP\_ERROR** "ERROR" /\* ntrip response: error \*/
- **#define FTP\_CMD** "wget" /\* ftp/http command \*/
- **#define FTP\_TIMEOUT** 30 /\* ftp/http timeout (s) \*/

## 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 char \*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 **decodetcp**path (const char \*path, char \*addr, char \*port, char \*user, char \*passwd, char \*mntpt, 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 **waitntrip** ([ntrip\\_t](#) \*ntrip, char \*msg)
- [ntrip\\_t](#) \* **openntrip** (const char \*path, int type, char \*msg)
- void **closetrip** ([ntrip\\_t](#) \*ntrip)
- int **readntrip** ([ntrip\\_t](#) \*ntrip, unsigned char \*buff, int n, char \*msg)
- int **writenrip** ([ntrip\\_t](#) \*ntrip, unsigned char \*buff, int n, char \*msg)
- int **statenrip** ([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 **stread** (**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)

### 11.302.1 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

## 11.303 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

### 11.303.1 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

## 11.304 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↵  
::socket &socket)  
*Send a command to rtl\_tcp over the given socket.*

#### 11.304.1 Detailed Description

Defines structures and constants for communicating with rtl\_tcp.

#### Author

Anthony Arnold, 2015. [anthony.arnold@uqconnect.edu.au](mailto:anthony.arnold@uqconnect.edu.au)

This file contains information taken from librtlsdr:

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

## 11.305 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.*

### 11.305.1 Detailed Description

Interface for a structure sent by rtl\_tcp defining the hardware.

#### Author

Anthony Arnold, 2015. [anthony.arnold\(at\)uqconnect.edu.au](mailto:anthony.arnold@uqconnect.edu.au)

This file contains information taken from librtlsdr:

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

## 11.306 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 "gnss_block_interface.h"
#include "rtl_tcp_signal_source_c.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>)*

### 11.306.1 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](mailto:anthony.arnold@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

## 11.307 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 <stdint>
#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)

#### 11.307.1 Detailed Description

Interface of an rtl\_tcp signal source reader.

#### Author

Anthony Arnold, 2015. [anthony.arnold@uqconnect.edu.au](mailto:anthony.arnold@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

## 11.308 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.*

### 11.308.1 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

## 11.309 sbas\_l1\_telemetry\_decoder.h File Reference

Interface of an adapter of a SBAS telemetry data decoder block to a [TelemetryDecoderInterface](#).

```
#include "gnss_satellite.h"  
#include "gnss_synchro.h"  
#include "sbas_l1_telemetry_decoder_gs.h"  
#include "telemetry_decoder_interface.h"  
#include <gnuradio/runtime_types.h>  
#include <stddef>  
#include <string>
```

### Classes

- class [SbasL1TelemetryDecoder](#)  
*This class implements a NAV data decoder for SBAS frames in L1 radio link.*

### 11.309.1 Detailed Description

Interface of an adapter of a SBAS telemetry data decoder block to a [TelemetryDecoderInterface](#).

#### 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

## 11.310 sbas\_l1\_telemetry\_decoder\_gs.h File Reference

Interface of a SBAS telemetry data decoder block.

```
#include "gnss_block_interface.h"
#include "gnss_satellite.h"
#include <boost/crc.hpp>
#include <gnuradio/block.h>
#include <gnuradio/types.h>
#include <cstdint>
#include <cstdint>
#include <deque>
#include <fstream>
#include <memory>
#include <string>
#include <utility>
#include <vector>
```

### Classes

- 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** (const [Gnss\\_Satellite](#) &satellite, bool dump)

### 11.310.1 Detailed Description

Interface of a SBAS telemetry data decoder block.

#### 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

## 11.311 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 <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.*

### 11.311.1 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

## 11.312 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 <memory>
#include <string>
#include <utility>
```

## Classes

- class [Serdes\\_Monitor\\_Pvt](#)

*This class implements serialization and deserialization of [Monitor\\_Pvt](#) objects using Protocol Buffers.*

### 11.312.1 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

## 11.313 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 ()`

### 11.313.1 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

## 11.314 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 <cstdint>
#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.*

### 11.314.1 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

## 11.315 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.*

### 11.315.1 Detailed Description

Adapter of a class that generates synthesized GNSS signal.

#### Author

Marc Molina, 2013. [marc.molina.pena@gmail.com](mailto: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

## 11.316 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 > &C/N0_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.*

### 11.316.1 Detailed Description

GNU Radio source block that generates synthesized GNSS signal.

#### Author

Marc Molina, 2013. [marc.molina.pena@gmail.com](mailto: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

## 11.316.2 Function Documentation

### 11.316.2.1 signal\_make\_generator\_c()

```
signal_generator_c_sptr signal_make_generator_c (
    const std::vector< std::string > & signall,
    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.

## 11.317 signal\_generator\_flags.h File Reference

Helper file for unit testing.

```
#include <gflags/gflags.h>
#include <limits>
#include <string>
```

### Functions

- **DEFINE\_bool** (disable\_generator, false, "Disable the signal generator (a external signal file must be available for the test)")
- **DEFINE\_string** (generator\_binary, std::string(SW\_GENERATOR\_BIN), "Path of software-defined signal generator binary")
- **DEFINE\_string** (rinex\_nav\_file, std::string(DEFAULT\_RINEX\_NAV), "Input RINEX navigation file")
- **DEFINE\_int32** (duration, 100, "Duration of the experiment [in seconds, max = 300]")
- **DEFINE\_string** (static\_position, "30.286502,120.032669,100", "Static receiver position [latitude,longitude,height]")
- **DEFINE\_string** (dynamic\_position, "", "Observer positions file, in .csv or .nmea format")
- **DEFINE\_string** (filename\_rinex\_obs, "sim.16o", "Filename of output RINEX navigation file")
- **DEFINE\_string** (filename\_raw\_data, "signal\_out.bin", "Filename of output raw data file")
- **DEFINE\_int32** (fs\_gen\_sps, 2600000, "Sampling frequency [sps]")
- **DEFINE\_int32** (test\_satellite\_PRN, 1, "PRN of the satellite under test (must be visible during the observation time)")
- **DEFINE\_int32** (test\_satellite\_PRN2, 2, "PRN of the satellite under test (must be visible during the observation time)")
- **DEFINE\_string** (test\_satellite\_PRN\_list, "1,2,3,6,9,10,12,17,20,23,28", "List of PRN of the satellites under test (must be visible during the observation time)")
- **DEFINE\_double** (CN0\_dBHz, std::numeric\_limits< double >::infinity(), "Enable noise generator and set the CN0 [dB-Hz]")

### 11.317.1 Detailed Description

Helper file for unit testing.

#### Author

Carles Fernandez-Prades, 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

## 11.318 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 "concurrent_queue.h"
#include "gnss_block_interface.h"
#include "unpack_intspir_lbit_samples.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>
```

### Classes

- class [SpirFileSignalSource](#)

*Class that reads signals samples from a file and adapts it to a `SignalSourceInterface`.*

### 11.318.1 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

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 not part of GNSS-SDR.

SPDX-License-Identifier: GPL-3.0-or-later

## 11.319 spir\_gss6450\_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 "concurrent_queue.h"
#include "gnss_block_interface.h"
#include "gnss_sdr_valve.h"
#include "unpack_spir_gss6450_samples.h"
#include <gnuradio/blocks/deinterleave.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/throttle.h>
#include <gnuradio/hier_block2.h>
#include <pmt/pmt.h>
#include <cstdint>
#include <string>
#include <vector>
```

### Classes

- class [SpirGSS6450FileSignalSource](#)

*Class that reads signals samples from a file and adapts it to a `SignalSourceInterface`.*

### 11.319.1 Detailed Description

Implementation of a class that reads signals samples from a SPIR file and adapts it to a `SignalSourceInterface`.

#### Author

Antonio Ramos, 2017 antonio.ramos(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 not part of GNSS-SDR.

SPDX-License-Identifier: GPL-3.0-or-later

## 11.320 spirent\_motion\_csv\_dump\_reader.h File Reference

Helper file for unit testing.

```
#include <cstdint>
#include <fstream>
#include <string>
#include <vector>
```

## Classes

- class [Spirent\\_Motion\\_Csv\\_Dump\\_Reader](#)

### 11.320.1 Detailed Description

Helper file for unit testing.

#### 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

## 11.321 string\_converter.h File Reference

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

```
#include <stdint>
#include <string>
```

## Classes

- class [StringConverter](#)  
*Class that interprets the contents of a string and converts it into different types.*

### 11.321.1 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)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

## 11.322 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) ((x) < 0 ? -(x) : (x))`
- `#define MIN(x, y) (((x) < (y)) ? (x) : (y))`
- `#define MAX(x, y) (((x) > (y)) ? (x) : (y))`
- `#define CLAMP_DIFF(a, b) (MAX((a), (b)) - (b))`

### 11.322.1 Detailed Description

Common definitions used throughout the libswiftnav library.

#### Author

Henry Hallam [henry@swift-nav.com](mailto:henry@swift-nav.com) Fergus Noble [fergus@swift-nav.com](mailto: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](mailto:henry@swift-nav.com) Fergus Noble [fergus@swift-nav.com](mailto:fergus@swift-nav.com)

SPDX-License-Identifier: LGPL-3.0-only

### 11.323 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](#)

### 11.323.1 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

## 11.324 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

### 11.324.1 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

## 11.325 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.*

### 11.325.1 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

## 11.326 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.*

### 11.326.1 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

## 11.327 test\_flags.h File Reference

Helper file for unit testing.

```
#include <gflags/gflags.h>
#include <string>
```

### Functions

- **DEFINE\_string** (gnuplot\_executable, "", "Gnuplot binary path")
- **DEFINE\_bool** (plot\_acq\_grid, false, "Plots acquisition grid with gnuplot")
- **DEFINE\_int32** (plot\_decimate, 1, "Decimate plots")

### 11.327.1 Detailed Description

Helper file for unit testing.

#### Author

Carles Fernandez-Prades, 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

## 11.328 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](#)

### 11.328.1 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

## 11.329 tlm\_dump\_reader.h File Reference

Helper file for unit testing.

```
#include <cstdint>
#include <fstream>
#include <string>
#include <vector>
```

### Classes

- class [Tlm\\_Dump\\_Reader](#)

### 11.329.1 Detailed Description

Helper file for unit testing.

#### 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

## 11.330 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)

### 11.330.1 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

## 11.331 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.*

### 11.331.1 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

## 11.332 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.*

### 11.332.1 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

## 11.333 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 [fil\\_four\\_quadrant\\_atan](#) (gr\_complex prompt\_s1, gr\_complex prompt\_s2, double t1, double t2)
- double [fil\\_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)

### 11.333.1 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

## 11.334 tracking\_dump\_reader.h File Reference

Helper file for unit testing.

```
#include <cstdint>
#include <fstream>
#include <string>
#include <vector>
```

### Classes

- class [Tracking\\_Dump\\_Reader](#)

#### 11.334.1 Detailed Description

Helper file for unit testing.

##### 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

## 11.335 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.*

#### 11.335.1 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

## 11.336 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 [Concurrent\\_Queue< Data >](#)  
*This class implements a thread-safe std::queue.*
- class [TrackingInterface](#)  
*This abstract class represents an interface to a tracking block.*

### 11.336.1 Detailed Description

This class represents an interface to a tracking block.

#### Author

Carlos Aviles, 2010. carlos.avilesr(at)gmail.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

## 11.337 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.*

### 11.337.1 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

## 11.338 tracking\_tests\_flags.h File Reference

Helper file for unit testing.

```
#include <gflags/gflags.h>
#include <limits>
#include <string>
```

### Functions

- **DEFINE\_string** (trk\_test\_implementation, std::string("GPS\_L1\_CA\_DLL\_PLL\_Tracking"), "Tracking block implementation under test, defaults to GPS\_L1\_CA\_DLL\_PLL\_Tracking")
- **DEFINE\_bool** (enable\_external\_signal\_file, false, "Use an external signal file capture instead of the software-defined signal generator")
- **DEFINE\_double** (external\_signal\_acquisition\_threshold, 2.5, "Threshold for satellite acquisition when external file is used")
- **DEFINE\_int32** (external\_signal\_acquisition\_dwells, 5, "Maximum dwells count for satellite acquisition when external file is used")
- **DEFINE\_double** (external\_signal\_acquisition\_doppler\_max\_hz, 5000.0, "Doppler max for satellite acquisition when external file is used")
- **DEFINE\_double** (external\_signal\_acquisition\_doppler\_step\_hz, 125.0, "Doppler step for satellite acquisition when external file is used")
- **DEFINE\_bool** (use\_acquisition\_resampler, false, "Reduce the sampling rate of the input signal for the acquisition in order to optimize the SNR and decrease the processor load")
- **DEFINE\_string** (signal\_file, std::string("signal\_out.bin"), "Path of the external signal capture file")
- **DEFINE\_double** (CN0\_dBHz\_start, std::numeric\_limits< double >::infinity(), "Enable noise generator and set the CN0 start sweep value [dB-Hz]")
- **DEFINE\_double** (CN0\_dBHz\_stop, std::numeric\_limits< double >::infinity(), "Enable noise generator and set the CN0 stop sweep value [dB-Hz]")
- **DEFINE\_double** (CN0\_dB\_step, 3.0, "Noise generator CN0 sweep step value [dB]")
- **DEFINE\_double** (PLL\_bw\_hz\_start, 20.0, "PLL Wide configuration start sweep value [Hz]")
- **DEFINE\_double** (PLL\_bw\_hz\_stop, 20.0, "PLL Wide configuration stop sweep value [Hz]")
- **DEFINE\_double** (PLL\_bw\_hz\_step, 5.0, "PLL Wide configuration sweep step value [Hz]")
- **DEFINE\_double** (DLL\_bw\_hz\_start, 1.0, "DLL Wide configuration start sweep value [Hz]")
- **DEFINE\_double** (DLL\_bw\_hz\_stop, 1.0, "DLL Wide configuration stop sweep value [Hz]")
- **DEFINE\_double** (DLL\_bw\_hz\_step, 0.25, "DLL Wide configuration sweep step value [Hz]")

- **DEFINE\_double** (fil\_bw\_hz, 4.0, "FLL filter bandwidth [Hz]")
- **DEFINE\_bool** (enable\_fil\_pull\_in, false, "Enable FLL in pull-in phase")
- **DEFINE\_bool** (enable\_fil\_steady\_state, false, "Enable FLL in steady-state phase")
- **DEFINE\_double** (PLL\_narrow\_bw\_hz, 5.0, "PLL Narrow configuration value [Hz]")
- **DEFINE\_double** (DLL\_narrow\_bw\_hz, 0.75, "DLL Narrow configuration value [Hz]")
- **DEFINE\_double** (acq\_Doppler\_error\_hz\_start, 1000.0, "Acquisition Doppler error start sweep value [Hz]")
- **DEFINE\_double** (acq\_Doppler\_error\_hz\_stop, -1000.0, "Acquisition Doppler error stop sweep value [Hz]")
- **DEFINE\_double** (acq\_Doppler\_error\_hz\_step, -50.0, "Acquisition Doppler error sweep step value [Hz]")
- **DEFINE\_double** (acq\_Delay\_error\_chips\_start, 2.0, "Acquisition Code Delay error start sweep value [Chips]")
- **DEFINE\_double** (acq\_Delay\_error\_chips\_stop, -2.0, "Acquisition Code Delay error stop sweep value [Chips]")
- **DEFINE\_double** (acq\_Delay\_error\_chips\_step, -0.1, "Acquisition Code Delay error sweep step value [Chips]")
- **DEFINE\_double** (acq\_to\_trk\_delay\_s, 0.0, "Acquisition to Tracking delay value [s]")
- **DEFINE\_int64** (skip\_samples, 0, "Skip an initial transitory in the processed signal file capture [samples]")
- **DEFINE\_int32** (plot\_detail\_level, 0, "Specify the desired plot detail (0,1,2): 0 - Minimum plots (default) 2 - Plot all tracking parameters")
- **DEFINE\_double** (skip\_trk\_transitory\_s, 1.0, "Skip the initial tracking output signal to avoid transitory results [s]")
- **DEFINE\_int32** (extend\_correlation\_symbols, 1, "Set the tracking coherent correlation to N symbols (up to 20 for GPS L1 C/A)")
- **DEFINE\_int32** (smoother\_length, 10, "Set the moving average size for the carrier phase and code phase in case of high dynamics")
- **DEFINE\_bool** (high\_dyn, false, "Activates the code [resampler](#) and NCO generator for high dynamics")
- **DEFINE\_bool** (plot\_gps\_l1\_tracking\_test, false, "Plots results of GpsL1CADIIPIITrackingTest with gnuplot")

### 11.338.1 Detailed Description

Helper file for unit testing.

#### 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

## 11.339 tracking\_true\_obs\_reader.h File Reference

Helper file for unit testing.

```
#include <stdint>
#include <fstream>
#include <string>
#include <vector>
```

### Classes

- class [Tracking\\_True\\_Obs\\_Reader](#)

### 11.339.1 Detailed Description

Helper file for unit testing.

#### 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

## 11.340 true\_observables\_reader.h File Reference

Helper file for unit testing.

```
#include <stdint>
#include <fstream>
#include <string>
#include <vector>
```

### Classes

- class [True\\_Observables\\_Reader](#)

### 11.340.1 Detailed Description

Helper file for unit testing.

#### 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

## 11.341 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 "concurrent_queue.h"
#include "gnss_block_interface.h"
#include "unpack_byte_2bit_cpx_samples.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/file_source.h>
#include <gnuradio/blocks/interleaved_short_to_complex.h>
#include <gnuradio/blocks/throttle.h>
#include <gnuradio/hier_block2.h>
#include <pmt/pmt.h>
#include <stdint>
#include <string>
```

## Classes

- class [TwoBitCpxFileSignalSource](#)

*Class that reads signals samples from a file and adapts it to a [SignalSourceInterface](#).*

### 11.341.1 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) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

## 11.342 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 "concurrent_queue.h"
#include "gnss_block_interface.h"
#include "unpack_2bit_samples.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/file_source.h>
#include <gnuradio/blocks/interleaved_char_to_complex.h>
#include <gnuradio/blocks/throttle.h>
#include <gnuradio/hier_block2.h>
#include <pmt/pmt.h>
#include <stdint>
#include <string>
```

## Classes

- class [TwoBitPackedFileSignalSource](#)

*Class that reads signals samples from a file and adapts it to a [SignalSourceInterface](#).*

### 11.342.1 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

## 11.343 uhd\_signal\_source.h File Reference

Interface for the Universal Hardware Driver signal source.

```
#include "concurrent_queue.h"
#include "gnss_block_interface.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>)*

### 11.343.1 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

## 11.344 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.*

### 11.344.1 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

## 11.345 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 <stdint>
#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)

### 11.345.1 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  $x_{n+3,1}$   $x_{n+3,0}$   $x_{n+2,1}$   $x_{n+2,0}$   $x_{n+1,1}$   $x_{n+1,0}$   $x_{n,1}$   $x_{n,0}$  : 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

## 11.346 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** ()

### 11.346.1 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

### 11.347 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** ()

### 11.347.1 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

## 11.348 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 <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** = std::shared\_ptr< [unpack\\_byte\\_4bit\\_samples](#) >

### Functions

- unpack\_byte\_4bit\_samples\_sptr **make\_unpack\_byte\_4bit\_samples** ()

### 11.348.1 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

## 11.349 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 ()`

### 11.349.1 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

## 11.350 unpack\_spir\_gss6450\_samples.h File Reference

Unpacks SPIR int samples.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_interpolator.h>
```

### Classes

- class [unpack\\_spir\\_gss6450\\_samples](#)

## Typedefs

- using **unpack\_spir\_gss6450\_samples\_sptr** = gnss\_shared\_ptr< [unpack\\_spir\\_gss6450\\_samples](#) >

## Functions

- `unpack_spir_gss6450_samples_sptr` **make\_unpack\_spir\_gss6450\_samples** (int adc\_nbit\_)

### 11.350.1 Detailed Description

Unpacks SPIR int samples.

#### Author

Antonio Ramos, antonio.ramos(at)cttc.es  
Javier Arribas jarribas (at) cttc.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

## 11.351 viterbi\_decoder.h File Reference

Interface of a Viterbi decoder class based on the Iterative Solutions Coded Modulation Library by Matthew C. Valenti.

```
#include <cstdint>
#include <deque>
#include <vector>
```

## Classes

- class [Viterbi\\_Decoder](#)  
*Class that implements a Viterbi decoder.*

### 11.351.1 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



# Index

- ~Acquisition\_msg\_rx
  - Acquisition\_msg\_rx, [296](#)
- ~ArraySignalConditioner
  - ArraySignalConditioner, [304](#)
- ~Channel
  - Channel, [358](#)
- ~ControlThread
  - ControlThread, [380](#)
- ~Exponential\_Smoothing
  - Exponential\_Smoothing, [397](#)
- ~FirFilter
  - FirFilter, [403](#)
- ~Fpga\_Acquisition
  - Fpga\_Acquisition, [407](#)
- ~Fpga\_Multicorrelator\_8sc
  - Fpga\_Multicorrelator\_8sc, [413](#)
- ~Fpga\_Switch
  - Fpga\_Switch, [417](#)
- ~Fpga\_dynamic\_bit\_selection
  - Fpga\_dynamic\_bit\_selection, [411](#)
- ~GNSSFlowgraph
  - GNSSFlowgraph, [640](#)
- ~GalileoE1DIIPIIVemlTrackingFpga
  - GalileoE1DIIPIIVemlTrackingFpga, [466](#)
- ~GalileoE1PcpsAmbiguousAcquisitionFpga
  - GalileoE1PcpsAmbiguousAcquisitionFpga, [479](#)
- ~GalileoE5aDIIPIITrackingFpga
  - GalileoE5aDIIPIITrackingFpga, [502](#)
- ~GalileoE5aPcpsAcquisitionFpga
  - GalileoE5aPcpsAcquisitionFpga, [515](#)
- ~GalileoE5bPcpsAcquisition
  - GalileoE5bPcpsAcquisition, [526](#)
- ~GalileoE5bPcpsAcquisitionFpga
  - GalileoE5bPcpsAcquisitionFpga, [532](#)
- ~GenSignalSource
  - GenSignalSource, [549](#)
- ~Gnss\_Satellite
  - Gnss\_Satellite, [610](#)
- ~Gnss\_Synchro
  - Gnss\_Synchro, [627](#)
- ~GpsL1CaDIIPIITrackingFpga
  - GpsL1CaDIIPIITrackingFpga, [713](#)
- ~GpsL1CaPcpsAcquisitionFpga
  - GpsL1CaPcpsAcquisitionFpga, [730](#)
- ~GpsL5DIIPIITrackingFpga
  - GpsL5DIIPIITrackingFpga, [771](#)
- ~GpsL5iPcpsAcquisitionFpga
  - GpsL5iPcpsAcquisitionFpga, [780](#)
- ~Nmea\_Printer
  - Nmea\_Printer, [811](#)
- ~Rinex\_Printer
  - Rinex\_Printer, [880](#)
- ~Rtcm\_Printer
  - Rtcm\_Printer, [900](#)
- ~SignalConditioner
  - SignalConditioner, [933](#)
- ~beidou\_b1i\_telemetry\_decoder\_gs
  - beidou\_b1i\_telemetry\_decoder\_gs, [308](#)
- ~beidou\_b3i\_telemetry\_decoder\_gs
  - beidou\_b3i\_telemetry\_decoder\_gs, [310](#)
- ~channel\_msg\_receiver\_cc
  - channel\_msg\_receiver\_cc, [361](#)
- ~channel\_status\_msg\_receiver
  - channel\_status\_msg\_receiver, [362](#)
- ~dll\_pll\_veml\_tracking\_fpga
  - dll\_pll\_veml\_tracking\_fpga, [393](#)
- ~galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc
  - galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc, [429](#)
- ~galileo\_pcps\_8ms\_acquisition\_cc
  - galileo\_pcps\_8ms\_acquisition\_cc, [453](#)
- ~glonass\_l1\_ca\_telemetry\_decoder\_gs
  - glonass\_l1\_ca\_telemetry\_decoder\_gs, [583](#)
- ~glonass\_l2\_ca\_telemetry\_decoder\_gs
  - glonass\_l2\_ca\_telemetry\_decoder\_gs, [587](#)
- ~gnss\_synchro\_monitor
  - gnss\_synchro\_monitor, [636](#)
- ~pcps\_acquisition\_fine\_doppler\_cc
  - pcps\_acquisition\_fine\_doppler\_cc, [828](#)
- ~pcps\_acquisition\_fpga
  - pcps\_acquisition\_fpga, [834](#)
- ~pcps\_assisted\_acquisition\_cc
  - pcps\_assisted\_acquisition\_cc, [840](#)
- ~pcps\_cccwsr\_acquisition\_cc
  - pcps\_cccwsr\_acquisition\_cc, [845](#)
- ~pcps\_openc1\_acquisition\_cc
  - pcps\_openc1\_acquisition\_cc, [849](#)
- ~pcps\_quicksync\_acquisition\_cc
  - pcps\_quicksync\_acquisition\_cc, [855](#)
- ~pcps\_tong\_acquisition\_cc
  - pcps\_tong\_acquisition\_cc, [860](#)
- ~rtklib\_pvt\_gs
  - rtklib\_pvt\_gs, [906](#)
- A\_1
  - Galileo\_Ephemeris, [436](#)
- ALPHA\_0
  - core\_system\_parameters, [231](#)
- ARMODE\_CONT

- algorithms\_libs\_rtklib, [129](#)
- ARMODE\_FIXHOLD
  - algorithms\_libs\_rtklib, [129](#)
- ARMODE\_INST
  - algorithms\_libs\_rtklib, [130](#)
- ARMODE\_OFF
  - algorithms\_libs\_rtklib, [130](#)
- ARMODE\_PPPAR\_ILS
  - algorithms\_libs\_rtklib, [130](#)
- ARMODE\_PPPAR
  - algorithms\_libs\_rtklib, [130](#)
- AS2R
  - core\_system\_parameters, [231](#)
- Acq\_Conf, [293](#)
- acq\_conf.h, [971](#)
- Acq\_delay\_samples
  - Gnss\_Synchro, [629](#)
- Acq\_doppler\_hz
  - Gnss\_Synchro, [629](#)
- Acq\_doppler\_step
  - Gnss\_Synchro, [629](#)
- Acq\_samplestamp\_samples
  - Gnss\_Synchro, [629](#)
- Acquisition, [71](#)
- Acquisition\_Dump\_Reader, [294](#)
  - Acquisition\_Dump\_Reader, [294](#), [295](#)
  - operator=, [295](#)
- acquisition\_adapters, [72](#)
- acquisition\_dump\_reader.h, [971](#)
- acquisition\_gr\_blocks, [74](#)
- acquisition\_interface.h, [972](#)
- acquisition\_libs, [76](#)
- acquisition\_manager
  - GNSSFlowgraph, [641](#)
- Acquisition\_msg\_rx, [295](#)
  - ~Acquisition\_msg\_rx, [296](#)
- acquisition\_msg\_rx.h, [973](#)
- AcquisitionInterface, [297](#)
- ad9361\_fpga\_signal\_source.h, [973](#)
- ad9361\_manager.h, [974](#)
- Ad9361FpgaSignalSource, [298](#)
  - implementation, [298](#)
  - start, [299](#)
- af0\_4
  - Galileo\_Ephemeris, [436](#)
- af1\_4
  - Galileo\_Ephemeris, [436](#)
- af2\_4
  - Galileo\_Ephemeris, [436](#)
- Agnss\_Ref\_Location, [299](#)
  - Agnss\_Ref\_Location, [300](#)
  - serialize, [300](#)
- Agnss\_Ref\_Time, [300](#)
  - Agnss\_Ref\_Time, [301](#)
  - serialize, [301](#)
- agnss\_ref\_location.h, [975](#)
- agnss\_ref\_time.h, [976](#)
- ai0\_5
  - Galileo\_Iono, [449](#)
- ai1\_5
  - Galileo\_Iono, [450](#)
- ai2\_5
  - Galileo\_Iono, [450](#)
- alert
  - cnav\_msg\_t, [368](#)
- Algorithms Common Library, [88](#)
- algorithms\_libs, [89](#)
  - at, [93](#)
  - back, [94](#)
  - beidou\_b1i\_code\_gen\_complex, [94](#)
  - beidou\_b1i\_code\_gen\_complex\_sampled, [94](#)
  - beidou\_b1i\_code\_gen\_float, [94](#)
  - beidou\_b1i\_code\_gen\_int, [95](#)
  - beidou\_b3i\_code\_gen\_complex, [95](#)
  - beidou\_b3i\_code\_gen\_complex\_sampled, [95](#)
  - beidou\_b3i\_code\_gen\_float, [95](#)
  - beidou\_b3i\_code\_gen\_int, [95](#)
  - cart2geo, [96](#)
  - cart2utm, [96](#)
  - clear, [96](#)
  - clksin, [96](#)
  - clsin, [97](#)
  - complex\_exp\_gen, [97](#)
  - complex\_exp\_gen\_conj, [97](#)
  - findUtmZone, [97](#)
  - front, [98](#)
  - galileo\_e1\_code\_gen\_complex\_sampled, [98](#)
  - galileo\_e1\_code\_gen\_float\_sampled, [98](#), [99](#)
  - galileo\_e1\_code\_gen\_sinboc11\_float, [99](#)
  - galileo\_e5\_a\_code\_gen\_complex\_primary, [99](#)
  - galileo\_e5\_a\_code\_gen\_complex\_sampled, [99](#)
  - galileo\_e5\_b\_code\_gen\_complex\_primary, [100](#)
  - galileo\_e5\_b\_code\_gen\_complex\_sampled, [100](#)
  - galileo\_e6\_b\_code\_gen\_complex\_primary, [100](#)
  - galileo\_e6\_b\_code\_gen\_complex\_sampled, [100](#)
  - galileo\_e6\_b\_code\_gen\_float\_primary, [101](#)
  - galileo\_e6\_c\_code\_gen\_complex\_primary, [101](#)
  - galileo\_e6\_c\_code\_gen\_complex\_sampled, [101](#)
  - galileo\_e6\_c\_code\_gen\_float\_primary, [101](#)
  - galileo\_e6\_c\_secondary\_code, [101](#)
  - galileo\_e6\_c\_secondary\_code\_gen\_complex, [102](#)
  - galileo\_e6\_c\_secondary\_code\_gen\_float, [102](#)
  - Geo\_to\_ECEF, [102](#)
  - get, [102](#)
  - glonass\_l1\_ca\_code\_gen\_complex, [103](#)
  - glonass\_l1\_ca\_code\_gen\_complex\_sampled, [103](#)
  - glonass\_l2\_ca\_code\_gen\_complex, [103](#)
  - glonass\_l2\_ca\_code\_gen\_complex\_sampled, [103](#)
  - Gnss\_circular\_deque, [103](#), [104](#)
  - gps\_l1\_ca\_code\_gen\_complex, [104](#)
  - gps\_l1\_ca\_code\_gen\_complex\_sampled, [104](#)
  - gps\_l1\_ca\_code\_gen\_float, [104](#)
  - gps\_l1\_ca\_code\_gen\_int, [105](#)
  - gps\_l2c\_m\_code\_gen\_complex, [105](#)
  - gps\_l2c\_m\_code\_gen\_complex\_sampled, [105](#)
  - gps\_l2c\_m\_code\_gen\_float, [105](#)

- gps\_l5i\_code\_gen\_complex, 105
- gps\_l5i\_code\_gen\_complex\_sampled, 106
- gps\_l5i\_code\_gen\_float, 106
- gps\_l5q\_code\_gen\_complex, 106
- gps\_l5q\_code\_gen\_complex\_sampled, 106
- gps\_l5q\_code\_gen\_float, 106
- Gravity\_ECEF, 107
- great\_circle\_distance, 107
- hex\_to\_binary\_converter, 107
- hex\_to\_binary\_string, 107
- item\_type\_is\_complex, 107
- item\_type\_size, 108
- item\_type\_valid, 108
- make\_vector\_converter, 108
- pop\_front, 109
- push\_back, 109
- pv\_Geo\_to\_ECEF, 109
- resampler, 109, 110
- reset, 110
- size, 110
- Skew\_symmetric, 111
- togeod, 111
- topocent, 111
- algorithms\_libs\_rtklib, 118
  - ARMODE\_CONT, 129
  - ARMODE\_FIXHOLD, 129
  - ARMODE\_INST, 130
  - ARMODE\_OFF, 130
  - ARMODE\_PPPAR\_ILS, 130
  - ARMODE\_PPPAR, 130
  - CHISQR, 130
  - DTTOL, 131
  - EFACT\_BDS, 131
  - EFACT\_GAL, 131
  - EFACT\_GLO, 131
  - EFACT\_GPS, 132
  - EFACT\_IRN, 132
  - EFACT\_QZS, 132
  - EFACT\_SBS, 132
  - EPHOPT\_BRDC, 132
  - EPHOPT\_LEX, 133
  - EPHOPT\_PREC, 133
  - EPHOPT\_SBAS, 133
  - EPHOPT\_SSRAPC, 133
  - EPHOPT\_SSRCOM, 133
  - ERR\_BRDCI, 134
  - ERR\_CBIAS, 134
  - ERR\_SAAS, 134
  - eph\_to\_rtklib, 129
  - FE\_WGS84, 134
  - FTP\_TIMEOUT, 134
  - fatalfunc\_t, 129
  - GAP\_RESION, 135
  - HION, 135
  - INT\_SWAP\_STAT, 135
  - INT\_SWAP\_TRAC, 135
  - IONOOPT\_BRDC, 135
  - IONOOPT\_EST, 136
  - IONOOPT\_IFLC, 136
  - IONOOPT\_LEX, 136
  - IONOOPT\_OFF, 136
  - IONOOPT\_QZS, 136
  - IONOOPT\_SBAS, 137
  - IONOOPT\_STEC, 137
  - IONOOPT\_TEC, 137
  - LAM\_CARR, 137
  - MAXANT, 137
  - MAXBAND, 138
  - MAXCLI, 138
  - MAXDToe\_BDS, 138
  - MAXDToe\_GAL, 138
  - MAXDToe\_GLO, 139
  - MAXDToe\_QZS, 139
  - MAXDToe\_SBS, 139
  - MAXDToe\_S, 139
  - MAXDToe, 138
  - MAXERRMSG, 139
  - MAXEXFILE, 140
  - MAXFREQ, 140
  - MAXGDOP, 140
  - MAXLEAPS, 140
  - MAXNGEO, 140
  - MAXNIGP, 141
  - MAXOBSBUF, 141
  - MAXOBSTYPE, 141
  - MAXOBS, 141
  - MAXPRNBDS, 141
  - MAXPRNGAL, 142
  - MAXPRNGLO, 142
  - MAXPRNGPS, 142
  - MAXPRNSBS, 142
  - MAXRAWLEN, 142
  - MAXRCV, 143
  - MAXSBSAGEF, 143
  - MAXSBSAGEL, 143
  - MAXSBSMSG, 143
  - MAXSBSURA, 143
  - MAXSOLBUF, 144
  - MAXSOLMSG, 144
  - MAXSOLQ, 144
  - MAXSTATMSG, 144
  - MAXSTRMSG, 144
  - MAXSTRPATH, 145
  - MINPRNBDS, 145
  - MINPRNGAL, 145
  - MINPRNGLO, 145
  - MINPRNGPS, 145
  - MINPRNSBS, 146
  - NEXOBS, 146
  - NFREQGLO, 146
  - NFREQ, 146
  - NSATBDS, 146
  - NSATGAL, 147
  - NSATGLO, 147
  - NSATGPS, 147
  - NSATSBS, 147

