



CryoSat2 IO Library : Version 2.3 : Fri May 20 2016

Generated by UCL CryoSat2 Software Team

Fri May 20 2016 09:51:26

<http://cryosat.mssl.ucl.ac.uk/public>



# Contents

<b>1</b>	<b>Library Overview</b>	<b>1</b>
1.1	Introduction	1
1.1.1	Platform compatability and requirements	1
1.1.2	Installation Instructions	2
1.1.3	Library Usage	2
1.1.4	Changes	2
<b>2</b>	<b>Module Index</b>	<b>3</b>
2.1	Modules	3
<b>3</b>	<b>Data Structure Index</b>	<b>5</b>
3.1	Data Structures	5
<b>4</b>	<b>File Index</b>	<b>7</b>
4.1	File List	7
<b>5</b>	<b>Module Documentation</b>	<b>9</b>
5.1	High-level shared routines	9
5.1.1	Detailed Description	9
5.1.2	Function Documentation	9
5.1.2.1	ptCSGetFileHandle(char *s_filename, BASELINE j_baseline)	9
5.1.2.2	jCSNumRecordsInDataset(t_cs_filehandle t_handle, long int j_dataset)	10
5.1.2.3	vCSFreeFileHandle(t_cs_filehandle t_handle)	10
5.1.2.4	jCSGetDSFileOffsetFromHeader(fpos_t *pt_dataset_file_offset, long int *pj_num_records, char *s_filename, CS_DS_NAME t_ds_name, long int *pj_actual_ds)	10
5.2	High Level CryoSat Level 1b File IO API	12
5.2.1	Detailed Description	12
5.2.2	Function Documentation	12
5.2.2.1	ptCSGetL1bLRM(t_cs_filehandle t_handle, long int j_dataset_index, unsigned long long int uk_num_records, unsigned long long int uk_skip_records, L1B_LRM *pt_data)	12
5.2.2.2	ptCSGetL1bSAR(t_cs_filehandle t_handle, long int j_dataset_index, unsigned long long int uk_num_records, unsigned long long int uk_skip_records, L1B_SAR *pt_data)	13

5.2.2.3	ptCSGetL1bSARIN(t_cs_filehandle t_handle, long int j_dataset_index, unsigned long long int uk_num_records, unsigned long long int uk_skip_records, L1B_SARIN *pt_data)	14
5.3	Low-Level CryoSat Level 1b File IO API	15
5.3.1	Detailed Description	15
5.3.2	Function Documentation	15
5.3.2.1	jFillL1bLRMStructFromFile(L1B_LRM *p_ToFill, FILE *pt_input, BASELINE j_Baseline)	15
5.3.2.2	jFillL1bSARStructFromFile(L1B_SAR *p_ToFill, FILE *pt_input, BASELINE j_Baseline)	15
5.3.2.3	jFillL1bSARINStructFromFile(L1B_SARIN *p_ToFill, FILE *pt_input, BASELINE j_Baseline)	16
5.4	High Level CryoSat Level 2 File IO API	17
5.4.1	Detailed Description	17
5.4.2	Function Documentation	17
5.4.2.1	ptCSGetL2(t_cs_filehandle t_handle, long int j_dataset_index, unsigned long long int uk_num_records, unsigned long long int uk_skip_records, L2Data *pt_data)	17
5.4.2.2	ptCSGetL2I(t_cs_filehandle t_handle, long int j_dataset_index, unsigned long long int uk_num_records, unsigned long long int uk_skip_records, L2IData *pt_data)	18
5.5	Low-Level CryoSat Level 2 File IO API	20
5.5.1	Detailed Description	20
5.5.2	Function Documentation	20
5.5.2.1	jFillL2StructFromFile(L2Data *pt_Level2, FILE *pt_File, BASELINE j_Baseline)	20
5.5.2.2	jFillL2IStructFromFile(L2IData *pt_Level2, FILE *pt_File, BASELINE j_Baseline)	20
5.6	Utility functions	22
5.6.1	Detailed Description	22
5.6.2	Function Documentation	22
5.6.2.1	vSetBitShort(uint16_t *pBitmask, uint16_t bit)	22
5.6.2.2	iCheckBitShort(uint16_t *pBitmask, uint16_t bit)	22
5.6.2.3	vClearBitShort(uint16_t *pBitmask, uint16_t bit)	23
5.6.2.4	vSetBitLong(uint32_t *pBitmask, uint16_t bit)	24
5.6.2.5	iCheckBitLong(uint32_t *pBitmask, uint16_t bit)	24
5.6.2.6	vSetBitLongLong(uint64_t *pBitmask, uint16_t bit)	24
5.6.2.7	iCheckBitLongLong(uint64_t *pBitmask, uint16_t bit)	24
5.6.2.8	vClearBitLongLong(uint64_t *pBitmask, uint16_t bit)	25
5.7	IDL Routines	26
5.7.1	IDL Overview	26
5.7.2	cs_l1b_read	26
5.7.3	cs_l2_read	26
5.7.4	cs_l2i_read	27

6.1	AveragedWaveformsLRM Struct Reference	29
6.1.1	Detailed Description	29
6.1.2	Field Documentation	30
6.1.2.1	j_Day_1Hz	30
6.1.2.2	uj_Sec_1Hz	30
6.1.2.3	uj_Micsec_1Hz	30
6.1.2.4	j_Lat_1Hz	30
6.1.2.5	j_Lon_1Hz	30
6.1.2.6	j_Alt_1Hz	30
6.1.2.7	k_TD_1Hz	30
6.1.2.8	pui_avg_wfm	30
6.1.2.9	j_Linear_wfm_mult	31
6.1.2.10	j_Power2_wfm_mult	31
6.1.2.11	ui_num_avg_echoes	31
6.1.2.12	ui_flags	31
6.2	AveragedWaveformsSAR Struct Reference	31
6.2.1	Detailed Description	32
6.2.2	Field Documentation	32
6.2.2.1	j_Day_1Hz	32
6.2.2.2	uj_Sec_1Hz	32
6.2.2.3	uj_Micsec_1Hz	32
6.2.2.4	j_Lat_1Hz	32
6.2.2.5	j_Lon_1Hz	32
6.2.2.6	j_Alt_1Hz	32
6.2.2.7	k_TD_1Hz	33
6.2.2.8	pui_avg_wfm	33
6.2.2.9	j_Linear_wfm_mult	33
6.2.2.10	j_Power2_wfm_mult	33
6.2.2.11	ui_num_avg_echoes	33
6.2.2.12	ui_flags	33
6.3	AveragedWaveformsSARIN Struct Reference	33
6.3.1	Detailed Description	34
6.3.2	Field Documentation	34
6.3.2.1	j_Day_1Hz	34
6.3.2.2	uj_Sec_1Hz	34
6.3.2.3	uj_Micsec_1Hz	35
6.3.2.4	j_Lat_1Hz	35
6.3.2.5	j_Lon_1Hz	35
6.3.2.6	j_Alt_1Hz	35
6.3.2.7	k_TD_1Hz	35

6.3.2.8	pui_avg_wfm . . . . .	35
6.3.2.9	j_Linear_wfm_mult . . . . .	35
6.3.2.10	j_Power2_wfm_mult . . . . .	35
6.3.2.11	ui_num_avg_echoes . . . . .	36
6.3.2.12	ui_flags . . . . .	36
6.4	Corrections Struct Reference . . . . .	36
6.4.1	Detailed Description . . . . .	37
6.4.2	Field Documentation . . . . .	37
6.4.2.1	j_Dry_C . . . . .	37
6.4.2.2	j_Wet_C . . . . .	37
6.4.2.3	j_IB_C . . . . .	37
6.4.2.4	j_DAC_C . . . . .	37
6.4.2.5	j_Iono_GIM . . . . .	37
6.4.2.6	j_Iono_mod . . . . .	38
6.4.2.7	j_H_OT . . . . .	38
6.4.2.8	j_H_LPEOT . . . . .	38
6.4.2.9	j_H_OLT . . . . .	38
6.4.2.10	j_H_SET . . . . .	38
6.4.2.11	j_H_GPT . . . . .	38
6.4.2.12	uj_Surf_type . . . . .	38
6.4.2.13	spare1 . . . . .	38
6.4.2.14	uj_Corr_status . . . . .	39
6.4.2.15	uj_Corr_error . . . . .	39
6.4.2.16	spare2 . . . . .	39
6.5	L1B_LRM Struct Reference . . . . .	39
6.5.1	Detailed Description . . . . .	39
6.5.2	Field Documentation . . . . .	40
6.5.2.1	at_Time_Orbit . . . . .	40
6.5.2.2	at_Measurements . . . . .	40
6.5.2.3	t_Geo_Corrections . . . . .	40
6.5.2.4	t_Average_PL . . . . .	40
6.5.2.5	at_Waveforms . . . . .	40
6.6	L1B_SAR Struct Reference . . . . .	40
6.6.1	Detailed Description . . . . .	41
6.6.2	Field Documentation . . . . .	41
6.6.2.1	at_Time_Orbit . . . . .	41
6.6.2.2	at_Measurements . . . . .	41
6.6.2.3	t_Geo_Corrections . . . . .	41
6.6.2.4	t_Average_PL . . . . .	41
6.6.2.5	at_Waveforms . . . . .	41

6.7	L1B_SARIN Struct Reference	41
6.7.1	Detailed Description	42
6.7.2	Field Documentation	42
6.7.2.1	at_Time_Orbit	42
6.7.2.2	at_Measurements	42
6.7.2.3	t_Geo_Corrections	42
6.7.2.4	t_Average_PL	42
6.7.2.5	at_Waveforms	43
6.8	L2Data Struct Reference	43
6.8.1	Detailed Description	43
6.8.2	Field Documentation	44
6.8.2.1	j_Day	44
6.8.2.2	uj_Sec	44
6.8.2.3	uj_Micsec	44
6.8.2.4	uk_Siral_mode	44
6.8.2.5	j_Lat_1Hz	44
6.8.2.6	j_Lon_1Hz	44
6.8.2.7	j_Alt_1Hz	44
6.8.2.8	i_Mispointing	45
6.8.2.9	j_Roll_angle	45
6.8.2.10	j_Pitch_angle	45
6.8.2.11	j_Yaw_angle	45
6.8.2.12	i_N_Valid	45
6.8.2.13	t_Corr	45
6.8.2.14	at_Meas	46
6.9	L2FinalC Struct Reference	46
6.9.1	Detailed Description	47
6.9.2	Field Documentation	47
6.9.2.1	i_Dry_C	47
6.9.2.2	i_Wet_C	47
6.9.2.3	i_IB_C	47
6.9.2.4	i_DAC_C	47
6.9.2.5	i_Iono_C	47
6.9.2.6	i_SSB	47
6.9.2.7	i_H_OT	47
6.9.2.8	i_H_LPEOT	48
6.9.2.9	i_H_OLT	48
6.9.2.10	i_H_SET	48
6.9.2.11	i_H_GPT	48
6.9.2.12	i_Spare1	48

6.9.2.13	uk_Surf_type	48
6.9.2.14	j_MSS_Geoid	48
6.9.2.15	j_ODLE	49
6.9.2.16	i_Ice_conc	49
6.9.2.17	i_Snow_depth	49
6.9.2.18	i_Snow_density	49
6.9.2.19	uj_C_status	49
6.9.2.20	i_SWH	49
6.9.2.21	ui_Wind_speed	49
6.10	L2FinalM Struct Reference	50
6.10.1	Detailed Description	50
6.10.2	Field Documentation	51
6.10.2.1	j_D_time_mics	51
6.10.2.2	j_Lat	51
6.10.2.3	j_Lon	51
6.10.2.4	j_Height	51
6.10.2.5	j_Height_2	51
6.10.2.6	j_Height_3	51
6.10.2.7	i_SSHA_interp	51
6.10.2.8	i_SSHA_num	52
6.10.2.9	i_SSHA_qual	52
6.10.2.10	i_Sig0	52
6.10.2.11	i_Sig0_2	52
6.10.2.12	i_Sig0_3	52
6.10.2.13	i_Peakiness	52
6.10.2.14	i_Freeboard	52
6.10.2.15	i_N_avg	53
6.10.2.16	uj_Quality_F	53
6.10.2.17	uj_C_applied	53
6.10.2.18	j_Quality_1	53
6.10.2.19	j_Quality_2	53
6.10.2.20	j_Quality_3	53
6.11	L2IData Struct Reference	54
6.11.1	Detailed Description	57
6.11.2	Field Documentation	57
6.11.2.1	j_Day	57
6.11.2.2	uj_Sec	57
6.11.2.3	uj_Micsec	57
6.11.2.4	j_USO_Corr	57
6.11.2.5	ui_Mode_ID	57



6.11.2.6	ui_SSC	57
6.11.2.7	uj_Inst_config	58
6.11.2.8	uj_Rec_Count	58
6.11.2.9	j_Lat	58
6.11.2.10	j_Lon	58
6.11.2.11	j_Alt	58
6.11.2.12	j_Alt_rate	58
6.11.2.13	aj_Sat_velocity	58
6.11.2.14	aj_Real_beam	59
6.11.2.15	aj_Baseline	59
6.11.2.16	i_ST_ID	59
6.11.2.17	j_Roll_angle	59
6.11.2.18	j_Pitch_angle	59
6.11.2.19	j_Yaw_angle	59
6.11.2.20	uj_L2_MCD	60
6.11.2.21	j_Height	60
6.11.2.22	j_Height_2	60
6.11.2.23	j_Height_3	60
6.11.2.24	j_Sig0	60
6.11.2.25	j_Sig0_2	60
6.11.2.26	j_Sig0_3	60
6.11.2.27	j_SWH	61
6.11.2.28	j_Peakiness	61
6.11.2.29	j_Retrk_range_C	61
6.11.2.30	j_Retrk_range_C_2	61
6.11.2.31	j_Retrk_range_C_3	61
6.11.2.32	j_Retrk_sig0_C	61
6.11.2.33	j_Retrk_sig0_C_2	61
6.11.2.34	j_Retrk_sig0_C_3	62
6.11.2.35	j_Retrk_Quality_1	62
6.11.2.36	j_Retrk_Quality_2	62
6.11.2.37	j_Retrk_Quality_3	62
6.11.2.38	j_Retrk_3	62
6.11.2.39	j_Retrk_4	62
6.11.2.40	j_Retrk_5	62
6.11.2.41	j_Retrk_6	63
6.11.2.42	j_Retrk_7	63
6.11.2.43	j_Retrk_8	63
6.11.2.44	j_Retrk_9	63
6.11.2.45	j_Retrk_10	63

6.11.2.46 j_Retrk_11 . . . . .	63
6.11.2.47 j_Retrk_12 . . . . .	63
6.11.2.48 j_Retrk_13 . . . . .	63
6.11.2.49 j_Retrk_14 . . . . .	64
6.11.2.50 j_Retrk_15 . . . . .	64
6.11.2.51 j_Retrk_16 . . . . .	64
6.11.2.52 j_Retrk_17 . . . . .	64
6.11.2.53 j_Retrk_18 . . . . .	64
6.11.2.54 j_Retrk_19 . . . . .	64
6.11.2.55 j_Retrk_20 . . . . .	65
6.11.2.56 j_Retrk_21 . . . . .	65
6.11.2.57 j_Retrk_22 . . . . .	65
6.11.2.58 j_Retrk_23 . . . . .	65
6.11.2.59 j_echo_shape . . . . .	65
6.11.2.60 ai_bb_param . . . . .	65
6.11.2.61 j_XTrack_angle . . . . .	66
6.11.2.62 j_XTrack_angle_C . . . . .	66
6.11.2.63 j_Coherence . . . . .	66
6.11.2.64 j_Ocean_ht . . . . .	66
6.11.2.65 j_Freeboard . . . . .	66
6.11.2.66 j_SHA . . . . .	66
6.11.2.67 j_SSHA_interp . . . . .	66
6.11.2.68 ui_interp_err . . . . .	67
6.11.2.69 ui_interp_cnt_fwd . . . . .	67
6.11.2.70 ui_interp_cnt_bkwd . . . . .	67
6.11.2.71 ui_interp_time_fwd . . . . .	67
6.11.2.72 ui_interp_time_bkwd . . . . .	67
6.11.2.73 ui_interp_error_F . . . . .	67
6.11.2.74 uj_Meas_Mode . . . . .	67
6.11.2.75 uj_Quality_F . . . . .	67
6.11.2.76 uj_Retracker_F . . . . .	68
6.11.2.77 uj_Ht_status . . . . .	68
6.11.2.78 uj_Freeb_status . . . . .	68
6.11.2.79 ui_n_avg . . . . .	68
6.11.2.80 ui_Wind_speed . . . . .	68
6.11.2.81 auc_spares1 . . . . .	68
6.11.2.82 j_Ice_conc . . . . .	68
6.11.2.83 j_Snow_depth . . . . .	69
6.11.2.84 j_Snow_density . . . . .	69
6.11.2.85 j_Discriminator . . . . .	69

6.11.2.86 j_SARin_disc_1 . . . . .	69
6.11.2.87 j_SARin_disc_2 . . . . .	69
6.11.2.88 j_SARin_disc_3 . . . . .	69
6.11.2.89 j_SARin_disc_4 . . . . .	69
6.11.2.90 j_SARin_disc_5 . . . . .	69
6.11.2.91 j_SARin_disc_6 . . . . .	70
6.11.2.92 j_SARin_disc_7 . . . . .	70
6.11.2.93 j_SARin_disc_8 . . . . .	70
6.11.2.94 j_SARin_disc_9 . . . . .	70
6.11.2.95 j_SARin_disc_10 . . . . .	70
6.11.2.96 uj_Discrim_F . . . . .	70
6.11.2.97 j_Attitude . . . . .	70
6.11.2.98 j_Azimuth . . . . .	70
6.11.2.99 j_Slope_Doppler_C . . . . .	71
6.11.2.100 j_Lat_sat . . . . .	71
6.11.2.101 j_Lon_sat . . . . .	71
6.11.2.102 j_Ambiguity . . . . .	71
6.11.2.103 j_MSS_mod . . . . .	71
6.11.2.104 j_Geoid_mod . . . . .	71
6.11.2.105 j_ODLE_mod . . . . .	71
6.11.2.106 j_DEM_elev . . . . .	72
6.11.2.107 j_DEM_id . . . . .	72
6.11.2.108 auc_spares2 . . . . .	72
6.11.2.109 j_Dry_C . . . . .	72
6.11.2.110 j_Wet_C . . . . .	72
6.11.2.111 j_IB_C . . . . .	72
6.11.2.112 j_DAC_C . . . . .	72
6.11.2.113 j_Iono_GIM . . . . .	73
6.11.2.114 j_Iono_mod . . . . .	73
6.11.2.115 j_H_OT . . . . .	73
6.11.2.116 j_H_LPEOT . . . . .	73
6.11.2.117 j_H_OLT . . . . .	73
6.11.2.118 j_H_SET . . . . .	73
6.11.2.119 j_H_GPT . . . . .	73
6.11.2.120 j_Surf_type . . . . .	73
6.11.2.121 j_Corr_status . . . . .	74
6.11.2.122 j_Corr_error . . . . .	74
6.11.2.123 j_SSB . . . . .	74
6.11.2.124 auc_spares3 . . . . .	74
6.11.2.125 j_Dopp_rc . . . . .	74

6.11.2.126	TR_inst_rc	74
6.11.2.127	R_inst_rc	74
6.11.2.128	TR_inst_gain_C	75
6.11.2.129	R_inst_gain_C	75
6.11.2.130	Int_phase_C	75
6.11.2.131	Ext_phase_C	75
6.11.2.132	Noise_pwr	75
6.11.2.133	Phase_slope_C	75
6.11.2.134	auc_spares4	75
6.12	Measurements Struct Reference	76
6.12.1	Detailed Description	77
6.12.2	Field Documentation	77
6.12.2.1	k_TD	77
6.12.2.2	j_H0	77
6.12.2.3	j_COR2	77
6.12.2.4	j_LAI	77
6.12.2.5	j_FAI	77
6.12.2.6	j_AGC_ch1	78
6.12.2.7	j_AGC_ch2	78
6.12.2.8	j_TR_gain_ch1	78
6.12.2.9	j_TR_gain_ch2	78
6.12.2.10	j_Tx_power	78
6.12.2.11	j_Dopp_rc	78
6.12.2.12	j_TR_inst_rc	78
6.12.2.13	j_R_inst_rc	78
6.12.2.14	j_TR_inst_gain_C	79
6.12.2.15	j_R_inst_gain_C	79
6.12.2.16	j_Int_phase_C	79
6.12.2.17	j_Ext_phase_C	79
6.12.2.18	j_Noise_pwr	79
6.12.2.19	j_Phase_slope_C	79
6.12.2.20	spares	79
6.13	t_cs_fileinfo Struct Reference	80
6.13.1	Detailed Description	80
6.13.2	Field Documentation	80
6.13.2.1	pt_filepointer	80
6.13.2.2	j_num_datasets	80
6.13.2.3	pat_data_start_offsets	80
6.13.2.4	paj_num_records	81
6.13.2.5	j_type	81

6.13.2.6	j_baseline	81
6.14	TimeAndOrbitGroup Struct Reference	81
6.14.1	Detailed Description	82
6.14.2	Field Documentation	82
6.14.2.1	j_Day	82
6.14.2.2	uj_Sec	82
6.14.2.3	uj_Micsec	82
6.14.2.4	j_USO_Corr	82
6.14.2.5	ui_Mode_ID	82
6.14.2.6	ui_SSC	83
6.14.2.7	uj_Inst_config	83
6.14.2.8	uj_Rec_Count	83
6.14.2.9	j_Lat	83
6.14.2.10	j_Lon	83
6.14.2.11	j_Alt	83
6.14.2.12	j_Alt_rate	83
6.14.2.13	aj_Sat_velocity	83
6.14.2.14	aj_Real_beam	84
6.14.2.15	aj_Baseline	84
6.14.2.16	i_ST_ID	84
6.14.2.17	j_Roll_angle	84
6.14.2.18	j_Pitch_angle	84
6.14.2.19	j_Yaw_angle	84
6.14.2.20	uj_MCD	84
6.15	WaveformsLRM Struct Reference	85
6.15.1	Detailed Description	85
6.15.2	Field Documentation	85
6.15.2.1	pui_wfm	85
6.15.2.2	j_Linear_wfm_mult	85
6.15.2.3	j_Power2_wfm_mult	85
6.15.2.4	ui_num_avg_echoes	86
6.15.2.5	ui_flags	86
6.16	WaveformsSAR Struct Reference	86
6.16.1	Detailed Description	86
6.16.2	Field Documentation	87
6.16.2.1	pui_wfm	87
6.16.2.2	j_Linear_wfm_mult	87
6.16.2.3	j_Power2_wfm_mult	87
6.16.2.4	ui_num_avg_echoes	87
6.16.2.5	ui_flags	87

6.16.2.6	ai_bb_param	87
6.17	WaveformsSARIN Struct Reference	87
6.17.1	Detailed Description	88
6.17.2	Field Documentation	88
6.17.2.1	pui_wfm	88
6.17.2.2	j_Linear_wfm_mult	88
6.17.2.3	j_Power2_wfm_mult	88
6.17.2.4	ui_num_avg_echoes	89
6.17.2.5	ui_flags	89
6.17.2.6	ai_bb_param	89
6.17.2.7	pui_coherence	89
6.17.2.8	pj_phase_diff	89
<b>7</b>	<b>File Documentation</b>	<b>91</b>
7.1	BitmaskUtilities.c File Reference	91
7.1.1	Detailed Description	92
7.1.2	Function Documentation	92
7.1.2.1	vClearBitLong(uint32_t *pBitmask, uint16_t bit)	92
7.2	BitmaskUtilities.h File Reference	92
7.2.1	Detailed Description	93
7.2.2	Function Documentation	93
7.2.2.1	vClearBitLong(uint32_t *pBitmask, uint16_t bit)	93
7.3	CS_Defines.c File Reference	93
7.3.1	Detailed Description	94
7.3.2	Variable Documentation	94
7.3.2.1	as_CS_DS_NAME	94
7.3.2.2	gj_verbose	94
7.3.2.3	J_CS_L2_RECORD_SIZE	94
7.3.2.4	J_CS_L2I_RECORD_SIZE	94
7.4	CS_Defines.h File Reference	95
7.4.1	Detailed Description	95
7.4.2	Macro Definition Documentation	96
7.4.2.1	J_MPH_SIZE	96
7.4.2.2	TRUE	96
7.4.2.3	FALSE	96
7.4.3	Enumeration Type Documentation	96
7.4.3.1	CS_DS_NAME	96
7.4.4	Variable Documentation	96
7.4.4.1	as_CS_DS_NAME	96
7.4.4.2	J_CS_L2_RECORD_SIZE	96

7.4.4.3	J_CS_L2I_RECORD_SIZE	96
7.4.4.4	gj_verbose	96
7.5	Documentation.h File Reference	97
7.5.1	Detailed Description	97
7.6	jCSGetDSFileOffsetFromHeader.c File Reference	97
7.6.1	Detailed Description	97
7.6.2	Function Documentation	97
7.6.2.1	jCheckDSDetails(char *ac_buffer, long int j_DSD_Size)	97
7.7	jCSGetDSFileOffsetFromHeader.h File Reference	98
7.7.1	Detailed Description	99
7.8	jCSNumRecordsInDataset.c File Reference	99
7.8.1	Detailed Description	99
7.9	jCSNumRecordsInDataset.h File Reference	99
7.9.1	Detailed Description	100
7.10	jFillL1bStructFromFile.c File Reference	100
7.10.1	Detailed Description	101
7.11	jFillL1bStructFromFile.h File Reference	101
7.11.1	Detailed Description	102
7.12	jFillL2IStructFromFile.c File Reference	102
7.12.1	Detailed Description	103
7.13	jFillL2IStructFromFile.h File Reference	103
7.13.1	Detailed Description	104
7.14	jFillL2StructFromFile.c File Reference	104
7.14.1	Detailed Description	105
7.15	jFillL2StructFromFile.h File Reference	105
7.15.1	Detailed Description	106
7.16	L1bStructure.h File Reference	106
7.16.1	Detailed Description	108
7.16.2	Macro Definition Documentation	108
7.16.2.1	J_NUM_BB_PARAMETERS	108
7.16.2.2	J_NUM_HRB	108
7.16.3	Typedef Documentation	108
7.16.3.1	t_Level1bAvgWfmSARinFlagBit	108
7.16.3.2	t_Level1bWfmSARinFlagBit	108
7.16.4	Enumeration Type Documentation	108
7.16.4.1	t_Level1bAvgWfmLRMFlagBit	108
7.16.4.2	t_Level1bAvgWfmSARFlagBit	109
7.16.4.3	t_Level1bWfmLRMFlagBit	109
7.16.4.4	t_Level1bWfmSARFlagBit	109
7.16.5	Function Documentation	110

7.16.5.1	<a href="#">vDump_L1B_LRM(L1B_LRM *pt, FILE *pf)</a>	110
7.16.5.2	<a href="#">vDump_L1B_SAR(L1B_SAR *pt, FILE *pf)</a>	110
7.16.5.3	<a href="#">vDump_L1B_SARIN(L1B_SARIN *pt, FILE *pf)</a>	110
7.16.6	Variable Documentation	110
7.16.6.1	<a href="#">J_LRM_WFM_BINS</a>	110
7.16.6.2	<a href="#">J_SAR_WFM_BINS</a>	110
7.16.6.3	<a href="#">J_SARIN_WFM_BINS</a>	110
7.16.6.4	<a href="#">J_CS_L1B_LRM_RECORD_SIZE</a>	111
7.16.6.5	<a href="#">J_CS_L1B_SAR_RECORD_SIZE</a>	111
7.16.6.6	<a href="#">J_CS_L1B_SARIN_RECORD_SIZE</a>	111
7.17	L2IStructure.c File Reference	111
7.17.1	Detailed Description	111
7.17.2	Function Documentation	111
7.17.2.1	<a href="#">vDump_L2IData(L2IData *pt, FILE *pf)</a>	111
7.18	L2IStructure.h File Reference	112
7.18.1	Detailed Description	114
7.18.2	Macro Definition Documentation	114
7.18.2.1	<a href="#">J_CS_L2I_STRUCT_SIZE</a>	114
7.18.3	Enumeration Type Documentation	114
7.18.3.1	<a href="#">t_InterpolationErrorFlagBit</a>	114
7.18.3.2	<a href="#">t_MeasurementMode</a>	114
7.18.3.3	<a href="#">t_MeasurementQualityFlagBit</a>	114
7.18.3.4	<a href="#">t_RetrackerFlagBit</a>	115
7.18.3.5	<a href="#">t_HeightStatusFlagBit</a>	115
7.18.3.6	<a href="#">t_FreeboardStatusFlagBit</a>	116
7.18.3.7	<a href="#">t_DiscriminatorResult</a>	117
7.18.3.8	<a href="#">t_DiscriminatorStatusFlagBit</a>	117
7.18.3.9	<a href="#">t_Level2AmbiguityErrorFlagBit</a>	118
7.18.4	Function Documentation	118
7.18.4.1	<a href="#">vDump_L2IData(L2IData *pt, FILE *pf)</a>	118
7.19	L2Structure.c File Reference	118
7.19.1	Detailed Description	119
7.19.2	Function Documentation	119
7.19.2.1	<a href="#">vDump_L2FinalC(L2FinalC *pt, FILE *pf)</a>	119
7.19.2.2	<a href="#">vDump_L2FinalM(L2FinalM *pt, FILE *pf)</a>	119
7.19.2.3	<a href="#">vDump_L2Data(L2Data *pt, FILE *pf)</a>	120
7.20	L2Structure.h File Reference	120
7.20.1	Detailed Description	122
7.20.2	Macro Definition Documentation	122
7.20.2.1	<a href="#">J_L2_STRUCT_SIZE</a>	122



7.20.2.2	J_L2_RECORD_SIZE	122
7.20.3	Enumeration Type Documentation	122
7.20.3.1	t_Level2QualityStatusFlagBit	122
7.20.3.2	t_Level2CorrectionsStatusFlagBit	123
7.20.3.3	t_Level2CorrectionsAppliedFlagBit	124
7.20.4	Function Documentation	124
7.20.4.1	vDump_L2Data(L2Data *pt, FILE *pf)	124
7.21	ptCSGetFileHandle.c File Reference	125
7.21.1	Detailed Description	126
7.22	ptCSGetFileHandle.h File Reference	126
7.22.1	Detailed Description	127
7.22.2	Typedef Documentation	127
7.22.2.1	t_cs_filehandle	127
7.22.3	Enumeration Type Documentation	127
7.22.3.1	BASELINE	127
7.23	ptCSGetL1b.c File Reference	127
7.23.1	Detailed Description	128
7.24	ptCSGetL1b.h File Reference	128
7.24.1	Detailed Description	129
7.25	ptCSGetL2.c File Reference	129
7.25.1	Detailed Description	130
7.26	ptCSGetL2.h File Reference	130
7.26.1	Detailed Description	131
7.27	ptCSGetL2l.c File Reference	131
7.27.1	Detailed Description	132
7.28	ptCSGetL2l.h File Reference	132
7.28.1	Detailed Description	133
7.29	SharedFlags.h File Reference	133
7.29.1	Detailed Description	134
7.29.2	Enumeration Type Documentation	134
7.29.2.1	t_MCDFlagBit	134
7.29.2.2	t_InstConf	135
7.29.2.3	t_ModelID	136
7.29.2.4	t_STID	137
7.29.2.5	t_SurfaceType	137
7.29.2.6	t_CorrectionStatusFlagBit	137
7.29.2.7	t_CorrectionErrorFlagBit	138
7.30	vCSFreeFileHandle.c File Reference	139
7.30.1	Detailed Description	139
7.31	vCSFreeFileHandle.h File Reference	139

---

7.31.1 Detailed Description . . . . .	140
<b>8 Example Documentation</b>	<b>141</b>
8.1 examples/cs_l1b_io_example.c . . . . .	141
8.2 examples/cs_l2_io_example.c . . . . .	146
8.3 examples/cs_l2i_io_example.c . . . . .	148
8.4 examples/example_bit_manipulation.c . . . . .	150
<b>Index</b>	<b>153</b>

# Chapter 1

## Library Overview

### 1.1 Introduction

This library contains C functions, and supporting header files, to read CryoSat Level 2 formatted files from disk into memory. The provided functions allow the user to create programs that read whole input files into memory or pick out specific records to be read.

CryoSat files come in a number of different formats that correspond to updates of the processing baseline (see <https://earth.esa.int/web/guest/missions/cryosat/ipf-baseline>). The correct baseline must be specified for the code to correctly ingest the data.

In addition to the C library, there is a library of IDL routines to read both Level 1b and Level 2 products, from both Baseline-B and Baseline-C. These may be found in the `idl` subdirectory of the distribution archive. See the [IDL Overview](#) subsection of the 'Module Documentation' section for more information.

The C library is split into six modules:

- Low Level API : Requires the user handle files at the system level and allows individual records to be read. There are two modules at this level, one each from Level 1b and Level2.
- High Level API : Hides all system interaction from the user and allows either entire files or a continuous section of a file to be read. There are three modules at this level, one each for Level 1b and Level 2, and a module of routines shared by both.
- Utilities : These functions are of use when processing the data.

The APIs (Application Program Interfaces) for these modules are documented in the section 'Module Documentation'.

Examples of how to use the C library functions are given in [8.1](#) and [8.3](#). Bitfield manipulation functions are shown in [8.4](#).

#### 1.1.1 Platform compatability and requirements

The library is distributed as a TAR file that has been compressed with the GZIP utility. You will need these utilities to unpack it.

The library has been built and successfully tested with the following combinations of platform, operating system, and compiler:

- CentOS 5 Linux with GCC 4.1.2
- Mac OSX 10.10 with Apple LLVM version 6.1.0

### 1.1.2 Installation Instructions

After unpacking the library, simply type `make` in the `c_code` subdirectory to build the libraries and examples. You should then copy the library and header files to the appropriate place on your system. A common example is to place the header files in `/usr/local/include` and the library files in `/usr/local/lib`.

### 1.1.3 Library Usage

To use the library, you must tell the C compiler where to find the include files and library files. If the library was installed at the default location, `/usr/local`, then the flags to pass to `gcc` are `-I/usr/local/include` during compilation and `-L/usr/local/lib -lmssl_cryosat -lmssl_shared` during linking.

### 1.1.4 Changes

Version 1.1 of the library has changes to allow use in 64-bit mode on 64-bit machines. Also, the variable `uj_↔Quality_F` in the L2 product structure is now (correctly) defined as a 32-bit (not a 16-bit) integer.

Version 2.0 of the library handles the different baselines of the product format up to Baseline C.

Version 2.1 aligns the IDL parameter names for the new products with previous releases and resolves a bug that prevented reading of L1b products.

Version 2.2 fixes an issue that resulted in the library being unable to handle Baseline-C L2 GDR files (only). This release also corrects some of the bit definitions for flag words to align with Baseline C.

Version 2.3 fixes an issue preventing the IDL code from reading Baseline C GDR files.

## Chapter 2

# Module Index

### 2.1 Modules

Here is a list of all modules:

High-level shared routines . . . . .	9
High Level CryoSat Level 1b File IO API . . . . .	12
Low-Level CryoSat Level 1b File IO API . . . . .	15
High Level CryoSat Level 2 File IO API . . . . .	17
Low-Level CryoSat Level 2 File IO API . . . . .	20
Utility functions . . . . .	22
IDL Routines . . . . .	26



## Chapter 3

# Data Structure Index

### 3.1 Data Structures

Here are the data structures with brief descriptions:

AveragedWaveformsLRM . . . . .	29
AveragedWaveformsSAR . . . . .	31
AveragedWaveformsSARIN . . . . .	33
Corrections . . . . .	36
L1B_LRM . . . . .	39
L1B_SAR . . . . .	40
L1B_SARIN . . . . .	41
L2Data . . . . .	43
L2FinalC . . . . .	46
L2FinalM . . . . .	50
L2IData . . . . .	54
Measurements . . . . .	76
t_cs_fileinfo . . . . .	80
TimeAndOrbitGroup . . . . .	81
WaveformsLRM . . . . .	85
WaveformsSAR . . . . .	86
WaveformsSARIN . . . . .	87





## Chapter 4

# File Index

### 4.1 File List

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

<a href="#">BitmaskUtilities.c</a>	
Implementation of functions to manipulate bitmasks . . . . .	91
<a href="#">BitmaskUtilities.h</a>	
Bitmask manipulation utilities . . . . .	92
<a href="#">CS_Defines.c</a>	
Defines and populates global variables . . . . .	93
<a href="#">CS_Defines.h</a>	
Contains types and macros needed by the functions in this library . . . . .	95
<a href="#">Documentation.h</a>	
Documentation for IDL commands . . . . .	97
<a href="#">jCSGetDSFileOffsetFromHeader.c</a>	
Implementation of functions to extract the offset of the data set from the ASCII header . . . . .	97
<a href="#">jCSGetDSFileOffsetFromHeader.h</a>	
Get the offset of a dataset from the header . . . . .	98
<a href="#">jCSNumRecordsInDataset.c</a>	
Implementation of function to retrieve number of records in a dataset within a file . . . . .	99
<a href="#">jCSNumRecordsInDataset.h</a>	
Function to retrieve number of records in a dataset within a file . . . . .	99
<a href="#">jFillL1bStructFromFile.c</a>	
Implementation of low-level functions to read Cryosat Level 1b data in LRM, SAR, or SARin modes . . . . .	100
<a href="#">jFillL1bStructFromFile.h</a>	
Low-level functions to read Cryosat Level 1b data in LRM, SAR, or SARin modes . . . . .	101
<a href="#">jFillL2IStructFromFile.c</a>	
Implementation of low-level functions to read Cryosat Level 2 interim data . . . . .	102
<a href="#">jFillL2IStructFromFile.h</a>	
Low-level functions to read Cryosat Level 2 interim data . . . . .	103
<a href="#">jFillL2StructFromFile.c</a>	
Implementation of low-level functions to read Cryosat Level 2 data . . . . .	104
<a href="#">jFillL2StructFromFile.h</a>	
Low-level functions to read Cryosat Level 2 data . . . . .	105
<a href="#">L1bStructure.h</a>	
Definition of the Level 1b structure . . . . .	106
<a href="#">L2IStructure.c</a>	
Convenience functions related to the Level 2i structure . . . . .	111
<a href="#">L2IStructure.h</a>	
Definition of the Level 2i structure . . . . .	112

<a href="#">L2Structure.c</a>	
Convenience functions for the L2 structure . . . . .	118
<a href="#">L2Structure.h</a>	
Structure and constant definitions for the Cryosat Level 2 format . . . . .	120
<a href="#">ptCSGetFileHandle.c</a>	
Implementation of a function to get a handle object representing a file . . . . .	125
<a href="#">ptCSGetFileHandle.h</a>	
Provide an opaque object that can store the information necessary to parse files . . . . .	126
<a href="#">ptCSGetL1b.c</a>	
Implementation of functions to read Cryosat Level 1b data files . . . . .	127
<a href="#">ptCSGetL1b.h</a>	
Functions to read Cryosat Level 1b data files . . . . .	128
<a href="#">ptCSGetL2.c</a>	
Implementation of functions to read Cryosat Level 2 data files . . . . .	129
<a href="#">ptCSGetL2.h</a>	
Functions to read Cryosat Level 2 data files . . . . .	130
<a href="#">ptCSGetL2i.c</a>	
Implementation of functions to read Cryosat Level 2 data files . . . . .	131
<a href="#">ptCSGetL2i.h</a>	
Functions to read Cryosat Level 2 data files . . . . .	132
<a href="#">SharedFlags.h</a>	
Enums for bitfield definitions and flags shared between L1b, L2, and L2i . . . . .	133
<a href="#">vCSFreeFileHandle.c</a>	
Implementation of functions to release file handles . . . . .	139
<a href="#">vCSFreeFileHandle.h</a>	
Functions to release file handles . . . . .	139

## Chapter 5

# Module Documentation

### 5.1 High-level shared routines

#### Functions

- [t\\_cs\\_filehandle](#) [ptCSGetFileHandle](#) (char \*[s\\_filename](#), [BASELINE](#) [j\\_baseline](#))
- long int [jCSNumRecordsInDataset](#) ([t\\_cs\\_filehandle](#) [t\\_handle](#), long int [j\\_dataset](#))
- void [vCSFreeFileHandle](#) ([t\\_cs\\_filehandle](#) [t\\_handle](#))
- long int [jCSGetDSFileOffsetFromHeader](#) (fpos\_t \*[pt\\_dataset\\_file\\_offset](#), long int \*[pj\\_num\\_records](#), char \*[s\\_filename](#), [CS\\_DS\\_NAME](#) [t\\_ds\\_name](#), long int \*[pj\\_actual\\_ds](#))

#### 5.1.1 Detailed Description

Use of these functions is necessary when calling the high-level APIs to read either Level 1b or Level 2 data.

#### 5.1.2 Function Documentation

##### 5.1.2.1 [t\\_cs\\_filehandle](#) [ptCSGetFileHandle](#) ( char \* [s\\_filename](#), [BASELINE](#) [j\\_baseline](#) )

Obtain a handle to the file with the given filename that is needed to call the other functions in this library.

#### Parameters

<a href="#">s_filename</a>	The name of the file to obtain a handle upon.
<a href="#">j_baseline</a>	The baseline (version) of the product format expected.

#### Returns

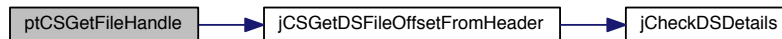
[t\\_cs\\_filehandle](#) Equal to NULL if a handle could not be obtained.

#### Examples:

[examples/cs\\_l1b\\_io\\_example.c](#), [examples/cs\\_l2\\_io\\_example.c](#), [examples/cs\\_l2i\\_io\\_example.c](#), and [examples/example\\_bit\\_manipulation.c](#).

Definition at line 98 of file [ptCSGetFileHandle.c](#).

Here is the call graph for this function:



#### 5.1.2.2 long int jCSNumRecordsInDataset ( t\_cs\_filehandle t\_handle, long int j\_dataset )

Retrieve the number of records in a certain dataset within a file from a handle attached to that file.

##### Parameters

<i>t_handle</i>	A handle to the file to examine.
<i>j_dataset</i>	The index of the dataset, starting at zero, to query.

##### Returns

The number of records in that dataset. The value -1 represents bad parameters passed to the function.

##### Examples:

[examples/cs\\_l1b\\_io\\_example.c](#), [examples/cs\\_l2\\_io\\_example.c](#), [examples/cs\\_l2i\\_io\\_example.c](#), and [examples/example\\_bit\\_manipulation.c](#).

Definition at line 81 of file jCSNumRecordsInDataset.c.

#### 5.1.2.3 void vCSFreeFileHandle ( t\_cs\_filehandle t\_handle )

Indicate that a file handle is no longer needed.

Calling this function indicates to the library that the given file handle is no longer needed. This means that all memory allocated by the file handle may be released and that the file can be closed.

##### Warning

The file handle must not be re-used once this function has been called.

##### Parameters

<i>t_handle</i>	The file handle to release.
-----------------	-----------------------------

##### Examples:

[examples/cs\\_l1b\\_io\\_example.c](#), [examples/cs\\_l2\\_io\\_example.c](#), and [examples/cs\\_l2i\\_io\\_example.c](#).

Definition at line 84 of file vCSFreeFileHandle.c.

#### 5.1.2.4 long int jCSGetDSFileOffsetFromHeader ( fpos\_t \* pt\_dataset\_file\_offset, long int \* pj\_num\_records, char \* s\_filename, CS\_DS\_NAME t\_ds\_name, long int \* pj\_actual\_ds )

Get the offset of a dataset from the header.

This function retrieves the offset into a file at which a specific dataset mentioned in the ASCII header can be found. If both of the output parameters are NULL, this function can be used to check for the presence of a specific type of dataset within a file.

## Parameters

<i>pt_dataset_file↔ _offset</i>	Pointer to an fpos_t into which the offset will be written. May be NULL.
<i>pj_num_records</i>	Pointer to an integer into which the number of records in the dataset will be written. May be NULL.
<i>s_filename</i>	The filename of the file to query.
<i>t_ds_name</i>	An enumerated type describing the type of data to look for in the file.
<i>pj_actual_ds</i>	Filled with the type of DS found. This is made as generic as possible e.g. CS_L2_DS_NAME↔ ME_ANY rather than CS_L2_DS_NAME_LRM.

## Returns

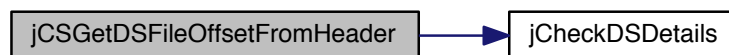
EXIT\_SUCCESS if all went well. EXIT\_FAILURE otherwise.

## Examples:

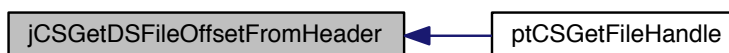
[examples/cs\\_l1b\\_io\\_example.c](#), and [examples/example\\_bit\\_manipulation.c](#).

Definition at line 91 of file jCSGetDSFileOffsetFromHeader.c.

Here is the call graph for this function:



Here is the caller graph for this function:



## 5.2 High Level CryoSat Level 1b File IO API

### Functions

- **L1B\_LRM** \* [ptCSGetL1bLRM](#) ([t\\_cs\\_filehandle](#) t\_handle, long int j\_dataset\_index, unsigned long long int uk\_num\_records, unsigned long long int uk\_skip\_records, **L1B\_LRM** \*pt\_data)
- **L1B\_SAR** \* [ptCSGetL1bSAR](#) ([t\\_cs\\_filehandle](#) t\_handle, long int j\_dataset\_index, unsigned long long int uk\_num\_records, unsigned long long int uk\_skip\_records, **L1B\_SAR** \*pt\_data)
- **L1B\_SARIN** \* [ptCSGetL1bSARIN](#) ([t\\_cs\\_filehandle](#) t\_handle, long int j\_dataset\_index, unsigned long long int uk\_num\_records, unsigned long long int uk\_skip\_records, **L1B\_SARIN** \*pt\_data)

### 5.2.1 Detailed Description

These functions comprise the high-level API for reading from CryoSat Level 1b format files in LRM, SAR, and SARin modes.

These functions correctly handle both complete files (with MPH and SPH) and interim files (raw data).

### 5.2.2 Function Documentation

**5.2.2.1** **L1B\_LRM**\* [ptCSGetL1bLRM](#) ( [t\\_cs\\_filehandle](#) t\_handle, long int j\_dataset\_index, unsigned long long int uk\_num\_records, unsigned long long int uk\_skip\_records, **L1B\_LRM** \* pt\_data )

Read records from a Cryosat Level 1b LRM mode formatted file.

This function may be called to read one or more records from a Cryosat Level 1b LRM file. Both complete and interim files are supported (files with or without an MPH and SPH). It is possible to request that reading starts at some distance into the file and also to read only part of the file. Memory to hold the records may either be allocated in advance or by the function.

#### Parameters

<i>t_handle</i>	A handle to the file returned from a call to <a href="#">ptCSGetFileHandle()</a> .
<i>j_dataset_index</i>	The index (from zero) of the dataset to return. Must be zero at present.
<i>uk_num_records</i>	The number of records to read from the file. If zero, read the whole file.
<i>uk_skip_records</i>	The number of records to skip at the start of the file.
<i>pt_data</i>	A pointer to one or more structures of type <b>L1B_LRM</b> . If this pointer is non-NULL then the memory pointed to will be filled with the data read. If this pointer is NULL then memory will be allocated by the function and it is the calling routines responsibility to free that memory with a call to <a href="#">free()</a> .

#### Returns

A pointer to the memory that has been filled with the data read from the file, or NULL in the case of error. If pt\_data is non-NULL then the return value is pt\_data. Otherwise it is a pointer to newly allocated memory.

#### Examples:

[examples/cs\\_l1b\\_io\\_example.c](#), and [examples/example\\_bit\\_manipulation.c](#).

