

Watchman_software

Generated by Doxygen 1.8.15

1 Data Structure Index	1
1.1 Data Structures	1
2 File Index	3
2.1 File List	3
3 Data Structure Documentation	5
3.1 coordinates_st Struct Reference	5
3.1.1 Detailed Description	5
3.1.2 Field Documentation	5
3.1.2.1 x	5
3.1.2.2 y	5
3.2 data_axi_st Struct Reference	6
3.2.1 Detailed Description	6
3.2.2 Field Documentation	6
3.2.2.1 data	6
3.2.2.2 info	6
3.2.2.3 PL_spare	6
3.2.2.4 wdo_id	6
3.2.2.5 wdo_time	7
3.3 data_axi_union Union Reference	7
3.3.1 Detailed Description	7
3.3.2 Field Documentation	7
3.3.2.1 data_array	7
3.3.2.2 data_struct	7
3.4 data_list_st Struct Reference	8
3.4.1 Field Documentation	8
3.4.1.1 data	8
3.4.1.2 next	8
3.4.1.3 previous	8
3.5 features_ext_st Struct Reference	8
3.5.1 Detailed Description	9
3.5.2 Field Documentation	9
3.5.2.1 amplitude	9
3.5.2.2 time	9
3.6 GMtype_CoordCartesian Struct Reference	9
3.6.1 Field Documentation	9
3.6.1.1 x	10
3.6.1.2 y	10
3.6.1.3 z	10
3.7 GMtype_CoordDipole Struct Reference	10
3.7.1 Field Documentation	10
3.7.1.1 lambda	10

3.7.1.2 phi	10
3.8 GMtype_CoordGeodetic Struct Reference	11
3.8.1 Field Documentation	11
3.8.1.1 HeightAboveEllipsoid	11
3.8.1.2 lambda	11
3.8.1.3 phi	11
3.9 GMtype_CoordSpherical Struct Reference	11
3.9.1 Field Documentation	12
3.9.1.1 lambda	12
3.9.1.2 phig	12
3.9.1.3 r	12
3.10 GMtype_Data Struct Reference	12
3.10.1 Field Documentation	12
3.10.1.1 element	12
3.10.1.2 size	13
3.11 GMtype_Date Struct Reference	13
3.11.1 Field Documentation	13
3.11.1.1 Day	13
3.11.1.2 DayNumber	13
3.11.1.3 DecimalYear	13
3.11.1.4 Month	14
3.11.1.5 Year	14
3.12 GMtype_Ellipsoid Struct Reference	14
3.12.1 Field Documentation	14
3.12.1.1 a	14
3.12.1.2 b	14
3.12.1.3 eps	15
3.12.1.4 epssq	15
3.12.1.5 fla	15
3.12.1.6 re	15
3.13 GMtype_Matrix Struct Reference	15
3.13.1 Field Documentation	15
3.13.1.1 columns	15
3.13.1.2 element	16
3.13.1.3 rows	16
3.14 GMtype_Model Struct Reference	16
3.14.1 Field Documentation	16
3.14.1.1 g0	16
3.14.1.2 g1	16
3.14.1.3 h1	17
3.15 GMtype_Pole Struct Reference	17
3.15.1 Field Documentation	17

3.15.1.1 lambda	17
3.15.1.2 M	17
3.15.1.3 phi	17
3.16 GMtype_Polynomial Struct Reference	18
3.16.1 Field Documentation	18
3.16.1.1 coef	18
3.16.1.2 degree	18
3.17 time_cplt_st Struct Reference	18
3.17.1 Detailed Description	19
3.17.2 Field Documentation	19
3.17.2.1 day	19
3.17.2.2 hour	19
3.17.2.3 milisecond	19
3.17.2.4 minute	19
3.17.2.5 month	19
3.17.2.6 second	19
3.17.2.7 year	20
3.18 time_union Union Reference	20
3.18.1 Detailed Description	20
3.18.2 Field Documentation	20
3.18.2.1 time_fl	20
3.18.2.2 time_t	20
3.19 TmrCntrSetup_st Struct Reference	21
3.19.1 Detailed Description	21
3.19.2 Field Documentation	21
3.19.2.1 Interval	21
3.19.2.2 Options	21
3.19.2.3 OutputHz	21
3.19.2.4 Prescaler	21
4 File Documentation	23
4.1 axis_peripheral.c File Reference	23
4.1.1 Detailed Description	24
4.1.2 Function Documentation	24
4.1.2.1 dma_received_data()	24
4.1.2.2 test_TPG()	24
4.1.2.3 XAxiDma_SimpleTransfer_hm()	25
4.1.3 Variable Documentation	25
4.1.3.1 empty_flag	25
4.1.3.2 first_element	26
4.1.3.3 flag_axidma_error	26
4.1.3.4 flag_axidma_rx_done	26

4.1.3.5 flag_scu_timer	26
4.1.3.6 flag_ttcps_timer	26
4.1.3.7 frame_buf	26
4.1.3.8 last_element	27
4.1.3.9 regptr	27
4.1.3.10 WdtScuInstance	27
4.2 axis_peripheral.h File Reference	27
4.2.1 Detailed Description	28
4.2.2 Macro Definition Documentation	28
4.2.2.1 FEATURES_ID	28
4.2.2.2 FULL_WAVEFORM_ID	28
4.2.3 Function Documentation	28
4.2.3.1 dma_received_data()	28
4.2.3.2 test_TPG()	29
4.2.3.3 XAxiDma_SimpleTransfer_hm()	29
4.3 data_analysis.c File Reference	30
4.3.1 Function Documentation	30
4.3.1.1 correct_data()	30
4.3.1.2 extract_features()	31
4.3.2 Variable Documentation	31
4.3.2.1 lookup_table	31
4.3.2.2 pedestal	31
4.4 data_analysis.h File Reference	32
4.4.1 Macro Definition Documentation	33
4.4.1.1 CHANNEL	33
4.4.1.2 LAST_SHIFT	33
4.4.1.3 MASK_INFO	33
4.4.1.4 MAX_WINDOW	34
4.4.1.5 SAMPLE	34
4.4.1.6 SIZE_DATA_ARRAY	34
4.4.1.7 SIZE_DATA_ARRAY_BYT	34
4.4.1.8 THRESHOLD_CMP	34
4.4.1.9 THRESHOLD_PULSE	34
4.4.1.10 TOO_LONG_SHIFT	35
4.4.1.11 TRIG_SHIFT	35
4.4.1.12 VPED_ANALOG	35
4.4.1.13 VPED_DIGITAL	35
4.4.2 Typedef Documentation	35
4.4.2.1 coordinates	35
4.4.2.2 data_axi	35
4.4.2.3 data_axi_un	36
4.4.2.4 data_list	36

4.4.2.5 features_ext	36
4.4.2.6 time_un	36
4.4.3 Function Documentation	36
4.4.3.1 correct_data()	36
4.4.3.2 extract_features()	37
4.5 data_test.c File Reference	37
4.5.1 Function Documentation	38
4.5.1.1 made_frame()	38
4.6 data_test.h File Reference	38
4.6.1 Function Documentation	38
4.6.1.1 made_frame()	38
4.7 file_hm.c File Reference	39
4.7.1 Function Documentation	39
4.7.1.1 create_logfile()	39
4.7.1.2 create_timefile()	40
4.7.1.3 log_event()	40
4.7.1.4 log_wdtevent()	41
4.7.1.5 mount_sd_card()	41
4.7.1.6 update_timefile()	42
4.7.2 Variable Documentation	42
4.7.2.1 Path	42
4.8 file_hm.h File Reference	42
4.8.1 Function Documentation	43
4.8.1.1 create_logfile()	43
4.8.1.2 create_timefile()	43
4.8.1.3 log_event()	44
4.8.1.4 log_wdtevent()	44
4.8.1.5 mount_sd_card()	45
4.8.1.6 update_timefile()	45
4.9 get_20_windows.c File Reference	46
4.9.1 Function Documentation	46
4.9.1.1 get_20_windows_fct()	46
4.9.2 Variable Documentation	47
4.9.2.1 flag_axidma_error	47
4.9.2.2 flag_axidma_rx_done	47
4.9.2.3 flag_scu_timer	47
4.9.2.4 flag_ttcps_timer	47
4.9.2.5 frame_buf	48
4.9.2.6 lookup_table	48
4.9.2.7 pedestal	48
4.9.2.8 regptr	48
4.9.2.9 WdtSculInstance	48

4.10 get_20_windows.h File Reference	48
4.10.1 Function Documentation	49
4.10.1.1 get_20_windows_fct()	49
4.11 get_transfer_fct.c File Reference	49
4.11.1 Function Documentation	50
4.11.1.1 send_data_transfer_fct()	50
4.11.2 Variable Documentation	50
4.11.2.1 flag_axidma_error	50
4.11.2.2 flag_axidma_rx_done	51
4.11.2.3 flag_scu_timer	51
4.11.2.4 flag_ttcps_timer	51
4.11.2.5 frame_buf	51
4.11.2.6 lookup_table	51
4.11.2.7 pedestal	51
4.11.2.8 regptr	52
4.11.2.9 WdtScuInstance	52
4.12 get_transfer_fct.h File Reference	52
4.12.1 Function Documentation	52
4.12.1.1 send_data_transfer_fct()	52
4.13 global.c File Reference	53
4.13.1 Detailed Description	54
4.13.2 Function Documentation	54
4.13.2.1 cleanup_global_var()	54
4.13.2.2 init_global_var()	55
4.13.3 Variable Documentation	55
4.13.3.1 AxiDmaInstance	55
4.13.3.2 count_scu_timer	56
4.13.3.3 count_ttcps_timer	56
4.13.3.4 echo_netif	56
4.13.3.5 empty_flag	56
4.13.3.6 first_element	56
4.13.3.7 flag_assertion	56
4.13.3.8 flag_axidma_error	57
4.13.3.9 flag_axidma_rx	57
4.13.3.10 flag_axidma_rx_done	57
4.13.3.11 flag_scu_timer	57
4.13.3.12 flag_ttcps_timer	57
4.13.3.13 flag_while_loop	57
4.13.3.14 frame_buf	58
4.13.3.15 frame_buf_cmd	58
4.13.3.16 frame_buf_cmd_tmp	58
4.13.3.17 frame_buf_tmp	58