- NSYS, 147
- PMODE\_DGPS, 148
- PMODE\_FIXED, 148
- PMODE\_KINEMA, 148
- PMODE\_MOVEB, 148
- PMODE\_PPP\_FIXED, 148
- PMODE\_PPP\_KINEMA, 149
- PMODE\_PPP\_STATIC, 149
- PMODE\_SINGLE, 149
- PMODE\_STATIC, 149
- POLYCRC24Q, 149
- POLYCRC32, 150
- POSOPT\_RINEX, 150
- PRN\_HWBIAS, 150
- RE\_WGS84, 150
- REL\_HUMI, 150
- SERIBUFFSIZE, 151
- SOLF\_ENU, 151
- SOLF\_GSIF, 151
- SOLF\_LLH, 151
- SOLF\_NMEA, 151
- SOLF\_STAT, 152
- SOLF\_XYZ, 152
- SOLQ\_DGPS, 152
- SOLQ\_DR, 152
- SOLQ\_FIX, 152
- SOLQ\_FLOAT, 153
- SOLQ\_NONE, 153
- SOLQ\_PPP, 153
- SOLQ\_SBAS, 153
- SOLQ\_SINGLE, 153
- SYS\_ALL, 154
- SYS\_BDS, 154
- SYS\_GAL, 154
- SYS\_GLO, 154
- SYS\_GPS, 154
- SYS\_IRN, 155
- SYS\_LEO, 155
- SYS\_NONE, 155
- SYS\_QZS, 155
- SYS\_SBS, 155
- TIMES\_GPST, 156
- TIMES\_JST, 156
- TIMES\_UTC, 156
- TIMETAGH\_LEN, 156
- TINTACT, 156
- TROPOPT\_CORG, 157
- TROPOPT\_COR, 157
- TROPOPT\_ESTG, 157
- TROPOPT\_EST, 157
- TROPOPT\_OFF, 157
- TROPOPT\_SAAS, 158
- TROPOPT\_SBAS, 158
- alm\_t, 302
- ambc\_t, 302
- apply\_action
  - GNSSFlowgraph, 641
- array\_signal\_conditioner.h, 976
- ArraySignalConditioner, 303
  - ~ArraySignalConditioner, 304
  - ArraySignalConditioner, 303
  - implementation, 304
- at
  - algorithms\_libs, 93
- AU
  - core\_system\_parameters, 232
- Azimuth
  - Gps\_Acq\_Assist, 648
- b\_L2\_P\_data\_flag
  - Gps\_Ephemeris, 683
- b\_alert\_flag
  - Beidou\_Dnav\_Ephemeris, 319
  - Gps\_CNAV\_Ephemeris, 659
  - Gps\_Ephemeris, 682
- b\_antispoofing\_flag
  - Beidou\_Dnav\_Ephemeris, 319
  - Gps\_CNAV\_Ephemeris, 659
  - Gps\_Ephemeris, 683
- b\_fit\_interval\_flag
  - Beidou\_Dnav\_Ephemeris, 320
  - Gps\_Ephemeris, 683
- b\_integrity\_status\_flag
  - Beidou\_Dnav\_Ephemeris, 320
  - Gps\_CNAV\_Ephemeris, 659
  - Gps\_Ephemeris, 683
- b\_sv\_do\_not\_use
  - Sbas\_Ephemeris, 916
- BEIDOU\_B1I\_CODE\_LENGTH\_CHIPS
  - core\_system\_parameters, 232
- BEIDOU\_B1I\_CODE\_PERIOD\_MS
  - core\_system\_parameters, 232
- BEIDOU\_B1I\_CODE\_PERIOD\_S
  - core\_system\_parameters, 232
- BEIDOU\_B1I\_CODE\_RATE\_CPS
  - core\_system\_parameters, 232
- BEIDOU\_B1I\_FREQ\_HZ
  - core\_system\_parameters, 233
- BEIDOU\_B3I\_CODE\_LENGTH\_CHIPS
  - core\_system\_parameters, 233
- BEIDOU\_B3I\_CODE\_PERIOD\_MS
  - core\_system\_parameters, 233
- BEIDOU\_B3I\_CODE\_PERIOD\_S
  - core\_system\_parameters, 233
- BEIDOU\_B3I\_CODE\_RATE\_CPS
  - core\_system\_parameters, 233
- BEIDOU\_B3I\_FREQ\_HZ
  - core\_system\_parameters, 234
- BEIDOU\_B3I\_TELEMETRY\_RATE\_BITS\_SECOND
  - core\_system\_parameters, 234
- BEIDOU\_DNAV\_SUBFRAME\_DATA\_BITS
  - core\_system\_parameters, 234
- BEIDOU\_GM
  - core\_system\_parameters, 234
- BEIDOU\_OMEGA\_EARTH\_DOT
  - core\_system\_parameters, 235
- BEIDOU\_F

- core\_system\_parameters, 234
- BGD\_E1E5a\_5
  - Galileo\_Ephemeris, 436
- BGD\_E1E5b\_5
  - Galileo\_Ephemeris, 437
- back
  - algorithms\_libs, 94
- bayesian\_estimation.h, 977
- Bayesian\_estimator, 304
- beamformer, 305
- beamformer.h, 978
- beamformer\_filter.h, 978
- BeamformerFilter, 306
  - implementation, 306
- Beidou\_B1I.h, 979
- Beidou\_B3I.h, 984
- Beidou\_DNAV.h, 988
- Beidou\_Dnav\_Almanac, 311
  - Beidou\_Dnav\_Almanac, 311
  - d\_A\_f0, 312
  - d\_A\_f1, 312
  - d\_M\_0, 312
  - d\_OMEGA0, 313
  - d\_OMEGA\_DOT, 313
  - d\_OMEGA, 312
  - d\_Toa, 313
  - d\_e\_eccentricity, 312
  - d\_sqrt\_A, 313
  - i\_SV\_health, 314
  - i\_satellite\_PRN, 313
- Beidou\_Dnav\_Ephemeris, 314
  - b\_alert\_flag, 319
  - b\_antispoofing\_flag, 319
  - b\_fit\_interval\_flag, 320
  - b\_integrity\_status\_flag, 320
  - Beidou\_Dnav\_Ephemeris, 317
  - d\_A\_f0, 320
  - d\_A\_f1, 320
  - d\_A\_f2, 321
  - d\_AODC, 321
  - d\_AODE, 321
  - d\_Cic, 321
  - d\_Cis, 322
  - d\_Crc, 322
  - d\_Crs, 322
  - d\_Cuc, 322
  - d\_Cus, 323
  - d\_Delta\_n, 323
  - d\_IDOT, 324
  - d\_M\_0, 324
  - d\_OMEGA0, 325
  - d\_OMEGA\_DOT, 325
  - d\_OMEGA, 324
  - d\_TGD1, 327
  - d\_TGD2, 327
  - d\_TOW, 328
  - d\_Toc, 327
  - d\_Toe, 327
- d\_dtr, 323
- d\_eccentricity, 323
- d\_i\_0, 324
- d\_satClkDrift, 325
- d\_satpos\_X, 325
- d\_satpos\_Y, 326
- d\_satpos\_Z, 326
- d\_satvel\_X, 326
- d\_satvel\_Y, 326
- d\_satvel\_Z, 326
- d\_sqrt\_A, 327
- i\_AODO, 328
- i\_BEIDOU\_week, 328
- i\_SV\_accuracy, 329
- i\_nav\_type, 328
- i\_satellite\_PRN, 329
- i\_sig\_type, 329
- satelliteBlock, 329
- satellitePosition, 317
- serialize, 317
- sv\_clock\_drift, 319
- sv\_clock\_relativistic\_term, 319
- Beidou\_Dnav\_Iono, 330
  - Beidou\_Dnav\_Iono, 330
  - d\_alpha0, 331
  - d\_alpha1, 331
  - d\_alpha2, 331
  - d\_alpha3, 332
  - d\_beta0, 332
  - d\_beta1, 332
  - d\_beta2, 332
  - d\_beta3, 333
  - serialize, 331
  - valid, 333
- Beidou\_Dnav\_Navigation\_Message, 333
  - Beidou\_Dnav\_Navigation\_Message, 334
  - d1\_subframe\_decoder, 335
  - d2\_subframe\_decoder, 335
  - get\_ephemeris, 335
  - get\_iono, 335
  - get\_utc\_model, 335
  - have\_new\_almanac, 335
  - have\_new\_ephemeris, 336
  - have\_new\_iono, 336
  - have\_new\_utc\_model, 336
  - satellitePosition, 336
  - set\_satellite\_PRN, 336
  - sv\_clock\_correction, 336
  - utc\_time, 337
- Beidou\_Dnav\_Utc\_Model, 337
  - d\_A0\_GAL, 338
  - d\_A0\_GLO, 338
  - d\_A0\_GPS, 338
  - d\_A0\_UTC, 338
  - d\_A1\_GAL, 338
  - d\_A1\_GLO, 339
  - d\_A1\_GPS, 339
  - d\_A1\_UTC, 339

- d\_DeltaT\_LSF, 339
- i\_DeltaT\_LS, 339
- i\_DN, 340
- i\_WN\_LSF, 340
- beidou\_b1i\_code\_gen\_complex
  - algorithms\_libs, 94
- beidou\_b1i\_code\_gen\_complex\_sampled
  - algorithms\_libs, 94
- beidou\_b1i\_code\_gen\_float
  - algorithms\_libs, 94
- beidou\_b1i\_code\_gen\_int
  - algorithms\_libs, 95
- beidou\_b1i\_dll\_pll\_tracking.h, 980
- beidou\_b1i\_pcps\_acquisition.h, 981
- beidou\_b1i\_signal\_replica.h, 981
- beidou\_b1i\_telemetry\_decoder.h, 982
- beidou\_b1i\_telemetry\_decoder\_gs, 307
  - ~beidou\_b1i\_telemetry\_decoder\_gs, 308
  - general\_work, 308
  - set\_channel, 308
  - set\_satellite, 308
- beidou\_b1i\_telemetry\_decoder\_gs.h, 983
- beidou\_b3i\_code\_gen\_complex
  - algorithms\_libs, 95
- beidou\_b3i\_code\_gen\_complex\_sampled
  - algorithms\_libs, 95
- beidou\_b3i\_code\_gen\_float
  - algorithms\_libs, 95
- beidou\_b3i\_code\_gen\_int
  - algorithms\_libs, 95
- beidou\_b3i\_dll\_pll\_tracking.h, 985
- beidou\_b3i\_pcps\_acquisition.h, 985
- beidou\_b3i\_signal\_replica.h, 986
- beidou\_b3i\_telemetry\_decoder.h, 987
- beidou\_b3i\_telemetry\_decoder\_gs, 309
  - ~beidou\_b3i\_telemetry\_decoder\_gs, 310
  - general\_work, 310
  - set\_channel, 310
  - set\_satellite, 310
- beidou\_b3i\_telemetry\_decoder\_gs.h, 987
- beidou\_dnav\_almanac.h, 993
- beidou\_dnav\_ephemeris.h, 993
- beidou\_dnav\_ephemeris\_map
  - Rtklib\_Solver, 909
- beidou\_dnav\_iono.h, 994
- beidou\_dnav\_navigation\_message.h, 994
- beidou\_dnav\_utc\_model.h, 995
- BeidouB1iDIIPIITracking, 340
  - set\_channel, 341
  - set\_gnss\_synchro, 341
  - stop\_tracking, 341
- BeidouB1iPcpsAcquisition, 342
  - implementation, 343
  - init, 343
  - mag, 343
  - reset, 344
  - set\_channel, 344
  - set\_channel\_fsm, 344
  - set\_doppler\_max, 344
  - set\_doppler\_step, 344
  - set\_gnss\_synchro, 345
  - set\_local\_code, 345
  - set\_resampler\_latency, 345
  - set\_state, 345
  - set\_threshold, 345
  - stop\_acquisition, 346
- BeidouB1iTelemetryDecoder, 346
  - implementation, 347
- BeidouB3iDIIPIITracking, 347
  - set\_channel, 348
  - set\_gnss\_synchro, 348
  - stop\_tracking, 348
- BeidouB3iPcpsAcquisition, 349
  - implementation, 350
  - init, 350
  - mag, 350
  - reset, 351
  - set\_channel, 351
  - set\_channel\_fsm, 351
  - set\_doppler\_max, 351
  - set\_doppler\_step, 352
  - set\_gnss\_synchro, 352
  - set\_local\_code, 352
  - set\_resampler\_latency, 352
  - set\_state, 352
  - set\_threshold, 353
  - stop\_acquisition, 353
- BeidouB3iTelemetryDecoder, 353
  - implementation, 354
- bin\_to\_binary\_data
  - Rtcm, 887
- bin\_to\_double
  - Rtcm, 887
- bin\_to\_hex
  - Rtcm, 888
- bin\_to\_uint
  - Rtcm, 888
- binary\_data\_to\_bin
  - Rtcm, 888
- bit\_selection
  - Fpga\_dynamic\_bit\_selection, 411
- bits.h, 995
- byte\_to\_short.h, 996
- byte\_x2\_to\_complex\_byte, 354
- byte\_x2\_to\_complex\_byte.h, 997
- ByteToShort, 355
  - implementation, 356
- C\_ic\_4
  - Galileo\_Ephemeris, 437
- C\_is\_4
  - Galileo\_Ephemeris, 437
- C\_rc\_3
  - Galileo\_Ephemeris, 437
- C\_rs\_3
  - Galileo\_Ephemeris, 438
- C\_uc\_3

- Galileo\_Ephemeris, [438](#)
- C\_us\_3
  - Galileo\_Ephemeris, [438](#)
- CHISQR
  - algorithms\_libs\_rtklib, [130](#)
- CN0\_dB\_hz
  - Gnss\_Synchro, [630](#)
- CODE\_L1A
  - core\_system\_parameters, [235](#)
- CODE\_L1B
  - core\_system\_parameters, [235](#)
- CODE\_L1C
  - core\_system\_parameters, [235](#)
- CODE\_L1E
  - core\_system\_parameters, [235](#)
- CODE\_L1I
  - core\_system\_parameters, [236](#)
- CODE\_L1L
  - core\_system\_parameters, [236](#)
- CODE\_L1M
  - core\_system\_parameters, [236](#)
- CODE\_L1N
  - core\_system\_parameters, [236](#)
- CODE\_L1P
  - core\_system\_parameters, [236](#)
- CODE\_L1Q
  - core\_system\_parameters, [237](#)
- CODE\_L1S
  - core\_system\_parameters, [237](#)
- CODE\_L1W
  - core\_system\_parameters, [237](#)
- CODE\_L1X
  - core\_system\_parameters, [237](#)
- CODE\_L1Y
  - core\_system\_parameters, [237](#)
- CODE\_L1Z
  - core\_system\_parameters, [238](#)
- CODE\_L2C
  - core\_system\_parameters, [238](#)
- CODE\_L2D
  - core\_system\_parameters, [238](#)
- CODE\_L2I
  - core\_system\_parameters, [238](#)
- CODE\_L2L
  - core\_system\_parameters, [238](#)
- CODE\_L2M
  - core\_system\_parameters, [239](#)
- CODE\_L2N
  - core\_system\_parameters, [239](#)
- CODE\_L2P
  - core\_system\_parameters, [239](#)
- CODE\_L2Q
  - core\_system\_parameters, [239](#)
- CODE\_L2S
  - core\_system\_parameters, [239](#)
- CODE\_L2W
  - core\_system\_parameters, [240](#)
- CODE\_L2X
  - core\_system\_parameters, [240](#)
- CODE\_L2Y
  - core\_system\_parameters, [240](#)
- CODE\_L3I
  - core\_system\_parameters, [240](#)
- CODE\_L3Q
  - core\_system\_parameters, [240](#)
- CODE\_L3X
  - core\_system\_parameters, [241](#)
- CODE\_L5A
  - core\_system\_parameters, [241](#)
- CODE\_L5B
  - core\_system\_parameters, [241](#)
- CODE\_L5C
  - core\_system\_parameters, [241](#)
- CODE\_L5I
  - core\_system\_parameters, [241](#)
- CODE\_L5Q
  - core\_system\_parameters, [242](#)
- CODE\_L5X
  - core\_system\_parameters, [242](#)
- CODE\_L6A
  - core\_system\_parameters, [242](#)
- CODE\_L6B
  - core\_system\_parameters, [242](#)
- CODE\_L6C
  - core\_system\_parameters, [242](#)
- CODE\_L6I
  - core\_system\_parameters, [243](#)
- CODE\_L6L
  - core\_system\_parameters, [243](#)
- CODE\_L6Q
  - core\_system\_parameters, [243](#)
- CODE\_L6S
  - core\_system\_parameters, [243](#)
- CODE\_L6X
  - core\_system\_parameters, [243](#)
- CODE\_L6Z
  - core\_system\_parameters, [244](#)
- CODE\_L7I
  - core\_system\_parameters, [244](#)
- CODE\_L7Q
  - core\_system\_parameters, [244](#)
- CODE\_L7X
  - core\_system\_parameters, [244](#)
- CODE\_L8I
  - core\_system\_parameters, [244](#)
- CODE\_L8Q
  - core\_system\_parameters, [245](#)
- CODE\_L8X
  - core\_system\_parameters, [245](#)
- CODE\_L9A
  - core\_system\_parameters, [245](#)
- CODE\_L9B
  - core\_system\_parameters, [245](#)
- CODE\_L9C
  - core\_system\_parameters, [245](#)
- CODE\_L9X
  - core\_system\_parameters, [245](#)

- core\_system\_parameters, 246
- CODE\_NONE
  - core\_system\_parameters, 246
- CODES\_BDS
  - rtklib\_rtc3.h, 1222
- CODES\_GAL
  - rtklib\_rtc3.h, 1222
- CODES\_GLO
  - rtklib\_rtc3.h, 1223
- CODES\_GPS
  - rtklib\_rtc3.h, 1223
- CODES\_QZS
  - rtklib\_rtc3.h, 1223
- CODES\_SBS
  - rtklib\_rtc3.h, 1224
- CRC\_test
  - Glonass\_Gnav\_Navigation\_Message, 574
- Carrier\_Doppler\_hz
  - Gnss\_Synchro, 629
- carrier\_lock\_detector
  - tracking\_libs, 189
- Carrier\_phase\_rads
  - Gnss\_Synchro, 630
- Carrier\_wipeoff\_multicorrelator\_resampler
  - Fpga\_Multicorrelator\_8sc, 413
- cart2geo
  - algorithms\_libs, 96
- cart2utm
  - algorithms\_libs, 96
- Channel, 77, 356
  - ~Channel, 358
  - Channel, 357
  - connect, 358
  - get\_left\_block\_acq, 358
  - get\_left\_block\_trk, 358
  - get\_right\_block, 359
  - get\_right\_block\_acq, 359
  - get\_right\_block\_trk, 359
  - implementation, 359
  - set\_signal, 359
  - start\_acquisition, 360
  - stop\_channel, 360
- channel.h, 998
- Channel\_Event, 360
- Channel\_ID
  - Gnss\_Synchro, 630
- channel\_adapters, 78
- channel\_event.h, 998
- channel\_fsm.h, 999
- channel\_interface.h, 1000
- channel\_libs, 79
- channel\_msg\_receiver\_cc, 361
  - ~channel\_msg\_receiver\_cc, 361
- channel\_msg\_receiver\_cc.h, 1000
- channel\_status\_msg\_receiver, 362
  - ~channel\_status\_msg\_receiver, 362
  - get\_current\_status\_map, 363
  - get\_current\_status\_pvt, 363
- channel\_status\_msg\_receiver.h, 1001
- ChannelFsm, 363
- ChannelInterface, 364
- check\_CRC
  - Rtc3, 888
- cl\_fft\_plan, 365
- clFFT.h, 1002
- clFFT\_Complex, 365
- clFFT\_Dim3, 366
- clFFT\_SplitComplex, 366
- clear
  - algorithms\_libs, 96
- clear\_ephemeris
  - rtklib\_pvt\_gs, 906
- clksin
  - algorithms\_libs, 96
- close\_device
  - Fpga\_Acquisition, 408
- clsln
  - algorithms\_libs, 97
- cn0\_m2m4\_estimator
  - tracking\_libs, 190
- cn0\_svn\_estimator
  - tracking\_libs, 190
- cnav\_msg.h, 1003
- cnav\_msg\_decoder\_t, 366
  - part1, 367
  - part2, 367
- cnav\_msg\_t, 367
  - alert, 368
  - msg\_id, 368
  - prn, 368
  - raw\_msg, 368
  - tow, 369
- cnav\_v27\_part\_t, 369
  - crc\_ok, 370
  - dec, 370
  - decisions, 370
  - decoded, 370
  - init, 370
  - invert, 370
  - message\_lock, 371
  - n\_crc\_fail, 371
  - n\_decoded, 371
  - n\_symbols, 371
  - preamble\_seen, 371
  - symbols, 372
- Code\_Phase
  - Gps\_Acq\_Assist, 648
- Code\_Phase\_int
  - Gps\_Acq\_Assist, 648
- Code\_Phase\_window
  - Gps\_Acq\_Assist, 648
- Code\_phase\_samples
  - Gnss\_Synchro, 630
- Command\_Event, 372
- command\_event.h, 1004
- complex\_byte\_to\_float\_x2, 373

- complex\_byte\_to\_float\_x2.h, 1004
- complex\_exp\_gen
  - algorithms\_libs, 97
- complex\_exp\_gen\_conj
  - algorithms\_libs, 97
- complex\_float\_to\_complex\_byte, 373
- complex\_float\_to\_complex\_byte.h, 1005
- compute\_GLONASS\_time
  - Glonass\_Gnav\_Ephemeris, 560
- Concurrent\_Map< Data >, 374
- Concurrent\_Queue< Data >, 375
- concurrent\_map.h, 1006
- concurrent\_queue.h, 1006
- conditioner\_adapters, 81
- configuration\_interface.h, 1007
- ConfigurationInterface, 375
- configure\_acquisition
  - Fpga\_Acquisition, 408
- conjugate\_cc, 376
- conjugate\_cc.h, 1008
- conjugate\_ic, 377
- conjugate\_ic.h, 1008
- conjugate\_sc, 378
- conjugate\_sc.h, 1009
- connect
  - Channel, 358
  - GNSSFlowgraph, 641
  - GalileoE1DIIPIIVemlTrackingFpga, 466
  - GalileoE1PcpsAmbiguousAcquisitionFpga, 479
  - GalileoE5aDIIPIITrackingFpga, 502
  - GalileoE5aPcpsAcquisitionFpga, 515
  - GalileoE5bDIIPIITracking, 522
  - GalileoE5bPcpsAcquisition, 526
  - GalileoE5bPcpsAcquisitionFpga, 533
  - GalileoE5bTelemetryDecoder, 538
  - GalileoE6DIIPIITracking, 540
  - GalileoE6TelemetryDecoder, 547
  - GpsL1CaDIIPIITrackingFpga, 713
  - GpsL1CaPcpsAcquisitionFpga, 730
  - GpsL5DIIPIITrackingFpga, 771
  - GpsL5iPcpsAcquisitionFpga, 780
- control\_thread.h, 1010
- ControlThread, 378
  - ~ControlThread, 380
  - ControlThread, 379
  - flowgraph, 380
  - run, 380
  - set\_control\_queue, 380
- convolutional.h, 1011
- Core GNSS Receiver, 193
- core\_libs, 195
  - find\_uio\_dev\_file\_name, 196
  - ini\_parse, 196
- core\_monitor, 197
- core\_receiver, 198
- core\_system\_parameters, 199
  - ALPHA\_0, 231
  - AS2R, 231
  - AU, 232
  - BEIDOU\_B1I\_CODE\_LENGTH\_CHIPS, 232
  - BEIDOU\_B1I\_CODE\_PERIOD\_MS, 232
  - BEIDOU\_B1I\_CODE\_PERIOD\_S, 232
  - BEIDOU\_B1I\_CODE\_RATE\_CPS, 232
  - BEIDOU\_B1I\_FREQ\_HZ, 233
  - BEIDOU\_B3I\_CODE\_LENGTH\_CHIPS, 233
  - BEIDOU\_B3I\_CODE\_PERIOD\_MS, 233
  - BEIDOU\_B3I\_CODE\_PERIOD\_S, 233
  - BEIDOU\_B3I\_CODE\_RATE\_CPS, 233
  - BEIDOU\_B3I\_FREQ\_HZ, 234
  - BEIDOU\_B3I\_TELEMETRY\_RATE\_BITS\_SEC↵
    - OND, 234
  - BEIDOU\_DNAV\_SUBFRAME\_DATA\_BITS, 234
  - BEIDOU\_GM, 234
  - BEIDOU\_OMEGA\_EARTH\_DOT, 235
  - BEIDOU\_F, 234
  - CODE\_L1A, 235
  - CODE\_L1B, 235
  - CODE\_L1C, 235
  - CODE\_L1E, 235
  - CODE\_L1I, 236
  - CODE\_L1L, 236
  - CODE\_L1M, 236
  - CODE\_L1N, 236
  - CODE\_L1P, 236
  - CODE\_L1Q, 237
  - CODE\_L1S, 237
  - CODE\_L1W, 237
  - CODE\_L1X, 237
  - CODE\_L1Y, 237
  - CODE\_L1Z, 238
  - CODE\_L2C, 238
  - CODE\_L2D, 238
  - CODE\_L2I, 238
  - CODE\_L2L, 238
  - CODE\_L2M, 239
  - CODE\_L2N, 239
  - CODE\_L2P, 239
  - CODE\_L2Q, 239
  - CODE\_L2S, 239
  - CODE\_L2W, 240
  - CODE\_L2X, 240
  - CODE\_L2Y, 240
  - CODE\_L3I, 240
  - CODE\_L3Q, 240
  - CODE\_L3X, 241
  - CODE\_L5A, 241
  - CODE\_L5B, 241
  - CODE\_L5C, 241
  - CODE\_L5I, 241
  - CODE\_L5Q, 242
  - CODE\_L5X, 242
  - CODE\_L6A, 242
  - CODE\_L6B, 242
  - CODE\_L6C, 242
  - CODE\_L6I, 243
  - CODE\_L6L, 243

- CODE\_L6Q, [243](#)
- CODE\_L6S, [243](#)
- CODE\_L6X, [243](#)
- CODE\_L6Z, [244](#)
- CODE\_L7I, [244](#)
- CODE\_L7Q, [244](#)
- CODE\_L7X, [244](#)
- CODE\_L8I, [244](#)
- CODE\_L8Q, [245](#)
- CODE\_L8X, [245](#)
- CODE\_L9A, [245](#)
- CODE\_L9B, [245](#)
- CODE\_L9C, [245](#)
- CODE\_L9X, [246](#)
- CODE\_NONE, [246](#)
- D2R, [246](#)
- DFRQ1\_GLO, [246](#)
- DFRQ2\_GLO, [246](#)
- FREQ1, [247](#)
- FREQ1\_BDS, [247](#)
- FREQ1\_GLO, [247](#)
- FREQ2, [247](#)
- FREQ2\_BDS, [247](#)
- FREQ2\_GLO, [248](#)
- FREQ3\_BDS, [248](#)
- FREQ3\_GLO, [248](#)
- FREQ5, [248](#)
- FREQ6, [248](#)
- FREQ7, [249](#)
- FREQ8, [249](#)
- FREQ9, [249](#)
- GALILEO\_CNAV\_PAGE\_MS, [249](#)
- GALILEO\_CNAV\_SYMBOLS\_PER\_PAGE, [249](#)
- GALILEO\_E1\_B\_CODE\_LENGTH\_CHIPS, [250](#)
- GALILEO\_E1\_B\_SAMPLES\_PER\_SYMBOL, [250](#)
- GALILEO\_E1\_B\_SYMBOL\_RATE\_BPS, [250](#)
- GALILEO\_E1\_C\_SECONDARY\_CODE\_LENGTH↵  
TH, [250](#)
- GALILEO\_E1\_CODE\_CHIP\_RATE\_CPS, [250](#)
- GALILEO\_E1\_CODE\_PERIOD\_MS, [251](#)
- GALILEO\_E1\_CODE\_PERIOD\_S, [251](#)
- GALILEO\_E1\_FREQ\_HZ, [251](#)
- GALILEO\_E1\_HISTORY\_DEEP, [251](#)
- GALILEO\_E1\_OPT\_ACQ\_FS\_SPS, [251](#)
- GALILEO\_E1\_SUB\_CARRIER\_A\_RATE\_HZ, [252](#)
- GALILEO\_E1\_SUB\_CARRIER\_B\_RATE\_HZ, [252](#)
- GALILEO\_E5A\_CODE\_CHIP\_RATE\_CPS, [252](#)
- GALILEO\_E5A\_CODE\_LENGTH\_CHIPS, [252](#)
- GALILEO\_E5A\_CODE\_PERIOD\_MS, [252](#)
- GALILEO\_E5A\_CODE\_PERIOD\_S, [253](#)
- GALILEO\_E5A\_FREQ\_HZ, [253](#)
- GALILEO\_E5A\_I\_SECONDARY\_CODE\_LENGTH↵  
TH, [253](#)
- GALILEO\_E5A\_I\_TIERED\_CODE\_PERIOD\_S, [253](#)
- GALILEO\_E5A\_OPT\_ACQ\_FS\_SPS, [253](#)
- GALILEO\_E5A\_Q\_SECONDARY\_CODE\_LENGTH↵  
GTH, [254](#)
- GALILEO\_E5A\_Q\_TIERED\_CODE\_PERIOD\_S, [254](#)
- GALILEO\_E5A\_SYMBOL\_RATE\_BPS, [254](#)
- GALILEO\_E5B\_CODE\_CHIP\_RATE\_CPS, [254](#)
- GALILEO\_E5B\_CODE\_LENGTH\_CHIPS, [254](#)
- GALILEO\_E5B\_CODE\_PERIOD\_MS, [255](#)
- GALILEO\_E5B\_CODE\_PERIOD\_S, [255](#)
- GALILEO\_E5B\_FREQ\_HZ, [255](#)
- GALILEO\_E5B\_I\_SECONDARY\_CODE\_LENGTH↵  
TH, [255](#)
- GALILEO\_E5B\_I\_TIERED\_CODE\_PERIOD\_S, [255](#)
- GALILEO\_E5B\_OPT\_ACQ\_FS\_SPS, [256](#)
- GALILEO\_E5B\_Q\_SECONDARY\_CODE\_LENGTH↵  
GTH, [256](#)
- GALILEO\_E5B\_Q\_TIERED\_CODE\_PERIOD\_S, [256](#)
- GALILEO\_E5B\_SYMBOL\_RATE\_BPS, [256](#)
- GALILEO\_E6\_B\_CODE\_CHIP\_RATE\_CPS, [256](#)
- GALILEO\_E6\_B\_CODE\_LENGTH\_CHIPS, [257](#)
- GALILEO\_E6\_C\_CODE\_CHIP\_RATE\_CPS, [257](#)
- GALILEO\_E6\_C\_CODE\_LENGTH\_CHIPS, [257](#)
- GALILEO\_E6\_C\_SECONDARY\_CODE\_LENGTH↵  
TH\_CHIPS, [257](#)
- GALILEO\_E6\_CODE\_PERIOD\_MS, [257](#)
- GALILEO\_E6\_CODE\_PERIOD\_S, [258](#)
- GALILEO\_E6\_FREQ\_HZ, [258](#)
- GALILEO\_GM, [258](#)
- GALILEO\_INAV\_PAGE\_PART\_SYMBOLS, [258](#)
- GALILEO\_INAV\_PAGE\_PART\_WITH\_PREABL↵  
E\_SECONDS, [259](#)
- GALILEO\_INAV\_PAGE\_SYMBOLS, [259](#)
- GALILEO\_F, [258](#)
- GLONASS\_C20, [259](#)
- GLONASS\_EARTH\_INCLINATION, [259](#)
- GLONASS\_EARTH\_RADIUS, [259](#)
- GLONASS\_F\_M\_A, [260](#)
- GLONASS\_FLATTENING, [260](#)
- GLONASS\_GNAV\_HAMMING\_CODE\_BITS, [260](#)
- GLONASS\_GNAV\_PREAMBLE, [231](#)
- GLONASS\_GNAV\_STRING\_BITS, [260](#)
- GLONASS\_GNAV\_STRING\_SYMBOLS, [261](#)
- GLONASS\_GNAV\_TELEMETRY\_RATE\_BITS↵  
SECOND, [261](#)
- GLONASS\_GNAV\_TELEMETRY\_RATE\_SYMB↵  
OLS\_SECOND, [261](#)
- GLONASS\_GRAVITY\_CORRECTION, [261](#)
- GLONASS\_GRAVITY, [261](#)
- GLONASS\_GM, [260](#)
- GLONASS\_J2, [262](#)
- GLONASS\_J4, [262](#)
- GLONASS\_J6, [262](#)
- GLONASS\_J8, [262](#)
- GLONASS\_L1\_CA\_CHIP\_PERIOD\_S, [262](#)
- GLONASS\_L1\_CA\_CODE\_LENGTH\_CHIPS, [263](#)
- GLONASS\_L1\_CA\_CODE\_PERIOD\_S, [263](#)
- GLONASS\_L1\_CA\_CODE\_RATE\_CPS, [263](#)
- GLONASS\_L1\_CA\_DFREQ\_HZ, [263](#)

- GLONASS\_L1\_CA\_FREQ\_HZ, [263](#)
- GLONASS\_L2\_CA\_CHIP\_PERIOD\_S, [264](#)
- GLONASS\_L2\_CA\_CODE\_LENGTH\_CHIPS, [264](#)
- GLONASS\_L2\_CA\_CODE\_PERIOD\_S, [264](#)
- GLONASS\_L2\_CA\_CODE\_RATE\_CPS, [264](#)
- GLONASS\_L2\_CA\_DFREQ\_HZ, [264](#)
- GLONASS\_L2\_CA\_FREQ\_HZ, [265](#)
- GLONASS\_LEAP\_SECONDS, [265](#)
- GLONASS\_MOON\_ECCENTRICITY, [265](#)
- GLONASS\_MOON\_GM, [266](#)
- GLONASS\_MOON\_INCLINATION, [266](#)
- GLONASS\_MOON\_OMEGA\_0, [266](#)
- GLONASS\_MOON\_OMEGA\_1, [266](#)
- GLONASS\_MOON\_Q0, [266](#)
- GLONASS\_MOON\_Q1, [267](#)
- GLONASS\_MOON\_SEMI\_MAJOR\_AXIS, [267](#)
- GLONASS\_OMEGA\_EARTH\_DOT, [267](#)
- GLONASS\_SEMI\_MAJOR\_AXIS, [267](#)
- GLONASS\_SUN\_ECCENTRICITY, [267](#)
- GLONASS\_SUN\_GM, [268](#)
- GLONASS\_SUN\_OMEGA, [268](#)
- GLONASS\_SUN\_Q0, [268](#)
- GLONASS\_SUN\_Q1, [268](#)
- GLONASS\_SUN\_SEMI\_MAJOR\_AXIS, [268](#)
- GLONASS\_TAU\_0, [269](#)
- GLONASS\_TAU\_1, [269](#)
- GLONASS\_U0, [269](#)
- GNSS\_OMEGA\_EARTH\_DOT, [269](#)
- GNSS\_PI, [269](#)
- GPS\_CA\_TELEMETRY\_RATE\_BITS\_SECOND, [270](#)
- GPS\_CA\_TELEMETRY\_RATE\_SYMBOLS\_SECOND, [270](#)
- GPS\_GM, [270](#)
- GPS\_L1\_CA\_BIT\_PERIOD\_MS, [270](#)
- GPS\_L1\_CA\_CHIP\_PERIOD\_S, [271](#)
- GPS\_L1\_CA\_CODE\_LENGTH\_CHIPS, [271](#)
- GPS\_L1\_CA\_CODE\_PERIOD\_MS, [271](#)
- GPS\_L1\_CA\_CODE\_PERIOD\_S, [271](#)
- GPS\_L1\_CA\_CODE\_RATE\_CPS, [271](#)
- GPS\_L1\_CA\_OPT\_ACQ\_FS\_SPS, [272](#)
- GPS\_L1\_FREQ\_HZ, [272](#)
- GPS\_L2\_CNAV\_DATA\_PAGE\_BITS, [272](#)
- GPS\_L2\_FREQ\_HZ, [272](#)
- GPS\_L2\_L\_CODE\_LENGTH\_CHIPS, [272](#)
- GPS\_L2\_L\_CODE\_RATE\_CPS, [273](#)
- GPS\_L2\_L\_PERIOD\_S, [273](#)
- GPS\_L2\_M\_CODE\_LENGTH\_CHIPS, [273](#)
- GPS\_L2\_M\_CODE\_RATE\_CPS, [273](#)
- GPS\_L2\_M\_PERIOD\_S, [273](#)
- GPS\_L2C\_M\_INIT\_REG, [274](#)
- GPS\_L2C\_OPT\_ACQ\_FS\_SPS, [274](#)
- GPS\_L5\_CNAV\_DATA\_PAGE\_BITS, [274](#)
- GPS\_L5\_FREQ\_HZ, [275](#)
- GPS\_L5\_OPT\_ACQ\_FS\_SPS, [275](#)
- GPS\_L5I\_CODE\_LENGTH\_CHIPS, [275](#)
- GPS\_L5I\_CODE\_RATE\_CPS, [275](#)
- GPS\_L5I\_PERIOD\_MS, [275](#)
- GPS\_L5I\_PERIOD\_S, [276](#)
- GPS\_L5I\_SYMBOL\_PERIOD\_MS, [276](#)
- GPS\_L5I\_SYMBOL\_PERIOD\_S, [276](#)
- GPS\_L5Q\_CODE\_LENGTH\_CHIPS, [276](#)
- GPS\_L5Q\_CODE\_RATE\_CPS, [276](#)
- GPS\_L5Q\_PERIOD\_S, [277](#)
- GPS\_SUBFRAME\_BITS, [277](#)
- GPS\_SUBFRAME\_LENGTH, [277](#)
- GPS\_SUBFRAME\_MS, [277](#)
- GPS\_SUBFRAME\_SECONDS, [277](#)
- GPS\_WORD\_BITS, [278](#)
- GPS\_WORD\_LENGTH, [278](#)
- GPS\_F, [270](#)
- HALF\_PI, [278](#)
- MAX\_TOA\_DELAY\_MS, [278](#)
- MAXCODE, [278](#)
- PI\_TWO\_N19, [279](#)
- PI\_TWO\_N23, [279](#)
- PI\_TWO\_N31, [279](#)
- PI\_TWO\_N38, [279](#)
- PI\_TWO\_N43, [279](#)
- R2D, [280](#)
- SC2RAD, [280](#)
- SPEED\_OF\_LIGHT\_M\_MS, [280](#)
- SPEED\_OF\_LIGHT\_M\_S, [280](#)
- T\_OA, [231](#)
- TWO\_N10, [280](#)
- TWO\_N11, [281](#)
- TWO\_N14, [281](#)
- TWO\_N15, [281](#)
- TWO\_N16, [281](#)
- TWO\_N17, [281](#)
- TWO\_N18, [282](#)
- TWO\_N19, [282](#)
- TWO\_N2, [282](#)
- TWO\_N20, [282](#)
- TWO\_N21, [282](#)
- TWO\_N23, [283](#)
- TWO\_N24, [283](#)
- TWO\_N25, [283](#)
- TWO\_N27, [283](#)
- TWO\_N29, [283](#)
- TWO\_N30, [284](#)
- TWO\_N31, [284](#)
- TWO\_N32, [284](#)
- TWO\_N33, [284](#)
- TWO\_N34, [284](#)
- TWO\_N35, [285](#)
- TWO\_N38, [285](#)
- TWO\_N39, [285](#)
- TWO\_N40, [285](#)
- TWO\_N43, [285](#)
- TWO\_N44, [286](#)
- TWO\_N46, [286](#)
- TWO\_N48, [286](#)
- TWO\_N5, [286](#)
- TWO\_N50, [286](#)
- TWO\_N51, [287](#)