Definition at line 88 of file [ptCSGetL1b.c](#).

Here is the call graph for this function:



**5.2.2.2 L1B\_SAR\*** `ptCSGetL1bSAR ( t_cs_filehandle t_handle, long int j_dataset_index, unsigned long long int uk_num_records, unsigned long long int uk_skip_records, L1B_SAR * pt_data )`

Read records from a Cryosat Level 1b SAR mode formatted file.

This function may be called to read one or more records from a Cryosat Level 1b SAR file. Both complete and interim files are supported (files with or without an MPH and SPH). It is possible to request that reading starts at some distance into the file and also to read only part of the file. Memory to hold the records may either be allocated in advance or by the function.

#### Parameters

<i>t_handle</i>	A handle to the file returned from a call to <a href="#">ptCSGetFileHandle()</a> .
<i>j_dataset_index</i>	The index (from zero) of the dataset to return. Must be zero at present.
<i>uk_num_records</i>	The number of records to read from the file. If zero, read the whole file.
<i>uk_skip_records</i>	The number of records to skip at the start of the file.
<i>pt_data</i>	A pointer to one or more structures of type <a href="#">L1B_SAR</a> . If this pointer is non-NULL then the memory pointed to will be filled with the data read. If this pointer is NULL then memory will be allocated by the function and it is the calling routines responsibility to free that memory with a call to <code>free()</code> .

#### Returns

A pointer to the memory that has been filled with the data read from the file, or NULL in the case of error. If `pt_data` is non-NULL then the return value is `pt_data`. Otherwise it is a pointer to newly allocated memory.

#### Examples:

[examples/cs\\_l1b\\_io\\_example.c](#).

Definition at line 216 of file `ptCSGetL1b.c`.

Here is the call graph for this function:



**5.2.2.3 L1B\_SARIN\*** `ptCSGetL1bSARIN ( t_cs_filehandle t_handle, long int j_dataset_index, unsigned long long int uk_num_records, unsigned long long int uk_skip_records, L1B_SARIN * pt_data )`

Read records from a Cryosat Level 1b SARin mode formatted file.

This function may be called to read one or more records from a Cryosat Level 1b SARin file. Both complete and interim files are supported (files with or without an MPH and SPH). It is possible to request that reading starts at some distance into the file and also to read only part of the file. Memory to hold the records may either be allocated in advance or by the function.

#### Parameters

<i>t_handle</i>	A handle to the file returned from a call to <a href="#">ptCSGetFileHandle()</a> .
<i>j_dataset_index</i>	The index (from zero) of the dataset to return. Must be zero at present.
<i>uk_num_records</i>	The number of records to read from the file. If zero, read the whole file.
<i>uk_skip_records</i>	The number of records to skip at the start of the file.
<i>pt_data</i>	A pointer to one or more structures of type <a href="#">L1B_SARIN</a> . If this pointer is non-NULL then the memory pointed to will be filled with the data read. If this pointer is NULL then memory will be allocated by the function and it is the calling routines responsibility to free that memory with a call to <code>free()</code> .

#### Returns

A pointer to the memory that has been filled with the data read from the file or NULL in the case of error. If `pt_data` is non-NULL then the return value is `pt_data`. Otherwise it is a pointer to newly allocated memory.

#### Examples:

[examples/cs\\_l1b\\_io\\_example.c](#).

Definition at line 338 of file `ptCSGetL1b.c`.

Here is the call graph for this function:





## 5.3 Low-Level CryoSat Level 1b File IO API

### Functions

- long int `jFillL1bLRMStructFromFile` (`L1B_LRM` \*`p_ToFill`, FILE \*`pt_input`, `BASELINE` `j_Baseline`)
- long int `jFillL1bSARStructFromFile` (`L1B_SAR` \*`p_ToFill`, FILE \*`pt_input`, `BASELINE` `j_Baseline`)
- long int `jFillL1bSARINStructFromFile` (`L1B_SARIN` \*`p_ToFill`, FILE \*`pt_input`, `BASELINE` `j_Baseline`)

### 5.3.1 Detailed Description

These functions comprise the low-level API for reading from CryoSat Level 1b format files in LRM, SAR, and SARin modes.

Use of these functions requires the calling routine to do all of the manipulation necessary to create a FILE\* pointer that is positioned to the start of a Level 1b record on the disk. The functions simply read that record into an area of memory that must be allocated by the calling routine.

### 5.3.2 Function Documentation

#### 5.3.2.1 long int `jFillL1bLRMStructFromFile` ( `L1B_LRM` \* `p_ToFill`, FILE \* `pt_input`, `BASELINE` `j_Baseline` )

Read a Cryosat Level 1b LRM mode record from disk into a pre-allocated structure.

#### Parameters

<code>p_ToFill</code>	Pointer to a pre-allocated structure to fill.
<code>pt_input</code>	File pointer to the record to read.
<code>j_Baseline</code>	Baseline (version) of the product format expected.

#### Returns

EXIT\_SUCCESS if the record was correctly read, EXIT\_FAILURE otherwise.

Definition at line 155 of file `jFillL1bStructFromFile.c`.

Here is the caller graph for this function:



#### 5.3.2.2 long int `jFillL1bSARStructFromFile` ( `L1B_SAR` \* `p_ToFill`, FILE \* `pt_input`, `BASELINE` `j_Baseline` )

Read a Cryosat Level 1b SAR mode record from disk into a pre-allocated structure.

## Parameters

<i>p_ToFill</i>	Pointer to a pre-allocated structure to fill.
<i>pt_input</i>	File pointer to the record to read.
<i>j_Baseline</i>	Baseline (version) of the product format expected.

## Returns

EXIT\_SUCCESS if the record was correctly read, EXIT\_FAILURE otherwise.

Definition at line 1062 of file jFillL1bStructFromFile.c.

Here is the caller graph for this function:



### 5.3.2.3 long int jFillL1bSARINStructFromFile ( L1B\_SARIN \* *p\_ToFill*, FILE \* *pt\_input*, BASELINE *j\_Baseline* )

Read a Cryosat Level 1b SARin mode record from disk into a pre-allocated structure.

## Parameters

<i>p_ToFill</i>	Pointer to a pre-allocated structure to fill.
<i>pt_input</i>	File pointer to the record to read.
<i>j_Baseline</i>	Baseline (version) of the product format expected.

## Returns

EXIT\_SUCCESS if the record was correctly read, EXIT\_FAILURE otherwise.

Definition at line 1984 of file jFillL1bStructFromFile.c.

Here is the caller graph for this function:



## 5.4 High Level CryoSat Level 2 File IO API

### Functions

- `L2Data * ptCSGetL2 ( t_cs_filehandle t_handle, long int j_dataset_index, unsigned long long int uk_num_records, unsigned long long int uk_skip_records, L2Data *pt_data)`
- `L2IData * ptCSGetL2I ( t_cs_filehandle t_handle, long int j_dataset_index, unsigned long long int uk_num_records, unsigned long long int uk_skip_records, L2IData *pt_data)`

### 5.4.1 Detailed Description

These functions comprise the high-level API for reading from CryoSat Level 2 format files.

These functions correctly handle both complete files (with MPH and SPH) and interim files (raw data).

### 5.4.2 Function Documentation

**5.4.2.1** `L2Data* ptCSGetL2 ( t_cs_filehandle t_handle, long int j_dataset_index, unsigned long long int uk_num_records, unsigned long long int uk_skip_records, L2Data * pt_data )`

Read records from a Cryosat Level 2 formatted file.

This function may be called to read one or more records from a Cryosat Level 2 file. Both complete and interim files are supported (files with or without an MPH and SPH). It is possible to request that reading starts at some distance into the file and also to read only part of the file. Memory to hold the records may either be allocated in advance or by the function.

#### Parameters

<i>t_handle</i>	A handle to the file returned from a call to <code>ptCSGetFileHandle()</code> .
<i>j_dataset_index</i>	The index (from zero) of the dataset to return. Must be zero at present.
<i>uk_num_records</i>	The number of records to read from the file. If zero, read the whole file.
<i>uk_skip_records</i>	The number of records to skip at the start of the file.
<i>pt_data</i>	A pointer to one or more structures of type <code>L2Data</code> . If this pointer is non-NULL then the memory pointed to will be filled with the data read. If this pointer is NULL then memory will be allocated by the function and it is the calling routines responsibility to free that memory with a call to <code>free()</code> .

**Returns**

A pointer to the memory that has been filled with the data read from the file, or NULL in the case of error. If `pt_data` is non-NULL then the return value is `pt_data`. Otherwise it is a pointer to newly allocated memory.

**Examples:**

[examples/cs\\_l2\\_io\\_example.c](#).

Definition at line 85 of file `ptCSGetL2.c`.

Here is the call graph for this function:



**5.4.2.2** `L2IData* ptCSGetL2I( t_cs_filehandle t_handle, long int j_dataset_index, unsigned long long int uk_num_records, unsigned long long int uk_skip_records, L2IData * pt_data )`

Read records from a Cryosat Level 2 Intermediate formatted file.

This function may be called to read one or more records from a Cryosat Level 2 Intermediate file. Both complete and interim files are supported (files with or without an MPH and SPH). It is possible to request that reading starts at some distance into the file and also to read only part of the file. Memory to hold the records may either be allocated in advance or by the function.

**Parameters**

<i>t_handle</i>	A handle to the file returned from a call to <a href="#">ptCSGetFileHandle()</a> .
<i>j_dataset_index</i>	The index (from zero) of the dataset to return. Must be zero at present.
<i>uk_num_records</i>	The number of records to read from the file. If zero, read the whole file.
<i>uk_skip_records</i>	The number of records to skip at the start of the file.
<i>pt_data</i>	A pointer to one or more structures of type <a href="#">L2IData</a> . If this pointer is non-NULL then the memory pointed to will be filled with the data read. If this pointer is NULL then memory will be allocated by the function and it is the calling routines responsibility to free that memory with a call to <code>free()</code> .

**Returns**

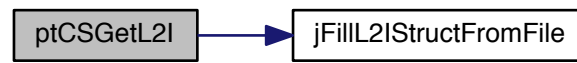
A pointer to the memory that has been filled with the data read from the file, or NULL in the case of error. If `pt_data` is non-NULL then the return value is `pt_data`. Otherwise it is a pointer to newly allocated memory.

**Examples:**

[examples/cs\\_l2i\\_io\\_example.c](#).

Definition at line 85 of file `ptCSGetL2I.c`.

Here is the call graph for this function:



## 5.5 Low-Level CryoSat Level 2 File IO API

### Functions

- long int [jFillL2StructFromFile](#) ([L2Data](#) \*pt\_Level2, FILE \*pt\_File, [BASELINE](#) j\_Baseline)
- long int [jFillL2IStructFromFile](#) ([L2IData](#) \*pt\_Level2, FILE \*pt\_File, [BASELINE](#) j\_Baseline)

#### 5.5.1 Detailed Description

These functions comprise the low-level API for reading from CryoSat Level 2 format files.

Use of these functions requires the calling routine to do all of the manipulation necessary to create a FILE\* pointer that is positioned to the start of a Level 2 record on the disk. The functions simply read that record into an area of memory that must be allocated by the calling routine.

#### 5.5.2 Function Documentation

##### 5.5.2.1 long int jFillL2StructFromFile ( [L2Data](#) \* *pt\_Level2*, FILE \* *pt\_File*, [BASELINE](#) *j\_Baseline* )

Read one record of Cryosat Level 2 data from a file.

##### Parameters

<i>pt_Level2</i>	Pointer to a <a href="#">L2Data</a> struct to fill with data.
<i>pt_File</i>	Pointer to the file to read from.
<i>j_Baseline</i>	Baseline (version) of the product format expected.

##### Returns

If reading is successful EXIT\_SUCCESS, otherwise EXIT\_FAILURE.

Definition at line 93 of file [jFillL2StructFromFile.c](#).

Here is the caller graph for this function:



##### 5.5.2.2 long int jFillL2IStructFromFile ( [L2IData](#) \* *pt\_Level2*, FILE \* *pt\_File*, [BASELINE](#) *j\_Baseline* )

Read one record of Cryosat Level 2 interim data from a file.

##### Parameters

<i>pt_Level2</i>	Pointer to a <a href="#">L2IData</a> struct to fill with data.
------------------	--

<i>pt_File</i>	Pointer to the file to read from.
<i>j_Baseline</i>	Baseline (version) of the product format expected.

#### Returns

If reading is successful EXIT\_SUCCESS, otherwise EXIT\_FAILURE.

Definition at line 92 of file jFillL2IStructFromFile.c.

Here is the caller graph for this function:



## 5.6 Utility functions

### Functions

- void [vSetBitShort](#) (uint16\_t \*pBitmask, uint16\_t bit)
- int16\_t [iCheckBitShort](#) (uint16\_t \*pBitmask, uint16\_t bit)
- void [vClearBitShort](#) (uint16\_t \*pBitmask, uint16\_t bit)
- void [vSetBitLong](#) (uint32\_t \*pBitmask, uint16\_t bit)
- int16\_t [iCheckBitLong](#) (uint32\_t \*pBitmask, uint16\_t bit)
- void [vSetBitLongLong](#) (uint64\_t \*pBitmask, uint16\_t bit)
- int16\_t [iCheckBitLongLong](#) (uint64\_t \*pBitmask, uint16\_t bit)
- void [vClearBitLongLong](#) (uint64\_t \*pBitmask, uint16\_t bit)

#### 5.6.1 Detailed Description

These are useful functions to aid in the processing of data.

These functions are not CryoSat specific. They are utility functions that may be of use when processing data read by the main functions of this library.

#### 5.6.2 Function Documentation

##### 5.6.2.1 void vSetBitShort ( uint16\_t \* *pBitmask*, uint16\_t *bit* )

Set a bit in a int16\_teger bitmask.

###### Parameters

<i>pBitmask</i>	A pointer to the bitmask.
<i>bit</i>	The bit index to set. Bit 0 is the LSB.

###### Examples:

[examples/example\\_bit\\_manipulation.c](#).

Definition at line 109 of file BitmaskUtilities.c.

##### 5.6.2.2 int16\_t iCheckBitShort ( uint16\_t \* *pBitmask*, uint16\_t *bit* )

Test the value of a bit in a int16\_teger bitmask.

###### Parameters

<i>pBitmask</i>	A pointer to the bitmask.
<i>bit</i>	The bit index to test. Bit 0 is the LSB.

###### Returns

true if the bit is set. false otherwise.

###### Examples:

[examples/example\\_bit\\_manipulation.c](#).

Definition at line 119 of file BitmaskUtilities.c.



5.6.2.3 void vClearBitShort ( uint16\_t\* *pBitmask*, uint16\_t *bit* )

Clear a bit in a int16\_teger bitmask.

## Parameters

<i>pBitmask</i>	A pointer to the bitmask.
<i>bit</i>	The bit index to clear. Bit 0 is the LSB.

## Examples:

[examples/example\\_bit\\_manipulation.c](#).

Definition at line 129 of file BitmaskUtilities.c.

**5.6.2.4** void vSetBitLong ( uint32\_t \* *pBitmask*, uint16\_t *bit* )

Set a bit in a int32\_teger bitmask.

## Parameters

<i>pBitmask</i>	A pointer to the bitmask.
<i>bit</i>	The bit index to set. Bit 0 is the LSB.

Definition at line 139 of file BitmaskUtilities.c.

**5.6.2.5** int16\_t iCheckBitLong ( uint32\_t \* *pBitmask*, uint16\_t *bit* )

Test the value of a bit in a int32\_teger bitmask.

## Parameters

<i>pBitmask</i>	A pointer to the bitmask.
<i>bit</i>	The bit index to test. Bit 0 is the LSB.

## Returns

true if the bit is set. false otherwise.

## Examples:

[examples/cs\\_l1b\\_io\\_example.c](#).

Definition at line 149 of file BitmaskUtilities.c.

**5.6.2.6** void vSetBitLongLong ( uint64\_t \* *pBitmask*, uint16\_t *bit* )

Set a bit in a long int32\_teger bitmask.

## Parameters

<i>pBitmask</i>	A pointer to the bitmask.
<i>bit</i>	The bit index to set. Bit 0 is the LSB.

Definition at line 171 of file BitmaskUtilities.c.

**5.6.2.7** int16\_t iCheckBitLongLong ( uint64\_t \* *pBitmask*, uint16\_t *bit* )

Test the value of a bit in a long int32\_teger bitmask.

## Parameters

<i>pBitmask</i>	A pointer to the bitmask.
<i>bit</i>	The bit index to test. Bit 0 is the LSB.

## Returns

true if the bit is set. false otherwise.

Definition at line 181 of file BitmaskUtilities.c.

**5.6.2.8 void vClearBitLongLong ( uint64\_t \* *pBitmask*, uint16\_t *bit* )**

Clear a bit in a long int32\_teger bitmask.

## Parameters

<i>pBitmask</i>	A pointer to the bitmask.
<i>bit</i>	The bit index to clear. Bit 0 is the LSB.

Definition at line 191 of file BitmaskUtilities.c.

## 5.7 IDL Routines

IDL routines to read Level 1b and Level 2 products.

### 5.7.1 IDL Overview

A number of IDL routines are provided in the `idl` directory that allow the reading of Level 1b and Level 2 formatted products into IDL. When a file format changes due to a new baseline being released, a new IDL routine is added with the baseline in the function name. For example for Baseline-C `cs_l2i_read()` becomes `cs_l2i_c_read()`.

The complete list of routines currently present and the Baselines that they read is:

- **`cs_l1b_read.pro`** reads Baseline A and B L1B files
- **`cs_l1b_c_read.pro`** reads Baseline BC and C L1B files
- **`cs_l2i_read.pro`** reads Baseline A and B L2I files
- **`cs_l2i_read_bc.pro`** reads Baseline BC L2I files
- **`cs_l2i_read_c.pro`** reads Baseline C L2I files
- **`cs_l2_read.pro`** reads Baseline A, B and BC L2 files
- **`cs_l2_c_read.pro`** reads Baseline C L2 files

### 5.7.2 `cs_l1b_read`

This function will read a Level 1b formatted product into IDL.

#### Usage (for a SARIN product):

Note that `mode` is an optional parameter. If not provided an attempt will be made to detect it from the filename.

```
IDL> t_l1b = cs_l1b_read( s_filename, mode='SARIN' )
```

#### Parameters:

- `s_filename` : The filename of the product.
- `mode` : The mode of the product. 'LRM', 'SAR', or 'SARIN'.

#### Returns:

- `t_l1b` : A structure containing the product.

### 5.7.3 `cs_l2_read`

This function will read a Level 2 formatted product into IDL.

#### Usage:

```
IDL> t_l2 = cs_l2_read( s_filename )
```

#### Parameters:

- `s_filename` : The filename of the product.

#### Returns:

- `t_l2` : A structure containing the product.

### 5.7.4 cs\_l2i\_read

This function will read a Level 2 interim formatted product into IDL.

**Usage:**

```
IDL> t_l2i = cs_l2i_read( s_filename )
```

**Parameters:**

- s\_filename : The filename of the product.

**Returns:**

- t\_l2i : A structure containing the product.



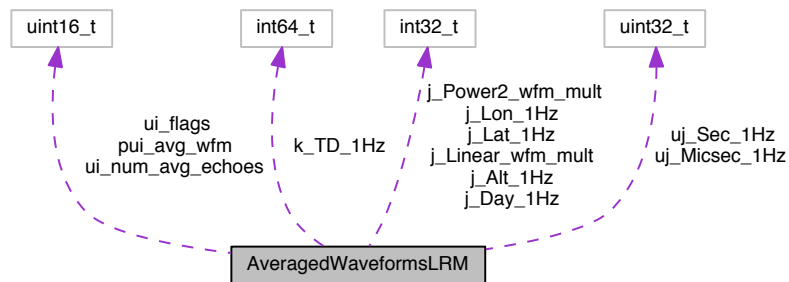
## Chapter 6

# Data Structure Documentation

### 6.1 AveragedWaveformsLRM Struct Reference

```
#include <L1bStructure.h>
```

Collaboration diagram for AveragedWaveformsLRM:



#### Data Fields

- `int32_t j_Day_1Hz`
- `uint32_t uj_Sec_1Hz`
- `uint32_t uj_Micsec_1Hz`
- `int32_t j_Lat_1Hz`
- `int32_t j_Lon_1Hz`
- `int32_t j_Alt_1Hz`
- `int64_t k_TD_1Hz`
- `uint16_t * pui_avg_wfm`
- `int32_t j_Linear_wfm_mult`
- `int32_t j_Power2_wfm_mult`
- `uint16_t ui_num_avg_echoes`
- `uint16_t ui_flags`

#### 6.1.1 Detailed Description

Level 1b record substructure (LRM mode specific) containing averaged waveform data.

Definition at line 339 of file L1bStructure.h.

## 6.1.2 Field Documentation

### 6.1.2.1 `int32_t AveragedWaveformsLRM::j_Day_1Hz`

MDSR time stamp days.

Definition at line 341 of file L1bStructure.h.

### 6.1.2.2 `uint32_t AveragedWaveformsLRM::uj_Sec_1Hz`

MDSR time stamp seconds.

Definition at line 342 of file L1bStructure.h.

### 6.1.2.3 `uint32_t AveragedWaveformsLRM::uj_Micsec_1Hz`

MDSR time stamp microseconds.

Definition at line 343 of file L1bStructure.h.

### 6.1.2.4 `int32_t AveragedWaveformsLRM::j_Lat_1Hz`

Latitude.

Units:  $10^{-7}$  degrees.

Definition at line 344 of file L1bStructure.h.

### 6.1.2.5 `int32_t AveragedWaveformsLRM::j_Lon_1Hz`

Longitude.

Units:  $10^{-7}$  degrees.

Definition at line 348 of file L1bStructure.h.

### 6.1.2.6 `int32_t AveragedWaveformsLRM::j_Alt_1Hz`

Altitude of COG above reference ellipsoid. Interpolated value.

Units: mm

Definition at line 352 of file L1bStructure.h.

### 6.1.2.7 `int64_t AveragedWaveformsLRM::k_TD_1Hz`

Window delay referenced to COM. Two-way, corrected for instrument delays.

Units: mm

Definition at line 358 of file L1bStructure.h.

### 6.1.2.8 `uint16_t* AveragedWaveformsLRM::pui_avg_wfm`

1Hz Averaged power waveform.



Definition at line 365 of file L1bStructure.h.

#### 6.1.2.9 int32\_t AveragedWaveformsLRM::j\_Linear\_wfm\_mult

Echo scale factor.

Definition at line 366 of file L1bStructure.h.

#### 6.1.2.10 int32\_t AveragedWaveformsLRM::j\_Power2\_wfm\_mult

Echo scale factor.

Definition at line 367 of file L1bStructure.h.

#### 6.1.2.11 uint16\_t AveragedWaveformsLRM::ui\_num\_avg\_echoes

Number of echoes averaged.

Definition at line 368 of file L1bStructure.h.

#### 6.1.2.12 uint16\_t AveragedWaveformsLRM::ui\_flags

Flags, see [t\\_Level1bAvgWfmLRMFlagBit](#)

Definition at line 369 of file L1bStructure.h.

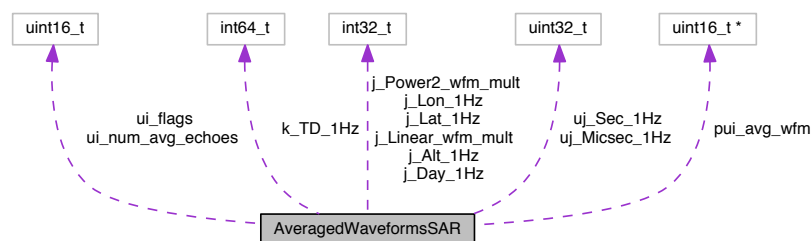
The documentation for this struct was generated from the following file:

- [L1bStructure.h](#)

## 6.2 AveragedWaveformsSAR Struct Reference

```
#include <L1bStructure.h>
```

Collaboration diagram for AveragedWaveformsSAR:



### Data Fields

- int32\_t [j\\_Day\\_1Hz](#)
- uint32\_t [uj\\_Sec\\_1Hz](#)
- uint32\_t [uj\\_Micsec\\_1Hz](#)
- int32\_t [j\\_Lat\\_1Hz](#)

- [int32\\_t j\\_Lon\\_1Hz](#)
- [int32\\_t j\\_Alt\\_1Hz](#)
- [int64\\_t k\\_TD\\_1Hz](#)
- [uint16\\_t \\* pui\\_avg\\_wfm](#)
- [int32\\_t j\\_Linear\\_wfm\\_mult](#)
- [int32\\_t j\\_Power2\\_wfm\\_mult](#)
- [uint16\\_t ui\\_num\\_avg\\_echoes](#)
- [uint16\\_t ui\\_flags](#)

### 6.2.1 Detailed Description

Level 1b record substructure (SAR mode specific) containing averaged waveform data.

Definition at line 374 of file L1bStructure.h.

### 6.2.2 Field Documentation

#### 6.2.2.1 `int32_t AveragedWaveformsSAR::j_Day_1Hz`

MDSR time stamp days.

Definition at line 376 of file L1bStructure.h.

#### 6.2.2.2 `uint32_t AveragedWaveformsSAR::uj_Sec_1Hz`

MDSR time stamp seconds.

Definition at line 377 of file L1bStructure.h.

#### 6.2.2.3 `uint32_t AveragedWaveformsSAR::uj_Micsec_1Hz`

MDSR time stamp microseconds.

Definition at line 378 of file L1bStructure.h.

#### 6.2.2.4 `int32_t AveragedWaveformsSAR::j_Lat_1Hz`

Latitude.

Units:  $10^{-7}$  degrees.

Definition at line 379 of file L1bStructure.h.

#### 6.2.2.5 `int32_t AveragedWaveformsSAR::j_Lon_1Hz`

Longitude.

Units:  $10^{-7}$  degrees.

Definition at line 383 of file L1bStructure.h.

#### 6.2.2.6 `int32_t AveragedWaveformsSAR::j_Alt_1Hz`

Altitude of COG above reference ellipsoid. Interpolated value.

Units: mm

Definition at line 387 of file L1bStructure.h.

#### 6.2.2.7 int64\_t AveragedWaveformsSAR::k\_TD\_1Hz

Window delay referenced to COM. Two-way, corrected for instrument delays.

Units: mm

Definition at line 393 of file L1bStructure.h.

#### 6.2.2.8 uint16\_t\* AveragedWaveformsSAR::pui\_avg\_wfm

1Hz Averaged power waveform.

Definition at line 400 of file L1bStructure.h.

#### 6.2.2.9 int32\_t AveragedWaveformsSAR::j\_Linear\_wfm\_mult

Echo scale factor.

Definition at line 401 of file L1bStructure.h.

#### 6.2.2.10 int32\_t AveragedWaveformsSAR::j\_Power2\_wfm\_mult

Echo scale factor.

Definition at line 402 of file L1bStructure.h.

#### 6.2.2.11 uint16\_t AveragedWaveformsSAR::ui\_num\_avg\_echoes

Number of echoes averaged.

Definition at line 403 of file L1bStructure.h.

#### 6.2.2.12 uint16\_t AveragedWaveformsSAR::ui\_flags

Flags, see [t\\_Level1bAvgWfmSARFlagBit](#)

Definition at line 404 of file L1bStructure.h.

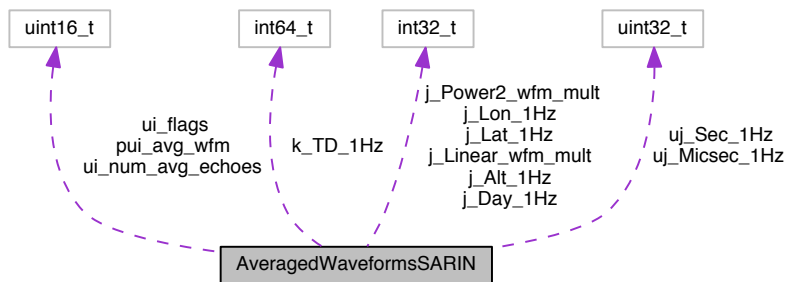
The documentation for this struct was generated from the following file:

- [L1bStructure.h](#)

## 6.3 AveragedWaveformsSARIN Struct Reference

```
#include <L1bStructure.h>
```

Collaboration diagram for AveragedWaveformsSARIN:



## Data Fields

- `int32_t j_Day_1Hz`
- `uint32_t uj_Sec_1Hz`
- `uint32_t uj_Micsec_1Hz`
- `int32_t j_Lat_1Hz`
- `int32_t j_Lon_1Hz`
- `int32_t j_Alt_1Hz`
- `int64_t k_TD_1Hz`
- `uint16_t * pui_avg_wfm`
- `int32_t j_Linear_wfm_mult`
- `int32_t j_Power2_wfm_mult`
- `uint16_t ui_num_avg_echoes`
- `uint16_t ui_flags`

### 6.3.1 Detailed Description

Level 1b record substructure (SARin mode specific) containing averaged waveform data.

Definition at line 409 of file `L1bStructure.h`.

### 6.3.2 Field Documentation

#### 6.3.2.1 `int32_t AveragedWaveformsSARIN::j_Day_1Hz`

MDSR time stamp days.

Definition at line 411 of file `L1bStructure.h`.

#### 6.3.2.2 `uint32_t AveragedWaveformsSARIN::uj_Sec_1Hz`

MDSR time stamp seconds.

Definition at line 412 of file `L1bStructure.h`.

#### 6.3.2.3 uint32\_t AveragedWaveformsSARIN::uj\_Micsec\_1Hz

MDSR time stamp microseconds.

Definition at line 413 of file L1bStructure.h.

#### 6.3.2.4 int32\_t AveragedWaveformsSARIN::j\_Lat\_1Hz

Latitude.

Units:  $10^{-7}$  degrees.

Definition at line 414 of file L1bStructure.h.

#### 6.3.2.5 int32\_t AveragedWaveformsSARIN::j\_Lon\_1Hz

Longitude.

Units:  $10^{-7}$  degrees.

Definition at line 418 of file L1bStructure.h.

#### 6.3.2.6 int32\_t AveragedWaveformsSARIN::j\_Alt\_1Hz

Altitude of COG above reference ellipsoid. Interpolated value.

Units: mm

Definition at line 422 of file L1bStructure.h.

#### 6.3.2.7 int64\_t AveragedWaveformsSARIN::k\_TD\_1Hz

Window delay referenced to COM. Two-way, corrected for instrument delays.

Units: mm

Definition at line 428 of file L1bStructure.h.

#### 6.3.2.8 uint16\_t\* AveragedWaveformsSARIN::pui\_avg\_wfm

1Hz Averaged power waveform.

Definition at line 435 of file L1bStructure.h.

#### 6.3.2.9 int32\_t AveragedWaveformsSARIN::j\_Linear\_wfm\_mult

Echo scale factor.

Definition at line 436 of file L1bStructure.h.

#### 6.3.2.10 int32\_t AveragedWaveformsSARIN::j\_Power2\_wfm\_mult

Echo scale factor.

Definition at line 437 of file L1bStructure.h.

### 6.3.2.11 uint16\_t AveragedWaveformsSARIN::ui\_num\_avg\_echoes

Number of echoes averaged.

Definition at line 438 of file L1bStructure.h.

### 6.3.2.12 uint16\_t AveragedWaveformsSARIN::ui\_flags

Flags, see [t\\_Level1bAvgWfmSARinFlagBit](#)

Definition at line 439 of file L1bStructure.h.

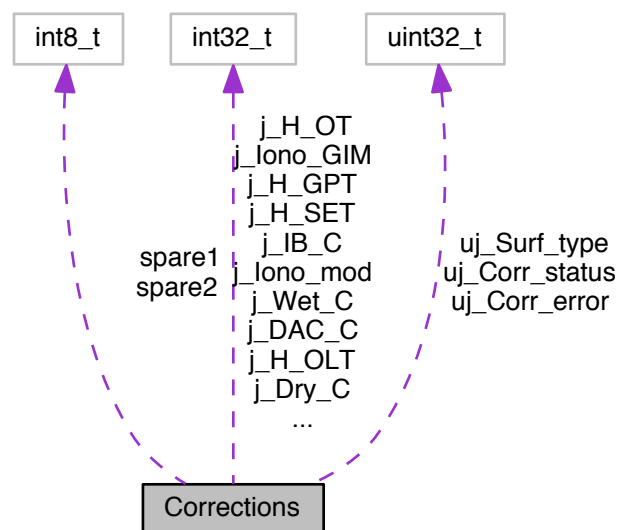
The documentation for this struct was generated from the following file:

- [L1bStructure.h](#)

## 6.4 Corrections Struct Reference

```
#include <L1bStructure.h>
```

Collaboration diagram for Corrections:



### Data Fields

- int32\_t [j\\_Dry\\_C](#)
- int32\_t [j\\_Wet\\_C](#)
- int32\_t [j\\_IB\\_C](#)
- int32\_t [j\\_DAC\\_C](#)
- int32\_t [j\\_lono\\_GIM](#)
- int32\_t [j\\_lono\\_mod](#)
- int32\_t [j\\_H\\_OT](#)

- [int32\\_t j\\_H\\_LPEOT](#)
- [int32\\_t j\\_H\\_OLT](#)
- [int32\\_t j\\_H\\_SET](#)
- [int32\\_t j\\_H\\_GPT](#)
- [uint32\\_t uj\\_Surf\\_type](#)
- [int8\\_t spare1](#) [4]
- [uint32\\_t uj\\_Corr\\_status](#)
- [uint32\\_t uj\\_Corr\\_error](#)
- [int8\\_t spare2](#) [4]

### 6.4.1 Detailed Description

Level 1b record substructure containing geophysical corrections.

Definition at line 277 of file L1bStructure.h.

### 6.4.2 Field Documentation

#### 6.4.2.1 `int32_t Corrections::j_Dry_C`

Dry tropospheric correction.

Units: mm

Definition at line 279 of file L1bStructure.h.

#### 6.4.2.2 `int32_t Corrections::j_Wet_C`

Wet tropospheric correction.

Units: mm

Definition at line 284 of file L1bStructure.h.

#### 6.4.2.3 `int32_t Corrections::j_IB_C`

Inverse barometric correction.

Units: mm

Definition at line 289 of file L1bStructure.h.

#### 6.4.2.4 `int32_t Corrections::j_DAC_C`

DAC correction.

Includes inverse barometric.

Units: mm

Definition at line 294 of file L1bStructure.h.

#### 6.4.2.5 `int32_t Corrections::j_Iono_GIM`

DORIS ionospheric correction.

Units: mm

Definition at line 300 of file L1bStructure.h.

#### 6.4.2.6 `int32_t Corrections::j_iono_mod`

Model ionospheric correction.

Units: mm

Definition at line 305 of file L1bStructure.h.

#### 6.4.2.7 `int32_t Corrections::j_H_OT`

Ocean equilibrium tide.

Units: mm

Definition at line 310 of file L1bStructure.h.

#### 6.4.2.8 `int32_t Corrections::j_H_LPEOT`

Long period equilibrium ocean tide.

Units: mm

Definition at line 314 of file L1bStructure.h.

#### 6.4.2.9 `int32_t Corrections::j_H_OLT`

Ocean loading tide.

Units: mm

Definition at line 318 of file L1bStructure.h.

#### 6.4.2.10 `int32_t Corrections::j_H_SET`

Solid Earth tide.

Units: mm

Definition at line 322 of file L1bStructure.h.

#### 6.4.2.11 `int32_t Corrections::j_H_GPT`

Geocentric Polar tide.

Units: mm

Definition at line 326 of file L1bStructure.h.

#### 6.4.2.12 `uint32_t Corrections::uj_Surf_type`

Surface type flag. See [t\\_SurfaceType](#)

Definition at line 330 of file L1bStructure.h.

#### 6.4.2.13 `int8_t Corrections::spare1[4]`

Spares.

Definition at line 331 of file L1bStructure.h.



#### 6.4.2.14 uint32\_t Corrections::uj\_Corr\_status

Correction status flag. See [t\\_CorrectionStatusFlagBit](#)

Definition at line 332 of file L1bStructure.h.

#### 6.4.2.15 uint32\_t Corrections::uj\_Corr\_error

Correction error flag. See [t\\_CorrectionErrorFlagBit](#)

Definition at line 333 of file L1bStructure.h.

#### 6.4.2.16 int8\_t Corrections::spare2[4]

Spares.

Definition at line 334 of file L1bStructure.h.

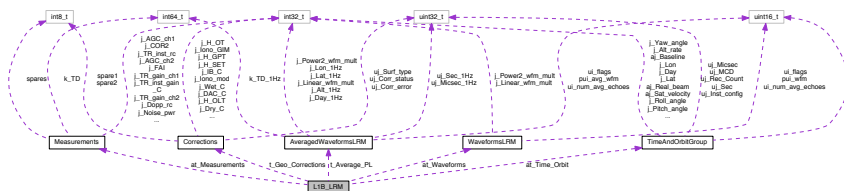
The documentation for this struct was generated from the following file:

- L1bStructure.h

## 6.5 L1B\_LRM Struct Reference

```
#include <L1bStructure.h>
```

Collaboration diagram for L1B\_LRM:



## Data Fields

- TimeAndOrbitGroup at Time\_Orbit [J\_NUM\_HRB]
- Measurements at Measurements [J\_NUM\_HRB]
- Corrections t\_Geo\_Corrections
- AveragedWaveformsLRM t\_Average\_PL
- WaveformsLRM at Waveforms [J\_NUM\_HRB]

### 6.5.1 Detailed Description

Level 1b record structure for LRM mode.

**Examples:**

examples/cs l1b io example.c, and examples/example bit manipulation.c.

Definition at line 485 of file L1bStructure.h.

## 6.5.2 Field Documentation

### 6.5.2.1 TimeAndOrbitGroup L1B\_LRM::at\_Time\_Orbit[J\_NUM\_HRB]

Time and orbit parameters substructure.

Definition at line 487 of file L1bStructure.h.

### 6.5.2.2 Measurements L1B\_LRM::at\_Measurements[J\_NUM\_HRB]

[Measurements](#) substructure.

Definition at line 490 of file L1bStructure.h.

### 6.5.2.3 Corrections L1B\_LRM::t\_Geo\_Corrections

Geo-corrections substructure.

Definition at line 492 of file L1bStructure.h.

### 6.5.2.4 AveragedWaveformsLRM L1B\_LRM::t\_Average\_PL

Averaged waveforms substructure.

Definition at line 493 of file L1bStructure.h.

### 6.5.2.5 WaveformsLRM L1B\_LRM::at\_Waveforms[J\_NUM\_HRB]

Waveforms substructure.

Definition at line 495 of file L1bStructure.h.

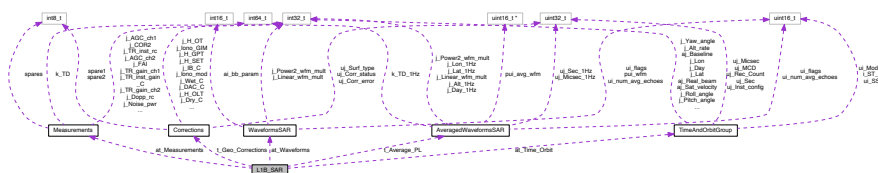
The documentation for this struct was generated from the following file:

- [L1bStructure.h](#)

## 6.6 L1B\_SAR Struct Reference

```
#include <L1bStructure.h>
```

Collaboration diagram for L1B\_SAR:



## Data Fields

- [TimeAndOrbitGroup](#) [at\\_Time\\_Orbit](#) [J\_NUM\_HRB]
- [Measurements](#) [at\\_Measurements](#) [J\_NUM\_HRB]
- [Corrections](#) [t\\_Geo\\_Corrections](#)

- [AveragedWaveformsSAR t\\_Average\\_PL](#)
- [WaveformsSAR at\\_Waveforms \[J\\_NUM\\_HRB\]](#)

### 6.6.1 Detailed Description

Level 1b record structure for SAR mode.

Examples:

[examples/cs\\_l1b\\_io\\_example.c](#).

Definition at line 499 of file L1bStructure.h.

### 6.6.2 Field Documentation

#### 6.6.2.1 TimeAndOrbitGroup L1B\_SAR::at\_Time\_Orbit[J\_NUM\_HRB]

Time and orbit parameters substructure.

Definition at line 501 of file L1bStructure.h.

#### 6.6.2.2 Measurements L1B\_SAR::at\_Measurements[J\_NUM\_HRB]

[Measurements](#) substructure.

Definition at line 504 of file L1bStructure.h.

#### 6.6.2.3 Corrections L1B\_SAR::t\_Geo\_Corrections

Geo-corrections substructure.

Definition at line 506 of file L1bStructure.h.

#### 6.6.2.4 AveragedWaveformsSAR L1B\_SAR::t\_Average\_PL

Averaged waveforms substructure.

Definition at line 507 of file L1bStructure.h.

#### 6.6.2.5 WaveformsSAR L1B\_SAR::at\_Waveforms[J\_NUM\_HRB]

Waveforms substructure.

Definition at line 509 of file L1bStructure.h.

The documentation for this struct was generated from the following file:

- [L1bStructure.h](#)

## 6.7 L1B\_SARIN Struct Reference

```
#include <L1bStructure.h>
```

[illegible]

- TimeAndOrbitGroup at Time\_Orbit [J\_NUM\_HRB]
- Measurements at Measurements [J\_NUM\_HRB]
- Corrections t\_Geo\_Corrections
- AveragedWaveformsSARIN t\_Average\_PL
- WaveformsSARIN at Waveforms [J\_NUM\_HRB]

Level 1b record structure for SARin mode.

[examples/cs\\_l1b\\_io\\_example.c.](#)

Definition at line 513 of file L1bStructure.h.

#### 6.7.2.1 TimeAndOrbitGroup L1B\_SARIN::at\_Time\_Orbit[J\_NUM\_HRB]

Definition at line 515 of file L1bStructure.h.

## Measurements substructure.

Definition at line 518 of file L1bStructure.h.

Geo-corrections substructure.

Definition at line 521 of file L1bStructure.h.

Averaged waveforms substructure.

Definition at line 522 of file L1bStructure.h.

## 6.7.2.5 WaveformsSARIN L1B\_SARIN::at\_Waveforms[J\_NUM\_HRB]

Waveforms substructure.

Definition at line 524 of file L1bStructure.h.

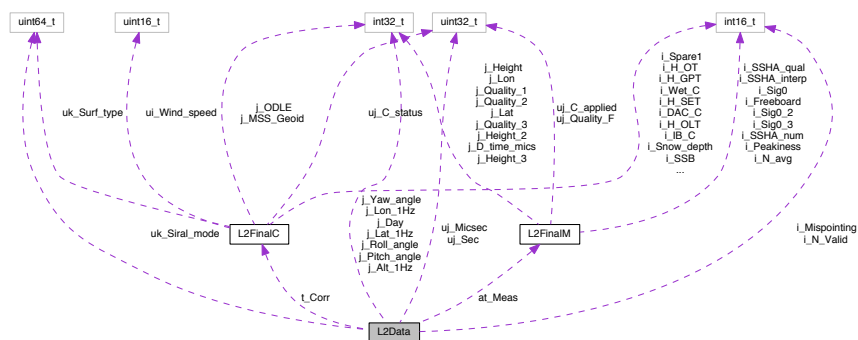
The documentation for this struct was generated from the following file:

- [L1bStructure.h](#)

## 6.8 L2Data Struct Reference

```
#include <L2Structure.h>
```

Collaboration diagram for L2Data:



## Data Fields

- [int32\\_t j\\_Day](#)
- [uint32\\_t uj\\_Sec](#)
- [uint32\\_t uj\\_Micsec](#)
- [uint64\\_t uk\\_Siral\\_mode](#)
- [int32\\_t j\\_Lat\\_1Hz](#)
- [int32\\_t j\\_Lon\\_1Hz](#)
- [int32\\_t j\\_Alt\\_1Hz](#)
- [int16\\_t i\\_Mispointing](#)
- [int32\\_t j\\_Roll\\_angle](#)
- [int32\\_t j\\_Pitch\\_angle](#)
- [int32\\_t j\\_Yaw\\_angle](#)
- [int16\\_t i\\_N\\_Valid](#)
- [L2FinalC t\\_Corr](#)
- [L2FinalM at\\_Meas](#) [20]

## 6.8.1 Detailed Description

The Cryosat Level 2 data structure.

Examples:

[examples/cs\\_l2\\_io\\_example.c](#).

Definition at line 300 of file L2Structure.h.

## 6.8.2 Field Documentation

### 6.8.2.1 `int32_t L2Data::j_Day`

Time: day part.

Units: DAY

Definition at line 302 of file L2Structure.h.

### 6.8.2.2 `uint32_t L2Data::uj_Sec`

Time: second part.

Units: sec

Definition at line 307 of file L2Structure.h.

### 6.8.2.3 `uint32_t L2Data::uj_Micsec`

Time: microsecond part.

Units: mu-sec

Definition at line 312 of file L2Structure.h.

### 6.8.2.4 `uint64_t L2Data::uk_Siral_mode`

SIRAL mode.

Packed at 2 bits for each of 20 records.

Units: -

Definition at line 316 of file L2Structure.h.

### 6.8.2.5 `int32_t L2Data::j_Lat_1Hz`

Latitude of measurement.

Units: 0.1 micro-degree

Definition at line 323 of file L2Structure.h.

### 6.8.2.6 `int32_t L2Data::j_Lon_1Hz`

Longitude of measurement.

Units: 0.1 micro-degree

Definition at line 327 of file L2Structure.h.

### 6.8.2.7 `int32_t L2Data::j_Alt_1Hz`

Altitude of COG above reference ellipsoid (interpolated value).

Units: mm

Definition at line 331 of file L2Structure.h.

**6.8.2.8 int16\_t L2Data::i\_Mispointing**

Mispointing angle. Not used in Baseline C.

Units: milli-degree

Definition at line 336 of file L2Structure.h.

**6.8.2.9 int32\_t L2Data::j\_Roll\_angle**

Spacecraft roll angle.

Baseline C only.

Derived from star trackers.

Units:  $10^{-7}$  degrees

Definition at line 341 of file L2Structure.h.

**6.8.2.10 int32\_t L2Data::j\_Pitch\_angle**

Spacecraft roll angle.

Baseline C only.

Derived from star trackers.

Units:  $10^{-7}$  degrees

Definition at line 349 of file L2Structure.h.

**6.8.2.11 int32\_t L2Data::j\_Yaw\_angle**

Spacecraft roll angle.

Baseline C only

Derived from star trackers.

Units:  $10^{-7}$  degrees

Definition at line 357 of file L2Structure.h.

**6.8.2.12 int16\_t L2Data::i\_N\_Valid**

Number of valid records.

The number of records in the block of twenty that actually contain data. The last few records of the last block of a dataset may be blank blocks inserted to bring the the file up to a multiple of twenty.

Units: -

Definition at line 365 of file L2Structure.h.

**6.8.2.13 L2FinalC L2Data::t\_Corr**

Correction substructure.

Definition at line 377 of file L2Structure.h.

### 6.8.2.14 L2FinalM L2Data::at\_Meas[20]

Measurement substructure array.

Definition at line 379 of file L2Structure.h.