4.13.3.18 get_20_windows_flag	58
4.13.3.19 get_transfer_fct_flag	58
4.13.3.20 last_element	59
4.13.3.21 lookup_table	59
4.13.3.22 nbre_of_bytes	59
4.13.3.23 pedestal	59
4.13.3.24 regptr	59
4.13.3.25 run_flag	59
4.13.3.26 stream_flag	60
4.13.3.27 WdtSculInstance	60
4.14 global.h File Reference	60
4.14.1 Detailed Description	60
4.14.2 Function Documentation	60
4.14.2.1 cleanup_global_var()	60
4.14.2.2 init_global_var()	61
4.15 GM_SubLibrary.c File Reference	61
4.15.1 Detailed Description	62
4.15.2 Function Documentation	63
4.15.2.1 GM_CartesianToSpherical()	63
4.15.2.2 GM_CORD()	63
4.15.2.3 GM_DateToYear()	63
4.15.2.4 GM_DotProduct()	63
4.15.2.5 GM_EarthCartToDipoleCartCD()	63
4.15.2.6 GM_GeodeticToSpherical()	64
4.15.2.7 GM_GetUserInput()	64
4.15.2.8 GM_LinearInterpolation()	64
4.15.2.9 GM_LUDecomposition()	64
4.15.2.10 GM_LUSolve()	64
4.15.2.11 GM_MatDet()	65
4.15.2.12 GM_MatInverse()	65
4.15.2.13 GM_MatMultiply()	65
4.15.2.14 GM_MatTranspose()	65
4.15.2.15 GM_Mean()	65
4.15.2.16 GM_Median()	65
4.15.2.17 GM_PoleLocation()	66
4.15.2.18 GM_PolyFit()	66
4.15.2.19 GM_Pow()	66
4.15.2.20 GM_PrintMatrix()	66
4.15.2.21 GM_PrintUserData()	66
4.15.2.22 GM_ScanIGRF()	66
4.15.2.23 GM_SetEllipsoid()	67
4.15.2.24 GM_SolvePolynomial()	67

4.15.2.25 GM_Sort()	67
4.15.2.26 GM_SphericalToCartesian()	67
4.15.2.27 GM_StandardDeviation()	67
4.15.2.28 GM_Swap()	67
4.15.2.29 GM_SwapRows()	68
4.15.2.30 GM_TimeAdjustCoefs()	68
4.16 GMHeader.h File Reference	68
4.16.1 Macro Definition Documentation	69
4.16.1.1 ATanH	69
4.16.1.2 DEG2RAD	70
4.16.1.3 FALSE	70
4.16.1.4 GM_STARTYEAR	70
4.16.1.5 M_PI	70
4.16.1.6 MU_0	70
4.16.1.7 R_e	70
4.16.1.8 RAD2DEG	70
4.16.1.9 TRUE	71
4.16.2 Function Documentation	71
4.16.2.1 GM_CartesianToSpherical()	71
4.16.2.2 GM_CORD()	71
4.16.2.3 GM_DateToYear()	71
4.16.2.4 GM_DotProduct()	71
4.16.2.5 GM_EarthCartToDipoleCartCD()	72
4.16.2.6 GM_GeodeticToSpherical()	72
4.16.2.7 GM_GetUserInput()	72
4.16.2.8 GM_LinearInterpolation()	72
4.16.2.9 GM_LUDecomposition()	72
4.16.2.10 GM_LUSolve()	73
4.16.2.11 GM_MatDet()	73
4.16.2.12 GM_MatInverse()	73
4.16.2.13 GM_MatMultiply()	73
4.16.2.14 GM_MatTranspose()	73
4.16.2.15 GM_Mean()	73
4.16.2.16 GM_Median()	74
4.16.2.17 GM_PoleLocation()	74
4.16.2.18 GM_PolyFit()	74
4.16.2.19 GM_Pow()	74
4.16.2.20 GM_PrintMatrix()	74
4.16.2.21 GM_PrintUserData()	74
4.16.2.22 GM_ScanIGRF()	75
4.16.2.23 GM_SetEllipsoid()	75
4.16.2.24 GM_SolvePolynomial()	75

4.16.2.25 GM_Sort()	75
4.16.2.26 GM_SphericalToCartesian()	75
4.16.2.27 GM_StandardDeviation()	75
4.16.2.28 GM_Swap()	76
4.16.2.29 GM_SwapRows()	76
4.16.2.30 GM_TimeAdjustCoefs()	76
4.17 iic_DAC_LTC2657.c File Reference	76
4.17.1 Detailed Description	77
4.17.2 Function Documentation	77
4.17.2.1 DAC_LTC2657_initialize()	77
4.17.2.2 DAC_LTC2657_SetChannelVoltage()	77
4.17.3 Variable Documentation	78
4.17.3.1 I2cInstance	78
4.18 iic_DAC_LTC2657.h File Reference	78
4.18.1 Detailed Description	79
4.18.2 Macro Definition Documentation	79
4.18.2.1 CHANNEL_A	80
4.18.2.2 CHANNEL_ALL	80
4.18.2.3 CHANNEL_B	80
4.18.2.4 CHANNEL_C	80
4.18.2.5 CHANNEL_D	80
4.18.2.6 CHANNEL_E	80
4.18.2.7 CHANNEL_F	81
4.18.2.8 CHANNEL_G	81
4.18.2.9 CHANNEL_H	81
4.18.2.10 DAC_GRP_0	81
4.18.2.11 DAC_GRP_1	81
4.18.2.12 DAC_GRP_2	81
4.18.2.13 DAC_GRP_3	82
4.18.2.14 DAC_VPED	82
4.18.2.15 IIC_DAC_LTC2657_H	82
4.18.2.16 IIC_DEVICE_ID	82
4.18.2.17 IIC_SLAVE_ADDRESS	82
4.18.3 Function Documentation	82
4.18.3.1 DAC_LTC2657_initialize()	82
4.18.3.2 DAC_LTC2657_SetChannelVoltage()	83
4.19 interrupt.c File Reference	83
4.19.1 Detailed Description	85
4.19.2 Function Documentation	85
4.19.2.1 assert_callback()	85
4.19.2.2 axidma_rx_callback()	85
4.19.2.3 cleanup_interrupts()	86

4.19.2.4 devices_initialization()	86
4.19.2.5 enable_interrupts()	87
4.19.2.6 interrupts_initialization()	87
4.19.2.7 setup_axidma_int()	88
4.19.2.8 setup_scu_timer_int()	88
4.19.2.9 setup_scu_wdt_int()	88
4.19.2.10 setup_ttcps_timer_int()	89
4.19.2.11 timer_scu_callback()	89
4.19.2.12 timer_ttcps_callback()	90
4.19.2.13 wdt_scu_callback()	90
4.19.3 Variable Documentation	91
4.19.3.1 AxiDmaInstance	91
4.19.3.2 count_scu_timer	91
4.19.3.3 echo_netif	91
4.19.3.4 empty_flag	91
4.19.3.5 flag_assertion	91
4.19.3.6 flag_axidma_error	92
4.19.3.7 flag_axidma_rx	92
4.19.3.8 flag_axidma_rx_done	92
4.19.3.9 flag_scu_timer	92
4.19.3.10 flag_ttcps_timer	92
4.19.3.11 flag_while_loop	92
4.19.3.12 last_element	93
4.19.3.13 stream_flag	93
4.19.3.14 WdtScuInstance	93
4.20 interrupt.h File Reference	93
4.20.1 Detailed Description	95
4.20.2 Macro Definition Documentation	95
4.20.2.1 INTC_BASE_ADDR	95
4.20.2.2 INTC_DEVICE_ID	95
4.20.2.3 INTC_DIST_BASE_ADDR	96
4.20.2.4 RESET_RX_CNTR_LIMIT	96
4.20.2.5 TIMER_DEVICE_ID	96
4.20.2.6 TIMER_IRPT_INTR	96
4.20.2.7 TTC_TICK_DEVICE_ID	96
4.20.2.8 TTC_TICK_INTR_ID	96
4.20.2.9 TTCPS_TIMER_FREQ_HZ	97
4.20.2.10 WDT_DEVICE_ID	97
4.20.2.11 WDT_IRPT_INTR	97
4.20.2.12 WDT_LOAD_VALUE	97
4.20.3 Typedef Documentation	97
4.20.3.1 TmrCntrSetup	97

4.20.4 Function Documentation	97
4.20.4.1 assert_callback()	97
4.20.4.2 axidma_rx_callback()	98
4.20.4.3 cleanup_interrupts()	98
4.20.4.4 devices_initialization()	99
4.20.4.5 enable_interrupts()	99
4.20.4.6 interrupts_initialization()	100
4.20.4.7 setup_axidma_int()	100
4.20.4.8 setup_scu_timer_int()	100
4.20.4.9 setup_ttcps_timer_int()	101
4.20.4.10 timer_scu_callback()	101
4.20.4.11 timer_ttcps_callback()	102
4.20.4.12 wdt_scu_callback()	102
4.21 main.c File Reference	103
4.21.1 Detailed Description	104
4.21.2 Typedef Documentation	104
4.21.2.1 clean_state_en	104
4.21.2.2 dma_stm_en	105
4.21.3 Enumeration Type Documentation	105
4.21.3.1 clean_state_enum	105
4.21.3.2 dma_stm_enum	105
4.21.4 Function Documentation	105
4.21.4.1 end_main()	105
4.21.4.2 main()	106
4.21.5 Variable Documentation	106
4.21.5.1 echo_netif	106
4.21.5.2 empty_flag	106
4.21.5.3 first_element	106
4.21.5.4 flag_assertion	106
4.21.5.5 flag_axidma_rx	107
4.21.5.6 flag_scu_timer	107
4.21.5.7 flag_ttcps_timer	107
4.21.5.8 flag_while_loop	107
4.21.5.9 get_20_windows_flag	107
4.21.5.10 get_transfer_fct_flag	107
4.21.5.11 regptr	108
4.21.5.12 run_flag	108
4.21.5.13 stream_flag	108
4.21.5.14 WdtScuInstance	108
4.22 pedestal.c File Reference	108
4.22.1 Detailed Description	109
4.22.2 Function Documentation	109

4.22.2.1 init_pedestals()	109
4.22.3 Variable Documentation	110
4.22.3.1 flag_axidma_error	110
4.22.3.2 flag_axidma_rx_done	110
4.22.3.3 flag_scu_timer	110
4.22.3.4 flag_ttcps_timer	110
4.22.3.5 pedestal	111
4.22.3.6 regptr	111
4.22.3.7 WdtScuInstance	111
4.23 pedestal.h File Reference	111
4.23.1 Detailed Description	111
4.23.2 Function Documentation	112
4.23.2.1 init_pedestals()	112
4.24 platform_config.h File Reference	112
4.24.1 Macro Definition Documentation	112
4.24.1.1 PLATFORM_EMAC_BASEADDR	112
4.24.1.2 PLATFORM_ZYNQ	112
4.25 platform_mb.c File Reference	113
4.26 platform_ppc.c File Reference	113
4.27 sfp.c File Reference	113
4.27.1 Detailed Description	113
4.28 si5324.c File Reference	113
4.28.1 Detailed Description	113
4.29 TARGETC_RegisterMap.c File Reference	114
4.29.1 Detailed Description	114
4.29.2 Function Documentation	114
4.29.2.1 ControlRegisterWrite()	114
4.29.2.2 GetTargetCControl()	115
4.29.2.3 GetTargetCStatus()	115
4.29.2.4 SetTargetCRegisters()	116
4.29.2.5 WriteReadBackRegister()	116
4.29.2.6 WriteRegister()	117
4.29.3 Variable Documentation	117
4.29.3.1 regptr	117
4.30 TARGETC_RegisterMap.h File Reference	117
4.30.1 Detailed Description	121
4.30.2 Macro Definition Documentation	121
4.30.2.1 BUSY_MASK	122
4.30.2.2 CPUMODE_MASK	122
4.30.2.3 DISABLE	122
4.30.2.4 ENABLE	122
4.30.2.5 INIT	122