- TWO\_N55, [287](#)
- TWO\_N57, [287](#)
- TWO\_N59, [287](#)
- TWO\_N6, [287](#)
- TWO\_N60, [288](#)
- TWO\_N66, [288](#)
- TWO\_N68, [288](#)
- TWO\_N8, [288](#)
- TWO\_N9, [288](#)
- TWO\_P11, [289](#)
- TWO\_P12, [289](#)
- TWO\_P14, [289](#)
- TWO\_P16, [289](#)
- TWO\_P19, [289](#)
- TWO\_P3, [290](#)
- TWO\_P31, [290](#)
- TWO\_P32, [290](#)
- TWO\_P4, [290](#)
- TWO\_P56, [290](#)
- TWO\_P57, [291](#)
- TWO\_PI, [291](#)
- correlation\_length\_ms
  - Gnss\_Synchro, [631](#)
- Cpu\_Multicorrelator, [381](#)
- Cpu\_Multicorrelator\_16sc, [381](#)
- Cpu\_Multicorrelator\_Real\_Codes, [382](#)
- cpu\_multicorrelator.h, [1012](#)
- cpu\_multicorrelator\_16sc.h, [1012](#)
- cpu\_multicorrelator\_real\_codes.h, [1013](#)
- crc\_ok
  - cnav\_v27\_part\_t, [370](#)
- createProtobuffer
  - Serdes\_Gnss\_Synchro, [925](#)
  - Serdes\_Monitor\_Pvt, [928](#)
- cshort\_to\_float\_x2, [383](#)
- cshort\_to\_float\_x2.h, [1013](#)
- CubatureFilter, [383](#)
- cuda\_multicorrelator, [384](#)
- cuda\_multicorrelator.h, [1014](#)
- custom\_udp\_signal\_source.h, [1015](#)
- CustomUDPSignalSource, [384](#)
  - implementation, [385](#)
- d1\_subframe\_decoder
  - Beidou\_Dnav\_Navigation\_Message, [335](#)
- d2\_subframe\_decoder
  - Beidou\_Dnav\_Navigation\_Message, [335](#)
- D2R
  - core\_system\_parameters, [246](#)
- d\_A0
  - Gps\_CNAV\_Utc\_Model, [676](#)
  - Gps\_Utc\_Model, [708](#)
- d\_A0\_GAL
  - Beidou\_Dnav\_Utc\_Model, [338](#)
- d\_A0\_GLO
  - Beidou\_Dnav\_Utc\_Model, [338](#)
- d\_A0\_GPS
  - Beidou\_Dnav\_Utc\_Model, [338](#)
- d\_A0\_UTC
  - Beidou\_Dnav\_Utc\_Model, [338](#)
- d\_A1
  - Gps\_CNAV\_Utc\_Model, [676](#)
  - Gps\_Utc\_Model, [708](#)
- d\_A1\_GAL
  - Beidou\_Dnav\_Utc\_Model, [338](#)
- d\_A1\_GLO
  - Beidou\_Dnav\_Utc\_Model, [339](#)
- d\_A1\_GPS
  - Beidou\_Dnav\_Utc\_Model, [339](#)
- d\_A1\_UTC
  - Beidou\_Dnav\_Utc\_Model, [339](#)
- d\_A2
  - Gps\_CNAV\_Utc\_Model, [676](#)
  - Gps\_Utc\_Model, [708](#)
- d\_A\_DOT
  - Gps\_CNAV\_Ephemeris, [659](#)
- d\_A\_f0
  - Beidou\_Dnav\_Almanac, [312](#)
  - Beidou\_Dnav\_Ephemeris, [320](#)
  - Galileo\_Almanac, [422](#)
  - Gps\_Almanac, [651](#)
  - Gps\_CNAV\_Ephemeris, [660](#)
  - Gps\_Ephemeris, [684](#)
- d\_A\_f1
  - Beidou\_Dnav\_Almanac, [312](#)
  - Beidou\_Dnav\_Ephemeris, [320](#)
  - Galileo\_Almanac, [423](#)
  - Gps\_Almanac, [651](#)
  - Gps\_CNAV\_Ephemeris, [660](#)
  - Gps\_Ephemeris, [684](#)
- d\_A\_f2
  - Beidou\_Dnav\_Ephemeris, [321](#)
  - Gps\_CNAV\_Ephemeris, [660](#)
  - Gps\_Ephemeris, [684](#)
- d\_AODC
  - Beidou\_Dnav\_Ephemeris, [321](#)
- d\_AODE
  - Beidou\_Dnav\_Ephemeris, [321](#)
- d\_AXn
  - Glionass\_Gnav\_Ephemeris, [563](#)
- d\_AYn
  - Glionass\_Gnav\_Ephemeris, [563](#)
- d\_AZn
  - Glionass\_Gnav\_Ephemeris, [564](#)
- d\_B1
  - Glionass\_Gnav\_Utc\_Model, [578](#)
- d\_B2
  - Glionass\_Gnav\_Utc\_Model, [578](#)
- d\_B\_n
  - Glionass\_Gnav\_Ephemeris, [564](#)
- d\_C\_n
  - Glionass\_Gnav\_Almanac, [553](#)
- d\_Cic
  - Beidou\_Dnav\_Ephemeris, [321](#)
  - Gps\_CNAV\_Ephemeris, [660](#)
  - Gps\_Ephemeris, [684](#)
- d\_Cis

- Beidou\_Dnav\_Ephemeris, [322](#)
- Gps\_CNAV\_Ephemeris, [661](#)
- Gps\_Ephemeris, [685](#)
- d\_Crc
  - Beidou\_Dnav\_Ephemeris, [322](#)
  - Gps\_CNAV\_Ephemeris, [661](#)
  - Gps\_Ephemeris, [685](#)
- d\_Crs
  - Beidou\_Dnav\_Ephemeris, [322](#)
  - Gps\_CNAV\_Ephemeris, [661](#)
  - Gps\_Ephemeris, [685](#)
- d\_Cuc
  - Beidou\_Dnav\_Ephemeris, [322](#)
  - Gps\_CNAV\_Ephemeris, [661](#)
  - Gps\_Ephemeris, [685](#)
- d\_Cus
  - Beidou\_Dnav\_Ephemeris, [323](#)
  - Gps\_CNAV\_Ephemeris, [662](#)
  - Gps\_Ephemeris, [686](#)
- d\_DELTA\_DOT\_N
  - Gps\_CNAV\_Ephemeris, [662](#)
- d\_DELTA\_OMEGA\_DOT
  - Gps\_CNAV\_Ephemeris, [662](#)
- d\_DELTA\_A
  - Gps\_CNAV\_Ephemeris, [662](#)
- d\_Delta\_T\_n\_A\_dot
  - Glionass\_Gnav\_Almanac, [554](#)
- d\_Delta\_T\_n\_A
  - Glionass\_Gnav\_Almanac, [554](#)
- d\_Delta\_i
  - Galileo\_Almanac, [423](#)
  - Gps\_Almanac, [651](#)
- d\_Delta\_i\_n\_A
  - Glionass\_Gnav\_Almanac, [553](#)
- d\_Delta\_n
  - Beidou\_Dnav\_Ephemeris, [323](#)
  - Gps\_CNAV\_Ephemeris, [662](#)
  - Gps\_Ephemeris, [686](#)
- d\_Delta\_sqrt\_A
  - Galileo\_Almanac, [423](#)
- d\_Delta\_tau\_n
  - Glionass\_Gnav\_Ephemeris, [564](#)
- d\_DeltaT\_LSF
  - Beidou\_Dnav\_Utc\_Model, [339](#)
  - Gps\_CNAV\_Utc\_Model, [676](#)
  - Gps\_Utc\_Model, [709](#)
- d\_DeltaT\_LS
  - Gps\_CNAV\_Utc\_Model, [676](#)
  - Gps\_Utc\_Model, [708](#)
- d\_Doppler0
  - Gps\_Acq\_Assist, [648](#)
- d\_Doppler1
  - Gps\_Acq\_Assist, [648](#)
- d\_E\_n
  - Glionass\_Gnav\_Ephemeris, [565](#)
- d\_F\_T
  - Glionass\_Gnav\_Ephemeris, [565](#)
- d\_H\_n\_A
  - Glionass\_Gnav\_Almanac, [554](#)
- d\_IDOT
  - Beidou\_Dnav\_Ephemeris, [324](#)
  - Gps\_CNAV\_Ephemeris, [663](#)
  - Gps\_Ephemeris, [687](#)
- d\_IODE\_SF2
  - Gps\_Ephemeris, [687](#)
- d\_IODE\_SF3
  - Gps\_Ephemeris, [688](#)
- d\_IODC
  - Gps\_Ephemeris, [687](#)
- d\_KP
  - Glionass\_Gnav\_Almanac, [555](#)
- d\_M\_0
  - Beidou\_Dnav\_Almanac, [312](#)
  - Beidou\_Dnav\_Ephemeris, [324](#)
  - Galileo\_Almanac, [423](#)
  - Gps\_Almanac, [652](#)
  - Gps\_CNAV\_Ephemeris, [664](#)
  - Gps\_Ephemeris, [688](#)
- d\_M\_n\_A
  - Glionass\_Gnav\_Almanac, [555](#)
- d\_N\_4
  - Glionass\_Gnav\_Utc\_Model, [578](#)
- d\_N\_A
  - Glionass\_Gnav\_Utc\_Model, [578](#)
- d\_N\_T
  - Glionass\_Gnav\_Ephemeris, [567](#)
- d\_OMEGA0
  - Beidou\_Dnav\_Almanac, [313](#)
  - Beidou\_Dnav\_Ephemeris, [325](#)
  - Galileo\_Almanac, [424](#)
  - Gps\_Almanac, [652](#)
  - Gps\_CNAV\_Ephemeris, [664](#)
  - Gps\_Ephemeris, [688](#)
- d\_OMEGA\_DOT
  - Beidou\_Dnav\_Almanac, [313](#)
  - Beidou\_Dnav\_Ephemeris, [325](#)
  - Galileo\_Almanac, [424](#)
  - Gps\_Almanac, [652](#)
  - Gps\_Ephemeris, [689](#)
- d\_OMEGA
  - Beidou\_Dnav\_Almanac, [312](#)
  - Beidou\_Dnav\_Ephemeris, [324](#)
  - Galileo\_Almanac, [424](#)
  - Gps\_Almanac, [652](#)
  - Gps\_CNAV\_Ephemeris, [664](#)
  - Gps\_Ephemeris, [688](#)
- d\_P\_1
  - Glionass\_Gnav\_Ephemeris, [567](#)
- d\_P\_2
  - Glionass\_Gnav\_Ephemeris, [568](#)
- d\_P\_3
  - Glionass\_Gnav\_Ephemeris, [568](#)
- d\_P\_4
  - Glionass\_Gnav\_Ephemeris, [568](#)
- d\_TGD1
  - Beidou\_Dnav\_Ephemeris, [327](#)

- d\_TGD2
  - Beidou\_Dnav\_Ephemeris, [327](#)
- d\_TGD
  - Gps\_CNAV\_Ephemeris, [666](#)
  - Gps\_Ephemeris, [690](#)
- d\_TOW
  - Beidou\_Dnav\_Ephemeris, [328](#)
  - Glionass\_Gnav\_Ephemeris, [570](#)
  - Gps\_Acq\_Assist, [649](#)
  - Gps\_CNAV\_Ephemeris, [667](#)
  - Gps\_Ephemeris, [691](#)
- d\_Toa
  - Beidou\_Dnav\_Almanac, [313](#)
- d\_Toc
  - Beidou\_Dnav\_Ephemeris, [327](#)
  - Gps\_CNAV\_Ephemeris, [666](#)
  - Gps\_Ephemeris, [691](#)
- d\_Toe
  - Beidou\_Dnav\_Ephemeris, [327](#)
  - Gps\_Ephemeris, [691](#)
- d\_Toe1
  - Gps\_CNAV\_Ephemeris, [666](#)
- d\_Toe2
  - Gps\_CNAV\_Ephemeris, [666](#)
- d\_Top
  - Gps\_CNAV\_Ephemeris, [667](#)
- d\_URA0
  - Gps\_CNAV\_Ephemeris, [667](#)
- d\_URA1
  - Gps\_CNAV\_Ephemeris, [667](#)
- d\_URA2
  - Gps\_CNAV\_Ephemeris, [667](#)
- d\_VXn
  - Glionass\_Gnav\_Ephemeris, [570](#)
- d\_VYn
  - Glionass\_Gnav\_Ephemeris, [570](#)
- d\_VZn
  - Glionass\_Gnav\_Ephemeris, [570](#)
- d\_WN
  - Glionass\_Gnav\_Ephemeris, [570](#)
- d\_Xn
  - Glionass\_Gnav\_Ephemeris, [571](#)
- d\_Yn
  - Glionass\_Gnav\_Ephemeris, [571](#)
- d\_Zn
  - Glionass\_Gnav\_Ephemeris, [571](#)
- d\_acc
  - Sbas\_Ephemeris, [916](#)
- d\_af0
  - Sbas\_Ephemeris, [916](#)
- d\_af1
  - Sbas\_Ephemeris, [916](#)
- d\_alpha0
  - Beidou\_Dnav\_Iono, [331](#)
  - Gps\_CNAV\_Iono, [670](#)
  - Gps\_Iono, [694](#)
- d\_alpha1
  - Beidou\_Dnav\_Iono, [331](#)
- Gps\_CNAV\_Iono, [670](#)
- Gps\_Iono, [694](#)
- d\_alpha2
  - Beidou\_Dnav\_Iono, [331](#)
  - Gps\_CNAV\_Iono, [670](#)
  - Gps\_Iono, [695](#)
- d\_alpha3
  - Beidou\_Dnav\_Iono, [332](#)
  - Gps\_CNAV\_Iono, [671](#)
  - Gps\_Iono, [695](#)
- d\_beta0
  - Beidou\_Dnav\_Iono, [332](#)
  - Gps\_CNAV\_Iono, [671](#)
  - Gps\_Iono, [695](#)
- d\_beta1
  - Beidou\_Dnav\_Iono, [332](#)
  - Gps\_CNAV\_Iono, [671](#)
  - Gps\_Iono, [695](#)
- d\_beta2
  - Beidou\_Dnav\_Iono, [332](#)
  - Gps\_CNAV\_Iono, [671](#)
  - Gps\_Iono, [696](#)
- d\_beta3
  - Beidou\_Dnav\_Iono, [333](#)
  - Gps\_CNAV\_Iono, [672](#)
  - Gps\_Iono, [696](#)
- d\_dtr
  - Beidou\_Dnav\_Ephemeris, [323](#)
  - Glionass\_Gnav\_Ephemeris, [564](#)
  - Gps\_CNAV\_Ephemeris, [663](#)
  - Gps\_Ephemeris, [686](#)
- d\_e\_eccentricity
  - Beidou\_Dnav\_Almanac, [312](#)
  - Galileo\_Almanac, [423](#)
  - Gps\_Almanac, [651](#)
  - Gps\_CNAV\_Ephemeris, [663](#)
  - Gps\_Ephemeris, [686](#)
- d\_eccentricity
  - Beidou\_Dnav\_Ephemeris, [323](#)
- d\_epsilon\_n\_A
  - Glionass\_Gnav\_Almanac, [554](#)
- d\_gamma\_n
  - Glionass\_Gnav\_Ephemeris, [565](#)
- d\_i\_0
  - Beidou\_Dnav\_Ephemeris, [324](#)
  - Gps\_CNAV\_Ephemeris, [663](#)
  - Gps\_Ephemeris, [687](#)
- d\_iode
  - Glionass\_Gnav\_Ephemeris, [565](#)
- d\_l3rd\_n
  - Glionass\_Gnav\_Ephemeris, [566](#)
- d\_l5th\_n
  - Glionass\_Gnav\_Ephemeris, [566](#)
- d\_l\_n
  - Glionass\_Gnav\_Almanac, [555](#)
- d\_lambda\_n\_A
  - Glionass\_Gnav\_Almanac, [555](#)
- d\_M

- Glonass\_Gnav\_Ephemeris, [566](#)
- d\_m
  - Glonass\_Gnav\_Ephemeris, [566](#)
- d\_n
  - Glonass\_Gnav\_Ephemeris, [567](#)
- d\_n\_A
  - Glonass\_Gnav\_Almanac, [556](#)
- d\_omega\_n\_A
  - Glonass\_Gnav\_Almanac, [556](#)
- d\_P
  - Glonass\_Gnav\_Ephemeris, [567](#)
- d\_pos
  - Sbas\_Ephemeris, [916](#)
- d\_satClkDrift
  - Beidou\_Dnav\_Ephemeris, [325](#)
  - Glonass\_Gnav\_Ephemeris, [568](#)
  - Gps\_CNAV\_Ephemeris, [664](#)
  - Gps\_Ephemeris, [689](#)
- d\_satpos\_X
  - Beidou\_Dnav\_Ephemeris, [325](#)
  - Galileo\_Ephemeris, [438](#)
  - Gps\_CNAV\_Ephemeris, [665](#)
  - Gps\_Ephemeris, [689](#)
- d\_satpos\_Y
  - Beidou\_Dnav\_Ephemeris, [326](#)
  - Galileo\_Ephemeris, [439](#)
  - Gps\_CNAV\_Ephemeris, [665](#)
  - Gps\_Ephemeris, [689](#)
- d\_satpos\_Z
  - Beidou\_Dnav\_Ephemeris, [326](#)
  - Galileo\_Ephemeris, [439](#)
  - Gps\_CNAV\_Ephemeris, [665](#)
  - Gps\_Ephemeris, [689](#)
- d\_satvel\_X
  - Beidou\_Dnav\_Ephemeris, [326](#)
  - Galileo\_Ephemeris, [439](#)
  - Gps\_CNAV\_Ephemeris, [665](#)
  - Gps\_Ephemeris, [690](#)
- d\_satvel\_Y
  - Beidou\_Dnav\_Ephemeris, [326](#)
  - Galileo\_Ephemeris, [439](#)
  - Gps\_CNAV\_Ephemeris, [665](#)
  - Gps\_Ephemeris, [690](#)
- d\_satvel\_Z
  - Beidou\_Dnav\_Ephemeris, [326](#)
  - Galileo\_Ephemeris, [439](#)
  - Gps\_CNAV\_Ephemeris, [666](#)
  - Gps\_Ephemeris, [690](#)
- d\_sqrt\_A
  - Beidou\_Dnav\_Almanac, [313](#)
  - Beidou\_Dnav\_Ephemeris, [327](#)
  - Gps\_Almanac, [652](#)
  - Gps\_Ephemeris, [690](#)
- d\_t\_OT
  - Gps\_CNAV\_Utc\_Model, [676](#)
  - Gps\_Utc\_Model, [709](#)
- d\_t\_b
  - Glonass\_Gnav\_Ephemeris, [569](#)
- d\_t\_k
  - Glonass\_Gnav\_Ephemeris, [569](#)
- d\_t\_lambda\_n\_A
  - Glonass\_Gnav\_Almanac, [556](#)
- d\_tau\_c
  - Glonass\_Gnav\_Ephemeris, [569](#)
  - Glonass\_Gnav\_Utc\_Model, [579](#)
- d\_tau\_gps
  - Glonass\_Gnav\_Utc\_Model, [579](#)
- d\_tau\_n
  - Glonass\_Gnav\_Ephemeris, [569](#)
- d\_tau\_n\_A
  - Glonass\_Gnav\_Almanac, [556](#)
- d\_tod
  - Glonass\_Gnav\_Ephemeris, [569](#)
- d\_tof
  - Sbas\_Ephemeris, [917](#)
- d\_vel
  - Sbas\_Ephemeris, [917](#)
- d\_yr
  - Glonass\_Gnav\_Ephemeris, [571](#)
- DECLARE\_bool
  - gnss\_sdr\_flags, [114](#)
- DECLARE\_double
  - gnss\_sdr\_flags, [114](#)
- DECLARE\_int32
  - gnss\_sdr\_flags, [114](#), [115](#)
- DECLARE\_string
  - gnss\_sdr\_flags, [115](#), [116](#)
- DFRQ1\_GLO
  - core\_system\_parameters, [246](#)
- DFRQ2\_GLO
  - core\_system\_parameters, [246](#)
- DTTOL
  - algorithms\_libs\_rtklib, [131](#)
- Data Type Adapters, [82](#)
- data\_type\_adapters, [83](#)
- data\_type\_gr\_blocks, [84](#)
- dec
  - cnav\_v27\_part\_t, [370](#)
- decisions
  - cnav\_v27\_part\_t, [370](#)
- decode\_block
  - Viterbi\_Decoder, [969](#)
- decoded
  - cnav\_v27\_part\_t, [370](#)
- delta\_n\_3
  - Galileo\_Ephemeris, [440](#)
- dgps\_t, [385](#)
- direct\_resampler\_conditioner.h, [1015](#)
- direct\_resampler\_conditioner\_cb, [386](#)
- direct\_resampler\_conditioner\_cb.h, [1016](#)
- direct\_resampler\_conditioner\_cc, [387](#)
- direct\_resampler\_conditioner\_cc.h, [1017](#)
- direct\_resampler\_conditioner\_cs, [387](#)
- direct\_resampler\_conditioner\_cs.h, [1017](#)
- DirectResamplerConditioner, [388](#)
- implementation, [389](#)

- disable\_secondary\_codes
  - Fpga\_Multicorrelator\_8sc, [413](#)
- disconnect
  - GNSSFlowgraph, [641](#)
  - GalileoE1DIIPIIVemlTrackingFpga, [467](#)
  - GalileoE1PcpsAmbiguousAcquisitionFpga, [480](#)
  - GalileoE5aDIIPIITrackingFpga, [502](#)
  - GalileoE5aPcpsAcquisitionFpga, [515](#)
  - GalileoE5bDIIPIITracking, [522](#)
  - GalileoE5bPcpsAcquisition, [526](#)
  - GalileoE5bPcpsAcquisitionFpga, [533](#)
  - GalileoE5bTelemetryDecoder, [539](#)
  - GalileoE6DIIPIITracking, [541](#)
  - GalileoE6TelemetryDecoder, [548](#)
  - GpsL1CaDIIPIITrackingFpga, [714](#)
  - GpsL1CaPcpsAcquisitionFpga, [730](#)
  - GpsL5DIIPIITrackingFpga, [771](#)
  - GpsL5iPcpsAcquisitionFpga, [780](#)
- display.h, [1018](#)
- Dll\_Pll\_Conf, [389](#)
- Dll\_Pll\_Conf\_Fpga, [390](#)
- dll\_nc\_e\_minus\_l\_normalized
  - tracking\_libs, [191](#)
- dll\_nc\_vemlp\_normalized
  - tracking\_libs, [191](#)
- dll\_pll\_conf.h, [1019](#)
- dll\_pll\_conf\_fpga.h, [1019](#)
- dll\_pll\_veml\_tracking, [392](#)
- dll\_pll\_veml\_tracking.h, [1020](#)
- dll\_pll\_veml\_tracking\_fpga, [393](#)
  - ~dll\_pll\_veml\_tracking\_fpga, [393](#)
  - general\_work, [394](#)
  - reset, [394](#)
  - set\_channel, [394](#)
  - set\_gnss\_synchro, [394](#)
  - start\_tracking, [394](#)
  - stop\_tracking, [395](#)
- dll\_pll\_veml\_tracking\_fpga.h, [1021](#)
- dopplerUncertainty
  - Gps\_Acq\_Assist, [649](#)
- E1B\_DVS\_5
  - Galileo\_Ephemeris, [440](#)
- E1B\_HS\_5
  - Galileo\_Ephemeris, [440](#)
- E5a\_DVS
  - Galileo\_Ephemeris, [440](#)
- E5a\_HS
  - Galileo\_Ephemeris, [441](#)
- E5b\_DVS\_5
  - Galileo\_Ephemeris, [441](#)
- E5b\_HS\_5
  - Galileo\_Ephemeris, [441](#)
- e\_1
  - Galileo\_Ephemeris, [441](#)
- EFACT\_BDS
  - algorithms\_libs\_rtklib, [131](#)
- EFACT\_GAL
  - algorithms\_libs\_rtklib, [131](#)
- EFACT\_GLO
  - algorithms\_libs\_rtklib, [131](#)
- EFACT\_GPS
  - algorithms\_libs\_rtklib, [132](#)
- EFACT\_IRN
  - algorithms\_libs\_rtklib, [132](#)
- EFACT\_QZS
  - algorithms\_libs\_rtklib, [132](#)
- EFACT\_SBS
  - algorithms\_libs\_rtklib, [132](#)
- EPHOPT\_BRDC
  - algorithms\_libs\_rtklib, [132](#)
- EPHOPT\_LEX
  - algorithms\_libs\_rtklib, [133](#)
- EPHOPT\_PREC
  - algorithms\_libs\_rtklib, [133](#)
- EPHOPT\_SBAS
  - algorithms\_libs\_rtklib, [133](#)
- EPHOPT\_SSRAPC
  - algorithms\_libs\_rtklib, [133](#)
- EPHOPT\_SSRCOM
  - algorithms\_libs\_rtklib, [133](#)
- ERR\_BRDCI
  - algorithms\_libs\_rtklib, [134](#)
- ERR\_CBIAS
  - algorithms\_libs\_rtklib, [134](#)
- ERR\_ION
  - rtklib\_pntpos.h, [1212](#)
- ERR\_SAAS
  - algorithms\_libs\_rtklib, [134](#)
- ERR\_TROP
  - rtklib\_pntpos.h, [1212](#)
- edc.h, [1022](#)
- Elevation
  - Gps\_Acq\_Assist, [649](#)
- enable\_secondary\_codes
  - Fpga\_Multicorrelator\_8sc, [414](#)
- eph\_t, [395](#)
- eph\_to\_rtklib
  - algorithms\_libs\_rtklib, [129](#)
- erp\_t, [396](#)
- erpd\_t, [396](#)
- estimate\_doppler\_from\_eph
  - FrontEndCal, [419](#)
- Exponential\_Smoothing, [397](#)
  - ~Exponential\_Smoothing, [397](#)
  - Exponential\_Smoothing, [397](#), [398](#)
  - operator=, [398](#)
  - set\_alpha, [398](#)
  - set\_samples\_for\_initialization, [398](#)
- exponential\_smoothing.h, [1023](#)
- exterr\_t, [399](#)
- FE\_WGS84
  - algorithms\_libs\_rtklib, [134](#)
- FREQ1
  - core\_system\_parameters, [247](#)
- FREQ1\_BDS
  - core\_system\_parameters, [247](#)

- FREQ1\_GLO
  - core\_system\_parameters, 247
- FREQ2
  - core\_system\_parameters, 247
- FREQ2\_BDS
  - core\_system\_parameters, 247
- FREQ2\_GLO
  - core\_system\_parameters, 248
- FREQ3\_BDS
  - core\_system\_parameters, 248
- FREQ3\_GLO
  - core\_system\_parameters, 248
- FREQ5
  - core\_system\_parameters, 248
- FREQ6
  - core\_system\_parameters, 248
- FREQ7
  - core\_system\_parameters, 249
- FREQ8
  - core\_system\_parameters, 249
- FREQ9
  - core\_system\_parameters, 249
- FTP\_TIMEOUT
  - algorithms\_libs\_rtklib, 134
- fatalfunc\_t
  - algorithms\_libs\_rtklib, 129
- fcdb\_t, 399
- fec.h, 1023
- fft\_base\_kernels.h, 1024
- fft\_internal.h, 1024
- file\_configuration.h, 1025
- file\_signal\_source.h, 1026
- file\_t, 400
- FileConfiguration, 400
- FileSignalSource, 401
  - implementation, 402
- find\_uio\_dev\_file\_name
  - core\_libs, 196
- findUtmZone
  - algorithms\_libs, 97
- fir\_filter.h, 1026
- FirFilter, 402
  - ~FirFilter, 403
  - FirFilter, 403
  - implementation, 404
- Flag\_valid\_acquisition
  - Gnss\_Synchro, 631
- Flag\_valid\_pseudorange
  - Gnss\_Synchro, 631
- Flag\_valid\_symbol\_output
  - Gnss\_Synchro, 631
- Flag\_valid\_word
  - Gnss\_Synchro, 632
- flexiband\_signal\_source.h, 1027
- FlexibandSignalSource, 404
  - implementation, 405
- fil\_four\_quadrant\_atan
  - tracking\_libs, 191
- flowgraph
  - ControlThread, 380
- fmcomms2\_signal\_source.h, 1028
- Fmcomms2SignalSource, 405
  - implementation, 406
- Fpga\_Acquisition, 406
  - ~Fpga\_Acquisition, 407
  - close\_device, 408
  - configure\_acquisition, 408
  - Fpga\_Acquisition, 407
  - open\_device, 408
  - read\_acquisition\_results, 408
  - read\_fpga\_total\_scale\_factor, 408
  - reset\_acquisition, 408
  - run\_acquisition, 409
  - set\_block\_exp, 409
  - set\_doppler\_max, 409
  - set\_doppler\_step, 409
  - set\_doppler\_sweep, 410
  - set\_local\_code, 410
  - write\_local\_code, 410
- Fpga\_Multicorrelator\_8sc, 412
  - ~Fpga\_Multicorrelator\_8sc, 413
  - Carrier\_wipeoff\_multicorrelator\_resampler, 413
  - disable\_secondary\_codes, 413
  - enable\_secondary\_codes, 414
  - Fpga\_Multicorrelator\_8sc, 413
  - free, 414
  - initialize\_secondary\_code, 414
  - lock\_channel, 414
  - open\_channel, 414
  - read\_sample\_counter, 414
  - set\_initial\_sample, 415
  - set\_local\_code\_and\_taps, 415
  - set\_output\_vectors, 415
  - set\_secondary\_code\_lengths, 415
  - unlock\_channel, 415
  - update\_local\_code, 416
  - update\_prn\_code\_length, 416
- Fpga\_Switch, 416
  - ~Fpga\_Switch, 417
  - Fpga\_Switch, 417
  - set\_switch\_position, 417
- fpga\_acquisition.h, 1028
- Fpga\_dynamic\_bit\_selection, 410
  - ~Fpga\_dynamic\_bit\_selection, 411
  - bit\_selection, 411
  - Fpga\_dynamic\_bit\_selection, 411
- fpga\_dynamic\_bit\_selection.h, 1029
- fpga\_multicorrelator.h, 1029
- fpga\_switch.h, 1030
- free
  - Fpga\_Multicorrelator\_8sc, 414
- freq\_xlating\_fir\_filter.h, 1031
- FreqXlatingFirFilter, 418
  - implementation, 418
- front
  - algorithms\_libs, 98