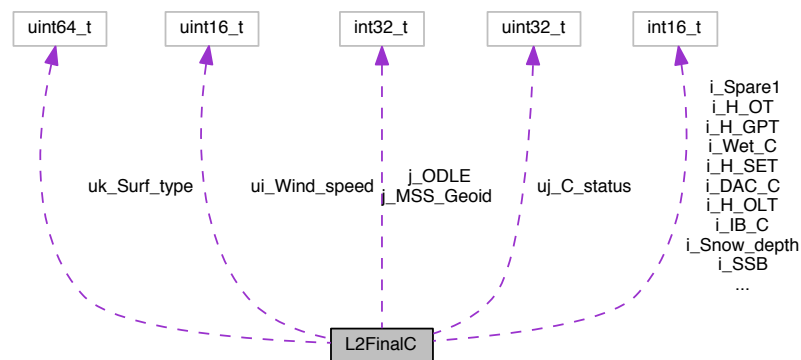
The documentation for this struct was generated from the following file:

- [L2Structure.h](#)

## 6.9 L2FinalC Struct Reference

```
#include <L2Structure.h>
```

Collaboration diagram for L2FinalC:



### Data Fields

- `int16_t i_Dry_C`
- `int16_t i_Wet_C`
- `int16_t i_IB_C`
- `int16_t i_DAC_C`
- `int16_t i_Iono_C`
- `int16_t i_SSB`
- `int16_t i_H_OT`
- `int16_t i_H_LPEOT`
- `int16_t i_H_OLT`
- `int16_t i_H_SET`
- `int16_t i_H_GPT`
- `int16_t i_Spare1`
- `uint64_t uk_Surf_type`
- `int32_t j_MSS_Geoid`
- `int32_t j_ODLE`
- `int16_t i_Ice_conc`
- `int16_t i_Snow_depth`
- `int16_t i_Snow_density`
- `uint32_t uj_C_status`
- `int16_t i_SWH`
- `uint16_t ui_Wind_speed`



### 6.9.1 Detailed Description

The corrections substructure of the Cryosat Level 2 structure.

Definition at line 95 of file L2Structure.h.

### 6.9.2 Field Documentation

#### 6.9.2.1 `int16_t L2FinalC::i_Dry_C`

Dry Tropospheric Correction.

Units: mm

Definition at line 97 of file L2Structure.h.

#### 6.9.2.2 `int16_t L2FinalC::i_Wet_C`

Wet Tropospheric Correction.

Units: mm

Definition at line 101 of file L2Structure.h.

#### 6.9.2.3 `int16_t L2FinalC::i_IB_C`

Inverse Barometric Correction.

Units: mm

Definition at line 105 of file L2Structure.h.

#### 6.9.2.4 `int16_t L2FinalC::i_DAC_C`

Dynamic Atmosphere Correction.

Units: mm

Definition at line 109 of file L2Structure.h.

#### 6.9.2.5 `int16_t L2FinalC::i_Iono_C`

Ionospheric Correction.

Units: mm

Definition at line 113 of file L2Structure.h.

#### 6.9.2.6 `int16_t L2FinalC::i_SSB`

Sea State Bias Correction.

Units: mm

Definition at line 117 of file L2Structure.h.

#### 6.9.2.7 `int16_t L2FinalC::i_H_OT`

Ocean tide.

Units: mm

Definition at line 121 of file L2Structure.h.

#### 6.9.2.8 int16\_t L2FinalC::i\_H\_LPEOT

Long period equilibrium ocean tide.

Units: mm

Definition at line 125 of file L2Structure.h.

#### 6.9.2.9 int16\_t L2FinalC::i\_H\_OLT

Ocean Loading Tide.

Units: mm

Definition at line 129 of file L2Structure.h.

#### 6.9.2.10 int16\_t L2FinalC::i\_H\_SET

Solid Earth Tide.

Units: mm

Definition at line 133 of file L2Structure.h.

#### 6.9.2.11 int16\_t L2FinalC::i\_H\_GPT

Geocentric Polar Tide.

Units: mm

Definition at line 137 of file L2Structure.h.

#### 6.9.2.12 int16\_t L2FinalC::i\_Spare1

spare

Units:

Definition at line 141 of file L2Structure.h.

#### 6.9.2.13 uint64\_t L2FinalC::uk\_Surf\_type

Surface Type.

Packed in groups of three bits for each of the 20 records.

Units: -

Definition at line 145 of file L2Structure.h.

#### 6.9.2.14 int32\_t L2FinalC::j\_MSS\_Geoid

Mean Sea Surface or Geoid.

Units: mm

Definition at line 152 of file L2Structure.h.

**6.9.2.15 int32\_t L2FinalC::j\_ODLE**

ODLE

Units: mm

Definition at line 156 of file L2Structure.h.

**6.9.2.16 int16\_t L2FinalC::i\_ice\_conc**

Ice Conc.

Units: %/100

Definition at line 160 of file L2Structure.h.

**6.9.2.17 int16\_t L2FinalC::i\_Snow\_depth**

Snow Depth. Units: mm

Definition at line 164 of file L2Structure.h.

**6.9.2.18 int16\_t L2FinalC::i\_Snow\_density**

Snow Density.

Units: kg/m<sup>3</sup>

Definition at line 167 of file L2Structure.h.

**6.9.2.19 uint32\_t L2FinalC::uj\_C\_status**

[Corrections](#) Status Flag, see [t\\_Level2CorrectionsStatusFlagBit](#)

Units: -

Definition at line 171 of file L2Structure.h.

**6.9.2.20 int16\_t L2FinalC::i\_SWH**

SWH

Units: mm

Definition at line 175 of file L2Structure.h.

**6.9.2.21 uint16\_t L2FinalC::ui\_Wind\_speed**

Wind Speed.

Units: mm/s

Definition at line 179 of file L2Structure.h.

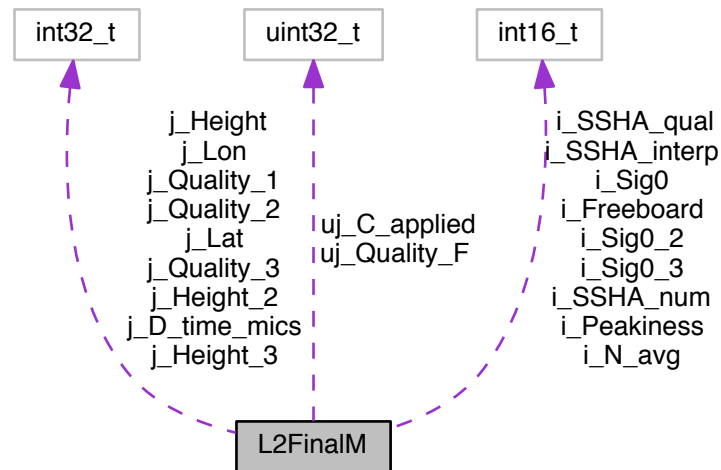
The documentation for this struct was generated from the following file:

- [L2Structure.h](#)

## 6.10 L2FinalM Struct Reference

```
#include <L2Structure.h>
```

Collaboration diagram for L2FinalM:



### Data Fields

- [int32\\_t j\\_D\\_time\\_mics](#)
- [int32\\_t j\\_Lat](#)
- [int32\\_t j\\_Lon](#)
- [int32\\_t j\\_Height](#)
- [int32\\_t j\\_Height\\_2](#)
- [int32\\_t j\\_Height\\_3](#)
- [int16\\_t i\\_SSHA\\_interp](#)
- [int16\\_t i\\_SSHA\\_num](#)
- [int16\\_t i\\_SSHA\\_qual](#)
- [int16\\_t i\\_Sig0](#)
- [int16\\_t i\\_Sig0\\_2](#)
- [int16\\_t i\\_Sig0\\_3](#)
- [int16\\_t i\\_Peakiness](#)
- [int16\\_t i\\_Freeboard](#)
- [int16\\_t i\\_N\\_avg](#)
- [uint32\\_t uj\\_Quality\\_F](#)
- [uint32\\_t uj\\_C\\_applied](#)
- [int32\\_t j\\_Quality\\_1](#)
- [int32\\_t j\\_Quality\\_2](#)
- [int32\\_t j\\_Quality\\_3](#)

### 6.10.1 Detailed Description

The measurements substructure of the Cryosat Level 2 structure.

Definition at line 189 of file L2Structure.h.

## 6.10.2 Field Documentation

### 6.10.2.1 `int32_t L2FinalM::j_D_time_mics`

Delta time.

The delta between the timestamp for this record and the timestamp given at 1HZ.

Units: microseconds

Definition at line 191 of file L2Structure.h.

### 6.10.2.2 `int32_t L2FinalM::j_Lat`

Latitude.

Units: 0.1 microdegrees

Definition at line 199 of file L2Structure.h.

### 6.10.2.3 `int32_t L2FinalM::j_Lon`

Longitude.

Units: 0.1 microdegrees

Definition at line 203 of file L2Structure.h.

### 6.10.2.4 `int32_t L2FinalM::j_Height`

Measured elevation above ellipsoid from retracker 1.

Units: mm

Definition at line 207 of file L2Structure.h.

### 6.10.2.5 `int32_t L2FinalM::j_Height_2`

Measured elevation above ellipsoid from retracker 2.

Baseline C only.

Units: mm

Definition at line 211 of file L2Structure.h.

### 6.10.2.6 `int32_t L2FinalM::j_Height_3`

Measured elevation above ellipsoid from retracker 3.

Baseline C only.

Units: mm

Definition at line 217 of file L2Structure.h.

### 6.10.2.7 `int16_t L2FinalM::i_SSHA_interp`

Interpolated Sea Surface Height Anomaly.

Units: mm

Definition at line 223 of file L2Structure.h.

**6.10.2.8 int16\_t L2FinalM::i\_SSHA\_num**

Interpolated Sea Surface Height Anomaly.

Units: mm

Definition at line 227 of file L2Structure.h.

**6.10.2.9 int16\_t L2FinalM::i\_SSHA\_qual**

Interpolation quality estimate RSS.

Units: mm

Definition at line 231 of file L2Structure.h.

**6.10.2.10 int16\_t L2FinalM::i\_Sig0**

Sigma Zero Backscatter for retracker 1.

Units:  $10^{-2}$  dB

Definition at line 235 of file L2Structure.h.

**6.10.2.11 int16\_t L2FinalM::i\_Sig0\_2**

Sigma Zero Backscatter for retracker 2.

Baseline C only.

Units:  $10^{-2}$  dB

Definition at line 239 of file L2Structure.h.

**6.10.2.12 int16\_t L2FinalM::i\_Sig0\_3**

Sigma Zero Backscatter for retracker 3.

Baseline C only.

Units:  $10^{-2}$  dB

Definition at line 245 of file L2Structure.h.

**6.10.2.13 int16\_t L2FinalM::i\_Peakiness**

Peakiness.

Units: 1/100

Definition at line 251 of file L2Structure.h.

**6.10.2.14 int16\_t L2FinalM::i\_Freeboard**

Freeboard.

-9999 is a default value indicating that the computation has not been performed.

Units: mm

Definition at line 255 of file L2Structure.h.

#### 6.10.2.15 int16\_t L2FinalM::i\_N\_avg

Number of averaged echoes or beams.

Units: -

Definition at line 262 of file L2Structure.h.

#### 6.10.2.16 uint32\_t L2FinalM::uj\_Quality\_F

Quality flags. See [t\\_Level2QualityStatusFlagBit](#)

Units: -

Definition at line 267 of file L2Structure.h.

#### 6.10.2.17 uint32\_t L2FinalM::uj\_C\_applied

[Corrections](#) Application Flag. See [t\\_Level2CorrectionsAppliedFlagBit](#)

Baseline C only.

Units: -

Definition at line 271 of file L2Structure.h.

#### 6.10.2.18 int32\_t L2FinalM::j\_Quality\_1

Quality metric for retracker 1.

Baseline C only.

Units: -

Definition at line 277 of file L2Structure.h.

#### 6.10.2.19 int32\_t L2FinalM::j\_Quality\_2

Quality metric for retracker 2.

Baseline C only.

Units: -

Definition at line 283 of file L2Structure.h.

#### 6.10.2.20 int32\_t L2FinalM::j\_Quality\_3

Quality metric for retracker 3.

Baseline C only.

Units: -

Definition at line 289 of file L2Structure.h.

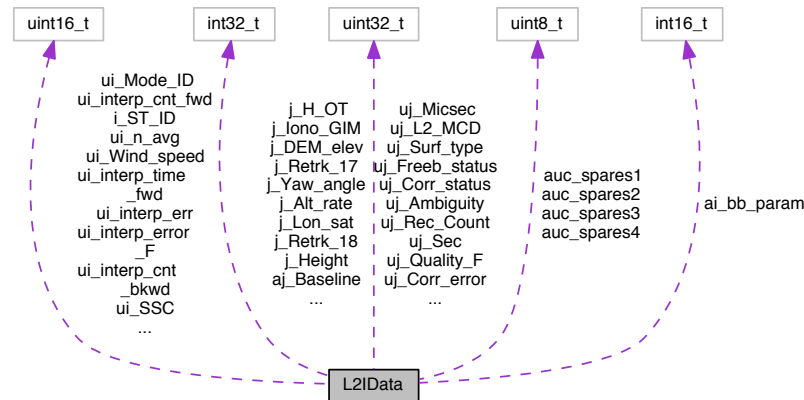
The documentation for this struct was generated from the following file:

- [L2Structure.h](#)

## 6.11 L2IData Struct Reference

```
#include <L2IStructure.h>
```

Collaboration diagram for L2IData:



### Data Fields

- `int32_t j_Day`
- `uint32_t uj_Sec`
- `uint32_t uj_Micsec`
- `int32_t j_USO_Corr`
- `uint16_t ui_Mode_ID`
- `uint16_t ui_SSC`
- `uint32_t uj_Inst_config`
- `uint32_t uj_Rec_Count`
- `int32_t j_Lat`
- `int32_t j_Lon`
- `int32_t j_Alt`
- `int32_t j_Alt_rate`
- `int32_t aj_Sat_velocity [3]`
- `int32_t aj_Real_beam [3]`
- `int32_t aj_Baseline [3]`
- `uint16_t i_ST_ID`
- `int32_t j_Roll_angle`
- `int32_t j_Pitch_angle`
- `int32_t j_Yaw_angle`
- `uint32_t uj_L2_MCD`
- `int32_t j_Height`
- `int32_t j_Height_2`
- `int32_t j_Height_3`
- `int32_t j_Sig0`
- `int32_t j_Sig0_2`
- `int32_t j_Sig0_3`
- `int32_t j_SWH`
- `int32_t j_Peakiness`
- `int32_t j_Retrk_range_C`



- [int32\\_t j\\_Retrk\\_range\\_C\\_2](#)
- [int32\\_t j\\_Retrk\\_range\\_C\\_3](#)
- [int32\\_t j\\_Retrk\\_sig0\\_C](#)
- [int32\\_t j\\_Retrk\\_sig0\\_C\\_2](#)
- [int32\\_t j\\_Retrk\\_sig0\\_C\\_3](#)
- [int32\\_t j\\_Retrk\\_Quality\\_1](#)
- [int32\\_t j\\_Retrk\\_Quality\\_2](#)
- [int32\\_t j\\_Retrk\\_Quality\\_3](#)
- [int32\\_t j\\_Retrk\\_3](#)
- [int32\\_t j\\_Retrk\\_4](#)
- [int32\\_t j\\_Retrk\\_5](#)
- [int32\\_t j\\_Retrk\\_6](#)
- [int32\\_t j\\_Retrk\\_7](#)
- [int32\\_t j\\_Retrk\\_8](#)
- [int32\\_t j\\_Retrk\\_9](#)
- [int32\\_t j\\_Retrk\\_10](#)
- [int32\\_t j\\_Retrk\\_11](#)
- [int32\\_t j\\_Retrk\\_12](#)
- [int32\\_t j\\_Retrk\\_13](#)
- [int32\\_t j\\_Retrk\\_14](#)
- [int32\\_t j\\_Retrk\\_15](#)
- [int32\\_t j\\_Retrk\\_16](#)
- [int32\\_t j\\_Retrk\\_17](#)
- [int32\\_t j\\_Retrk\\_18](#)
- [int32\\_t j\\_Retrk\\_19](#)
- [int32\\_t j\\_Retrk\\_20](#)
- [int32\\_t j\\_Retrk\\_21](#)
- [int32\\_t j\\_Retrk\\_22](#)
- [int32\\_t j\\_Retrk\\_23](#)
- [int32\\_t j\\_echo\\_shape](#)
- [int16\\_t ai\\_bb\\_param \[50\]](#)
- [int32\\_t j\\_XTrack\\_angle](#)
- [int32\\_t j\\_XTrack\\_angle\\_C](#)
- [int32\\_t j\\_Coherence](#)
- [int32\\_t j\\_Ocean\\_ht](#)
- [int32\\_t j\\_Freeboard](#)
- [int32\\_t j\\_SHA](#)
- [int32\\_t j\\_SSHA\\_interp](#)
- [uint16\\_t ui\\_interp\\_err](#)
- [uint16\\_t ui\\_interp\\_cnt\\_fwd](#)
- [uint16\\_t ui\\_interp\\_cnt\\_bkwd](#)
- [uint16\\_t ui\\_interp\\_time\\_fwd](#)
- [uint16\\_t ui\\_interp\\_time\\_bkwd](#)
- [uint16\\_t ui\\_interp\\_error\\_F](#)
- [uint32\\_t uj\\_Meas\\_Mode](#)
- [uint32\\_t uj\\_Quality\\_F](#)
- [uint32\\_t uj\\_Retracker\\_F](#)
- [uint32\\_t uj\\_Ht\\_status](#)
- [uint32\\_t uj\\_Freeb\\_status](#)
- [uint16\\_t ui\\_n\\_avg](#)
- [uint16\\_t ui\\_Wind\\_speed](#)
- [uint8\\_t auc\\_spares1 \[12\]](#)
- [int32\\_t j\\_Ice\\_conc](#)
- [int32\\_t j\\_Snow\\_depth](#)
- [int32\\_t j\\_Snow\\_density](#)

- [int32\\_t j\\_Discriminator](#)
- [int32\\_t j\\_SARin\\_disc\\_1](#)
- [int32\\_t j\\_SARin\\_disc\\_2](#)
- [int32\\_t j\\_SARin\\_disc\\_3](#)
- [int32\\_t j\\_SARin\\_disc\\_4](#)
- [int32\\_t j\\_SARin\\_disc\\_5](#)
- [int32\\_t j\\_SARin\\_disc\\_6](#)
- [int32\\_t j\\_SARin\\_disc\\_7](#)
- [int32\\_t j\\_SARin\\_disc\\_8](#)
- [int32\\_t j\\_SARin\\_disc\\_9](#)
- [int32\\_t j\\_SARin\\_disc\\_10](#)
- [uint32\\_t uj\\_Discrim\\_F](#)
- [int32\\_t j\\_Attitude](#)
- [int32\\_t j\\_Azimuth](#)
- [int32\\_t j\\_Slope\\_Doppler\\_C](#)
- [int32\\_t j\\_Lat\\_sat](#)
- [int32\\_t j\\_Lon\\_sat](#)
- [uint32\\_t uj\\_Ambiguity](#)
- [int32\\_t j\\_MSS\\_mod](#)
- [int32\\_t j\\_Geoid\\_mod](#)
- [int32\\_t j\\_ODLE\\_mod](#)
- [int32\\_t j\\_DEM\\_elev](#)
- [uint32\\_t uj\\_DEM\\_id](#)
- [uint8\\_t auc\\_spares2 \[16\]](#)
- [int32\\_t j\\_Dry\\_C](#)
- [int32\\_t j\\_Wet\\_C](#)
- [int32\\_t j\\_IB\\_C](#)
- [int32\\_t j\\_DAC\\_C](#)
- [int32\\_t j\\_lono\\_GIM](#)
- [int32\\_t j\\_lono\\_mod](#)
- [int32\\_t j\\_H\\_OT](#)
- [int32\\_t j\\_H\\_LPEOT](#)
- [int32\\_t j\\_H\\_OLT](#)
- [int32\\_t j\\_H\\_SET](#)
- [int32\\_t j\\_H\\_GPT](#)
- [uint32\\_t uj\\_Surf\\_type](#)
- [uint32\\_t uj\\_Corr\\_status](#)
- [uint32\\_t uj\\_Corr\\_error](#)
- [int32\\_t j\\_SSB](#)
- [uint8\\_t auc\\_spares3 \[8\]](#)
- [int32\\_t j\\_Dopp\\_rc](#)
- [int32\\_t j\\_TR\\_inst\\_rc](#)
- [int32\\_t j\\_R\\_inst\\_rc](#)
- [int32\\_t j\\_TR\\_inst\\_gain\\_C](#)
- [int32\\_t j\\_R\\_inst\\_gain\\_C](#)
- [int32\\_t j\\_Int\\_phase\\_C](#)
- [int32\\_t j\\_Ext\\_phase\\_C](#)
- [int32\\_t j\\_Noise\\_pwr](#)
- [int32\\_t j\\_Phase\\_slope\\_C](#)
- [uint8\\_t auc\\_spares4 \[8\]](#)

### 6.11.1 Detailed Description

The interim L2 structure

Examples:

[examples/cs\\_l2i\\_io\\_example.c](#).

Definition at line 90 of file L2IStructure.h.

### 6.11.2 Field Documentation

#### 6.11.2.1 int32\_t L2IData::j\_Day

Time: day part.

Units: DAY

Definition at line 93 of file L2IStructure.h.

#### 6.11.2.2 uint32\_t L2IData::uj\_Sec

Time: second part.

Units: msec

Definition at line 98 of file L2IStructure.h.

#### 6.11.2.3 uint32\_t L2IData::uj\_Micsec

Time: microsecond part.

Units: mu-sec

Definition at line 103 of file L2IStructure.h.

#### 6.11.2.4 int32\_t L2IData::j\_USO\_Corr

USO Correction factor.

Units: -

Definition at line 108 of file L2IStructure.h.

#### 6.11.2.5 uint16\_t L2IData::ui\_Mode\_ID

SIRAL Mode ID. See [t\\_ModelID](#)

Units: -

Definition at line 113 of file L2IStructure.h.

#### 6.11.2.6 uint16\_t L2IData::ui\_SSC

Source Sequence Counter. Units: -

Definition at line 118 of file L2IStructure.h.

#### 6.11.2.7 uint32\_t L2IData::uj\_Inst\_config

Instrument configuration. See [t\\_InstConf](#)

Includes loop status.

Units: -

Definition at line 122 of file L2IStructure.h.

#### 6.11.2.8 uint32\_t L2IData::uj\_Rec\_Count

Surface Sample Counter.

Record counter.

Units: -

Definition at line 129 of file L2IStructure.h.

#### 6.11.2.9 int32\_t L2IData::j\_Lat

Latitude of measurement.

Units: 0.1 micro-degree

Definition at line 136 of file L2IStructure.h.

#### 6.11.2.10 int32\_t L2IData::j\_Lon

Longitude of measurement.

Units: 0.1 micro-degree

Definition at line 141 of file L2IStructure.h.

#### 6.11.2.11 int32\_t L2IData::j\_Alt

Altitude of COG above reference ellipsoid (interpolated value)

Units: mm

Definition at line 146 of file L2IStructure.h.

#### 6.11.2.12 int32\_t L2IData::j\_Alt\_rate

Instantaneous altitude rate derived from orbit.

Units: mm/s

Definition at line 152 of file L2IStructure.h.

#### 6.11.2.13 int32\_t L2IData::aj\_Sat\_velocity[3]

Satellite velocity vector. In CRF

Units: mm/s

Definition at line 158 of file L2IStructure.h.

**6.11.2.14 int32\_t L2IData::aj\_Real\_beam[3]**

Real beam direction vector.

In CRF

Units: micro-metre

Definition at line 164 of file L2IStructure.h.

**6.11.2.15 int32\_t L2IData::aj\_Baseline[3]**

Interferometer baseline vector.

In CRF

Units: micro-metre

Definition at line 171 of file L2IStructure.h.

**6.11.2.16 uint16\_t L2IData::i\_ST\_ID**

Star Tracker ID. See [t\\_STID](#)

Baseline BC and later only

Units: enum

Definition at line 177 of file L2IStructure.h.

**6.11.2.17 int32\_t L2IData::j\_Roll\_angle**

Spacecraft roll angle.

Baseline C only.

Derived from star trackers.

Units:  $10^{-7}$  degrees

Definition at line 183 of file L2IStructure.h.

**6.11.2.18 int32\_t L2IData::j\_Pitch\_angle**

Spacecraft roll angle.

Baseline C only.

Derived from star trackers.

Units:  $10^{-7}$  degrees

Definition at line 191 of file L2IStructure.h.

**6.11.2.19 int32\_t L2IData::j\_Yaw\_angle**

Spacecraft roll angle.

Baseline C only.

Derived from star trackers.

Units:  $10^{-7}$  degrees

Definition at line 199 of file L2IStructure.h.

**6.11.2.20    `uint32_t L2IData::uj_L2_MCD`**

Level 2 Confidence data (flag). See [t\\_MCDFlagBit](#)

Units: -

Definition at line 207 of file L2IStructure.h.

**6.11.2.21    `int32_t L2IData::j_Height`**

Height with respect to the ellipsoid.

Units: mm

Definition at line 213 of file L2IStructure.h.

**6.11.2.22    `int32_t L2IData::j_Height_2`**

Height with respect to the ellipsoid from retracker 2.

Baseline BC and above only.

Units: mm

Definition at line 219 of file L2IStructure.h.

**6.11.2.23    `int32_t L2IData::j_Height_3`**

Height with respect to the ellipsoid from retracker 3.

Baseline BC and above only.

Units: mm

Definition at line 227 of file L2IStructure.h.

**6.11.2.24    `int32_t L2IData::j_Sig0`**

Sigma 0. Units: dB/100

Definition at line 235 of file L2IStructure.h.

**6.11.2.25    `int32_t L2IData::j_Sig0_2`**

Sigma 0 from retracker 2.

Baseline BC and above only.

Units: dB/100

Definition at line 239 of file L2IStructure.h.

**6.11.2.26    `int32_t L2IData::j_Sig0_3`**

Sigma 0 from retracker 3.

Baseline BC and above only.

Units: dB/100

Definition at line 246 of file L2IStructure.h.

**6.11.2.27 int32\_t L2IData::j\_SWH**

Significant Wave Height.

Units: mm

Definition at line 253 of file L2IStructure.h.

**6.11.2.28 int32\_t L2IData::j\_Peakiness**

Peakiness.

Units: 1/1000

Definition at line 259 of file L2IStructure.h.

**6.11.2.29 int32\_t L2IData::j\_Retrk\_range\_C**

Retracked range correction.

Units: mm

Definition at line 264 of file L2IStructure.h.

**6.11.2.30 int32\_t L2IData::j\_Retrk\_range\_C\_2**

Retracked range correction for retracker 2.

Baseline BC and above.

Units: mm

Definition at line 269 of file L2IStructure.h.

**6.11.2.31 int32\_t L2IData::j\_Retrk\_range\_C\_3**

Retracked range correction for retracker 3.

Baseline BC and above.

Units: mm

Definition at line 276 of file L2IStructure.h.

**6.11.2.32 int32\_t L2IData::j\_Retrk\_sig0\_C**

Retracked sigma 0 correction.

Spare in Baseline BC and above.

Units: db/100

Definition at line 283 of file L2IStructure.h.

**6.11.2.33 int32\_t L2IData::j\_Retrk\_sig0\_C\_2**

Retracked sigma 0 correction for retracker 2.

Spare in Baseline BC and above.

Units: db/100

Definition at line 290 of file L2IStructure.h.

**6.11.2.34    int32\_t L2IData::j\_Retrk\_sig0\_C\_3**

Retracked sigma 0 correction for retracker 3.

Spare in Baseline BC and above.

Units: db/100

Definition at line 297 of file L2IStructure.h.

**6.11.2.35    int32\_t L2IData::j\_Retrk\_Quality\_1**

Retracker 1 quality metric.

Baseline BC and above.

Units: -

Definition at line 304 of file L2IStructure.h.

**6.11.2.36    int32\_t L2IData::j\_Retrk\_Quality\_2**

Retracker 2 quality metric.

Baseline BC and above.

Units: -

Definition at line 311 of file L2IStructure.h.

**6.11.2.37    int32\_t L2IData::j\_Retrk\_Quality\_3**

Retracker 3 quality metric.

Baseline BC and above.

Units: -

Definition at line 318 of file L2IStructure.h.

**6.11.2.38    int32\_t L2IData::j\_Retrk\_3**

3rd retracker output.

Units: -

Definition at line 325 of file L2IStructure.h.

**6.11.2.39    int32\_t L2IData::j\_Retrk\_4**

4th retracker output.

Units: -

Definition at line 330 of file L2IStructure.h.

**6.11.2.40    int32\_t L2IData::j\_Retrk\_5**

5th retracker output.

Units: -

Definition at line 335 of file L2IStructure.h.



**6.11.2.41 int32\_t L2IData::j\_Retrk\_6**

6th retracker output.

Units: -

Definition at line 340 of file L2IStructure.h.

**6.11.2.42 int32\_t L2IData::j\_Retrk\_7**

7th retracker output.

Units: -

Definition at line 345 of file L2IStructure.h.

**6.11.2.43 int32\_t L2IData::j\_Retrk\_8**

8th retracker output.

Units: -

Definition at line 350 of file L2IStructure.h.

**6.11.2.44 int32\_t L2IData::j\_Retrk\_9**

9th retracker output.

Units: -

Definition at line 355 of file L2IStructure.h.

**6.11.2.45 int32\_t L2IData::j\_Retrk\_10**

10th retracker output.

Units: -

Definition at line 360 of file L2IStructure.h.

**6.11.2.46 int32\_t L2IData::j\_Retrk\_11**

11th retracker output.

Units: -

Definition at line 365 of file L2IStructure.h.

**6.11.2.47 int32\_t L2IData::j\_Retrk\_12**

12th retracker output.

Units: -

Definition at line 370 of file L2IStructure.h.

**6.11.2.48 int32\_t L2IData::j\_Retrk\_13**

13th retracker output.

Units: -

Definition at line 375 of file L2IStructure.h.

#### 6.11.2.49 `int32_t L2IData::j_Retrk_14`

14th retracker output.

Baseline BC and later.

Units: -

Definition at line 380 of file L2IStructure.h.

#### 6.11.2.50 `int32_t L2IData::j_Retrk_15`

15th retracker output.

Baseline BC and later.

Units: -

Definition at line 387 of file L2IStructure.h.

#### 6.11.2.51 `int32_t L2IData::j_Retrk_16`

16th retracker output.

Baseline BC and later.

Units: -

Definition at line 394 of file L2IStructure.h.

#### 6.11.2.52 `int32_t L2IData::j_Retrk_17`

17th retracker output.

Baseline BC and later.

Units: -

Definition at line 401 of file L2IStructure.h.

#### 6.11.2.53 `int32_t L2IData::j_Retrk_18`

18th retracker output.

Baseline BC and later.

Units: -

Definition at line 408 of file L2IStructure.h.

#### 6.11.2.54 `int32_t L2IData::j_Retrk_19`

19th retracker output.

Units: -

Definition at line 415 of file L2IStructure.h.

**6.11.2.55 int32\_t L2IData::j\_Retrk\_20**

20th retracker output.

Baseline BC and later.

Units: -

Definition at line 420 of file L2IStructure.h.

**6.11.2.56 int32\_t L2IData::j\_Retrk\_21**

21st retracker output.

Baseline BC and later.

Units: -

Definition at line 427 of file L2IStructure.h.

**6.11.2.57 int32\_t L2IData::j\_Retrk\_22**

22nd retracker output.

Baseline BC and later.

Units: -

Definition at line 434 of file L2IStructure.h.

**6.11.2.58 int32\_t L2IData::j\_Retrk\_23**

23rd retracker output.

Baseline BC and later.

Units: -

Definition at line 441 of file L2IStructure.h.

**6.11.2.59 int32\_t L2IData::j\_echo\_shape**

Power echo shape parameter.

Units: dB/100

Definition at line 448 of file L2IStructure.h.

**6.11.2.60 int16\_t L2IData::ai\_bb\_param[50]**

Beam behaviour parameter.

The beam behaviour parameter is a  
unitless code number related to  
surface characteristics.

Units: -

Definition at line 453 of file L2IStructure.h.

**6.11.2.61 int32\_t L2IData::j\_XTrack\_angle**

Cross track angle.

At retrack point.

Units: micro-radians

Definition at line 462 of file L2IStructure.h.

**6.11.2.62 int32\_t L2IData::j\_XTrack\_angle\_C**

Cross track angle correction.

Baseline BC and later.

Units: micro-radians

Definition at line 469 of file L2IStructure.h.

**6.11.2.63 int32\_t L2IData::j\_Coherence**

Coherence (leading edge)

At retrack point.

Units: 1/1000

Definition at line 476 of file L2IStructure.h.

**6.11.2.64 int32\_t L2IData::j\_Ocean\_ht**

Interpolated Ocean Height.

- Units: mm

Definition at line 483 of file L2IStructure.h.

**6.11.2.65 int32\_t L2IData::j\_Freeboard**

Freeboard.

Units: mm

Definition at line 488 of file L2IStructure.h.

**6.11.2.66 int32\_t L2IData::j\_SHA**

Surface height anomaly.

Units: mm

Definition at line 493 of file L2IStructure.h.

**6.11.2.67 int32\_t L2IData::j\_SSHA\_interp**

Interpolated sea surface height anomaly.

Units: mm

Definition at line 498 of file L2IStructure.h.

**6.11.2.68    uint16\_t L2IData::ui\_interp\_err**

Error in ocean height interpolation.

Estimated.

Units: mm

Definition at line 504 of file L2IStructure.h.

**6.11.2.69    uint16\_t L2IData::ui\_interp\_cnt\_fwd**

Number of forward records interpolated.

Units: count

Definition at line 511 of file L2IStructure.h.

**6.11.2.70    uint16\_t L2IData::ui\_interp\_cnt\_bkwd**

Number of backward records interpolated.

Units: count

Definition at line 517 of file L2IStructure.h.

**6.11.2.71    uint16\_t L2IData::ui\_interp\_time\_fwd**

Distance in time of most forward record interpolated.

Units: milli-seconds

Definition at line 523 of file L2IStructure.h.

**6.11.2.72    uint16\_t L2IData::ui\_interp\_time\_bkwd**

Distance in time of most backward record interpolated.

Units: milli-seconds

Definition at line 529 of file L2IStructure.h.

**6.11.2.73    uint16\_t L2IData::ui\_interp\_error\_F**

Interpolation error flag. See [t\\_InterpolationErrorFlagBit](#)

Definition at line 535 of file L2IStructure.h.

**6.11.2.74    uint32\_t L2IData::uj\_Meas\_Mode**

Measurement mode. See [t\\_MeasurementMode](#)

Definition at line 538 of file L2IStructure.h.

**6.11.2.75    uint32\_t L2IData::uj\_Quality\_F**

Quality flags. See [t\\_MeasurementQualityFlagBit](#)

Units: -

Definition at line 542 of file L2IStructure.h.

**6.11.2.76    uint32\_t L2IData::uj\_Retracker\_F**

Retracker flags. See [t\\_RetrackerFlagBit](#)

Units: -

Definition at line 547 of file L2IStructure.h.

**6.11.2.77    uint32\_t L2IData::uj\_Ht\_status**

Height calculation details. See [t\\_HeightStatusFlagBit](#)

Specifies what was applied during the height calculation.

Units: -

Definition at line 552 of file L2IStructure.h.

**6.11.2.78    uint32\_t L2IData::uj\_Freeb\_status**

SAR freeboard status flag. See [t\\_FreeboardStatusFlagBit](#)

Units: -

Definition at line 560 of file L2IStructure.h.

**6.11.2.79    uint16\_t L2IData::ui\_n\_avg**

Number of echoes or beams averaged.

Units: -

Definition at line 565 of file L2IStructure.h.

**6.11.2.80    uint16\_t L2IData::ui\_Wind\_speed**

Wind speed.

Units: mm/s

Definition at line 571 of file L2IStructure.h.

**6.11.2.81    uint8\_t L2IData::auc\_spares1[12]**

Spares.

Units: -

Definition at line 575 of file L2IStructure.h.

**6.11.2.82    int32\_t L2IData::j\_Ice\_conc**

Ice concentration parameter.

Units: %/1000

Definition at line 581 of file L2IStructure.h.

**6.11.2.83 int32\_t L2IData::j\_Snow\_depth**

Snow depth.

Units: mm

Definition at line 586 of file L2IStructure.h.

**6.11.2.84 int32\_t L2IData::j\_Snow\_density**

Snow density.

Units: kg/m<sup>3</sup>

Definition at line 591 of file L2IStructure.h.

**6.11.2.85 int32\_t L2IData::j\_Discriminator**

Discriminator result. See [t\\_DiscriminatorResult](#)

Units: Enumeration

Definition at line 596 of file L2IStructure.h.

**6.11.2.86 int32\_t L2IData::j\_SARin\_disc\_1**

SARin discriminator parameter 1.

Units: TBD

Definition at line 601 of file L2IStructure.h.

**6.11.2.87 int32\_t L2IData::j\_SARin\_disc\_2**

SARin discriminator parameter 2.

Units: TBD

Definition at line 606 of file L2IStructure.h.

**6.11.2.88 int32\_t L2IData::j\_SARin\_disc\_3**

SARin discriminator parameter 3.

Units: TBD

Definition at line 611 of file L2IStructure.h.

**6.11.2.89 int32\_t L2IData::j\_SARin\_disc\_4**

SARin discriminator parameter 4.

Units: TBD

Definition at line 616 of file L2IStructure.h.

**6.11.2.90 int32\_t L2IData::j\_SARin\_disc\_5**

SARin discriminator parameter 5. Units: TBD

Definition at line 621 of file L2IStructure.h.

**6.11.2.91 int32\_t L2IData::j\_SARin\_disc\_6**

SARin discriminator parameter 6.

Units: TBD

Definition at line 625 of file L2IStructure.h.

**6.11.2.92 int32\_t L2IData::j\_SARin\_disc\_7**

SARin discriminator parameter 7.

Units: TBD

Definition at line 630 of file L2IStructure.h.

**6.11.2.93 int32\_t L2IData::j\_SARin\_disc\_8**

SARin discriminator parameter 8.

Units: TBD

Definition at line 635 of file L2IStructure.h.

**6.11.2.94 int32\_t L2IData::j\_SARin\_disc\_9**

SARin discriminator parameter 9.

Units: TBD

Definition at line 640 of file L2IStructure.h.

**6.11.2.95 int32\_t L2IData::j\_SARin\_disc\_10**

SARin discriminator parameter 10.

Units: TBD

Definition at line 645 of file L2IStructure.h.

**6.11.2.96 uint32\_t L2IData::uj\_Discrim\_F**

Discriminator flags. See [t\\_DiscriminatorStatusFlagBit](#)

Units: -

Definition at line 650 of file L2IStructure.h.

**6.11.2.97 int32\_t L2IData::j\_Attitude**

Slope model correction.

Attitude of echo.

Units: micro-degrees

Definition at line 655 of file L2IStructure.h.

**6.11.2.98 int32\_t L2IData::j\_Azimuth**

Slope model correction.



Azimuth of echo.

Units: micro-degrees

Definition at line 662 of file L2IStructure.h.

#### 6.11.2.99 `int32_t L2IData::j_Slope_Doppler_C`

Slope doppler correction.

Baseline BC and later.

Units: mm

Definition at line 669 of file L2IStructure.h.

#### 6.11.2.100 `int32_t L2IData::j_Lat_sat`

The original lattitude of the satellite.

Units: micro-degrees

Definition at line 676 of file L2IStructure.h.

#### 6.11.2.101 `int32_t L2IData::j_Lon_sat`

The original longitude of the satellite.

Units: micro degree (-180,180) +ve E does NOT contain 180. -ve W DOES contain -180

Definition at line 682 of file L2IStructure.h.

#### 6.11.2.102 `uint32_t L2IData::uj_Ambiguity`

Ambiguity indicator. See [t\\_Level2AmbiguityErrorFlagBit](#)

Units: -

Definition at line 692 of file L2IStructure.h.

#### 6.11.2.103 `int32_t L2IData::j_MSS_mod`

MSS from standard model.

Units: mm

Definition at line 697 of file L2IStructure.h.

#### 6.11.2.104 `int32_t L2IData::j_Geoid_mod`

Geoid from standard model.

Units: mm

Definition at line 702 of file L2IStructure.h.

#### 6.11.2.105 `int32_t L2IData::j_ODLE_mod`

ODLE from standard model.

Units: mm

Definition at line 707 of file L2IStructure.h.

#### 6.11.2.106 `int32_t L2IData::j_DEM_elev`

The interpolated elevation value obtained from the DEM.

Units: mm

Definition at line 712 of file L2IStructure.h.

#### 6.11.2.107 `uint32_t L2IData::uj_DEM_id`

Identification of DEM used in SARin ambiguity test.

Units: -

Definition at line 718 of file L2IStructure.h.

#### 6.11.2.108 `uint8_t L2IData::auc_spares2[16]`

Spares.

Units: -

Definition at line 723 of file L2IStructure.h.

#### 6.11.2.109 `int32_t L2IData::j_Dry_C`

Dry Tropospheric Correction.

Units: mm

Definition at line 729 of file L2IStructure.h.

#### 6.11.2.110 `int32_t L2IData::j_Wet_C`

Wet Tropospheric Correction.

Units: mm

Definition at line 734 of file L2IStructure.h.

#### 6.11.2.111 `int32_t L2IData::j_IB_C`

Inverse Barometric Correction.

Units: mm

Definition at line 739 of file L2IStructure.h.

#### 6.11.2.112 `int32_t L2IData::j_DAC_C`

Delta Inverse Barometric Correction.

Units: mm

Definition at line 744 of file L2IStructure.h.

**6.11.2.113 int32\_t L2IData::j\_Iono\_GIM**

GIM Ionospheric Correction.

Units: mm

Definition at line 749 of file L2IStructure.h.

**6.11.2.114 int32\_t L2IData::j\_Iono\_mod**

Model Ionospheric Correction.

Units: mm

Definition at line 754 of file L2IStructure.h.

**6.11.2.115 int32\_t L2IData::j\_H\_OT**

Ocean Tide .

Units: mm

Definition at line 759 of file L2IStructure.h.

**6.11.2.116 int32\_t L2IData::j\_H\_LPEOT**

Long-period equilibrium ocean tide.

Units: mm

Definition at line 764 of file L2IStructure.h.

**6.11.2.117 int32\_t L2IData::j\_H\_OLT**

Ocean Loading Tide.

Units: mm

Definition at line 769 of file L2IStructure.h.

**6.11.2.118 int32\_t L2IData::j\_H\_SET**

Solid Earth Tide.

Units: mm

Definition at line 774 of file L2IStructure.h.

**6.11.2.119 int32\_t L2IData::j\_H\_GPT**

Geocentric Polar Tide.

Units: mm

Definition at line 779 of file L2IStructure.h.

**6.11.2.120 uint32\_t L2IData::uj\_Surf\_type**

Surface type flag. See [t\\_SurfaceType](#)

Units: -

Definition at line 784 of file L2IStructure.h.

#### 6.11.2.121 `uint32_t L2IData::uj_Corr_status`

Correction status flag. See [t\\_CorrectionStatusFlagBit](#)

Units: -

Definition at line 789 of file L2IStructure.h.

#### 6.11.2.122 `uint32_t L2IData::uj_Corr_error`

Correction error flags. See [t\\_CorrectionErrorFlagBit](#)

Units: -

Definition at line 794 of file L2IStructure.h.

#### 6.11.2.123 `int32_t L2IData::j_SSB`

Sea state bias correction.

Units: mm

Definition at line 799 of file L2IStructure.h.

#### 6.11.2.124 `uint8_t L2IData::auc_spares3[8]`

Spare.

Units: -

Definition at line 804 of file L2IStructure.h.

#### 6.11.2.125 `int32_t L2IData::j_Dopp_rc`

Doppler range correction.

Radial + slope.

Units: mm

Definition at line 810 of file L2IStructure.h.

#### 6.11.2.126 `int32_t L2IData::j_TR_inst_rc`

Instrument Range Correction.

t-r antenna.

Units: mm

Definition at line 817 of file L2IStructure.h.

#### 6.11.2.127 `int32_t L2IData::j_R_inst_rc`

Instrument Range Correction.

r-only antenna.

Units: mm

Definition at line 824 of file L2IStructure.h.

#### 6.11.2.128 int32\_t L2IData::j\_TR\_inst\_gain\_C

Instrument Sigma 0 Correction.

t-r antenna.

Units: dB/100

Definition at line 831 of file L2IStructure.h.

#### 6.11.2.129 int32\_t L2IData::j\_R\_inst\_gain\_C

Instrument Sigma 0 Correction.

r-only antenna.

Units: dB/100

Definition at line 838 of file L2IStructure.h.

#### 6.11.2.130 int32\_t L2IData::j\_Int\_phase\_C

Internal Phase Correction.

Units: milli-radians

Definition at line 845 of file L2IStructure.h.

#### 6.11.2.131 int32\_t L2IData::j\_Ext\_phase\_C

External Phase Correction.

Units: milli-radians

Definition at line 850 of file L2IStructure.h.

#### 6.11.2.132 int32\_t L2IData::j\_Noise\_pwr

Noise Power measurement.

Noise sampled average. Bandwidth 320 MHz or 40 MHz.

Units: TBD

Definition at line 855 of file L2IStructure.h.

#### 6.11.2.133 int32\_t L2IData::j\_Phase\_slope\_C

Phase slope correction.

Units: microradians

Definition at line 863 of file L2IStructure.h.

#### 6.11.2.134 uint8\_t L2IData::auc\_spares4[8]

Spares.

Units: -

Definition at line 868 of file L2IStructure.h.

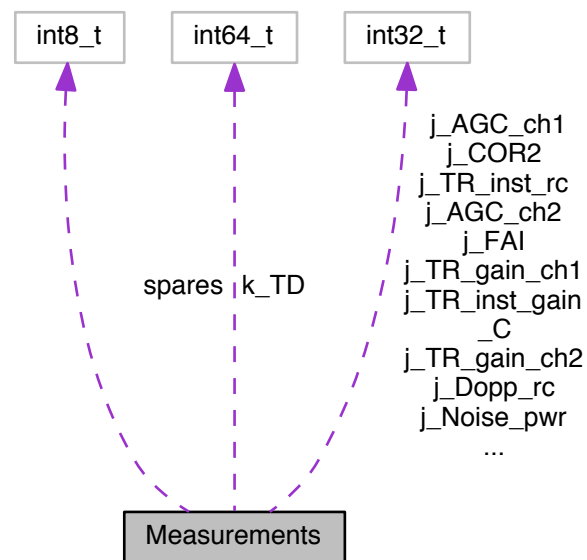
The documentation for this struct was generated from the following file:

- [L2IStructure.h](#)

## 6.12 Measurements Struct Reference

```
#include <L1bStructure.h>
```

Collaboration diagram for Measurements:



### Data Fields

- [int64\\_t k\\_TD](#)
- [int32\\_t j\\_H0](#)
- [int32\\_t j\\_COR2](#)
- [int32\\_t j\\_LAI](#)
- [int32\\_t j\\_FAI](#)
- [int32\\_t j\\_AGC\\_ch1](#)
- [int32\\_t j\\_AGC\\_ch2](#)
- [int32\\_t j\\_TR\\_gain\\_ch1](#)
- [int32\\_t j\\_TR\\_gain\\_ch2](#)
- [int32\\_t j\\_Tx\\_power](#)
- [int32\\_t j\\_Dopp\\_rc](#)
- [int32\\_t j\\_TR\\_inst\\_rc](#)
- [int32\\_t j\\_R\\_inst\\_rc](#)
- [int32\\_t j\\_TR\\_inst\\_gain\\_C](#)

- [int32\\_t j\\_R\\_inst\\_gain\\_C](#)
- [int32\\_t j\\_Int\\_phase\\_C](#)
- [int32\\_t j\\_Ext\\_phase\\_C](#)
- [int32\\_t j\\_Noise\\_pwr](#)
- [int32\\_t j\\_Phase\\_slope\\_C](#)
- [int8\\_t spares](#) [4]

### 6.12.1 Detailed Description

Level 1b record substructure containing measurement related parameters.

Definition at line 177 of file L1bStructure.h.