4.30.2.6 LAST_REGISTER_ADDR	122
4.30.2.7 LOCKED_MASK	123
4.30.2.8 PSBUSY_MASK	123
4.30.2.9 REGCLR_MASK	123
4.30.2.10 SMODE_MASK	123
4.30.2.11 SS_TPG_MASK	123
4.30.2.12 SSVALID_MASK	123
4.30.2.13 STORAGE_MASK	124
4.30.2.14 SWRESET_MASK	124
4.30.2.15 TARGETC_REGISTERMAP_H	124
4.30.2.16 TC_ADDR_REG	124
4.30.2.17 TC_CMPBIAS2_REG	124
4.30.2.18 TC_CMPBIASIN_REG	124
4.30.2.19 TC_CONTROL_REG	125
4.30.2.20 TC_DATA_OUT_REG	125
4.30.2.21 TC_DBBIAS_REG	125
4.30.2.22 TC_eDO_CH0_REG	125
4.30.2.23 TC_eDO_CH10_REG	125
4.30.2.24 TC_eDO_CH11_REG	125
4.30.2.25 TC_eDO_CH12_REG	126
4.30.2.26 TC_eDO_CH13_REG	126
4.30.2.27 TC_eDO_CH14_REG	126
4.30.2.28 TC_eDO_CH15_REG	126
4.30.2.29 TC_eDO_CH1_REG	126
4.30.2.30 TC_eDO_CH2_REG	126
4.30.2.31 TC_eDO_CH3_REG	127
4.30.2.32 TC_eDO_CH4_REG	127
4.30.2.33 TC_eDO_CH5_REG	127
4.30.2.34 TC_eDO_CH6_REG	127
4.30.2.35 TC_eDO_CH7_REG	127
4.30.2.36 TC_eDO_CH8_REG	127
4.30.2.37 TC_eDO_CH9_REG	128
4.30.2.38 TC_FSTWINDOW_REG	128
4.30.2.39 TC_ISEL_REG	128
4.30.2.40 TC_MISCDIG_REG	128
4.30.2.41 TC_MONTIMING_REG	128
4.30.2.42 TC_MT_PASS_MASK	128
4.30.2.43 TC_MT_SSPIN_MASK	129
4.30.2.44 TC_MT_SSPOUT_MASK	129
4.30.2.45 TC_MT_SSTOUT_MASK	129
4.30.2.46 TC_MT_SSTOUTFB_MASK	129
4.30.2.47 TC_MT_VDD_MASK	129

4.30.2.48 TC_MT_WR1_ADDR_SYNC_MASK	129
4.30.2.49 TC_MT_WR2_ADDR_SYNC_MASK	130
4.30.2.50 TC_MT_WR_STRB1_MASK	130
4.30.2.51 TC_MT_WR_STRB2_MASK	130
4.30.2.52 TC_NBRWINDOW_REG	130
4.30.2.53 TC_PUBIAS_REG	130
4.30.2.54 TC_QBIAS_REG	130
4.30.2.55 TC_SBBIAS_REG	131
4.30.2.56 TC_SSPIN_LE_REG	131
4.30.2.57 TC_SSPIN_TE_REG	131
4.30.2.58 TC_SSTOUTFB_REG	131
4.30.2.59 TC_STATUS_REG	131
4.30.2.60 TC_TPG_REG	131
4.30.2.61 TC_VADJN_REG	132
4.30.2.62 TC_VADJP_REG	132
4.30.2.63 TC_VANBUFF_REG	132
4.30.2.64 TC_VAPBUFF_REG	132
4.30.2.65 TC_VBIAS_REG	132
4.30.2.66 TC_VDISCH_REG	132
4.30.2.67 TC_VDLYTUNE_REG	133
4.30.2.68 TC_VQBUFF_REG	133
4.30.2.69 TC_VTRIMT_REG	133
4.30.2.70 TC_WR1_ADDR_LE_REG	133
4.30.2.71 TC_WR1_ADDR_TE_REG	133
4.30.2.72 TC_WR2_ADDR_LE_REG	133
4.30.2.73 TC_WR2_ADDR_TE_REG	134
4.30.2.74 TC_WR_STRB1_LE_REG	134
4.30.2.75 TC_WR_STRB1_TE_REG	134
4.30.2.76 TC_WR_STRB2_LE_REG	134
4.30.2.77 TC_WR_STRB2_TE_REG	134
4.30.2.78 TESTFIFO_MASK	134
4.30.2.79 TESTSTREAM_MASK	135
4.30.2.80 WINDOW_MASK	135
4.30.2.81 WINDOWBUSY_MASK	135
4.30.2.82 WRITE_MASK	135
4.30.3 Function Documentation	135
4.30.3.1 ControlRegisterWrite()	135
4.30.3.2 GetTargetCControl()	136
4.30.3.3 GetTargetCStatus()	136
4.30.3.4 SetTargetCRegisters()	137
4.30.3.5 WriteReadBackRegister()	137
4.30.3.6 WriteRegister()	137

4.31 time_hm.c File Reference	138
4.31.1 Detailed Description	138
4.31.2 Function Documentation	139
4.31.2.1 addtime()	139
4.31.2.2 gettime_hm()	139
4.31.2.3 isALeapYear()	140
4.31.2.4 settime_hm()	141
4.31.2.5 stringtotime()	141
4.31.3 Variable Documentation	142
4.31.3.1 day_per_month	142
4.31.3.2 offset_counter	142
4.31.3.3 offset_time	142
4.32 time_hm.h File Reference	142
4.32.1 Detailed Description	143
4.32.2 Typedef Documentation	143
4.32.2.1 time_cplt	143
4.32.3 Function Documentation	143
4.32.3.1 addtime()	143
4.32.3.2 gettime_hm()	144
4.32.3.3 isALeapYear()	144
4.32.3.4 settime_hm()	145
4.32.3.5 stringtotime()	145
4.33 transfer_function.c File Reference	146
4.33.1 Detailed Description	146
4.33.2 Function Documentation	147
4.33.2.1 init_transfer_function()	147
4.33.3 Variable Documentation	147
4.33.3.1 flag_axidma_error	147
4.33.3.2 flag_axidma_rx_done	147
4.33.3.3 flag_scu_timer	147
4.33.3.4 flag_ttcs_timer	148
4.33.3.5 lookup_table	148
4.33.3.6 pedestal	148
4.33.3.7 regptr	148
4.33.3.8 WdtScuInstance	148
4.34 transfer_function.h File Reference	148
4.34.1 Detailed Description	149
4.34.2 Function Documentation	149
4.34.2.1 init_transfer_function()	149
4.35 udp_peripheral.c File Reference	150
4.35.1 Detailed Description	151
4.35.2 Function Documentation	151

4.35.2.1 cleanup_udp()	151
4.35.2.2 command_parser()	152
4.35.2.3 print_ip()	152
4.35.2.4 print_ip_settings()	153
4.35.2.5 setup_pcb_cmd()	153
4.35.2.6 setup_pcb_data()	154
4.35.2.7 setup_udp_settings()	154
4.35.2.8 transfer_cmd()	155
4.35.2.9 transfer_data()	155
4.35.2.10 udp_cmd_rcv()	156
4.35.3 Variable Documentation	156
4.35.3.1 buf_cmd	156
4.35.3.2 buf_data	157
4.35.3.3 count_scu_timer	157
4.35.3.4 count_ttcs_timer	157
4.35.3.5 frame_buf_cmd	157
4.35.3.6 get_20_windows_flag	157
4.35.3.7 get_transfer_fct_flag	157
4.35.3.8 nbre_of_bytes	158
4.35.3.9 pcb_cmd	158
4.35.3.10 pcb_data	158
4.35.3.11 regptr	158
4.35.3.12 run_flag	158
4.35.3.13 stream_flag	158
4.36 udp_peripheral.h File Reference	159
4.36.1 Macro Definition Documentation	160
4.36.1.1 BUF_HEADER_SIZE	160
4.36.1.2 MAX_CMD_SIZE	160
4.36.1.3 MAX_DATA_SIZE	160
4.36.1.4 PORT_CMD	160
4.36.1.5 PORT_DATA	160
4.36.1.6 REGMAP_SIZE_UDP	160
4.36.2 Function Documentation	160
4.36.2.1 cleanup_udp()	160
4.36.2.2 command_parser()	161
4.36.2.3 print_ip()	161
4.36.2.4 print_ip_settings()	162
4.36.2.5 setup_pcb_cmd()	162
4.36.2.6 setup_pcb_data()	163
4.36.2.7 setup_udp_settings()	163
4.36.2.8 tcp_fasttmr()	164
4.36.2.9 tcp_slowtmr()	164

4.36.2.10 transfer_data()	164
4.36.2.11 udp_cmd_recv()	164
4.37 utility.c File Reference	165
4.37.1 Detailed Description	165
4.37.2 Function Documentation	166
4.37.2.1 decToBin()	166
4.37.2.2 decToHexa()	166
4.38 utility.h File Reference	166
4.38.1 Detailed Description	167
4.38.2 Function Documentation	167
4.38.2.1 decToBin()	167
4.38.2.2 decToHexa()	167
Index	169

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

coordinates_st	Structure containing the coordinations of a point	5
data_axi_st	Structure of an element of the list which represent a window	6
data_axi_union	Union to access an element as an array, used to give the element's address to the DMA	7
data_list_st	8
features_ext_st	Structure containing the features extracted from a normal pulse	8
GMtype_CoordCartesian	9
GMtype_CoordDipole	10
GMtype_CoordGeodetic	11
GMtype_CoordSpherical	11
GMtype_Data	12
GMtype_Date	13
GMtype_Ellipsoid	14
GMtype_Matrix	15
GMtype_Model	16
GMtype_Pole	17
GMtype_Polynomial	18
time_cplt_st	Structure of the timestamp	18
time_union	Union to convert access a float as an int without converting it in int	20
TmrCntrSetup_st	Structure containing all the settings to set up the Triple Timer Counter	21

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

axis_peripheral.c	23
axis_peripheral.h	27
data_analysis.c	30
data_analysis.h	32
data_test.c	37
data_test.h	38
file_hm.c	39
file_hm.h	42
get_20_windows.c	46
get_20_windows.h	48
get_transfer_fct.c	49
get_transfer_fct.h	52
global.c	53
global.h	60
GM_SubLibrary.c	61
GMHeader.h	68
iic_DAC_LTC2657.c	76
iic_DAC_LTC2657.h	78
interrupt.c	83
interrupt.h	93
main.c	103
pedestal.c	108
pedestal.h	111
platform_config.h	112
platform_mb.c	113
platform_ppc.c	113
sfp.c	113
si5324.c	113
TARGETC_RegisterMap.c	114
TARGETC_RegisterMap.h	117
time_hm.c	138
time_hm.h	142
transfer_function.c	146
transfer_function.h	148
udp_peripheral.c	150
udp_peripheral.h	159
utility.c	165
utility.h	166

Chapter 3

Data Structure Documentation

3.1 `coordinates_st` Struct Reference

Structure containing the coordinations of a point.

```
#include <data_analysis.h>
```

Data Fields

- float [x](#)
- float [y](#)

3.1.1 Detailed Description

Structure containing the coordinations of a point.