- front\_end\_cal.h, [1031](#)
- FrontEndCal, [419](#)
  - estimate\_doppler\_from\_eph, [419](#)
  - GPS\_L1\_front\_end\_model\_E4000, [420](#)
  - get\_ephemeris, [420](#)
  - set\_configuration, [420](#)
- fs
  - Gnss\_Synchro, [632](#)
- ftp\_t, [421](#)
- GALILEO\_CNAV\_PAGE\_MS
  - core\_system\_parameters, [249](#)
- GALILEO\_CNAV\_SYMBOLS\_PER\_PAGE
  - core\_system\_parameters, [249](#)
- GALILEO\_E1\_B\_CODE\_LENGTH\_CHIPS
  - core\_system\_parameters, [250](#)
- GALILEO\_E1\_B\_SAMPLES\_PER\_SYMBOL
  - core\_system\_parameters, [250](#)
- GALILEO\_E1\_B\_SYMBOL\_RATE\_BPS
  - core\_system\_parameters, [250](#)
- GALILEO\_E1\_C\_SECONDARY\_CODE\_LENGTH
  - core\_system\_parameters, [250](#)
- GALILEO\_E1\_CODE\_CHIP\_RATE\_CPS
  - core\_system\_parameters, [250](#)
- GALILEO\_E1\_CODE\_PERIOD\_MS
  - core\_system\_parameters, [251](#)
- GALILEO\_E1\_CODE\_PERIOD\_S
  - core\_system\_parameters, [251](#)
- GALILEO\_E1\_FREQ\_HZ
  - core\_system\_parameters, [251](#)
- GALILEO\_E1\_HISTORY\_DEEP
  - core\_system\_parameters, [251](#)
- GALILEO\_E1\_OPT\_ACQ\_FS\_SPS
  - core\_system\_parameters, [251](#)
- GALILEO\_E1\_SUB\_CARRIER\_A\_RATE\_HZ
  - core\_system\_parameters, [252](#)
- GALILEO\_E1\_SUB\_CARRIER\_B\_RATE\_HZ
  - core\_system\_parameters, [252](#)
- GALILEO\_E5A\_CODE\_CHIP\_RATE\_CPS
  - core\_system\_parameters, [252](#)
- GALILEO\_E5A\_CODE\_LENGTH\_CHIPS
  - core\_system\_parameters, [252](#)
- GALILEO\_E5A\_CODE\_PERIOD\_MS
  - core\_system\_parameters, [252](#)
- GALILEO\_E5A\_CODE\_PERIOD\_S
  - core\_system\_parameters, [253](#)
- GALILEO\_E5A\_FREQ\_HZ
  - core\_system\_parameters, [253](#)
- GALILEO\_E5A\_I\_SECONDARY\_CODE\_LENGTH
  - core\_system\_parameters, [253](#)
- GALILEO\_E5A\_I\_TIERED\_CODE\_PERIOD\_S
  - core\_system\_parameters, [253](#)
- GALILEO\_E5A\_OPT\_ACQ\_FS\_SPS
  - core\_system\_parameters, [253](#)
- GALILEO\_E5A\_Q\_SECONDARY\_CODE\_LENGTH
  - core\_system\_parameters, [254](#)
- GALILEO\_E5A\_Q\_TIERED\_CODE\_PERIOD\_S
  - core\_system\_parameters, [254](#)
- GALILEO\_E5A\_SYMBOL\_RATE\_BPS
  - core\_system\_parameters, [254](#)
- GALILEO\_E5B\_CODE\_CHIP\_RATE\_CPS
  - core\_system\_parameters, [254](#)
- GALILEO\_E5B\_CODE\_LENGTH\_CHIPS
  - core\_system\_parameters, [254](#)
- GALILEO\_E5B\_CODE\_PERIOD\_MS
  - core\_system\_parameters, [255](#)
- GALILEO\_E5B\_CODE\_PERIOD\_S
  - core\_system\_parameters, [255](#)
- GALILEO\_E5B\_FREQ\_HZ
  - core\_system\_parameters, [255](#)
- GALILEO\_E5B\_I\_SECONDARY\_CODE\_LENGTH
  - core\_system\_parameters, [255](#)
- GALILEO\_E5B\_I\_TIERED\_CODE\_PERIOD\_S
  - core\_system\_parameters, [255](#)
- GALILEO\_E5B\_OPT\_ACQ\_FS\_SPS
  - core\_system\_parameters, [256](#)
- GALILEO\_E5B\_Q\_SECONDARY\_CODE\_LENGTH
  - core\_system\_parameters, [256](#)
- GALILEO\_E5B\_Q\_TIERED\_CODE\_PERIOD\_S
  - core\_system\_parameters, [256](#)
- GALILEO\_E5B\_SYMBOL\_RATE\_BPS
  - core\_system\_parameters, [256](#)
- GALILEO\_E6\_B\_CODE\_CHIP\_RATE\_CPS
  - core\_system\_parameters, [256](#)
- GALILEO\_E6\_B\_CODE\_LENGTH\_CHIPS
  - core\_system\_parameters, [257](#)
- GALILEO\_E6\_C\_CODE\_CHIP\_RATE\_CPS
  - core\_system\_parameters, [257](#)
- GALILEO\_E6\_C\_CODE\_LENGTH\_CHIPS
  - core\_system\_parameters, [257](#)
- GALILEO\_E6\_C\_SECONDARY\_CODE\_LENGTH\_C↔HIPS
  - core\_system\_parameters, [257](#)
- GALILEO\_E6\_CODE\_PERIOD\_MS
  - core\_system\_parameters, [257](#)
- GALILEO\_E6\_CODE\_PERIOD\_S
  - core\_system\_parameters, [258](#)
- GALILEO\_E6\_FREQ\_HZ
  - core\_system\_parameters, [258](#)
- GALILEO\_GM
  - core\_system\_parameters, [258](#)
- GALILEO\_INAV\_PAGE\_PART\_SYMBOLS
  - core\_system\_parameters, [258](#)
- GALILEO\_INAV\_PAGE\_PART\_WITH\_PREABLE\_S↔ECONDS
  - core\_system\_parameters, [259](#)
- GALILEO\_INAV\_PAGE\_SYMBOLS
  - core\_system\_parameters, [259](#)
- GALILEO\_F
  - core\_system\_parameters, [258](#)
- GAP\_RESION
  - algorithms\_libs\_rtklib, [135](#)
- GLONASS\_C20
  - core\_system\_parameters, [259](#)
- GLONASS\_EARTH\_INCLINATION
  - core\_system\_parameters, [259](#)
- GLONASS\_EARTH\_RADIUS

- core\_system\_parameters, [259](#)
- GLONASS\_F\_M\_A
  - core\_system\_parameters, [260](#)
- GLONASS\_FLATTENING
  - core\_system\_parameters, [260](#)
- GLONASS\_GNAV\_HAMMING\_CODE\_BITS
  - core\_system\_parameters, [260](#)
- GLONASS\_GNAV\_PREAMBLE
  - core\_system\_parameters, [231](#)
- GLONASS\_GNAV\_STRING\_BITS
  - core\_system\_parameters, [260](#)
- GLONASS\_GNAV\_STRING\_SYMBOLS
  - core\_system\_parameters, [261](#)
- GLONASS\_GNAV\_TELEMETRY\_RATE\_BITS\_SEC↔  
OND
  - core\_system\_parameters, [261](#)
- GLONASS\_GNAV\_TELEMETRY\_RATE\_SYMBOLS↔  
\_SECOND
  - core\_system\_parameters, [261](#)
- GLONASS\_GRAVITY\_CORRECTION
  - core\_system\_parameters, [261](#)
- GLONASS\_GRAVITY
  - core\_system\_parameters, [261](#)
- GLONASS\_GM
  - core\_system\_parameters, [260](#)
- GLONASS\_J2
  - core\_system\_parameters, [262](#)
- GLONASS\_J4
  - core\_system\_parameters, [262](#)
- GLONASS\_J6
  - core\_system\_parameters, [262](#)
- GLONASS\_J8
  - core\_system\_parameters, [262](#)
- GLONASS\_L1\_CA\_CHIP\_PERIOD\_S
  - core\_system\_parameters, [262](#)
- GLONASS\_L1\_CA\_CODE\_LENGTH\_CHIPS
  - core\_system\_parameters, [263](#)
- GLONASS\_L1\_CA\_CODE\_PERIOD\_S
  - core\_system\_parameters, [263](#)
- GLONASS\_L1\_CA\_CODE\_RATE\_CPS
  - core\_system\_parameters, [263](#)
- GLONASS\_L1\_CA\_DFREQ\_HZ
  - core\_system\_parameters, [263](#)
- GLONASS\_L1\_CA\_FREQ\_HZ
  - core\_system\_parameters, [263](#)
- GLONASS\_L1\_L2\_CA.h, [1084](#)
- GLONASS\_L2\_CA\_CHIP\_PERIOD\_S
  - core\_system\_parameters, [264](#)
- GLONASS\_L2\_CA\_CODE\_LENGTH\_CHIPS
  - core\_system\_parameters, [264](#)
- GLONASS\_L2\_CA\_CODE\_PERIOD\_S
  - core\_system\_parameters, [264](#)
- GLONASS\_L2\_CA\_CODE\_RATE\_CPS
  - core\_system\_parameters, [264](#)
- GLONASS\_L2\_CA\_DFREQ\_HZ
  - core\_system\_parameters, [264](#)
- GLONASS\_L2\_CA\_FREQ\_HZ
  - core\_system\_parameters, [265](#)
- GLONASS\_LEAP\_SECONDS
  - core\_system\_parameters, [265](#)
- GLONASS\_MOON\_ECCENTRICITY
  - core\_system\_parameters, [265](#)
- GLONASS\_MOON\_GM
  - core\_system\_parameters, [266](#)
- GLONASS\_MOON\_INCLINATION
  - core\_system\_parameters, [266](#)
- GLONASS\_MOON\_OMEGA\_0
  - core\_system\_parameters, [266](#)
- GLONASS\_MOON\_OMEGA\_1
  - core\_system\_parameters, [266](#)
- GLONASS\_MOON\_Q0
  - core\_system\_parameters, [266](#)
- GLONASS\_MOON\_Q1
  - core\_system\_parameters, [267](#)
- GLONASS\_MOON\_SEMI\_MAJOR\_AXIS
  - core\_system\_parameters, [267](#)
- GLONASS\_OMEGA\_EARTH\_DOT
  - core\_system\_parameters, [267](#)
- GLONASS\_SEMI\_MAJOR\_AXIS
  - core\_system\_parameters, [267](#)
- GLONASS\_SUN\_ECCENTRICITY
  - core\_system\_parameters, [267](#)
- GLONASS\_SUN\_GM
  - core\_system\_parameters, [268](#)
- GLONASS\_SUN\_OMEGA
  - core\_system\_parameters, [268](#)
- GLONASS\_SUN\_Q0
  - core\_system\_parameters, [268](#)
- GLONASS\_SUN\_Q1
  - core\_system\_parameters, [268](#)
- GLONASS\_SUN\_SEMI\_MAJOR\_AXIS
  - core\_system\_parameters, [268](#)
- GLONASS\_TAU\_0
  - core\_system\_parameters, [269](#)
- GLONASS\_TAU\_1
  - core\_system\_parameters, [269](#)
- GLONASS\_U0
  - core\_system\_parameters, [269](#)
- GNSS block interfaces, [194](#)
- GNSS\_OMEGA\_EARTH\_DOT
  - core\_system\_parameters, [269](#)
- GNSS\_PI
  - core\_system\_parameters, [269](#)
- GNSSBlockFactory, [636](#)
  - GetBlock, [637](#)
- GNSSBlockInterface, [638](#)
  - start, [639](#)
- GNSSFlowgraph, [639](#)
  - ~GNSSFlowgraph, [640](#)
  - acquisition\_manager, [641](#)
  - apply\_action, [641](#)
  - connect, [641](#)
  - disconnect, [641](#)
  - GNSSFlowgraph, [640](#)
  - get\_pvt, [642](#)
  - priorize\_satellites, [642](#)

- send\_telemetry\_msg, [642](#)
- set\_configuration, [642](#)
- start, [642](#)
- stop, [642](#)
- wait, [643](#)
- GPS\_Bit\_Number
  - Gps\_Acq\_Assist, [649](#)
- GPS\_CA\_TELEMETRY\_RATE\_BITS\_SECOND
  - core\_system\_parameters, [270](#)
- GPS\_CA\_TELEMETRY\_RATE\_SYMBOLS\_SECOND
  - core\_system\_parameters, [270](#)
- GPS\_CNAV.h, [1115](#)
- GPS\_GM
  - core\_system\_parameters, [270](#)
- GPS\_L1\_CA.h, [1121](#)
- GPS\_L1\_CA\_BIT\_PERIOD\_MS
  - core\_system\_parameters, [270](#)
- GPS\_L1\_CA\_CHIP\_PERIOD\_S
  - core\_system\_parameters, [271](#)
- GPS\_L1\_CA\_CODE\_LENGTH\_CHIPS
  - core\_system\_parameters, [271](#)
- GPS\_L1\_CA\_CODE\_PERIOD\_MS
  - core\_system\_parameters, [271](#)
- GPS\_L1\_CA\_CODE\_PERIOD\_S
  - core\_system\_parameters, [271](#)
- GPS\_L1\_CA\_CODE\_RATE\_CPS
  - core\_system\_parameters, [271](#)
- GPS\_L1\_CA\_OPT\_ACQ\_FS\_SPS
  - core\_system\_parameters, [272](#)
- GPS\_L1\_FREQ\_HZ
  - core\_system\_parameters, [272](#)
- GPS\_L1\_front\_end\_model\_E4000
  - FrontEndCal, [420](#)
- GPS\_L2\_CNAV\_DATA\_PAGE\_BITS
  - core\_system\_parameters, [272](#)
- GPS\_L2\_FREQ\_HZ
  - core\_system\_parameters, [272](#)
- GPS\_L2\_L\_CODE\_LENGTH\_CHIPS
  - core\_system\_parameters, [272](#)
- GPS\_L2\_L\_CODE\_RATE\_CPS
  - core\_system\_parameters, [273](#)
- GPS\_L2\_L\_PERIOD\_S
  - core\_system\_parameters, [273](#)
- GPS\_L2\_M\_CODE\_LENGTH\_CHIPS
  - core\_system\_parameters, [273](#)
- GPS\_L2\_M\_CODE\_RATE\_CPS
  - core\_system\_parameters, [273](#)
- GPS\_L2\_M\_PERIOD\_S
  - core\_system\_parameters, [273](#)
- GPS\_L2\_V27\_HISTORY\_LENGTH\_BITS
  - telemetry\_decoder\_libswiftcnv, [182](#)
- GPS\_L2C.h, [1139](#)
- GPS\_L2C\_M\_INIT\_REG
  - core\_system\_parameters, [274](#)
- GPS\_L2C\_OPT\_ACQ\_FS\_SPS
  - core\_system\_parameters, [274](#)
- GPS\_L2C\_V27\_DECODE\_BITS
  - telemetry\_decoder\_libswiftcnv, [183](#)
- GPS\_L2C\_V27\_DELAY\_BITS
  - telemetry\_decoder\_libswiftcnv, [183](#)
- GPS\_L2C\_V27\_INIT\_BITS
  - telemetry\_decoder\_libswiftcnv, [183](#)
- GPS\_L5.h, [1142](#)
- GPS\_L5\_CNAV\_DATA\_PAGE\_BITS
  - core\_system\_parameters, [274](#)
- GPS\_L5\_FREQ\_HZ
  - core\_system\_parameters, [275](#)
- GPS\_L5\_OPT\_ACQ\_FS\_SPS
  - core\_system\_parameters, [275](#)
- GPS\_L5I\_CODE\_LENGTH\_CHIPS
  - core\_system\_parameters, [275](#)
- GPS\_L5I\_CODE\_RATE\_CPS
  - core\_system\_parameters, [275](#)
- GPS\_L5I\_PERIOD\_MS
  - core\_system\_parameters, [275](#)
- GPS\_L5I\_PERIOD\_S
  - core\_system\_parameters, [276](#)
- GPS\_L5I\_SYMBOL\_PERIOD\_MS
  - core\_system\_parameters, [276](#)
- GPS\_L5I\_SYMBOL\_PERIOD\_S
  - core\_system\_parameters, [276](#)
- GPS\_L5Q\_CODE\_LENGTH\_CHIPS
  - core\_system\_parameters, [276](#)
- GPS\_L5Q\_CODE\_RATE\_CPS
  - core\_system\_parameters, [276](#)
- GPS\_L5Q\_PERIOD\_S
  - core\_system\_parameters, [277](#)
- GPS\_SUBFRAME\_BITS
  - core\_system\_parameters, [277](#)
- GPS\_SUBFRAME\_LENGTH
  - core\_system\_parameters, [277](#)
- GPS\_SUBFRAME\_MS
  - core\_system\_parameters, [277](#)
- GPS\_SUBFRAME\_SECONDS
  - core\_system\_parameters, [277](#)
- GPS\_WORD\_BITS
  - core\_system\_parameters, [278](#)
- GPS\_WORD\_LENGTH
  - core\_system\_parameters, [278](#)
- GPS\_F
  - core\_system\_parameters, [270](#)
- GPU\_Complex, [786](#)
- GPU\_Complex\_Short, [786](#)
- GST\_to\_UTC\_time
  - Galileo\_Utc\_Model, [461](#)
- Galileo\_Almanac, [421](#)
  - d\_A\_f0, [422](#)
  - d\_A\_f1, [423](#)
  - d\_Delta\_i, [423](#)
  - d\_Delta\_sqrt\_A, [423](#)
  - d\_M\_0, [423](#)
  - d\_OMEGA0, [424](#)
  - d\_OMEGA\_DOT, [424](#)
  - d\_OMEGA, [424](#)
  - d\_e\_eccentricity, [423](#)
  - Galileo\_Almanac, [422](#)

- [i\\_satellite\\_PRN](#), [424](#)
- [Galileo\\_Almanac\\_Helper](#), [425](#)
  - [Galileo\\_Almanac\\_Helper](#), [426](#)
- [Galileo\\_CNAV.h](#), [1033](#)
- [Galileo\\_Cnav\\_Message](#), [426](#)
- [Galileo\\_E1.h](#), [1035](#)
- [Galileo\\_E1\\_Tcp\\_Connector\\_Tracking\\_cc](#), [427](#)
- [Galileo\\_E5a.h](#), [1045](#)
- [Galileo\\_E5b.h](#), [1052](#)
- [Galileo\\_E6.h](#), [1056](#)
- [Galileo\\_Ephemeris](#), [432](#)
  - [A\\_1](#), [436](#)
  - [af0\\_4](#), [436](#)
  - [af1\\_4](#), [436](#)
  - [af2\\_4](#), [436](#)
  - [BGD\\_E1E5a\\_5](#), [436](#)
  - [BGD\\_E1E5b\\_5](#), [437](#)
  - [C\\_ic\\_4](#), [437](#)
  - [C\\_is\\_4](#), [437](#)
  - [C\\_rc\\_3](#), [437](#)
  - [C\\_rs\\_3](#), [438](#)
  - [C\\_uc\\_3](#), [438](#)
  - [C\\_us\\_3](#), [438](#)
  - [d\\_satpos\\_X](#), [438](#)
  - [d\\_satpos\\_Y](#), [439](#)
  - [d\\_satpos\\_Z](#), [439](#)
  - [d\\_satvel\\_X](#), [439](#)
  - [d\\_satvel\\_Y](#), [439](#)
  - [d\\_satvel\\_Z](#), [439](#)
  - [delta\\_n\\_3](#), [440](#)
  - [E1B\\_DVS\\_5](#), [440](#)
  - [E1B\\_HS\\_5](#), [440](#)
  - [E5a\\_DVS](#), [440](#)
  - [E5a\\_HS](#), [441](#)
  - [E5b\\_DVS\\_5](#), [441](#)
  - [E5b\\_HS\\_5](#), [441](#)
  - [e\\_1](#), [441](#)
  - [Galileo\\_System\\_Time](#), [434](#)
  - [Galileo\\_dtr](#), [442](#)
  - [i\\_0\\_2](#), [442](#)
  - [i\\_satellite\\_PRN](#), [442](#)
  - [iDot\\_2](#), [442](#)
  - [M0\\_1](#), [443](#)
  - [OMEGA\\_0\\_2](#), [443](#)
  - [OMEGA\\_dot\\_3](#), [443](#)
  - [omega\\_2](#), [443](#)
  - [satellitePosition](#), [435](#)
  - [serialize](#), [435](#)
  - [sv\\_clock\\_drift](#), [435](#)
  - [sv\\_clock\\_relativistic\\_term](#), [435](#)
  - [t0c\\_4](#), [444](#)
  - [t0e\\_1](#), [444](#)
  - [TOW\\_5](#), [444](#)
  - [WN\\_5](#), [444](#)
- [Galileo\\_FNAV.h](#), [1060](#)
- [Galileo\\_Fnav\\_Message](#), [445](#)
- [Galileo\\_HAS\\_data](#), [446](#)
- [Galileo\\_INAV.h](#), [1064](#)
- [Galileo\\_Inav\\_Message](#), [447](#)
- [Galileo\\_Iono](#), [448](#)
  - [ai0\\_5](#), [449](#)
  - [ai1\\_5](#), [450](#)
  - [ai2\\_5](#), [450](#)
  - [Galileo\\_Iono](#), [449](#)
  - [Region1\\_flag\\_5](#), [450](#)
  - [Region2\\_flag\\_5](#), [450](#)
  - [Region3\\_flag\\_5](#), [451](#)
  - [Region4\\_flag\\_5](#), [451](#)
  - [Region5\\_flag\\_5](#), [451](#)
  - [serialize](#), [449](#)
  - [TOW\\_5](#), [451](#)
  - [WN\\_5](#), [452](#)
- [Galileo\\_System\\_Time](#)
  - [Galileo\\_Ephemeris](#), [434](#)
- [Galileo\\_Utc\\_Model](#), [460](#)
  - [GST\\_to\\_UTC\\_time](#), [461](#)
  - [Galileo\\_Utc\\_Model](#), [460](#)
  - [serialize](#), [461](#)
  - [t0t\\_6](#), [461](#)
  - [WNot\\_6](#), [461](#)
- [galileo\\_almanac.h](#), [1032](#)
- [galileo\\_almanac\\_helper.h](#), [1032](#)
- [galileo\\_cnav\\_message.h](#), [1035](#)
- [Galileo\\_dtr](#)
  - [Galileo\\_Ephemeris](#), [442](#)
- [galileo\\_e1\\_code\\_gen\\_complex\\_sampled](#)
  - [algorithms\\_libs](#), [98](#)
- [galileo\\_e1\\_code\\_gen\\_float\\_sampled](#)
  - [algorithms\\_libs](#), [98, 99](#)
- [galileo\\_e1\\_code\\_gen\\_sinboc11\\_float](#)
  - [algorithms\\_libs](#), [99](#)
- [galileo\\_e1\\_dll\\_pll\\_veml\\_tracking.h](#), [1037](#)
- [galileo\\_e1\\_dll\\_pll\\_veml\\_tracking\\_fpga.h](#), [1037](#)
- [galileo\\_e1\\_pcps\\_8ms\\_ambiguous\\_acquisition.h](#), [1038](#)
- [galileo\\_e1\\_pcps\\_ambiguous\\_acquisition.h](#), [1039](#)
- [galileo\\_e1\\_pcps\\_ambiguous\\_acquisition\\_fpga.h](#), [1039](#)
- [galileo\\_e1\\_pcps\\_cccwsr\\_ambiguous\\_acquisition.h](#), [1040](#)
- [galileo\\_e1\\_pcps\\_quicksync\\_ambiguous\\_acquisition.h](#), [1040](#)
- [galileo\\_e1\\_pcps\\_tong\\_ambiguous\\_acquisition.h](#), [1041](#)
- [galileo\\_e1\\_signal\\_replica.h](#), [1042](#)
- [galileo\\_e1\\_tcp\\_connector\\_tracking.h](#), [1042](#)
- [galileo\\_e1\\_tcp\\_connector\\_tracking\\_cc.h](#), [1043](#)
- [galileo\\_e1b\\_telemetry\\_decoder.h](#), [1044](#)
- [galileo\\_e5\\_a\\_code\\_gen\\_complex\\_primary](#)
  - [algorithms\\_libs](#), [99](#)
- [galileo\\_e5\\_a\\_code\\_gen\\_complex\\_sampled](#)
  - [algorithms\\_libs](#), [99](#)
- [galileo\\_e5\\_b\\_code\\_gen\\_complex\\_primary](#)
  - [algorithms\\_libs](#), [100](#)
- [galileo\\_e5\\_b\\_code\\_gen\\_complex\\_sampled](#)
  - [algorithms\\_libs](#), [100](#)
- [galileo\\_e5\\_signal\\_replica.h](#), [1045](#)
- [galileo\\_e5a\\_dll\\_pll\\_tracking.h](#), [1047](#)
- [galileo\\_e5a\\_dll\\_pll\\_tracking\\_fpga.h](#), [1047](#)

- galileo\_e5a\_noncoherent\_iq\_acquisition\_caf.h, 1048
- galileo\_e5a\_noncoherent\_iq\_acquisition\_caf\_cc.h, 1049
- galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc, 428
  - ~galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc, 429
  - general\_work, 429
  - init, 429
  - mag, 429
  - set\_active, 429
  - set\_channel, 430
  - set\_channel\_fsm, 430
  - set\_doppler\_max, 430
  - set\_doppler\_step, 430
  - set\_gnss\_synchro, 431
  - set\_local\_code, 431
  - set\_state, 431
  - set\_threshold, 432
- galileo\_e5a\_pcps\_acquisition.h, 1050
- galileo\_e5a\_pcps\_acquisition\_fpga.h, 1050
- galileo\_e5a\_telemetry\_decoder.h, 1051
- galileo\_e5b\_dll\_pll\_tracking.h, 1053
- galileo\_e5b\_pcps\_acquisition.h, 1054
- galileo\_e5b\_pcps\_acquisition\_fpga.h, 1054
- galileo\_e5b\_telemetry\_decoder.h, 1055
- galileo\_e6\_b\_code\_gen\_complex\_primary
  - algorithms\_libs, 100
- galileo\_e6\_b\_code\_gen\_complex\_sampled
  - algorithms\_libs, 100
- galileo\_e6\_b\_code\_gen\_float\_primary
  - algorithms\_libs, 101
- galileo\_e6\_c\_code\_gen\_complex\_primary
  - algorithms\_libs, 101
- galileo\_e6\_c\_code\_gen\_complex\_sampled
  - algorithms\_libs, 101
- galileo\_e6\_c\_code\_gen\_float\_primary
  - algorithms\_libs, 101
- galileo\_e6\_c\_secondary\_code
  - algorithms\_libs, 101
- galileo\_e6\_c\_secondary\_code\_gen\_complex
  - algorithms\_libs, 102
- galileo\_e6\_c\_secondary\_code\_gen\_float
  - algorithms\_libs, 102
- galileo\_e6\_dll\_pll\_tracking.h, 1057
- galileo\_e6\_pcps\_acquisition.h, 1057
- galileo\_e6\_signal\_replica.h, 1058
- galileo\_e6\_telemetry\_decoder.h, 1059
- galileo\_ephemeris.h, 1059
- galileo\_ephemeris\_map
  - Rtklib\_Solver, 909
- galileo\_fnav\_message.h, 1063
- galileo\_has\_data.h, 1064
- galileo\_inav\_message.h, 1068
- galileo\_iono.h, 1069
- galileo\_pcps\_8ms\_acquisition\_cc, 452
  - ~galileo\_pcps\_8ms\_acquisition\_cc, 453
  - general\_work, 454
  - init, 454
  - mag, 454
  - set\_active, 454
  - set\_channel, 455
  - set\_channel\_fsm, 455
  - set\_doppler\_max, 455
  - set\_doppler\_step, 455
  - set\_gnss\_synchro, 456
  - set\_local\_code, 456
  - set\_state, 456
  - set\_threshold, 458
- galileo\_pcps\_8ms\_acquisition\_cc.h, 1070
- galileo\_telemetry\_decoder\_gs, 458
  - general\_work, 459
  - set\_channel, 459
  - set\_satellite, 459
- galileo\_telemetry\_decoder\_gs.h, 1071
- galileo\_utc\_model.h, 1071
- GalileoE1BTelemetryDecoder, 462
  - implementation, 463
- GalileoE1DIPIIVemlTracking, 463
  - implementation, 464
  - set\_channel, 464
  - set\_gnss\_synchro, 464
  - stop\_tracking, 464
- GalileoE1DIPIIVemlTrackingFpga, 465
  - ~GalileoE1DIPIIVemlTrackingFpga, 466
  - connect, 466
  - disconnect, 467
  - GalileoE1DIPIIVemlTrackingFpga, 466
  - get\_left\_block, 467
  - get\_right\_block, 467
  - implementation, 467
  - item\_size, 467
  - role, 468
  - set\_channel, 468
  - set\_gnss\_synchro, 468
  - start\_tracking, 468
  - stop\_tracking, 469
- GalileoE1Pcps8msAmbiguousAcquisition, 469
  - implementation, 470
  - init, 471
  - mag, 471
  - reset, 471
  - set\_channel, 471
  - set\_channel\_fsm, 471
  - set\_doppler\_max, 472
  - set\_doppler\_step, 472
  - set\_gnss\_synchro, 472
  - set\_local\_code, 472
  - set\_threshold, 473
  - stop\_acquisition, 473
- GalileoE1PcpsAmbiguousAcquisition, 473
  - implementation, 474
  - init, 475
  - mag, 475
  - reset, 475
  - set\_channel, 475
  - set\_channel\_fsm, 475

- set\_doppler\_center, 476
- set\_doppler\_max, 476
- set\_doppler\_step, 476
- set\_gnss\_synchro, 476
- set\_local\_code, 476
- set\_resampler\_latency, 477
- set\_state, 477
- set\_threshold, 477
- stop\_acquisition, 477
- GalileoE1PcpsAmbiguousAcquisitionFpga, 478
  - ~GalileoE1PcpsAmbiguousAcquisitionFpga, 479
  - connect, 479
  - disconnect, 480
  - GalileoE1PcpsAmbiguousAcquisitionFpga, 479
  - get\_left\_block, 480
  - get\_right\_block, 480
  - implementation, 480
  - init, 480
  - item\_size, 481
  - mag, 481
  - reset, 481
  - role, 481
  - set\_channel, 481
  - set\_channel\_fsm, 482
  - set\_doppler\_center, 482
  - set\_doppler\_max, 482
  - set\_doppler\_step, 482
  - set\_gnss\_synchro, 482
  - set\_local\_code, 483
  - set\_resampler\_latency, 483
  - set\_state, 483
  - set\_threshold, 483
  - stop\_acquisition, 483
- GalileoE1PcpsCccwsrAmbiguousAcquisition, 484
  - implementation, 485
  - init, 485
  - mag, 485
  - reset, 486
  - set\_channel, 486
  - set\_channel\_fsm, 486
  - set\_doppler\_max, 486
  - set\_doppler\_step, 487
  - set\_gnss\_synchro, 487
  - set\_state, 487
  - set\_threshold, 487
  - stop\_acquisition, 487
- GalileoE1PcpsQuickSyncAmbiguousAcquisition, 488
  - implementation, 489
  - init, 489
  - mag, 489
  - reset, 490
  - set\_channel, 490
  - set\_channel\_fsm, 490
  - set\_doppler\_max, 490
  - set\_doppler\_step, 491
  - set\_gnss\_synchro, 491
  - set\_local\_code, 491
  - set\_state, 491
  - set\_threshold, 491
  - stop\_acquisition, 492
- GalileoE1PcpsTongAmbiguousAcquisition, 492
  - implementation, 493
  - init, 494
  - mag, 494
  - reset, 494
  - set\_channel, 494
  - set\_channel\_fsm, 494
  - set\_doppler\_max, 495
  - set\_doppler\_step, 495
  - set\_gnss\_synchro, 495
  - set\_local\_code, 495
  - set\_state, 496
  - set\_threshold, 496
  - stop\_acquisition, 496
- GalileoE1TcpConnectorTracking, 497
  - implementation, 497
  - set\_channel, 498
  - set\_gnss\_synchro, 498
  - stop\_tracking, 498
- GalileoE5aDIIPIITracking, 499
  - implementation, 499
  - set\_channel, 500
  - set\_gnss\_synchro, 500
  - stop\_tracking, 500
- GalileoE5aDIIPIITrackingFpga, 501
  - ~GalileoE5aDIIPIITrackingFpga, 502
  - connect, 502
  - disconnect, 502
  - GalileoE5aDIIPIITrackingFpga, 502
  - get\_left\_block, 502
  - get\_right\_block, 503
  - implementation, 503
  - item\_size, 503
  - role, 503
  - set\_channel, 503
  - set\_gnss\_synchro, 504
  - start\_tracking, 504
  - stop\_tracking, 504
- GalileoE5aNoncoherentIQAcquisitionCaf, 505
  - implementation, 506
  - init, 506
  - mag, 506
  - reset, 506
  - set\_channel, 506
  - set\_channel\_fsm, 507
  - set\_doppler\_max, 507
  - set\_doppler\_step, 507
  - set\_gnss\_synchro, 507
  - set\_local\_code, 508
  - set\_state, 508
  - set\_threshold, 508
  - stop\_acquisition, 508
- GalileoE5aPcpsAcquisition, 509
  - init, 510
  - mag, 510
  - reset, 510

- set\_channel, [511](#)
- set\_channel\_fsm, [511](#)
- set\_doppler\_center, [511](#)
- set\_doppler\_max, [511](#)
- set\_doppler\_step, [511](#)
- set\_gnss\_synchro, [512](#)
- set\_local\_code, [512](#)
- set\_resampler\_latency, [512](#)
- set\_state, [512](#)
- set\_threshold, [513](#)
- stop\_acquisition, [513](#)
- GalileoE5aPcpsAcquisitionFpga, [513](#)
  - ~GalileoE5aPcpsAcquisitionFpga, [515](#)
  - connect, [515](#)
  - disconnect, [515](#)
  - GalileoE5aPcpsAcquisitionFpga, [515](#)
  - get\_left\_block, [516](#)
  - get\_right\_block, [516](#)
  - implementation, [516](#)
  - init, [516](#)
  - item\_size, [516](#)
  - mag, [517](#)
  - reset, [517](#)
  - role, [517](#)
  - set\_channel, [517](#)
  - set\_channel\_fsm, [517](#)
  - set\_doppler\_center, [518](#)
  - set\_doppler\_max, [518](#)
  - set\_doppler\_step, [518](#)
  - set\_gnss\_synchro, [518](#)
  - set\_local\_code, [518](#)
  - set\_resampler\_latency, [519](#)
  - set\_single\_doppler\_flag, [519](#)
  - set\_state, [519](#)
  - set\_threshold, [519](#)
  - stop\_acquisition, [520](#)
- GalileoE5aTelemetryDecoder, [520](#)
  - implementation, [521](#)
- GalileoE5bDIIPIITracking, [521](#)
  - connect, [522](#)
  - disconnect, [522](#)
  - get\_left\_block, [523](#)
  - get\_right\_block, [523](#)
  - implementation, [523](#)
  - set\_channel, [523](#)
  - set\_gnss\_synchro, [523](#)
  - stop\_tracking, [524](#)
- GalileoE5bPcpsAcquisition, [524](#)
  - ~GalileoE5bPcpsAcquisition, [526](#)
  - connect, [526](#)
  - disconnect, [526](#)
  - GalileoE5bPcpsAcquisition, [525](#)
  - get\_left\_block, [526](#)
  - get\_right\_block, [526](#)
  - implementation, [527](#)
  - init, [527](#)
  - item\_size, [527](#)
  - mag, [527](#)
  - reset, [527](#)
  - role, [528](#)
  - set\_channel, [528](#)
  - set\_channel\_fsm, [528](#)
  - set\_doppler\_center, [528](#)
  - set\_doppler\_max, [529](#)
  - set\_doppler\_step, [529](#)
  - set\_gnss\_synchro, [529](#)
  - set\_local\_code, [529](#)
  - set\_resampler\_latency, [529](#)
  - set\_state, [530](#)
  - set\_threshold, [530](#)
  - stop\_acquisition, [530](#)
- GalileoE5bPcpsAcquisitionFpga, [531](#)
  - ~GalileoE5bPcpsAcquisitionFpga, [532](#)
  - connect, [533](#)
  - disconnect, [533](#)
  - GalileoE5bPcpsAcquisitionFpga, [532](#)
  - get\_left\_block, [533](#)
  - get\_right\_block, [533](#)
  - implementation, [533](#)
  - init, [534](#)
  - item\_size, [534](#)
  - mag, [534](#)
  - reset, [534](#)
  - role, [534](#)
  - set\_channel, [535](#)
  - set\_channel\_fsm, [535](#)
  - set\_doppler\_center, [535](#)
  - set\_doppler\_max, [535](#)
  - set\_doppler\_step, [536](#)
  - set\_gnss\_synchro, [536](#)
  - set\_local\_code, [536](#)
  - set\_resampler\_latency, [536](#)
  - set\_single\_doppler\_flag, [536](#)
  - set\_state, [537](#)
  - set\_threshold, [537](#)
  - stop\_acquisition, [537](#)
- GalileoE5bTelemetryDecoder, [538](#)
  - connect, [538](#)
  - disconnect, [539](#)
  - get\_left\_block, [539](#)
  - get\_right\_block, [539](#)
  - implementation, [539](#)
- GalileoE6DIIPIITracking, [540](#)
  - connect, [540](#)
  - disconnect, [541](#)
  - get\_left\_block, [541](#)
  - get\_right\_block, [541](#)
  - implementation, [541](#)
  - set\_channel, [541](#)
  - set\_gnss\_synchro, [542](#)
  - stop\_tracking, [542](#)
- GalileoE6PcpsAcquisition, [542](#)
  - implementation, [543](#)
  - init, [544](#)
  - mag, [544](#)
  - reset, [544](#)