### 6.12.2 Field Documentation

#### 6.12.2.1 [int64\\_t Measurements::k\\_TD](#)

Window delay referenced to COM. Two-way and corrected for instrument delays but not geophysical effects.

Units:  $10^{-12}$  seconds

Definition at line 179 of file L1bStructure.h.

#### 6.12.2.2 [int32\\_t Measurements::j\\_H0](#)

H0 initial height word.

Units: 48.8 ps

Definition at line 186 of file L1bStructure.h.

#### 6.12.2.3 [int32\\_t Measurements::j\\_COR2](#)

COR2 Height Rate

Units: 3.05 ps/rc

Definition at line 190 of file L1bStructure.h.

#### 6.12.2.4 [int32\\_t Measurements::j\\_LAI](#)

Coarse Range Word LAI.

Units: 12.5 ns

Definition at line 194 of file L1bStructure.h.

#### 6.12.2.5 [int32\\_t Measurements::j\\_FAI](#)

Fine Range Word FAI.

Units: 12.5/256 ns

Definition at line 198 of file L1bStructure.h.

#### 6.12.2.6 `int32_t Measurements::j_AGC_ch1`

AGC channel 1. Gain calibration has been applied.

Units: dB/100

Definition at line 202 of file L1bStructure.h.

#### 6.12.2.7 `int32_t Measurements::j_AGC_ch2`

AGC channel 2. Gain calibration has been applied.

Units: dB/100

Definition at line 207 of file L1bStructure.h.

#### 6.12.2.8 `int32_t Measurements::j_TR_gain_ch1`

Total fixed gain on channel 1.

Units: dB/100

Definition at line 212 of file L1bStructure.h.

#### 6.12.2.9 `int32_t Measurements::j_TR_gain_ch2`

Total fixed gain on channel 2.

Units: dB/100

Definition at line 216 of file L1bStructure.h.

#### 6.12.2.10 `int32_t Measurements::j_Tx_power`

Transmit power.

Units: micro-watts

Definition at line 220 of file L1bStructure.h.

#### 6.12.2.11 `int32_t Measurements::j_Dopp_rc`

Doppler range correction. Radial component.

Units: mm

Definition at line 224 of file L1bStructure.h.

#### 6.12.2.12 `int32_t Measurements::j_TR_inst_rc`

Instrument range correction. Tx-Rx antenna. From CAL1.

Units: mm

Definition at line 229 of file L1bStructure.h.

#### 6.12.2.13 `int32_t Measurements::j_R_inst_rc`

Instrument range correction. Rx only antenna. From CAL1.

Units: mm



Definition at line 235 of file L1bStructure.h.

#### 6.12.2.14 int32\_t Measurements::j\_TR\_inst\_gain\_C

Instrument gain correction. Tx-Rx antenna. From CAL1.

Units: dB/100

Definition at line 241 of file L1bStructure.h.

#### 6.12.2.15 int32\_t Measurements::j\_R\_inst\_gain\_C

Instrument gain correction. Rx only antenna. From CAL1.

Units: dB/100

Definition at line 247 of file L1bStructure.h.

#### 6.12.2.16 int32\_t Measurements::j\_Int\_phase\_C

Internal phase correction from CAL4.

Units: microradians

Definition at line 254 of file L1bStructure.h.

#### 6.12.2.17 int32\_t Measurements::j\_Ext\_phase\_C

External phase correction.

Units: microradians

Definition at line 259 of file L1bStructure.h.

#### 6.12.2.18 int32\_t Measurements::j\_Noise\_pwr

Noise power measurement.

Units: dB/100

Definition at line 264 of file L1bStructure.h.

#### 6.12.2.19 int32\_t Measurements::j\_Phase\_slope\_C

Phase slope correction.

Units: Microradians

Definition at line 268 of file L1bStructure.h.

#### 6.12.2.20 int8\_t Measurements::spares[4]

Spares.

Definition at line 272 of file L1bStructure.h.

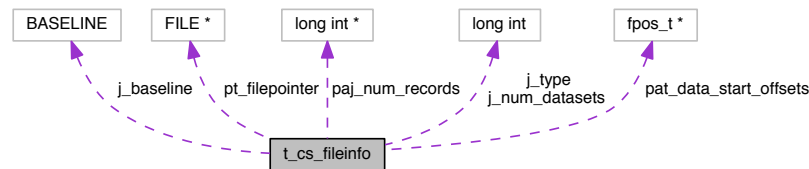
The documentation for this struct was generated from the following file:

- [L1bStructure.h](#)

## 6.13 t\_cs\_fileinfo Struct Reference

```
#include <ptCSGetFileHandle.h>
```

Collaboration diagram for t\_cs\_fileinfo:



### Data Fields

- FILE \* [pt\\_filepointer](#)
- long int [j\\_num\\_datasets](#)
- fpos\_t \* [pat\\_data\\_start\\_offsets](#)
- long int \* [paj\\_num\\_records](#)
- long int [j\\_type](#)
- BASELINE [j\\_baseline](#)

### 6.13.1 Detailed Description

< File information struture/

Examples:

[examples/cs\\_l1b\\_io\\_example.c](#), [examples/cs\\_l2\\_io\\_example.c](#), [examples/cs\\_l2i\\_io\\_example.c](#), and [examples/example\\_bit\\_manipulation.c](#).

Definition at line 92 of file `ptCSGetFileHandle.h`.

### 6.13.2 Field Documentation

#### 6.13.2.1 FILE\* t\_cs\_fileinfo::pt\_filepointer

Filepointer to this file.

Definition at line 94 of file `ptCSGetFileHandle.h`.

#### 6.13.2.2 long int t\_cs\_fileinfo::j\_num\_datasets

Number of datasets in this file.

Definition at line 95 of file `ptCSGetFileHandle.h`.

#### 6.13.2.3 fpos\_t\* t\_cs\_fileinfo::pat\_data\_start\_offsets

Array of byte offsets to datasets.

Definition at line 96 of file `ptCSGetFileHandle.h`.

6.13.2.4 `long int* t_cs_fileinfo::paj_num_records`

Array of counts of records in each dataset.

Definition at line 97 of file `ptCSGetFileHandle.h`.

6.13.2.5 `long int t_cs_fileinfo::j_type`

Filetype of this file.

Definition at line 98 of file `ptCSGetFileHandle.h`.

6.13.2.6 **BASELINE** `t_cs_fileinfo::j_baseline`

Baseline that the product format of this file conforms to.

Definition at line 99 of file `ptCSGetFileHandle.h`.

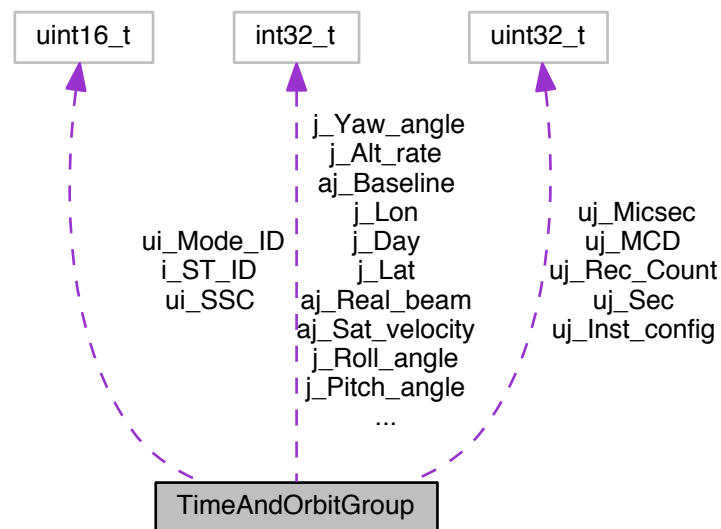
The documentation for this struct was generated from the following file:

- [ptCSGetFileHandle.h](#)

## 6.14 TimeAndOrbitGroup Struct Reference

```
#include <L1bStructure.h>
```

Collaboration diagram for TimeAndOrbitGroup:



### Data Fields

- `int32_t j_Day`
- `uint32_t uj_Sec`

- [uint32\\_t uj\\_Micsec](#)
- [int32\\_t j\\_USO\\_Corr](#)
- [uint16\\_t ui\\_Mode\\_ID](#)
- [uint16\\_t ui\\_SSC](#)
- [uint32\\_t uj\\_Inst\\_config](#)
- [uint32\\_t uj\\_Rec\\_Count](#)
- [int32\\_t j\\_Lat](#)
- [int32\\_t j\\_Lon](#)
- [int32\\_t j\\_Alt](#)
- [int32\\_t j\\_Alt\\_rate](#)
- [int32\\_t aj\\_Sat\\_velocity](#) [3]
- [int32\\_t aj\\_Real\\_beam](#) [3]
- [int32\\_t aj\\_Baseline](#) [3]
- [uint16\\_t i\\_ST\\_ID](#)
- [int32\\_t j\\_Roll\\_angle](#)
- [int32\\_t j\\_Pitch\\_angle](#)
- [int32\\_t j\\_Yaw\\_angle](#)
- [uint32\\_t uj\\_MCD](#)

### 6.14.1 Detailed Description

Level 1b record substructure containing time and orbit parameters.

Definition at line 101 of file L1bStructure.h.

### 6.14.2 Field Documentation

#### 6.14.2.1 `int32_t TimeAndOrbitGroup::j_Day`

MDSR Time stamp days.

Definition at line 103 of file L1bStructure.h.

#### 6.14.2.2 `uint32_t TimeAndOrbitGroup::uj_Sec`

MDSR Time stamp seconds.

Definition at line 104 of file L1bStructure.h.

#### 6.14.2.3 `uint32_t TimeAndOrbitGroup::uj_Micsec`

MDSR Time stamp microseconds.

Definition at line 105 of file L1bStructure.h.

#### 6.14.2.4 `int32_t TimeAndOrbitGroup::j_USO_Corr`

USO Correction Factor.

Definition at line 106 of file L1bStructure.h.

#### 6.14.2.5 `uint16_t TimeAndOrbitGroup::ui_Mode_ID`

Mode ID. See [t\\_ModelID](#)

Definition at line 107 of file L1bStructure.h.

#### 6.14.2.6 uint16\_t TimeAndOrbitGroup::ui\_SSC

Source sequence counter.

Definition at line 108 of file L1bStructure.h.

#### 6.14.2.7 uint32\_t TimeAndOrbitGroup::uj\_Inst\_config

Instrument Configuration. See [t\\_InstConf](#)

Definition at line 109 of file L1bStructure.h.

#### 6.14.2.8 uint32\_t TimeAndOrbitGroup::uj\_Rec\_Count

Record Counter.

Definition at line 110 of file L1bStructure.h.

#### 6.14.2.9 int32\_t TimeAndOrbitGroup::j\_Lat

Latitude at nadir.

Units:  $10^{-7}$  degrees

Definition at line 111 of file L1bStructure.h.

#### 6.14.2.10 int32\_t TimeAndOrbitGroup::j\_Lon

Longitude at nadir.

Units:  $10^{-7}$  degrees

Definition at line 115 of file L1bStructure.h.

#### 6.14.2.11 int32\_t TimeAndOrbitGroup::j\_Alt

Altitude of COG above reference ellipsoid.

Units: mm

Definition at line 119 of file L1bStructure.h.

#### 6.14.2.12 int32\_t TimeAndOrbitGroup::j\_Alt\_rate

Instantaneous altitude rate derived from orbit.

Units: mm/s

Definition at line 124 of file L1bStructure.h.

#### 6.14.2.13 int32\_t TimeAndOrbitGroup::aj\_Sat\_velocity[3]

Satellite velocity vector. In ITRF.

Units: mm/s

Definition at line 129 of file L1bStructure.h.

#### 6.14.2.14 `int32_t TimeAndOrbitGroup::aj_Real_beam[3]`

Real beam direction vector. In CRF.

Units: micrometer

Definition at line 135 of file L1bStructure.h.

#### 6.14.2.15 `int32_t TimeAndOrbitGroup::aj_Baseline[3]`

Interferometer baseline vector. In CRF.

Units: micrometer

Definition at line 141 of file L1bStructure.h.

#### 6.14.2.16 `uint16_t TimeAndOrbitGroup::i_ST_ID`

Star Tracker ID. See [t\\_STID](#) Baseline BC and later only

Units: enum

Definition at line 147 of file L1bStructure.h.

#### 6.14.2.17 `int32_t TimeAndOrbitGroup::j_Roll_angle`

Spacecraft (antenna bench) roll angle. Baseline C only. Derived from star trackers.

Units:  $10^{-7}$  degrees

Definition at line 152 of file L1bStructure.h.

#### 6.14.2.18 `int32_t TimeAndOrbitGroup::j_Pitch_angle`

Spacecraft (antenna bench) pitch angle. Baseline C only. Derived from star trackers.

Units:  $10^{-7}$  degrees

Definition at line 158 of file L1bStructure.h.

#### 6.14.2.19 `int32_t TimeAndOrbitGroup::j_Yaw_angle`

Spacecraft (antenna bench) yaw angle. Baseline C only. Derived from star trackers.

Units:  $10^{-7}$  degrees

Definition at line 164 of file L1bStructure.h.

#### 6.14.2.20 `uint32_t TimeAndOrbitGroup::uj_MCD`

L1b Measurement Confidence Data, see [t\\_MCDFlagBit](#)

Definition at line 170 of file L1bStructure.h.

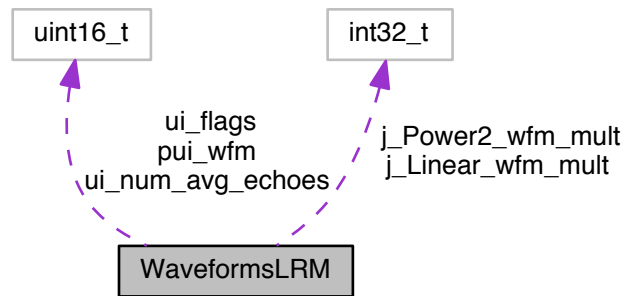
The documentation for this struct was generated from the following file:

- [L1bStructure.h](#)

## 6.15 WaveformsLRM Struct Reference

```
#include <L1bStructure.h>
```

Collaboration diagram for WaveformsLRM:



### Data Fields

- uint16\_t \* [pui\\_wfm](#)
- int32\_t [j\\_Linear\\_wfm\\_mult](#)
- int32\_t [j\\_Power2\\_wfm\\_mult](#)
- uint16\_t [ui\\_num\\_avg\\_echoes](#)
- uint16\_t [ui\\_flags](#)

### 6.15.1 Detailed Description

Level 1b record substructure (LRM mode specific) containing waveform data.

Definition at line 444 of file L1bStructure.h.

### 6.15.2 Field Documentation

#### 6.15.2.1 uint16\_t\* WaveformsLRM::pui\_wfm

Power waveform.

Definition at line 446 of file L1bStructure.h.

#### 6.15.2.2 int32\_t WaveformsLRM::j\_Linear\_wfm\_mult

Echo scale factor.

Definition at line 447 of file L1bStructure.h.

#### 6.15.2.3 int32\_t WaveformsLRM::j\_Power2\_wfm\_mult

Echo scale factor.

Definition at line 448 of file L1bStructure.h.

#### 6.15.2.4 uint16\_t WaveformsLRM::ui\_num\_avg\_echoes

Number of echoes averaged.

Definition at line 449 of file L1bStructure.h.

#### 6.15.2.5 uint16\_t WaveformsLRM::ui\_flags

Flags, see [t\\_Level1bWfmLRMFlagBit](#)

Definition at line 450 of file L1bStructure.h.

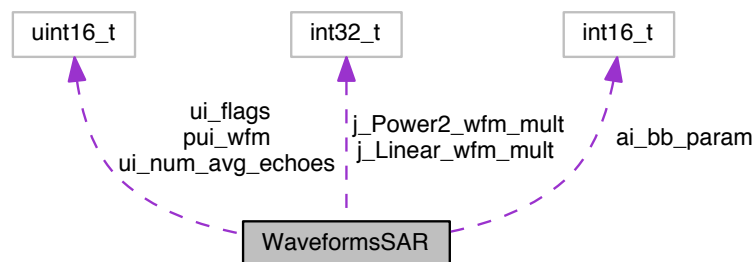
The documentation for this struct was generated from the following file:

- [L1bStructure.h](#)

## 6.16 WaveformsSAR Struct Reference

```
#include <L1bStructure.h>
```

Collaboration diagram for WaveformsSAR:



### Data Fields

- uint16\_t \* [pui\\_wfm](#)
- int32\_t [j\\_Linear\\_wfm\\_mult](#)
- int32\_t [j\\_Power2\\_wfm\\_mult](#)
- uint16\_t [ui\\_num\\_avg\\_echoes](#)
- uint16\_t [ui\\_flags](#)
- int16\_t [ai\\_bb\\_param](#) [[J\\_NUM\\_BB\\_PARAMETERS](#)]

### 6.16.1 Detailed Description

Level 1b record substructure (SAR mode specific) containing waveform data.

Definition at line 455 of file L1bStructure.h.



## 6.16.2 Field Documentation

### 6.16.2.1 uint16\_t\* WaveformsSAR::pui\_wfm

Power waveform.

Definition at line 457 of file L1bStructure.h.

### 6.16.2.2 int32\_t WaveformsSAR::j\_Linear\_wfm\_mult

Echo scale factor.

Definition at line 458 of file L1bStructure.h.

### 6.16.2.3 int32\_t WaveformsSAR::j\_Power2\_wfm\_mult

Echo scale factor.

Definition at line 459 of file L1bStructure.h.

### 6.16.2.4 uint16\_t WaveformsSAR::ui\_num\_avg\_echoes

Number of echoes averaged.

Definition at line 460 of file L1bStructure.h.

### 6.16.2.5 uint16\_t WaveformsSAR::ui\_flags

Flags, see [t\\_Level1bWfmSARFlagBit](#)

Definition at line 461 of file L1bStructure.h.

### 6.16.2.6 int16\_t WaveformsSAR::ai\_bb\_param[J\_NUM\_BB\_PARAMETERS]

Beam behaviour parameters.

Definition at line 462 of file L1bStructure.h.

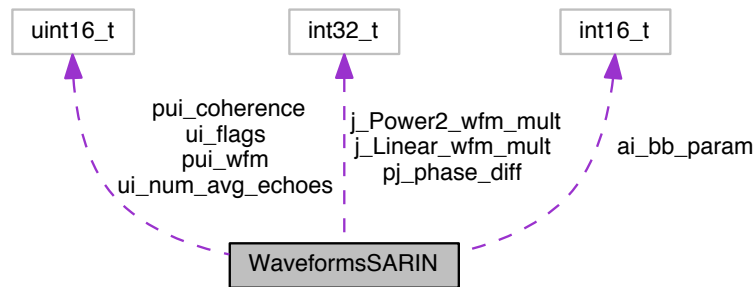
The documentation for this struct was generated from the following file:

- [L1bStructure.h](#)

## 6.17 WaveformsSARIN Struct Reference

```
#include <L1bStructure.h>
```

Collaboration diagram for WaveformsSARIN:



## Data Fields

- uint16\_t \* [pui\\_wfm](#)
- int32\_t [j\\_Linear\\_wfm\\_mult](#)
- int32\_t [j\\_Power2\\_wfm\\_mult](#)
- uint16\_t [ui\\_num\\_avg\\_echoes](#)
- uint16\_t [ui\\_flags](#)
- int16\_t [ai\\_bb\\_param](#) [[J\\_NUM\\_BB\\_PARAMETERS](#)]
- uint16\_t \* [pui\\_coherence](#)
- int32\_t \* [pj\\_phase\\_diff](#)

### 6.17.1 Detailed Description

Level 1b record substructure (SARin mode specific) containing waveform data.

Definition at line 469 of file L1bStructure.h.

### 6.17.2 Field Documentation

#### 6.17.2.1 uint16\_t\* WaveformsSARIN::pui\_wfm

Power waveform.

Definition at line 471 of file L1bStructure.h.

#### 6.17.2.2 int32\_t WaveformsSARIN::j\_Linear\_wfm\_mult

Echo scale factor.

Definition at line 472 of file L1bStructure.h.

#### 6.17.2.3 int32\_t WaveformsSARIN::j\_Power2\_wfm\_mult

Echo scale factor.

Definition at line 473 of file L1bStructure.h.

#### 6.17.2.4 `uint16_t WaveformsSARIN::ui_num_avg_echoes`

Number of echoes averaged.

Definition at line 474 of file `L1bStructure.h`.

#### 6.17.2.5 `uint16_t WaveformsSARIN::ui_flags`

Flags, see `t_Level1bWfmSARinFlagBit`

Definition at line 475 of file `L1bStructure.h`.

#### 6.17.2.6 `int16_t WaveformsSARIN::ai_bb_param[J_NUM_BB_PARAMETERS]`

Beam behaviour parameters.

Definition at line 476 of file `L1bStructure.h`.

#### 6.17.2.7 `uint16_t* WaveformsSARIN::pui_coherence`

Coherence waveform.

Definition at line 478 of file `L1bStructure.h`.

#### 6.17.2.8 `int32_t* WaveformsSARIN::pj_phase_diff`

Phase difference waveform.

Definition at line 479 of file `L1bStructure.h`.

The documentation for this struct was generated from the following file:

- [L1bStructure.h](#)



## Chapter 7

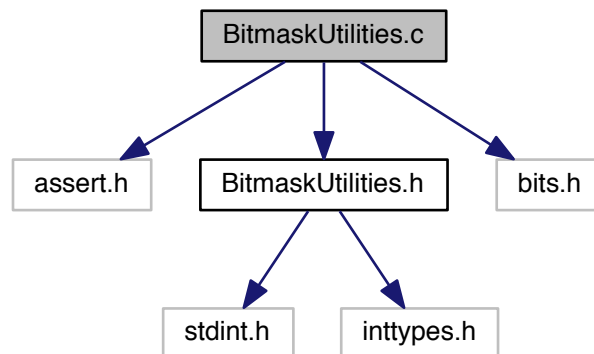
# File Documentation

### 7.1 BitmaskUtilities.c File Reference

Implementation of functions to manipulate bitmasks.

```
#include <assert.h>
#include "BitmaskUtilities.h"
#include "bits.h"
```

Include dependency graph for BitmaskUtilities.c:



### Functions

- void **vSetBitShort** (uint16\_t \*pBitmask, uint16\_t bit)
- int16\_t **iCheckBitShort** (uint16\_t \*pBitmask, uint16\_t bit)
- void **vClearBitShort** (uint16\_t \*pBitmask, uint16\_t bit)
- void **vSetBitLong** (uint32\_t \*pBitmask, uint16\_t bit)
- int16\_t **iCheckBitLong** (uint32\_t \*pBitmask, uint16\_t bit)
- void **vClearBitLong** (uint32\_t \*pBitmask, uint16\_t bit)
- void **vSetBitLongLong** (uint64\_t \*pBitmask, uint16\_t bit)
- int16\_t **iCheckBitLongLong** (uint64\_t \*pBitmask, uint16\_t bit)
- void **vClearBitLongLong** (uint64\_t \*pBitmask, uint16\_t bit)

### 7.1.1 Detailed Description

Implementation of functions to manipulate bitmasks.

### 7.1.2 Function Documentation

7.1.2.1 void vClearBitLong ( uint32\_t \* *pBitmask*, uint16\_t *bit* )

Clear a bit in a int32\_teger bitmask.

Parameters

<i>pBitmask</i>	A pointer to the bitmask.
<i>bit</i>	The bit index to clear. Bit 0 is the LSB.

Definition at line 161 of file BitmaskUtilities.c.

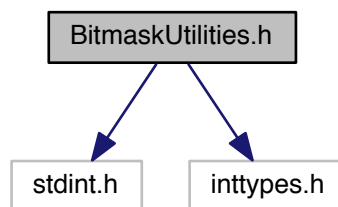
## 7.2 BitmaskUtilities.h File Reference

Bitmask manipulation utilities.

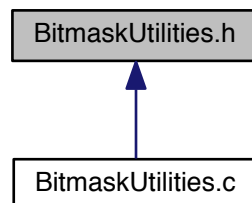
```
#include <stdint.h>
```

```
#include <inttypes.h>
```

Include dependency graph for BitmaskUtilities.h:



This graph shows which files directly or indirectly include this file:



## Functions

- void [vSetBitShort](#) (uint16\_t \*pBitmask, uint16\_t bit)
- int16\_t [iCheckBitShort](#) (uint16\_t \*pBitmask, uint16\_t bit)
- void [vClearBitShort](#) (uint16\_t \*pBitmask, uint16\_t bit)
- void [vSetBitLong](#) (uint32\_t \*pBitmask, uint16\_t bit)
- int16\_t [iCheckBitLong](#) (uint32\_t \*pBitmask, uint16\_t bit)
- void [vClearBitLong](#) (uint32\_t \*pBitmask, uint16\_t bit)
- void [vSetBitLongLong](#) (uint64\_t \*pBitmask, uint16\_t bit)
- int16\_t [iCheckBitLongLong](#) (uint64\_t \*pBitmask, uint16\_t bit)
- void [vClearBitLongLong](#) (uint64\_t \*pBitmask, uint16\_t bit)

### 7.2.1 Detailed Description

Bitmask manipulation utilities.

### 7.2.2 Function Documentation

#### 7.2.2.1 void [vClearBitLong](#) ( uint32\_t \* *pBitmask*, uint16\_t *bit* )

Clear a bit in a int32\_teger bitmask.

##### Parameters

<i>pBitmask</i>	A pointer to the bitmask.
<i>bit</i>	The bit index to clear. Bit 0 is the LSB.

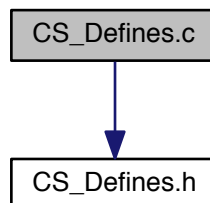
Definition at line 161 of file BitmaskUtilities.c.

## 7.3 CS\_Defines.c File Reference

Defines and populates global variables.

```
#include "CS_Defines.h"
```

Include dependency graph for CS\_Defines.c:



## Variables

- char \* [as\\_CS\\_DS\\_NAME](#) [CS\_DS\_NAME\_COUNT]
- int [gj\\_verbose](#) = 0

- `int J_CS_L2_RECORD_SIZE = 980`
- `int J_CS_L2I_RECORD_SIZE = 556`

### 7.3.1 Detailed Description

Defines and populates global variables.

### 7.3.2 Variable Documentation

#### 7.3.2.1 `char* as_CS_DS_NAME[CS_DS_NAME_COUNT]`

**Initial value:**

```
= {
    "SIR_LRM_L2",
    "SIR_FDM_L2",
    "SIR_SAR_L2A",
    "SIR_SAR_L2B",
    "SIR_SAR_L2",
    "SIR_SIN_L2",
    "SIR_SID_L2",
    "SIR_GDR_2_",
    "SIR_GDR_2A",
    "SIR_GDR_2B",
    "SIR_LRM_L2_I",
    "SIR_SAR_L2A_I",
    "SIR_SAR_L2B_I",
    "SIR_SAR_L2_I",
    "SIR_SIN_L2_I",
    "SIR_SID_L2_I",
    "SIR_GDR_2_I",
    "SIR_GDR_2A_I",
    "SIR_GDR_2B_I",
    "SIR_L1B_LRM",
    "SIR_L1B_FDM",
    "SIR_L1B_SAR",
    "SIR_L1B_SARIN"
}
```

Strings to match within DS\_NAME field of ASCII header to identify file type.

Definition at line 89 of file CS\_Defines.c.

#### 7.3.2.2 `int gj_verbose = 0`

Verbose logging flag. 0 = quiet, 1 = verbose

Definition at line 116 of file CS\_Defines.c.

#### 7.3.2.3 `int J_CS_L2_RECORD_SIZE = 980`

Size of the CryoSat Level 2 record on disk, in bytes.

Definition at line 119 of file CS\_Defines.c.

#### 7.3.2.4 `int J_CS_L2I_RECORD_SIZE = 556`

Size of the CryoSat Level 2I record on disk, in bytes.

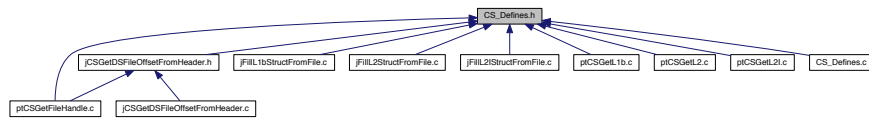
Definition at line 122 of file CS\_Defines.c.



## 7.4 CS\_Defines.h File Reference

Contains types and macros needed by the functions in this library.

This graph shows which files directly or indirectly include this file:



### Macros

- `#define J_MPH_SIZE` 1247
- `#define TRUE` 1
- `#define FALSE` 0

### Enumerations

- enum `CS_DS_NAME` {  
`CS_L2_DS_NAME_LRM`, `CS_L2_DS_NAME_FDM`, `CS_L2_DS_NAME_SAR_A`, `CS_L2_DS_NAME_SAR_B`,  
`CS_L2_DS_NAME_SAR`, `CS_L2_DS_NAME_SIN`, `CS_L2_DS_NAME_SID`, `CS_L2_DS_NAME_GDR`,  
`CS_L2_DS_NAME_GDR_A`, `CS_L2_DS_NAME_GDR_B`, `CS_L2I_DS_NAME_LRM`, `CS_L2I_DS_NAME_SAR_A`,  
`CS_L2I_DS_NAME_SAR_B`, `CS_L2I_DS_NAME_SAR`, `CS_L2I_DS_NAME_SIN`, `CS_L2I_DS_NAME_SID`,  
`CS_L2I_DS_NAME_GDR`, `CS_L2I_DS_NAME_GDR_A`, `CS_L2I_DS_NAME_GDR_B`, `CS_L1B_DS_NAME_LRM`,  
`CS_L1B_DS_NAME_FDM`, `CS_L1B_DS_NAME_SAR`, `CS_L1B_DS_NAME_SARIN`, `CS_DS_NAME_COUNT`,  
`CS_L2_DS_NAME_START` = ( `CS_L2_DS_NAME_LRM` ), `CS_L2_DS_NAME_END` = ( `CS_L2_DS_NAME_GDR_B` ),  
`CS_L2I_DS_NAME_START` = ( `CS_L2I_DS_NAME_LRM` ), `CS_L2I_DS_NAME_END` = ( `CS_L2I_DS_NAME_GDR_B` ),  
`CS_L1B_DS_NAME_START` = ( `CS_L1B_DS_NAME_LRM` ), `CS_L1B_DS_NAME_END` = ( `CS_L1B_DS_NAME_SARIN` ),  
`CS_L2_DS_NAME_ANY`, `CS_L2I_DS_NAME_ANY`,  
`CS_L1B_DS_NAME_ANY`, `CS_DS_NAME_ANY`, `CS_DS_NAME_NONE` }

### Variables

- `char * as_CS_DS_NAME` [`CS_DS_NAME_COUNT`]
- `int J_CS_L2_RECORD_SIZE`
- `int J_CS_L2I_RECORD_SIZE`
- `int gj_verbose`

#### 7.4.1 Detailed Description

Contains types and macros needed by the functions in this library.

This file also contains the top-level documentation to be extracted by Doxygen to form the API reference.

## 7.4.2 Macro Definition Documentation

### 7.4.2.1 `#define J_MPH_SIZE 1247`

Size of the MPH in bytes

Definition at line 294 of file `CS_Defines.h`.

### 7.4.2.2 `#define TRUE 1`

TRUE

Definition at line 303 of file `CS_Defines.h`.

### 7.4.2.3 `#define FALSE 0`

FALSE

Definition at line 304 of file `CS_Defines.h`.

## 7.4.3 Enumeration Type Documentation

### 7.4.3.1 `enum CS_DS_NAME`

Representation of DS\_NAME types read by the CryoSat Level 2 IO Library.

Definition at line 241 of file `CS_Defines.h`.

## 7.4.4 Variable Documentation

### 7.4.4.1 `char* as_CS_DS_NAME[CS_DS_NAME_COUNT]`

External declaration. See [CS\\_Defines.c](#) for definition.

Strings to match within DS\_NAME field of ASCII header to identify file type.

Definition at line 89 of file `CS_Defines.c`.

### 7.4.4.2 `int J_CS_L2_RECORD_SIZE`

Size of the CryoSat Level 2 record on disk, in bytes.

Definition at line 119 of file `CS_Defines.c`.

### 7.4.4.3 `int J_CS_L2I_RECORD_SIZE`

Size of the CryoSat Level 2I record on disk, in bytes.

Definition at line 122 of file `CS_Defines.c`.

### 7.4.4.4 `int gj_verbose`

Verbose logging flag. 0 = quiet, 1 = verbose

Examples:

[examples/cs\\_l1b\\_io\\_example.c](#), [examples/cs\\_l2\\_io\\_example.c](#), and [examples/cs\\_l2i\\_io\\_example.c](#).

Definition at line 116 of file CS\_Defines.c.

## 7.5 Documentation.h File Reference

Documentation for IDL commands.

### 7.5.1 Detailed Description

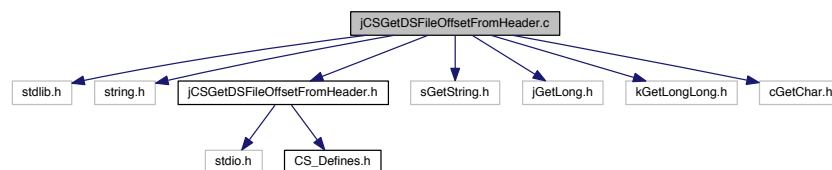
Documentation for IDL commands.

## 7.6 jCSGetDSFileOffsetFromHeader.c File Reference

Implementation of functions to extract the offset of the data set from the ASCII header.

```
#include <stdlib.h>
#include <string.h>
#include "jCSGetDSFileOffsetFromHeader.h"
#include "sGetString.h"
#include "jGetLong.h"
#include "kGetLongLong.h"
#include "cGetChar.h"
```

Include dependency graph for jCSGetDSFileOffsetFromHeader.c:



## Functions

- static long int [jCheckDSDetails](#) (char \*ac\_buffer, long int j\_DSD\_Size)  
*Check that this is a measurement dataset.*
- long int [jCSGetDSFileOffsetFromHeader](#) (fpos\_t \*pt\_dataset\_file\_offset, long int \*pj\_num\_records, char \*s←\_filename, [CS\\_DS\\_NAME](#) t\_ds\_name, long int \*pj\_actual\_ds)

### 7.6.1 Detailed Description

Implementation of functions to extract the offset of the data set from the ASCII header.

### 7.6.2 Function Documentation

7.6.2.1 static long int [jCheckDSDetails](#) ( char \* ac\_buffer, long int j\_DSD\_Size ) [static]

Check that this is a measurement dataset.

**Parameters**

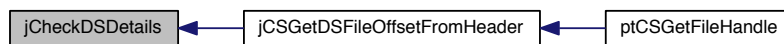
<i>ac_buffer</i>	Pointer to ASCII buffer.
<i>j_DSD_Size</i>	Size of buffer.

**Returns**

-1 in the case of error, 0 if this is not a measurement dataset, 1 if this is a measurement dataset.

Definition at line 448 of file jCSGetDSFileOffsetFromHeader.c.

Here is the caller graph for this function:

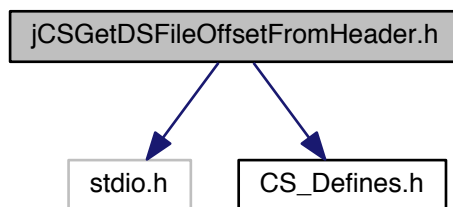


## 7.7 jCSGetDSFileOffsetFromHeader.h File Reference

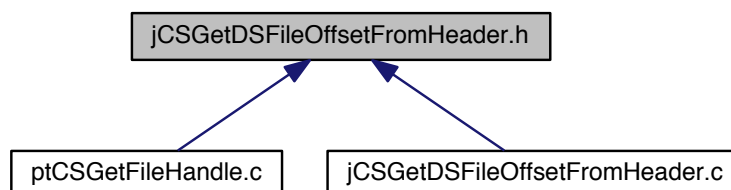
Get the offset of a dataset from the header.

```
#include <stdio.h>
#include "CS_Defines.h"
```

Include dependency graph for jCSGetDSFileOffsetFromHeader.h:



This graph shows which files directly or indirectly include this file:



## Functions

- long int [jCSGetDSFileOffsetFromHeader](#) (fpos\_t \*pt\_dataset\_file\_offset, long int \*pj\_num\_records, char \*s←\_filename, [CS\\_DS\\_NAME](#) t\_ds\_name, long int \*pj\_actual\_ds)

### 7.7.1 Detailed Description

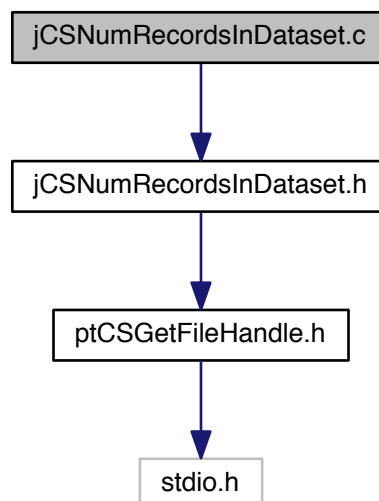
Get the offset of a dataset from the header.

## 7.8 jCSNumRecordsInDataset.c File Reference

Implementation of function to retrieve number of records in a dataset within a file.

```
#include "jCSNumRecordsInDataset.h"
```

Include dependency graph for jCSNumRecordsInDataset.c:



## Functions

- long int [jCSNumRecordsInDataset](#) ([t\\_cs\\_filehandle](#) t\_handle, long int j\_dataset)

### 7.8.1 Detailed Description

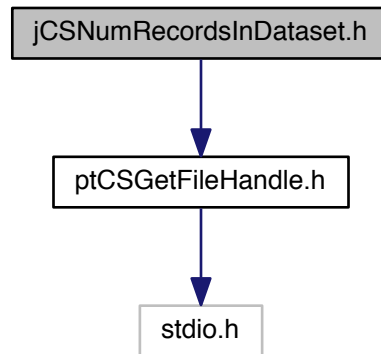
Implementation of function to retrieve number of records in a dataset within a file.

## 7.9 jCSNumRecordsInDataset.h File Reference

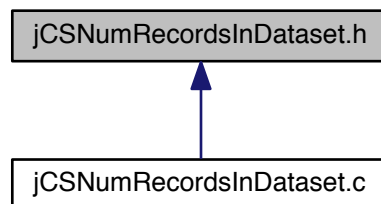
Function to retrieve number of records in a dataset within a file.

```
#include "ptCSGetFileHandle.h"
```

Include dependency graph for `jCSNumRecordsInDataset.h`:



This graph shows which files directly or indirectly include this file:



## Functions

- `long int jCSNumRecordsInDataset (t_cs_filehandle t_handle, long int j_dataset)`

### 7.9.1 Detailed Description

Function to retrieve number of records in a dataset within a file.

## 7.10 jFillL1bStructFromFile.c File Reference

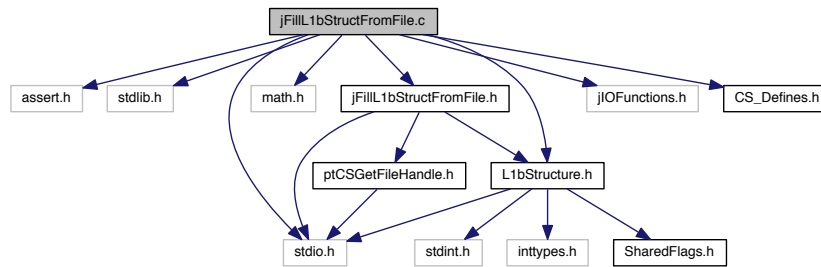
Implementation of low-level functions to read Cryosat Level 1b data in LRM, SAR, or SARin modes.

```

#include <assert.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include "jFiLL1bStructFromFile.h"
#include "L1bStructure.h"
#include "jIOFunctions.h"
#include "CS_Defines.h"

```

Include dependency graph for jFiLL1bStructFromFile.c:



## Functions

- long int [jFiLL1bLRMStructFromFile](#) ([L1B\\_LRM](#) \*p\_ToFill, FILE \*pt\_input, [BASELINE](#) j\_Baseline)
- long int [jFiLL1bSARStructFromFile](#) ([L1B\\_SAR](#) \*p\_ToFill, FILE \*pt\_input, [BASELINE](#) j\_Baseline)
- long int [jFiLL1bSARINStructFromFile](#) ([L1B\\_SARIN](#) \*p\_ToFill, FILE \*pt\_input, [BASELINE](#) j\_Baseline)

### 7.10.1 Detailed Description

Implementation of low-level functions to read Cryosat Level 1b data in LRM, SAR, or SARin modes.

## 7.11 jFiLL1bStructFromFile.h File Reference

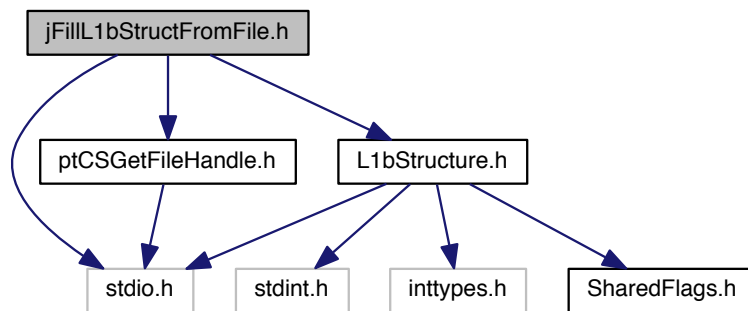
Low-level functions to read Cryosat Level 1b data in LRM, SAR, or SARin modes.

```

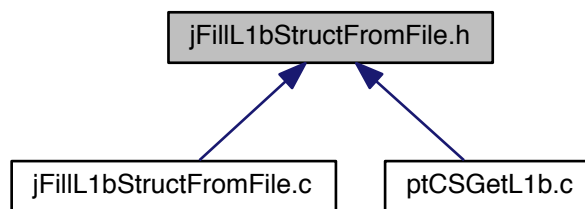
#include <stdio.h>
#include "L1bStructure.h"
#include "ptCSGetFileHandle.h"

```

Include dependency graph for jFillL1bStructFromFile.h:



This graph shows which files directly or indirectly include this file:



## Functions

- long int `jFillL1bLRMStructFromFile` (`L1B_LRM` \*p\_ToFill, FILE \*pt\_input, [BASELINE](#) j\_Baseline)
- long int `jFillL1bSARStructFromFile` (`L1B_SAR` \*p\_ToFill, FILE \*pt\_input, [BASELINE](#) j\_Baseline)
- long int `jFillL1bSARINStructFromFile` (`L1B_SARIN` \*p\_ToFill, FILE \*pt\_input, [BASELINE](#) j\_Baseline)

### 7.11.1 Detailed Description

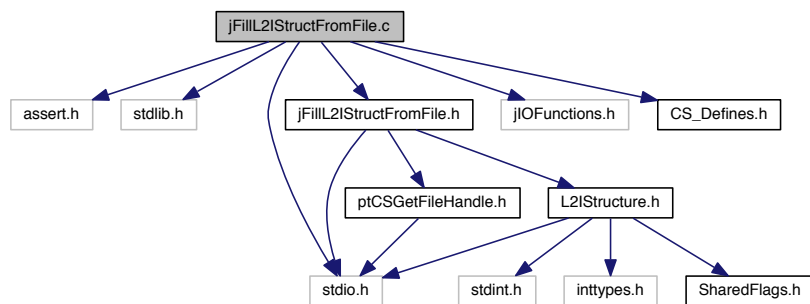
Low-level functions to read Cryosat Level 1b data in LRM, SAR, or SARin modes.

## 7.12 jFillL2StructFromFile.c File Reference

Implementation of low-level functions to read Cryosat Level 2 interim data.



```
#include <assert.h>
#include <stdlib.h>
#include <stdio.h>
#include "jFillL2IStructFromFile.h"
#include "jIOFunctions.h"
#include "CS_Defines.h"
Include dependency graph for jFillL2IStructFromFile.c:
```



## Functions

- long int [jFillL2IStructFromFile](#) ([L2IData](#) \*pt\_Level2, FILE \*pt\_File, [BASELINE](#) j\_Baseline)

### 7.12.1 Detailed Description

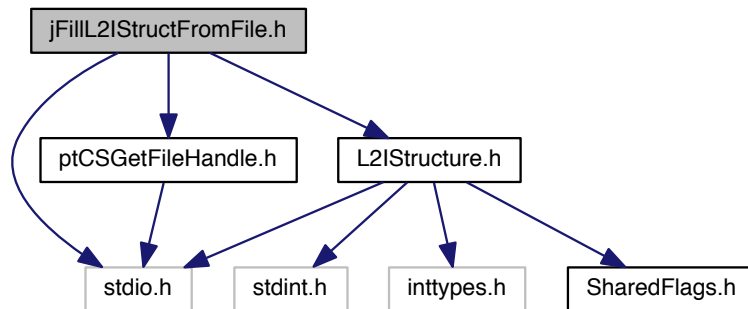
Implementation of low-level functions to read Cryosat Level 2 interim data.

## 7.13 jFillL2IStructFromFile.h File Reference

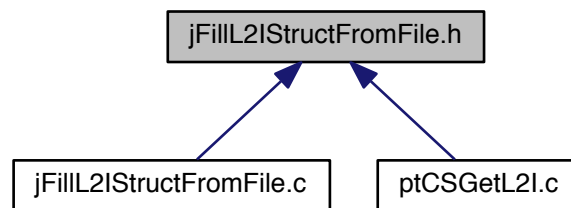
Low-level functions to read Cryosat Level 2 interim data.

```
#include <stdio.h>
#include "L2IStructure.h"
#include "ptCSGetFileHandle.h"
```

Include dependency graph for jFillL2IStructFromFile.h:



This graph shows which files directly or indirectly include this file:



## Functions

- long int [jFillL2IStructFromFile](#) ([L2IData](#) \*pt\_Level2, FILE \*pt\_File, [BASELINE](#) j\_Baseline)

### 7.13.1 Detailed Description

Low-level functions to read Cryosat Level 2 interim data.

## 7.14 jFillL2StructFromFile.c File Reference

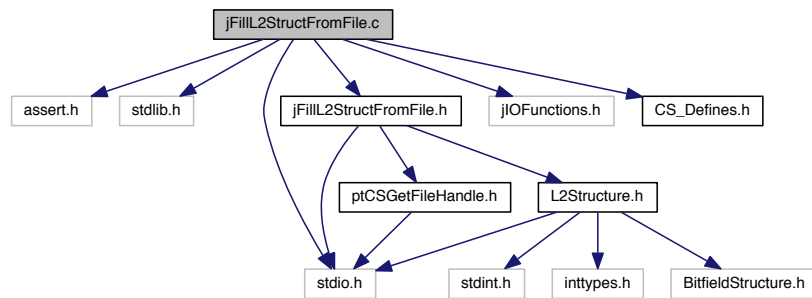
Implementation of low-level functions to read Cryosat Level 2 data.

```

#include <assert.h>
#include <stdlib.h>
#include <stdio.h>
#include "jFillL2StructFromFile.h"
#include "jIOFunctions.h"
#include "CS_Defines.h"

```

Include dependency graph for jFillL2StructFromFile.c:



## Functions

- long int [jFillL2StructFromFile](#) (L2Data \*pt\_Level2, FILE \*pt\_File, [BASELINE](#) j\_Baseline)

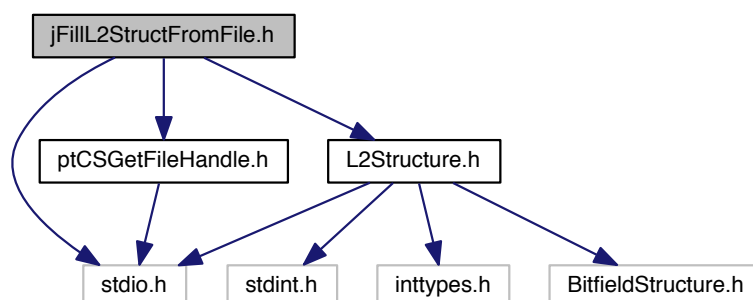
### 7.14.1 Detailed Description

Implementation of low-level functions to read Cryosat Level 2 data.