3.1.2 Field Documentation

3.1.2.1 `x`

```
float x
```

Coordination X

3.1.2.2 `y`

```
float y
```

Coordination Y

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

- [data_analysis.h](#)

3.2 data_axi_st Struct Reference

Structure of an element of the list which represent a window.

```
#include <data_analysis.h>
```

Data Fields

- uint64_t [wdo_time](#)
- uint64_t [PL_spare](#)
- uint32_t [info](#)
- uint32_t [wdo_id](#)
- uint32_t [data](#) [16][32]

3.2.1 Detailed Description

Structure of an element of the list which represent a window.

3.2.2 Field Documentation

3.2.2.1 data

```
uint32_t data[16][32]
```

Voltage measured by every sample

3.2.2.2 info

```
uint32_t info
```

Information about the window, bits 0-3 TRIG bits | bits 4-7 LAST bits | bits 8-11 TOO_LONG bits (use the defines to access correctly these bits)

3.2.2.3 PL_spare

```
uint64_t PL_spare
```

Spare bits for the development used to return the command send to the round buffer

3.2.2.4 wdo_id

```
uint32_t wdo_id
```

ID of the window (0 to 511)

3.2.2.5 wdo_time

```
uint64_t wdo_time
```

Timestampe of the window

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

- [data_analysis.h](#)

3.3 data_axi_union Union Reference

Union to access an element as an array, used to give the element's address to the DMA.

```
#include <data_analysis.h>
```

Data Fields

- struct [data_axi_st](#) [data_struct](#)
- uint32_t [data_array](#) [[SIZE_DATA_ARRAY](#)]

3.3.1 Detailed Description

Union to access an element as an array, used to give the element's address to the DMA.

3.3.2 Field Documentation

3.3.2.1 data_array

```
uint32_t data_array[SIZE\_DATA\_ARRAY]
```

Array of same size, pointer passed to DMA

3.3.2.2 data_struct

```
struct data\_axi\_st data_struct
```

Structure of the element

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

- [data_analysis.h](#)

3.4 data_list_st Struct Reference

```
#include <data_analysis.h>
```

Data Fields

- [data_axi_un data](#)
- [data_list * previous](#)
- [data_list * next](#)

3.4.1 Field Documentation

3.4.1.1 data

```
data_axi_un data
```

The element

3.4.1.2 next

```
data_list* next
```

Pointer on the next element (NULL if this is the last one)

3.4.1.3 previous

```
data_list* previous
```

Pointer on the previous element (NULL if this is the first one)

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

- [data_analysis.h](#)

3.5 features_ext_st Struct Reference

Structure containing the features extracted from a normal pulse.

```
#include <data_analysis.h>
```

Data Fields

- int [amplitude](#)
- [time_un](#) time

3.5.1 Detailed Description

Structure containing the features extracted from a normal pulse.

3.5.2 Field Documentation

3.5.2.1 amplitude

```
int amplitude
```

Amplitude maximum of the pulse

3.5.2.2 time

```
time\_un time
```

Time when the signal was a 20% of the maximum amplitude

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

- [data_analysis.h](#)

3.6 GMtype_CoordCartesian Struct Reference

```
#include <GMHeader.h>
```

Data Fields

- double [x](#)
- double [y](#)
- double [z](#)

3.6.1 Field Documentation

3.6.1.1 x

double x

3.6.1.2 y

double y

3.6.1.3 z

double z

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

- [GMHeader.h](#)

3.7 GMtype_CoordDipole Struct Reference

```
#include <GMHeader.h>
```

Data Fields

- double [lambda](#)
- double [phi](#)

3.7.1 Field Documentation

3.7.1.1 lambda

double lambda

3.7.1.2 phi

double phi

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

- [GMHeader.h](#)

3.8 GMtype_CoordGeodetic Struct Reference

```
#include <GMHeader.h>
```

Data Fields

- double [lambda](#)
- double [phi](#)
- double [HeightAboveEllipsoid](#)

3.8.1 Field Documentation

3.8.1.1 HeightAboveEllipsoid

```
double HeightAboveEllipsoid
```

3.8.1.2 lambda

```
double lambda
```

3.8.1.3 phi

```
double phi
```

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

- [GMHeader.h](#)

3.9 GMtype_CoordSpherical Struct Reference

```
#include <GMHeader.h>
```

Data Fields

- double [r](#)
- double [phig](#)
- double [lambda](#)

3.9.1 Field Documentation

3.9.1.1 lambda

```
double lambda
```

3.9.1.2 phig

```
double phig
```

3.9.1.3 r

```
double r
```

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

- [GMHeader.h](#)

3.10 GMtype_Data Struct Reference

```
#include <GMHeader.h>
```

Data Fields

- int [size](#)
- double [element](#) [30]

3.10.1 Field Documentation

3.10.1.1 element

```
double element[30]
```


3.10.1.2 size

```
int size
```

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

- [GMHeader.h](#)

3.11 GMtype_Date Struct Reference

```
#include <GMHeader.h>
```

Data Fields

- int [Day](#)
- int [Month](#)
- int [Year](#)
- double [DecimalYear](#)
- int [DayNumber](#)

3.11.1 Field Documentation

3.11.1.1 Day

```
int Day
```

3.11.1.2 DayNumber

```
int DayNumber
```

3.11.1.3 DecimalYear

```
double DecimalYear
```

3.11.1.4 Month

```
int Month
```

3.11.1.5 Year

```
int Year
```

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

- [GMHeader.h](#)

3.12 GMtype_Ellipsoid Struct Reference

```
#include <GMHeader.h>
```

Data Fields

- double [a](#)
- double [b](#)
- double [fla](#)
- double [epssq](#)
- double [eps](#)
- double [re](#)

3.12.1 Field Documentation

3.12.1.1 a

```
double a
```

3.12.1.2 b

```
double b
```

3.12.1.3 eps

double eps

3.12.1.4 epssq

double epssq

3.12.1.5 fla

double fla

3.12.1.6 re

double re

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

- [GMHeader.h](#)

3.13 GMtype_Matrix Struct Reference

```
#include <GMHeader.h>
```

Data Fields

- int [rows](#)
- int [columns](#)
- double [element](#) [30][30]

3.13.1 Field Documentation

3.13.1.1 columns

int columns

3.13.1.2 element

```
double element[30][30]
```

3.13.1.3 rows

```
int rows
```

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

- [GMHeader.h](#)

3.14 GMtype_Model Struct Reference

```
#include <GMHeader.h>
```

Data Fields

- double [h1](#)
- double [g1](#)
- double [g0](#)

3.14.1 Field Documentation

3.14.1.1 g0

```
double g0
```

3.14.1.2 g1

```
double g1
```

3.14.1.3 h1

double h1

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

- [GMHeader.h](#)

3.15 GMtype_Pole Struct Reference

```
#include <GMHeader.h>
```

Data Fields

- double [M](#)
- double [phi](#)
- double [lambda](#)

3.15.1 Field Documentation

3.15.1.1 lambda

double lambda

3.15.1.2 M

double M

3.15.1.3 phi

double phi

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

- [GMHeader.h](#)

3.16 GMtype_Polynomial Struct Reference

```
#include <GMHeader.h>
```

Data Fields

- int [degree](#)
- double [coef](#) [30]

3.16.1 Field Documentation

3.16.1.1 coef

```
double coef[30]
```

3.16.1.2 degree

```
int degree
```

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

- [GMHeader.h](#)

3.17 time_cplt_st Struct Reference

Structure of the timestamp.

```
#include <time_hm.h>
```

Data Fields

- int [year](#)
- int [month](#)
- int [day](#)
- int [hour](#)
- int [minute](#)
- int [second](#)
- int [millisecond](#)

3.17.1 Detailed Description

Structure of the timestamp.

3.17.2 Field Documentation

3.17.2.1 day

```
int day
```

Contain the day of the month

3.17.2.2 hour

```
int hour
```

Contain the hour (in 24h hour mode)

3.17.2.3 millisecond

```
int millisecond
```

Contain the millisecond

3.17.2.4 minute

```
int minute
```

Contain the minutes

3.17.2.5 month

```
int month
```

Contain the month

3.17.2.6 second

```
int second
```

Contain the second

3.17.2.7 year

```
int year
```

Contain the year

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

- [time_hm.h](#)

3.18 time_union Union Reference

Union to convert access a float as an int without converting it in int.

```
#include <data_analysis.h>
```

Data Fields

- float [time_fl](#)
- int [time_t](#)

3.18.1 Detailed Description

Union to convert access a float as an int without converting it in int.

3.18.2 Field Documentation

3.18.2.1 time_fl

```
float time_fl
```

The time in float

3.18.2.2 time_t

```
int time_t
```

The same time in float but considerate as int (without conversion)

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

- [data_analysis.h](#)

3.19 TmrCntSetup_st Struct Reference

Structure containing all the settings to set up the Triple Timer Counter.

```
#include <interrupt.h>
```

Data Fields

- u32 [OutputHz](#)
- XInterval [Interval](#)
- u8 [Prescaler](#)
- u32 [Options](#)

3.19.1 Detailed Description

Structure containing all the settings to set up the Triple Timer Counter.