- set\_channel, [544](#)
- set\_channel\_fsm, [544](#)
- set\_doppler\_center, [545](#)
- set\_doppler\_max, [545](#)
- set\_doppler\_step, [545](#)
- set\_gnss\_synchro, [545](#)
- set\_local\_code, [545](#)
- set\_resampler\_latency, [546](#)
- set\_state, [546](#)
- set\_threshold, [546](#)
- stop\_acquisition, [546](#)
- GalileoE6TelemetryDecoder, [547](#)
  - connect, [547](#)
  - disconnect, [548](#)
  - get\_left\_block, [548](#)
  - get\_right\_block, [548](#)
  - implementation, [548](#)
- Gamma
  - telemetry\_decoder\_libs, [178](#)
- gen\_signal\_source.h, [1072](#)
- GenSignalSource, [549](#)
  - ~GenSignalSource, [549](#)
  - GenSignalSource, [549](#)
  - implementation, [550](#)
- general\_work
  - beidou\_b1i\_telemetry\_decoder\_gs, [308](#)
  - beidou\_b3i\_telemetry\_decoder\_gs, [310](#)
  - dll\_pll\_veml\_tracking\_fpga, [394](#)
  - galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc, [429](#)
  - galileo\_pcps\_8ms\_acquisition\_cc, [454](#)
  - galileo\_telemetry\_decoder\_gs, [459](#)
  - glonass\_l1\_ca\_telemetry\_decoder\_gs, [583](#)
  - glonass\_l2\_ca\_telemetry\_decoder\_gs, [587](#)
  - gps\_l1\_ca\_telemetry\_decoder\_gs, [700](#)
  - gps\_l2c\_telemetry\_decoder\_gs, [701](#)
  - pcps\_acquisition, [824](#)
  - pcps\_acquisition\_fine\_doppler\_cc, [829](#)
  - pcps\_assisted\_acquisition\_cc, [840](#)
  - pcps\_cccwsr\_acquisition\_cc, [845](#)
  - pcps\_openc1\_acquisition\_cc, [850](#)
  - pcps\_quicksync\_acquisition\_cc, [856](#)
  - pcps\_tong\_acquisition\_cc, [860](#)
  - sbas\_l1\_telemetry\_decoder\_gs, [919](#)
- Geo\_to\_ECEF
  - algorithms\_libs, [102](#)
- GeoJSON\_Printer, [550](#)
- geofunctions.h, [1073](#)
- geojson\_printer.h, [1074](#)
- geph\_t, [551](#)
- Get
  - INIReader, [794](#)
- get
  - algorithms\_libs, [102](#)
- get\_GPS\_week
  - Gps\_Navigation\_Message, [705](#)
- get\_LLH
  - TcpCmdInterface, [947](#)
- get\_PRN
  - Gnss\_Satellite, [611](#)
- get\_TOW
  - Gps\_Navigation\_Message, [706](#)
- get\_almanac
  - Glonass\_Gnav\_Navigation\_Message, [574](#)
- get\_avg\_height
  - Pvt\_Solution, [873](#)
- get\_avg\_latitude
  - Pvt\_Solution, [873](#)
- get\_avg\_longitude
  - Pvt\_Solution, [873](#)
- get\_beidou\_dnav\_almanac\_map
  - rtklib\_pvt\_gs, [906](#)
- get\_beidou\_dnav\_ephemeris\_map
  - rtklib\_pvt\_gs, [906](#)
- get\_block
  - Gnss\_Satellite, [611](#)
- get\_clock\_drift\_ppm
  - Pvt\_Solution, [873](#)
- get\_code\_nco
  - Tracking\_2nd\_DLL\_filter, [951](#)
- get\_course\_over\_ground
  - Pvt\_Solution, [873](#)
- get\_current\_status\_map
  - channel\_status\_msg\_receiver, [363](#)
- get\_current\_status\_pvt
  - channel\_status\_msg\_receiver, [363](#)
- get\_ephemeris
  - Beidou\_Dnav\_Navigation\_Message, [335](#)
  - FrontEndCal, [420](#)
  - Glonass\_Gnav\_Navigation\_Message, [574](#)
  - Gps\_CNAV\_Navigation\_Message, [673](#)
  - Gps\_Navigation\_Message, [705](#)
- get\_flag\_iono\_valid
  - Gps\_Navigation\_Message, [705](#)
- get\_flag\_utc\_model\_valid
  - Gps\_Navigation\_Message, [705](#)
- get\_frame\_number
  - Glonass\_Gnav\_Navigation\_Message, [574](#)
- get\_galileo\_almanac\_map
  - rtklib\_pvt\_gs, [906](#)
- get\_galileo\_ephemeris\_map
  - rtklib\_pvt\_gs, [907](#)
- get\_gps\_almanac\_map
  - rtklib\_pvt\_gs, [907](#)
- get\_gps\_ephemeris\_map
  - rtklib\_pvt\_gs, [907](#)
- get\_height
  - Pvt\_Solution, [873](#)
- get\_iono
  - Beidou\_Dnav\_Navigation\_Message, [335](#)
  - Gps\_CNAV\_Navigation\_Message, [673](#)
  - Gps\_Navigation\_Message, [705](#)
- get\_latest\_PVT
  - rtklib\_pvt\_gs, [907](#)
- get\_latitude
  - Pvt\_Solution, [874](#)

- get\_left\_block
  - GalileoE1DIIPIIVemlTrackingFpga, [467](#)
  - GalileoE1PcpsAmbiguousAcquisitionFpga, [480](#)
  - GalileoE5aDIIPIITrackingFpga, [502](#)
  - GalileoE5aPcpsAcquisitionFpga, [516](#)
  - GalileoE5bDIIPIITracking, [523](#)
  - GalileoE5bPcpsAcquisition, [526](#)
  - GalileoE5bPcpsAcquisitionFpga, [533](#)
  - GalileoE5bTelemetryDecoder, [539](#)
  - GalileoE6DIIPIITracking, [541](#)
  - GalileoE6TelemetryDecoder, [548](#)
  - GpsL1CaDIIPIITrackingFpga, [714](#)
  - GpsL1CaPcpsAcquisitionFpga, [731](#)
  - GpsL5DIIPIITrackingFpga, [771](#)
  - GpsL5iPcpsAcquisitionFpga, [781](#)
- get\_left\_block\_acq
  - Channel, [358](#)
- get\_left\_block\_trk
  - Channel, [358](#)
- get\_longitude
  - Pvt\_Solution, [874](#)
- get\_navfilename
  - Rinex\_Printer, [880](#)
- get\_num\_valid\_observations
  - Pvt\_Solution, [874](#)
- get\_obsfilename
  - Rinex\_Printer, [881](#)
- get\_pvt
  - GNSSFlowgraph, [642](#)
- get\_rf\_link
  - Gnss\_Satellite, [611](#)
- get\_right\_block
  - Channel, [359](#)
  - GalileoE1DIIPIIVemlTrackingFpga, [467](#)
  - GalileoE1PcpsAmbiguousAcquisitionFpga, [480](#)
  - GalileoE5aDIIPIITrackingFpga, [503](#)
  - GalileoE5aPcpsAcquisitionFpga, [516](#)
  - GalileoE5bDIIPIITracking, [523](#)
  - GalileoE5bPcpsAcquisition, [526](#)
  - GalileoE5bPcpsAcquisitionFpga, [533](#)
  - GalileoE5bTelemetryDecoder, [539](#)
  - GalileoE6DIIPIITracking, [541](#)
  - GalileoE6TelemetryDecoder, [548](#)
  - GpsL1CaDIIPIITrackingFpga, [714](#)
  - GpsL1CaPcpsAcquisitionFpga, [731](#)
  - GpsL5DIIPIITrackingFpga, [772](#)
  - GpsL5iPcpsAcquisitionFpga, [781](#)
- get\_right\_block\_acq
  - Channel, [359](#)
- get\_right\_block\_trk
  - Channel, [359](#)
- get\_satellite
  - Gnss\_Signal, [624](#)
- get\_satellite\_PRN
  - Gps\_Navigation\_Message, [705](#)
- get\_signal\_str
  - Gnss\_Signal, [624](#)
- get\_speed\_over\_ground
  - Pvt\_Solution, [874](#)
- get\_system
  - Gnss\_Satellite, [611](#)
- get\_system\_short
  - Gnss\_Satellite, [612](#)
- get\_time\_offset\_s
  - Pvt\_Solution, [874](#)
- get\_utc\_model
  - Beidou\_Dnav\_Navigation\_Message, [335](#)
  - Glonass\_Gnav\_Navigation\_Message, [575](#)
  - Gps\_CNAV\_Navigation\_Message, [674](#)
  - Gps\_Navigation\_Message, [706](#)
- get\_utc\_time
  - TcpCmdInterface, [947](#)
- GetBlock
  - GNSSBlockFactory, [637](#)
- GetInteger
  - INIReader, [794](#)
- Glonass\_Gnav\_Almanac, [551](#)
  - d\_C\_n, [553](#)
  - d\_Delta\_T\_n\_A\_dot, [554](#)
  - d\_Delta\_T\_n\_A, [554](#)
  - d\_Delta\_i\_n\_A, [553](#)
  - d\_H\_n\_A, [554](#)
  - d\_KP, [555](#)
  - d\_M\_n\_A, [555](#)
  - d\_epsilon\_n\_A, [554](#)
  - d\_I\_n, [555](#)
  - d\_lambda\_n\_A, [555](#)
  - d\_n\_A, [556](#)
  - d\_omega\_n\_A, [556](#)
  - d\_t\_lambda\_n\_A, [556](#)
  - d\_tau\_n\_A, [556](#)
  - Glonass\_Gnav\_Almanac, [553](#)
  - i\_satellite\_PRN, [557](#)
  - i\_satellite\_freq\_channel, [557](#)
  - i\_satellite\_slot\_number, [557](#)
  - serialize, [553](#)
- Glonass\_Gnav\_Ephemeris, [558](#)
  - compute\_GLONASS\_time, [560](#)
  - d\_AXn, [563](#)
  - d\_AYn, [563](#)
  - d\_AZn, [564](#)
  - d\_B\_n, [564](#)
  - d\_Delta\_tau\_n, [564](#)
  - d\_E\_n, [565](#)
  - d\_F\_T, [565](#)
  - d\_N\_T, [567](#)
  - d\_P\_1, [567](#)
  - d\_P\_2, [568](#)
  - d\_P\_3, [568](#)
  - d\_P\_4, [568](#)
  - d\_TOW, [570](#)
  - d\_VXn, [570](#)
  - d\_VYn, [570](#)
  - d\_VZn, [570](#)
  - d\_WN, [570](#)
  - d\_Xn, [571](#)

- d\_Yn, [571](#)
- d\_Zn, [571](#)
- d\_dtr, [564](#)
- d\_gamma\_n, [565](#)
- d\_iode, [565](#)
- d\_l3rd\_n, [566](#)
- d\_l5th\_n, [566](#)
- d\_M, [566](#)
- d\_m, [566](#)
- d\_n, [567](#)
- d\_P, [567](#)
- d\_satClkDrift, [568](#)
- d\_t\_b, [569](#)
- d\_t\_k, [569](#)
- d\_tau\_c, [569](#)
- d\_tau\_n, [569](#)
- d\_tod, [569](#)
- d\_yr, [571](#)
- Glonass\_Gnav\_Ephemeris, [560](#)
- glot\_to\_gpst, [561](#)
- glot\_to\_utc, [561](#)
- i\_satellite\_PRN, [572](#)
- i\_satellite\_freq\_channel, [571](#)
- i\_satellite\_slot\_number, [572](#)
- serialize, [562](#)
- sv\_clock\_drift, [563](#)
- Glonass\_Gnav\_Navigation\_Message, [572](#)
  - CRC\_test, [574](#)
  - get\_almanac, [574](#)
  - get\_ephemeris, [574](#)
  - get\_frame\_number, [574](#)
  - get\_utc\_model, [575](#)
  - Glonass\_Gnav\_Navigation\_Message, [573](#)
  - have\_new\_almanac, [575](#)
  - have\_new\_ephemeris, [575](#)
  - have\_new\_utc\_model, [575](#)
  - string\_decoder, [575](#)
- Glonass\_Gnav\_Utc\_Model, [576](#)
  - d\_B1, [578](#)
  - d\_B2, [578](#)
  - d\_N\_4, [578](#)
  - d\_N\_A, [578](#)
  - d\_tau\_c, [579](#)
  - d\_tau\_gps, [579](#)
  - Glonass\_Gnav\_Utc\_Model, [577](#)
  - serialize, [577](#)
  - utc\_time, [577](#)
- Glonass\_L1\_Ca\_Dll\_Pll\_Tracking\_cc, [581](#)
- Glonass\_L2\_Ca\_Dll\_Pll\_Tracking\_cc, [585](#)
- glonass\_gnav\_almanac
  - Rtklib\_Solver, [909](#)
- glonass\_gnav\_almanac.h, [1074](#)
- glonass\_gnav\_ephemeris.h, [1075](#)
- glonass\_gnav\_ephemeris\_map
  - Rtklib\_Solver, [910](#)
- glonass\_gnav\_navigation\_message.h, [1076](#)
- glonass\_gnav\_utc\_model
  - Rtklib\_Solver, [910](#)
- glonass\_gnav\_utc\_model.h, [1077](#)
- glonass\_l1\_ca\_code\_gen\_complex
  - algorithms\_libs, [103](#)
- glonass\_l1\_ca\_code\_gen\_complex\_sampled
  - algorithms\_libs, [103](#)
- glonass\_l1\_ca\_dll\_pll\_c\_aid\_tracking.h, [1077](#)
- glonass\_l1\_ca\_dll\_pll\_c\_aid\_tracking\_cc, [579](#)
- glonass\_l1\_ca\_dll\_pll\_c\_aid\_tracking\_cc.h, [1078](#)
- glonass\_l1\_ca\_dll\_pll\_c\_aid\_tracking\_sc, [580](#)
- glonass\_l1\_ca\_dll\_pll\_c\_aid\_tracking\_sc.h, [1079](#)
- glonass\_l1\_ca\_dll\_pll\_tracking.h, [1080](#)
- glonass\_l1\_ca\_dll\_pll\_tracking\_cc.h, [1081](#)
- glonass\_l1\_ca\_pcps\_acquisition.h, [1082](#)
- glonass\_l1\_ca\_telemetry\_decoder.h, [1082](#)
- glonass\_l1\_ca\_telemetry\_decoder\_gs, [582](#)
  - ~glonass\_l1\_ca\_telemetry\_decoder\_gs, [583](#)
- general\_work, [583](#)
- set\_channel, [583](#)
- set\_satellite, [583](#)
- glonass\_l1\_ca\_telemetry\_decoder\_gs.h, [1083](#)
- glonass\_l2\_ca\_code\_gen\_complex
  - algorithms\_libs, [103](#)
- glonass\_l2\_ca\_code\_gen\_complex\_sampled
  - algorithms\_libs, [103](#)
- glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking.h, [1088](#)
- glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking\_cc, [584](#)
- glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking\_cc.h, [1089](#)
- glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking\_sc, [585](#)
- glonass\_l2\_ca\_dll\_pll\_c\_aid\_tracking\_sc.h, [1090](#)
- glonass\_l2\_ca\_dll\_pll\_tracking.h, [1091](#)
- glonass\_l2\_ca\_dll\_pll\_tracking\_cc.h, [1091](#)
- glonass\_l2\_ca\_pcps\_acquisition.h, [1092](#)
- glonass\_l2\_ca\_telemetry\_decoder.h, [1093](#)
- glonass\_l2\_ca\_telemetry\_decoder\_gs, [586](#)
  - ~glonass\_l2\_ca\_telemetry\_decoder\_gs, [587](#)
- general\_work, [587](#)
- set\_channel, [587](#)
- set\_satellite, [587](#)
- glonass\_l2\_ca\_telemetry\_decoder\_gs.h, [1094](#)
- glonass\_l2\_signal\_replica.h, [1095](#)
- GlonassL1CaDllPllCAidTracking, [588](#)
  - implementation, [589](#)
  - set\_channel, [589](#)
  - set\_gnss\_synchro, [589](#)
  - stop\_tracking, [589](#)
- GlonassL1CaDllPllTracking, [590](#)
  - implementation, [591](#)
  - set\_channel, [591](#)
  - set\_gnss\_synchro, [591](#)
  - stop\_tracking, [591](#)
- GlonassL1CaPcpsAcquisition, [592](#)
  - implementation, [593](#)
  - init, [593](#)
  - mag, [593](#)
  - reset, [594](#)
  - set\_channel, [594](#)
  - set\_channel\_fsm, [594](#)
  - set\_doppler\_max, [594](#)

- set\_doppler\_step, 595
  - set\_gnss\_synchro, 595
  - set\_local\_code, 595
  - set\_state, 595
  - set\_threshold, 595
  - stop\_acquisition, 596
- GlonassL1CaTelemetryDecoder, 596
  - implementation, 597
- GlonassL2CaDIIPICAidTracking, 597
  - implementation, 598
  - set\_channel, 598
  - set\_gnss\_synchro, 598
  - stop\_tracking, 599
- GlonassL2CaDIPIITracking, 599
  - implementation, 600
  - set\_channel, 600
  - set\_gnss\_synchro, 600
  - stop\_tracking, 600
- GlonassL2CaPcpsAcquisition, 601
  - implementation, 602
  - init, 602
  - mag, 602
  - reset, 603
  - set\_channel, 603
  - set\_channel\_fsm, 603
  - set\_doppler\_max, 603
  - set\_doppler\_step, 604
  - set\_gnss\_synchro, 604
  - set\_local\_code, 604
  - set\_state, 604
  - set\_threshold, 604
  - stop\_acquisition, 605
- GlonassL2CaTelemetryDecoder, 605
  - implementation, 606
- glot\_to\_gpst
  - Glonass\_Gnav\_Ephemeris, 561
- glot\_to\_utc
  - Glonass\_Gnav\_Ephemeris, 561
- gn3s\_signal\_source.h, 1095
- Gn3sSignalSource, 606
  - implementation, 607
- GnMaxSignalSource, 607
  - implementation, 608
- gnmax\_signal\_source.h, 1096
- Gnss\_Satellite, 609
  - ~Gnss\_Satellite, 610
  - get\_PRN, 611
  - get\_block, 611
  - get\_rf\_link, 611
  - get\_system, 611
  - get\_system\_short, 612
  - Gnss\_Satellite, 610, 611
  - operator<<, 613
  - operator=, 612
  - operator==, 613
  - update\_PRN, 612
  - what\_block, 612
- Gnss\_Sdr\_Supl\_Client, 614
  - load\_cnav\_ephemeris\_xml, 616
  - load\_cnav\_utc\_xml, 617
  - load\_ephemeris\_xml, 617
  - load\_gal\_almanac\_xml, 617
  - load\_gal\_ephemeris\_xml, 617
  - load\_gal\_iono\_xml, 617
  - load\_gal\_utc\_xml, 617
  - load\_glo\_utc\_xml, 618
  - load\_gnav\_ephemeris\_xml, 618
  - load\_gps\_almanac\_xml, 618
  - load\_iono\_xml, 618
  - load\_ref\_location\_xml, 618
  - load\_ref\_time\_xml, 618
  - load\_utc\_xml, 619
  - save\_cnav\_ephemeris\_map\_xml, 619
  - save\_cnav\_utc\_xml, 619
  - save\_ephemeris\_map\_xml, 619
  - save\_gal\_almanac\_xml, 619
  - save\_gal\_ephemeris\_map\_xml, 620
  - save\_gal\_iono\_xml, 620
  - save\_gal\_utc\_xml, 620
  - save\_glo\_utc\_xml, 620
  - save\_gnav\_ephemeris\_map\_xml, 620
  - save\_gps\_almanac\_xml, 621
  - save\_iono\_xml, 621
  - save\_ref\_location\_xml, 621
  - save\_ref\_time\_xml, 621
  - save\_utc\_xml, 621
- Gnss\_Sdr\_Valve, 623
- Gnss\_Signal, 623
  - get\_satellite, 624
  - get\_signal\_str, 624
  - operator<<, 624
  - operator==, 625
- Gnss\_Synchro, 625
  - ~Gnss\_Synchro, 627
  - Acq\_delay\_samples, 629
  - Acq\_doppler\_hz, 629
  - Acq\_doppler\_step, 629
  - Acq\_samplestamp\_samples, 629
  - CN0\_dB\_hz, 630
  - Carrier\_Doppler\_hz, 629
  - Carrier\_phase\_rads, 630
  - Channel\_ID, 630
  - Code\_phase\_samples, 630
  - correlation\_length\_ms, 631
  - Flag\_valid\_acquisition, 631
  - Flag\_valid\_pseudorange, 631
  - Flag\_valid\_symbol\_output, 631
  - Flag\_valid\_word, 632
  - fs, 632
  - Gnss\_Synchro, 627
  - interp\_TOW\_ms, 632
  - operator=, 628
  - PRN, 632
  - Prompt\_I, 633
  - Prompt\_Q, 633
  - Pseudorange\_m, 633

- RX\_time, [633](#)
- serialize, [628](#)
- Signal, [634](#)
- System, [634](#)
- TOW\_at\_current\_symbol\_ms, [634](#)
- Tracking\_sample\_counter, [634](#)
- Gnss\_Synchro\_Udp\_Sink, [636](#)
- gnss\_block\_factory.h, [1097](#)
- gnss\_block\_interface.h, [1097](#)
- Gnss\_circular\_deque
  - algorithms\_libs, [103](#), [104](#)
- Gnss\_circular\_deque< T >, [608](#)
- gnss\_circular\_deque.h, [1098](#)
- gnss\_flowgraph.h, [1099](#)
- gnss\_frequencies.h, [1099](#)
- gnss\_obs\_codes.h, [1100](#)
- gnss\_satellite.h, [1103](#)
- gnss\_sdr\_create\_directory.h, [1104](#)
- gnss\_sdr\_flags, [113](#)
  - DECLARE\_bool, [114](#)
  - DECLARE\_double, [114](#)
  - DECLARE\_int32, [114](#), [115](#)
  - DECLARE\_string, [115](#), [116](#)
- gnss\_sdr\_flags.h, [1104](#)
- gnss\_sdr\_fpga\_sample\_counter, [613](#)
- gnss\_sdr\_fpga\_sample\_counter.h, [1106](#)
- gnss\_sdr\_make\_unique.h, [1106](#)
- gnss\_sdr\_sample\_counter, [614](#)
- gnss\_sdr\_sample\_counter.h, [1107](#)
- gnss\_sdr\_supl\_client.h, [1108](#)
- gnss\_sdr\_time\_counter, [622](#)
- gnss\_sdr\_time\_counter.h, [1108](#)
- gnss\_sdr\_valve.h, [1109](#)
- gnss\_signal.h, [1110](#)
- gnss\_signal\_replica.h, [1110](#)
- gnss\_synchro.h, [1111](#)
- gnss\_synchro\_monitor, [635](#)
  - ~gnss\_synchro\_monitor, [636](#)
- gnss\_synchro\_monitor.h, [1112](#)
- gnss\_synchro\_udp\_sink.h, [1113](#)
- Gnuplot, [643](#)
  - Gnuplot, [645](#)
  - operator<<, [645](#)
  - replot, [645](#)
  - unset\_title, [645](#)
- gnuplot\_i.h, [1113](#)
- GnuplotException, [646](#)
- Gps\_Acq\_Assist, [646](#)
  - Azimuth, [648](#)
  - Code\_Phase, [648](#)
  - Code\_Phase\_int, [648](#)
  - Code\_Phase\_window, [648](#)
  - d\_Doppler0, [648](#)
  - d\_Doppler1, [648](#)
  - d\_TOW, [649](#)
  - dopplerUncertainty, [649](#)
  - Elevation, [649](#)
  - GPS\_Bit\_Number, [649](#)
  - Gps\_Acq\_Assist, [647](#)
  - i\_satellite\_PRN, [649](#)
- Gps\_Almanac, [650](#)
  - d\_A\_f0, [651](#)
  - d\_A\_f1, [651](#)
  - d\_Delta\_i, [651](#)
  - d\_M\_0, [652](#)
  - d\_OMEGA0, [652](#)
  - d\_OMEGA\_DOT, [652](#)
  - d\_OMEGA, [652](#)
  - d\_e\_eccentricity, [651](#)
  - d\_sqrt\_A, [652](#)
  - Gps\_Almanac, [651](#)
  - i\_AS\_status, [653](#)
  - i\_SV\_health, [653](#)
  - i\_Toa, [653](#)
  - i\_WNa, [653](#)
  - i\_satellite\_PRN, [653](#)
- Gps\_CNAV\_Ephemeris, [654](#)
  - b\_alert\_flag, [659](#)
  - b\_antispoofing\_flag, [659](#)
  - b\_integrity\_status\_flag, [659](#)
  - d\_A\_DOT, [659](#)
  - d\_A\_f0, [660](#)
  - d\_A\_f1, [660](#)
  - d\_A\_f2, [660](#)
  - d\_Cic, [660](#)
  - d\_Cis, [661](#)
  - d\_Crc, [661](#)
  - d\_Crs, [661](#)
  - d\_Cuc, [661](#)
  - d\_Cus, [662](#)
  - d\_DELTA\_DOT\_N, [662](#)
  - d\_DELTA\_OMEGA\_DOT, [662](#)
  - d\_DELTA\_A, [662](#)
  - d\_Delta\_n, [662](#)
  - d\_IDOT, [663](#)
  - d\_M\_0, [664](#)
  - d\_OMEGA0, [664](#)
  - d\_OMEGA, [664](#)
  - d\_TGD, [666](#)
  - d\_TOW, [667](#)
  - d\_Toc, [666](#)
  - d\_Toe1, [666](#)
  - d\_Toe2, [666](#)
  - d\_Top, [667](#)
  - d\_URA0, [667](#)
  - d\_URA1, [667](#)
  - d\_URA2, [667](#)
  - d\_dtr, [663](#)
  - d\_e\_eccentricity, [663](#)
  - d\_i\_0, [663](#)
  - d\_satClkDrift, [664](#)
  - d\_satpos\_X, [665](#)
  - d\_satpos\_Y, [665](#)
  - d\_satpos\_Z, [665](#)
  - d\_satvel\_X, [665](#)
  - d\_satvel\_Y, [665](#)

- d\_satvel\_Z, 666
- Gps\_CNAV\_Ephemeris, 656
- i\_GPS\_week, 668
- i\_URA, 668
- i\_signal\_health, 668
- satellitePosition, 657
- serialize, 657
- sv\_clock\_drift, 658
- sv\_clock\_relativistic\_term, 658
- Gps\_CNAV\_Iono, 668
  - d\_alpha0, 670
  - d\_alpha1, 670
  - d\_alpha2, 670
  - d\_alpha3, 671
  - d\_beta0, 671
  - d\_beta1, 671
  - d\_beta2, 671
  - d\_beta3, 672
  - Gps\_CNAV\_Iono, 669
  - serialize, 670
  - valid, 672
- Gps\_CNAV\_Navigation\_Message, 672
  - get\_ephemeris, 673
  - get\_iono, 673
  - get\_utc\_model, 674
  - Gps\_CNAV\_Navigation\_Message, 673
  - have\_new\_ephemeris, 674
  - have\_new\_iono, 674
  - have\_new\_utc\_model, 674
- Gps\_CNAV\_Utc\_Model, 674
  - d\_A0, 676
  - d\_A1, 676
  - d\_A2, 676
  - d\_DeltaT\_LSF, 676
  - d\_DeltaT\_LS, 676
  - d\_t\_OT, 676
  - Gps\_CNAV\_Utc\_Model, 675
  - i\_DN, 677
  - i\_WN\_LSF, 677
  - i\_WN\_T, 677
- Gps\_Ephemeris, 677
  - b\_L2\_P\_data\_flag, 683
  - b\_alert\_flag, 682
  - b\_antispoofing\_flag, 683
  - b\_fit\_interval\_flag, 683
  - b\_integrity\_status\_flag, 683
  - d\_A\_f0, 684
  - d\_A\_f1, 684
  - d\_A\_f2, 684
  - d\_Cic, 684
  - d\_Cis, 685
  - d\_Crc, 685
  - d\_Crs, 685
  - d\_Cuc, 685
  - d\_Cus, 686
  - d\_Delta\_n, 686
  - d\_IDOT, 687
  - d\_IODE\_SF2, 687
  - d\_IODE\_SF3, 688
  - d\_IODC, 687
  - d\_M\_0, 688
  - d\_OMEGA0, 688
  - d\_OMEGA\_DOT, 689
  - d\_OMEGA, 688
  - d\_TGD, 690
  - d\_TOW, 691
  - d\_Toc, 691
  - d\_Toe, 691
  - d\_dtr, 686
  - d\_e\_eccentricity, 686
  - d\_i\_0, 687
  - d\_satClkDrift, 689
  - d\_satpos\_X, 689
  - d\_satpos\_Y, 689
  - d\_satpos\_Z, 689
  - d\_satvel\_X, 690
  - d\_satvel\_Y, 690
  - d\_satvel\_Z, 690
  - d\_sqrt\_A, 690
  - Gps\_Ephemeris, 680
  - i\_AODO, 691
  - i\_GPS\_week, 692
  - i\_SV\_accuracy, 692
  - i\_code\_on\_L2, 692
  - satelliteBlock, 692
  - satellitePosition, 680
  - serialize, 680
  - sv\_clock\_drift, 682
  - sv\_clock\_relativistic\_term, 682
- Gps\_Iono, 693
  - d\_alpha0, 694
  - d\_alpha1, 694
  - d\_alpha2, 695
  - d\_alpha3, 695
  - d\_beta0, 695
  - d\_beta1, 695
  - d\_beta2, 696
  - d\_beta3, 696
  - Gps\_Iono, 694
  - serialize, 694
  - valid, 696
- Gps\_L1\_Ca\_Dll\_Pll\_Tracking\_GPU\_cc, 697
- Gps\_L1\_Ca\_Kf\_Tracking\_cc, 697
- Gps\_L1\_Ca\_Tcp\_Connector\_Tracking\_cc, 698
- Gps\_Navigation\_Message, 703
  - get\_GPS\_week, 705
  - get\_TOW, 706
  - get\_ephemeris, 705
  - get\_flag\_iono\_valid, 705
  - get\_flag\_utc\_model\_valid, 705
  - get\_iono, 705
  - get\_satellite\_PRN, 705
  - get\_utc\_model, 706
  - Gps\_Navigation\_Message, 704
  - set\_channel, 706
  - set\_satellite\_PRN, 706

- subframe\_decoder, 706
- utc\_time, 707
- Gps\_Utc\_Model, 707
  - d\_A0, 708
  - d\_A1, 708
  - d\_A2, 708
  - d\_DeltaT\_LSF, 709
  - d\_DeltaT\_LS, 708
  - d\_t\_OT, 709
  - Gps\_Utc\_Model, 708
  - i\_DN, 709
  - i\_WN\_LSF, 709
  - i\_WN\_T, 709
- gps\_acq\_assist.h, 1114
- gps\_almanac.h, 1115
- gps\_cnav\_ephemeris.h, 1118
- gps\_cnav\_ephemeris\_map
  - Rtklib\_Solver, 910
- gps\_cnav\_iono.h, 1118
- gps\_cnav\_navigation\_message.h, 1119
- gps\_cnav\_utc\_model.h, 1120
- gps\_ephemeris.h, 1120
- gps\_ephemeris\_map
  - Rtklib\_Solver, 910
- gps\_iono.h, 1121
- gps\_l1\_ca\_code\_gen\_complex
  - algorithms\_libs, 104
- gps\_l1\_ca\_code\_gen\_complex\_sampled
  - algorithms\_libs, 104
- gps\_l1\_ca\_code\_gen\_float
  - algorithms\_libs, 104
- gps\_l1\_ca\_code\_gen\_int
  - algorithms\_libs, 105
- gps\_l1\_ca\_dll\_pll\_tracking.h, 1125
- gps\_l1\_ca\_dll\_pll\_tracking\_fpga.h, 1125
- gps\_l1\_ca\_dll\_pll\_tracking\_gpu.h, 1126
- gps\_l1\_ca\_dll\_pll\_tracking\_gpu\_cc.h, 1127
- gps\_l1\_ca\_kf\_tracking.h, 1128
- gps\_l1\_ca\_kf\_tracking\_cc.h, 1128
- gps\_l1\_ca\_pcps\_acquisition.h, 1129
- gps\_l1\_ca\_pcps\_acquisition\_fine\_doppler.h, 1130
- gps\_l1\_ca\_pcps\_acquisition\_fpga.h, 1131
- gps\_l1\_ca\_pcps\_assisted\_acquisition.h, 1131
- gps\_l1\_ca\_pcps\_opencl\_acquisition.h, 1132
- gps\_l1\_ca\_pcps\_quicksync\_acquisition.h, 1133
- gps\_l1\_ca\_pcps\_tong\_acquisition.h, 1133
- gps\_l1\_ca\_tcp\_connector\_tracking.h, 1134
- gps\_l1\_ca\_tcp\_connector\_tracking\_cc.h, 1135
- gps\_l1\_ca\_telemetry\_decoder.h, 1136
- gps\_l1\_ca\_telemetry\_decoder\_gs, 699
  - general\_work, 700
  - set\_channel, 700
  - set\_satellite, 700
- gps\_l1\_ca\_telemetry\_decoder\_gs.h, 1136
- gps\_l2\_m\_dll\_pll\_tracking.h, 1137
- gps\_l2\_m\_dll\_pll\_tracking\_fpga.h, 1138
- gps\_l2\_m\_pcps\_acquisition.h, 1138
- gps\_l2\_m\_pcps\_acquisition\_fpga.h, 1139
- gps\_l2c\_m\_code\_gen\_complex
  - algorithms\_libs, 105
- gps\_l2c\_m\_code\_gen\_complex\_sampled
  - algorithms\_libs, 105
- gps\_l2c\_m\_code\_gen\_float
  - algorithms\_libs, 105
- gps\_l2c\_signal\_replica.h, 1140
- gps\_l2c\_telemetry\_decoder.h, 1141
- gps\_l2c\_telemetry\_decoder\_gs, 701
  - general\_work, 701
  - set\_channel, 701
  - set\_satellite, 702
- gps\_l2c\_telemetry\_decoder\_gs.h, 1142
- gps\_l5\_dll\_pll\_tracking.h, 1144
- gps\_l5\_dll\_pll\_tracking\_fpga.h, 1144
- gps\_l5\_signal\_replica.h, 1145
- gps\_l5\_telemetry\_decoder.h, 1146
- gps\_l5\_telemetry\_decoder\_gs, 702
  - set\_channel, 703
  - set\_satellite, 703
- gps\_l5\_telemetry\_decoder\_gs.h, 1146
- gps\_l5i\_code\_gen\_complex
  - algorithms\_libs, 105
- gps\_l5i\_code\_gen\_complex\_sampled
  - algorithms\_libs, 106
- gps\_l5i\_code\_gen\_float
  - algorithms\_libs, 106
- gps\_l5i\_pcps\_acquisition.h, 1147
- gps\_l5i\_pcps\_acquisition\_fpga.h, 1148
- gps\_l5q\_code\_gen\_complex
  - algorithms\_libs, 106
- gps\_l5q\_code\_gen\_complex\_sampled
  - algorithms\_libs, 106
- gps\_l5q\_code\_gen\_float
  - algorithms\_libs, 106
- gps\_navigation\_message.h, 1148
- gps\_sdr\_signal\_replica.h, 1149
- gps\_utc\_model.h, 1150
- GpsL1CaDIIPIITracking, 710
  - implementation, 711
  - set\_channel, 711
  - set\_gnss\_synchro, 711
  - stop\_tracking, 711
- GpsL1CaDIIPIITrackingFpga, 712
  - ~GpsL1CaDIIPIITrackingFpga, 713
  - connect, 713
  - disconnect, 714
  - get\_left\_block, 714
  - get\_right\_block, 714
  - GpsL1CaDIIPIITrackingFpga, 713
  - implementation, 714
  - item\_size, 714
  - role, 715
  - set\_channel, 715
  - set\_gnss\_synchro, 715
  - start\_tracking, 715
  - stop\_tracking, 716
- GpsL1CaDIIPIITrackingGPU, 716