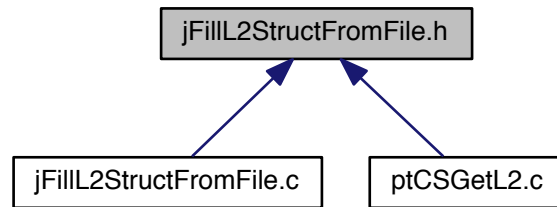
## 7.15 jFillL2StructFromFile.h File Reference

Low-level functions to read Cryosat Level 2 data.

```
#include <stdio.h>
#include "L2Structure.h"
#include "ptCSGetFileHandle.h"
Include dependency graph for jFillL2StructFromFile.h:
```



This graph shows which files directly or indirectly include this file:



## Functions

- long int `jFillL2StructFromFile` (`L2Data` \*pt\_Level2, FILE \*pt\_File, `BASELINE` j\_Baseline)

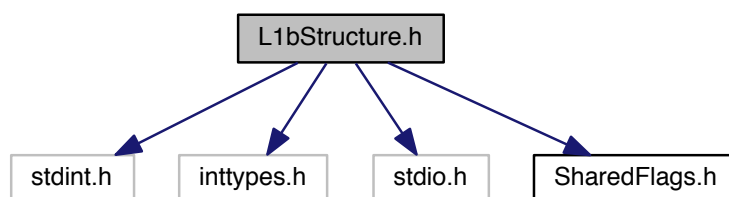
### 7.15.1 Detailed Description

Low-level functions to read Cryosat Level 2 data.

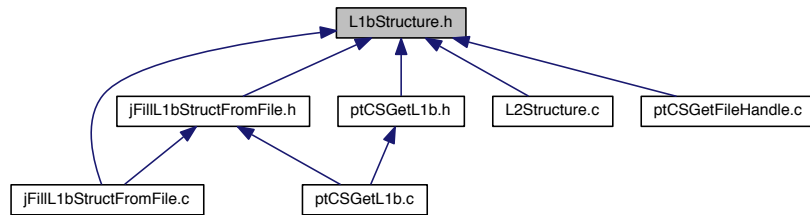
## 7.16 L1bStructure.h File Reference

Definition of the Level 1b structure.

```
#include <stdint.h>
#include <inttypes.h>
#include <stdio.h>
#include "SharedFlags.h"
Include dependency graph for L1bStructure.h:
```



This graph shows which files directly or indirectly include this file:



## Data Structures

- struct [TimeAndOrbitGroup](#)
- struct [Measurements](#)
- struct [Corrections](#)
- struct [AveragedWaveformsLRM](#)
- struct [AveragedWaveformsSAR](#)
- struct [AveragedWaveformsSARIN](#)
- struct [WaveformsLRM](#)
- struct [WaveformsSAR](#)
- struct [WaveformsSARIN](#)
- struct [L1B\\_LRM](#)
- struct [L1B\\_SAR](#)
- struct [L1B\\_SARIN](#)

## Macros

- `#define J_NUM_BB_PARAMETERS 50`
- `#define J_NUM_HRB 20`

## Typedefs

- typedef [t\\_Level1bAvgWfmSARFlagBit](#) [t\\_Level1bAvgWfmSARinFlagBit](#)
- typedef [t\\_Level1bWfmSARFlagBit](#) [t\\_Level1bWfmSARinFlagBit](#)

## Enumerations

- enum [t\\_Level1bAvgWfmLRMFlagBit](#) { [WFM\\_LRM\\_ERROR](#) = 15 }
- enum [t\\_Level1bAvgWfmSARFlagBit](#) { [WFM\\_SAR\\_ERROR](#) = 15, [WFM\\_SAR\\_MISPOINT](#) = 0 }
- enum [t\\_Level1bWfmLRMFlagBit](#) { [WFM\\_LRM\\_TRK\\_2](#) = 2, [WFM\\_LRM\\_TRK\\_1](#) = 1, [WFM\\_LRM\\_TRK\\_0](#) = 0 }
- enum [t\\_Level1bWfmSARFlagBit](#) { [WFM\\_APPROX\\_BEAM\\_STEER](#) = 15, [WFM\\_EXACT\\_BEAM\\_STEER](#) = 14, [WFM\\_DOPPLER\\_COMP](#) = 13, [WFM\\_DOPPLER\\_APPLY](#) = 12, [WFM\\_MULTI\\_LOOK\\_INCOMPLETE](#) = 11, [WFM\\_STEER\\_ERROR](#) = 10, [WFM\\_AA\\_POWER\\_ECHOES](#) = 9, [WFM\\_AUTO\\_STEER](#) = 8 }

## Functions

- void [vDump\\_L1B\\_LRM](#) ([L1B\\_LRM](#) \*pt, FILE \*pf)
- void [vDump\\_L1B\\_SAR](#) ([L1B\\_SAR](#) \*pt, FILE \*pf)
- void [vDump\\_L1B\\_SARIN](#) ([L1B\\_SARIN](#) \*pt, FILE \*pf)

## Variables

- int [J\\_LRM\\_WFM\\_BINS](#)
- int [J\\_SAR\\_WFM\\_BINS](#)
- int [J\\_SARIN\\_WFM\\_BINS](#)
- int [J\\_CS\\_L1B\\_LRM\\_RECORD\\_SIZE](#)
- int [J\\_CS\\_L1B\\_SAR\\_RECORD\\_SIZE](#)
- int [J\\_CS\\_L1B\\_SARIN\\_RECORD\\_SIZE](#)

### 7.16.1 Detailed Description

Definition of the Level 1b structure.

### 7.16.2 Macro Definition Documentation

#### 7.16.2.1 `#define J_NUM_BB_PARAMETERS 50`

Number of beam behaviour parameters.

Definition at line 86 of file L1bStructure.h.

#### 7.16.2.2 `#define J_NUM_HRB 20`

Number of high-rate blocks contained within each record.

Definition at line 89 of file L1bStructure.h.

### 7.16.3 Typedef Documentation

#### 7.16.3.1 `typedef t_Level1bAvgWfmSARFlagBit t_Level1bAvgWfmSARinFlagBit`

Identical for SAR and SARin

Definition at line 552 of file L1bStructure.h.

#### 7.16.3.2 `typedef t_Level1bWfmSARFlagBit t_Level1bWfmSARinFlagBit`

Identical for SAR and SARin

Definition at line 609 of file L1bStructure.h.

### 7.16.4 Enumeration Type Documentation

#### 7.16.4.1 `enum t_Level1bAvgWfmLRMFlagBit`

Values for Averaged Waveform Flag field in LRM mode. See [AveragedWaveformsLRM::ui\\_flags](#)

## Warning

These values are valid in LRM mode ONLY.

## Enumerator

**WFM\_LRM\_ERROR** 1 Hz echo error. Not computed.

Definition at line 533 of file L1bStructure.h.

## 7.16.4.2 enum t\_Level1bAvgWfmSARFlagBit

Values for Averaged Waveform Flag field in SAR and SARin mode. See [AveragedWaveformsSAR::ui\\_flags](#), [AveragedWaveformsSARIN::ui\\_flags](#)

## Enumerator

**WFM\_SAR\_ERROR** 1 Hz echo error. Not computed.

**WFM\_SAR\_MISPOINT** Bad angle.

Definition at line 543 of file L1bStructure.h.

## 7.16.4.3 enum t\_Level1bWfmLRMFlagBit

Values for Waveform Flag field in LRM mode. See [WaveformsLRM::ui\\_flags](#)

The three-bit integer assembled from the WFM\_LRM\_TRK bits has the following values:

Value	Meaning
0	no errors
1	loss of echo
2	run time error
3	echo saturation error
7	unknown error

## Enumerator

**WFM\_LRM\_TRK\_2** Bit 2 for WFM\_LRM\_TRK integer

**WFM\_LRM\_TRK\_1** Bit 1 for WFM\_LRM\_TRK integer

**WFM\_LRM\_TRK\_0** Bit 0 for WFM\_LRM\_TRK integer

Definition at line 568 of file L1bStructure.h.

## 7.16.4.4 enum t\_Level1bWfmSARFlagBit

Values for Waveform Flag field in SAR and SARin mode. See [WaveformsSAR::ui\\_flags](#), [WaveformsSARIN::ui\\_flags](#)

## Enumerator

**WFM\_APPROX\_BEAM\_STEER** Set if approximate beam steering was used.

**WFM\_EXACT\_BEAM\_STEER** Set if exact beam steering was used.

**WFM\_DOPPLER\_COMP** Set if doppler weighting was computed.

**WFM\_DOPPLER\_APPLY** Set if doppler weighting was applied before stacking.

**WFM\_MULTI\_LOOK\_INCOMPLETE** Set if multi-look was incomplete.

**WFM\_STEER\_ERROR** Set if a beam angle steering error occurred.

**WFM\_AA\_POWER\_ECHOES** Set if power echos are anti-aliased.

**WFM\_AUTO\_STEER** Set if auto beam steering was used.

Definition at line 579 of file L1bStructure.h.

### 7.16.5 Function Documentation

#### 7.16.5.1 void vDump\_L1B\_LRM ( L1B\_LRM \* *pt*, FILE \* *pf* )

Create an ASCII dump of a L1B LRM product.

##### Parameters

<i>pt</i>	Pointer to the product to dump.
<i>pf</i>	File pointer to dump to

##### Examples:

[examples/cs\\_l1b\\_io\\_example.c](#).

#### 7.16.5.2 void vDump\_L1B\_SAR ( L1B\_SAR \* *pt*, FILE \* *pf* )

Create an ASCII dump of a L1B SAR product.

##### Parameters

<i>pt</i>	Pointer to the product to dump.
<i>pf</i>	File pointer to dump to

##### Examples:

[examples/cs\\_l1b\\_io\\_example.c](#).

#### 7.16.5.3 void vDump\_L1B\_SARIN ( L1B\_SARIN \* *pt*, FILE \* *pf* )

Create an ASCII dump of a L1B SARIN product.

##### Parameters

<i>pt</i>	Pointer to the product to dump.
<i>pf</i>	File pointer to dump to

##### Examples:

[examples/cs\\_l1b\\_io\\_example.c](#).

### 7.16.6 Variable Documentation

#### 7.16.6.1 int J\_LRM\_WFM\_BINS

Number of bins in the range window for LRM mode.

#### 7.16.6.2 int J\_SAR\_WFM\_BINS

Number of bins in the range window for SAR mode.

#### 7.16.6.3 int J\_SARIN\_WFM\_BINS

Number of bins in the range window for SARin mode.



## 7.16.6.4 int J\_CS\_L1B\_LRM\_RECORD\_SIZE

Size in bytes of the Level 1b LRM mode record on disk.

## 7.16.6.5 int J\_CS\_L1B\_SAR\_RECORD\_SIZE

Size in bytes of the Level 1b SAR mode record on disk.

## 7.16.6.6 int J\_CS\_L1B\_SARIN\_RECORD\_SIZE

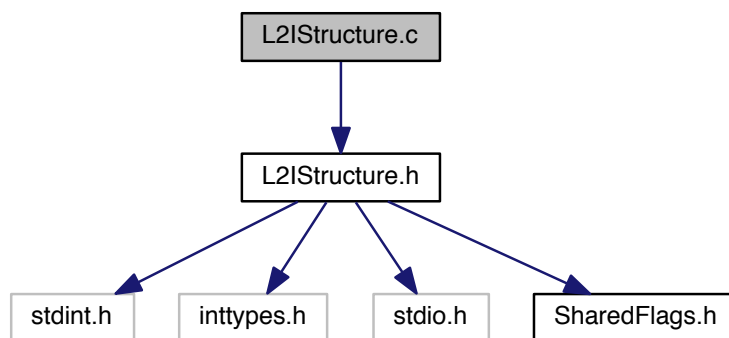
Size in bytes of the Level 1b SARin mode record on disk.

## 7.17 L2IStructure.c File Reference

Convenience functions related to the Level 2i structure.

```
#include "L2IStructure.h"
```

Include dependency graph for L2IStructure.c:



### Functions

- void [vDump\\_L2IData](#) ([L2IData](#) \*pt, FILE \*pf)

#### 7.17.1 Detailed Description

Convenience functions related to the Level 2i structure.

#### 7.17.2 Function Documentation

##### 7.17.2.1 void vDump\_L2IData ( L2IData \* pt, FILE \* pf )

Dump an L2i product to a file in ASCII format.

## Parameters

<i>pt</i>	Pointer to the L2i product to dump.
<i>pf</i>	Pointer to the file to dump to.

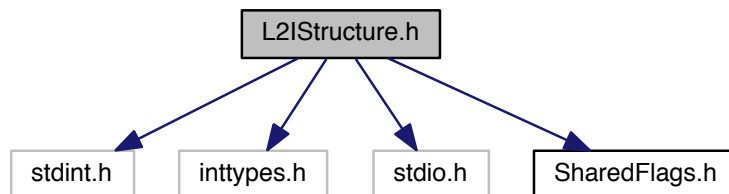
Definition at line 11 of file L2iStructure.c.

## 7.18 L2iStructure.h File Reference

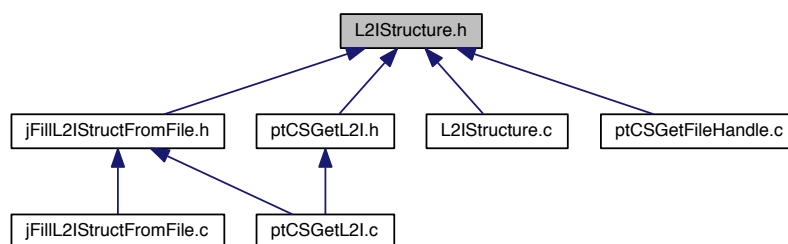
Definition of the Level 2i structure.

```
#include <stdint.h>
#include <inttypes.h>
#include <stdio.h>
#include "SharedFlags.h"
```

Include dependency graph for L2iStructure.h:



This graph shows which files directly or indirectly include this file:



## Data Structures

- struct [L2iData](#)

## Macros

- `#define J_CS_L2I_STRUCT_SIZE ( sizeof( L2iData ) )`

## Enumerations

- enum `t_InterpolationErrorFlagBit` { `INTERP_SAR_UNREL` = 0, `INTERP_SAR_ONE_SIDED` = 1, `INTERP_SAR_NO_POINTS` = 2 }
- enum `t_MeasurementMode` { `MODE_LRM` = 1, `MODE_SAR` = 2, `MODE_SARIN` = 3 }
- enum `t_MeasurementQualityFlagBit` {  
`MQ_HEIGHT_1_ERR` = 31, `MQ_HEIGHT_2_ERR` = 30, `MQ_HEIGHT_3_ERR` = 29, `MQ_SIGMA0_1_ERR` = 28,  
`MQ_SIGMA0_2_ERR` = 27, `MQ_SIGMA0_3_ERR` = 26, `MQ_PK_ERR` = 25, `MQ_ECHO_SHAPE_ERR` = 24,  
`MQ_XTRACK_ERR` = 23, `MQ_COHERENCE_ERR` = 22, `MQ_ARITHMETIC_ERR` = 21, `MQ_ALTWIND_ERR` = 20,  
`MQ_SWH_ERR` = 19 }
- enum `t_RetrackerFlagBit` {  
`RTRK_FAIL` = 31, `RTRK_LO_PWR` = 30, `RTRK_LO_PK` = 29, `RTRK_HI_PK` = 28,  
`RTRK_HI_NOISE` = 27, `RTRK_LO_VAR` = 26, `RTRK_LE_BAD` = 25, `RTRK_OUT_OF_RANGE` = 24,  
`RTRK_BAD_BB` = 23, `RTRK_SPARE` = 22, `RTRK_SPARE1` = 21, `RTRK_BAD_U10` = 20,  
`RTRK_SPARE2` = 19, `RTRK_SARIN_INTERPOL` = 17, `RTRK_SARIN_LO_COHERENCE` = 16, `RTRK_FIT_FAILED` = 15,  
`RTRK_FDM_OCOG_FAIL` = 14, `RTRK_POOR_FIT` = 13, `RTRK_POOR_PHASE_FIT` = 12, `RTRK_1_FAIL` = 11,  
`RTRK_2_FAIL` = 10, `RTRK_3_FAIL` = 9 }
- enum `t_HeightStatusFlagBit` {  
`HGT_INTERNAL` = 31, `HGT_RAD_DOPPLER` = 30, `HGT_DRY_TROPO` = 29, `HGT_WET_TROPO` = 28,  
`HGT_INV_BARO` = 27, `HGT_DAC` = 26, `HGT_IONO_GIM` = 25, `HGT_IONO_MODEL` = 24,  
`HGT_OT` = 23, `HGT_LPEOT` = 22, `HGT_OLT` = 21, `HGT_SET` = 20,  
`HGT_GPT` = 19, `HGT_SLOPE_DOPPLER` = 18, `HGT_WINDOW_OFFSET` = 17, `HGT_SAR_RETRK` = 16,  
`HGT_SARIN_RETRK` = 15, `HGT_LRM_RETRK` = 14, `HGT_LRM_OCEANBIAS` = 13, `HGT_LRM_ICEBIAS` = 12,  
`HGT_SAR_OCEANBIAS` = 11, `HGT_SAR_ICEBIAS` = 10, `HGT_SARIN_OCEANBIAS` = 9, `HGT_SARIN_ICEBIAS` = 8,  
`LRM_SLOPE_DATA_VALID` = 7, `HGT_SARIN_BAD_BASELINE` = 6, `HGT_SARIN_OUT_OF_RANGE` = 5,  
`HGT_SARIN_BAD_VELOCITY` = 4,  
`HGT_SSB` = 3, `HGT_FAIL` = 0 }
- enum `t_FreeboardStatusFlagBit` { `FREB_SAR_AVAIL` = 0, `FREB_SAR_UNREL` = 1, `FREB_SAR_IN_NORTH_BOUND` = 2, `FREB_SAR_IN_SOUTH_BOUND` = 3 }
- enum `t_DiscriminatorResult` {  
`CL2_LRM_IS_UNDEFINED` = 1, `CL2_LRM_IS_OCEAN` = 2, `CL2_LRM_IS_LAND_ICE_PLATEAU` = 3, `CL2_SARIN_IS_UNDEFINED` = 101,  
`CL2_SARIN_IS_ALTIMETER_MODE` = 102, `CL2_SAR_IS_UNDEFINED` = 201, `CL2_SAR_IS_OCEAN` = 202, `CL2_SAR_IS_SEA_ICE` = 203,  
`CL2_SAR_IS_LEADS` = 204 }
- enum `t_DiscriminatorStatusFlagBit` {  
`DISC_FAIL` = 31, `DISC_TBD` = 30, `DISC_SARIN_LO_VARIANCE` = 21, `DISC_SARIN_BAD_LE` = 20,  
`DISC_SARIN_HI_NOISE` = 19, `DISC_SARIN_LOW_PEAKINESS` = 18, `DISC_SARIN_LOW_POWER` = 17,  
`DISC_SARIN_HI_PEAKINESS` = 16,  
`DISC_SAR_HI_PK` = 11, `DISC_SAR_LO_PK` = 10, `DISC_SAR_LO_POWER` = 9, `DISC_SAR_BAD_BB` = 8,  
`DISC_SAR_ICECONC_UNAV` = 7, `DISC_SAR_ICECONC_UNREAL` = 6, `DISC_SAR_BAD_NOISE` = 5, `DISC_SAR_BAD_WIDTH` = 4,  
`DISC_SAR_NO_MATCH` = 28, `DISC_SAR_MULTIPLE_MATCH` = 29 }
- enum `t_Level2AmbiguityErrorFlagBit` {  
`AMB_FAIL` = 31, `AMB_SARIN_NO_DEM` = 21, `AMB_SARIN_DIFF_ELEV` = 20, `AMB_SARIN_NO_RTRK` = 19,  
`AMB_SARIN_BAD_MATH` = 18 }

## Functions

- void `vDump_L2IData` (`L2IData` \*pt, FILE \*pf)

### 7.18.1 Detailed Description

Definition of the Level 2i structure.

### 7.18.2 Macro Definition Documentation

#### 7.18.2.1 `#define J_CS_L2I_STRUCT_SIZE ( sizeof( L2IData ) )`

Size of an L2i product record.

Definition at line 1078 of file L2IStructure.h.

### 7.18.3 Enumeration Type Documentation

#### 7.18.3.1 `enum t_InterpolationErrorFlagBit`

Bit IDs for Interpolation error flag. See [L2IData::ui\\_interp\\_error\\_F](#)

Enumerator

**INTERP\_SAR\_UNREL** Interpolation result may be unreliable.

**INTERP\_SAR\_ONE\_SIDED** All of the interpolation points were taken from one side of the centre point.

**INTERP\_SAR\_NO\_POINTS** No ocean points were within the interpolation radius.

Definition at line 880 of file L2IStructure.h.

#### 7.18.3.2 `enum t_MeasurementMode`

Values for Measurement mode. See [L2IData::uj\\_Meas\\_Mode](#)

Vaue	Mode
1	LRM
2	SAR
3	SARin

Enumerator

**MODE\_LRM** Measurement mode is LRM.

**MODE\_SAR** Measurement mode is SAR.

**MODE\_SARIN** Measurement mode is SARin

Definition at line 903 of file L2IStructure.h.

#### 7.18.3.3 `enum t_MeasurementQualityFlagBit`

Bit IDs for Quality flags. See [L2IData::uj\\_Quality\\_F](#)

Enumerator

**MQ\_HEIGHT\_1\_ERR** Height Error.

**MQ\_HEIGHT\_2\_ERR** Height Error.

**MQ\_HEIGHT\_3\_ERR** Height Error.

**MQ\_SIGMA0\_1\_ERR** Sigma 0 Error.

**MQ\_SIGMA0\_2\_ERR** Sigma 0 Error.

***MQ\_SIGMA0\_3\_ERR*** Sigma 0 Error.  
***MQ\_PK\_ERR*** Peakiness Error.  
***MQ\_ECHO\_SHAPE\_ERR*** Echo shape error.  
***MQ\_XTRACK\_ERR*** X-Track angle error.  
***MQ\_COHERENCE\_ERR*** Coherence error.  
***MQ\_ARITHMETIC\_ERR*** IEEE Maths flag set.  
***MQ\_ALTWIND\_ERR*** Error calculating altimeter windspeed  
***MQ\_SWH\_ERR*** Error calculating SWH

Definition at line 910 of file L2IStructure.h.

#### 7.18.3.4 enum t\_RetrackerFlagBit

Bit IDs for Retracker flags. See [L2IData::uj\\_Retracker\\_F](#)

Enumerator

***RTRK\_FAIL*** Overall retracker warning. Not used.  
***RTRK\_LO\_PWR*** Zero or low power waveform.  
***RTRK\_LO\_PK*** Low peakiness.  
***RTRK\_HI\_PK*** High peakiness.  
***RTRK\_HI\_NOISE*** High noise.  
***RTRK\_LO\_VAR*** Low variance.  
***RTRK\_LE\_BAD*** Bad leading edge.  
***RTRK\_OUT\_OF\_RNGE*** Retracker position out of range.  
***RTRK\_BAD\_BB*** Abnormal Beam Behaviour Params.  
***RTRK\_SPARE*** Spare.  
***RTRK\_SPARE1*** Spare.  
***RTRK\_BAD\_U10*** Reserved U10.  
***RTRK\_SPARE2*** Spare 2.  
***RTRK\_SARIN\_INTERPOL*** Retracker interpolation failure.  
***RTRK\_SARIN\_LO\_COHERENCE*** Low coherence.  
***RTRK\_FIT\_FAILED*** Retracker fit failure.  
***RTRK\_FDM\_OCOG\_FAIL*** OCOG retracker failed  
***RTRK\_POOR\_FIT*** Retracker fit was of poor quality.  
***RTRK\_POOR\_PHASE\_FIT*** SARin phase fit was of poor quality.  
***RTRK\_1\_FAIL*** Retracker 1 failed.  
***RTRK\_2\_FAIL*** Retracker 2 failed.  
***RTRK\_3\_FAIL*** Retracker 3 failed.

Definition at line 928 of file L2IStructure.h.

#### 7.18.3.5 enum t\_HeightStatusFlagBit

Bit IDs for Height calculation details. See [L2IData::uj\\_Ht\\_status](#)

Enumerator

***HGT\_INTERNAL*** Internal calibration.

**HGT\_RAD\_DOPPLER** Radial doppler.  
**HGT\_DRY\_TROPO** Dry tropospheric.  
**HGT\_WET\_TROPO** Wet tropospheric.  
**HGT\_INV\_BARO** Inverse barometric.  
**HGT\_DAC** Dynamic Atmospheric Correction.  
**HGT\_IONO\_GIM** GIM ionospheric.  
**HGT\_IONO\_MODEL** Model ionospheric.  
**HGT\_OT** Ocean Tide.  
**HGT\_LPEOT** LPEOT.  
**HGT\_OLT** Ocean loading time.  
**HGT\_SET** Solid Earth tide.  
**HGT\_GPT** Geocentric Polar tide.  
**HGT\_SLOPE\_DOPPLER** Slope Doppler.  
**HGT\_WINDOW\_OFFSET** Mode specific window offset.  
**HGT\_SAR\_RETRK** SAR retracker offset.  
**HGT\_SARIN\_RETRK** SARIN retracker offset.  
**HGT\_LRM\_RETRK** LRM retracker offset.  
**HGT\_LRM\_OCEANBIAS** LRM ocean bias applied.  
**HGT\_LRM\_ICEBIAS** LRM ice bias applied.  
**HGT\_SAR\_OCEANBIAS** SAR ocean bias applied.  
**HGT\_SAR\_ICEBIAS** SAR ice bias applied.  
**HGT\_SARIN\_OCEANBIAS** SARin ocean bias applied.  
**HGT\_SARIN\_ICEBIAS** SARin ice bias applied.  
**LRM\_SLOPE\_DATA\_VALID** LRM slope model data valid.  
**HGT\_SARIN\_BAD\_BASELINE** SARin Bad Baseline downward.  
**HGT\_SARIN\_OUT\_OF\_RANGE** SARin out of range.  
**HGT\_SARIN\_BAD\_VELOCITY** SARin Bad Velocity.  
**HGT\_SSB** SSB Applied  
**HGT\_FAIL** Generic failure flag. Not used.

Definition at line 955 of file L2IStructure.h.

#### 7.18.3.6 enum `t_FreeboardStatusFlagBit`

Bit IDs for SAR freeboard status flag. See [L2IData::uj\\_Freeb\\_status](#)

Enumerator

**FREB\_SAR\_AVAIL** Freeboard estimation unavailable.  
**FREB\_SAR\_UNREL** Freeboard estimation unreliable.  
**FREB\_SAR\_IN\_NORTH\_BOUND** Freeboard estimation is in northern geographical boundary.  
**FREB\_SAR\_IN\_SOUTH\_BOUND** Freeboard estimation is in southern geographical boundary.

Definition at line 990 of file L2IStructure.h.

7.18.3.7 enum `t_DiscriminatorResult`

Values for Discriminator result. See [L2IData::j\\_Discriminator](#)

## Enumerator

- `CL2_LRM_IS_UNDEFINED`** Result of LRM discrimination is undefined.
- `CL2_LRM_IS_OCEAN`** Result of LRM discrimination is ocean.
- `CL2_LRM_IS_LAND_ICE_PLATEAU`** Result of LRM discrimination is land ice plateau.
- `CL2_SARIN_IS_UNDEFINED`** Result of SARIN discrimination is undefined.
- `CL2_SARIN_IS_ALTIMETER_MODE`** Result of SARIN discrimination is altimeter mode.
- `CL2_SAR_IS_UNDEFINED`** Result of SAR discrimination is undefined.
- `CL2_SAR_IS_OCEAN`** Result of SAR discrimination is ocean.
- `CL2_SAR_IS_SEA_ICE`** Result of SAR discrimination is sea ice.
- `CL2_SAR_IS_LEADS`** Result of SAR discrimination is leads.

Definition at line 1004 of file `L2IStructure.h`.

7.18.3.8 enum `t_DiscriminatorStatusFlagBit`

Bit IDs for Discriminator flags. See [L2IData::uj\\_Discrim\\_F](#)

## Enumerator

- `DISC_FAIL`** Overall discriminator failure.
- `DISC_TBD`** LRM discriminator flag TBD.
- `DISC_SARIN_LO_VARIANCE`** SARin Low Variance.
- `DISC_SARIN_BAD_LE`** SARin Bad Leading Edge.
- `DISC_SARIN_HI_NOISE`** SARin High Noise.
- `DISC_SARIN_LOW_PEAKINESS`** SARin Low Peakiness.
- `DISC_SARIN_LOW_POWER`** SARin Low Power.
- `DISC_SARIN_HI_PEAKINESS`** SARin High Peakiness.
- `DISC_SAR_HI_PK`** SAR very high peakiness.
- `DISC_SAR_LO_PK`** SAR very low peakiness.
- `DISC_SAR_LO_POWER`** SAR low or zero power.
- `DISC_SAR_BAD_BB`** SAR abnormal beam behaviour.
- `DISC_SAR_ICECONC_UNAV`** SAR ice conc unavailable.
- `DISC_SAR_ICECONC_UNREAL`** SAR ice conc unreliable.
- `DISC_SAR_BAD_NOISE`** SAR signal to noise too low.
- `DISC_SAR_BAD_WIDTH`** SAR waveform too wide.
- `DISC_SAR_NO_MATCH`** SAR none of the available classifications match for this record.
- `DISC_SAR_MULTIPLE_MATCH`** More than one of the available classifications match for this record.

Definition at line 1036 of file `L2IStructure.h`.

### 7.18.3.9 enum t\_Level2AmbiguityErrorFlagBit

Bit IDs for Ambiguity indicator.

#### Enumerator

**AMB\_FAIL** Ambiguity test failed. Result is possibly affected by phase wrapping.

**AMB\_SARIN\_NO\_DEM** Test failed because there is no DEM for this location.

**AMB\_SARIN\_DIFF\_ELEV** Test failed because the elevation difference was beyond threshold.

**AMB\_SARIN\_NO\_RTRK** Test failed because retracking failed.

**AMB\_SARIN\_BAD\_MATH** Test failed due to a mathematical error.

Definition at line 1066 of file L2IStructure.h.

## 7.18.4 Function Documentation

### 7.18.4.1 void vDump\_L2IData ( L2IData \* *pt*, FILE \* *pf* )

Dump an L2i product to a file in ASCII format.

#### Parameters

<i>pt</i>	Pointer to the L2i product to dump.
<i>pf</i>	Pointer to the file to dump to.

#### Examples:

[examples/cs\\_l2i\\_io\\_example.c](#).

Definition at line 11 of file L2IStructure.c.

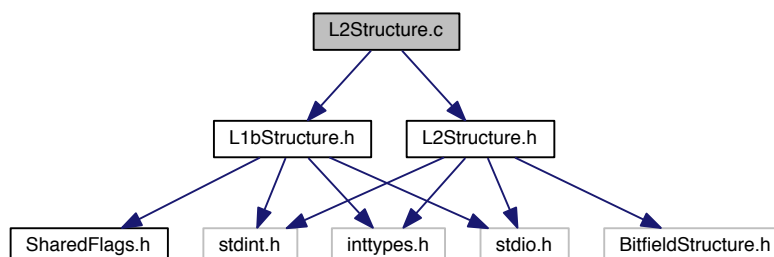
## 7.19 L2Structure.c File Reference

Convenience functions for the L2 structure.

```
#include "L2Structure.h"
```

```
#include "L1bStructure.h"
```

Include dependency graph for L2Structure.c:





## Functions

- static void `vDump_L2FinalC` (`L2FinalC` \**pt*, FILE \**pf*)
- static void `vDump_L2FinalM` (`L2FinalM` \**pt*, FILE \**pf*)
- void `vDump_L2Data` (`L2Data` \**pt*, FILE \**pf*)

### 7.19.1 Detailed Description

Convenience functions for the L2 structure.

### 7.19.2 Function Documentation

**7.19.2.1** void `vDump_L2FinalC` ( `L2FinalC` \* *pt*, FILE \* *pf* ) [static]

Create an ASCII dump of an L2 corrections struture to a file.

Parameters

<i>pt</i>	Pointer to the structure to dump.
<i>pf</i>	Pointer to the file to dump to.

Definition at line 51 of file L2Structure.c.

Here is the caller graph for this function:



**7.19.2.2** void `vDump_L2FinalM` ( `L2FinalM` \* *pt*, FILE \* *pf* ) [static]

Create an ASCII dump of an L2 measurements structure to a file.

Parameters

<i>pt</i>	Pointer to the structure to dump.
<i>pf</i>	Pointer to the file to dump to.

Definition at line 83 of file L2Structure.c.

Here is the caller graph for this function:



### 7.19.2.3 void vDump\_L2Data ( L2Data \* *pt*, FILE \* *pf* )

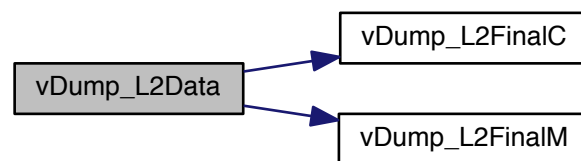
Create an ASCII dump of an L2 product to a file.

Parameters

<i>pt</i>	Pointer to the product to dump.
<i>pf</i>	Pointer to the file to dump to.

Definition at line 21 of file L2Structure.c.

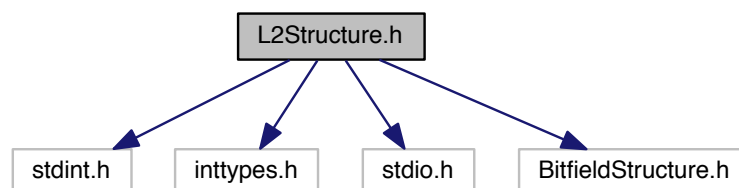
Here is the call graph for this function:



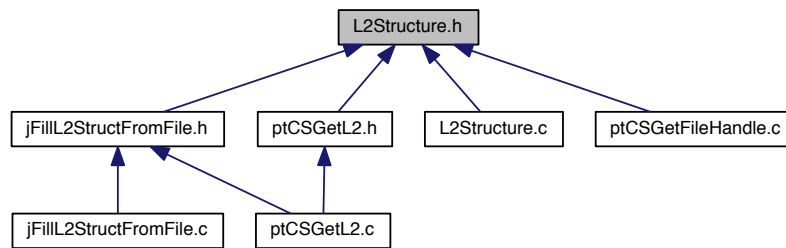
## 7.20 L2Structure.h File Reference

Structure and constant definitions for the Cryosat Level 2 format.

```
#include <stdint.h>
#include <inttypes.h>
#include <stdio.h>
#include "BitfieldStructure.h"
Include dependency graph for L2Structure.h:
```



This graph shows which files directly or indirectly include this file:



## Data Structures

- struct [L2FinalC](#)
- struct [L2FinalM](#)
- struct [L2Data](#)

## Macros

- `#define J_L2_STRUCT_SIZE ( sizeof( L2Data ) )`
- `#define J_L2_RECORD_SIZE 980`

## Enumerations

- enum [t\\_Level2QualityStatusFlagBit](#) {  
[L2Q\\_BLOCK\\_DEGRADED](#) = 31, [L2Q\\_ORBIT\\_ERROR](#) = 30, [L2Q\\_ORBIT\\_DISCONTINUITY](#) = 29, [L2Q\\_HEIGHT\\_1\\_ERROR](#) = 28,  
[L2Q\\_HEIGHT\\_2\\_ERROR](#) = 27, [L2Q\\_HEIGHT\\_3\\_ERROR](#) = 26, [L2Q\\_BACKSCATTER\\_1\\_ERROR](#) = 25,  
[L2Q\\_BACKSCATTER\\_2\\_ERROR](#) = 24,  
[L2Q\\_BACKSCATTER\\_3\\_ERROR](#) = 23, [L2Q\\_SSHA\\_INTERP\\_ERROR](#) = 22, [L2Q\\_PEAKINESS\\_ERROR](#) = 21, [L2Q\\_FREEBOARD\\_ERROR](#) = 20,  
[L2Q\\_SAR\\_DISCRIM\\_OCEAN](#) = 19, [L2Q\\_SAR\\_DISCRIM\\_LEAD](#) = 18, [L2Q\\_SAR\\_DISCRIM\\_ICE](#) = 17, [L2Q\\_SAR\\_DISCRIM\\_UNKNOWN](#) = 16,  
[L2Q\\_SARIN\\_XTRACK\\_ERROR](#) = 15, [L2Q\\_SARIN\\_RX1\\_ERROR](#) = 14, [L2Q\\_SARIN\\_RX2\\_ERROR](#) = 13,  
[L2Q\\_SARIN\\_SIDE\\_B](#) = 12,  
[L2Q\\_SURFACE\\_MOD](#) = 11, [L2Q\\_MISPOINTING\\_ERROR](#) = 10, [L2Q\\_DELTA\\_TIME\\_ERROR](#) = 9, [L2Q\\_LRM\\_SLOPE\\_INVALID](#) = 8,  
[L2Q\\_SARIN\\_BASELINE\\_ERROR](#) = 7, [L2Q\\_SARIN\\_RANGE\\_ERROR](#) = 6, [L2Q\\_SARIN\\_VELOCITY\\_ERROR](#) = 5, [L2Q\\_CALIBRATION\\_WARNING](#) = 4 }
- enum [t\\_Level2CorrectionsStatusFlagBit](#) {  
[L2C\\_DRY\\_TROPO](#) = 31, [L2C\\_WET\\_TROPO](#) = 30, [L2C\\_INV\\_BARO](#) = 29, [L2C\\_DAC](#) = 28,  
[L2C\\_IONO](#) = 27, [L2C\\_SSB](#) = 26, [L2C\\_OCEAN\\_TIDE](#) = 25, [L2C\\_LPEOT](#) = 24,  
[L2C\\_OLT](#) = 23, [L2C\\_SET](#) = 22, [L2C\\_GPT](#) = 21, [L2C\\_SURFACE\\_TYPE](#) = 20,  
[L2C\\_MSS\\_GEOID](#) = 19, [L2C\\_ODLE\\_MODEL](#) = 18, [L2C\\_ICE\\_CONC](#) = 17, [L2C\\_SNOW\\_DEPTH](#) = 16,  
[L2C\\_SNOW\\_DENSITY](#) = 15, [L2C\\_SWH](#) = 14, [L2C\\_WINDSPEED](#) = 13 }
- enum [t\\_Level2CorrectionsAppliedFlagBit](#) {  
[HGT\\_INTERNAL\\_PDS](#) = 31, [HGT\\_RAD\\_DOPPLER\\_PDS](#) = 30, [HGT\\_DRY\\_TROPO\\_PDS](#) = 29, [HGT\\_WET\\_TROPO\\_PDS](#) = 28,  
[HGT\\_INV\\_BARO\\_PDS](#) = 27, [HGT\\_DAC\\_PDS](#) = 26, [HGT\\_IONO\\_GIM\\_PDS](#) = 25, [HGT\\_IONO\\_MODEL\\_PDS](#) = 24 }

```

DS = 24,
HGT_OT_PDS = 23, HGT_LPEOT_PDS = 22, HGT_OLT_PDS = 21, HGT_SET_PDS = 20,
HGT_GPT_PDS = 19, HGT_SLOPE_DOPPLER_PDS = 18, HGT_WINDOW_OFFSET_PDS = 17, HGT_↵
SAR_RETRK_PDS = 16,
HGT_SARIN_RETRK_PDS = 15, HGT_LRM_RETRK_PDS = 14, HGT_LRM_OCEANBIAS_PDS = 13, H↵
GT_LRM_ICEBIAS_PDS = 12,
HGT_SAR_OCEANBIAS_PDS = 11, HGT_SAR_ICEBIAS_PDS = 10, HGT_SARIN_OCEANBIAS_PDS = 9,
HGT_SARIN_ICEBIAS_PDS = 8,
LRM_SLOPE_DATA_VALID_PDS = 7, HGT_SARIN_BAD_BASELINE_PDS = 6, HGT_SARIN_OUT_OF↵
_RANGE_PDS = 5, HGT_SARIN_BAD_VELOCITY_PDS = 4,
HGT_SSB_PDS = 3 }

```

## Functions

- void [vDump\\_L2Data](#) ([L2Data](#) \*pt, FILE \*pf)

### 7.20.1 Detailed Description

Structure and constant definitions for the Cryosat Level 2 format.

### 7.20.2 Macro Definition Documentation

#### 7.20.2.1 `#define J_L2_STRUCT_SIZE ( sizeof( L2Data ) )`

Number of bytes IN MEMORY for the Level 2 Struture.

Definition at line 383 of file L2Structure.h.

#### 7.20.2.2 `#define J_L2_RECORD_SIZE 980`

Number of bytes ON DISK for the Level 2 structure.

Definition at line 386 of file L2Structure.h.

### 7.20.3 Enumeration Type Documentation

#### 7.20.3.1 `enum t_Level2QualityStatusFlagBit`

Bit definitions for the L2 Quality Status Flag, see [L2FinalM::uj\\_Quality\\_F](#)

#### Enumerator

**L2Q\_BLOCK\_DEGRADED** Set if block is degraded.

**L2Q\_ORBIT\_ERROR** Set if an orbit-related error occurred.

**L2Q\_ORBIT\_DISCONTINUITY** Set if the orbit was discontinuous.

**L2Q\_HEIGHT\_1\_ERROR** Set if any error occurred during the calculation of the height value for retracker 1.

**L2Q\_HEIGHT\_2\_ERROR** Set if any error occurred during the calculation of the height value for retracker 2.

**L2Q\_HEIGHT\_3\_ERROR** Set if any error occurred during the calculation of the height value for retracker 3.

**L2Q\_BACKSCATTER\_1\_ERROR** Set if the backscctter value from retracker 1 is invalid.

**L2Q\_BACKSCATTER\_2\_ERROR** Set if the backscctter value from retracker 2 is invalid.

**L2Q\_BACKSCATTER\_3\_ERROR** Set if the backscctter value from retracker 3 is invalid.

**L2Q\_SSHA\_INTERP\_ERROR** Set if an error occurred during interpolation of the sea-surface height anomaly.

**L2Q\_PEAKINESS\_ERROR** Set if the peakiness value is invalid.

**L2Q\_FREEBOARD\_ERROR** Set if the freeboard value is invalid.

**L2Q\_SAR\_DISCRIM\_OCEAN** Set if the SAR discriminator determined the surface to be ocean.

**L2Q\_SAR\_DISCRIM\_LEAD** Set if the SAR discriminator determined the surface to be a lead.

**L2Q\_SAR\_DISCRIM\_ICE** Set if the SAR discriminato determined the surface to be ice.

**L2Q\_SAR\_DISCRIM\_UNKNOWN** Set if the SAR discriminator could not determine the surface type.

**L2Q\_SARIN\_XTRACK\_ERROR** Set if the SARin chain could not determine the cross-track angle.

**L2Q\_SARIN\_RX1\_ERROR** Set if the Rx1 echo was bad.

**L2Q\_SARIN\_RX2\_ERROR** Set if the Rx2 echo was bad.

**L2Q\_SARIN\_SIDE\_B** Set if SIRAL is on the redundant side.

**L2Q\_SURFACE\_MOD** Set if no DEM/SLOPE model for the current location.

**L2Q\_MISPOINTING\_ERROR** Set if error during mispointing derivation.

**L2Q\_DELTA\_TIME\_ERROR** Set if error during delta time calculation.

**L2Q\_LRM\_SLOPE\_INVALID** Set if the LRM slope correction was invalid.

**L2Q\_SARIN\_BASELINE\_ERROR** Set if the baseline vector was invalid.

**L2Q\_SARIN\_RANGE\_ERROR** Set if the computed range was invalid.

**L2Q\_SARIN\_VELOCITY\_ERROR** Set if the velocity vector was invalid.

**L2Q\_CALIBRATION\_WARNING** Set if any calibration-related error condition was set.

Definition at line 389 of file L2Structure.h.

#### 7.20.3.2 enum t\_Level2CorrectionsStatusFlagBit

Bit definitions for the L2 [Corrections](#) Status flag, see [L2FinalC::uj\\_C\\_status](#)

##### Enumerator

**L2C\_DRY\_TROPO** Set if dry tropospheric correction error flag set at L1b or if correction not available at L1b.

**L2C\_WET\_TROPO** Set if wet tropospheric correction error flag set at L1b or if correction not available at L1b.

**L2C\_INV\_BARO** Set if inverse barometric correction error flag set at L1b or if correction not available at L1b.

**L2C\_DAC** Set if DAC correction error flag set at L1b or if correction not available at L1b.

**L2C\_IONO** Set if ionospheric correction error flag set at L1b or if correction not available at L1b.

**L2C\_SSB** Set if sea state bias value is invalid.

**L2C\_OCEAN\_TIDE** Set if ocean tide correction error flag set at L1b or if correction not available at L1b.

**L2C\_LPEOT** Set if long period equilibrium ocean tide correction error flag set at L1b or if correction not available at L1b.

**L2C\_OLT** Set if ocean loading tide correction error flag set at L1b or if correction not available at L1b.

**L2C\_SET** Set if solid Earth tide correction error flag set at L1b or if correction not available at L1b.

**L2C\_GPT** Set if geocentric polar tide correction error flag set at L1b or if correction not available at L1b.

**L2C\_SURFACE\_TYPE** Set if surface type flag error flag set at L1b or if correction not available at L1b.

**L2C\_MSS\_GEOID** Set if snow depth value is invalid.

**L2C\_ODLE\_MODEL** Set if the ODLE model is invalid.

**L2C\_ICE\_CONC** Set if the ice concentraion moedel is invalid.

**L2C\_SNOW\_DEPTH** Set if dry tropospheric correction error flag set at L1b or if correction not available at L1b.

**L2C\_SNOW\_DENSITY** Set if the snow density value is invalid.

**L2C\_SWH** Set if significant wave height value is invalid.

**L2C\_WINDSPEED** Set if windspeed value is invalid.

Definition at line 467 of file L2Structure.h.

### 7.20.3.3 enum t\_Level2CorrectionsAppliedFlagBit

Bit definitions for the L2 [Corrections](#) Application flag, see [L2FinalM::uj\\_C\\_applied](#)

#### Enumerator

**HGT\_INTERNAL\_PDS** Internal callibration.  
**HGT\_RAD\_DOPPLER\_PDS** Radial doppler.  
**HGT\_DRY\_TROPO\_PDS** Dry tropospheric.  
**HGT\_WET\_TROPO\_PDS** Wet tropospheric.  
**HGT\_INV\_BARO\_PDS** Inverse barometric.  
**HGT\_DAC\_PDS** Dynamic Atmospheric Correction.  
**HGT\_IONO\_GIM\_PDS** GIM ionospheric.  
**HGT\_IONO\_MODEL\_PDS** Model ionospheric.  
**HGT\_OT\_PDS** Ocean Tide.  
**HGT\_LPEOT\_PDS** LPEOT.  
**HGT\_OLT\_PDS** Ocean loading time.  
**HGT\_SET\_PDS** Solid Earth tide.  
**HGT\_GPT\_PDS** Geocentric Polar tide.  
**HGT\_SLOPE\_DOPPLER\_PDS** Slope Doppler.  
**HGT\_WINDOW\_OFFSET\_PDS** Mode specific window offset.  
**HGT\_SAR\_RETRK\_PDS** SAR retracker offset.  
**HGT\_SARIN\_RETRK\_PDS** SARIN retracker offset.  
**HGT\_LRM\_RETRK\_PDS** LRM retracker offset.  
**HGT\_LRM\_OCEANBIAS\_PDS** LRM ocean bias applied.  
**HGT\_LRM\_ICEBIAS\_PDS** LRM ice bias applied.  
**HGT\_SAR\_OCEANBIAS\_PDS** SAR ocean bias applied.  
**HGT\_SAR\_ICEBIAS\_PDS** SAR ice bias applied.  
**HGT\_SARIN\_OCEANBIAS\_PDS** SARin ocean bias applied.  
**HGT\_SARIN\_ICEBIAS\_PDS** SARin ice bias applied.  
**LRM\_SLOPE\_DATA\_VALID\_PDS** LRM slope model data valid.  
**HGT\_SARIN\_BAD\_BASELINE\_PDS** SARin Bad Baseline downward.  
**HGT\_SARIN\_OUT\_OF\_RANGE\_PDS** SARin out of range.  
**HGT\_SARIN\_BAD\_VELOCITY\_PDS** SARin Bad Velocity.  
**HGT\_SSB\_PDS** SSB Applied

Definition at line 549 of file L2Structure.h.