3.19.2 Field Documentation

3.19.2.1 Interval

```
XInterval Interval
```

Interval value

3.19.2.2 Options

```
u32 Options
```

Option settings

3.19.2.3 OutputHz

```
u32 OutputHz
```

Output frequency

3.19.2.4 Prescaler

```
u8 Prescaler
```

Prescaler value

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

- [interrupt.h](#)

Chapter 4

File Documentation

4.1 axis_peripheral.c File Reference

```
#include "axis_peripheral.h"
```

Functions

- void [XAxiDma_SimpleTransfer_hm](#) (UINTPTR BuffAddr, int LengthOfBytes)
Pass the address to the DMA device with the number of bytes asked.
- void [dma_received_data](#) (int pmt)
Process a complete waveform received by the DMA in trigger mode.
- int [test_TPG](#) (void)
Test the TARGET with the Test Pattern Generator.

Variables

- [data_list](#) * [first_element](#)
Pointer on the first element of the list used in trigger mode.
- [data_list](#) * [last_element](#)
Pointer on the last element of the list used in trigger mode.
- char * [frame_buf](#)
Buffer used to send the data (50 bytes above it reserved for protocol header)
- int * [regptr](#)
Array containing registers of AXI-lite.
- volatile bool [flag_axidma_error](#)
Flag raised when AXI-DMA has an error.
- volatile bool [flag_axidma_rx_done](#)
Flag raised when AXI-DMA has finished an transfer, in OnDemand mode.
- volatile bool [empty_flag](#)
Flag true when the list is empty (first_element = last_element)
- volatile bool [flag_ttcps_timer](#)
Flag raised when the Triple Timer Counter overflows.
- volatile bool [flag_scu_timer](#)
Flag raised when the SCU timer overflows.
- XScuWdt [WdtScuInstance](#)
Instance of the device watchdog.

4.1.1 Detailed Description

Author

Anthony Schluchin

Date

24th October 2018

Version

0.0

4.1.2 Function Documentation

4.1.2.1 dma_received_data()

```
void dma_received_data (
    int pmt )
```

Process a complete waveform received by the DMA in trigger mode.

Parameters

<i>pmt</i>	ID of the pmt which triggered a pulse
------------	---------------------------------------

Returns

-

Note

-

4.1.2.2 test_TPG()

```
int test_TPG (
    void )
```

Test the TARGET with the Test Pattern Generator.

Parameters

-	
---	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.1.2.3 XAxiDma_SimpleTransfer_hm()

```
void XAxiDma_SimpleTransfer_hm (
    UINTPTR BuffAddr,
    int LengthOfBytes )
```

Pass the address to the DMA device with the number of bytes asked.

Parameters

<i>BuffAddr</i>	pointer on array to return the data
<i>LengthOfBytes</i>	Number of bytes to write in array

Returns

-

Note

-

4.1.3 Variable Documentation**4.1.3.1 empty_flag**

```
volatile bool empty_flag
```

Flag true when the list is empty (first_element = last_element)

4.1.3.2 first_element

```
data_list* first_element
```

Pointer on the first element of the list used in trigger mode.

4.1.3.3 flag_axidma_error

```
volatile bool flag_axidma_error
```

Flag raised when AXI-DMA has an error.

4.1.3.4 flag_axidma_rx_done

```
volatile bool flag_axidma_rx_done
```

Flag raised when AXI-DMA has finished an transfer, in OnDemand mode.

4.1.3.5 flag_scu_timer

```
volatile bool flag_scu_timer
```

Flag raised when the SCU timer overflows.

4.1.3.6 flag_ttcps_timer

```
volatile bool flag_ttcps_timer
```

Flag raised when the Triple Timer Counter overflows.

4.1.3.7 frame_buf

```
char* frame_buf
```

Buffer used to send the data (50 bytes above it reserved for protocol header)

4.1.3.8 last_element

```
data_list* last_element
```

Pointer on the last element of the list used in trigger mode.

4.1.3.9 regptr

```
int* regptr
```

Array containing registers of AXI-lite.

4.1.3.10 WdtScuInstance

```
XScuWdt WdtScuInstance
```

Instance of the device watchdog.

4.2 axis_peripheral.h File Reference

```
#include <stdio.h>
#include <unistd.h>
#include <errno.h>
#include <stdlib.h>
#include "xaxidma.h"
#include "xparameters.h"
#include "interrupt.h"
#include "xtime_l.h"
#include "xstatus.h"
#include "data_analysis.h"
#include "TARGETC_RegisterMap.h"
#include "file_hm.h"
#include "xscuwdt.h"
```

Macros

- `#define FEATURES_ID 0`
Data frame contains extracted features (for the receiver)
- `#define FULL_WAVEFORM_ID 1`
Data frame contains the full waveform (for the receiver)

Functions

- void [XAxiDma_SimpleTransfer_hm](#) (UINTPTR BuffAddr, int LengthOfBytes)
Pass the address to the DMA device with the number of bytes asked.
- void [dma_received_data](#) (int pmt)
Process a complete waveform received by the DMA in trigger mode.
- int [test_TPG](#) (void)
Test the TARGET with the Test Pattern Generator.

4.2.1 Detailed Description

Author

Anthony Schluchin

Date

24th October 2018

Version

0.0

4.2.2 Macro Definition Documentation

4.2.2.1 FEATURES_ID

```
#define FEATURES_ID 0
```

Data frame contains extracted features (for the receiptier)

4.2.2.2 FULL_WAVEFORM_ID

```
#define FULL_WAVEFORM_ID 1
```

Data frame contains the full waveform (for the receiptier)

4.2.3 Function Documentation

4.2.3.1 dma_received_data()

```
void dma_received_data (  
    int pmt )
```

Process a complete waveform received by the DMA in trigger mode.

Parameters

<i>pmt</i>	ID of the pmt which triggered a pulse
------------	---------------------------------------

Returns

-

Note

-

4.2.3.2 test_TPG()

```
int test_TPG (
    void )
```

Test the TARGET with the Test Pattern Generator.

Parameters

-	
---	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.2.3.3 XAxiDma_SimpleTransfer_hm()

```
void XAxiDma_SimpleTransfer_hm (
    UINTPTR BuffAddr,
    int LengthOfBytes )
```

Pass the address to the DMA device with the number of bytes asked.

Parameters

<i>BuffAddr</i>	pointer on array to return the data
<i>LengthOfBytes</i>	Number of bytes to write in array

Returns

-

Note

-

4.3 data_analysis.c File Reference

```
#include "data_analysis.h"
```

Functions

- int [correct_data](#) (uint16_t *data, int pmt, char nbr_wdo, uint32_t *info, [data_list](#) *tmp_first_element)
Correct the data received from the PL side (pedestal subtraction & transfer function correction) and choose the gain stage (channel)
- void [extract_features](#) (uint16_t *data, int length, [features_ext](#) *features)
Extract the minimum amplitude of the pulse, and the time when it was 20% of its value.

Variables

- uint16_t [pedestal](#) [512][16][32]
Array containing the pedestal correction for every sample.
- uint16_t [lookup_table](#) [2048]
Lookup table to correct the transfer function.

4.3.1 Function Documentation

4.3.1.1 correct_data()

```
int correct_data (
    uint16_t * data,
    int pmt,
    char nbr_wdo,
    uint32_t * info,
    data_list * tmp_first_element )
```

Correct the data received from the PL side (pedestal subtraction & transfer function correction) and choose the gain stage (channel)

Parameters

<i>data</i>	pointer on array to return the data corrected
<i>pmt</i>	PMT's ID (4 PMTs per ASIC, 16 ch per ASIC -> 4 ch per PMT)
<i>nbr_wdo</i>	number of window for the pulse
<i>info</i>	pointer to return info of pulse too long or not

Returns

ch: channel chose (gain stage chose)

Note

-

4.3.1.2 extract_features()

```
void extract_features (
    uint16_t * data,
    int length,
    features_ext * features )
```

Extract the minimum amplitude of the pulse, and the time when it was 20% of its value.

Parameters

<i>data</i>	pointer on the pulse's data
<i>length</i>	size of data
<i>features</i>	pointer on structure to return the amplitude and the time

Returns

None

Note

-

4.3.2 Variable Documentation**4.3.2.1 lookup_table**

```
uint16_t lookup_table[2048]
```

Lookup table to correct the transfer function.

4.3.2.2 pedestal

```
uint16_t pedestal[512][16][32]
```

Array containing the pedestal correction for every sample.

4.4 data_analysis.h File Reference

```
#include <math.h>
#include <stdbool.h>
#include "xtime_l.h"
#include "axis_peripheral.h"
```

Data Structures

- union [time_union](#)
Union to convert access a float as an int without converting it in int.
- struct [features_ext_st](#)
Structure containing the features extracted from a normal pulse.
- struct [coordinates_st](#)
Structure containing the coordinations of a point.
- struct [data_axi_st](#)
Structure of an element of the list which represent a window.
- union [data_axi_union](#)
Union to access an element as an array, used to give the element's address to the DMA.
- struct [data_list_st](#)

Macros

- #define [SIZE_DATA_ARRAY](#) 518
*Size of an element given to the DMA for a transfer (32samples * 16ch + header = 518 int32_t)*
- #define [SIZE_DATA_ARRAY_BYT](#) [SIZE_DATA_ARRAY](#)*4
SIZE_DATA_ARRAY but in bytes (int32_t is 4 bytes)
- #define [CHANNEL](#) 16
Number of channels per TARGET C.
- #define [SAMPLE](#) 32
Number of sample per window.
- #define [MAX_WINDOW](#) 4
Maximum number of window to represent a pulse.
- #define [TRIG_SHIFT](#) 0
Position of TRIG bit in window's information.
- #define [LAST_SHIFT](#) 4
Position of LAST bit in window's information.
- #define [TOO_LONG_SHIFT](#) 8
Position of TOO_LONG bit in window's information.
- #define [MASK_INFO](#) 0xF
Mask for one of the groupe of bit in window's information.
- #define [VPED_DIGITAL](#) 1024
Vped value, but in digital (2.5V <=> 2047 -> 2V <=> 1638 or 1.25V <=> 1024)
- #define [VPED_ANALOG](#) 1.25
Vped voltage to set up in DAC (in float)
- #define [THRESHOLD_CMP](#) 1.75
Threshold voltage to set up in DAC for the comparator of the trigger system (in float)
- #define [THRESHOLD_PULSE](#) 500
Treshold used to select the gain stage in function correct_data to send (digital value)

Typedefs

- typedef union [time_union](#) [time_un](#)
Union to convert access a float as an int without converting it in int.
- typedef struct [features_ext_st](#) [features_ext](#)
Structure containing the features extracted from a normal pulse.
- typedef struct [coordinates_st](#) [coordinates](#)
Structure containing the coordinations of a point.
- typedef struct [data_axi_st](#) [data_axi](#)
Structure of an element of the list which represent a window.
- typedef union [data_axi_union](#) [data_axi_un](#)
Union to access an element as an array, used to give the element's address to the DMA.
- typedef struct [data_list_st](#) [data_list](#)
Structure to create the list of the element.