- implementation, 717
- set\_channel, 717
- set\_gnss\_synchro, 717
- stop\_tracking, 717
- GpsL1CaKfTracking, 718
  - implementation, 719
  - set\_channel, 719
  - set\_gnss\_synchro, 719
  - stop\_tracking, 719
- GpsL1CaPcpsAcquisition, 720
  - implementation, 721
  - init, 721
  - mag, 721
  - reset, 722
  - set\_channel, 722
  - set\_channel\_fsm, 722
  - set\_doppler\_center, 722
  - set\_doppler\_max, 722
  - set\_doppler\_step, 723
  - set\_gnss\_synchro, 723
  - set\_local\_code, 723
  - set\_resampler\_latency, 723
  - set\_state, 723
  - set\_threshold, 724
  - stop\_acquisition, 724
- GpsL1CaPcpsAcquisitionFineDoppler, 724
  - implementation, 725
  - init, 726
  - mag, 726
  - reset, 726
  - set\_channel, 726
  - set\_channel\_fsm, 726
  - set\_doppler\_max, 727
  - set\_doppler\_step, 727
  - set\_gnss\_synchro, 727
  - set\_state, 727
  - set\_threshold, 728
  - stop\_acquisition, 728
- GpsL1CaPcpsAcquisitionFpga, 728
  - ~GpsL1CaPcpsAcquisitionFpga, 730
  - connect, 730
  - disconnect, 730
  - get\_left\_block, 731
  - get\_right\_block, 731
  - GpsL1CaPcpsAcquisitionFpga, 730
  - implementation, 731
  - init, 731
  - item\_size, 731
  - mag, 732
  - reset, 732
  - role, 732
  - set\_channel, 732
  - set\_channel\_fsm, 732
  - set\_doppler\_center, 733
  - set\_doppler\_max, 733
  - set\_doppler\_step, 733
  - set\_gnss\_synchro, 733
  - set\_local\_code, 733
  - set\_resampler\_latency, 734
  - set\_state, 734
  - set\_threshold, 734
  - stop\_acquisition, 734
- GpsL1CaPcpsAssistedAcquisition, 735
  - implementation, 736
  - init, 736
  - mag, 736
  - reset, 736
  - set\_channel, 736
  - set\_channel\_fsm, 737
  - set\_doppler\_max, 737
  - set\_doppler\_step, 737
  - set\_gnss\_synchro, 737
  - set\_threshold, 738
  - stop\_acquisition, 738
- GpsL1CaPcpsOpenCIAcquisition, 738
  - implementation, 739
  - init, 740
  - mag, 740
  - reset, 740
  - set\_channel, 740
  - set\_channel\_fsm, 740
  - set\_doppler\_max, 741
  - set\_doppler\_step, 741
  - set\_gnss\_synchro, 741
  - set\_local\_code, 741
  - set\_threshold, 742
  - stop\_acquisition, 742
- GpsL1CaPcpsQuickSyncAcquisition, 742
  - implementation, 743
  - init, 744
  - mag, 744
  - reset, 744
  - set\_channel, 744
  - set\_channel\_fsm, 744
  - set\_doppler\_max, 745
  - set\_doppler\_step, 745
  - set\_gnss\_synchro, 745
  - set\_local\_code, 745
  - set\_state, 746
  - set\_threshold, 746
  - stop\_acquisition, 746
- GpsL1CaPcpsTongAcquisition, 747
  - implementation, 748
  - init, 748
  - mag, 748
  - reset, 748
  - set\_channel, 748
  - set\_channel\_fsm, 749
  - set\_doppler\_max, 749
  - set\_doppler\_step, 749
  - set\_gnss\_synchro, 749
  - set\_local\_code, 750
  - set\_state, 750
  - set\_threshold, 750
  - stop\_acquisition, 750
- GpsL1CaTcpConnectorTracking, 751

- implementation, [751](#)
- set\_channel, [752](#)
- set\_gnss\_synchro, [752](#)
- stop\_tracking, [752](#)
- GpsL1CaTelemetryDecoder, [753](#)
  - implementation, [753](#)
- GpsL2CTelemetryDecoder, [754](#)
  - implementation, [754](#)
- GpsL2MDIIPITracking, [755](#)
  - implementation, [755](#)
  - set\_channel, [756](#)
  - set\_gnss\_synchro, [756](#)
  - stop\_tracking, [756](#)
- GpsL2MDIIPITrackingFpga, [757](#)
  - implementation, [757](#)
  - set\_channel, [758](#)
  - set\_gnss\_synchro, [758](#)
  - stop\_tracking, [758](#)
- GpsL2MPcpsAcquisition, [759](#)
  - implementation, [760](#)
  - init, [760](#)
  - mag, [760](#)
  - reset, [760](#)
  - set\_channel, [761](#)
  - set\_channel\_fsm, [761](#)
  - set\_doppler\_center, [761](#)
  - set\_doppler\_max, [761](#)
  - set\_doppler\_step, [761](#)
  - set\_gnss\_synchro, [762](#)
  - set\_local\_code, [762](#)
  - set\_resampler\_latency, [762](#)
  - set\_state, [762](#)
  - set\_threshold, [762](#)
  - stop\_acquisition, [763](#)
- GpsL2MPcpsAcquisitionFpga, [763](#)
  - implementation, [764](#)
  - init, [765](#)
  - mag, [765](#)
  - reset, [765](#)
  - set\_channel, [765](#)
  - set\_channel\_fsm, [765](#)
  - set\_doppler\_max, [766](#)
  - set\_doppler\_step, [766](#)
  - set\_gnss\_synchro, [766](#)
  - set\_local\_code, [766](#)
  - set\_state, [767](#)
  - set\_threshold, [767](#)
  - stop\_acquisition, [767](#)
- GpsL5DIIPITracking, [768](#)
  - implementation, [768](#)
  - set\_channel, [769](#)
  - set\_gnss\_synchro, [769](#)
  - stop\_tracking, [769](#)
- GpsL5DIIPITrackingFpga, [770](#)
  - ~GpsL5DIIPITrackingFpga, [771](#)
  - connect, [771](#)
  - disconnect, [771](#)
  - get\_left\_block, [771](#)
  - get\_right\_block, [772](#)
  - GpsL5DIIPITrackingFpga, [771](#)
  - implementation, [772](#)
  - item\_size, [772](#)
  - role, [772](#)
  - set\_channel, [772](#)
  - set\_gnss\_synchro, [773](#)
  - start\_tracking, [773](#)
  - stop\_tracking, [773](#)
- GpsL5TelemetryDecoder, [785](#)
  - implementation, [785](#)
- GpsL5iPcpsAcquisition, [774](#)
  - implementation, [775](#)
  - init, [775](#)
  - mag, [775](#)
  - reset, [775](#)
  - set\_channel, [776](#)
  - set\_channel\_fsm, [776](#)
  - set\_doppler\_center, [776](#)
  - set\_doppler\_max, [776](#)
  - set\_doppler\_step, [776](#)
  - set\_gnss\_synchro, [777](#)
  - set\_local\_code, [777](#)
  - set\_resampler\_latency, [777](#)
  - set\_state, [777](#)
  - set\_threshold, [777](#)
  - stop\_acquisition, [778](#)
- GpsL5iPcpsAcquisitionFpga, [778](#)
  - ~GpsL5iPcpsAcquisitionFpga, [780](#)
  - connect, [780](#)
  - disconnect, [780](#)
  - get\_left\_block, [781](#)
  - get\_right\_block, [781](#)
  - GpsL5iPcpsAcquisitionFpga, [780](#)
  - implementation, [781](#)
  - init, [781](#)
  - item\_size, [781](#)
  - mag, [782](#)
  - reset, [782](#)
  - role, [782](#)
  - set\_channel, [782](#)
  - set\_channel\_fsm, [782](#)
  - set\_doppler\_center, [783](#)
  - set\_doppler\_max, [783](#)
  - set\_doppler\_step, [783](#)
  - set\_gnss\_synchro, [783](#)
  - set\_local\_code, [783](#)
  - set\_resampler\_latency, [784](#)
  - set\_state, [784](#)
  - set\_threshold, [784](#)
  - stop\_acquisition, [784](#)
- Gpx\_Printer, [787](#)
- gpx\_printer.h, [1150](#)
- Gr\_Complex\_Ip\_Packet\_Source, [787](#)
- gr\_complex\_ip\_packet\_source.h, [1151](#)
- Gravity\_ECEF
  - algorithms\_libs, [107](#)
- great\_circle\_distance

- algorithms\_libs, [107](#)
- gtime\_t, [788](#)
- HALF\_PI
  - core\_system\_parameters, [278](#)
- HION
  - algorithms\_libs\_rtklib, [135](#)
- half\_cyc\_tag, [788](#)
- have\_new\_almanac
  - Beidou\_Dnav\_Navigation\_Message, [335](#)
  - Glonass\_Gnav\_Navigation\_Message, [575](#)
- have\_new\_ephemeris
  - Beidou\_Dnav\_Navigation\_Message, [336](#)
  - Glonass\_Gnav\_Navigation\_Message, [575](#)
  - Gps\_CNAV\_Navigation\_Message, [674](#)
- have\_new\_iono
  - Beidou\_Dnav\_Navigation\_Message, [336](#)
  - Gps\_CNAV\_Navigation\_Message, [674](#)
- have\_new\_utc\_model
  - Beidou\_Dnav\_Navigation\_Message, [336](#)
  - Glonass\_Gnav\_Navigation\_Message, [575](#)
  - Gps\_CNAV\_Navigation\_Message, [674](#)
- hex\_to\_bin
  - Rtcm, [888](#)
- hex\_to\_binary\_converter
  - algorithms\_libs, [107](#)
- hex\_to\_binary\_string
  - algorithms\_libs, [107](#)
- hex\_to\_int
  - Rtcm, [888](#)
- hex\_to\_uint
  - Rtcm, [889](#)
- hybrid\_observables.h, [1151](#)
- hybrid\_observables\_gs, [789](#)
- hybrid\_observables\_gs.h, [1152](#)
- HybridObservables, [789](#)
  - implementation, [790](#)
  - item\_size, [790](#)
- i\_0\_2
  - Galileo\_Ephemeris, [442](#)
- i\_AODO
  - Beidou\_Dnav\_Ephemeris, [328](#)
  - Gps\_Ephemeris, [691](#)
- i\_AS\_status
  - Gps\_Almanac, [653](#)
- i\_BEIDOU\_week
  - Beidou\_Dnav\_Ephemeris, [328](#)
- i\_DeltaT\_LS
  - Beidou\_Dnav\_Utc\_Model, [339](#)
- i\_DN
  - Beidou\_Dnav\_Utc\_Model, [340](#)
  - Gps\_CNAV\_Utc\_Model, [677](#)
  - Gps\_Utc\_Model, [709](#)
- i\_GPS\_week
  - Gps\_CNAV\_Ephemeris, [668](#)
  - Gps\_Ephemeris, [692](#)
- i\_SV\_accuracy
  - Beidou\_Dnav\_Ephemeris, [329](#)
  - Gps\_Ephemeris, [692](#)
- i\_SV\_health
  - Beidou\_Dnav\_Almanac, [314](#)
  - Gps\_Almanac, [653](#)
- i\_Toa
  - Gps\_Almanac, [653](#)
- i\_URA
  - Gps\_CNAV\_Ephemeris, [668](#)
- i\_WN\_LSF
  - Beidou\_Dnav\_Utc\_Model, [340](#)
  - Gps\_CNAV\_Utc\_Model, [677](#)
  - Gps\_Utc\_Model, [709](#)
- i\_WN\_T
  - Gps\_CNAV\_Utc\_Model, [677](#)
  - Gps\_Utc\_Model, [709](#)
- i\_WNa
  - Gps\_Almanac, [653](#)
- i\_code\_on\_L2
  - Gps\_Ephemeris, [692](#)
- i\_nav\_type
  - Beidou\_Dnav\_Ephemeris, [328](#)
- i\_prn
  - Sbas\_Ephemeris, [917](#)
- i\_satellite\_PRN
  - Beidou\_Dnav\_Almanac, [313](#)
  - Beidou\_Dnav\_Ephemeris, [329](#)
  - Galileo\_Almanac, [424](#)
  - Galileo\_Ephemeris, [442](#)
  - Glonass\_Gnav\_Almanac, [557](#)
  - Glonass\_Gnav\_Ephemeris, [572](#)
  - Gps\_Acq\_Assist, [649](#)
  - Gps\_Almanac, [653](#)
- i\_satellite\_freq\_channel
  - Glonass\_Gnav\_Almanac, [557](#)
  - Glonass\_Gnav\_Ephemeris, [571](#)
- i\_satellite\_slot\_number
  - Glonass\_Gnav\_Almanac, [557](#)
  - Glonass\_Gnav\_Ephemeris, [572](#)
- i\_sig\_type
  - Beidou\_Dnav\_Ephemeris, [329](#)
- i\_signal\_health
  - Gps\_CNAV\_Ephemeris, [668](#)
- i\_sv\_ura
  - Sbas\_Ephemeris, [917](#)
- i\_t0
  - Sbas\_Ephemeris, [917](#)
- iDot\_2
  - Galileo\_Ephemeris, [442](#)
- IGPBAND1
  - rtklib\_sbas.h, [1235](#)
- IGPBAND2
  - rtklib\_sbas.h, [1235](#)
- INIReader, [794](#)
  - Get, [794](#)
  - GetInteger, [794](#)
  - INIReader, [794](#)
  - ParseError, [795](#)
- INIReader.h, [1156](#)

- INT\_SWAP\_STAT
  - algorithms\_libs\_rtklib, [135](#)
- INT\_SWAP\_TRAC
  - algorithms\_libs\_rtklib, [135](#)
- IONOOPT\_BRDC
  - algorithms\_libs\_rtklib, [135](#)
- IONOOPT\_EST
  - algorithms\_libs\_rtklib, [136](#)
- IONOOPT\_IFLC
  - algorithms\_libs\_rtklib, [136](#)
- IONOOPT\_LEX
  - algorithms\_libs\_rtklib, [136](#)
- IONOOPT\_OFF
  - algorithms\_libs\_rtklib, [136](#)
- IONOOPT\_QZS
  - algorithms\_libs\_rtklib, [136](#)
- IONOOPT\_SBAS
  - algorithms\_libs\_rtklib, [137](#)
- IONOOPT\_STEC
  - algorithms\_libs\_rtklib, [137](#)
- IONOOPT\_TEC
  - algorithms\_libs\_rtklib, [137](#)
- ibyte\_to\_cbyte.h, [1153](#)
- ibyte\_to\_complex.h, [1153](#)
- ibyte\_to\_cshort.h, [1154](#)
- lbyteToCbyte, [791](#)
  - implementation, [791](#)
- lbyteToComplex, [792](#)
  - implementation, [792](#)
- lbyteToCshort, [793](#)
  - implementation, [793](#)
- implementation
  - Ad9361FpgaSignalSource, [298](#)
  - ArraySignalConditioner, [304](#)
  - BeamformerFilter, [306](#)
  - BeidouB1iPcpsAcquisition, [343](#)
  - BeidouB1iTelemetryDecoder, [347](#)
  - BeidouB3iPcpsAcquisition, [350](#)
  - BeidouB3iTelemetryDecoder, [354](#)
  - ByteToShort, [356](#)
  - Channel, [359](#)
  - CustomUDPSignalSource, [385](#)
  - DirectResamplerConditioner, [389](#)
  - FileSignalSource, [402](#)
  - FirFilter, [404](#)
  - FlexibandSignalSource, [405](#)
  - Fmcomms2SignalSource, [406](#)
  - FreqXlatingFirFilter, [418](#)
  - GalileoE1BTelemetryDecoder, [463](#)
  - GalileoE1DIIPIIVemITracking, [464](#)
  - GalileoE1DIIPIIVemITrackingFpga, [467](#)
  - GalileoE1Pcps8msAmbiguousAcquisition, [470](#)
  - GalileoE1PcpsAmbiguousAcquisition, [474](#)
  - GalileoE1PcpsAmbiguousAcquisitionFpga, [480](#)
  - GalileoE1PcpsCccwsrAmbiguousAcquisition, [485](#)
  - GalileoE1PcpsQuickSyncAmbiguousAcquisition, [489](#)
  - GalileoE1PcpsTongAmbiguousAcquisition, [493](#)
  - GalileoE1TcpConnectorTracking, [497](#)
  - GalileoE5aDIIPIITracking, [499](#)
  - GalileoE5aDIIPIITrackingFpga, [503](#)
  - GalileoE5aNoncoherentIQAcquisitionCaf, [506](#)
  - GalileoE5aPcpsAcquisitionFpga, [516](#)
  - GalileoE5aTelemetryDecoder, [521](#)
  - GalileoE5bDIIPIITracking, [523](#)
  - GalileoE5bPcpsAcquisition, [527](#)
  - GalileoE5bPcpsAcquisitionFpga, [533](#)
  - GalileoE5bTelemetryDecoder, [539](#)
  - GalileoE6DIIPIITracking, [541](#)
  - GalileoE6PcpsAcquisition, [543](#)
  - GalileoE6TelemetryDecoder, [548](#)
  - GenSignalSource, [550](#)
  - GlonassL1CaDIIPIICAidTracking, [589](#)
  - GlonassL1CaDIIPIITracking, [591](#)
  - GlonassL1CaPcpsAcquisition, [593](#)
  - GlonassL1CaTelemetryDecoder, [597](#)
  - GlonassL2CaDIIPIICAidTracking, [598](#)
  - GlonassL2CaDIIPIITracking, [600](#)
  - GlonassL2CaPcpsAcquisition, [602](#)
  - GlonassL2CaTelemetryDecoder, [606](#)
  - Gn3sSignalSource, [607](#)
  - GnMaxSignalSource, [608](#)
  - GpsL1CaDIIPIITracking, [711](#)
  - GpsL1CaDIIPIITrackingFpga, [714](#)
  - GpsL1CaDIIPIITrackingGPU, [717](#)
  - GpsL1CaKfTracking, [719](#)
  - GpsL1CaPcpsAcquisition, [721](#)
  - GpsL1CaPcpsAcquisitionFineDoppler, [725](#)
  - GpsL1CaPcpsAcquisitionFpga, [731](#)
  - GpsL1CaPcpsAssistedAcquisition, [736](#)
  - GpsL1CaPcpsOpenCIAcquisition, [739](#)
  - GpsL1CaPcpsQuickSyncAcquisition, [743](#)
  - GpsL1CaPcpsTongAcquisition, [748](#)
  - GpsL1CaTcpConnectorTracking, [751](#)
  - GpsL1CaTelemetryDecoder, [753](#)
  - GpsL2CTelemetryDecoder, [754](#)
  - GpsL2MDIIPIITracking, [755](#)
  - GpsL2MDIIPIITrackingFpga, [757](#)
  - GpsL2MPcpsAcquisition, [760](#)
  - GpsL2MPcpsAcquisitionFpga, [764](#)
  - GpsL5DIIPIITracking, [768](#)
  - GpsL5DIIPIITrackingFpga, [772](#)
  - GpsL5TelemetryDecoder, [785](#)
  - GpsL5iPcpsAcquisition, [775](#)
  - GpsL5iPcpsAcquisitionFpga, [781](#)
  - HybridObservables, [790](#)
  - lbyteToCbyte, [791](#)
  - lbyteToComplex, [792](#)
  - lbyteToCshort, [793](#)
  - lshortToComplex, [799](#)
  - lshortToCshort, [800](#)
  - LabsatSignalSource, [802](#)
  - MultichannelFileSignalSource, [809](#)
  - NotchFilter, [813](#)
  - NotchFilterLite, [814](#)
  - NsrFileSignalSource, [816](#)

- OsmosdrSignalSource, 821
- Pass\_Through, 822
- PlutosdrSignalSource, 866
- PulseBlankingFilter, 870
- RawArraySignalSource, 879
- Rtklib\_Pvt, 904
- RtlTcpSignalSource, 914
- SbasL1TelemetryDecoder, 920
- SignalConditioner, 933
- SignalGenerator, 934
- SpirFileSignalSource, 939
- TwoBitCpxFileSignalSource, 959
- TwoBitPackedFileSignalSource, 960
- UhdSignalSource, 962
- in\_memory\_configuration.h, 1155
- InMemoryConfiguration, 795
- ini.h, 1155
- ini\_parse
  - core\_libs, 196
- init
  - BeidouB1iPcpsAcquisition, 343
  - BeidouB3iPcpsAcquisition, 350
  - cnav\_v27\_part\_t, 370
  - galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc, 429
  - galileo\_pcps\_8ms\_acquisition\_cc, 454
  - GalileoE1Pcps8msAmbiguousAcquisition, 471
  - GalileoE1PcpsAmbiguousAcquisition, 475
  - GalileoE1PcpsAmbiguousAcquisitionFpga, 480
  - GalileoE1PcpsCccwsrAmbiguousAcquisition, 485
  - GalileoE1PcpsQuickSyncAmbiguousAcquisition, 489
  - GalileoE1PcpsTongAmbiguousAcquisition, 494
  - GalileoE5aNoncoherentIQAcquisitionCaf, 506
  - GalileoE5aPcpsAcquisition, 510
  - GalileoE5aPcpsAcquisitionFpga, 516
  - GalileoE5bPcpsAcquisition, 527
  - GalileoE5bPcpsAcquisitionFpga, 534
  - GalileoE6PcpsAcquisition, 544
  - GlonassL1CaPcpsAcquisition, 593
  - GlonassL2CaPcpsAcquisition, 602
  - GpsL1CaPcpsAcquisition, 721
  - GpsL1CaPcpsAcquisitionFineDoppler, 726
  - GpsL1CaPcpsAcquisitionFpga, 731
  - GpsL1CaPcpsAssistedAcquisition, 736
  - GpsL1CaPcpsOpenCIAcquisition, 740
  - GpsL1CaPcpsQuickSyncAcquisition, 744
  - GpsL1CaPcpsTongAcquisition, 748
  - GpsL2MPcpsAcquisition, 760
  - GpsL2MPcpsAcquisitionFpga, 765
  - GpsL5iPcpsAcquisition, 775
  - GpsL5iPcpsAcquisitionFpga, 781
  - pcps\_acquisition, 824
  - pcps\_acquisition\_fine\_doppler\_cc, 829
  - pcps\_acquisition\_fpga, 835
  - pcps\_assisted\_acquisition\_cc, 841
  - pcps\_cccwsr\_acquisition\_cc, 845
  - pcps\_openc1\_acquisition\_cc, 850
  - pcps\_quicksync\_acquisition\_cc, 856
  - pcps\_tong\_acquisition\_cc, 861
- initialize
  - Tracking\_2nd\_DLL\_filter, 951
- initialize\_secondary\_code
  - Fpga\_Multicorrelator\_8sc, 414
- Input Filter, 85
- input\_filter\_adapters, 86
- input\_filter\_gr\_blocks, 87
- interleaved\_byte\_to\_complex\_byte, 796
- interleaved\_byte\_to\_complex\_byte.h, 1157
- interleaved\_byte\_to\_complex\_short, 797
- interleaved\_byte\_to\_complex\_short.h, 1157
- interleaved\_short\_to\_complex\_short, 797
- interleaved\_short\_to\_complex\_short.h, 1158
- interp\_TOW\_ms
  - Gnss\_Synchro, 632
- invert
  - cnav\_v27\_part\_t, 370
- is\_rinex\_header\_written
  - Rinex\_Printer, 881
- is\_server\_running
  - Rtcm, 889
- ishort\_to\_complex.h, 1159
- ishort\_to\_cshort.h, 1159
- IshortToComplex, 798
  - implementation, 799
- IshortToCshort, 799
  - implementation, 800
- item\_size
  - GalileoE1DIIPIIVemlTrackingFpga, 467
  - GalileoE1PcpsAmbiguousAcquisitionFpga, 481
  - GalileoE5aDIIPIITrackingFpga, 503
  - GalileoE5aPcpsAcquisitionFpga, 516
  - GalileoE5bPcpsAcquisition, 527
  - GalileoE5bPcpsAcquisitionFpga, 534
  - GpsL1CaDIIPIITrackingFpga, 714
  - GpsL1CaPcpsAcquisitionFpga, 731
  - GpsL5DIIPIITrackingFpga, 772
  - GpsL5iPcpsAcquisitionFpga, 781
  - HybridObservables, 790
  - Rtklib\_Pvt, 904
- item\_type\_helpers.h, 1160
- item\_type\_is\_complex
  - algorithms\_libs, 107
- item\_type\_size
  - algorithms\_libs, 108
- item\_type\_valid
  - algorithms\_libs, 108
- kernel\_info\_t, 800
- Kml\_Printer, 801
- kml\_printer.h, 1161
- LAM\_CARR
  - algorithms\_libs\_rtklib, 137
- labsat23\_source, 801
- labsat23\_source.h, 1161
- labsat\_signal\_source.h, 1162

- LabsatSignalSource, [802](#)
  - implementation, [802](#)
- lex\_t, [803](#)
- lexeph\_t, [803](#)
- lexion\_t, [804](#)
- lexmsg\_t, [804](#)
- load\_cnav\_ephemeris\_xml
  - Gnss\_Sdr\_Supl\_Client, [616](#)
- load\_cnav\_utc\_xml
  - Gnss\_Sdr\_Supl\_Client, [617](#)
- load\_ephemeris\_xml
  - Gnss\_Sdr\_Supl\_Client, [617](#)
- load\_gal\_almanac\_xml
  - Gnss\_Sdr\_Supl\_Client, [617](#)
- load\_gal\_ephemeris\_xml
  - Gnss\_Sdr\_Supl\_Client, [617](#)
- load\_gal\_iono\_xml
  - Gnss\_Sdr\_Supl\_Client, [617](#)
- load\_gal\_utc\_xml
  - Gnss\_Sdr\_Supl\_Client, [617](#)
- load\_glo\_utc\_xml
  - Gnss\_Sdr\_Supl\_Client, [618](#)
- load\_gnav\_ephemeris\_xml
  - Gnss\_Sdr\_Supl\_Client, [618](#)
- load\_gps\_almanac\_xml
  - Gnss\_Sdr\_Supl\_Client, [618](#)
- load\_iono\_xml
  - Gnss\_Sdr\_Supl\_Client, [618](#)
- load\_ref\_location\_xml
  - Gnss\_Sdr\_Supl\_Client, [618](#)
- load\_ref\_time\_xml
  - Gnss\_Sdr\_Supl\_Client, [618](#)
- load\_utc\_xml
  - Gnss\_Sdr\_Supl\_Client, [619](#)
- lock\_channel
  - Fpga\_Multicorrelator\_8sc, [414](#)
- lock\_detectors.h, [1163](#)
- lock\_time
  - Rtcm, [889](#), [890](#)
  - Rtcm\_Printer, [901](#)
- log\_rinex\_nav\_bds\_dnav
  - Rinex\_Printer, [881](#)
- log\_rinex\_nav\_gal\_nav
  - Rinex\_Printer, [881](#)
- log\_rinex\_nav\_glo\_gnav
  - Rinex\_Printer, [881](#)
- log\_rinex\_nav\_gps\_cnav
  - Rinex\_Printer, [882](#)
- log\_rinex\_nav\_gps\_nav
  - Rinex\_Printer, [882](#)
- M0\_1
  - Galileo\_Ephemeris, [443](#)
- MATH\_CONSTANTS.h, [1163](#)
- MAX\_TOA\_DELAY\_MS
  - core\_system\_parameters, [278](#)
- MAXANT
  - algorithms\_libs\_rtklib, [137](#)
- MAXBAND
  - algorithms\_libs\_rtklib, [138](#)
- MAXCLI
  - algorithms\_libs\_rtklib, [138](#)
- MAXCODE
  - core\_system\_parameters, [278](#)
- MAXDTOE\_BDS
  - algorithms\_libs\_rtklib, [138](#)
- MAXDTOE\_GAL
  - algorithms\_libs\_rtklib, [138](#)
- MAXDTOE\_GLO
  - algorithms\_libs\_rtklib, [139](#)
- MAXDTOE\_QZS
  - algorithms\_libs\_rtklib, [139](#)
- MAXDTOE\_SBS
  - algorithms\_libs\_rtklib, [139](#)
- MAXDTOE\_S
  - algorithms\_libs\_rtklib, [139](#)
- MAXDTOE
  - algorithms\_libs\_rtklib, [138](#)
- MAXERRMSG
  - algorithms\_libs\_rtklib, [139](#)
- MAXEXFILE
  - algorithms\_libs\_rtklib, [140](#)
- MAXFREQ
  - algorithms\_libs\_rtklib, [140](#)
- MAXGDOP
  - algorithms\_libs\_rtklib, [140](#)
- MAXITR
  - rtklib\_pntpos.h, [1212](#)
- MAXLEAPS
  - algorithms\_libs\_rtklib, [140](#)
- MAXNGEO
  - algorithms\_libs\_rtklib, [140](#)
- MAXNIGP
  - algorithms\_libs\_rtklib, [141](#)
- MAXOBSBUF
  - algorithms\_libs\_rtklib, [141](#)
- MAXOBSTYPE
  - algorithms\_libs\_rtklib, [141](#)
- MAXOBS
  - algorithms\_libs\_rtklib, [141](#)
- MAXPRNBDS
  - algorithms\_libs\_rtklib, [141](#)
- MAXPRNGAL
  - algorithms\_libs\_rtklib, [142](#)
- MAXPRNGLO
  - algorithms\_libs\_rtklib, [142](#)
- MAXPRNGPS
  - algorithms\_libs\_rtklib, [142](#)
- MAXPRNSBS
  - algorithms\_libs\_rtklib, [142](#)
- MAXRAWLEN
  - algorithms\_libs\_rtklib, [142](#)
- MAXRCV
  - algorithms\_libs\_rtklib, [143](#)
- MAXSBSAGEF
  - algorithms\_libs\_rtklib, [143](#)
- MAXSBSAGEL

- algorithms\_libs\_rtklib, [143](#)
- MAXSBSMSG
  - algorithms\_libs\_rtklib, [143](#)
- MAXSBSURA
  - algorithms\_libs\_rtklib, [143](#)
- MAXSOLBUF
  - algorithms\_libs\_rtklib, [144](#)
- MAXSOLMSG
  - algorithms\_libs\_rtklib, [144](#)
- MAXSOLQ
  - algorithms\_libs\_rtklib, [144](#)
- MAXSTATMSG
  - algorithms\_libs\_rtklib, [144](#)
- MAXSTRMSG
  - algorithms\_libs\_rtklib, [144](#)
- MAXSTRPATH
  - algorithms\_libs\_rtklib, [145](#)
- MINPRNBDS
  - algorithms\_libs\_rtklib, [145](#)
- MINPRNGAL
  - algorithms\_libs\_rtklib, [145](#)
- MINPRNGLO
  - algorithms\_libs\_rtklib, [145](#)
- MINPRNGPS
  - algorithms\_libs\_rtklib, [145](#)
- MINPRNSBS
  - algorithms\_libs\_rtklib, [146](#)
- mag
  - BeidouB1iPcpsAcquisition, [343](#)
  - BeidouB3iPcpsAcquisition, [350](#)
  - galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc, [429](#)
  - galileo\_pcps\_8ms\_acquisition\_cc, [454](#)
  - GalileoE1Pcps8msAmbiguousAcquisition, [471](#)
  - GalileoE1PcpsAmbiguousAcquisition, [475](#)
  - GalileoE1PcpsAmbiguousAcquisitionFpga, [481](#)
  - GalileoE1PcpsCccwsrAmbiguousAcquisition, [485](#)
  - GalileoE1PcpsQuickSyncAmbiguousAcquisition, [489](#)
  - GalileoE1PcpsTongAmbiguousAcquisition, [494](#)
  - GalileoE5aNoncoherentIQAcquisitionCaf, [506](#)
  - GalileoE5aPcpsAcquisition, [510](#)
  - GalileoE5aPcpsAcquisitionFpga, [517](#)
  - GalileoE5bPcpsAcquisition, [527](#)
  - GalileoE5bPcpsAcquisitionFpga, [534](#)
  - GalileoE6PcpsAcquisition, [544](#)
  - GlonassL1CaPcpsAcquisition, [593](#)
  - GlonassL2CaPcpsAcquisition, [602](#)
  - GpsL1CaPcpsAcquisition, [721](#)
  - GpsL1CaPcpsAcquisitionFineDoppler, [726](#)
  - GpsL1CaPcpsAcquisitionFpga, [732](#)
  - GpsL1CaPcpsAssistedAcquisition, [736](#)
  - GpsL1CaPcpsOpenCIAcquisition, [740](#)
  - GpsL1CaPcpsQuickSyncAcquisition, [744](#)
  - GpsL1CaPcpsTongAcquisition, [748](#)
  - GpsL2MPcpsAcquisition, [760](#)
  - GpsL2MPcpsAcquisitionFpga, [765](#)
  - GpsL5iPcpsAcquisition, [775](#)
  - GpsL5iPcpsAcquisitionFpga, [782](#)
  - pcps\_acquisition, [824](#)
  - pcps\_acquisition\_fine\_doppler\_cc, [829](#)
  - pcps\_acquisition\_fpga, [835](#)
  - pcps\_assisted\_acquisition\_cc, [841](#)
  - pcps\_cccwsr\_acquisition\_cc, [845](#)
  - pcps\_openc1\_acquisition\_cc, [850](#)
  - pcps\_quicksync\_acquisition\_cc, [856](#)
  - pcps\_tong\_acquisition\_cc, [861](#)
- make\_vector\_converter
  - algorithms\_libs, [108](#)
- message\_lock
  - cnav\_v27\_part\_t, [371](#)
- mmse\_resampler\_conditioner.h, [1167](#)
- MmseResamplerConditioner, [805](#)
- ModelFunction, [805](#)
- Monitor\_Pvt, [806](#)
  - serialize, [807](#)
- Monitor\_Pvt\_Udp\_Sink, [807](#)
- monitor\_pvt.h, [1167](#)
- monitor\_pvt\_udp\_sink.h, [1168](#)
- msg\_id
  - cnav\_msg\_t, [368](#)
- msm\_h\_t, [807](#)
- mt1\_header, [808](#)
- multichannel\_file\_signal\_source.h, [1169](#)
- MultichannelFileSignalSource, [808](#)
  - implementation, [809](#)
- n\_crc\_fail
  - cnav\_v27\_part\_t, [371](#)
- n\_decoded
  - cnav\_v27\_part\_t, [371](#)
- n\_symbols
  - cnav\_v27\_part\_t, [371](#)
- NEXOBS
  - algorithms\_libs\_rtklib, [146](#)
- NFREQGLO
  - algorithms\_libs\_rtklib, [146](#)
- NFREQ
  - algorithms\_libs\_rtklib, [146](#)
- NSATBDS
  - algorithms\_libs\_rtklib, [146](#)
- NSATGAL
  - algorithms\_libs\_rtklib, [147](#)
- NSATGLO
  - algorithms\_libs\_rtklib, [147](#)
- NSATGPS
  - algorithms\_libs\_rtklib, [147](#)
- NSATSBS
  - algorithms\_libs\_rtklib, [147](#)
- NSYS
  - algorithms\_libs\_rtklib, [147](#)
- nav\_t, [810](#)
- nextPowerOf2
  - pcps\_acquisition\_fine\_doppler\_cc, [829](#)
- Nmea\_Printer, [811](#)
  - ~Nmea\_Printer, [811](#)
  - Nmea\_Printer, [811](#)

- Print\_Nmea\_Line, 812
- nmea\_printer.h, 1169
- nonlinear\_tracking.h, 1170
- Notch, 812
- notch\_cc.h, 1171
- notch\_filter.h, 1171
- notch\_filter\_lite.h, 1172
- notch\_lite\_cc.h, 1173
- NotchFilter, 813
  - implementation, 813
- NotchFilterLite, 814
  - implementation, 814
- NotchLite, 815
- nsc\_enc\_bit
  - telemetry\_decoder\_libs, 179
- nsc\_transit
  - telemetry\_decoder\_libs, 179
- nsr\_file\_signal\_source.h, 1173
- NsrFileSignalSource, 816
  - implementation, 816
- ntrip\_t, 817
- NX
  - rtklib\_pntpos.h, 1212
- OMEGA\_0\_2
  - Galileo\_Ephemeris, 443
- OMEGA\_dot\_3
  - Galileo\_Ephemeris, 443
- Obs\_Conf, 817
- obs\_adapters, 160
- obs\_conf.h, 1174
- obs\_gr\_blocks, 161
- obs\_t, 818
- obsd\_t, 818
- obsdiff\_flags.h, 1175
- observable\_tests\_flags.h, 1175
- Observables, 159
- Observables\_Dump\_Reader, 818
- observables\_dump\_reader.h, 1176
- observables\_interface.h, 1177
- observables\_libs, 162
- ObservablesInterface, 819
- omega\_2
  - Galileo\_Ephemeris, 443
- open\_channel
  - Fpga\_Multicorrelator\_8sc, 414
- open\_device
  - Fpga\_Acquisition, 408
- operator<<
  - Gnss\_Satellite, 613
  - Gnss\_Signal, 624
  - Gnuplot, 645
- operator=
  - Acquisition\_Dump\_Reader, 295
  - Exponential\_Smoother, 398
  - Gnss\_Satellite, 612
  - Gnss\_Synchro, 628
  - Serdes\_Gnss\_Synchro, 926
  - Serdes\_Monitor\_Pvt, 929
  - Tracking\_loop\_filter, 956
- operator==
  - Gnss\_Satellite, 613
  - Gnss\_Signal, 625
- opt\_t, 820
- osmosdr\_signal\_source.h, 1177
- OsmosdrSignalSource, 820
  - implementation, 821
- PI\_TWO\_N19
  - core\_system\_parameters, 279
- PI\_TWO\_N23
  - core\_system\_parameters, 279
- PI\_TWO\_N31
  - core\_system\_parameters, 279
- PI\_TWO\_N38
  - core\_system\_parameters, 279
- PI\_TWO\_N43
  - core\_system\_parameters, 279
- PMODE\_DGPS
  - algorithms\_libs\_rtklib, 148
- PMODE\_FIXED
  - algorithms\_libs\_rtklib, 148
- PMODE\_KINEMA
  - algorithms\_libs\_rtklib, 148
- PMODE\_MOVEB
  - algorithms\_libs\_rtklib, 148
- PMODE\_PPP\_FIXED
  - algorithms\_libs\_rtklib, 148
- PMODE\_PPP\_KINEMA
  - algorithms\_libs\_rtklib, 149
- PMODE\_PPP\_STATIC
  - algorithms\_libs\_rtklib, 149
- PMODE\_SINGLE
  - algorithms\_libs\_rtklib, 149
- PMODE\_STATIC
  - algorithms\_libs\_rtklib, 149
- POLYCRC24Q
  - algorithms\_libs\_rtklib, 149
- POLYCRC32
  - algorithms\_libs\_rtklib, 150
- POSOPT\_RINEX
  - algorithms\_libs\_rtklib, 150
- PRCOPT\_DEFAULT
  - rtklib\_rtksvr.h, 1233
- PRN\_HWBIAS
  - algorithms\_libs\_rtklib, 150
- PRN
  - Gnss\_Synchro, 632
- PVT, 117
- parity\_counter
  - telemetry\_decoder\_libs, 180
- ParseError
  - INIReader, 795
- part1
  - cnav\_msg\_decoder\_t, 367
- part2
  - cnav\_msg\_decoder\_t, 367
- Pass\_Through, 821