## 7.20.4 Function Documentation

### 7.20.4.1 void vDump\_L2Data ( L2Data \* pt, FILE \* pf )

Dump an L2 product in ASCII to a file.

#### Parameters

---

<i>pt</i>	Pointer to the product to dump.
<i>pf</i>	Pointer to the file to dump to.

Create an ASCII dump of an L2 product to a file.

#### Parameters

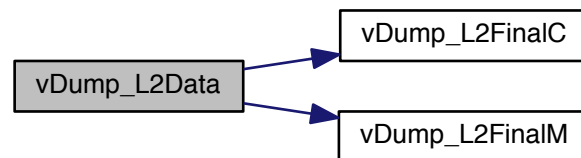
<i>pt</i>	Pointer to the product to dump.
<i>pf</i>	Pointer to the file to dump to.

#### Examples:

[examples/cs\\_l2\\_io\\_example.c](#).

Definition at line 21 of file L2Structure.c.

Here is the call graph for this function:



## 7.21 ptCSGetFileHandle.c File Reference

Implementation of a function to get a handle object representing a file.

```

#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <libgen.h>
#include "ptCSGetFileHandle.h"
#include "CS_Defines.h"
#include "kGetFileSize.h"
#include "L2Structure.h"
#include "L2IStructure.h"
#include "L1bStructure.h"
#include "jCheckFileForMPH.h"
#include "jCSGetDSFileOffsetFromHeader.h"

```

Include dependency graph for `ptCSGetFileHandle.c`:







## Functions

- `t_cs_filehandle` `ptCSGetFileHandle` (char \*s\_filename, `BASELINE` j\_baseline)

### 7.22.1 Detailed Description

Provide an opaque object that can store the information necessary to parse files.

### 7.22.2 Typedef Documentation

#### 7.22.2.1 typedef `t_cs_fileinfo*` `t_cs_filehandle`

Cryosat File Handle Type.

An opaque object that contains the information necessary for the Cryosat file handling library to access the file.

Definition at line 109 of file `ptCSGetFileHandle.h`.

### 7.22.3 Enumeration Type Documentation

#### 7.22.3.1 enum `BASELINE`

< Identifiers to use to select products conforming to a specific baseline format.

#### Enumerator

**`BASELINE_UNKNOWN`** Use for variable initialisation only.

**`BASELINE_A`** Attempt to read Baseline A

**`BASELINE_B`** Attempt to read Baseline B

**`BASELINE_BC`** Attempt to read Baseline BC (internal and selected projects only)

**`BASELINE_C`** Attempt to read Baseline C

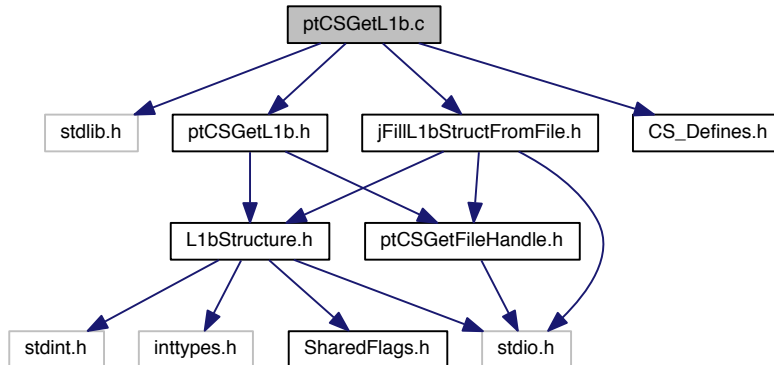
Definition at line 73 of file `ptCSGetFileHandle.h`.

## 7.23 ptCSGetL1b.c File Reference

Implementation of functions to read Cryosat Level 1b data files.

```
#include <stdlib.h>
#include "ptCSGetL1b.h"
#include "CS_Defines.h"
#include "jFillL1bStructFromFile.h"
```

Include dependency graph for ptCSGetL1b.c:



## Functions

- **L1B\_LRM** \* **ptCSGetL1bLRM** (**t\_cs\_filehandle** t\_handle, long int j\_dataset\_index, unsigned long long int uk\_num\_records, unsigned long long int uk\_skip\_records, **L1B\_LRM** \*pt\_data)
- **L1B\_SAR** \* **ptCSGetL1bSAR** (**t\_cs\_filehandle** t\_handle, long int j\_dataset\_index, unsigned long long int uk\_num\_records, unsigned long long int uk\_skip\_records, **L1B\_SAR** \*pt\_data)
- **L1B\_SARIN** \* **ptCSGetL1bSARIN** (**t\_cs\_filehandle** t\_handle, long int j\_dataset\_index, unsigned long long int uk\_num\_records, unsigned long long int uk\_skip\_records, **L1B\_SARIN** \*pt\_data)

### 7.23.1 Detailed Description

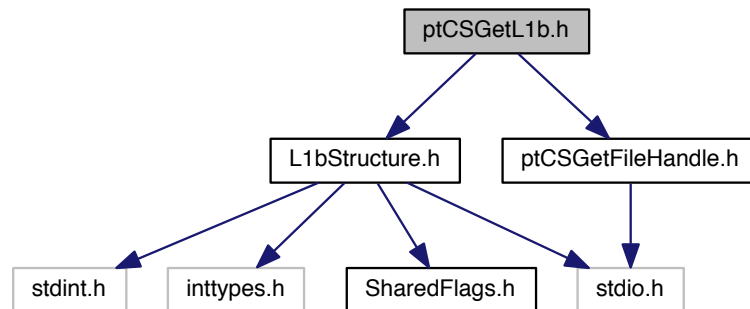
Implementation of functions to read Cryosat Level 1b data files.

## 7.24 ptCSGetL1b.h File Reference

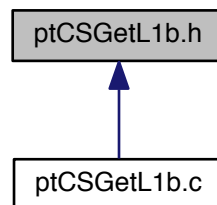
Functions to read Cryosat Level 1b data files.

```
#include "L1bStructure.h"
#include "ptCSGetFileHandle.h"
```

Include dependency graph for ptCSGetL1b.h:



This graph shows which files directly or indirectly include this file:



## Functions

- `L1B_LRM * ptCSGetL1bLRM (t_cs_filehandle t_handle, long int j_dataset_index, unsigned long long int uk_num_records, unsigned long long int uk_skip_records, L1B_LRM *pt_data)`
- `L1B_SAR * ptCSGetL1bSAR (t_cs_filehandle t_handle, long int j_dataset_index, unsigned long long int uk_num_records, unsigned long long int uk_skip_records, L1B_SAR *pt_data)`
- `L1B_SARIN * ptCSGetL1bSARIN (t_cs_filehandle t_handle, long int j_dataset_index, unsigned long long int uk_num_records, unsigned long long int uk_skip_records, L1B_SARIN *pt_data)`

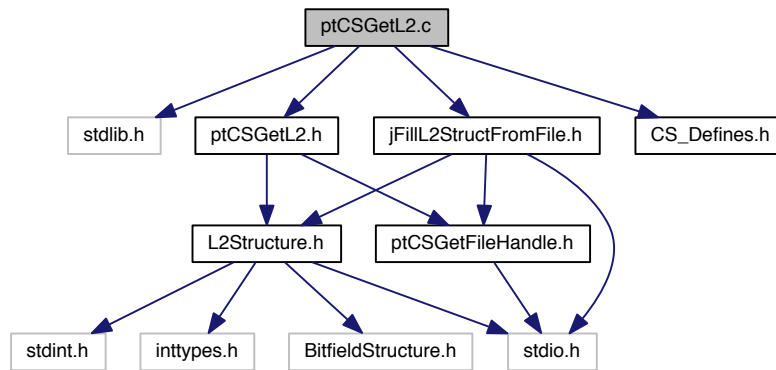
### 7.24.1 Detailed Description

Functions to read Cryosat Level 1b data files.

## 7.25 ptCSGetL2.c File Reference

Implementation of functions to read Cryosat Level 2 data files.

```
#include <stdlib.h>
#include "ptCSGetL2.h"
#include "CS_Defines.h"
#include "jFillL2StructFromFile.h"
Include dependency graph for ptCSGetL2.c:
```



## Functions

- [L2Data](#) \* [ptCSGetL2](#) ([t\\_cs\\_filehandle](#) t\_handle, long int j\_dataset\_index, unsigned long long int uk\_num\_↔ records, unsigned long long int uk\_skip\_records, [L2Data](#) \*pt\_data)

### 7.25.1 Detailed Description

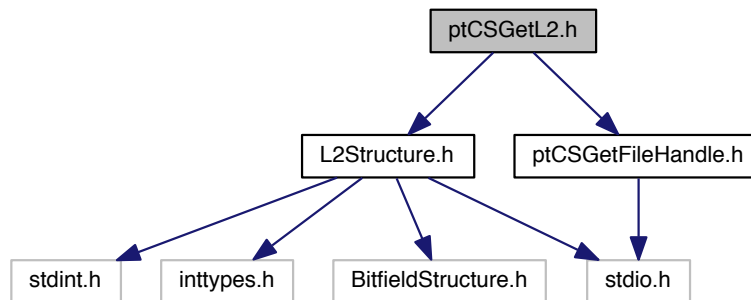
Implementation of functions to read Cryosat Level 2 data files.

## 7.26 ptCSGetL2.h File Reference

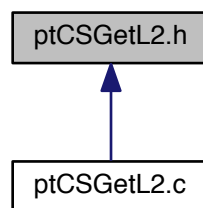
Functions to read Cryosat Level 2 data files.

```
#include "L2Structure.h"
#include "ptCSGetFileHandle.h"
```

Include dependency graph for ptCSGetL2.h:



This graph shows which files directly or indirectly include this file:



## Functions

- `L2Data * ptCSGetL2 (t_cs_filehandle t_handle, long int j_dataset_index, unsigned long long int uk_num_↔ records, unsigned long long int uk_skip_records, L2Data *pt_data)`

### 7.26.1 Detailed Description

Functions to read Cryosat Level 2 data files.

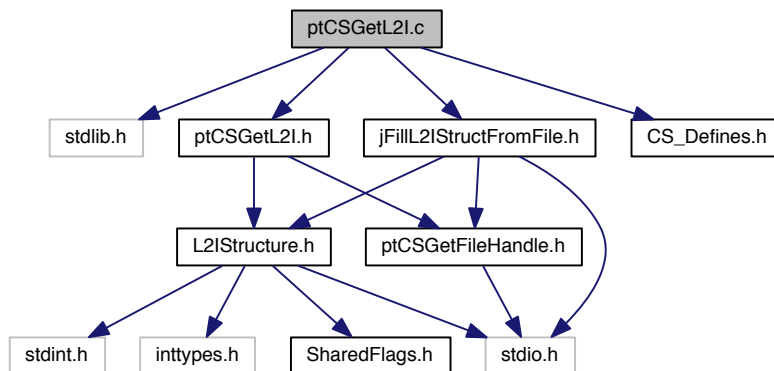
## 7.27 ptCSGetL2I.c File Reference

Implementation of functions to read Cryosat Level 2 data files.

```

#include <stdlib.h>
#include "ptCSGetL2I.h"
#include "CS_Defines.h"
#include "jFillL2IStructFromFile.h"
  
```

Include dependency graph for ptCSGetL2I.c:



## Functions

- [L2IData](#) \* [ptCSGetL2I](#) ([t\\_cs\\_filehandle](#) t\_handle, long int j\_dataset\_index, unsigned long long int uk\_num\_← records, unsigned long long int uk\_skip\_records, [L2IData](#) \*pt\_data)

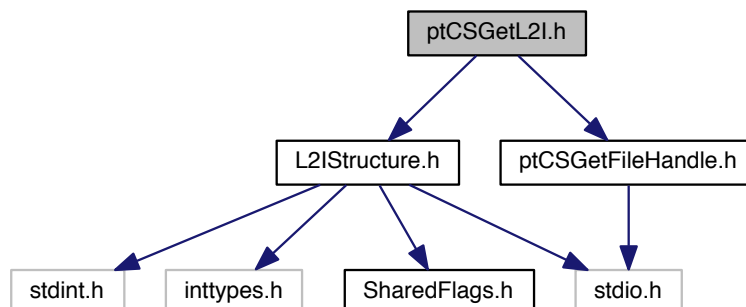
### 7.27.1 Detailed Description

Implementation of functions to read Cryosat Level 2 data files.

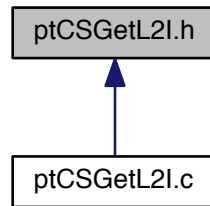
## 7.28 ptCSGetL2I.h File Reference

Functions to read Cryosat Level 2 data files.

```
#include "L2IStructure.h"
#include "ptCSGetFileHandle.h"
Include dependency graph for ptCSGetL2I.h:
```



This graph shows which files directly or indirectly include this file:



## Functions

- `L2IData * ptCSGetL2I (t_cs_filehandle t_handle, long int j_dataset_index, unsigned long long int uk_num_← records, unsigned long long int uk_skip_records, L2IData *pt_data)`

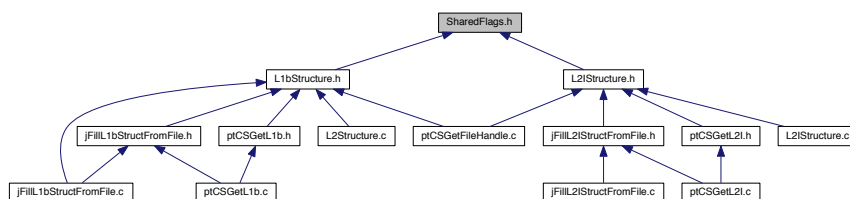
### 7.28.1 Detailed Description

## Functions to read Cryosat Level 2 data files.

## 7.29 SharedFlags.h File Reference

Enums for bitfield definitions and flags shared between L1b, L2, and L2i.

This graph shows which files directly or indirectly include this file:



## Enumerations

- enum t\_MCDFlagBit {  
MCD\_BLOCK\_DEGRADED = 31, MCD\_BLOCK\_BLANK = 30, MCD\_DATATION\_DEGRADED = 29, MCD\_ORBIT\_PROP\_ERROR = 28,  
MCD\_ORBIT\_FILE\_CHANGE = 27, MCD\_ORBIT\_DISCONTINUITY = 26, MCD\_ECHO\_SATURATION = 25, MCD\_ECHO\_ERROR = 24,  
MCD\_CH1\_ERROR = 23, MCD\_CH2\_ERROR = 22, MCD\_WINDOW\_DELAY\_INCONSISTANT = 21, MCD\_AGC\_INCONSISTANT = 20,  
MCD\_CAL\_1\_CORR\_MISSING = 19, MCD\_CAL\_1\_IPF\_DB\_USED = 18, MCD\_DORIS\_USO\_CORR\_MISSING = 17, MCD\_C\_CAL\_1\_IPF\_DB\_USED = 16,  
MCD\_TRK\_ECHO\_ERROR = 15, MCD\_ECHO\_RX1\_ERROR = 14, MCD\_ECHO\_RX2\_ERROR = 13, MCD\_ECHO\_RX3\_ERROR = 12, MCD\_ECHO\_RX4\_ERROR = 11,  
MCD\_ECHO\_RX5\_ERROR = 10, MCD\_ECHO\_RX6\_ERROR = 9, MCD\_ECHO\_RX7\_ERROR = 8, MCD\_ECHO\_RX8\_ERROR = 7, MCD\_ECHO\_RX9\_ERROR = 6,  
MCD\_ECHO\_RX10\_ERROR = 5, MCD\_ECHO\_RX11\_ERROR = 4, MCD\_ECHO\_RX12\_ERROR = 3, MCD\_ECHO\_RX13\_ERROR = 2, MCD\_ECHO\_RX14\_ERROR = 1,  
MCD\_ECHO\_RX15\_ERROR = 0;  
};

```

CD_NPM_INCONSISTANT = 12,
MCD_CAL1_TYPE = 11, MCD_PHASE_PETURBATION_APPLICATION = 7, MCD_CAL_2_CORR_MIS←
SING = 6, MCD_CAL_2_CORR_IPF_DB = 5,
MCD_POWER_SCALE_ERROR = 4, MCD_ATT_CORR_MISSING = 3, MCD_PHASE_PETURBATION_←
MODE = 1 }
• enum t_InstConf {
  RX_CHAIN_1 = 31, RX_CHAIN_0 = 30, SIRAL_ID = 29, BANDWIDTH_1 = 27,
  BANDWIDTH_0 = 26, TRK_MODE_1 = 23, TRK_MODE_0 = 22, EXTERNAL_CAL = 21,
  LOOP_STATUS = 19, LOSS_OF_ECHO = 18, RT_ERROR = 17, ECHO_SATURATION = 16,
  RX_BAND_ATT = 15, CYCLE_ERROR = 14, STR_ATTREF_USED = 10 }
• enum t_ModelID {
  MODE_ID_5 = 15, MODE_ID_4 = 14, MODE_ID_3 = 13, MODE_ID_2 = 12,
  MODE_ID_1 = 11, MODE_ID_0 = 10, SIN_DEGRADED = 9, CAL4_MODE = 8,
  PLAT_ATT_1 = 6, PLAT_ATT_0 = 5 }
• enum t_STID { ST_NONE = 0, ST_ATTREF = 4 }
• enum t_SurfaceType { SURFACE_OPEN_OCEAN = 0, SURFACE_CLOSED_SEA = 1, SURFACE_CONT←
INENTAL_ICE = 2, SURFACE_LAND = 3 }
• enum t_CorrectionStatusFlagBit {
  COR_DRY_TROPO = 31, COR_WET_TROPO = 30, COR_INV_BARO = 29, COR_DAC = 28,
  COR_IONO_GIM = 27, COR_IONO_MODEL = 26, COR_OT = 25, COR_LPEOT = 24,
  COR_OLT = 23, COR_SET = 22, COR_GPT = 21, COR_SURFACE_TYPE = 20,
  COR_ICE_CONC = 19, COR_SNOW_DEPTH = 18, COR_SNOW_DENSITY = 17, COR_MSS_MODEL =
  16,
  COR_GEOID_MODEL = 15, COR_ODLE_MODEL = 14, COR_DEM_MODEL = 13, COR_SLOPE_MODEL
  = 12,
  COR_SSB = 11 }
• enum t_CorrectionErrorFlagBit {
  ERROR_DRY_TROPO = 31, ERROR_WET_TROPO = 30, ERROR_INV_BARO = 29, ERROR_DAC = 28,
  ERROR_IONO_GIM = 27, ERROR_IONO_MODEL = 26, ERROR_OT = 25, ERROR_LPEOT = 24,
  ERROR_OLT = 23, ERROR_SET = 22, ERROR_GPT = 21, ERROR_SURFACE_TYPE = 20,
  ERROR_ICE_CONC = 19, ERROR_SNOW_DEPTH = 18, ERROR_SNOW_DENSITY = 17, ERROR_MS←
S_MODEL = 16,
  ERROR_GEOID_MODEL = 15, ERROR_ODLE_MODEL = 14, ERROR_DEM_MODEL = 13, ERROR_SL←
OPE_MODEL = 12,
  ERROR_SSB = 11, ERROR_SWH = 10, ERROR_WIND = 9 }

```

### 7.29.1 Detailed Description

Enums for bitfield definitions and flags shared between L1b, L2, and L2i.

This file contains convenience enums that describe bitfield settings for flags that are used unchanged in more than one of the products.

### 7.29.2 Enumeration Type Documentation

#### 7.29.2.1 enum t\_MCDFlagBit

Level 1b MCD bit definitions for [TimeAndOrbitGroup::uj\\_MCD](#), [L2IData::uj\\_L2\\_MCD](#)

Enumerator

**MCD\_BLOCK\_DEGRADED** Block degraded  
**MCD\_BLOCK\_BLANK** Blank inserted for padding  
**MCD\_DATATION\_DEGRADED** Datation blank or not set  
**MCD\_ORBIT\_PROP\_ERROR** Error returned by CFI or other check  
**MCD\_ORBIT\_FILE\_CHANGE** Orbit file changed since previous record



**MCD\_ORBIT\_DISCONTINUITY** Discontinuity (gap)

**MCD\_ECHO\_SATURATION** Saturated echo

**MCD\_ECHO\_ERROR** Another kind of echo error

**MCD\_CH1\_ERROR** Recieve channel degraded or missing

**MCD\_CH2\_ERROR** Recieve channel degraded or missing

**MCD\_WINDOW\_DELAY\_INCONSISTANT** Window delay inconsistency.

**MCD\_AGC\_INCONSISTANT** AGC inconsistency.

**MCD\_CAL\_1\_CORR\_MISSING** CAL 1 correction missing. Correction was not applied.

**MCD\_CAL\_1\_IPF\_DB\_USED** CAL 1 correction applied was from the IPF DB.

**MCD\_DORIS\_USO\_CORR\_MISSING** DORIS USO correction not available.

**MCD\_C\_CAL\_1\_IPF\_DB\_USED** Complex CAL1 correction applied was from the IPF DB.

**MCD\_TRK\_ECHO\_ERROR** Degraded tracking echo.

**MCD\_ECHO\_RX1\_ERROR** Bad raw echo on recieve chain 1.

**MCD\_ECHO\_RX2\_ERROR** Bad raw echo on recieve chain 2.

**MCD\_NPM\_INCONSISTANT** NPM inconsistency.

**MCD\_CAL1\_TYPE** CAL1 Correction Type. 0 = Peak Power. 1 = Integrated power.

**MCD\_CAL\_2\_CORR\_MISSING** Application of phase correction 0 = applied ; 1 = not appliedCAL 2 correction missing.

**MCD\_CAL\_2\_CORR\_IPF\_DB** CAL 2 correction applied was from the IPF DB.

**MCD\_POWER\_SCALE\_ERROR** Error in power scaling. Only used in LRM mode.

**MCD\_ATT\_CORR\_MISSING** Attitude correction missing.

**MCD\_PHASE\_PETURBATION\_MODE** Phase peturbation mode. 0 = CCAL1 ; 1 = IPFDB

Definition at line 14 of file SharedFlags.h.

#### 7.29.2.2 enum t\_InstConf

Bit ID's for the instrument config flag. See [TimeAndOrbitGroup::uj\\_Inst\\_config](#) and [L2IData::uj\\_Inst\\_config](#)

The two-bit integer assembled from the RX\_CHAIN bits has the following values:

Value	Meaning
0	unknown
1	chain1
2	chain 2
3	both chains

The two-bit integer assembled from the BANDWIDTH bits has the following values:

Value	Meaning
0	unknown
1	320 MHz
2	40 MHz

The two-bit integer assembled from the TRK\_MODE bits has the following values:

Value	Meaning
0	unknown
1	LRM
2	SAR

3	SARin
---	-------

#### Enumerator

***RX\_CHAIN\_1*** Bit 1 for RX\_CHAIN integer.  
***RX\_CHAIN\_0*** Bit 0 for RX\_CHAIN integer.  
***SIRAL\_ID*** SIRAL side identifier. 0 = Nominal, 1 = Redundant.  
***BANDWIDTH\_1*** Bit 1 for BANDWIDTH integer.  
***BANDWIDTH\_0*** Bit 0 for BANDWIDTH integer.  
***TRK\_MODE\_1*** Bit 1 for TRK\_MODE integer.  
***TRK\_MODE\_0*** Bit 0 for TRK\_MODE integer.  
***EXTERNAL\_CAL*** Set if external calibration.  
***LOOP\_STATUS*** 0 = closed loop, 1 = open loop  
***LOSS\_OF\_ECHO*** Set if loss of echo has occurred.  
***RT\_ERROR*** Set if the computing cycle took too long.  
***ECHO\_SATURATION*** Set if echo saturation was detected.  
***RX\_BAND\_ATT*** Set if reception band attenuated.  
***CYCLE\_ERROR*** Set if there was an error from the cycle report.  
***STR\_ATTREF\_USED*** Set if the STR\_ATTREF file was used for mispointing.

Definition at line 126 of file SharedFlags.h.

#### 7.29.2.3 enum t\_ModelID

Bit ID's for the mode ID flag. See [TimeAndOrbitGroup::ui\\_Mode\\_ID](#) and [L2IData::ui\\_Mode\\_ID](#)

The six-bit integer assembled from the MODE\_ID bits has the following values:

Value	Meaning
1	LRM
2	SAR
3	SARin
11	CAL1 LRM
12	CAL1 SAR
13	CAL1 SARin
22	CAL2 SAR
23	CAL2 SARin
33	CAL3

The two-bit integer assembled from the PLAT\_ATT bits has the following values:

Value	Meaning
0	unknown
1	Local normal pointing
2	Yaw steering mode

#### Enumerator

***MODE\_ID\_5*** Bit 5 for MODE\_ID integer  
***MODE\_ID\_4*** Bit 4 for MODE\_ID integer  
***MODE\_ID\_3*** Bit 3 for MODE\_ID integer  
***MODE\_ID\_2*** Bit 2 for MODE\_ID integer  
***MODE\_ID\_1*** Bit 1 for MODE\_ID integer  
***MODE\_ID\_0*** Bit 0 for MODE\_ID integer

***SIN\_DEGRADED*** Set to 1 if one receive chain is missing.  
***CAL4\_MODE*** Set to 1 for CAL 4 packets.  
***PLAT\_ATT\_1*** Bit 1 for PLAT\_ATT integer  
***PLAT\_ATT\_0*** Bit 0 for PLAT\_ATT integer

Definition at line 171 of file SharedFlags.h.

#### 7.29.2.4 enum t\_STID

Flag values for Star Tracker ID. See [TimeAndOrbitGroup::i\\_ST\\_ID](#) and [L2IData::i\\_ST\\_ID](#)

Enumerator

***ST\_NONE*** No star tracker data used  
***ST\_ATTREF*** STR\_ATTREF file used for product generation

Definition at line 187 of file SharedFlags.h.

#### 7.29.2.5 enum t\_SurfaceType

Values for Surface Type field, see [Corrections::uj\\_Surf\\_type](#) and [L2IData::uj\\_Surf\\_type](#).

Enumerator

***SURFACE\_OPEN\_OCEAN*** Surface is open ocean.  
***SURFACE\_CLOSED\_SEA*** Surface is closed sea.  
***SURFACE\_CONTINENTAL\_ICE*** Surface is continental ice.  
***SURFACE\_LAND*** Surface is land.

Definition at line 193 of file SharedFlags.h.

#### 7.29.2.6 enum t\_CorrectionStatusFlagBit

Bit IDs for Correction status flag. See [Corrections::uj\\_Corr\\_status](#) and [L2IData::uj\\_Corr\\_status](#)

Warning

Some bits are only used in the L2i product.

Enumerator

***COR\_DRY\_TROPO*** Dry tropospheric correction called.  
***COR\_WET\_TROPO*** Wet tropospheric correction called.  
***COR\_INV\_BARO*** Inverse barometric correction called.  
***COR\_DAC*** Inverse barometric correction called.  
***COR\_IONO\_GIM*** GIM ionospheric correction called.  
***COR\_IONO\_MODEL*** Model ionospheric correction called.  
***COR\_OT*** Ocean Tide called.  
***COR\_LPEOT*** LPEOT called.  
***COR\_OLT*** Ocean Loading Tide called.  
***COR\_SET*** Solid Earth Tide called.  
***COR\_GPT*** Geocentric Polar Tide called.

***COR\_SURFACE\_TYPE*** Surface Type Flag called.

***COR\_ICE\_CONC*** Ice concentration model called. L2i product only.

***COR\_SNOW\_DEPTH*** Snow depth model called. L2i product only.

***COR\_SNOW\_DENSITY*** Snow density model called. L2i product only.

***COR\_MSS\_MODEL*** MSS model called. L2i product only.

***COR\_GEOID\_MODEL*** Geoid model called. L2i product only.

***COR\_ODLE\_MODEL*** ODLE model called. L2i product only.

***COR\_DEM\_MODEL*** DEM model called. L2i product only.

***COR\_SLOPE\_MODEL*** Slope model called. L2i product only.

***COR\_SSB*** SSB called/calculated. L2i product only.

Definition at line 207 of file SharedFlags.h.

#### 7.29.2.7 enum **t\_CorrectionErrorFlagBit**

Bit IDs for Correction error flags. See [Corrections::uj\\_Corr\\_error](#) and [L2IData::uj\\_Corr\\_error](#)

#### Warning

Some bits are only used in the L2i product.

#### Enumerator

***ERROR\_DRY\_TROPO*** Dry tropospheric correction error.

***ERROR\_WET\_TROPO*** Wet tropospheric correction error.

***ERROR\_INV\_BARO*** Inverse barometric correction error.

***ERROR\_DAC*** DAC correction error.

***ERROR\_IONO\_GIM*** GIM ionospheric correction error.

***ERROR\_IONO\_MODEL*** Model ionospheric correction error.

***ERROR\_OT*** Ocean Tide error.

***ERROR\_LPEOT*** LPEOT error.

***ERROR\_OLT*** Ocean Loading Tide error.

***ERROR\_SET*** Solid Earth Tide error.

***ERROR\_GPT*** Geocentric Polar Tide error.

***ERROR\_SURFACE\_TYPE*** Surface Type Flag error.

***ERROR\_ICE\_CONC*** Ice conc error. L2i product only.

***ERROR\_SNOW\_DEPTH*** Snow depth error. L2i product only.

***ERROR\_SNOW\_DENSITY*** Snow density error. L2i product only.

***ERROR\_MSS\_MODEL*** MSS model error. L2i product only.

***ERROR\_GEOID\_MODEL*** Geoid model error. L2i product only.

***ERROR\_ODLE\_MODEL*** ODLE model error. L2i product only.

***ERROR\_DEM\_MODEL*** DEM model error. L2i product only.

***ERROR\_SLOPE\_MODEL*** Slope model error. L2i product only.

***ERROR\_SSB*** SSB error. Used for L2 output only.

***ERROR\_SWH*** SWH error. Used for L2 output only.

***ERROR\_WIND*** Wind speed error. Used for L2 output only.

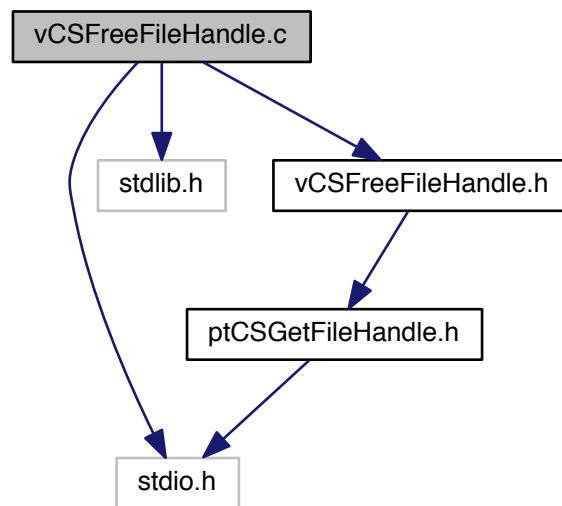
Definition at line 243 of file SharedFlags.h.

## 7.30 vCSFreeFileHandle.c File Reference

Implementation of functions to release file handles.

```
#include <stdio.h>
#include <stdlib.h>
#include "vCSFreeFileHandle.h"
```

Include dependency graph for vCSFreeFileHandle.c:



### Functions

- void `vCSFreeFileHandle` (`t_cs_filehandle` `t_handle`)

#### 7.30.1 Detailed Description

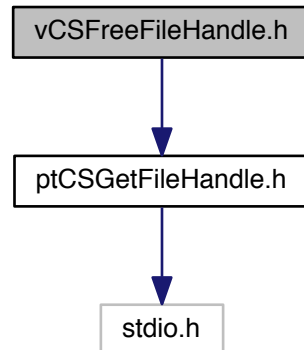
Implementation of functions to release file handles.

## 7.31 vCSFreeFileHandle.h File Reference

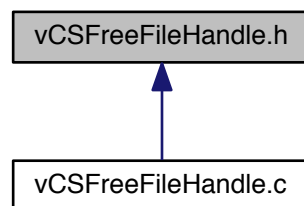
Functions to release file handles.

```
#include "ptCSGetFileHandle.h"
```

Include dependency graph for vCSFreeFileHandle.h:



This graph shows which files directly or indirectly include this file:



## Functions

- void `vCSFreeFileHandle` (`t_cs_filehandle` t\_handle)

### 7.31.1 Detailed Description

Functions to release file handles.

## Chapter 8

# Example Documentation

### 8.1 examples/cs\_l1b\_io\_example.c

This is an example of the most common way to use the Cryosat IO Library. The entire L1b file is read from the filename specified on the commandline. The day field is printed from the data read in as a test that the file has been read correctly. No complex error handling is performed.

```
/*-- pre-processor statements for conditional code --*/

/* =====
+
System: UCL Cryosat IO Library

Sub-system: Examples

Description: Example code for reading of Cryosat Level 1b products.

-----
Documentation cross reference:
Cryosat IO Library manual
-----

This file lives in ---
./src

Coding by:
David J Brockley
http://cryosat.mssl.ucl.ac.uk/public

-----
Filename      : $RCSfile: cs_l1b_io_example.c,v $
CVS Version Number : $Revision: 1.3 $
Last Update    : $Date: 2014/12/01 11:30:14 $
What String    : @(#)$Id: cs_l1b_io_example.c,v 1.3 2014/12/01 11:30:14 djb Exp $
Log           :
               $Log: cs_l1b_io_example.c,v $
               Revision 1.3  2014/12/01 11:30:14  djb
               Pre-testing for release v2.1

               Revision 1.2  2010/08/27 09:22:58  djb
               Prototype changes to fix on 64 bit machines and to correct the size of the
               quality flag in L2.

               Revision 1.1  2010/07/16 14:37:55  djb
               Port in CryoSat tools from internal branch.

               Revision 1.3  2005/10/13 14:04:46  djb
               More updates for 0.3 release

               Revision 1.2  2005/10/12 13:40:27  djb
               Updates for 0.3 release.

               Revision 1.1  2005/10/04 08:53:06  djb
               Added more files and documentation.

-----
```

```

Routines:
main

-----
This file contains the Unpublished Intellectual Property of
University College London and All Rights Are Reserved.
Copyright (c) University College London, 2005
-----
-
*/

#pragma ident "@(#) $Id: cs_l1b_io_example.c,v 1.3 2014/12/01 11:30:14 djb Exp $ MSSL"

/--- pre-processor statements ---/

/--- standard C headers ---/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdint.h>
#include <inttypes.h>

/--- programmer created headers ---/
#include "ptCSGetFileHandle.h"
#include "vCSFreeFileHandle.h"
#include "ptCSGetL1b.h"
#include "jCSNumRecordsInDataset.h"
#include "jCSGetDSFileOffsetFromHeader.h"
#include "BitmaskUtilities.h"
#include "jIOFunctions.h"
#include "CS_Defines.h"

/--- constant and macro headers ---/

/--- type definitions ---/

/--- global and external variable declarations ---/

/--- module private (static) variable declarations ---/

/--- functions ---/

int
main( int   argc,
      char* argv[] )
{
    L1B_LRM*      pt_L1b_LRM   = NULL;
    L1B_SAR*      pt_L1b_SAR   = NULL;
    L1B_SARIN*    pt_L1b_SARIN = NULL;

    t_cs_filehandle t_handle    = NULL;

    long int      j_records     = 0;
    long int      j_result      = 0;
    long int      i             = 0;
    long int      j_actual      = 0;

    CS_DS_NAME    t_ds_type = CS_DS_NAME_NONE;
    BASELINE      t_baseline = BASELINE_A;

    gj_verbose = 1;

    if( argc != 2 && argc != 3 )
    {
        printf( "usage: %s filename [baseline ID]\n", argv[ 0 ] );
        printf( "  baseline ID: 1=A 2=B 3=BC 4=C\n" );
        return EXIT_FAILURE;
    }

    if( argc == 3 )
    {
        switch( atol( argv[2] ) )
        {
            case 1:
                t_baseline = BASELINE_A;
                printf( "Expecting Baseline A data\n" );
                break;
            case 2:
                t_baseline = BASELINE_B;
                printf( "Expecting Baseline B data\n" );
                break;
            case 3:
                t_baseline = BASELINE_BC;
                printf( "Expecting Baseline BC data\n" );
                break;
            case 4:
                t_baseline = BASELINE_C;

```



```

        printf( "Expecting Baseline C data\n" );
        break;
    default:
        printf( "Unknown baseline ID\n" );
        return EXIT_FAILURE;
    }
}

iTestEndian();

t_handle = ptCSGetFileHandle(argv[ 1 ], t_baseline);

if( t_handle != NULL )
{
    /*
     * This is one way to determine what kind of file this is, by
     * looking at the header for each of the DS_NAMES that we can
     * read in turn.
     */

    j_result = jCSGetDSFileOffsetFromHeader(NULL, NULL,
                                             argv[ 1 ], CS_L1B_DS_NAME_LRM, &j_actual );

    if( j_result != EXIT_SUCCESS )
    {
        if( strstr( argv[1], "INT_LRM_1B" ) )
        {
            j_result = EXIT_SUCCESS;
        }
    }

    if( j_result == EXIT_SUCCESS )
    {
        printf( "Dataset of type L1b LRM detected in file.\n" );
        t_ds_type = CS_L1B_DS_NAME_LRM;
    }

    if( t_ds_type == CS_DS_NAME_NONE )
    {
        j_result = jCSGetDSFileOffsetFromHeader(NULL, NULL,
                                             argv[ 1 ], CS_L1B_DS_NAME_SAR, &j_actual );

        if( j_result != EXIT_SUCCESS )
        {
            if( strstr( argv[1], "INT_SAR_1B" ) || strstr( argv[1], "INT_SAR_A" ) )
            {
                j_result = EXIT_SUCCESS;
            }
        }

        if( j_result == EXIT_SUCCESS )
        {
            printf( "Dataset of type L1b SAR detected in file.\n" );
            t_ds_type = CS_L1B_DS_NAME_SAR;
        }
    }

    if( t_ds_type == CS_DS_NAME_NONE )
    {
        j_result = jCSGetDSFileOffsetFromHeader(NULL, NULL,
                                             argv[ 1 ], CS_L1B_DS_NAME_SARIN, &j_actual );

        if( j_result != EXIT_SUCCESS )
        {
            if( strstr( argv[1], "INT_SIN_1B" ) )
            {
                j_result = EXIT_SUCCESS;
            }
        }

        if( j_result == EXIT_SUCCESS )
        {
            printf( "Dataset of type L1b SARIN detected in file.\n" );
            t_ds_type = CS_L1B_DS_NAME_SARIN;
        }
    }

    if( t_ds_type == CS_DS_NAME_NONE )
    {
        printf( "Unable to determine type of file.\n" );
        return EXIT_FAILURE;
    }

    j_records = jCSNumRecordsInDataset( t_handle, 0 );

    if( j_records == -1 )
    {
        printf( "Unable to get number of records in file.\n" );
    }
}

```

```

    return EXIT_FAILURE;
}

printf( "There are %ld records in the file %s\n",
        j_records, argv[ 1 ] );

switch( t_ds_type )
{
    case CS_L1B_DS_NAME_LRM:
    {
        pt_L1b_LRM = ptCSGetL1bLRM( t_handle,
                                     0,
                                     j_records,
                                     0,
                                     NULL );

        if( pt_L1b_LRM != NULL )
        {
            printf( "L1b day [0][0]= %PRId32\n",
                    pt_L1b_LRM[ 0 ].at_Time_Orbit[ 0 ].j_Day );
            printf( "L1b day [0][1]= %PRId32\n",
                    pt_L1b_LRM[ 0 ].at_Time_Orbit[ 1 ].j_Day );
            printf( "L1b day [1][0]= %PRId32\n",
                    pt_L1b_LRM[ 1 ].at_Time_Orbit[ 0 ].j_Day );

            /*
             * Demonstrate how to check bits in flag words by
             * counting the number of blank blocks at the end of
             * the file.
             */
            for( i = 19; i > 0; i-- )
            {
                j_result = iCheckBitLong( &pt_L1b_LRM[ j_records - 1 ].
                                          at_Time_Orbit[ i ].uj_MCD,
                                          MCD_BLOCK_BLANK );

                /* Stop as soon as a non-blank block is found. */
                if( !j_result )
                {
                    printf( "There are %ld blank blocks at the end of the file.\n",
                            20 - i - 1 );
                    break;
                }
            }

            printf( "\n\n ** DUMP[0] ** \n\n" );
            vDump_L1B_LRM( &pt_L1b_LRM[ 0 ], NULL );
            printf( "\n\n ** DUMP[1] ** \n\n" );
            vDump_L1B_LRM( &pt_L1b_LRM[ 1 ], NULL );
        }
        else
        {
            printf( "Unable to read from file.\n" );
            return EXIT_FAILURE;
        }

        free( pt_L1b_LRM );
    } break;

    case CS_L1B_DS_NAME_SAR:
    {
        pt_L1b_SAR = ptCSGetL1bSAR( t_handle,
                                     0,
                                     j_records,
                                     0,
                                     NULL );

        if( pt_L1b_SAR != NULL )
        {
            printf( "L1b day [0][0]= %PRId32\n",
                    pt_L1b_SAR[ 0 ].at_Time_Orbit[ 0 ].j_Day );
            printf( "L1b day [0][1]= %PRId32\n",
                    pt_L1b_SAR[ 0 ].at_Time_Orbit[ 1 ].j_Day );
            printf( "L1b day [1][0]= %PRId32\n",
                    pt_L1b_SAR[ 1 ].at_Time_Orbit[ 0 ].j_Day );

            /*
             * Demonstrate how to check bits in flag words by
             * counting the number of blank blocks at the end of
             * the file.
             */
            for( i = 19; i > 0; i-- )
            {
                j_result = iCheckBitLong( &pt_L1b_SAR[ j_records - 1 ].
                                          at_Time_Orbit[ i ].uj_MCD,
                                          MCD_BLOCK_BLANK );

                /* Stop as soon as a non-blank block is found. */

```

```

        if( !j_result )
        {
            printf( "There are %ld blank blocks at the end of the file.\n",
                    20 - i - 1 );
            break;
        }
    }

    printf( "\n\n ** DUMP[0] ** \n\n" );
    vDump_L1B_SAR( &pt_L1b_SAR[ 0 ], NULL );
    printf( "\n\n ** DUMP[1] ** \n\n" );
    vDump_L1B_SAR( &pt_L1b_SAR[ 1 ], NULL );
}
else
{
    printf( "Unable to read from file.\n" );
    return EXIT_FAILURE;
}

free( pt_L1b_SAR );
} break;

case CS_L1B_DS_NAME_SARIN:
{
    pt_L1b_SARIN = ptCSGetL1bSARIN( t_handle,
                                     0,
                                     j_records,
                                     0,
                                     NULL );

    if( pt_L1b_SARIN != NULL )
    {
        printf( "L1b day [0][0]= %\"PRId32\"\n",
                pt_L1b_SARIN[ 0 ].at_Time_Orbit[ 0 ].j_Day );
        printf( "L1b day [0][1]= %\"PRId32\"\n",
                pt_L1b_SARIN[ 0 ].at_Time_Orbit[ 1 ].j_Day );
        printf( "L1b day [1][0]= %\"PRId32\"\n",
                pt_L1b_SARIN[ 1 ].at_Time_Orbit[ 0 ].j_Day );
        /*
         * Demonstrate how to check bits in flag words by
         * counting the number of blank blocks at the end of
         * the file.
         */
        for( i = 19; i > 0; i-- )
        {
            j_result = iCheckBitLong( &pt_L1b_SARIN[ j_records - 1 ].
                                     at_Time_Orbit[ i ].uj_MCD,
                                     MCD_BLOCK_BLANK );

            /* Stop as soon as a non-blank block is found. */
            if( !j_result )
            {
                printf( "There are %ld blank blocks at the end of the file.\n",
                        20 - i - 1 );
                break;
            }
        }

        if( i == 19 )
        {
            printf( "There are no blank blocks at the end of this file.\n" );
        }

        if( i == 0 )
        {
            printf( "This file has an entirely blank record at the end.\n" );
        }

        printf( "\n\n ** DUMP[0] ** \n\n" );
        vDump_L1B_SARIN( &pt_L1b_SARIN[ 0 ], NULL );
        printf( "\n\n ** DUMP[1] ** \n\n" );
        vDump_L1B_SARIN( &pt_L1b_SARIN[ 1 ], NULL );
    }
    else
    {
        printf( "Unable to read from file.\n" );
        return EXIT_FAILURE;
    }

    free( pt_L1b_SARIN );
} break;

default:
    printf( "Unable to find a dataset of a type that the library can read. No header and odd
filename?\n" );
} /* switch */

```

```

        vCSFreeFileHandle( t_handle );
    }
    else
    {
        printf( "Unable to open file.\n" );
        return EXIT_FAILURE;
    }

    return EXIT_SUCCESS;
} /* main */

```

## 8.2 examples/cs\_l2\_io\_example.c

This is an example of the most common way to use the Cryosat IO Library. The entire L2 file is read from the filename specified on the commandline. The day field is printed from the data read in as a test that the file has been read correctly. No complex error handling is performed.

```

/*-- pre-processor statements for conditional code --*/

/* =====
+

System: UCL Cryosat IO Library

Sub-system: Examples

Description: Example code for reading of Cryosat Level 2 products.

-----
Documentation cross reference:
Cryosat IO Library manual

-----
This file lives in ---
./src

Coding by:
David J Brockley
http://cryosat.mssl.ucl.ac.uk/public

-----
Filename      : $RCSfile: cs_l2_io_example.c,v $
CVS Version Number : $Revision: 1.3 $
Last Update    : $Date: 2014/12/01 11:30:14 $
What String    : @(#)$Id: cs_l2_io_example.c,v 1.3 2014/12/01 11:30:14 djb Exp $
Log           :

                  $Log: cs_l2_io_example.c,v $
                  Revision 1.3 2014/12/01 11:30:14 djb
                  Pre-testing for release v2.1

                  Revision 1.2 2010/08/27 09:22:59 djb
                  Prototype changes to fix on 64 bit machines and to correct the size of the
quality flag in L2.

                  Revision 1.1 2010/07/16 14:37:55 djb
                  Port in CryoSat tools from internal branch.

                  Revision 1.2 2005/09/26 09:50:21 djb
                  Changes to documentation.

                  Revision 1.1.1.1 2005/09/23 13:51:56 djb
                  Initial import of sources to CVS.

-----
Routines:
main

-----
This file contains the Unpublished Intellectual Property of
University College London and All Rights Are Reserved.
Copyright (c) University College London, 2005
-----
-
*/

#pragma ident "@(#) $Id: cs_l2_io_example.c,v 1.3 2014/12/01 11:30:14 djb Exp $ MSSL"

/*-- pre-processor statements --*/

```

```

/*-- standard C headers --*/
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <inttypes.h>

/*-- programmer created headers --*/
#include "ptCSGetFileHandle.h"
#include "vCSFreeFileHandle.h"
#include "ptCSGetL2.h"
#include "jCSNumRecordsInDataset.h"
#include "jIOFunctions.h"
#include "CS_Defines.h"

/*-- constant and macro headers --*/

/*-- type definitions --*/

/*-- global and external variable declarations --*/

/*-- module private (static) variable declarations --*/

/*-- functions --*/

int
main( int   argc,
      char* argv[] )
{
    L2Data*   pt_L2      = NULL;
    t_cs_filehandle t_handle = NULL;
    BASELINE  t_baseline = BASELINE_A;

    long int   j_records = 0;

    gj_verbose = 1;

    if( argc != 2 && argc != 3 )
    {
        printf( "usage: %s filename [baseline ID]\n", argv[ 0 ] );
        printf( "  baseline ID: 1=A 2=B 3=BC 4=C\n" );
        return EXIT_FAILURE;
    }

    if( argc == 3 )
    {
        switch( atol( argv[2] ) )
        {
            case 1:
                t_baseline = BASELINE_A;
                printf( "Expecting Baseline A data\n" );
                break;
            case 2:
                t_baseline = BASELINE_B;
                printf( "Expecting Baseline B data\n" );
                break;
            case 3:
                t_baseline = BASELINE_BC;
                printf( "Expecting Baseline BC data\n" );
                break;
            case 4:
                t_baseline = BASELINE_C;
                printf( "Expecting Baseline C data\n" );
                break;
            default:
                printf( "Unknown baseline ID\n" );
                return EXIT_FAILURE;
        }
    }

    iTestEndian();

    t_handle = ptCSGetFileHandle(argv[ 1 ], t_baseline );

    if( t_handle != NULL )
    {
        j_records = jCSNumRecordsInDataset( t_handle, 0 );

        if( j_records == -1 )
        {
            printf( "Unable to get number of records in file.\n" );
            return EXIT_FAILURE;
        }

        printf( "There are %ld records in the file %s\n",
                j_records, argv[ 1 ] );

        pt_L2 = ptCSGetL2( t_handle,

```

```

        0,
        j_records,
        0,
        NULL );

    if( pt_L2 != NULL )
    {
        printf( "L2 day [0] = %PRId32\n", pt_L2[ 0 ].j_Day );
        printf( "L2 day [1] = %PRId32\n", pt_L2[ 1 ].j_Day );

        printf( "\n\n ** DUMP[0] ** \n\n" );
        vDump_L2Data( &pt_L2[ 0 ], NULL );
        printf( "\n\n ** DUMP[1] ** \n\n" );
        vDump_L2Data( &pt_L2[ 1 ], NULL );
    }
    else
    {
        printf( "Unable to read from file.\n" );
        return EXIT_FAILURE;
    }

    free( pt_L2 );

    vCSFreeFileHandle( t_handle );
}
else
{
    printf( "Unable to open file.\n" );
    return EXIT_FAILURE;
}

return EXIT_SUCCESS;
} /* main */

```

### 8.3 examples/cs\_l2i\_io\_example.c

This is an example of the most common way to use the Cryosat IO Library. The entire L2I file is read from the filename specified on the commandline. The day field is printed from the data read in as a test that the file has been read correctly. No complex error handling is performed.

```

/*-- pre-processor statements for conditional code --*/

/* =====
+

System: UCL Cryosat IO Library

Sub-system: Examples

Description: Example code for reading of Cryosat Level 2 Intermediate products.

-----
Documentation cross reference:
Cryosat IO Library manual

-----
This file lives in ---
./src

Coding by:
David J Brockley
http://cryosat.mssl.ucl.ac.uk/public

-----
Filename           : $RCSfile: cs_l2i_io_example.c,v $
CVS Version Number : $Revision: 1.3 $
Last Update        : $Date: 2014/12/01 11:30:14 $
What String         : @(#) $Id: cs_l2i_io_example.c,v 1.3 2014/12/01 11:30:14 djb Exp $
Log                :
                    $Log: cs_l2i_io_example.c,v $
                    Revision 1.3 2014/12/01 11:30:14 djb
                    Pre-testing for release v2.1

                    Revision 1.2 2010/08/27 09:22:59 djb
                    Prototype changes to fix on 64 bit machines and to correct the size of the
                    quality flag in L2.

                    Revision 1.1 2010/07/16 14:37:55 djb
                    Port in CryoSat tools from internal branch.