Functions

- int [correct_data](#) (uint16_t *data, int pmt, char nbr_wdo, uint32_t *info, [data_list](#) *tmp_first_element)
Correct the data received from the PL side (pedestal subtraction & transfer function correction) and choose the gain stage (channel)
- void [extract_features](#) (uint16_t *data, int length, [features_ext](#) *features)
Extract the minimum amplitude of the pulse, and the time when it was 20% of its value.

4.4.1 Macro Definition Documentation

4.4.1.1 CHANNEL

```
#define CHANNEL 16
```

Number of channels per TARGET C.

4.4.1.2 LAST_SHIFT

```
#define LAST_SHIFT 4
```

Position of LAST bit in window's information.

4.4.1.3 MASK_INFO

```
#define MASK_INFO 0xF
```

Mask for one of the groupe of bit in window's information.

4.4.1.4 MAX_WINDOW

```
#define MAX_WINDOW 4
```

Maximum number of window to represent a pulse.

4.4.1.5 SAMPLE

```
#define SAMPLE 32
```

Number of sample per window.

4.4.1.6 SIZE_DATA_ARRAY

```
#define SIZE_DATA_ARRAY 518
```

Size of an element given to the DMA for a transfer (32samples * 16ch + header = 518 int32_t)

4.4.1.7 SIZE_DATA_ARRAY_BYT

```
#define SIZE_DATA_ARRAY_BYT SIZE\_DATA\_ARRAY*4
```

SIZE_DATA_ARRAY but in bytes (int32_t is 4 bytes)

4.4.1.8 THRESHOLD_CMP

```
#define THRESHOLD_CMP 1.75
```

Threshold voltage to set up in DAC for the comparator of the trigger system (in float)

4.4.1.9 THRESHOLD_PULSE

```
#define THRESHOLD_PULSE 500
```

Threshold used to select the gain stage in function correct_data to send (digital value)

4.4.1.10 TOO_LONG_SHIFT

```
#define TOO_LONG_SHIFT 8
```

Position of TOO_LONG bit in window's information.

4.4.1.11 TRIG_SHIFT

```
#define TRIG_SHIFT 0
```

Position of TRIG bit in window's information.

4.4.1.12 VPED_ANALOG

```
#define VPED_ANALOG 1.25
```

Vped voltage to set up in DAC (in float)

4.4.1.13 VPED_DIGITAL

```
#define VPED_DIGITAL 1024
```

Vped value, but in digital (2.5V \Leftrightarrow 2047 \rightarrow 2V \Leftrightarrow 1638 or 1.25V \Leftrightarrow 1024)

4.4.2 Typedef Documentation

4.4.2.1 coordinates

```
typedef struct coordinates_st coordinates
```

Structure containing the coordinations of a point.

4.4.2.2 data_axi

```
typedef struct data_axi_st data_axi
```

Structure of an element of the list which represent a window.

4.4.2.3 data_axi_un

```
typedef union data_axi_union data_axi_un
```

Union to access an element as an array, used to give the element's address to the DMA.

4.4.2.4 data_list

```
typedef struct data_list_st data_list
```

Structure to create the list of the element.

4.4.2.5 features_ext

```
typedef struct features_ext_st features_ext
```

Structure containing the features extracted from a normal pulse.

4.4.2.6 time_un

```
typedef union time_union time_un
```

Union to convert access a float as an int without converting it in int.

4.4.3 Function Documentation

4.4.3.1 correct_data()

```
int correct_data (
    uint16_t * data,
    int pmt,
    char nbr_wdo,
    uint32_t * info,
    data_list * tmp_first_element )
```

Correct the data received from the PL side (pedestal subtraction & transfer function correction) and choose the gain stage (channel)

Parameters

<i>data</i>	pointer on array to return the data corrected
<i>pmt</i>	PMT's ID (4 PMTs per ASIC, 16 ch per ASIC -> 4 ch per PMT)
<i>nbr_wdo</i>	number of window for the pulse
<i>info</i>	pointer to return info of pulse too long or not

Returns

ch: channel chose (gain stage chose)

Note

-

4.4.3.2 extract_features()

```
void extract_features (
    uint16_t * data,
    int length,
    features_ext * features )
```

Extract the minimum amplitude of the pulse, and the time when it was 20% of its value.

Parameters

<i>data</i>	pointer on the pulse's data
<i>length</i>	size of data
<i>features</i>	pointer on structure to return the amplitude and the time

Returns

None

Note

-

4.5 data_test.c File Reference

```
#include "data_test.h"
```

Functions

- uint16_t [made_frame](#) (char stream[], uint16_t length)
Generate frame with random data.

4.5.1 Function Documentation

4.5.1.1 made_frame()

```
uint16_t made_frame (
    char stream[],
    uint16_t length )
```

Generate frame with random data.

Parameters

<i>stream</i>	pointer on array to return the frame length: maximal size of the frame
---------------	------------------------------------------------------------------------

Returns

Size of frame generated

Note

-

4.6 data_test.h File Reference

```
#include <stdio.h>
#include <string.h>
#include "lwip/err.h"
#include "lwip/udp.h"
#include "xil_printf.h"
```

Functions

- uint16_t [made_frame](#) (char stream[], uint16_t length)
Generate frame with random data.

4.6.1 Function Documentation

4.6.1.1 made_frame()

```
uint16_t made_frame (
    char stream[],
    uint16_t length )
```

Generate frame with random data.

Parameters

<i>stream</i>	pointer on array to return the frame length: maximal size of the frame
---------------	------------------------------------------------------------------------

Returns

Size of frame generated

Note

-

4.7 file_hm.c File Reference

```
#include "file_hm.h"
```

Functions

- FRESULT [mount_sd_card](#) (void)
Mount the SD card.
- FRESULT [create_logfile](#) (void)
Create the log file to save the log message.
- FRESULT [log_event](#) (char *tmp_text, uint length)
Write a log message in the log file with the date and hour.
- FRESULT [log_wdtevent](#) (void)
Write a message in the log file to indicate when the wdt occurs.
- FRESULT [create_timefile](#) (void)
Create the time file to save the time.
- FRESULT [update_timefile](#) (void)
Write the time in the time file to know when the wdt occurs.

Variables

- char * [Path](#) = "0:"
String pointer to the logical drive numer.

4.7.1 Function Documentation

4.7.1.1 create_logfile()

```
FRESULT create_logfile (
    void )
```

Create the log file to save the log message.

Parameters

<i>None</i>	
-------------	--

Returns

FRESULT: see enumeration in ff.h and possibility on note's website

Note

http://elm-chan.org/fsw/ff/00index_e.html for informations about ff.c (Generic FAT Filesystem Module)

4.7.1.2 create_timefile()

```
FRESULT create_timefile (
    void )
```

Create the time file to save the time.

Parameters

<i>None</i>	
-------------	--

Returns

FRESULT: see enumeration in ff.h and possibility on note's website

Note

http://elm-chan.org/fsw/ff/00index_e.html for informations about ff.c (Generic FAT Filesystem Module)

4.7.1.3 log_event()

```
FRESULT log_event (
    char * tmp_text,
    uint length )
```

Write a log message in the log file with the date and hour.

Parameters

<i>tmp_text</i>	text to be written length: size of tmp_text (in bytes)
-----------------	--------------------------------------------------------

Returns

FRESULT: see enumeration in ff.h and possibility on note's website

Note

http://elm-chan.org/fsw/ff/00index_e.html for informations about ff.c (Generic FAT Filesystem Module)

4.7.1.4 log_wdtevent()

```
FRESULT log_wdtevent (  
    void )
```

Write a message in the log file to indicate when the wdt occurs.

Parameters

None	
------	--

Returns

FRESULT: see enumeration in ff.h and possibility on note's website

Note

http://elm-chan.org/fsw/ff/00index_e.html for informations about ff.c (Generic FAT Filesystem Module)

4.7.1.5 mount_sd_card()

```
FRESULT mount_sd_card (  
    void )
```

Mount the SD card.

Parameters

None	
------	--

Returns

FRESULT: see enumeration in ff.h and possibility on note's website

Note

the instance of the SD card must be static, that is why this function job is only to call another function http://elm-chan.org/fsw/ff/00index_e.html for informations about ff.c (Generic FAT Filesystem Module)

4.7.1.6 update_timefile()

```
FRESULT update_timefile (
    void )
```

Write the time in the time file to know when the wdt occurs.

Parameters

None	
------	--

Returns

FRESULT: see enumeration in ff.h and possibility on note's website

Note

http://elm-chan.org/fsw/ff/00index_e.html for informations about ff.c (Generic FAT Filesystem Module)

4.7.2 Variable Documentation**4.7.2.1 Path**

```
char* Path = "0:/"
```

String pointer to the logical drive numer.

4.8 file_hm.h File Reference

```
#include <unistd.h>
#include <stdlib.h>
#include <stdbool.h>
#include <stdio.h>
#include "ff.h"
#include "time_hm.h"
```

Functions

- FRESULT [mount_sd_card](#) (void)
Mount the SD card.
- FRESULT [create_logfile](#) (void)
Create the log file to save the log message.
- FRESULT [log_event](#) (char *tmp_text, uint length)
Write a log message in the log file with the date and hour.
- FRESULT [log_wdtevent](#) (void)
Write a message in the log file to indicate when the wdt occurs.
- FRESULT [create_timefile](#) (void)
Create the time file to save the time.
- FRESULT [update_timefile](#) (void)
Write the time in the time file to know when the wdt occurs.

4.8.1 Function Documentation

4.8.1.1 [create_logfile\(\)](#)

```
FRESULT create_logfile (  
    void )
```

Create the log file to save the log message.

Parameters

<i>None</i>	
-------------	--

Returns

FRESULT: see enumeration in ff.h and possibility on note's website

Note

http://elm-chan.org/fsw/ff/00index_e.html for informations about ff.c (Generic FAT Filesystem Module)

4.8.1.2 [create_timefile\(\)](#)

```
FRESULT create_timefile (  
    void )
```

Create the time file to save the time.

Parameters

<i>None</i>	
-------------	--

Returns

FRESULT: see enumeration in ff.h and possibility on note's website

Note

http://elm-chan.org/fsw/ff/00index_e.html for informations about ff.c (Generic FAT Filesystem Module)

4.8.1.3 log_event()

```
FRESULT log_event (
    char * tmp_text,
    uint length )
```

Write a log message in the log file with the date and hour.