- implementation, 822
- pass\_through.h, 1178
- pclk\_t, 822
- pcps\_acquisition, 823
  - general\_work, 824
  - init, 824
  - mag, 824
  - set\_active, 824
  - set\_channel, 825
  - set\_channel\_fsm, 825
  - set\_doppler\_center, 825
  - set\_doppler\_max, 825
  - set\_doppler\_step, 826
  - set\_gnss\_synchro, 826
  - set\_local\_code, 826
  - set\_state, 827
  - set\_threshold, 827
- pcps\_acquisition.h, 1179
- pcps\_acquisition\_fine\_doppler\_cc, 827
  - ~pcps\_acquisition\_fine\_doppler\_cc, 828
  - general\_work, 829
  - init, 829
  - mag, 829
  - nextPowerOf2, 829
  - set\_active, 830
  - set\_channel, 830
  - set\_channel\_fsm, 830
  - set\_doppler\_max, 830
  - set\_doppler\_step, 832
  - set\_gnss\_synchro, 832
  - set\_local\_code, 832
  - set\_state, 833
  - set\_threshold, 833
- pcps\_acquisition\_fine\_doppler\_cc.h, 1180
- pcps\_acquisition\_fpga, 833
  - ~pcps\_acquisition\_fpga, 834
  - init, 835
  - mag, 835
  - reset\_acquisition, 835
  - set\_active, 835
  - set\_channel, 835
  - set\_channel\_fsm, 837
  - set\_doppler\_center, 837
  - set\_doppler\_max, 837
  - set\_doppler\_step, 837
  - set\_gnss\_synchro, 838
  - set\_local\_code, 838
  - set\_state, 838
  - set\_threshold, 839
- pcps\_acquisition\_fpga.h, 1181
- pcps\_assisted\_acquisition\_cc, 839
  - ~pcps\_assisted\_acquisition\_cc, 840
  - general\_work, 840
  - init, 841
  - mag, 841
  - set\_active, 841
  - set\_channel, 841
  - set\_channel\_fsm, 842
  - set\_doppler\_max, 842
  - set\_doppler\_step, 842
  - set\_gnss\_synchro, 842
  - set\_local\_code, 843
  - set\_threshold, 843
- pcps\_assisted\_acquisition\_cc.h, 1182
- pcps\_cccwsr\_acquisition\_cc, 844
  - ~pcps\_cccwsr\_acquisition\_cc, 845
  - general\_work, 845
  - init, 845
  - mag, 845
  - set\_active, 845
  - set\_channel, 846
  - set\_channel\_fsm, 846
  - set\_doppler\_max, 846
  - set\_doppler\_step, 846
  - set\_gnss\_synchro, 847
  - set\_local\_code, 847
  - set\_state, 847
  - set\_threshold, 848
- pcps\_cccwsr\_acquisition\_cc.h, 1183
- pcps\_opencl\_acquisition\_cc, 848
  - ~pcps\_opencl\_acquisition\_cc, 849
  - general\_work, 850
  - init, 850
  - mag, 850
  - set\_active, 850
  - set\_channel, 851
  - set\_channel\_fsm, 851
  - set\_doppler\_max, 851
  - set\_doppler\_step, 851
  - set\_gnss\_synchro, 852
  - set\_local\_code, 852
  - set\_state, 852
  - set\_threshold, 854
- pcps\_opencl\_acquisition\_cc.h, 1184
- pcps\_quicksync\_acquisition\_cc, 854
  - ~pcps\_quicksync\_acquisition\_cc, 855
  - general\_work, 856
  - init, 856
  - mag, 856
  - set\_active, 856
  - set\_channel, 856
  - set\_channel\_fsm, 857
  - set\_doppler\_max, 857
  - set\_doppler\_step, 857
  - set\_gnss\_synchro, 858
  - set\_local\_code, 858
  - set\_state, 858
  - set\_threshold, 859
- pcps\_quicksync\_acquisition\_cc.h, 1185
- pcps\_tong\_acquisition\_cc, 859
  - ~pcps\_tong\_acquisition\_cc, 860
  - general\_work, 860
  - init, 861
  - mag, 861
  - set\_active, 861
  - set\_channel, 861

- set\_channel\_fsm, 862
- set\_doppler\_max, 862
- set\_doppler\_step, 862
- set\_gnss\_synchro, 862
- set\_local\_code, 863
- set\_state, 863
- set\_threshold, 863
- pcps\_tong\_acquisition\_cc.h, 1187
- pcpsconf\_fpga\_t, 864
- pcv\_t, 864
- pcvs\_t, 865
- peph\_t, 865
- phase\_unwrap
  - tracking\_libs, 192
- pll\_cloop\_two\_quadrant\_atan
  - tracking\_libs, 192
- pll\_four\_quadrant\_atan
  - tracking\_libs, 192
- plutosdr\_signal\_source.h, 1188
- PlutosdrSignalSource, 866
  - implementation, 866
- pntpos
  - rtklib\_pntpos.h, 1211
- pop\_front
  - algorithms\_libs, 109
- position\_test\_flags.h, 1189
- pppcorr\_t, 867
- prcopt\_t, 867
- preamble\_seen
  - cnav\_v27\_part\_t, 371
- print\_MSM\_1
  - Rtcm, 890
- print\_MSM\_2
  - Rtcm, 890
- print\_MSM\_3
  - Rtcm, 891
- print\_MSM\_4
  - Rtcm, 891
- print\_MSM\_5
  - Rtcm, 891
- print\_MSM\_6
  - Rtcm, 892
- print\_MSM\_7
  - Rtcm, 892
- print\_MT1001
  - Rtcm, 892
- 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, 896
- print\_MT1012
  - Rtcm, 896
- print\_MT1019
  - Rtcm, 897
- print\_MT1020
  - Rtcm, 897
- print\_MT1029
  - Rtcm, 897
- print\_MT1045
  - Rtcm, 898
- Print\_Nmea\_Line
  - Nmea\_Printer, 812
- Print\_Rtcm\_Messages
  - Rtcm\_Printer, 901
- print\_rinex\_annotation
  - Rinex\_Printer, 882
- priorize\_satellites
  - GNSSFlowgraph, 642
- prn
  - cnav\_msg\_t, 368
- Prompt\_I
  - Gnss\_Synchro, 633
- Prompt\_Q
  - Gnss\_Synchro, 633
- Pseudorange\_m
  - Gnss\_Synchro, 633
- pulse\_blanking\_cc, 868
- pulse\_blanking\_cc.h, 1190
- pulse\_blanking\_filter.h, 1190
- PulseBlankingFilter, 869
  - implementation, 870
- push\_back
  - algorithms\_libs, 109
- pv\_Geo\_to\_ECEF
  - algorithms\_libs, 109
- Pvt\_Conf, 870
- Pvt\_Solution, 871
  - get\_avg\_height, 873
  - get\_avg\_latitude, 873
  - get\_avg\_longitude, 873
  - get\_clock\_drift\_ppm, 873
  - get\_course\_over\_ground, 873
  - get\_height, 873
  - get\_latitude, 874
  - get\_longitude, 874
  - get\_num\_valid\_observations, 874
  - get\_speed\_over\_ground, 874
  - get\_time\_offset\_s, 874
  - set\_averaging\_depth, 874

- set\_clock\_drift\_ppm, 875
  - set\_course\_over\_ground, 875
  - set\_num\_valid\_observations, 875
  - set\_pre\_2009\_file, 875
  - set\_rx\_pos, 875
  - set\_rx\_vel, 875
  - set\_speed\_over\_ground, 876
  - set\_time\_offset\_s, 876
- pvt\_adapters, 163
- pvt\_conf.h, 1191
- pvt\_gr\_blocks, 164
- pvt\_interface.h, 1191
- pvt\_libs, 165
- pvt\_solution.h, 1192
- PvtInterface, 876
- R2D
  - core\_system\_parameters, 280
- RE\_WGS84
  - algorithms\_libs\_rtklib, 150
- REL\_HUMI
  - algorithms\_libs\_rtklib, 150
- RTL\_TCP\_COMMAND
  - signal\_source\_libs, 173
- RX\_time
  - Gnss\_Synchro, 633
- raw\_array\_signal\_source.h, 1193
- raw\_msg
  - cnav\_msg\_t, 368
- raw\_t, 877
- RawArraySignalSource, 878
  - implementation, 879
- read\_MT1005
  - Rtcm, 898
- read\_MT1019
  - Rtcm, 898
- read\_MT1020
  - Rtcm, 898
- read\_MT1045
  - Rtcm, 899
- read\_acquisition\_results
  - Fpga\_Acquisition, 408
- read\_fpga\_total\_scale\_factor
  - Fpga\_Acquisition, 408
- read\_sample\_counter
  - Fpga\_Multicorrelator\_8sc, 414
- readProtobuffer
  - Serdes\_Gnss\_Synchro, 926
  - Serdes\_Monitor\_Pvt, 929
- Region1\_flag\_5
  - Galileo\_Iono, 450
- Region2\_flag\_5
  - Galileo\_Iono, 450
- Region3\_flag\_5
  - Galileo\_Iono, 451
- Region4\_flag\_5
  - Galileo\_Iono, 451
- Region5\_flag\_5
  - Galileo\_Iono, 451
- replot
  - Gnuplot, 645
- Resampler, 166
- resampler
  - algorithms\_libs, 109, 110
- resampler\_adapters, 167
- resampler\_gr\_blocks, 168
- reset
  - algorithms\_libs, 110
  - BeidouB1iPcpsAcquisition, 344
  - BeidouB3iPcpsAcquisition, 351
  - dll\_pll\_veml\_tracking\_fpga, 394
  - GalileoE1Pcps8msAmbiguousAcquisition, 471
  - GalileoE1PcpsAmbiguousAcquisition, 475
  - GalileoE1PcpsAmbiguousAcquisitionFpga, 481
  - GalileoE1PcpsCccwsrAmbiguousAcquisition, 486
  - GalileoE1PcpsQuickSyncAmbiguousAcquisition, 490
  - GalileoE1PcpsTongAmbiguousAcquisition, 494
  - GalileoE5aNoncoherentIQAcquisitionCaf, 506
  - GalileoE5aPcpsAcquisition, 510
  - GalileoE5aPcpsAcquisitionFpga, 517
  - GalileoE5bPcpsAcquisition, 527
  - GalileoE5bPcpsAcquisitionFpga, 534
  - GalileoE6PcpsAcquisition, 544
  - GlonassL1CaPcpsAcquisition, 594
  - GlonassL2CaPcpsAcquisition, 603
  - GpsL1CaPcpsAcquisition, 722
  - GpsL1CaPcpsAcquisitionFineDoppler, 726
  - GpsL1CaPcpsAcquisitionFpga, 732
  - GpsL1CaPcpsAssistedAcquisition, 736
  - GpsL1CaPcpsOpenCIAcquisition, 740
  - GpsL1CaPcpsQuickSyncAcquisition, 744
  - GpsL1CaPcpsTongAcquisition, 748
  - GpsL2MPcpsAcquisition, 760
  - GpsL2MPcpsAcquisitionFpga, 765
  - GpsL5iPcpsAcquisition, 775
  - GpsL5iPcpsAcquisitionFpga, 782
- reset\_acquisition
  - Fpga\_Acquisition, 408
  - pcps\_acquisition\_fpga, 835
- Rinex\_Printer, 879
  - ~Rinex\_Printer, 880
  - get\_navfilename, 880
  - get\_obsfilename, 881
  - is\_rinex\_header\_written, 881
  - log\_rinex\_nav\_bds\_dnav, 881
  - log\_rinex\_nav\_gal\_nav, 881
  - log\_rinex\_nav\_glo\_gnav, 881
  - log\_rinex\_nav\_gps\_cnav, 882
  - log\_rinex\_nav\_gps\_nav, 882
  - print\_rinex\_annotation, 882
  - Rinex\_Printer, 880
  - set\_pre\_2009\_file, 884
- rinex\_printer.h, 1193
- role
  - GalileoE1DIIPIVemlTrackingFpga, 468
  - GalileoE1PcpsAmbiguousAcquisitionFpga, 481

- GalileoE5aDIIPIITrackingFpga, [503](#)
- GalileoE5aPcpsAcquisitionFpga, [517](#)
- GalileoE5bPcpsAcquisition, [528](#)
- GalileoE5bPcpsAcquisitionFpga, [534](#)
- GpsL1CaDIIPIITrackingFpga, [715](#)
- GpsL1CaPcpsAcquisitionFpga, [732](#)
- GpsL5DIIPIITrackingFpga, [772](#)
- GpsL5iPcpsAcquisitionFpga, [782](#)
- Rtcm, [884](#)
  - bin\_to\_binary\_data, [887](#)
  - bin\_to\_double, [887](#)
  - 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\_MSM\_1, [890](#)
  - print\_MSM\_2, [890](#)
  - print\_MSM\_3, [891](#)
  - print\_MSM\_4, [891](#)
  - print\_MSM\_5, [891](#)
  - print\_MSM\_6, [892](#)
  - print\_MSM\_7, [892](#)
  - print\_MT1001, [892](#)
  - 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, [896](#)
  - print\_MT1012, [896](#)
  - print\_MT1019, [897](#)
  - print\_MT1020, [897](#)
  - print\_MT1029, [897](#)
  - print\_MT1045, [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.h, [1194](#)
- Rtcm\_Printer, [900](#)
  - ~Rtcm\_Printer, [900](#)
  - lock\_time, [901](#)
  - print\_MT1005\_test, [901](#)
  - Print\_Rtcm\_Messages, [901](#)
  - Rtcm\_Printer, [900](#)
- rtcm\_printer.h, [1195](#)
- rtcm\_t, [902](#)
- rtk\_t, [903](#)
- rtklib.h, [1196](#)
- Rtklib\_Pvt, [903](#)
  - implementation, [904](#)
  - item\_size, [904](#)
- Rtklib\_Solver, [908](#)
  - beidou\_dnav\_ephemeris\_map, [909](#)
  - galileo\_ephemeris\_map, [909](#)
  - glonass\_gnav\_almanac, [909](#)
  - glonass\_gnav\_ephemeris\_map, [910](#)
  - glonass\_gnav\_utc\_model, [910](#)
  - gps\_cnav\_ephemeris\_map, [910](#)
  - gps\_ephemeris\_map, [910](#)
- Rtklib\_Solver\_Dump\_Reader, [911](#)
- rtklib\_conversions.h, [1205](#)
- rtklib\_ephemeris.h, [1206](#)
- rtklib\_ionex.h, [1207](#)
- rtklib\_lambda.h, [1208](#)
  - SWAP\_LAMBDA, [1209](#)
- rtklib\_pntpos.h, [1210](#)
  - ERR\_ION, [1212](#)
  - ERR\_TROP, [1212](#)
  - MAXITR, [1212](#)
  - NX, [1212](#)
  - pntpos, [1211](#)
- rtklib\_ppp.h, [1213](#)
  - SWAP\_D, [1215](#)
  - SWAP\_I, [1215](#)
- rtklib\_preceph.h, [1215](#)
- rtklib\_pvt.h, [1217](#)
- rtklib\_pvt\_gs, [905](#)
  - ~rtklib\_pvt\_gs, [906](#)
  - clear\_ephemeris, [906](#)
  - get\_beidou\_dnav\_almanac\_map, [906](#)
  - get\_beidou\_dnav\_ephemeris\_map, [906](#)
  - get\_galileo\_almanac\_map, [906](#)
  - get\_galileo\_ephemeris\_map, [907](#)
  - get\_gps\_almanac\_map, [907](#)
  - get\_gps\_ephemeris\_map, [907](#)
  - get\_latest\_PVT, [907](#)
  - work, [907](#)
- rtklib\_pvt\_gs.h, [1217](#)
- rtklib\_rtcm.h, [1218](#)
- rtklib\_rtcm2.h, [1219](#)
- rtklib\_rtcm3.h, [1220](#)
  - CODES\_BDS, [1222](#)
  - CODES\_GAL, [1222](#)
  - CODES\_GLO, [1223](#)
  - CODES\_GPS, [1223](#)
  - CODES\_QZS, [1223](#)
  - CODES\_SBS, [1224](#)
  - SSRUDINT, [1224](#)
- rtklib\_rtkcmn.h, [1224](#)
  - Rx, [1228](#)
  - Ry, [1228](#)
  - Rz, [1229](#)

- rtklib\_rtkpos.h, [1229](#)
- rtklib\_rtksvr.h, [1231](#)
  - PRCOPT\_DEFAULT, [1233](#)
  - SOLOPT\_DEFAULT, [1233](#)
- rtklib\_sbas.h, [1233](#)
  - IGPBAND1, [1235](#)
  - IGPBAND2, [1235](#)
- rtklib\_solution.h, [1236](#)
- rtklib\_solver.h, [1238](#)
- rtklib\_solver\_dump\_reader.h, [1239](#)
- rtklib\_stream.h, [1239](#)
- rtklib\_tides.h, [1242](#)
- rtksvr\_t, [911](#)
- Rtl\_Tcp\_Dongle\_Info, [912](#)
- rtl\_tcp\_command
  - signal\_source\_libs, [173](#)
- rtl\_tcp\_commands.h, [1243](#)
- rtl\_tcp\_dongle\_info.h, [1243](#)
- rtl\_tcp\_signal\_source.h, [1244](#)
- rtl\_tcp\_signal\_source\_c, [913](#)
- rtl\_tcp\_signal\_source\_c.h, [1245](#)
- RtlTcpSignalSource, [914](#)
  - implementation, [914](#)
- run
  - ControlThread, [380](#)
- run\_acquisition
  - Fpga\_Acquisition, [409](#)
- run\_server
  - Rtcm, [899](#)
- Rx
  - rtklib\_rtkcmn.h, [1228](#)
- Ry
  - rtklib\_rtkcmn.h, [1228](#)
- Rz
  - rtklib\_rtkcmn.h, [1229](#)
- SC2RAD
  - core\_system\_parameters, [280](#)
- SERIBUFFSIZE
  - algorithms\_libs\_rtklib, [151](#)
- SOLF\_ENU
  - algorithms\_libs\_rtklib, [151](#)
- SOLF\_GSIF
  - algorithms\_libs\_rtklib, [151](#)
- SOLF\_LLH
  - algorithms\_libs\_rtklib, [151](#)
- SOLF\_NMEA
  - algorithms\_libs\_rtklib, [151](#)
- SOLF\_STAT
  - algorithms\_libs\_rtklib, [152](#)
- SOLF\_XYZ
  - algorithms\_libs\_rtklib, [152](#)
- SOLOPT\_DEFAULT
  - rtklib\_rtksvr.h, [1233](#)
- SOLQ\_DGPS
  - algorithms\_libs\_rtklib, [152](#)
- SOLQ\_DR
  - algorithms\_libs\_rtklib, [152](#)
- SOLQ\_FIX
  - algorithms\_libs\_rtklib, [152](#)
- SOLQ\_FLOAT
  - algorithms\_libs\_rtklib, [153](#)
- SOLQ\_NONE
  - algorithms\_libs\_rtklib, [153](#)
- SOLQ\_PPP
  - algorithms\_libs\_rtklib, [153](#)
- SOLQ\_SBAS
  - algorithms\_libs\_rtklib, [153](#)
- SOLQ\_SINGLE
  - algorithms\_libs\_rtklib, [153](#)
- SPEED\_OF\_LIGHT\_M\_MS
  - core\_system\_parameters, [280](#)
- SPEED\_OF\_LIGHT\_M\_S
  - core\_system\_parameters, [280](#)
- SSRUDINT
  - rtklib\_rtc3.h, [1224](#)
- SWAP\_LAMBDA
  - rtklib\_lambda.h, [1209](#)
- SWAP\_D
  - rtklib\_ppp.h, [1215](#)
- SWAP\_I
  - rtklib\_ppp.h, [1215](#)
- SYS\_ALL
  - algorithms\_libs\_rtklib, [154](#)
- SYS\_BDS
  - algorithms\_libs\_rtklib, [154](#)
- SYS\_GAL
  - algorithms\_libs\_rtklib, [154](#)
- SYS\_GLO
  - algorithms\_libs\_rtklib, [154](#)
- SYS\_GPS
  - algorithms\_libs\_rtklib, [154](#)
- SYS\_IRN
  - algorithms\_libs\_rtklib, [155](#)
- SYS\_LEO
  - algorithms\_libs\_rtklib, [155](#)
- SYS\_NONE
  - algorithms\_libs\_rtklib, [155](#)
- SYS\_QZS
  - algorithms\_libs\_rtklib, [155](#)
- SYS\_SBS
  - algorithms\_libs\_rtklib, [155](#)
- satelliteBlock
  - Beidou\_Dnav\_Ephemeris, [329](#)
  - Gps\_Ephemeris, [692](#)
- satellitePosition
  - Beidou\_Dnav\_Ephemeris, [317](#)
  - Beidou\_Dnav\_Navigation\_Message, [336](#)
  - Galileo\_Ephemeris, [435](#)
  - Gps\_CNAV\_Ephemeris, [657](#)
  - Gps\_Ephemeris, [680](#)
- save\_cnav\_ephemeris\_map\_xml
  - Gnss\_Sdr\_Supl\_Client, [619](#)
- save\_cnav\_utc\_xml
  - Gnss\_Sdr\_Supl\_Client, [619](#)
- save\_ephemeris\_map\_xml
  - Gnss\_Sdr\_Supl\_Client, [619](#)

- save\_gal\_almanac\_xml
  - Gnss\_Sdr\_Supl\_Client, [619](#)
- save\_gal\_ephemeris\_map\_xml
  - Gnss\_Sdr\_Supl\_Client, [620](#)
- save\_gal\_iono\_xml
  - Gnss\_Sdr\_Supl\_Client, [620](#)
- save\_gal\_utc\_xml
  - Gnss\_Sdr\_Supl\_Client, [620](#)
- save\_glo\_utc\_xml
  - Gnss\_Sdr\_Supl\_Client, [620](#)
- save\_gnav\_ephemeris\_map\_xml
  - Gnss\_Sdr\_Supl\_Client, [620](#)
- save\_gps\_almanac\_xml
  - Gnss\_Sdr\_Supl\_Client, [621](#)
- save\_iono\_xml
  - Gnss\_Sdr\_Supl\_Client, [621](#)
- save\_ref\_location\_xml
  - Gnss\_Sdr\_Supl\_Client, [621](#)
- save\_ref\_time\_xml
  - Gnss\_Sdr\_Supl\_Client, [621](#)
- save\_utc\_xml
  - Gnss\_Sdr\_Supl\_Client, [621](#)
- Sbas\_Ephemeris, [915](#)
  - b\_sv\_do\_not\_use, [916](#)
  - d\_acc, [916](#)
  - d\_af0, [916](#)
  - d\_af1, [916](#)
  - d\_pos, [916](#)
  - d\_tof, [917](#)
  - d\_vel, [917](#)
  - i\_prn, [917](#)
  - i\_sv\_ura, [917](#)
  - i\_t0, [917](#)
- sbas\_ephemeris.h, [1246](#)
- sbas\_l1\_telemetry\_decoder.h, [1246](#)
- sbas\_l1\_telemetry\_decoder\_gs, [918](#)
  - general\_work, [919](#)
  - set\_channel, [919](#)
  - set\_satellite, [919](#)
- sbas\_l1\_telemetry\_decoder\_gs.h, [1247](#)
- SbasL1TelemetryDecoder, [919](#)
  - implementation, [920](#)
- sbs\_t, [920](#)
- sbsfcrr\_t, [921](#)
- sbsigp\_t, [921](#)
- sbsigpband\_t, [922](#)
- sbsion\_t, [922](#)
- sbslcorr\_t, [922](#)
- sbsmsg\_t, [923](#)
- sbssat\_t, [923](#)
- sbssatp\_t, [924](#)
- send\_message
  - Rtcm, [899](#)
- send\_telemetry\_msg
  - GNSSFlowgraph, [642](#)
- seph\_t, [924](#)
- Serdes\_Gnss\_Synchro, [924](#)
  - createProtobuffer, [925](#)
  - operator=, [926](#)
  - readProtobuffer, [926](#)
  - Serdes\_Gnss\_Synchro, [925](#)
- Serdes\_Monitor\_Pvt, [927](#)
  - createProtobuffer, [928](#)
  - operator=, [929](#)
  - readProtobuffer, [929](#)
  - Serdes\_Monitor\_Pvt, [927](#)
- serdes\_gnss\_synchro.h, [1248](#)
- serdes\_monitor\_pvt.h, [1248](#)
- serial\_t, [929](#)
- serialize
  - Agnss\_Ref\_Location, [300](#)
  - Agnss\_Ref\_Time, [301](#)
  - Beidou\_Dnav\_Ephemeris, [317](#)
  - Beidou\_Dnav\_Iono, [331](#)
  - Galileo\_Ephemeris, [435](#)
  - Galileo\_Iono, [449](#)
  - Galileo\_Utc\_Model, [461](#)
  - Glonass\_Gnav\_Almanac, [553](#)
  - Glonass\_Gnav\_Ephemeris, [562](#)
  - Glonass\_Gnav\_Utc\_Model, [577](#)
  - Gnss\_Synchro, [628](#)
  - Gps\_CNAV\_Ephemeris, [657](#)
  - Gps\_CNAV\_Iono, [670](#)
  - Gps\_Ephemeris, [680](#)
  - Gps\_Iono, [694](#)
  - Monitor\_Pvt, [807](#)
- set\_DLL\_BW
  - Tracking\_2nd\_DLL\_filter, [952](#)
- set\_PLL\_BW
  - Tracking\_2nd\_PLL\_filter, [953](#)
- set\_active
  - galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc, [429](#)
  - galileo\_pcps\_8ms\_acquisition\_cc, [454](#)
  - pcps\_acquisition, [824](#)
  - pcps\_acquisition\_fine\_doppler\_cc, [830](#)
  - pcps\_acquisition\_fpga, [835](#)
  - pcps\_assisted\_acquisition\_cc, [841](#)
  - pcps\_cccwsr\_acquisition\_cc, [845](#)
  - pcps\_openc1\_acquisition\_cc, [850](#)
  - pcps\_quicksync\_acquisition\_cc, [856](#)
  - pcps\_tong\_acquisition\_cc, [861](#)
- set\_alpha
  - Exponential\_Smoother, [398](#)
- set\_averaging\_depth
  - Pvt\_Solution, [874](#)
- set\_block\_exp
  - Fpga\_Acquisition, [409](#)
- set\_channel
  - beidou\_b1i\_telemetry\_decoder\_gs, [308](#)
  - beidou\_b3i\_telemetry\_decoder\_gs, [310](#)
  - BeidouB1iDIIPIITracking, [341](#)
  - BeidouB1iPcpsAcquisition, [344](#)
  - BeidouB3iDIIPIITracking, [348](#)
  - BeidouB3iPcpsAcquisition, [351](#)
  - dll\_pll\_veml\_tracking\_fpga, [394](#)

- galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc, 430
- galileo\_pcps\_8ms\_acquisition\_cc, 455
- galileo\_telemetry\_decoder\_gs, 459
- GalileoE1DIIPIIVemlTracking, 464
- GalileoE1DIIPIIVemlTrackingFpga, 468
- GalileoE1Pcps8msAmbiguousAcquisition, 471
- GalileoE1PcpsAmbiguousAcquisition, 475
- GalileoE1PcpsAmbiguousAcquisitionFpga, 481
- GalileoE1PcpsCccwsrAmbiguousAcquisition, 486
- GalileoE1PcpsQuickSyncAmbiguousAcquisition, 490
- GalileoE1PcpsTongAmbiguousAcquisition, 494
- GalileoE1TcpConnectorTracking, 498
- GalileoE5aDIIPIITracking, 500
- GalileoE5aDIIPIITrackingFpga, 503
- GalileoE5aNoncoherentIQAcquisitionCaf, 506
- GalileoE5aPcpsAcquisition, 511
- GalileoE5aPcpsAcquisitionFpga, 517
- GalileoE5bDIIPIITracking, 523
- GalileoE5bPcpsAcquisition, 528
- GalileoE5bPcpsAcquisitionFpga, 535
- GalileoE6DIIPIITracking, 541
- GalileoE6PcpsAcquisition, 544
- glonass\_l1\_ca\_telemetry\_decoder\_gs, 583
- glonass\_l2\_ca\_telemetry\_decoder\_gs, 587
- GlonassL1CaDIIPIICAidTracking, 589
- GlonassL1CaDIIPIITracking, 591
- GlonassL1CaPcpsAcquisition, 594
- GlonassL2CaDIIPIICAidTracking, 598
- GlonassL2CaDIIPIITracking, 600
- GlonassL2CaPcpsAcquisition, 603
- Gps\_Navigation\_Message, 706
- gps\_l1\_ca\_telemetry\_decoder\_gs, 700
- gps\_l2c\_telemetry\_decoder\_gs, 701
- gps\_l5\_telemetry\_decoder\_gs, 703
- GpsL1CaDIIPIITracking, 711
- GpsL1CaDIIPIITrackingFpga, 715
- GpsL1CaDIIPIITrackingGPU, 717
- GpsL1CaKfTracking, 719
- GpsL1CaPcpsAcquisition, 722
- GpsL1CaPcpsAcquisitionFineDoppler, 726
- GpsL1CaPcpsAcquisitionFpga, 732
- GpsL1CaPcpsAssistedAcquisition, 736
- GpsL1CaPcpsOpenCIAcquisition, 740
- GpsL1CaPcpsQuickSyncAcquisition, 744
- GpsL1CaPcpsTongAcquisition, 748
- GpsL1CaTcpConnectorTracking, 752
- GpsL2MDIIPIITracking, 756
- GpsL2MDIIPIITrackingFpga, 758
- GpsL2MPcpsAcquisition, 761
- GpsL2MPcpsAcquisitionFpga, 765
- GpsL5DIIPIITracking, 769
- GpsL5DIIPIITrackingFpga, 772
- GpsL5iPcpsAcquisition, 776
- GpsL5iPcpsAcquisitionFpga, 782
- pcps\_acquisition, 825
- pcps\_acquisition\_fine\_doppler\_cc, 830
- pcps\_acquisition\_fpga, 835
- pcps\_assisted\_acquisition\_cc, 841
- pcps\_cccwsr\_acquisition\_cc, 846
- pcps\_opencl\_acquisition\_cc, 851
- pcps\_quicksync\_acquisition\_cc, 856
- pcps\_tong\_acquisition\_cc, 861
- sbas\_l1\_telemetry\_decoder\_gs, 919
- set\_channel\_fsm
  - BeidouB1iPcpsAcquisition, 344
  - BeidouB3iPcpsAcquisition, 351
  - galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc, 430
  - galileo\_pcps\_8ms\_acquisition\_cc, 455
  - GalileoE1Pcps8msAmbiguousAcquisition, 471
  - GalileoE1PcpsAmbiguousAcquisition, 475
  - GalileoE1PcpsAmbiguousAcquisitionFpga, 482
  - GalileoE1PcpsCccwsrAmbiguousAcquisition, 486
  - GalileoE1PcpsQuickSyncAmbiguousAcquisition, 490
  - GalileoE1PcpsTongAmbiguousAcquisition, 494
  - GalileoE5aNoncoherentIQAcquisitionCaf, 507
  - GalileoE5aPcpsAcquisition, 511
  - GalileoE5aPcpsAcquisitionFpga, 517
  - GalileoE5bPcpsAcquisition, 528
  - GalileoE5bPcpsAcquisitionFpga, 535
  - GalileoE6PcpsAcquisition, 544
  - GlonassL1CaPcpsAcquisition, 594
  - GlonassL2CaPcpsAcquisition, 603
  - GpsL1CaPcpsAcquisition, 722
  - GpsL1CaPcpsAcquisitionFineDoppler, 726
  - GpsL1CaPcpsAcquisitionFpga, 732
  - GpsL1CaPcpsAssistedAcquisition, 737
  - GpsL1CaPcpsOpenCIAcquisition, 740
  - GpsL1CaPcpsQuickSyncAcquisition, 744
  - GpsL1CaPcpsTongAcquisition, 749
  - GpsL2MPcpsAcquisition, 761
  - GpsL2MPcpsAcquisitionFpga, 765
  - GpsL5iPcpsAcquisition, 776
  - GpsL5iPcpsAcquisitionFpga, 782
- pcps\_acquisition, 825
- pcps\_acquisition\_fine\_doppler\_cc, 830
- pcps\_acquisition\_fpga, 837
- pcps\_assisted\_acquisition\_cc, 842
- pcps\_cccwsr\_acquisition\_cc, 846
- pcps\_opencl\_acquisition\_cc, 851
- pcps\_quicksync\_acquisition\_cc, 857
- pcps\_tong\_acquisition\_cc, 862
- set\_clock\_drift\_ppm
  - Pvt\_Solution, 875
- set\_configuration
  - FrontEndCal, 420
  - GNSSFlowgraph, 642
- set\_control\_queue
  - ControlThread, 380
- set\_course\_over\_ground
  - Pvt\_Solution, 875
- set\_doppler\_center
  - GalileoE1PcpsAmbiguousAcquisition, 476