```

Revision 1.2 2005/09/26 09:50:21 djb  
Changes to documentation.

Revision 1.1.1.1 2005/09/23 13:51:56 djb  
Initial import of sources to CVS.

```

-----
Routines:
main

-----

This file contains the Unpublished Intellectual Property of
University College London and All Rights Are Reserved.
Copyright (c) University College London, 2005
-----
-
*/

#pragma ident "@(#) $Id: cs_l2i_io_example.c,v 1.3 2014/12/01 11:30:14 djb Exp $ MSSSL"

/*-- pre-processor statements --*/

/*-- standard C headers --*/
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <inttypes.h>

/*-- programmer created headers --*/
#include "ptCSGetFileHandle.h"
#include "vCSFreeFileHandle.h"
#include "ptCSGetL2I.h"
#include "jCSNumRecordsInDataset.h"
#include "jIOFunctions.h"
#include "CS_Defines.h"

/*-- constant and macro headers --*/

/*-- type definitions --*/

/*-- global and external variable declarations --*/

/*-- module private (static) variable declarations --*/

/*-- functions --*/

int
main( int   argc,
      char* argv[] )
{
    L2IData*   pt_L2      = NULL;
    t_cs_filehandle t_handle = NULL;
    BASELINE   t_baseline = BASELINE_A;

    long int    j_records = 0;

    gj_verbose = 1;

    if( argc != 2 && argc != 3 )
    {
        printf( "usage: %s filename [baseline ID]\n", argv[ 0 ] );
        printf( "  baseline ID: 1=A 2=B 3=BC 4=C\n" );
        return EXIT_FAILURE;
    }

    if( argc == 3 )
    {
        switch( atol( argv[2] ) )
        {
            case 1:
                t_baseline = BASELINE_A;
                printf( "Expecting Baseline A data\n" );
                break;
            case 2:
                t_baseline = BASELINE_B;
                printf( "Expecting Baseline B data\n" );
                break;
            case 3:
                t_baseline = BASELINE_BC;
                printf( "Expecting Baseline BC data\n" );
                break;
            case 4:
                t_baseline = BASELINE_C;
                printf( "Expecting Baseline C data\n" );
                break;
            default:

```

```

        printf( "Unknown baseline ID\n" );
        return EXIT_FAILURE;
    }
}

iTestEndian();

t_handle = ptCSGetFileHandle(argv[ 1 ], t_baseline );

if( t_handle != NULL )
{
    j_records = jCSNumRecordsInDataset( t_handle, 0 );

    if( j_records == -1 )
    {
        printf( "Unable to get number of records in file.\n" );
        return EXIT_FAILURE;
    }

    printf( "There are %ld records in the file %s\n",
           j_records, argv[ 1 ] );

    pt_L2 = ptCSGetL2I( t_handle,
                       0,
                       j_records,
                       0,
                       NULL );

    if( pt_L2 != NULL )
    {
        printf( "L2 day [0] = %PRId32\n", pt_L2[ 0 ].j_Day );
        printf( "L2 day [1] = %PRId32\n", pt_L2[ 1 ].j_Day );

        printf( "\n\n ** DUMP[0] ** \n\n" );
        vDump_L2IData( &pt_L2[ 0 ], NULL );
        printf( "\n\n ** DUMP[1] ** \n\n" );
        vDump_L2IData( &pt_L2[ 1 ], NULL );
    }
    else
    {
        printf( "Unable to read from file.\n" );
        return EXIT_FAILURE;
    }

    free( pt_L2 );

    vCSFreeFileHandle( t_handle );
}
else
{
    printf( "Unable to open file.\n" );
    return EXIT_FAILURE;
}

return EXIT_SUCCESS;
} /* main */

```

## 8.4 examples/example\_bit\_manipulation.c

This is an example of the use of the bitmask handling functions to check the status of a bit contained within a bitfield stored within a `int16_t`. The process is identical for bitfields stored within a `int32_t`, except for the replacement of the word Short in the function names with Long e.g. `vSetBitShort` becomes `vSetBitLong`.

```

/*-- pre-processor statements for conditional code --*/

/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
+
System: UCL Cryosat IO Library

Sub-system: Examples

Description: Example code testing, clearing and setting bits

-----
Documentation cross reference:
Cryosat IO Library manual
-----

This file lives in ---

```



```

./src

Coding by:
David J Brockley
http://cryosat.mssl.ucl.ac.uk/public

-----
Filename      : $RCSfile: example_bit_manipulation.c,v $
CVS Version Number : $Revision: 1.2 $
Last Update   : $Date: 2014/12/01 11:30:14 $
What String   : @(#) $Id: example_bit_manipulation.c,v 1.2 2014/12/01 11:30:14 djb Exp $
Log           :

                $Log: example_bit_manipulation.c,v $
                Revision 1.2  2014/12/01 11:30:14  djb
                Pre-testing for release v2.1

                Revision 1.1  2010/07/16 14:37:55  djb
                Port in CryoSat tools from internal branch.

-----

Routines:
main

-----

This file contains the Unpublished Intellectual Property of
University College London and All Rights Are Reserved.
    Copyright (c) University College London, 2005
-----
-
*/

#pragma ident "@(#) $Id: example_bit_manipulation.c,v 1.2 2014/12/01 11:30:14 djb Exp $ MSSL"

/*-- pre-processor statements --*/

/*-- standard C headers --*/
#include <stdio.h>
#include <stdlib.h>

/*-- programmer created headers --*/
#include "ptCSGetFileHandle.h"
#include "vCSFreeFileHandle.h"
#include "ptCSGetLlb.h"
#include "jCSNumRecordsInDataset.h"
#include "jCSGetDSFileOffsetFromHeader.h"
#include "BitmaskUtilities.h"
#include "jIOFunctions.h"

/*-- constant and macro headers --*/

/*-- type definitions --*/

/*-- global and external variable declarations --*/

/*-- module private (static) variable declarations --*/

/*-- functions --*/

int
main( int   argc,
      char* argv[] )
{
    Llb_LRM*      pt_Llb_LRM = NULL;

    t_cs_filehandle t_handle = NULL;

    long int      j_records = 0;
    long int      j_result  = 0;
    long int      i         = 0;
    long int      j_actual  = 0;

    CS_DS_NAME    t_ds_type = CS_DS_NAME_NONE;

    if( argc != 2 )
    {
        printf( "usage: %s filename\n", argv[ 0 ] );

        return EXIT_FAILURE;
    }

    iTestEndian();

    t_handle = ptCSGetFileHandle( argv[ 1 ], BASELINE_A );

```

```

if( t_handle != NULL )
{
    /*
     * This is one way to determine what kind of file this is, by
     * looking at the header for each of the DS_NAMES that we can
     * read in turn.
     */

    j_result = jCSGetDSFileOffsetFromHeader( NULL, NULL,
                                              argv[ 1 ], CS_L1B_DS_NAME_LRM, &j_actual );

    if( j_result == EXIT_SUCCESS )
    {
        printf( "Dataset of type L1b LRM detected in file.\n" );
        t_ds_type = CS_L1B_DS_NAME_LRM;

        j_records = jCSNumRecordsInDataset( t_handle, 0 );

        if( j_records == -1 )
        {
            printf( "Unable to get number of records in file.\n" );
            return EXIT_FAILURE;
        }

        /* Read in one record for test purposes */
        pt_L1b_LRM = ptCSGetL1bLRM( t_handle,
                                     0,
                                     j_records,
                                     1,
                                     NULL );

        if( pt_L1b_LRM != NULL )
        {
            /* Now to look at the bits */
            j_result = iCheckBitShort( &pt_L1b_LRM->
                                      t_Average_PL.ui_flags,
                                      WFM_LRM_ERROR );

            if( j_result )
            {
                /* The bit was set */
                printf( "Averaged waveform error\n" );
            }
            else
            {
                /* The bit was clear */
                printf( "Averaged waveform ok\n" );
            }

            /*
             * Just to show how, this is how to set the bit to what
             * we have already determined its value to be.
             */
            if( j_result )
            {
                vSetBitShort( &pt_L1b_LRM->
                             t_Average_PL.ui_flags,
                             WFM_LRM_ERROR );
            }
            else
            {
                vClearBitShort( &pt_L1b_LRM->
                               t_Average_PL.ui_flags,
                               WFM_LRM_ERROR );
            }
        }
    }
}
}
}
} /* main */

```

# Index

AMB\_FAIL  
    L2IStructure.h, [118](#)  
AMB\_SARIN\_BAD\_MATH  
    L2IStructure.h, [118](#)  
AMB\_SARIN\_DIFF\_ELEV  
    L2IStructure.h, [118](#)  
AMB\_SARIN\_NO\_DEM  
    L2IStructure.h, [118](#)  
AMB\_SARIN\_NO\_RTRK  
    L2IStructure.h, [118](#)  
ai\_bb\_param  
    L2IData, [65](#)  
    WaveformsSAR, [87](#)  
    WaveformsSARIN, [89](#)  
aj\_Baseline  
    L2IData, [59](#)  
    TimeAndOrbitGroup, [84](#)  
aj\_Real\_beam  
    L2IData, [58](#)  
    TimeAndOrbitGroup, [83](#)  
aj\_Sat\_velocity  
    L2IData, [58](#)  
    TimeAndOrbitGroup, [83](#)  
as\_CS\_DS\_NAME  
    CS\_Defines.c, [94](#)  
    CS\_Defines.h, [96](#)  
at\_Meas  
    L2Data, [45](#)  
at\_Measurements  
    L1B\_LRM, [40](#)  
    L1B\_SAR, [41](#)  
    L1B\_SARIN, [42](#)  
at\_Time\_Orbit  
    L1B\_LRM, [40](#)  
    L1B\_SAR, [41](#)  
    L1B\_SARIN, [42](#)  
at\_Waveforms  
    L1B\_LRM, [40](#)  
    L1B\_SAR, [41](#)  
    L1B\_SARIN, [42](#)  
auc\_spares1  
    L2IData, [68](#)  
auc\_spares2  
    L2IData, [72](#)  
auc\_spares3  
    L2IData, [74](#)  
auc\_spares4  
    L2IData, [75](#)  
AveragedWaveformsLRM, [29](#)  
    j\_Alt\_1Hz, [30](#)  
    j\_Day\_1Hz, [30](#)  
    j\_Lat\_1Hz, [30](#)  
    j\_Linear\_wfm\_mult, [31](#)  
    j\_Lon\_1Hz, [30](#)  
    j\_Power2\_wfm\_mult, [31](#)  
    k\_TD\_1Hz, [30](#)  
    pui\_avg\_wfm, [30](#)  
    ui\_flags, [31](#)  
    ui\_num\_avg\_echoes, [31](#)  
    uj\_Micsec\_1Hz, [30](#)  
    uj\_Sec\_1Hz, [30](#)  
AveragedWaveformsSAR, [31](#)  
    j\_Alt\_1Hz, [32](#)  
    j\_Day\_1Hz, [32](#)  
    j\_Lat\_1Hz, [32](#)  
    j\_Linear\_wfm\_mult, [33](#)  
    j\_Lon\_1Hz, [32](#)  
    j\_Power2\_wfm\_mult, [33](#)  
    k\_TD\_1Hz, [32](#)  
    pui\_avg\_wfm, [33](#)  
    ui\_flags, [33](#)  
    ui\_num\_avg\_echoes, [33](#)  
    uj\_Micsec\_1Hz, [32](#)  
    uj\_Sec\_1Hz, [32](#)  
AveragedWaveformsSARIN, [33](#)  
    j\_Alt\_1Hz, [35](#)  
    j\_Day\_1Hz, [34](#)  
    j\_Lat\_1Hz, [35](#)  
    j\_Linear\_wfm\_mult, [35](#)  
    j\_Lon\_1Hz, [35](#)  
    j\_Power2\_wfm\_mult, [35](#)  
    k\_TD\_1Hz, [35](#)  
    pui\_avg\_wfm, [35](#)  
    ui\_flags, [36](#)  
    ui\_num\_avg\_echoes, [35](#)  
    uj\_Micsec\_1Hz, [34](#)  
    uj\_Sec\_1Hz, [34](#)  
  
BANDWIDTH\_0  
    SharedFlags.h, [136](#)  
BANDWIDTH\_1  
    SharedFlags.h, [136](#)  
BASELINE  
    ptCSGetFileHandle.h, [127](#)  
BASELINE\_A  
    ptCSGetFileHandle.h, [127](#)  
BASELINE\_B  
    ptCSGetFileHandle.h, [127](#)  
BASELINE\_BC

- ptCSGetFileHandle.h, 127
- BASELINE\_C
  - ptCSGetFileHandle.h, 127
- BASELINE\_UNKNOWN
  - ptCSGetFileHandle.h, 127
- BitmaskUtilities.c, 91
  - vClearBitLong, 92
- BitmaskUtilities.h, 92
  - vClearBitLong, 93
- CAL4\_MODE
  - SharedFlags.h, 137
- CL2\_LRM\_IS\_LAND\_ICE\_PLATEAU
  - L2IStructure.h, 117
- CL2\_LRM\_IS\_OCEAN
  - L2IStructure.h, 117
- CL2\_LRM\_IS\_UNDEFINED
  - L2IStructure.h, 117
- CL2\_SAR\_IS\_LEADS
  - L2IStructure.h, 117
- CL2\_SAR\_IS\_OCEAN
  - L2IStructure.h, 117
- CL2\_SAR\_IS\_SEA\_ICE
  - L2IStructure.h, 117
- CL2\_SAR\_IS\_UNDEFINED
  - L2IStructure.h, 117
- CL2\_SARIN\_IS\_ALTIMETER\_MODE
  - L2IStructure.h, 117
- CL2\_SARIN\_IS\_UNDEFINED
  - L2IStructure.h, 117
- COR\_DAC
  - SharedFlags.h, 137
- COR\_DEM\_MODEL
  - SharedFlags.h, 138
- COR\_DRY\_TROPO
  - SharedFlags.h, 137
- COR\_GEOID\_MODEL
  - SharedFlags.h, 138
- COR\_GPT
  - SharedFlags.h, 137
- COR\_ICE\_CONC
  - SharedFlags.h, 138
- COR\_INV\_BARO
  - SharedFlags.h, 137
- COR\_IONO\_GIM
  - SharedFlags.h, 137
- COR\_IONO\_MODEL
  - SharedFlags.h, 137
- COR\_LPEOT
  - SharedFlags.h, 137
- COR\_MSS\_MODEL
  - SharedFlags.h, 138
- COR\_ODLE\_MODEL
  - SharedFlags.h, 138
- COR\_OLT
  - SharedFlags.h, 137
- COR\_OT
  - SharedFlags.h, 137
- COR\_SET
  - SharedFlags.h, 137
- SharedFlags.h, 137
- COR\_SLOPE\_MODEL
  - SharedFlags.h, 138
- COR\_SNOW\_DENSITY
  - SharedFlags.h, 138
- COR\_SNOW\_DEPTH
  - SharedFlags.h, 138
- COR\_SSB
  - SharedFlags.h, 138
- COR\_SURFACE\_TYPE
  - SharedFlags.h, 137
- COR\_WET\_TROPO
  - SharedFlags.h, 137
- CS\_DS\_NAME
  - CS\_Defines.h, 96
- CS\_Defines.c, 93
  - as\_CS\_DS\_NAME, 94
  - gj\_verbose, 94
  - J\_CS\_L2\_RECORD\_SIZE, 94
  - J\_CS\_L2I\_RECORD\_SIZE, 94
- CS\_Defines.h, 95
  - as\_CS\_DS\_NAME, 96
  - CS\_DS\_NAME, 96
  - FALSE, 96
  - gj\_verbose, 96
  - J\_CS\_L2\_RECORD\_SIZE, 96
  - J\_CS\_L2I\_RECORD\_SIZE, 96
  - J\_MPH\_SIZE, 96
  - TRUE, 96
- CYCLE\_ERROR
  - SharedFlags.h, 136
- Corrections, 36
  - j\_DAC\_C, 37
  - j\_Dry\_C, 37
  - j\_H\_GPT, 38
  - j\_H\_LPEOT, 38
  - j\_H\_OLT, 38
  - j\_H\_OT, 38
  - j\_H\_SET, 38
  - j\_IB\_C, 37
  - j\_Iono\_GIM, 37
  - j\_Iono\_mod, 37
  - j\_Wet\_C, 37
  - spare1, 38
  - spare2, 39
  - uj\_Corr\_error, 39
  - uj\_Corr\_status, 38
  - uj\_Surf\_type, 38
- DISC\_FAIL
  - L2IStructure.h, 117
- DISC\_SAR\_BAD\_BB
  - L2IStructure.h, 117
- DISC\_SAR\_BAD\_NOISE
  - L2IStructure.h, 117
- DISC\_SAR\_BAD\_WIDTH
  - L2IStructure.h, 117
- DISC\_SAR\_HI\_PK
  - L2IStructure.h, 117

- DISC\_SAR\_ICECONC\_UNAV
  - L2IStructure.h, [117](#)
- DISC\_SAR\_ICECONC\_UNREAL
  - L2IStructure.h, [117](#)
- DISC\_SAR\_LO\_PK
  - L2IStructure.h, [117](#)
- DISC\_SAR\_LO\_POWER
  - L2IStructure.h, [117](#)
- DISC\_SAR\_MULTIPLE\_MATCH
  - L2IStructure.h, [117](#)
- DISC\_SAR\_NO\_MATCH
  - L2IStructure.h, [117](#)
- DISC\_SARIN\_BAD\_LE
  - L2IStructure.h, [117](#)
- DISC\_SARIN\_HI\_NOISE
  - L2IStructure.h, [117](#)
- DISC\_SARIN\_HI\_PEAKINESS
  - L2IStructure.h, [117](#)
- DISC\_SARIN\_LO\_VARIANCE
  - L2IStructure.h, [117](#)
- DISC\_SARIN\_LOW\_PEAKINESS
  - L2IStructure.h, [117](#)
- DISC\_SARIN\_LOW\_POWER
  - L2IStructure.h, [117](#)
- DISC\_TBD
  - L2IStructure.h, [117](#)
- Documentation.h, [97](#)
- ECHO\_SATURATION
  - SharedFlags.h, [136](#)
- ERROR\_DAC
  - SharedFlags.h, [138](#)
- ERROR\_DEM\_MODEL
  - SharedFlags.h, [138](#)
- ERROR\_DRY\_TROPO
  - SharedFlags.h, [138](#)
- ERROR\_GEOID\_MODEL
  - SharedFlags.h, [138](#)
- ERROR\_GPT
  - SharedFlags.h, [138](#)
- ERROR\_ICE\_CONC
  - SharedFlags.h, [138](#)
- ERROR\_INV\_BARO
  - SharedFlags.h, [138](#)
- ERROR\_IONO\_GIM
  - SharedFlags.h, [138](#)
- ERROR\_IONO\_MODEL
  - SharedFlags.h, [138](#)
- ERROR\_LPEOT
  - SharedFlags.h, [138](#)
- ERROR\_MSS\_MODEL
  - SharedFlags.h, [138](#)
- ERROR\_ODLE\_MODEL
  - SharedFlags.h, [138](#)
- ERROR\_OLT
  - SharedFlags.h, [138](#)
- ERROR\_OT
  - SharedFlags.h, [138](#)
- ERROR\_SET
  - SharedFlags.h, [138](#)
- ERROR\_SLOPE\_MODEL
  - SharedFlags.h, [138](#)
- ERROR\_SNOW\_DENSITY
  - SharedFlags.h, [138](#)
- ERROR\_SNOW\_DEPTH
  - SharedFlags.h, [138](#)
- ERROR\_SSB
  - SharedFlags.h, [138](#)
- ERROR\_SURFACE\_TYPE
  - SharedFlags.h, [138](#)
- ERROR\_SWH
  - SharedFlags.h, [138](#)
- ERROR\_WET\_TROPO
  - SharedFlags.h, [138](#)
- ERROR\_WIND
  - SharedFlags.h, [138](#)
- EXTERNAL\_CAL
  - SharedFlags.h, [136](#)
- FALSE
  - CS\_Defines.h, [96](#)
- FREB\_SAR\_AVAIL
  - L2IStructure.h, [116](#)
- FREB\_SAR\_IN\_NORTH\_BOUND
  - L2IStructure.h, [116](#)
- FREB\_SAR\_IN\_SOUTH\_BOUND
  - L2IStructure.h, [116](#)
- FREB\_SAR\_UNREL
  - L2IStructure.h, [116](#)
- gj\_verbose
  - CS\_Defines.c, [94](#)
  - CS\_Defines.h, [96](#)
- HGT\_DAC
  - L2IStructure.h, [116](#)
- HGT\_DAC\_PDS
  - L2Structure.h, [124](#)
- HGT\_DRY\_TROPO
  - L2IStructure.h, [116](#)
- HGT\_DRY\_TROPO\_PDS
  - L2Structure.h, [124](#)
- HGT\_FAIL
  - L2IStructure.h, [116](#)
- HGT\_GPT
  - L2IStructure.h, [116](#)
- HGT\_GPT\_PDS
  - L2Structure.h, [124](#)
- HGT\_INTERNAL
  - L2IStructure.h, [115](#)
- HGT\_INTERNAL\_PDS
  - L2Structure.h, [124](#)
- HGT\_INV\_BARO
  - L2IStructure.h, [116](#)
- HGT\_INV\_BARO\_PDS
  - L2Structure.h, [124](#)
- HGT\_IONO\_GIM
  - L2IStructure.h, [116](#)

- HGT\_IONO\_GIM\_PDS
  - L2Structure.h, [124](#)
- HGT\_IONO\_MODEL
  - L2Structure.h, [116](#)
- HGT\_IONO\_MODEL\_PDS
  - L2Structure.h, [124](#)
- HGT\_LPEOT
  - L2Structure.h, [116](#)
- HGT\_LPEOT\_PDS
  - L2Structure.h, [124](#)
- HGT\_LRM\_ICEBIAS
  - L2Structure.h, [116](#)
- HGT\_LRM\_ICEBIAS\_PDS
  - L2Structure.h, [124](#)
- HGT\_LRM\_OCEANBIAS
  - L2Structure.h, [116](#)
- HGT\_LRM\_OCEANBIAS\_PDS
  - L2Structure.h, [124](#)
- HGT\_LRM\_RETRK
  - L2Structure.h, [116](#)
- HGT\_LRM\_RETRK\_PDS
  - L2Structure.h, [124](#)
- HGT\_OLT
  - L2Structure.h, [116](#)
- HGT\_OLT\_PDS
  - L2Structure.h, [124](#)
- HGT\_OT
  - L2Structure.h, [116](#)
- HGT\_OT\_PDS
  - L2Structure.h, [124](#)
- HGT\_RAD\_DOPPLER
  - L2Structure.h, [115](#)
- HGT\_RAD\_DOPPLER\_PDS
  - L2Structure.h, [124](#)
- HGT\_SAR\_ICEBIAS
  - L2Structure.h, [116](#)
- HGT\_SAR\_ICEBIAS\_PDS
  - L2Structure.h, [124](#)
- HGT\_SAR\_OCEANBIAS
  - L2Structure.h, [116](#)
- HGT\_SAR\_OCEANBIAS\_PDS
  - L2Structure.h, [124](#)
- HGT\_SAR\_RETRK
  - L2Structure.h, [116](#)
- HGT\_SAR\_RETRK\_PDS
  - L2Structure.h, [124](#)
- HGT\_SARIN\_BAD\_BASELINE
  - L2Structure.h, [116](#)
- HGT\_SARIN\_BAD\_BASELINE\_PDS
  - L2Structure.h, [124](#)
- HGT\_SARIN\_BAD\_VELOCITY
  - L2Structure.h, [116](#)
- HGT\_SARIN\_BAD\_VELOCITY\_PDS
  - L2Structure.h, [124](#)
- HGT\_SARIN\_ICEBIAS
  - L2Structure.h, [116](#)
- HGT\_SARIN\_ICEBIAS\_PDS
  - L2Structure.h, [124](#)
- HGT\_SARIN\_OCEANBIAS
  - L2Structure.h, [116](#)
- HGT\_SARIN\_OCEANBIAS\_PDS
  - L2Structure.h, [124](#)
- HGT\_SARIN\_OUT\_OF\_RANGE
  - L2Structure.h, [116](#)
- HGT\_SARIN\_OUT\_OF\_RANGE\_PDS
  - L2Structure.h, [124](#)
- HGT\_SARIN\_RETRK
  - L2Structure.h, [116](#)
- HGT\_SARIN\_RETRK\_PDS
  - L2Structure.h, [124](#)
- HGT\_SET
  - L2Structure.h, [116](#)
- HGT\_SET\_PDS
  - L2Structure.h, [124](#)
- HGT\_SLOPE\_DOPPLER
  - L2Structure.h, [116](#)
- HGT\_SLOPE\_DOPPLER\_PDS
  - L2Structure.h, [124](#)
- HGT\_SSB
  - L2Structure.h, [116](#)
- HGT\_SSB\_PDS
  - L2Structure.h, [124](#)
- HGT\_WET\_TROPO
  - L2Structure.h, [116](#)
- HGT\_WET\_TROPO\_PDS
  - L2Structure.h, [124](#)
- HGT\_WINDOW\_OFFSET
  - L2Structure.h, [116](#)
- HGT\_WINDOW\_OFFSET\_PDS
  - L2Structure.h, [124](#)
- High Level CryoSat Level 1b File IO API, [12](#)
  - ptCSGetL1bLRM, [12](#)
  - ptCSGetL1bSAR, [13](#)
  - ptCSGetL1bSARIN, [13](#)
- High Level CryoSat Level 2 File IO API, [17](#)
  - ptCSGetL2, [17](#)
  - ptCSGetL2I, [18](#)
- High-level shared routines, [9](#)
  - jCSGetDSFileOffsetFromHeader, [10](#)
  - jCSNumRecordsInDataset, [10](#)
  - ptCSGetFileHandle, [9](#)
  - vCSFreeFileHandle, [10](#)
- i\_DAC\_C
  - L2FinalC, [47](#)
- i\_Dry\_C
  - L2FinalC, [47](#)
- i\_Freeboard
  - L2FinalM, [52](#)
- i\_H\_GPT
  - L2FinalC, [48](#)
- i\_H\_LPEOT
  - L2FinalC, [48](#)
- i\_H\_OLT
  - L2FinalC, [48](#)
- i\_H\_OT
  - L2FinalC, [47](#)

- i\_H\_SET
  - L2FinalC, [48](#)
- i\_IB\_C
  - L2FinalC, [47](#)
- i\_Ice\_conc
  - L2FinalC, [49](#)
- i\_Iono\_C
  - L2FinalC, [47](#)
- i\_Mispointing
  - L2Data, [44](#)
- i\_N\_Valid
  - L2Data, [45](#)
- i\_N\_avg
  - L2FinalM, [52](#)
- i\_Peakiness
  - L2FinalM, [52](#)
- i\_SSB
  - L2FinalC, [47](#)
- i\_SSHA\_interp
  - L2FinalM, [51](#)
- i\_SSHA\_num
  - L2FinalM, [51](#)
- i\_SSHA\_qual
  - L2FinalM, [52](#)
- i\_ST\_ID
  - L2IData, [59](#)
  - TimeAndOrbitGroup, [84](#)
- i\_SWH
  - L2FinalC, [49](#)
- i\_Sig0
  - L2FinalM, [52](#)
- i\_Sig0\_2
  - L2FinalM, [52](#)
- i\_Sig0\_3
  - L2FinalM, [52](#)
- i\_Snow\_density
  - L2FinalC, [49](#)
- i\_Snow\_depth
  - L2FinalC, [49](#)
- i\_Spare1
  - L2FinalC, [48](#)
- i\_Wet\_C
  - L2FinalC, [47](#)
- iCheckBitLong
  - Utility functions, [24](#)
- iCheckBitLongLong
  - Utility functions, [24](#)
- iCheckBitShort
  - Utility functions, [22](#)
- IDL Routines, [26](#)
- INTERP\_SAR\_NO\_POINTS
  - L2IStructure.h, [114](#)
- INTERP\_SAR\_ONE\_SIDED
  - L2IStructure.h, [114](#)
- INTERP\_SAR\_UNREL
  - L2IStructure.h, [114](#)
- j\_AGC\_ch1
  - Measurements, [77](#)
- j\_AGC\_ch2
  - Measurements, [78](#)
- j\_Alt
  - L2IData, [58](#)
  - TimeAndOrbitGroup, [83](#)
- j\_Alt\_1Hz
  - AveragedWaveformsLRM, [30](#)
  - AveragedWaveformsSAR, [32](#)
  - AveragedWaveformsSARIN, [35](#)
  - L2Data, [44](#)
- j\_Alt\_rate
  - L2IData, [58](#)
  - TimeAndOrbitGroup, [83](#)
- j\_Attitude
  - L2IData, [70](#)
- j\_Azimuth
  - L2IData, [70](#)
- j\_COR2
  - Measurements, [77](#)
- J\_CS\_L1B\_LRM\_RECORD\_SIZE
  - L1bStructure.h, [110](#)
- J\_CS\_L1B\_SAR\_RECORD\_SIZE
  - L1bStructure.h, [111](#)
- J\_CS\_L1B\_SARIN\_RECORD\_SIZE
  - L1bStructure.h, [111](#)
- J\_CS\_L2\_RECORD\_SIZE
  - CS\_Defines.c, [94](#)
  - CS\_Defines.h, [96](#)
- J\_CS\_L2I\_RECORD\_SIZE
  - CS\_Defines.c, [94](#)
  - CS\_Defines.h, [96](#)
- J\_CS\_L2I\_STRUCT\_SIZE
  - L2IStructure.h, [114](#)
- j\_Coherence
  - L2IData, [66](#)
- j\_D\_time\_mics
  - L2FinalM, [51](#)
- j\_DAC\_C
  - Corrections, [37](#)
  - L2IData, [72](#)
- j\_DEM\_elev
  - L2IData, [72](#)
- j\_Day
  - L2Data, [44](#)
  - L2IData, [57](#)
  - TimeAndOrbitGroup, [82](#)
- j\_Day\_1Hz
  - AveragedWaveformsLRM, [30](#)
  - AveragedWaveformsSAR, [32](#)
  - AveragedWaveformsSARIN, [34](#)
- j\_Discriminator
  - L2IData, [69](#)
- j\_Dopp\_rc
  - L2IData, [74](#)
  - Measurements, [78](#)
- j\_Dry\_C
  - Corrections, [37](#)
  - L2IData, [72](#)

- j\_Ext\_phase\_C
  - L2IData, [75](#)
  - Measurements, [79](#)
- j\_FAI
  - Measurements, [77](#)
- j\_Freeboard
  - L2IData, [66](#)
- j\_Geoid\_mod
  - L2IData, [71](#)
- j\_H0
  - Measurements, [77](#)
- j\_H\_GPT
  - Corrections, [38](#)
  - L2IData, [73](#)
- j\_H\_LPEOT
  - Corrections, [38](#)
  - L2IData, [73](#)
- j\_H\_OLT
  - Corrections, [38](#)
  - L2IData, [73](#)
- j\_H\_OT
  - Corrections, [38](#)
  - L2IData, [73](#)
- j\_H\_SET
  - Corrections, [38](#)
  - L2IData, [73](#)
- j\_Height
  - L2FinalM, [51](#)
  - L2IData, [60](#)
- j\_Height\_2
  - L2FinalM, [51](#)
  - L2IData, [60](#)
- j\_Height\_3
  - L2FinalM, [51](#)
  - L2IData, [60](#)
- j\_IB\_C
  - Corrections, [37](#)
  - L2IData, [72](#)
- j\_Ice\_conc
  - L2IData, [68](#)
- j\_Int\_phase\_C
  - L2IData, [75](#)
  - Measurements, [79](#)
- j\_lono\_GIM
  - Corrections, [37](#)
  - L2IData, [72](#)
- j\_lono\_mod
  - Corrections, [37](#)
  - L2IData, [73](#)
- J\_L2\_RECORD\_SIZE
  - L2Structure.h, [122](#)
- J\_L2\_STRUCT\_SIZE
  - L2Structure.h, [122](#)
- j\_LAI
  - Measurements, [77](#)
- J\_LRM\_WFM\_BINS
  - L1bStructure.h, [110](#)
- j\_Lat
  - L2FinalM, [51](#)
  - L2IData, [58](#)
  - TimeAndOrbitGroup, [83](#)
- j\_Lat\_1Hz
  - AveragedWaveformsLRM, [30](#)
  - AveragedWaveformsSAR, [32](#)
  - AveragedWaveformsSARIN, [35](#)
  - L2Data, [44](#)
- j\_Lat\_sat
  - L2IData, [71](#)
- j\_Linear\_wfm\_mult
  - AveragedWaveformsLRM, [31](#)
  - AveragedWaveformsSAR, [33](#)
  - AveragedWaveformsSARIN, [35](#)
  - WaveformsLRM, [85](#)
  - WaveformsSAR, [87](#)
  - WaveformsSARIN, [88](#)
- j\_Lon
  - L2FinalM, [51](#)
  - L2IData, [58](#)
  - TimeAndOrbitGroup, [83](#)
- j\_Lon\_1Hz
  - AveragedWaveformsLRM, [30](#)
  - AveragedWaveformsSAR, [32](#)
  - AveragedWaveformsSARIN, [35](#)
  - L2Data, [44](#)
- j\_Lon\_sat
  - L2IData, [71](#)
- J\_MPH\_SIZE
  - CS\_Defines.h, [96](#)
- j\_MSS\_Geoid
  - L2FinalC, [48](#)
- j\_MSS\_mod
  - L2IData, [71](#)
- J\_NUM\_BB\_PARAMETERS
  - L1bStructure.h, [108](#)
- J\_NUM\_HRB
  - L1bStructure.h, [108](#)
- j\_Noise\_pwr
  - L2IData, [75](#)
  - Measurements, [79](#)
- j\_ODLE
  - L2FinalC, [48](#)
- j\_ODLE\_mod
  - L2IData, [71](#)
- j\_Ocean\_ht
  - L2IData, [66](#)
- j\_Peakiness
  - L2IData, [61](#)
- j\_Phase\_slope\_C
  - L2IData, [75](#)
  - Measurements, [79](#)
- j\_Pitch\_angle
  - L2Data, [45](#)
  - L2IData, [59](#)
  - TimeAndOrbitGroup, [84](#)
- j\_Power2\_wfm\_mult
  - AveragedWaveformsLRM, [31](#)



- AveragedWaveformsSAR, 33
- AveragedWaveformsSARIN, 35
- WaveformsLRM, 85
- WaveformsSAR, 87
- WaveformsSARIN, 88
- j\_Quality\_1
  - L2FinalM, 53
- j\_Quality\_2
  - L2FinalM, 53
- j\_Quality\_3
  - L2FinalM, 53
- j\_R\_inst\_gain\_C
  - L2IData, 75
  - Measurements, 79
- j\_R\_inst\_rc
  - L2IData, 74
  - Measurements, 78
- j\_Retrk\_10
  - L2IData, 63
- j\_Retrk\_11
  - L2IData, 63
- j\_Retrk\_12
  - L2IData, 63
- j\_Retrk\_13
  - L2IData, 63
- j\_Retrk\_14
  - L2IData, 64
- j\_Retrk\_15
  - L2IData, 64
- j\_Retrk\_16
  - L2IData, 64
- j\_Retrk\_17
  - L2IData, 64
- j\_Retrk\_18
  - L2IData, 64
- j\_Retrk\_19
  - L2IData, 64
- j\_Retrk\_20
  - L2IData, 64
- j\_Retrk\_21
  - L2IData, 65
- j\_Retrk\_22
  - L2IData, 65
- j\_Retrk\_23
  - L2IData, 65
- j\_Retrk\_3
  - L2IData, 62
- j\_Retrk\_4
  - L2IData, 62
- j\_Retrk\_5
  - L2IData, 62
- j\_Retrk\_6
  - L2IData, 62
- j\_Retrk\_7
  - L2IData, 63
- j\_Retrk\_8
  - L2IData, 63
- j\_Retrk\_9
  - L2IData, 63
- j\_Retrk\_Quality\_1
  - L2IData, 62
- j\_Retrk\_Quality\_2
  - L2IData, 62
- j\_Retrk\_Quality\_3
  - L2IData, 62
- j\_Retrk\_range\_C
  - L2IData, 61
- j\_Retrk\_range\_C\_2
  - L2IData, 61
- j\_Retrk\_range\_C\_3
  - L2IData, 61
- j\_Retrk\_sig0\_C
  - L2IData, 61
- j\_Retrk\_sig0\_C\_2
  - L2IData, 61
- j\_Retrk\_sig0\_C\_3
  - L2IData, 61
- j\_Roll\_angle
  - L2Data, 45
  - L2IData, 59
  - TimeAndOrbitGroup, 84
- J\_SAR\_WFM\_BINS
  - L1bStructure.h, 110
- J\_SARIN\_WFM\_BINS
  - L1bStructure.h, 110
- j\_SARin\_disc\_1
  - L2IData, 69
- j\_SARin\_disc\_10
  - L2IData, 70
- j\_SARin\_disc\_2
  - L2IData, 69
- j\_SARin\_disc\_3
  - L2IData, 69
- j\_SARin\_disc\_4
  - L2IData, 69
- j\_SARin\_disc\_5
  - L2IData, 69
- j\_SARin\_disc\_6
  - L2IData, 69
- j\_SARin\_disc\_7
  - L2IData, 70
- j\_SARin\_disc\_8
  - L2IData, 70
- j\_SARin\_disc\_9
  - L2IData, 70
- j\_SHA
  - L2IData, 66
- j\_SSB
  - L2IData, 74
- j\_SSHA\_interp
  - L2IData, 66
- j\_SWH
  - L2IData, 60
- j\_Sig0
  - L2IData, 60
- j\_Sig0\_2

- L2IData, 60
- j\_Sig0\_3
  - L2IData, 60
- j\_Slope\_Doppler\_C
  - L2IData, 71
- j\_Snow\_density
  - L2IData, 69
- j\_Snow\_depth
  - L2IData, 68
- j\_TR\_gain\_ch1
  - Measurements, 78
- j\_TR\_gain\_ch2
  - Measurements, 78
- j\_TR\_inst\_gain\_C
  - L2IData, 75
  - Measurements, 79
- j\_TR\_inst\_rc
  - L2IData, 74
  - Measurements, 78
- j\_Tx\_power
  - Measurements, 78
- j\_USO\_Corr
  - L2IData, 57
  - TimeAndOrbitGroup, 82
- j\_Wet\_C
  - Corrections, 37
  - L2IData, 72
- j\_XTrack\_angle
  - L2IData, 65
- j\_XTrack\_angle\_C
  - L2IData, 66
- j\_Yaw\_angle
  - L2Data, 45
  - L2IData, 59
  - TimeAndOrbitGroup, 84
- j\_baseline
  - t\_cs\_fileinfo, 81
- j\_echo\_shape
  - L2IData, 65
- j\_num\_datasets
  - t\_cs\_fileinfo, 80
- j\_type
  - t\_cs\_fileinfo, 81
- jCSGetDSFileOffsetFromHeader
  - High-level shared routines, 10
- jCSGetDSFileOffsetFromHeader.c, 97
- jCheckDSDetails, 97
- jCSGetDSFileOffsetFromHeader.h, 98
- jCSNumRecordsInDataset
  - High-level shared routines, 10
- jCSNumRecordsInDataset.c, 99
- jCSNumRecordsInDataset.h, 99
- jCheckDSDetails
  - jCSGetDSFileOffsetFromHeader.c, 97
- jFillL1bLRMStructFromFile
  - Low-Level CryoSat Level 1b File IO API, 15
- jFillL1bSARINStructFromFile
  - Low-Level CryoSat Level 1b File IO API, 16
- jFillL1bSARStructFromFile
  - Low-Level CryoSat Level 1b File IO API, 15
- jFillL1bStructFromFile.c, 100
- jFillL1bStructFromFile.h, 101
- jFillL2IStructFromFile
  - Low-Level CryoSat Level 2 File IO API, 20
- jFillL2IStructFromFile.c, 102
- jFillL2IStructFromFile.h, 103
- jFillL2StructFromFile
  - Low-Level CryoSat Level 2 File IO API, 20
- jFillL2StructFromFile.c, 104
- jFillL2StructFromFile.h, 105
- k\_TD
  - Measurements, 77
- k\_TD\_1Hz
  - AveragedWaveformsLRM, 30
  - AveragedWaveformsSAR, 32
  - AveragedWaveformsSARIN, 35
- L1B\_LRM, 39
  - at\_Measurements, 40
  - at\_Time\_Orbit, 40
  - at\_Waveforms, 40
  - t\_Average\_PL, 40
  - t\_Geo\_Corrections, 40
- L1B\_SAR, 40
  - at\_Measurements, 41
  - at\_Time\_Orbit, 41
  - at\_Waveforms, 41
  - t\_Average\_PL, 41
  - t\_Geo\_Corrections, 41
- L1B\_SARIN, 41
  - at\_Measurements, 42
  - at\_Time\_Orbit, 42
  - at\_Waveforms, 42
  - t\_Average\_PL, 42
  - t\_Geo\_Corrections, 42
- L1bStructure.h, 106
  - J\_CS\_L1B\_LRM\_RECORD\_SIZE, 110
  - J\_CS\_L1B\_SAR\_RECORD\_SIZE, 111
  - J\_CS\_L1B\_SARIN\_RECORD\_SIZE, 111
  - J\_LRM\_WFM\_BINS, 110
  - J\_NUM\_BB\_PARAMETERS, 108
  - J\_NUM\_HRB, 108
  - J\_SAR\_WFM\_BINS, 110
  - J\_SARIN\_WFM\_BINS, 110
  - t\_Level1bAvgWfmLRMFlagBit, 108
  - t\_Level1bAvgWfmSARFlagBit, 109
  - t\_Level1bAvgWfmSARinFlagBit, 108
  - t\_Level1bWfmLRMFlagBit, 109
  - t\_Level1bWfmSARFlagBit, 109
  - t\_Level1bWfmSARinFlagBit, 108
  - vDump\_L1B\_LRM, 110
  - vDump\_L1B\_SAR, 110
  - vDump\_L1B\_SARIN, 110
  - WFM\_AA\_POWER\_ECHOES, 109
  - WFM\_APPROX\_BEAM\_STEER, 109
  - WFM\_AUTO\_STEER, 109