Parameters

<i>tmp_text</i>	text to be written length: size of tmp_text (in bytes)
-----------------	--------------------------------------------------------

Returns

FRESULT: see enumeration in ff.h and possibility on note's website

Note

http://elm-chan.org/fsw/ff/00index_e.html for informations about ff.c (Generic FAT Filesystem Module)

4.8.1.4 log_wdtevent()

```
FRESULT log_wdtevent (
    void )
```

Write a message in the log file to indicate when the wdt occurs.

Parameters

<i>None</i>	
-------------	--

Returns

FRESULT: see enumeration in ff.h and possibility on note's website

Note

http://elm-chan.org/fsw/ff/00index_e.html for informations about ff.c (Generic FAT Filesystem Module)

4.8.1.5 mount_sd_card()

```
FRESULT mount_sd_card (  
    void )
```

Mount the SD card.

Parameters

None	
------	--

Returns

FRESULT: see enumeration in ff.h and possibility on note's website

Note

the instance of the SD card must be static, that is why this function job is only to call another function http://elm-chan.org/fsw/ff/00index_e.html for informations about ff.c (Generic FAT Filesystem Module)

4.8.1.6 update_timefile()

```
FRESULT update_timefile (  
    void )
```

Write the time in the time file to know when the wdt occurs.

Parameters

None	
------	--

Returns

FRESULT: see enumeration in ff.h and possibility on note's website

Note

http://elm-chan.org/fsw/ff/00index_e.html for informations about ff.c (Generic FAT Filesystem Module)

4.9 get_20_windows.c File Reference

```
#include "get_20_windows.h"
```

Functions

- int [get_20_windows_fct](#) (void)
Recover 20 consecutive windows and send them to the computer.

Variables

- int * [regptr](#)
Array containing registers of AXI-lite.
- volatile bool [flag_axidma_error](#)
Flag raised when AXI-DMA has an error.
- volatile bool [flag_axidma_rx_done](#)
Flag raised when AXI-DMA has finished an transfer, in OnDemand mode.
- uint16_t [pedestal](#) [512][16][32]
Array containing the pedestal correction for every sample.
- char * [frame_buf](#)
Buffer used to send the data (50 bytes above it reserved for protocol header)
- uint16_t [lookup_table](#) [2048]
Lookup table to correct the transfer function.
- volatile bool [flag_ttcps_timer](#)
Flag raised when the Triple Timer Counter overflows.
- volatile bool [flag_scu_timer](#)
Flag raised when the SCU timer overflows.
- XScuWdt [WdtScuInstance](#)
Instance of the device watchdog.

4.9.1 Function Documentation

4.9.1.1 get_20_windows_fct()

```
int get_20_windows_fct (
    void )
```

Recover 20 consecutive windows and send them to the computer.

Parameters

-	
---	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.9.2 Variable Documentation

4.9.2.1 flag_axidma_error

```
volatile bool flag_axidma_error
```

Flag raised when AXI-DMA has an error.

4.9.2.2 flag_axidma_rx_done

```
volatile bool flag_axidma_rx_done
```

Flag raised when AXI-DMA has finished an transfer, in OnDemand mode.

4.9.2.3 flag_scu_timer

```
volatile bool flag_scu_timer
```

Flag raised when the SCU timer overflows.

4.9.2.4 flag_ttcps_timer

```
volatile bool flag_ttcps_timer
```

Flag raised when the Triple Timer Counter overflows.

4.9.2.5 frame_buf

```
char* frame_buf
```

Buffer used to send the data (50 bytes above it reserved for protocol header)

4.9.2.6 lookup_table

```
uint16_t lookup_table[2048]
```

Lookup table to correct the transfer function.

4.9.2.7 pedestal

```
uint16_t pedestal[512][16][32]
```

Array containing the pedestal correction for every sample.

4.9.2.8 regptr

```
int* regptr
```

Array containing registers of AXI-lite.

4.9.2.9 WdtScuInstance

```
XScuWdt WdtScuInstance
```

Instance of the device watchdog.

4.10 get_20_windows.h File Reference

```
#include "xstatus.h"
#include "data_analysis.h"
#include "xil_types.h"
#include "axis_peripheral.h"
#include "TARGETC_RegisterMap.h"
#include "iic_DAC_LTC2657.h"
#include "udp_peripheral.h"
#include "file_hm.h"
#include "xscuwdt.h"
```

Functions

- int [get_20_windows_fct](#) (void)
Recover 20 consecutive windows and send them to the computer.

4.10.1 Function Documentation

4.10.1.1 get_20_windows_fct()

```
int get_20_windows_fct (
    void )
```

Recover 20 consecutive windows and send them to the computer.

Parameters

-	
---	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.11 get_transfer_fct.c File Reference

```
#include "get_transfer_fct.h"
```

Functions

- int [send_data_transfer_fct](#) (void)
Recover windows to plot the transfer function offline.

Variables

- int * [regptr](#)
Array containing registers of AXI-lite.
- volatile bool [flag_axidma_error](#)
Flag raised when AXI-DMA has an error.
- volatile bool [flag_axidma_rx_done](#)

- Flag raised when AXI-DMA has finished an transfer, in OnDemand mode.*

 - uint16_t [pedestal](#) [512][16][32]
Array containing the pedestal correction for every sample.
 - char * [frame_buf](#)
Buffer used to send the data (50 bytes above it reserved for protocol header)
 - uint16_t [lookup_table](#) [2048]
Lookup table to correct the transfer function.
 - volatile bool [flag_ttcps_timer](#)
Flag raised when the Triple Timer Counter overflows.
 - volatile bool [flag_scu_timer](#)
Flag raised when the SCU timer overflows.
 - XScuWdt [WdtScuInstance](#)
Instance of the device watchdog.

4.11.1 Function Documentation

4.11.1.1 send_data_transfer_fct()

```
int send_data_transfer_fct (
    void )
```

Recover windows to plot the transfer function offline.

Parameters

-	
---	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.11.2 Variable Documentation

4.11.2.1 flag_axidma_error

```
volatile bool flag_axidma_error
```

Flag raised when AXI-DMA has an error.

4.11.2.2 flag_axidma_rx_done

```
volatile bool flag_axidma_rx_done
```

Flag raised when AXI-DMA has finished an transfer, in OnDemand mode.

4.11.2.3 flag_scu_timer

```
volatile bool flag_scu_timer
```

Flag raised when the SCU timer overflows.

4.11.2.4 flag_ttcps_timer

```
volatile bool flag_ttcps_timer
```

Flag raised when the Triple Timer Counter overflows.

4.11.2.5 frame_buf

```
char* frame_buf
```

Buffer used to send the data (50 bytes above it reserved for protocol header)

4.11.2.6 lookup_table

```
uint16_t lookup_table[2048]
```

Lookup table to correct the transfer function.

4.11.2.7 pedestal

```
uint16_t pedestal[512][16][32]
```

Array containing the pedestal correction for every sample.

4.11.2.8 regptr

```
int* regptr
```

Array containing registers of AXI-lite.

4.11.2.9 WdtScuInstance

```
XScuWdt WdtScuInstance
```

Instance of the device watchdog.

4.12 get_transfer_fct.h File Reference

```
#include "xstatus.h"
#include "data_analysis.h"
#include "xil_types.h"
#include "axis_peripheral.h"
#include "TARGETC_RegisterMap.h"
#include "iic_DAC_LTC2657.h"
#include "udp_peripheral.h"
#include "file_hm.h"
#include "xscuwdt.h"
```

Functions

- int [send_data_transfer_fct](#) (void)
Recover windows to plot the transfer function offline.

4.12.1 Function Documentation

4.12.1.1 send_data_transfer_fct()

```
int send_data_transfer_fct (
    void )
```

Recover windows to plot the transfer function offline.

Parameters

-	
---	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.13 global.c File Reference

```
#include "global.h"
```

Functions

- int [init_global_var](#) (void)
Initiate all the global variables declared in [global.h](#) file.
- void [cleanup_global_var](#) (void)
Free memory from the mallocs done in function [initt_global_var](#).

Variables

- struct netif * [echo_netif](#)
Pointer on the network interface.
- volatile int [count_ttcps_timer](#)
Counter of the TTC.
- volatile int [count_scu_timer](#)
Counter of the SCU timer.
- volatile bool [run_flag](#)
Flag reset when the user send the command "stop uC".
- volatile bool [stream_flag](#)
Flag raised when the user send the command "start streaming".
- volatile bool [get_transfer_fct_flag](#)
Flag raised when the user send the command "get transfer function".
- volatile bool [get_20_windows_flag](#)
Flag raised when the user send the command "get 20 windows".
- volatile bool [empty_flag](#)
Flag true when the list is empty (first_element = last_element)
- volatile bool [flag_ttcps_timer](#)
Flag raised when the Triple Timer Counter overflows.
- volatile bool [flag_scu_timer](#)
Flag raised when the SCU timer overflows.
- XAxiDma [AxiDmaInstance](#)
Instance of AXI-DMA.
- XScuWdt [WdtScuInstance](#)
Instance of the device watchdog.
- volatile bool [flag_axidma_error](#)
Flag raised when AXI-DMA has an error.

- volatile bool `flag_axidma_rx_done`
Flag raised when AXI-DMA has finished an transfer, in OnDemand mode.
- int `flag_axidma_rx` [4]
Array of flag, one for each PMT.
- int `nbre_of_bytes`
Number of bytes sent during streaming (trigger mode)
- `data_list * first_element`
Pointer on the first element of the list used in trigger mode.
- `data_list * last_element`
Pointer on the last element of the list used in trigger mode.
- volatile bool `flag_assertion`
Flag raised when an assertion has occurred.
- volatile bool `flag_while_loop`
Flag raised when the program has entered the while loop.
- char * `frame_buf_tmp`
- char * `frame_buf`
Buffer used to send the data (50 bytes above it reserved for protocol header)
- char * `frame_buf_cmd_tmp`
- char * `frame_buf_cmd`
Buffer used to send the command (50 bytes above it reserved for protocol header)
- int * `regptr`
Array containing registers of AXI-lite.
- uint16_t `pedestal` [512][16][32]
Array containing the pedestal correction for every sample.
- uint16_t `lookup_table` [2048]
Lookup table to correct the transfer function.

4.13.1 Detailed Description

Author

Anthony Schluchin

Date

28th November 2018

Version

0.0

4.13.2 Function Documentation

4.13.2.1 `cleanup_global_var()`

```
void cleanup_global_var (
    void )
```

Free memory from the mallocs done in function `initt_global_var`.

Parameters

<i>None</i>	
-------------	--

Returns

None

Note

-

4.13.2.2 init_global_var()

```
int init_global_var (  
    void )
```

Initiate all the global variables declared in [global.h](#) file.