- GalileoE1PcpsAmbiguousAcquisitionFpga, [482](#)
- GalileoE5aPcpsAcquisition, [511](#)
- GalileoE5aPcpsAcquisitionFpga, [518](#)
- GalileoE5bPcpsAcquisition, [528](#)
- GalileoE5bPcpsAcquisitionFpga, [535](#)
- GalileoE6PcpsAcquisition, [545](#)
- GpsL1CaPcpsAcquisition, [722](#)
- GpsL1CaPcpsAcquisitionFpga, [733](#)
- GpsL2MPcpsAcquisition, [761](#)
- GpsL5iPcpsAcquisition, [776](#)
- GpsL5iPcpsAcquisitionFpga, [783](#)
- pcps\_acquisition, [825](#)
- pcps\_acquisition\_fpga, [837](#)
- set\_doppler\_max
  - BeidouB1iPcpsAcquisition, [344](#)
  - BeidouB3iPcpsAcquisition, [351](#)
  - Fpga\_Acquisition, [409](#)
  - galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc, [430](#)
  - galileo\_pcps\_8ms\_acquisition\_cc, [455](#)
  - GalileoE1Pcps8msAmbiguousAcquisition, [472](#)
  - GalileoE1PcpsAmbiguousAcquisition, [476](#)
  - GalileoE1PcpsAmbiguousAcquisitionFpga, [482](#)
  - GalileoE1PcpsCccwsrAmbiguousAcquisition, [486](#)
  - GalileoE1PcpsQuickSyncAmbiguousAcquisition, [490](#)
  - GalileoE1PcpsTongAmbiguousAcquisition, [495](#)
  - GalileoE5aNoncoherentIQAcquisitionCaf, [507](#)
  - GalileoE5aPcpsAcquisition, [511](#)
  - GalileoE5aPcpsAcquisitionFpga, [518](#)
  - GalileoE5bPcpsAcquisition, [529](#)
  - GalileoE5bPcpsAcquisitionFpga, [535](#)
  - GalileoE6PcpsAcquisition, [545](#)
  - GlionassL1CaPcpsAcquisition, [594](#)
  - GlionassL2CaPcpsAcquisition, [603](#)
  - GpsL1CaPcpsAcquisition, [722](#)
  - GpsL1CaPcpsAcquisitionFineDoppler, [727](#)
  - GpsL1CaPcpsAcquisitionFpga, [733](#)
  - GpsL1CaPcpsAssistedAcquisition, [737](#)
  - GpsL1CaPcpsOpenCIAcquisition, [741](#)
  - GpsL1CaPcpsQuickSyncAcquisition, [745](#)
  - GpsL1CaPcpsTongAcquisition, [749](#)
  - GpsL2MPcpsAcquisition, [761](#)
  - GpsL2MPcpsAcquisitionFpga, [766](#)
  - GpsL5iPcpsAcquisition, [776](#)
  - GpsL5iPcpsAcquisitionFpga, [783](#)
  - pcps\_acquisition, [825](#)
  - pcps\_acquisition\_fine\_doppler\_cc, [830](#)
  - pcps\_acquisition\_fpga, [837](#)
  - pcps\_assisted\_acquisition\_cc, [842](#)
  - pcps\_cccwsr\_acquisition\_cc, [846](#)
  - pcps\_openc1\_acquisition\_cc, [851](#)
  - pcps\_quicksync\_acquisition\_cc, [857](#)
  - pcps\_tong\_acquisition\_cc, [862](#)
- set\_doppler\_step
  - BeidouB1iPcpsAcquisition, [344](#)
  - BeidouB3iPcpsAcquisition, [352](#)
  - Fpga\_Acquisition, [409](#)
- galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc, [430](#)
- galileo\_pcps\_8ms\_acquisition\_cc, [455](#)
- GalileoE1Pcps8msAmbiguousAcquisition, [472](#)
- GalileoE1PcpsAmbiguousAcquisition, [476](#)
- GalileoE1PcpsAmbiguousAcquisitionFpga, [482](#)
- GalileoE1PcpsCccwsrAmbiguousAcquisition, [487](#)
- GalileoE1PcpsQuickSyncAmbiguousAcquisition, [491](#)
- GalileoE1PcpsTongAmbiguousAcquisition, [495](#)
- GalileoE5aNoncoherentIQAcquisitionCaf, [507](#)
- GalileoE5aPcpsAcquisition, [511](#)
- GalileoE5aPcpsAcquisitionFpga, [518](#)
- GalileoE5bPcpsAcquisition, [529](#)
- GalileoE5bPcpsAcquisitionFpga, [536](#)
- GalileoE6PcpsAcquisition, [545](#)
- GlionassL1CaPcpsAcquisition, [595](#)
- GlionassL2CaPcpsAcquisition, [604](#)
- GpsL1CaPcpsAcquisition, [723](#)
- GpsL1CaPcpsAcquisitionFineDoppler, [727](#)
- GpsL1CaPcpsAcquisitionFpga, [733](#)
- GpsL1CaPcpsAssistedAcquisition, [737](#)
- GpsL1CaPcpsOpenCIAcquisition, [741](#)
- GpsL1CaPcpsQuickSyncAcquisition, [745](#)
- GpsL1CaPcpsTongAcquisition, [749](#)
- GpsL2MPcpsAcquisition, [761](#)
- GpsL2MPcpsAcquisitionFpga, [766](#)
- GpsL5iPcpsAcquisition, [776](#)
- GpsL5iPcpsAcquisitionFpga, [783](#)
- pcps\_acquisition, [826](#)
- pcps\_acquisition\_fine\_doppler\_cc, [832](#)
- pcps\_acquisition\_fpga, [837](#)
- pcps\_assisted\_acquisition\_cc, [842](#)
- pcps\_cccwsr\_acquisition\_cc, [846](#)
- pcps\_openc1\_acquisition\_cc, [851](#)
- pcps\_quicksync\_acquisition\_cc, [857](#)
- pcps\_tong\_acquisition\_cc, [862](#)
- set\_doppler\_sweep
  - Fpga\_Acquisition, [410](#)
- set\_gnss\_synchro
  - BeidouB1iDIIPIITracking, [341](#)
  - BeidouB1iPcpsAcquisition, [345](#)
  - BeidouB3iDIIPIITracking, [348](#)
  - BeidouB3iPcpsAcquisition, [352](#)
  - dll\_pll\_veml\_tracking\_fpga, [394](#)
  - galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc, [431](#)
  - galileo\_pcps\_8ms\_acquisition\_cc, [456](#)
  - GalileoE1DIIPIIVemlTracking, [464](#)
  - GalileoE1DIIPIIVemlTrackingFpga, [468](#)
  - GalileoE1Pcps8msAmbiguousAcquisition, [472](#)
  - GalileoE1PcpsAmbiguousAcquisition, [476](#)
  - GalileoE1PcpsAmbiguousAcquisitionFpga, [482](#)
  - GalileoE1PcpsCccwsrAmbiguousAcquisition, [487](#)
  - GalileoE1PcpsQuickSyncAmbiguousAcquisition, [491](#)
  - GalileoE1PcpsTongAmbiguousAcquisition, [495](#)
  - GalileoE1TcpConnectorTracking, [498](#)

- GalileoE5aDIIPIITracking, [500](#)
- GalileoE5aDIIPIITrackingFpga, [504](#)
- GalileoE5aNoncoherentIQAcquisitionCaf, [507](#)
- GalileoE5aPcpsAcquisition, [512](#)
- GalileoE5aPcpsAcquisitionFpga, [518](#)
- GalileoE5bDIIPIITracking, [523](#)
- GalileoE5bPcpsAcquisition, [529](#)
- GalileoE5bPcpsAcquisitionFpga, [536](#)
- GalileoE6DIIPIITracking, [542](#)
- GalileoE6PcpsAcquisition, [545](#)
- GlonassL1CaDIIPIICAidTracking, [589](#)
- GlonassL1CaDIIPIITracking, [591](#)
- GlonassL1CaPcpsAcquisition, [595](#)
- GlonassL2CaDIIPIICAidTracking, [598](#)
- GlonassL2CaDIIPIITracking, [600](#)
- GlonassL2CaPcpsAcquisition, [604](#)
- GpsL1CaDIIPIITracking, [711](#)
- GpsL1CaDIIPIITrackingFpga, [715](#)
- GpsL1CaDIIPIITrackingGPU, [717](#)
- GpsL1CaKfTracking, [719](#)
- GpsL1CaPcpsAcquisition, [723](#)
- GpsL1CaPcpsAcquisitionFineDoppler, [727](#)
- GpsL1CaPcpsAcquisitionFpga, [733](#)
- GpsL1CaPcpsAssistedAcquisition, [737](#)
- GpsL1CaPcpsOpenCIAcquisition, [741](#)
- GpsL1CaPcpsQuickSyncAcquisition, [745](#)
- GpsL1CaPcpsTongAcquisition, [749](#)
- GpsL1CaTcpConnectorTracking, [752](#)
- GpsL2MDIIPIITracking, [756](#)
- GpsL2MDIIPIITrackingFpga, [758](#)
- GpsL2MPcpsAcquisition, [762](#)
- GpsL2MPcpsAcquisitionFpga, [766](#)
- GpsL5DIIPIITracking, [769](#)
- GpsL5DIIPIITrackingFpga, [773](#)
- GpsL5iPcpsAcquisition, [777](#)
- GpsL5iPcpsAcquisitionFpga, [783](#)
- pcps\_acquisition, [826](#)
- pcps\_acquisition\_fine\_doppler\_cc, [832](#)
- pcps\_acquisition\_fpga, [838](#)
- pcps\_assisted\_acquisition\_cc, [842](#)
- pcps\_cccwsr\_acquisition\_cc, [847](#)
- pcps\_openc1\_acquisition\_cc, [852](#)
- pcps\_quicksync\_acquisition\_cc, [858](#)
- pcps\_tong\_acquisition\_cc, [862](#)
- set\_initial\_sample
  - Fpga\_Multicorrelator\_8sc, [415](#)
- set\_local\_code
  - BeidouB1iPcpsAcquisition, [345](#)
  - BeidouB3iPcpsAcquisition, [352](#)
  - Fpga\_Acquisition, [410](#)
  - galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc, [431](#)
  - galileo\_pcps\_8ms\_acquisition\_cc, [456](#)
  - GalileoE1Pcps8msAmbiguousAcquisition, [472](#)
  - GalileoE1PcpsAmbiguousAcquisition, [476](#)
  - GalileoE1PcpsAmbiguousAcquisitionFpga, [483](#)
  - GalileoE1PcpsQuickSyncAmbiguousAcquisition, [491](#)
  - GalileoE1PcpsTongAmbiguousAcquisition, [495](#)
  - GalileoE5aNoncoherentIQAcquisitionCaf, [508](#)
  - GalileoE5aPcpsAcquisition, [512](#)
  - GalileoE5aPcpsAcquisitionFpga, [518](#)
  - GalileoE5bPcpsAcquisition, [529](#)
  - GalileoE5bPcpsAcquisitionFpga, [536](#)
  - GalileoE6PcpsAcquisition, [545](#)
  - GlonassL1CaPcpsAcquisition, [595](#)
  - GlonassL2CaPcpsAcquisition, [604](#)
  - GpsL1CaPcpsAcquisition, [723](#)
  - GpsL1CaPcpsAcquisitionFpga, [733](#)
  - GpsL1CaPcpsOpenCIAcquisition, [741](#)
  - GpsL1CaPcpsQuickSyncAcquisition, [745](#)
  - GpsL1CaPcpsTongAcquisition, [750](#)
  - GpsL2MPcpsAcquisition, [762](#)
  - GpsL2MPcpsAcquisitionFpga, [766](#)
  - GpsL5iPcpsAcquisition, [777](#)
  - GpsL5iPcpsAcquisitionFpga, [783](#)
  - pcps\_acquisition, [826](#)
  - pcps\_acquisition\_fine\_doppler\_cc, [832](#)
  - pcps\_acquisition\_fpga, [838](#)
  - pcps\_assisted\_acquisition\_cc, [843](#)
  - pcps\_cccwsr\_acquisition\_cc, [847](#)
  - pcps\_openc1\_acquisition\_cc, [852](#)
  - pcps\_quicksync\_acquisition\_cc, [858](#)
  - pcps\_tong\_acquisition\_cc, [863](#)
- set\_local\_code\_and\_taps
  - Fpga\_Multicorrelator\_8sc, [415](#)
- set\_num\_valid\_observations
  - Pvt\_Solution, [875](#)
- set\_output\_vectors
  - Fpga\_Multicorrelator\_8sc, [415](#)
- set\_pdi
  - Tracking\_2nd\_DLL\_filter, [952](#)
  - Tracking\_2nd\_PLL\_filter, [953](#)
- set\_pre\_2009\_file
  - Pvt\_Solution, [875](#)
  - Rinex\_Printer, [884](#)
- set\_resampler\_latency
  - BeidouB1iPcpsAcquisition, [345](#)
  - BeidouB3iPcpsAcquisition, [352](#)
  - GalileoE1PcpsAmbiguousAcquisition, [477](#)
  - GalileoE1PcpsAmbiguousAcquisitionFpga, [483](#)
  - GalileoE5aPcpsAcquisition, [512](#)
  - GalileoE5aPcpsAcquisitionFpga, [519](#)
  - GalileoE5bPcpsAcquisition, [529](#)
  - GalileoE5bPcpsAcquisitionFpga, [536](#)
  - GalileoE6PcpsAcquisition, [546](#)
  - GpsL1CaPcpsAcquisition, [723](#)
  - GpsL1CaPcpsAcquisitionFpga, [734](#)
  - GpsL2MPcpsAcquisition, [762](#)
  - GpsL5iPcpsAcquisition, [777](#)
  - GpsL5iPcpsAcquisitionFpga, [784](#)
- set\_rx\_pos
  - Pvt\_Solution, [875](#)
- set\_rx\_vel
  - Pvt\_Solution, [875](#)
- set\_samples\_for\_initialization

- Exponential\_Smoother, [398](#)
- set\_satellite
  - beidou\_b1i\_telemetry\_decoder\_gs, [308](#)
  - beidou\_b3i\_telemetry\_decoder\_gs, [310](#)
  - galileo\_telemetry\_decoder\_gs, [459](#)
  - glonass\_l1\_ca\_telemetry\_decoder\_gs, [583](#)
  - glonass\_l2\_ca\_telemetry\_decoder\_gs, [587](#)
  - gps\_l1\_ca\_telemetry\_decoder\_gs, [700](#)
  - gps\_l2c\_telemetry\_decoder\_gs, [702](#)
  - gps\_l5\_telemetry\_decoder\_gs, [703](#)
  - sbas\_l1\_telemetry\_decoder\_gs, [919](#)
- set\_satellite\_PRN
  - Beidou\_Dnav\_Navigation\_Message, [336](#)
  - Gps\_Navigation\_Message, [706](#)
- set\_secondary\_code\_lengths
  - Fpga\_Multicorrelator\_8sc, [415](#)
- set\_signal
  - Channel, [359](#)
- set\_single\_doppler\_flag
  - GalileoE5aPcpsAcquisitionFpga, [519](#)
  - GalileoE5bPcpsAcquisitionFpga, [536](#)
- set\_speed\_over\_ground
  - Pvt\_Solution, [876](#)
- set\_state
  - BeidouB1iPcpsAcquisition, [345](#)
  - BeidouB3iPcpsAcquisition, [352](#)
  - galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc, [431](#)
  - galileo\_pcps\_8ms\_acquisition\_cc, [456](#)
  - GalileoE1PcpsAmbiguousAcquisition, [477](#)
  - GalileoE1PcpsAmbiguousAcquisitionFpga, [483](#)
  - GalileoE1PcpsCccwsrAmbiguousAcquisition, [487](#)
  - GalileoE1PcpsQuickSyncAmbiguousAcquisition, [491](#)
  - GalileoE1PcpsTongAmbiguousAcquisition, [496](#)
  - GalileoE5aNoncoherentIQAcquisitionCaf, [508](#)
  - GalileoE5aPcpsAcquisition, [512](#)
  - GalileoE5aPcpsAcquisitionFpga, [519](#)
  - GalileoE5bPcpsAcquisition, [530](#)
  - GalileoE5bPcpsAcquisitionFpga, [537](#)
  - GalileoE6PcpsAcquisition, [546](#)
  - GlonassL1CaPcpsAcquisition, [595](#)
  - GlonassL2CaPcpsAcquisition, [604](#)
  - GpsL1CaPcpsAcquisition, [723](#)
  - GpsL1CaPcpsAcquisitionFineDoppler, [727](#)
  - GpsL1CaPcpsAcquisitionFpga, [734](#)
  - GpsL1CaPcpsQuickSyncAcquisition, [746](#)
  - GpsL1CaPcpsTongAcquisition, [750](#)
  - GpsL2MPcpsAcquisition, [762](#)
  - GpsL2MPcpsAcquisitionFpga, [767](#)
  - GpsL5iPcpsAcquisition, [777](#)
  - GpsL5iPcpsAcquisitionFpga, [784](#)
  - pcps\_acquisition, [827](#)
  - pcps\_acquisition\_fine\_doppler\_cc, [833](#)
  - pcps\_acquisition\_fpga, [838](#)
  - pcps\_cccwsr\_acquisition\_cc, [847](#)
  - pcps\_openc1\_acquisition\_cc, [852](#)
  - pcps\_quicksync\_acquisition\_cc, [858](#)
  - pcps\_tong\_acquisition\_cc, [863](#)
- set\_switch\_position
  - Fpga\_Switch, [417](#)
- set\_threshold
  - BeidouB1iPcpsAcquisition, [345](#)
  - BeidouB3iPcpsAcquisition, [353](#)
  - galileo\_e5a\_noncoherentIQ\_acquisition\_caf\_cc, [432](#)
  - galileo\_pcps\_8ms\_acquisition\_cc, [458](#)
  - GalileoE1Pcps8msAmbiguousAcquisition, [473](#)
  - GalileoE1PcpsAmbiguousAcquisition, [477](#)
  - GalileoE1PcpsAmbiguousAcquisitionFpga, [483](#)
  - GalileoE1PcpsCccwsrAmbiguousAcquisition, [487](#)
  - GalileoE1PcpsQuickSyncAmbiguousAcquisition, [491](#)
  - GalileoE1PcpsTongAmbiguousAcquisition, [496](#)
  - GalileoE5aNoncoherentIQAcquisitionCaf, [508](#)
  - GalileoE5aPcpsAcquisition, [513](#)
  - GalileoE5aPcpsAcquisitionFpga, [519](#)
  - GalileoE5bPcpsAcquisition, [530](#)
  - GalileoE5bPcpsAcquisitionFpga, [537](#)
  - GalileoE6PcpsAcquisition, [546](#)
  - GlonassL1CaPcpsAcquisition, [595](#)
  - GlonassL2CaPcpsAcquisition, [604](#)
  - GpsL1CaPcpsAcquisition, [724](#)
  - GpsL1CaPcpsAcquisitionFineDoppler, [728](#)
  - GpsL1CaPcpsAcquisitionFpga, [734](#)
  - GpsL1CaPcpsAssistedAcquisition, [738](#)
  - GpsL1CaPcpsOpenCIAcquisition, [742](#)
  - GpsL1CaPcpsQuickSyncAcquisition, [746](#)
  - GpsL1CaPcpsTongAcquisition, [750](#)
  - GpsL2MPcpsAcquisition, [762](#)
  - GpsL2MPcpsAcquisitionFpga, [767](#)
  - GpsL5iPcpsAcquisition, [777](#)
  - GpsL5iPcpsAcquisitionFpga, [784](#)
  - pcps\_acquisition, [827](#)
  - pcps\_acquisition\_fine\_doppler\_cc, [833](#)
  - pcps\_acquisition\_fpga, [839](#)
  - pcps\_assisted\_acquisition\_cc, [843](#)
  - pcps\_cccwsr\_acquisition\_cc, [848](#)
  - pcps\_openc1\_acquisition\_cc, [854](#)
  - pcps\_quicksync\_acquisition\_cc, [859](#)
  - pcps\_tong\_acquisition\_cc, [863](#)
- set\_time\_offset\_s
  - Pvt\_Solution, [876](#)
- short\_x2\_to\_cshort, [930](#)
- short\_x2\_to\_cshort.h, [1249](#)
- Signal
  - Gnss\_Synchro, [634](#)
- Signal Conditioner, [80](#)
- Signal Source, [169](#)
- signal\_conditioner.h, [1250](#)
- signal\_generator.h, [1250](#)
- signal\_generator\_c, [931](#)
- signal\_make\_generator\_c, [931](#)
- signal\_generator\_c.h, [1251](#)
- signal\_make\_generator\_c, [1252](#)
- signal\_generator\_flags.h, [1252](#)

- signal\_make\_generator\_c
  - signal\_generator\_c, [931](#)
  - signal\_generator\_c.h, [1252](#)
- signal\_source\_adapters, [170](#)
- signal\_source\_gr\_blocks, [171](#)
- signal\_source\_libs, [172](#)
  - RTL\_TCP\_COMMAND, [173](#)
  - rtl\_tcp\_command, [173](#)
- SignalConditioner, [932](#)
  - ~SignalConditioner, [933](#)
  - implementation, [933](#)
  - SignalConditioner, [933](#)
- SignalGenerator, [934](#)
  - implementation, [934](#)
- size
  - algorithms\_libs, [110](#)
- Skew\_symmetric
  - algorithms\_libs, [111](#)
- snrmask\_t, [935](#)
- sol\_t, [935](#)
- solbuf\_t, [935](#)
- solo\_t, [936](#)
- solstat\_t, [936](#)
- solstatbuf\_t, [937](#)
- spir\_file\_signal\_source.h, [1253](#)
- spir\_gss6450\_file\_signal\_source.h, [1254](#)
- SpirFileSignalSource, [939](#)
  - implementation, [939](#)
- SpirGSS6450FileSignalSource, [940](#)
- Spirent\_Motion\_Csv\_Dump\_Reader, [937](#)
- spirent\_motion\_csv\_dump\_reader.h, [1254](#)
- ssat\_t, [941](#)
- ssr\_t, [941](#)
- sta\_t, [942](#)
- start
  - Ad9361FpgaSignalSource, [299](#)
  - GNSSBlockInterface, [639](#)
  - GNSSFlowgraph, [642](#)
- start\_acquisition
  - Channel, [360](#)
- start\_tracking
  - dll\_pll\_veml\_tracking\_fpga, [394](#)
  - GalileoE1DIIPIIVemlTrackingFpga, [468](#)
  - GalileoE5aDIIPIITrackingFpga, [504](#)
  - GpsL1CaDIIPIITrackingFpga, [715](#)
  - GpsL5DIIPIITrackingFpga, [773](#)
- stec\_t, [942](#)
- stop
  - GNSSFlowgraph, [642](#)
- stop\_acquisition
  - BeidouB1iPcpsAcquisition, [346](#)
  - BeidouB3iPcpsAcquisition, [353](#)
  - GalileoE1Pcps8msAmbiguousAcquisition, [473](#)
  - GalileoE1PcpsAmbiguousAcquisition, [477](#)
  - GalileoE1PcpsAmbiguousAcquisitionFpga, [483](#)
  - GalileoE1PcpsCccwsrAmbiguousAcquisition, [487](#)
  - GalileoE1PcpsQuickSyncAmbiguousAcquisition, [492](#)
  - GalileoE1PcpsTongAmbiguousAcquisition, [496](#)
  - GalileoE5aNoncoherentIQAcquisitionCaf, [508](#)
  - GalileoE5aPcpsAcquisition, [513](#)
  - GalileoE5aPcpsAcquisitionFpga, [520](#)
  - GalileoE5bPcpsAcquisition, [530](#)
  - GalileoE5bPcpsAcquisitionFpga, [537](#)
  - GalileoE6PcpsAcquisition, [546](#)
  - GlomassL1CaPcpsAcquisition, [596](#)
  - GlomassL2CaPcpsAcquisition, [605](#)
  - GpsL1CaPcpsAcquisition, [724](#)
  - GpsL1CaPcpsAcquisitionFineDoppler, [728](#)
  - GpsL1CaPcpsAcquisitionFpga, [734](#)
  - GpsL1CaPcpsAssistedAcquisition, [738](#)
  - GpsL1CaPcpsOpenCIAcquisition, [742](#)
  - GpsL1CaPcpsQuickSyncAcquisition, [746](#)
  - GpsL1CaPcpsTongAcquisition, [750](#)
  - GpsL2MPcpsAcquisition, [763](#)
  - GpsL2MPcpsAcquisitionFpga, [767](#)
  - GpsL5iPcpsAcquisition, [778](#)
  - GpsL5iPcpsAcquisitionFpga, [784](#)
- stop\_channel
  - Channel, [360](#)
- stop\_server
  - Rtcm, [899](#)
- stop\_tracking
  - BeidouB1iDIIPIITracking, [341](#)
  - BeidouB3iDIIPIITracking, [348](#)
  - dll\_pll\_veml\_tracking\_fpga, [395](#)
  - GalileoE1DIIPIIVemlTracking, [464](#)
  - GalileoE1DIIPIIVemlTrackingFpga, [469](#)
  - GalileoE1TcpConnectorTracking, [498](#)
  - GalileoE5aDIIPIITracking, [500](#)
  - GalileoE5aDIIPIITrackingFpga, [504](#)
  - GalileoE5bDIIPIITracking, [524](#)
  - GalileoE6DIIPIITracking, [542](#)
  - GlomassL1CaDIIPIICAidTracking, [589](#)
  - GlomassL1CaDIIPIITracking, [591](#)
  - GlomassL2CaDIIPIICAidTracking, [599](#)
  - GlomassL2CaDIIPIITracking, [600](#)
  - GpsL1CaDIIPIITracking, [711](#)
  - GpsL1CaDIIPIITrackingFpga, [716](#)
  - GpsL1CaDIIPIITrackingGPU, [717](#)
  - GpsL1CaKfTracking, [719](#)
  - GpsL1CaTcpConnectorTracking, [752](#)
  - GpsL2MDIIPIITracking, [756](#)
  - GpsL2MDIIPIITrackingFpga, [758](#)
  - GpsL5DIIPIITracking, [769](#)
  - GpsL5DIIPIITrackingFpga, [773](#)
- stream\_cfg, [943](#)
- stream\_t, [943](#)
- string\_converter.h, [1255](#)
- string\_decoder
  - Glomass\_Gnav\_Navigation\_Message, [575](#)
- StringConverter, [944](#)
- subframe\_decoder
  - Gps\_Navigation\_Message, [706](#)
- sv\_clock\_correction
  - Beidou\_Dnav\_Navigation\_Message, [336](#)

- sv\_clock\_drift
  - Beidou\_Dnav\_Ephemeris, [319](#)
  - Galileo\_Ephemeris, [435](#)
  - Glonass\_Gnav\_Ephemeris, [563](#)
  - Gps\_CNAV\_Ephemeris, [658](#)
  - Gps\_Ephemeris, [682](#)
- sv\_clock\_relativistic\_term
  - Beidou\_Dnav\_Ephemeris, [319](#)
  - Galileo\_Ephemeris, [435](#)
  - Gps\_CNAV\_Ephemeris, [658](#)
  - Gps\_Ephemeris, [682](#)
- swift\_common.h, [1255](#)
- symbols
  - cnav\_v27\_part\_t, [372](#)
- System
  - Gnss\_Synchro, [634](#)
- t0c\_4
  - Galileo\_Ephemeris, [444](#)
- t0e\_1
  - Galileo\_Ephemeris, [444](#)
- t0t\_6
  - Galileo\_Utc\_Model, [461](#)
- T\_OA
  - core\_system\_parameters, [231](#)
- TIMES\_GPST
  - algorithms\_libs\_rtklib, [156](#)
- TIMES\_JST
  - algorithms\_libs\_rtklib, [156](#)
- TIMES\_UTC
  - algorithms\_libs\_rtklib, [156](#)
- TIMETAGH\_LEN
  - algorithms\_libs\_rtklib, [156](#)
- TINTACT
  - algorithms\_libs\_rtklib, [156](#)
- TOW\_5
  - Galileo\_Ephemeris, [444](#)
  - Galileo\_Iono, [451](#)
- TOW\_at\_current\_symbol\_ms
  - Gnss\_Synchro, [634](#)
- TROPOPT\_CORG
  - algorithms\_libs\_rtklib, [157](#)
- TROPOPT\_COR
  - algorithms\_libs\_rtklib, [157](#)
- TROPOPT\_ESTG
  - algorithms\_libs\_rtklib, [157](#)
- TROPOPT\_EST
  - algorithms\_libs\_rtklib, [157](#)
- TROPOPT\_OFF
  - algorithms\_libs\_rtklib, [157](#)
- TROPOPT\_SAAS
  - algorithms\_libs\_rtklib, [158](#)
- TROPOPT\_SBAS
  - algorithms\_libs\_rtklib, [158](#)
- TWO\_N10
  - core\_system\_parameters, [280](#)
- TWO\_N11
  - core\_system\_parameters, [281](#)
- TWO\_N14
  - core\_system\_parameters, [281](#)
- TWO\_N15
  - core\_system\_parameters, [281](#)
- TWO\_N16
  - core\_system\_parameters, [281](#)
- TWO\_N17
  - core\_system\_parameters, [281](#)
- TWO\_N18
  - core\_system\_parameters, [282](#)
- TWO\_N19
  - core\_system\_parameters, [282](#)
- TWO\_N2
  - core\_system\_parameters, [282](#)
- TWO\_N20
  - core\_system\_parameters, [282](#)
- TWO\_N21
  - core\_system\_parameters, [282](#)
- TWO\_N23
  - core\_system\_parameters, [283](#)
- TWO\_N24
  - core\_system\_parameters, [283](#)
- TWO\_N25
  - core\_system\_parameters, [283](#)
- TWO\_N27
  - core\_system\_parameters, [283](#)
- TWO\_N29
  - core\_system\_parameters, [283](#)
- TWO\_N30
  - core\_system\_parameters, [284](#)
- TWO\_N31
  - core\_system\_parameters, [284](#)
- TWO\_N32
  - core\_system\_parameters, [284](#)
- TWO\_N33
  - core\_system\_parameters, [284](#)
- TWO\_N34
  - core\_system\_parameters, [284](#)
- TWO\_N35
  - core\_system\_parameters, [285](#)
- TWO\_N38
  - core\_system\_parameters, [285](#)
- TWO\_N39
  - core\_system\_parameters, [285](#)
- TWO\_N40
  - core\_system\_parameters, [285](#)
- TWO\_N43
  - core\_system\_parameters, [285](#)
- TWO\_N44
  - core\_system\_parameters, [286](#)
- TWO\_N46
  - core\_system\_parameters, [286](#)
- TWO\_N48
  - core\_system\_parameters, [286](#)
- TWO\_N5
  - core\_system\_parameters, [286](#)
- TWO\_N50
  - core\_system\_parameters, [286](#)
- TWO\_N51

- core\_system\_parameters, 287
- TWO\_N55
  - core\_system\_parameters, 287
- TWO\_N57
  - core\_system\_parameters, 287
- TWO\_N59
  - core\_system\_parameters, 287
- TWO\_N6
  - core\_system\_parameters, 287
- TWO\_N60
  - core\_system\_parameters, 288
- TWO\_N66
  - core\_system\_parameters, 288
- TWO\_N68
  - core\_system\_parameters, 288
- TWO\_N8
  - core\_system\_parameters, 288
- TWO\_N9
  - core\_system\_parameters, 288
- TWO\_P11
  - core\_system\_parameters, 289
- TWO\_P12
  - core\_system\_parameters, 289
- TWO\_P14
  - core\_system\_parameters, 289
- TWO\_P16
  - core\_system\_parameters, 289
- TWO\_P19
  - core\_system\_parameters, 289
- TWO\_P3
  - core\_system\_parameters, 290
- TWO\_P31
  - core\_system\_parameters, 290
- TWO\_P32
  - core\_system\_parameters, 290
- TWO\_P4
  - core\_system\_parameters, 290
- TWO\_P56
  - core\_system\_parameters, 290
- TWO\_P57
  - core\_system\_parameters, 291
- TWO\_PI
  - core\_system\_parameters, 291
- Tcp\_Communication, 944
- Tcp\_Packet\_Data, 945
- tcp\_cmd\_interface.h, 1256
- tcp\_communication.h, 1257
- tcp\_packet\_data.h, 1258
- tcp\_t, 946
- TcpCmdInterface, 946
  - get\_LLH, 947
  - get\_utc\_time, 947
- tcpcli\_t, 946
- tcpsvr\_t, 947
- tec\_t, 948
- Telemetry Decoder, 174
- telemetry\_decoder\_adapters, 175
- telemetry\_decoder\_gr\_blocks, 176
- telemetry\_decoder\_interface.h, 1258
- telemetry\_decoder\_libs, 178
  - Gamma, 178
  - nsc\_enc\_bit, 179
  - nsc\_transit, 179
  - parity\_counter, 180
  - Viterbi, 180
- telemetry\_decoder\_libswiftcnv, 182
  - GPS\_L2\_V27\_HISTORY\_LENGTH\_BITS, 182
  - GPS\_L2C\_V27\_DECODE\_BITS, 183
  - GPS\_L2C\_V27\_DELAY\_BITS, 183
  - GPS\_L2C\_V27\_INIT\_BITS, 183
- TelemetryDecoderInterface, 948
- test\_flags.h, 1259
- tle\_t, 949
- tled\_t, 949
- Tlm\_Conf, 950
- Tlm\_Dump\_Reader, 950
- tlm\_conf.h, 1259
- tlm\_dump\_reader.h, 1260
- tlm\_utils.h, 1260
- togeod
  - algorithms\_libs, 111
- topocent
  - algorithms\_libs, 111
- tow
  - cnv\_msg\_t, 369
- Tracking, 184
- Tracking\_2nd\_DLL\_filter, 951
  - get\_code\_nco, 951
  - initialize, 951
  - set\_DLL\_BW, 952
  - set\_pdi, 952
- tracking\_2nd\_DLL\_filter.h, 1261
- Tracking\_2nd\_PLL\_filter, 952
  - set\_PLL\_BW, 953
  - set\_pdi, 953
- tracking\_2nd\_PLL\_filter.h, 1261
- Tracking\_Dump\_Reader, 953
- Tracking\_FLL\_PLL\_filter, 954
- tracking\_FLL\_PLL\_filter.h, 1263
- Tracking\_True\_Obs\_Reader, 956
- tracking\_adapters, 185
- tracking\_discriminators.h, 1262
- tracking\_dump\_reader.h, 1263
- tracking\_gr\_blocks, 186
- tracking\_interface.h, 1264
- tracking\_libs, 188
  - carrier\_lock\_detector, 189
  - cn0\_m2m4\_estimator, 190
  - cn0\_svn\_estimator, 190
  - dll\_nc\_e\_minus\_l\_normalized, 191
  - dll\_nc\_vemlp\_normalized, 191
  - fil\_four\_quadrant\_atan, 191
  - phase\_unwrap, 192
  - pll\_cloop\_two\_quadrant\_atan, 192
  - pll\_four\_quadrant\_atan, 192
- Tracking\_loop\_filter, 955

- operator=, [956](#)
- Tracking\_loop\_filter, [955](#)
- tracking\_loop\_filter.h, [1264](#)
- Tracking\_sample\_counter
  - Gnss\_Synchro, [634](#)
- tracking\_tests\_flags.h, [1265](#)
- tracking\_true\_obs\_reader.h, [1266](#)
- TrackingInterface, [957](#)
- trop\_t, [958](#)
- True\_Observables\_Reader, [958](#)
- true\_observables\_reader.h, [1267](#)
- two\_bit\_cpx\_file\_signal\_source.h, [1267](#)
- two\_bit\_packed\_file\_signal\_source.h, [1268](#)
- TwoBitCpxFileSignalSource, [959](#)
  - implementation, [959](#)
- TwoBitPackedFileSignalSource, [960](#)
  - implementation, [960](#)
- uhd\_signal\_source.h, [1269](#)
- UhdSignalSource, [961](#)
  - implementation, [962](#)
- uio\_fpga.h, [1269](#)
- unlock\_channel
  - Fpga\_Multicorrelator\_8sc, [415](#)
- unpack\_2bit\_samples, [962](#)
- unpack\_2bit\_samples.h, [1270](#)
- unpack\_byte\_2bit\_cpx\_samples, [963](#)
- unpack\_byte\_2bit\_cpx\_samples.h, [1271](#)
- unpack\_byte\_2bit\_samples, [964](#)
- unpack\_byte\_2bit\_samples.h, [1272](#)
- unpack\_byte\_4bit\_samples, [964](#)
- unpack\_byte\_4bit\_samples.h, [1273](#)
- unpack\_intspir\_1bit\_samples, [965](#)
- unpack\_intspir\_1bit\_samples.h, [1274](#)
- unpack\_spir\_gss6450\_samples, [966](#)
- unpack\_spir\_gss6450\_samples.h, [1274](#)
- UnscentedFilter, [966](#)
- unset\_title
  - Gnuplot, [645](#)
- update\_PRN
  - Gnss\_Satellite, [612](#)
- update\_local\_code
  - Fpga\_Multicorrelator\_8sc, [416](#)
- update\_prn\_code\_length
  - Fpga\_Multicorrelator\_8sc, [416](#)
- url\_t, [967](#)
- utc\_time
  - Beidou\_Dnav\_Navigation\_Message, [337](#)
  - Glonass\_Gnav\_Utc\_Model, [577](#)
  - Gps\_Navigation\_Message, [707](#)
- v27\_decision\_t, [967](#)
- v27\_poly\_t, [967](#)
- v27\_t, [968](#)
- valid
  - Beidou\_Dnav\_Iono, [333](#)
  - Gps\_CNAV\_Iono, [672](#)
  - Gps\_Iono, [696](#)
- Viterbi
  - telemetry\_decoder\_libs, [180](#)
- Viterbi\_Decoder, [968](#)
  - decode\_block, [969](#)
- viterbi\_decoder.h, [1275](#)
- WN\_5
  - Galileo\_Ephemeris, [444](#)
  - Galileo\_Iono, [452](#)
- WNot\_6
  - Galileo\_Utc\_Model, [461](#)
- wait
  - GNSSFlowgraph, [643](#)
- what\_block
  - Gnss\_Satellite, [612](#)
- work
  - rtklib\_pvt\_gs, [907](#)
- write\_local\_code
  - Fpga\_Acquisition, [410](#)