- WFM\_DOPPLER\_APPLY, 109
- WFM\_DOPPLER\_COMP, 109
- WFM\_EXACT\_BEAM\_STEER, 109
- WFM\_LRM\_ERROR, 109
- WFM\_LRM\_TRK\_0, 109
- WFM\_LRM\_TRK\_1, 109
- WFM\_LRM\_TRK\_2, 109
- WFM\_MULTI\_LOOK\_INCOMPLETE, 109
- WFM\_SAR\_ERROR, 109
- WFM\_SAR\_MISPOINT, 109
- WFM\_STEER\_ERROR, 109
- L2C\_DAC
  - L2Structure.h, 123
- L2C\_DRY\_TROPO
  - L2Structure.h, 123
- L2C\_GPT
  - L2Structure.h, 123
- L2C\_ICE\_CONC
  - L2Structure.h, 123
- L2C\_INV\_BARO
  - L2Structure.h, 123
- L2C\_IONO
  - L2Structure.h, 123
- L2C\_LPEOT
  - L2Structure.h, 123
- L2C\_MSS\_GEOID
  - L2Structure.h, 123
- L2C\_OCEAN\_TIDE
  - L2Structure.h, 123
- L2C\_ODLE\_MODEL
  - L2Structure.h, 123
- L2C\_OLT
  - L2Structure.h, 123
- L2C\_SET
  - L2Structure.h, 123
- L2C\_SNOW\_DENSITY
  - L2Structure.h, 123
- L2C\_SNOW\_DEPTH
  - L2Structure.h, 123
- L2C\_SSB
  - L2Structure.h, 123
- L2C\_SURFACE\_TYPE
  - L2Structure.h, 123
- L2C\_SWH
  - L2Structure.h, 123
- L2C\_WET\_TROPO
  - L2Structure.h, 123
- L2C\_WINDSPEED
  - L2Structure.h, 123
- L2Data, 43
  - at\_Meas, 45
  - i\_Mispointing, 44
  - i\_N\_Valid, 45
  - j\_Alt\_1Hz, 44
  - j\_Day, 44
  - j\_Lat\_1Hz, 44
  - j\_Lon\_1Hz, 44
  - j\_Pitch\_angle, 45
  - j\_Roll\_angle, 45
  - j\_Yaw\_angle, 45
  - t\_Corr, 45
  - uj\_Micsec, 44
  - uj\_Sec, 44
  - uk\_Siral\_mode, 44
- L2FinalC, 46
  - i\_DAC\_C, 47
  - i\_Dry\_C, 47
  - i\_H\_GPT, 48
  - i\_H\_LPEOT, 48
  - i\_H\_OLT, 48
  - i\_H\_OT, 47
  - i\_H\_SET, 48
  - i\_IB\_C, 47
  - i\_Ice\_conc, 49
  - i\_Iono\_C, 47
  - i\_SSB, 47
  - i\_SWH, 49
  - i\_Snow\_density, 49
  - i\_Snow\_depth, 49
  - i\_Spare1, 48
  - i\_Wet\_C, 47
  - j\_MSS\_Geoid, 48
  - j\_ODLE, 48
  - ui\_Wind\_speed, 49
  - uj\_C\_status, 49
  - uk\_Surf\_type, 48
- L2FinalM, 50
  - i\_Freeboard, 52
  - i\_N\_avg, 52
  - i\_Peakiness, 52
  - i\_SSHA\_interp, 51
  - i\_SSHA\_num, 51
  - i\_SSHA\_qual, 52
  - i\_Sig0, 52
  - i\_Sig0\_2, 52
  - i\_Sig0\_3, 52
  - j\_D\_time\_mics, 51
  - j\_Height, 51
  - j\_Height\_2, 51
  - j\_Height\_3, 51
  - j\_Lat, 51
  - j\_Lon, 51
  - j\_Quality\_1, 53
  - j\_Quality\_2, 53
  - j\_Quality\_3, 53
  - uj\_C\_applied, 53
  - uj\_Quality\_F, 53
- L2IData, 54
  - ai\_bb\_param, 65
  - aj\_Baseline, 59
  - aj\_Real\_beam, 58
  - aj\_Sat\_velocity, 58
  - auc\_spares1, 68
  - auc\_spares2, 72
  - auc\_spares3, 74
  - auc\_spares4, 75

i\_ST\_ID, 59  
 j\_Alt, 58  
 j\_Alt\_rate, 58  
 j\_Attitude, 70  
 j\_Azimuth, 70  
 j\_Coherence, 66  
 j\_DAC\_C, 72  
 j\_DEM\_elev, 72  
 j\_Day, 57  
 j\_Discriminator, 69  
 j\_Dopp\_rc, 74  
 j\_Dry\_C, 72  
 j\_Ext\_phase\_C, 75  
 j\_Freeboard, 66  
 j\_Geoid\_mod, 71  
 j\_H\_GPT, 73  
 j\_H\_LPEOT, 73  
 j\_H\_OLT, 73  
 j\_H\_OT, 73  
 j\_H\_SET, 73  
 j\_Height, 60  
 j\_Height\_2, 60  
 j\_Height\_3, 60  
 j\_IB\_C, 72  
 j\_Ice\_conc, 68  
 j\_Int\_phase\_C, 75  
 j\_Iono\_GIM, 72  
 j\_Iono\_mod, 73  
 j\_Lat, 58  
 j\_Lat\_sat, 71  
 j\_Lon, 58  
 j\_Lon\_sat, 71  
 j\_MSS\_mod, 71  
 j\_Noise\_pwr, 75  
 j\_ODLE\_mod, 71  
 j\_Ocean\_ht, 66  
 j\_Peakiness, 61  
 j\_Phase\_slope\_C, 75  
 j\_Pitch\_angle, 59  
 j\_R\_inst\_gain\_C, 75  
 j\_R\_inst\_rc, 74  
 j\_Retrk\_10, 63  
 j\_Retrk\_11, 63  
 j\_Retrk\_12, 63  
 j\_Retrk\_13, 63  
 j\_Retrk\_14, 64  
 j\_Retrk\_15, 64  
 j\_Retrk\_16, 64  
 j\_Retrk\_17, 64  
 j\_Retrk\_18, 64  
 j\_Retrk\_19, 64  
 j\_Retrk\_20, 64  
 j\_Retrk\_21, 65  
 j\_Retrk\_22, 65  
 j\_Retrk\_23, 65  
 j\_Retrk\_3, 62  
 j\_Retrk\_4, 62  
 j\_Retrk\_5, 62  
 j\_Retrk\_6, 62  
 j\_Retrk\_7, 63  
 j\_Retrk\_8, 63  
 j\_Retrk\_9, 63  
 j\_Retrk\_Quality\_1, 62  
 j\_Retrk\_Quality\_2, 62  
 j\_Retrk\_Quality\_3, 62  
 j\_Retrk\_range\_C, 61  
 j\_Retrk\_range\_C\_2, 61  
 j\_Retrk\_range\_C\_3, 61  
 j\_Retrk\_sig0\_C, 61  
 j\_Retrk\_sig0\_C\_2, 61  
 j\_Retrk\_sig0\_C\_3, 61  
 j\_Roll\_angle, 59  
 j\_SARin\_disc\_1, 69  
 j\_SARin\_disc\_10, 70  
 j\_SARin\_disc\_2, 69  
 j\_SARin\_disc\_3, 69  
 j\_SARin\_disc\_4, 69  
 j\_SARin\_disc\_5, 69  
 j\_SARin\_disc\_6, 69  
 j\_SARin\_disc\_7, 70  
 j\_SARin\_disc\_8, 70  
 j\_SARin\_disc\_9, 70  
 j\_SHA, 66  
 j\_SSB, 74  
 j\_SSHA\_interp, 66  
 j\_SWH, 60  
 j\_Sig0, 60  
 j\_Sig0\_2, 60  
 j\_Sig0\_3, 60  
 j\_Slope\_Doppler\_C, 71  
 j\_Snow\_density, 69  
 j\_Snow\_depth, 68  
 j\_TR\_inst\_gain\_C, 75  
 j\_TR\_inst\_rc, 74  
 j\_USO\_Corr, 57  
 j\_Wet\_C, 72  
 j\_XTrack\_angle, 65  
 j\_XTrack\_angle\_C, 66  
 j\_Yaw\_angle, 59  
 j\_echo\_shape, 65  
 ui\_Mode\_ID, 57  
 ui\_SSC, 57  
 ui\_Wind\_speed, 68  
 ui\_interp\_cnt\_bkwd, 67  
 ui\_interp\_cnt\_fwd, 67  
 ui\_interp\_err, 66  
 ui\_interp\_error\_F, 67  
 ui\_interp\_time\_bkwd, 67  
 ui\_interp\_time\_fwd, 67  
 ui\_n\_avg, 68  
 uj\_Ambiguity, 71  
 uj\_Corr\_error, 74  
 uj\_Corr\_status, 74  
 uj\_DEM\_id, 72  
 uj\_Discrim\_F, 70  
 uj\_Freeb\_status, 68

- uj\_Ht\_status, 68
- uj\_Inst\_config, 57
- uj\_L2\_MCD, 59
- uj\_Meas\_Mode, 67
- uj\_Micsec, 57
- uj\_Quality\_F, 67
- uj\_Rec\_Count, 58
- uj\_Retracker\_F, 67
- uj\_Sec, 57
- uj\_Surf\_type, 73
- L2IStructure.c, 111
  - vDump\_L2IData, 111
- L2IStructure.h, 112
  - AMB\_FAIL, 118
  - AMB\_SARIN\_BAD\_MATH, 118
  - AMB\_SARIN\_DIFF\_ELEV, 118
  - AMB\_SARIN\_NO\_DEM, 118
  - AMB\_SARIN\_NO\_RTRK, 118
  - CL2\_LRM\_IS\_LAND\_ICE\_PLATEAU, 117
  - CL2\_LRM\_IS\_OCEAN, 117
  - CL2\_LRM\_IS\_UNDEFINED, 117
  - CL2\_SAR\_IS\_LEADS, 117
  - CL2\_SAR\_IS\_OCEAN, 117
  - CL2\_SAR\_IS\_SEA\_ICE, 117
  - CL2\_SAR\_IS\_UNDEFINED, 117
  - CL2\_SARIN\_IS\_ALTIMETER\_MODE, 117
  - CL2\_SARIN\_IS\_UNDEFINED, 117
  - DISC\_FAIL, 117
  - DISC\_SAR\_BAD\_BB, 117
  - DISC\_SAR\_BAD\_NOISE, 117
  - DISC\_SAR\_BAD\_WIDTH, 117
  - DISC\_SAR\_HI\_PK, 117
  - DISC\_SAR\_ICECONC\_UNAV, 117
  - DISC\_SAR\_ICECONC\_UNREAL, 117
  - DISC\_SAR\_LO\_PK, 117
  - DISC\_SAR\_LO\_POWER, 117
  - DISC\_SAR\_MULTIPLE\_MATCH, 117
  - DISC\_SAR\_NO\_MATCH, 117
  - DISC\_SARIN\_BAD\_LE, 117
  - DISC\_SARIN\_HI\_NOISE, 117
  - DISC\_SARIN\_HI\_PEAKINESS, 117
  - DISC\_SARIN\_LO\_VARIANCE, 117
  - DISC\_SARIN\_LOW\_PEAKINESS, 117
  - DISC\_SARIN\_LOW\_POWER, 117
  - DISC\_TBD, 117
  - FREB\_SAR\_AVAIL, 116
  - FREB\_SAR\_IN\_NORTH\_BOUND, 116
  - FREB\_SAR\_IN\_SOUTH\_BOUND, 116
  - FREB\_SAR\_UNREL, 116
  - HGT\_DAC, 116
  - HGT\_DRY\_TROPO, 116
  - HGT\_FAIL, 116
  - HGT\_GPT, 116
  - HGT\_INTERNAL, 115
  - HGT\_INV\_BARO, 116
  - HGT\_IONO\_GIM, 116
  - HGT\_IONO\_MODEL, 116
  - HGT\_LPEOT, 116
  - HGT\_LRM\_ICEBIAS, 116
  - HGT\_LRM\_OCEANBIAS, 116
  - HGT\_LRM\_RETRK, 116
  - HGT\_OLT, 116
  - HGT\_OT, 116
  - HGT\_RAD\_DOPPLER, 115
  - HGT\_SAR\_ICEBIAS, 116
  - HGT\_SAR\_OCEANBIAS, 116
  - HGT\_SAR\_RETRK, 116
  - HGT\_SARIN\_BAD\_BASELINE, 116
  - HGT\_SARIN\_BAD\_VELOCITY, 116
  - HGT\_SARIN\_ICEBIAS, 116
  - HGT\_SARIN\_OCEANBIAS, 116
  - HGT\_SARIN\_OUT\_OF\_RANGE, 116
  - HGT\_SARIN\_RETRK, 116
  - HGT\_SET, 116
  - HGT\_SLOPE\_DOPPLER, 116
  - HGT\_SSB, 116
  - HGT\_WET\_TROPO, 116
  - HGT\_WINDOW\_OFFSET, 116
  - INTERP\_SAR\_NO\_POINTS, 114
  - INTERP\_SAR\_ONE\_SIDED, 114
  - INTERP\_SAR\_UNREL, 114
  - J\_CS\_L2I\_STRUCT\_SIZE, 114
  - LRM\_SLOPE\_DATA\_VALID, 116
  - MODE\_LRM, 114
  - MODE\_SAR, 114
  - MODE\_SARIN, 114
  - MQ\_ALTWIND\_ERR, 115
  - MQ\_ARITHMETIC\_ERR, 115
  - MQ\_COHERENCE\_ERR, 115
  - MQ\_ECHO\_SHAPE\_ERR, 115
  - MQ\_HEIGHT\_1\_ERR, 114
  - MQ\_HEIGHT\_2\_ERR, 114
  - MQ\_HEIGHT\_3\_ERR, 114
  - MQ\_PK\_ERR, 115
  - MQ\_SIGMA0\_1\_ERR, 114
  - MQ\_SIGMA0\_2\_ERR, 114
  - MQ\_SIGMA0\_3\_ERR, 114
  - MQ\_SWH\_ERR, 115
  - MQ\_XTRACK\_ERR, 115
  - RTRK\_1\_FAIL, 115
  - RTRK\_2\_FAIL, 115
  - RTRK\_3\_FAIL, 115
  - RTRK\_BAD\_BB, 115
  - RTRK\_BAD\_U10, 115
  - RTRK\_FAIL, 115
  - RTRK\_FDM\_OCOG\_FAIL, 115
  - RTRK\_FIT\_FAILED, 115
  - RTRK\_HI\_NOISE, 115
  - RTRK\_HI\_PK, 115
  - RTRK\_LE\_BAD, 115
  - RTRK\_LO\_PK, 115
  - RTRK\_LO\_PWR, 115
  - RTRK\_LO\_VAR, 115
  - RTRK\_OUT\_OF\_RNGE, 115
  - RTRK\_POOR\_FIT, 115
  - RTRK\_POOR\_PHASE\_FIT, 115

- RTRK\_SARIN\_INTERPOL, 115
- RTRK\_SARIN\_LO\_COHERENCE, 115
- RTRK\_SPARE, 115
- RTRK\_SPARE1, 115
- RTRK\_SPARE2, 115
- t\_DiscriminatorResult, 116
- t\_DiscriminatorStatusFlagBit, 117
- t\_FreeboardStatusFlagBit, 116
- t\_HeightStatusFlagBit, 115
- t\_InterpolationErrorFlagBit, 114
- t\_Level2AmbiguityErrorFlagBit, 117
- t\_MeasurementMode, 114
- t\_MeasurementQualityFlagBit, 114
- t\_RetrackerFlagBit, 115
- vDump\_L2IData, 118
- L2Q\_BACKSCATTER\_1\_ERROR
  - L2Structure.h, 122
- L2Q\_BACKSCATTER\_2\_ERROR
  - L2Structure.h, 122
- L2Q\_BACKSCATTER\_3\_ERROR
  - L2Structure.h, 122
- L2Q\_BLOCK\_DEGRADED
  - L2Structure.h, 122
- L2Q\_CALIBRATION\_WARNING
  - L2Structure.h, 123
- L2Q\_DELTA\_TIME\_ERROR
  - L2Structure.h, 123
- L2Q\_FREEBOARD\_ERROR
  - L2Structure.h, 123
- L2Q\_HEIGHT\_1\_ERROR
  - L2Structure.h, 122
- L2Q\_HEIGHT\_2\_ERROR
  - L2Structure.h, 122
- L2Q\_HEIGHT\_3\_ERROR
  - L2Structure.h, 122
- L2Q\_LRM\_SLOPE\_INVALID
  - L2Structure.h, 123
- L2Q\_MISPOINTING\_ERROR
  - L2Structure.h, 123
- L2Q\_ORBIT\_DISCONTINUITY
  - L2Structure.h, 122
- L2Q\_ORBIT\_ERROR
  - L2Structure.h, 122
- L2Q\_PEAKINESS\_ERROR
  - L2Structure.h, 122
- L2Q\_SAR\_DISCRIM\_ICE
  - L2Structure.h, 123
- L2Q\_SAR\_DISCRIM\_LEAD
  - L2Structure.h, 123
- L2Q\_SAR\_DISCRIM\_OCEAN
  - L2Structure.h, 123
- L2Q\_SAR\_DISCRIM\_UNKNOWN
  - L2Structure.h, 123
- L2Q\_SARIN\_BASELINE\_ERROR
  - L2Structure.h, 123
- L2Q\_SARIN\_RANGE\_ERROR
  - L2Structure.h, 123
- L2Q\_SARIN\_RX1\_ERROR
  - L2Structure.h, 123
- L2Q\_SARIN\_RX2\_ERROR
  - L2Structure.h, 123
- L2Q\_SARIN\_SIDE\_B
  - L2Structure.h, 123
- L2Q\_SARIN\_VELOCITY\_ERROR
  - L2Structure.h, 123
- L2Q\_SARIN\_XTRACK\_ERROR
  - L2Structure.h, 123
- L2Q\_SSHA\_INTERP\_ERROR
  - L2Structure.h, 122
- L2Q\_SURFACE\_MOD
  - L2Structure.h, 123
- L2Structure.c, 118
  - vDump\_L2Data, 119
  - vDump\_L2FinalC, 119
  - vDump\_L2FinalM, 119
- L2Structure.h, 120
  - HGT\_DAC\_PDS, 124
  - HGT\_DRY\_TROPO\_PDS, 124
  - HGT\_GPT\_PDS, 124
  - HGT\_INTERNAL\_PDS, 124
  - HGT\_INV\_BARO\_PDS, 124
  - HGT\_IONO\_GIM\_PDS, 124
  - HGT\_IONO\_MODEL\_PDS, 124
  - HGT\_LPEOT\_PDS, 124
  - HGT\_LRM\_ICEBIAS\_PDS, 124
  - HGT\_LRM\_OCEANBIAS\_PDS, 124
  - HGT\_LRM\_RETRK\_PDS, 124
  - HGT\_OLT\_PDS, 124
  - HGT\_OT\_PDS, 124
  - HGT\_RAD\_DOPPLER\_PDS, 124
  - HGT\_SAR\_ICEBIAS\_PDS, 124
  - HGT\_SAR\_OCEANBIAS\_PDS, 124
  - HGT\_SAR\_RETRK\_PDS, 124
  - HGT\_SARIN\_BAD\_BASELINE\_PDS, 124
  - HGT\_SARIN\_BAD\_VELOCITY\_PDS, 124
  - HGT\_SARIN\_ICEBIAS\_PDS, 124
  - HGT\_SARIN\_OCEANBIAS\_PDS, 124
  - HGT\_SARIN\_OUT\_OF\_RANGE\_PDS, 124
  - HGT\_SARIN\_RETRK\_PDS, 124
  - HGT\_SET\_PDS, 124
  - HGT\_SLOPE\_DOPPLER\_PDS, 124
  - HGT\_SSB\_PDS, 124
  - HGT\_WET\_TROPO\_PDS, 124
  - HGT\_WINDOW\_OFFSET\_PDS, 124
- J\_L2\_RECORD\_SIZE, 122
- J\_L2\_STRUCT\_SIZE, 122
- L2C\_DAC, 123
- L2C\_DRY\_TROPO, 123
- L2C\_GPT, 123
- L2C\_ICE\_CONC, 123
- L2C\_INV\_BARO, 123
- L2C\_IONO, 123
- L2C\_LPEOT, 123
- L2C\_MSS\_GEOID, 123
- L2C\_OCEAN\_TIDE, 123
- L2C\_ODLE\_MODEL, 123

- L2C\_OLT, [123](#)
- L2C\_SET, [123](#)
- L2C\_SNOW\_DENSITY, [123](#)
- L2C\_SNOW\_DEPTH, [123](#)
- L2C\_SSB, [123](#)
- L2C\_SURFACE\_TYPE, [123](#)
- L2C\_SWH, [123](#)
- L2C\_WET\_TROPO, [123](#)
- L2C\_WINDSPEED, [123](#)
- L2Q\_BACKSCATTER\_1\_ERROR, [122](#)
- L2Q\_BACKSCATTER\_2\_ERROR, [122](#)
- L2Q\_BACKSCATTER\_3\_ERROR, [122](#)
- L2Q\_BLOCK\_DEGRADED, [122](#)
- L2Q\_CALIBRATION\_WARNING, [123](#)
- L2Q\_DELTA\_TIME\_ERROR, [123](#)
- L2Q\_FREEBOARD\_ERROR, [123](#)
- L2Q\_HEIGHT\_1\_ERROR, [122](#)
- L2Q\_HEIGHT\_2\_ERROR, [122](#)
- L2Q\_HEIGHT\_3\_ERROR, [122](#)
- L2Q\_LRM\_SLOPE\_INVALID, [123](#)
- L2Q\_MISPOINTING\_ERROR, [123](#)
- L2Q\_ORBIT\_DISCONTINUITY, [122](#)
- L2Q\_ORBIT\_ERROR, [122](#)
- L2Q\_PEAKINESS\_ERROR, [122](#)
- L2Q\_SAR\_DISCRIM\_ICE, [123](#)
- L2Q\_SAR\_DISCRIM\_LEAD, [123](#)
- L2Q\_SAR\_DISCRIM\_OCEAN, [123](#)
- L2Q\_SAR\_DISCRIM\_UNKNOWN, [123](#)
- L2Q\_SARIN\_BASELINE\_ERROR, [123](#)
- L2Q\_SARIN\_RANGE\_ERROR, [123](#)
- L2Q\_SARIN\_RX1\_ERROR, [123](#)
- L2Q\_SARIN\_RX2\_ERROR, [123](#)
- L2Q\_SARIN\_SIDE\_B, [123](#)
- L2Q\_SARIN\_VELOCITY\_ERROR, [123](#)
- L2Q\_SARIN\_XTRACK\_ERROR, [123](#)
- L2Q\_SSHA\_INTERP\_ERROR, [122](#)
- L2Q\_SURFACE\_MOD, [123](#)
- LRM\_SLOPE\_DATA\_VALID\_PDS, [124](#)
- t\_Level2CorrectionsAppliedFlagBit, [123](#)
- t\_Level2CorrectionsStatusFlagBit, [123](#)
- t\_Level2QualityStatusFlagBit, [122](#)
- vDump\_L2Data, [124](#)
- LOOP\_STATUS
  - SharedFlags.h, [136](#)
- LOSS\_OF\_ECHO
  - SharedFlags.h, [136](#)
- LRM\_SLOPE\_DATA\_VALID
  - L2IStructure.h, [116](#)
- LRM\_SLOPE\_DATA\_VALID\_PDS
  - L2Structure.h, [124](#)
- Low-Level CryoSat Level 1b File IO API, [15](#)
  - jFillL1bLRMStructFromFile, [15](#)
  - jFillL1bSARINStructFromFile, [16](#)
  - jFillL1bSARStructFromFile, [15](#)
- Low-Level CryoSat Level 2 File IO API, [20](#)
  - jFillL2IStructFromFile, [20](#)
  - jFillL2StructFromFile, [20](#)
- MCD\_AGC\_INCONSISTANT
  - SharedFlags.h, [135](#)
- MCD\_ATT\_CORR\_MISSING
  - SharedFlags.h, [135](#)
- MCD\_BLOCK\_BLANK
  - SharedFlags.h, [134](#)
- MCD\_BLOCK\_DEGRADED
  - SharedFlags.h, [134](#)
- MCD\_C\_CAL\_1\_IPF\_DB\_USED
  - SharedFlags.h, [135](#)
- MCD\_CAL1\_TYPE
  - SharedFlags.h, [135](#)
- MCD\_CAL\_1\_CORR\_MISSING
  - SharedFlags.h, [135](#)
- MCD\_CAL\_1\_IPF\_DB\_USED
  - SharedFlags.h, [135](#)
- MCD\_CAL\_2\_CORR\_IPF\_DB
  - SharedFlags.h, [135](#)
- MCD\_CAL\_2\_CORR\_MISSING
  - SharedFlags.h, [135](#)
- MCD\_CH1\_ERROR
  - SharedFlags.h, [135](#)
- MCD\_CH2\_ERROR
  - SharedFlags.h, [135](#)
- MCD\_DATATION\_DEGRADED
  - SharedFlags.h, [134](#)
- MCD\_DORIS\_USO\_CORR\_MISSING
  - SharedFlags.h, [135](#)
- MCD\_ECHO\_ERROR
  - SharedFlags.h, [135](#)
- MCD\_ECHO\_RX1\_ERROR
  - SharedFlags.h, [135](#)
- MCD\_ECHO\_RX2\_ERROR
  - SharedFlags.h, [135](#)
- MCD\_ECHO\_SATURATION
  - SharedFlags.h, [135](#)
- MCD\_NPM\_INCONSISTANT
  - SharedFlags.h, [135](#)
- MCD\_ORBIT\_DISCONTINUITY
  - SharedFlags.h, [134](#)
- MCD\_ORBIT\_FILE\_CHANGE
  - SharedFlags.h, [134](#)
- MCD\_ORBIT\_PROP\_ERROR
  - SharedFlags.h, [134](#)
- MCD\_PHASE\_PETURBATION\_MODE
  - SharedFlags.h, [135](#)
- MCD\_POWER\_SCALE\_ERROR
  - SharedFlags.h, [135](#)
- MCD\_TRK\_ECHO\_ERROR
  - SharedFlags.h, [135](#)
- MCD\_WINDOW\_DELAY\_INCONSISTANT
  - SharedFlags.h, [135](#)
- MODE\_ID\_0
  - SharedFlags.h, [136](#)
- MODE\_ID\_1
  - SharedFlags.h, [136](#)
- MODE\_ID\_2
  - SharedFlags.h, [136](#)
- MODE\_ID\_3



- SharedFlags.h, 136
- MODE\_ID\_4
  - SharedFlags.h, 136
- MODE\_ID\_5
  - SharedFlags.h, 136
- MODE\_LRM
  - L2IStructure.h, 114
- MODE\_SAR
  - L2IStructure.h, 114
- MODE\_SARIN
  - L2IStructure.h, 114
- MQ\_ALT\_WIND\_ERR
  - L2IStructure.h, 115
- MQ\_ARITHMETIC\_ERR
  - L2IStructure.h, 115
- MQ\_COHERENCE\_ERR
  - L2IStructure.h, 115
- MQ\_ECHO\_SHAPE\_ERR
  - L2IStructure.h, 115
- MQ\_HEIGHT\_1\_ERR
  - L2IStructure.h, 114
- MQ\_HEIGHT\_2\_ERR
  - L2IStructure.h, 114
- MQ\_HEIGHT\_3\_ERR
  - L2IStructure.h, 114
- MQ\_PK\_ERR
  - L2IStructure.h, 115
- MQ\_SIGMA0\_1\_ERR
  - L2IStructure.h, 114
- MQ\_SIGMA0\_2\_ERR
  - L2IStructure.h, 114
- MQ\_SIGMA0\_3\_ERR
  - L2IStructure.h, 114
- MQ\_SWH\_ERR
  - L2IStructure.h, 115
- MQ\_XTRACK\_ERR
  - L2IStructure.h, 115
- Measurements, 76
  - j\_AGC\_ch1, 77
  - j\_AGC\_ch2, 78
  - j\_COR2, 77
  - j\_Dopp\_rc, 78
  - j\_Ext\_phase\_C, 79
  - j\_FAI, 77
  - j\_H0, 77
  - j\_Int\_phase\_C, 79
  - j\_LAI, 77
  - j\_Noise\_pwr, 79
  - j\_Phase\_slope\_C, 79
  - j\_R\_inst\_gain\_C, 79
  - j\_R\_inst\_rc, 78
  - j\_TR\_gain\_ch1, 78
  - j\_TR\_gain\_ch2, 78
  - j\_TR\_inst\_gain\_C, 79
  - j\_TR\_inst\_rc, 78
  - j\_Tx\_power, 78
  - k\_TD, 77
  - spares, 79
- PLAT\_ATT\_0
  - SharedFlags.h, 137
- PLAT\_ATT\_1
  - SharedFlags.h, 137
- paj\_num\_records
  - t\_cs\_fileinfo, 80
- pat\_data\_start\_offsets
  - t\_cs\_fileinfo, 80
- pj\_phase\_diff
  - WaveformsSARIN, 89
- pt\_filepointer
  - t\_cs\_fileinfo, 80
- ptCSGetFileHandle
  - High-level shared routines, 9
- ptCSGetFileHandle.c, 125
- ptCSGetFileHandle.h, 126
  - BASELINE, 127
  - BASELINE\_A, 127
  - BASELINE\_B, 127
  - BASELINE\_BC, 127
  - BASELINE\_C, 127
  - BASELINE\_UNKNOWN, 127
  - t\_cs\_filehandle, 127
- ptCSGetL1b.c, 127
- ptCSGetL1b.h, 128
- ptCSGetL1bLRM
  - High Level CryoSat Level 1b File IO API, 12
- ptCSGetL1bSAR
  - High Level CryoSat Level 1b File IO API, 13
- ptCSGetL1bSARIN
  - High Level CryoSat Level 1b File IO API, 13
- ptCSGetL2
  - High Level CryoSat Level 2 File IO API, 17
- ptCSGetL2.c, 129
- ptCSGetL2.h, 130
- ptCSGetL2I
  - High Level CryoSat Level 2 File IO API, 18
- ptCSGetL2I.c, 131
- ptCSGetL2I.h, 132
- pui\_avg\_wfm
  - AveragedWaveformsLRM, 30
  - AveragedWaveformsSAR, 33
  - AveragedWaveformsSARIN, 35
- pui\_coherence
  - WaveformsSARIN, 89
- pui\_wfm
  - WaveformsLRM, 85
  - WaveformsSAR, 87
  - WaveformsSARIN, 88
- RT\_ERROR
  - SharedFlags.h, 136
- RTRK\_1\_FAIL
  - L2IStructure.h, 115
- RTRK\_2\_FAIL
  - L2IStructure.h, 115
- RTRK\_3\_FAIL
  - L2IStructure.h, 115
- RTRK\_BAD\_BB



L2IStructure.h, 115  
RTRK\_BAD\_U10  
    L2IStructure.h, 115  
RTRK\_FAIL  
    L2IStructure.h, 115  
RTRK\_FDM\_OCOG\_FAIL  
    L2IStructure.h, 115  
RTRK\_FIT\_FAILED  
    L2IStructure.h, 115  
RTRK\_HI\_NOISE  
    L2IStructure.h, 115  
RTRK\_HI\_PK  
    L2IStructure.h, 115  
RTRK\_LE\_BAD  
    L2IStructure.h, 115  
RTRK\_LO\_PK  
    L2IStructure.h, 115  
RTRK\_LO\_PWR  
    L2IStructure.h, 115  
RTRK\_LO\_VAR  
    L2IStructure.h, 115  
RTRK\_OUT\_OF\_RNGE  
    L2IStructure.h, 115  
RTRK\_POOR\_FIT  
    L2IStructure.h, 115  
RTRK\_POOR\_PHASE\_FIT  
    L2IStructure.h, 115  
RTRK\_SARIN\_INTERPOL  
    L2IStructure.h, 115  
RTRK\_SARIN\_LO\_COHERENCE  
    L2IStructure.h, 115  
RTRK\_SPARE  
    L2IStructure.h, 115  
RTRK\_SPARE1  
    L2IStructure.h, 115  
RTRK\_SPARE2  
    L2IStructure.h, 115  
RX\_BAND\_ATT  
    SharedFlags.h, 136  
RX\_CHAIN\_0  
    SharedFlags.h, 136  
RX\_CHAIN\_1  
    SharedFlags.h, 136  
  
SIN\_DEGRADED  
    SharedFlags.h, 136  
SIRAL\_ID  
    SharedFlags.h, 136  
ST\_ATTREF  
    SharedFlags.h, 137  
ST\_NONE  
    SharedFlags.h, 137  
STR\_ATTREF\_USED  
    SharedFlags.h, 136  
SURFACE\_CLOSED\_SEA  
    SharedFlags.h, 137  
SURFACE\_CONTINENTAL\_ICE  
    SharedFlags.h, 137  
SURFACE\_LAND  
    SharedFlags.h, 137  
SURFACE\_OPEN\_OCEAN  
    SharedFlags.h, 137  
SharedFlags.h, 133  
    BANDWIDTH\_0, 136  
    BANDWIDTH\_1, 136  
    CAL4\_MODE, 137  
    COR\_DAC, 137  
    COR\_DEM\_MODEL, 138  
    COR\_DRY\_TROPO, 137  
    COR\_GEOID\_MODEL, 138  
    COR\_GPT, 137  
    COR\_ICE\_CONC, 138  
    COR\_INV\_BARO, 137  
    COR\_IONO\_GIM, 137  
    COR\_IONO\_MODEL, 137  
    COR\_LPEOT, 137  
    COR\_MSS\_MODEL, 138  
    COR\_ODLE\_MODEL, 138  
    COR\_OLT, 137  
    COR\_OT, 137  
    COR\_SET, 137  
    COR\_SLOPE\_MODEL, 138  
    COR\_SNOW\_DENSITY, 138  
    COR\_SNOW\_DEPTH, 138  
    COR\_SSB, 138  
    COR\_SURFACE\_TYPE, 137  
    COR\_WET\_TROPO, 137  
    CYCLE\_ERROR, 136  
    ECHO\_SATURATION, 136  
    ERROR\_DAC, 138  
    ERROR\_DEM\_MODEL, 138  
    ERROR\_DRY\_TROPO, 138  
    ERROR\_GEOID\_MODEL, 138  
    ERROR\_GPT, 138  
    ERROR\_ICE\_CONC, 138  
    ERROR\_INV\_BARO, 138  
    ERROR\_IONO\_GIM, 138  
    ERROR\_IONO\_MODEL, 138  
    ERROR\_LPEOT, 138  
    ERROR\_MSS\_MODEL, 138  
    ERROR\_ODLE\_MODEL, 138  
    ERROR\_OLT, 138  
    ERROR\_OT, 138  
    ERROR\_SET, 138  
    ERROR\_SLOPE\_MODEL, 138  
    ERROR\_SNOW\_DENSITY, 138  
    ERROR\_SNOW\_DEPTH, 138  
    ERROR\_SSB, 138  
    ERROR\_SURFACE\_TYPE, 138  
    ERROR\_SWH, 138  
    ERROR\_WET\_TROPO, 138  
    ERROR\_WIND, 138  
    EXTERNAL\_CAL, 136  
    LOOP\_STATUS, 136  
    LOSS\_OF\_ECHO, 136  
    MCD\_AGC\_INCONSISTANT, 135  
    MCD\_ATT\_CORR\_MISSING, 135

- MCD\_BLOCK\_BLANK, [134](#)
- MCD\_BLOCK\_DEGRADED, [134](#)
- MCD\_C\_CAL\_1\_IPF\_DB\_USED, [135](#)
- MCD\_CAL1\_TYPE, [135](#)
- MCD\_CAL\_1\_CORR\_MISSING, [135](#)
- MCD\_CAL\_1\_IPF\_DB\_USED, [135](#)
- MCD\_CAL\_2\_CORR\_IPF\_DB, [135](#)
- MCD\_CAL\_2\_CORR\_MISSING, [135](#)
- MCD\_CH1\_ERROR, [135](#)
- MCD\_CH2\_ERROR, [135](#)
- MCD\_DATATION\_DEGRADED, [134](#)
- MCD\_DORIS\_USO\_CORR\_MISSING, [135](#)
- MCD\_ECHO\_ERROR, [135](#)
- MCD\_ECHO\_RX1\_ERROR, [135](#)
- MCD\_ECHO\_RX2\_ERROR, [135](#)
- MCD\_ECHO\_SATURATION, [135](#)
- MCD\_NPM\_INCONSISTANT, [135](#)
- MCD\_ORBIT\_DISCONTINUITY, [134](#)
- MCD\_ORBIT\_FILE\_CHANGE, [134](#)
- MCD\_ORBIT\_PROP\_ERROR, [134](#)
- MCD\_PHASE\_PETURBATION\_MODE, [135](#)
- MCD\_POWER\_SCALE\_ERROR, [135](#)
- MCD\_TRK\_ECHO\_ERROR, [135](#)
- MCD\_WINDOW\_DELAY\_INCONSISTANT, [135](#)
- MODE\_ID\_0, [136](#)
- MODE\_ID\_1, [136](#)
- MODE\_ID\_2, [136](#)
- MODE\_ID\_3, [136](#)
- MODE\_ID\_4, [136](#)
- MODE\_ID\_5, [136](#)
- PLAT\_ATT\_0, [137](#)
- PLAT\_ATT\_1, [137](#)
- RT\_ERROR, [136](#)
- RX\_BAND\_ATT, [136](#)
- RX\_CHAIN\_0, [136](#)
- RX\_CHAIN\_1, [136](#)
- SIN\_DEGRADED, [136](#)
- SIRAL\_ID, [136](#)
- ST\_ATTREF, [137](#)
- ST\_NONE, [137](#)
- STR\_ATTREF\_USED, [136](#)
- SURFACE\_CLOSED\_SEA, [137](#)
- SURFACE\_CONTINENTAL\_ICE, [137](#)
- SURFACE\_LAND, [137](#)
- SURFACE\_OPEN\_OCEAN, [137](#)
- t\_CorrectionErrorFlagBit, [138](#)
- t\_CorrectionStatusFlagBit, [137](#)
- t\_InstConf, [135](#)
- t\_MCDFlagBit, [134](#)
- t\_ModelID, [136](#)
- t\_STID, [137](#)
- t\_SurfaceType, [137](#)
- TRK\_MODE\_0, [136](#)
- TRK\_MODE\_1, [136](#)
- spare1
  - Corrections, [38](#)
- spare2
  - Corrections, [39](#)
- spares
  - Measurements, [79](#)
- t\_Average\_PL
  - L1B\_LRM, [40](#)
  - L1B\_SAR, [41](#)
  - L1B\_SARIN, [42](#)
- t\_Corr
  - L2Data, [45](#)
- t\_CorrectionErrorFlagBit
  - SharedFlags.h, [138](#)
- t\_CorrectionStatusFlagBit
  - SharedFlags.h, [137](#)
- t\_DiscriminatorResult
  - L2IStructure.h, [116](#)
- t\_DiscriminatorStatusFlagBit
  - L2IStructure.h, [117](#)
- t\_FreeboardStatusFlagBit
  - L2IStructure.h, [116](#)
- t\_Geo\_Corrections
  - L1B\_LRM, [40](#)
  - L1B\_SAR, [41](#)
  - L1B\_SARIN, [42](#)
- t\_HeightStatusFlagBit
  - L2IStructure.h, [115](#)
- t\_InstConf
  - SharedFlags.h, [135](#)
- t\_InterpolationErrorFlagBit
  - L2IStructure.h, [114](#)
- t\_Level1bAvgWfmLRMFlagBit
  - L1bStructure.h, [108](#)
- t\_Level1bAvgWfmSARFlagBit
  - L1bStructure.h, [109](#)
- t\_Level1bAvgWfmSARinFlagBit
  - L1bStructure.h, [108](#)
- t\_Level1bWfmLRMFlagBit
  - L1bStructure.h, [109](#)
- t\_Level1bWfmSARFlagBit
  - L1bStructure.h, [109](#)
- t\_Level1bWfmSARinFlagBit
  - L1bStructure.h, [108](#)
- t\_Level2AmbiguityErrorFlagBit
  - L2IStructure.h, [117](#)
- t\_Level2CorrectionsAppliedFlagBit
  - L2Structure.h, [123](#)
- t\_Level2CorrectionsStatusFlagBit
  - L2Structure.h, [123](#)
- t\_Level2QualityStatusFlagBit
  - L2Structure.h, [122](#)
- t\_MCDFlagBit
  - SharedFlags.h, [134](#)
- t\_MeasurementMode
  - L2IStructure.h, [114](#)
- t\_MeasurementQualityFlagBit
  - L2IStructure.h, [114](#)
- t\_ModelID
  - SharedFlags.h, [136](#)
- t\_RetrackerFlagBit
  - L2IStructure.h, [115](#)

- t\_STID
  - SharedFlags.h, 137
- t\_SurfaceType
  - SharedFlags.h, 137
- t\_cs\_filehandle
  - ptCSGetFileHandle.h, 127
- t\_cs\_fileinfo, 80
  - j\_baseline, 81
  - j\_num\_datasets, 80
  - j\_type, 81
  - paj\_num\_records, 80
  - pat\_data\_start\_offsets, 80
  - pt\_filepointer, 80
- TRK\_MODE\_0
  - SharedFlags.h, 136
- TRK\_MODE\_1
  - SharedFlags.h, 136
- TRUE
  - CS\_Defines.h, 96
- TimeAndOrbitGroup, 81
  - aj\_Baseline, 84
  - aj\_Real\_beam, 83
  - aj\_Sat\_velocity, 83
  - i\_ST\_ID, 84
  - j\_Alt, 83
  - j\_Alt\_rate, 83
  - j\_Day, 82
  - j\_Lat, 83
  - j\_Lon, 83
  - j\_Pitch\_angle, 84
  - j\_Roll\_angle, 84
  - j\_USO\_Corr, 82
  - j\_Yaw\_angle, 84
  - ui\_Mode\_ID, 82
  - ui\_SSC, 82
  - uj\_Inst\_config, 83
  - uj\_MCD, 84
  - uj\_Micsec, 82
  - uj\_Rec\_Count, 83
  - uj\_Sec, 82
- ui\_Mode\_ID
  - L2IData, 57
  - TimeAndOrbitGroup, 82
- ui\_SSC
  - L2IData, 57
  - TimeAndOrbitGroup, 82
- ui\_Wind\_speed
  - L2FinalC, 49
  - L2IData, 68
- ui\_flags
  - AveragedWaveformsLRM, 31
  - AveragedWaveformsSAR, 33
  - AveragedWaveformsSARIN, 36
  - WaveformsLRM, 86
  - WaveformsSAR, 87
  - WaveformsSARIN, 89
- ui\_interp\_cnt\_bkwd
  - L2IData, 67
- ui\_interp\_cnt\_fwd
  - L2IData, 67
- ui\_interp\_err
  - L2IData, 66
- ui\_interp\_error\_F
  - L2IData, 67
- ui\_interp\_time\_bkwd
  - L2IData, 67
- ui\_interp\_time\_fwd
  - L2IData, 67
- ui\_n\_avg
  - L2IData, 68
- ui\_num\_avg\_echoes
  - AveragedWaveformsLRM, 31
  - AveragedWaveformsSAR, 33
  - AveragedWaveformsSARIN, 35
  - WaveformsLRM, 85
  - WaveformsSAR, 87
  - WaveformsSARIN, 88
- uj\_Ambiguity
  - L2IData, 71
- uj\_C\_applied
  - L2FinalM, 53
- uj\_C\_status
  - L2FinalC, 49
- uj\_Corr\_error
  - Corrections, 39
  - L2IData, 74
- uj\_Corr\_status
  - Corrections, 38
  - L2IData, 74
- uj\_DEM\_id
  - L2IData, 72
- uj\_Discrim\_F
  - L2IData, 70
- uj\_Freeb\_status
  - L2IData, 68
- uj\_Ht\_status
  - L2IData, 68
- uj\_Inst\_config
  - L2IData, 57
  - TimeAndOrbitGroup, 83
- uj\_L2\_MCD
  - L2IData, 59
- uj\_MCD
  - TimeAndOrbitGroup, 84
- uj\_Meas\_Mode
  - L2IData, 67
- uj\_Micsec
  - L2Data, 44
  - L2IData, 57
  - TimeAndOrbitGroup, 82
- uj\_Micsec\_1Hz
  - AveragedWaveformsLRM, 30
  - AveragedWaveformsSAR, 32
  - AveragedWaveformsSARIN, 34
- uj\_Quality\_F
  - L2FinalM, 53

- L2IData, [67](#)
- uj\_Rec\_Count
  - L2IData, [58](#)
  - TimeAndOrbitGroup, [83](#)
- uj\_Retracker\_F
  - L2IData, [67](#)
- uj\_Sec
  - L2Data, [44](#)
  - L2IData, [57](#)
  - TimeAndOrbitGroup, [82](#)
- uj\_Sec\_1Hz
  - AveragedWaveformsLRM, [30](#)
  - AveragedWaveformsSAR, [32](#)
  - AveragedWaveformsSARIN, [34](#)
- uj\_Surf\_type
  - Corrections, [38](#)
  - L2IData, [73](#)
- uk\_Siral\_mode
  - L2Data, [44](#)
- uk\_Surf\_type
  - L2FinalC, [48](#)
- Utility functions, [22](#)
  - iCheckBitLong, [24](#)
  - iCheckBitLongLong, [24](#)
  - iCheckBitShort, [22](#)
  - vClearBitLongLong, [25](#)
  - vClearBitShort, [22](#)
  - vSetBitLong, [24](#)
  - vSetBitLongLong, [24](#)
  - vSetBitShort, [22](#)
- vCSFreeFileHandle
  - High-level shared routines, [10](#)
- vCSFreeFileHandle.c, [139](#)
- vCSFreeFileHandle.h, [139](#)
- vClearBitLong
  - BitmaskUtilities.c, [92](#)
  - BitmaskUtilities.h, [93](#)
- vClearBitLongLong
  - Utility functions, [25](#)
- vClearBitShort
  - Utility functions, [22](#)
- vDump\_L1B\_LRM
  - L1bStructure.h, [110](#)
- vDump\_L1B\_SAR
  - L1bStructure.h, [110](#)
- vDump\_L1B\_SARIN
  - L1bStructure.h, [110](#)
- vDump\_L2Data
  - L2Structure.c, [119](#)
  - L2Structure.h, [124](#)
- vDump\_L2FinalC
  - L2Structure.c, [119](#)
- vDump\_L2FinalM
  - L2Structure.c, [119](#)
- vDump\_L2IData
  - L2IStructure.c, [111](#)
  - L2IStructure.h, [118](#)
- vSetBitLong
  - Utility functions, [24](#)
- vSetBitLongLong
  - Utility functions, [24](#)
- vSetBitShort
  - Utility functions, [22](#)
- WFM\_AA\_POWER\_ECHOES
  - L1bStructure.h, [109](#)
- WFM\_APPROX\_BEAM\_STEER
  - L1bStructure.h, [109](#)
- WFM\_AUTO\_STEER
  - L1bStructure.h, [109](#)
- WFM\_DOPPLER\_APPLY
  - L1bStructure.h, [109](#)
- WFM\_DOPPLER\_COMP
  - L1bStructure.h, [109](#)
- WFM\_EXACT\_BEAM\_STEER
  - L1bStructure.h, [109](#)
- WFM\_LRM\_ERROR
  - L1bStructure.h, [109](#)
- WFM\_LRM\_TRK\_0
  - L1bStructure.h, [109](#)
- WFM\_LRM\_TRK\_1
  - L1bStructure.h, [109](#)
- WFM\_LRM\_TRK\_2
  - L1bStructure.h, [109](#)
- WFM\_MULTI\_LOOK\_INCOMPLETE
  - L1bStructure.h, [109](#)
- WFM\_SAR\_ERROR
  - L1bStructure.h, [109](#)
- WFM\_SAR\_MISPOINT
  - L1bStructure.h, [109](#)
- WFM\_STEER\_ERROR
  - L1bStructure.h, [109](#)
- WaveformsLRM, [85](#)
  - j\_Linear\_wfm\_mult, [85](#)
  - j\_Power2\_wfm\_mult, [85](#)
  - pui\_wfm, [85](#)
  - ui\_flags, [86](#)
  - ui\_num\_avg\_echoes, [85](#)
- WaveformsSAR, [86](#)
  - ai\_bb\_param, [87](#)
  - j\_Linear\_wfm\_mult, [87](#)
  - j\_Power2\_wfm\_mult, [87](#)
  - pui\_wfm, [87](#)
  - ui\_flags, [87](#)
  - ui\_num\_avg\_echoes, [87](#)
- WaveformsSARIN, [87](#)
  - ai\_bb\_param, [89](#)
  - j\_Linear\_wfm\_mult, [88](#)
  - j\_Power2\_wfm\_mult, [88](#)
  - pj\_phase\_diff, [89](#)
  - pui\_coherence, [89](#)
  - pui\_wfm, [88](#)
  - ui\_flags, [89](#)
  - ui\_num\_avg\_echoes, [88](#)