Parameters

<i>None</i>	
-------------	--

Returns

None

Note

-

4.13.3 Variable Documentation**4.13.3.1 AxiDmaInstance**

```
XAxiDma AxiDmaInstance
```

Instance of AXI-DMA.

4.13.3.2 count_scu_timer

```
volatile int count_scu_timer
```

Counter of the SCU timer.

Instance of AXI-DMA.

4.13.3.3 count_ttcps_timer

```
volatile int count_ttcps_timer
```

Counter of the TTC.

4.13.3.4 echo_netif

```
struct netif* echo_netif
```

Pointer on the network interface.

4.13.3.5 empty_flag

```
volatile bool empty_flag
```

Flag true when the list is empty (first_element = last_element)

4.13.3.6 first_element

```
data_list* first_element
```

Pointer on the first element of the list used in trigger mode.

4.13.3.7 flag_assertion

```
volatile bool flag_assertion
```

Flag raised when an assertion has occurred.

4.13.3.8 flag_axidma_error

```
volatile bool flag_axidma_error
```

Flag raised when AXI-DMA has an error.

4.13.3.9 flag_axidma_rx

```
int flag_axidma_rx[4]
```

Array of flag, one for each PMT.

4.13.3.10 flag_axidma_rx_done

```
volatile bool flag_axidma_rx_done
```

Flag raised when AXI-DMA has finished an transfer, in OnDemand mode.

4.13.3.11 flag_scu_timer

```
volatile bool flag_scu_timer
```

Flag raised when the SCU timer overflows.

4.13.3.12 flag_ttcps_timer

```
volatile bool flag_ttcps_timer
```

Flag raised when the Triple Timer Counter overflows.

4.13.3.13 flag_while_loop

```
volatile bool flag_while_loop
```

Flag raised when the program has entered the while loop.

4.13.3.14 frame_buf

```
char* frame_buf
```

Buffer used to send the data (50 bytes above it reserved for protocol header)

4.13.3.15 frame_buf_cmd

```
char* frame_buf_cmd
```

Buffer used to send the command (50 bytes above it reserved for protocol header)

4.13.3.16 frame_buf_cmd_tmp

```
char* frame_buf_cmd_tmp
```

4.13.3.17 frame_buf_tmp

```
char* frame_buf_tmp
```

4.13.3.18 get_20_windows_flag

```
volatile bool get_20_windows_flag
```

Flag raised when the user send the command "get 20 windows".

4.13.3.19 get_transfer_fct_flag

```
volatile bool get_transfer_fct_flag
```

Flag raised when the user send the command "get transfer function".

4.13.3.20 last_element

```
data_list* last_element
```

Pointer on the last element of the list used in trigger mode.

4.13.3.21 lookup_table

```
uint16_t lookup_table[2048]
```

Lookup table to correct the transfer function.

4.13.3.22 nbre_of_bytes

```
int nbre_of_bytes
```

Number of bytes sent during streaming (trigger mode)

4.13.3.23 pedestal

```
uint16_t pedestal[512][16][32]
```

Array containing the pedestal correction for every sample.

4.13.3.24 regptr

```
int* regptr
```

Array containing registers of AXI-lite.

4.13.3.25 run_flag

```
volatile bool run_flag
```

Flag reset when the user send the command "stop uC".

4.13.3.26 stream_flag

```
volatile bool stream_flag
```

Flag raised when the user send the command "start streaming".

4.13.3.27 WdtScuInstance

```
XScuWdt WdtScuInstance
```

Instance of the device watchdog.

4.14 global.h File Reference

```
#include <stdbool.h>
#include <inttypes.h>
#include "xaxidma.h"
#include "axis_peripheral.h"
#include "xparameters.h"
#include "TARGETC_RegisterMap.h"
#include "udp_peripheral.h"
```

Functions

- int [init_global_var](#) (void)
Initiate all the global variables declared in [global.h](#) file.
- void [cleanup_global_var](#) (void)
Free memory from the mallocs done in function [initt_global_var](#).

4.14.1 Detailed Description

Author

Anthony Schluchin

Date

28th November 2018

Version

0.0

4.14.2 Function Documentation

4.14.2.1 cleanup_global_var()

```
void cleanup_global_var (  
    void )
```

Free memory from the mallocs done in function [initt_global_var](#).

Parameters

None	
------	--

Returns

None

Note

-

4.14.2.2 init_global_var()

```
int init_global_var (
    void )
```

Initiate all the global variables declared in [global.h](#) file.

Parameters

None	
------	--

Returns

None

Note

-

4.15 GM_SubLibrary.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>
#include "GMHeader.h"
```

Functions

- void [GM_CartesianToSpherical](#) ([GMtype_CoordCartesian](#) CoordCartesian, [GMtype_CoordSpherical](#) *CoordSpherical)

- void `GM_CORD` (`GMtype_CoordGeodetic` location, `GMtype_Date` *date, `GMtype_Ellipsoid` Ellip, `GMtype_Data` g0d, `GMtype_Data` g1d, `GMtype_Data` h1d, `GMtype_CoordDipole` *CoordDipole)
- int `GM_DateToYear` (`GMtype_Date` *CalendarDate)
- void `GM_EarthCartToDipoleCartCD` (`GMtype_Pole` Pole, `GMtype_CoordCartesian` EarthCoord, `GMtype_CoordCartesian` *DipoleCoords)
- void `GM_GeodeticToSpherical` (`GMtype_Ellipsoid` Ellip, `GMtype_CoordGeodetic` CoordGeodetic, `GMtype_CoordSpherical` *CoordSpherical)
- void `GM_GetUserInput` (`GMtype_CoordGeodetic` *location, `GMtype_Date` *date)
- void `GM_PoleLocation` (`GMtype_Model` Model, `GMtype_Pole` *Pole)
- void `GM_PrintUserData` (`GMtype_CoordGeodetic` location, `GMtype_Date` date, `GMtype_CoordDipole` Dip↵ Location)
- void `GM_ScanIGRF` (`GMtype_Data` *G0, `GMtype_Data` *G1, `GMtype_Data` *H1)
- void `GM_SetEllipsoid` (`GMtype_Ellipsoid` *Ellip)
- void `GM_SphericalToCartesian` (`GMtype_CoordSpherical` CoordSpherical, `GMtype_CoordCartesian` *CoordCartesian)
- void `GM_TimeAdjustCoefs` (`GMtype_Date` Date, `GMtype_Data` g0d, `GMtype_Data` g1d, `GMtype_Data` h1d, `GMtype_Model` *Model)
- double `GM_DotProduct` (`GMtype_Data` VectorA, `GMtype_Data` VectorB)
- double `GM_LinearInterpolation` (double x1, double x2, double y1, double y2, double x)
- void `GM_LUDecomposition` (`GMtype_Matrix` A, `GMtype_Matrix` *L, `GMtype_Matrix` *U, `GMtype_Matrix` *P)
- void `GM_LUSolve` (`GMtype_Matrix` L, `GMtype_Matrix` U, `GMtype_Matrix` P, `GMtype_Matrix` *x, `GMtype_Matrix` b)
- double `GM_MatDet` (`GMtype_Matrix` Matrix)
- void `GM_MatInverse` (`GMtype_Matrix` Matrix, `GMtype_Matrix` *InvertedMatrix)
- void `GM_MatMultiply` (`GMtype_Matrix` MatrixA, `GMtype_Matrix` MatrixB, `GMtype_Matrix` *MatrixC)
- void `GM_MatTranspose` (`GMtype_Matrix` Matrix, `GMtype_Matrix` *TMatrix)
- double `GM_Mean` (`GMtype_Data` Data)
- void `GM_Median` (`GMtype_Data` Data, double *upper, double *lower)
- void `GM_PolyFit` (`GMtype_Data` DataX, `GMtype_Data` DataY, `GMtype_Polynomial` *Polynomial)
- double `GM_Pow` (double x, int y)
- void `GM_PrintMatrix` (`GMtype_Matrix` X)
- double `GM_SolvePolynomial` (`GMtype_Polynomial` Polynomial, double x)
- void `GM_Sort` (`GMtype_Data` *Data)
- double `GM_StandardDeviation` (`GMtype_Data` Data)
- void `GM_Swap` (double *x, double *y)
- void `GM_SwapRows` (`GMtype_Matrix` *Matrix, int Row1, int Row2)

4.15.1 Detailed Description

Author

ftp://ftp.ngdc.noaa.gov/geomag/Utilities/GM_SubLibrary.c

Date

16th January 2018

Version

0.0

4.15.2 Function Documentation

4.15.2.1 GM_CartesianToSpherical()

```
void GM_CartesianToSpherical (
    GMtype_CoordCartesian CoordCartesian,
    GMtype_CoordSpherical * CoordSpherical )
```

4.15.2.2 GM_CORD()

```
void GM_CORD (
    GMtype_CoordGeodetic location,
    GMtype_Date * date,
    GMtype_Ellipsoid Ellip,
    GMtype_Data g0d,
    GMtype_Data g1d,
    GMtype_Data h1d,
    GMtype_CoordDipole * CoordDipole )
```

4.15.2.3 GM_DateToYear()

```
int GM_DateToYear (
    GMtype_Date * CalendarDate )
```

4.15.2.4 GM_DotProduct()

```
double GM_DotProduct (
    GMtype_Data VectorA,
    GMtype_Data VectorB )
```

4.15.2.5 GM_EarthCartToDipoleCartCD()

```
void GM_EarthCartToDipoleCartCD (
    GMtype_Pole Pole,
    GMtype_CoordCartesian EarthCoord,
    GMtype_CoordCartesian * DipoleCoords )
```

4.15.2.6 GM_GeodeticToSpherical()

```
void GM_GeodeticToSpherical (
    GMtype_Ellipsoid Ellip,
    GMtype_CoordGeodetic CoordGeodetic,
    GMtype_CoordSpherical * CoordSpherical )
```

4.15.2.7 GM_GetUserInput()

```
void GM_GetUserInput (
    GMtype_CoordGeodetic * location,
    GMtype_Date * date )
```

4.15.2.8 GM_LinearInterpolation()

```
double GM_LinearInterpolation (
    double x1,
    double x2,
    double y1,
    double y2,
    double x )
```

4.15.2.9 GM_LUDecomposition()

```
void GM_LUDecomposition (
    GMtype_Matrix A,
    GMtype_Matrix * L,
    GMtype_Matrix * U,
    GMtype_Matrix * P )
```

4.15.2.10 GM_LUSolve()

```
void GM_LUSolve (
    GMtype_Matrix L,
    GMtype_Matrix U,
    GMtype_Matrix P,
    GMtype_Matrix * x,
    GMtype_Matrix b )
```

4.15.2.11 GM_MatDet()

```
double GM_MatDet (
    GMtype_Matrix Matrix )
```

4.15.2.12 GM_MatInverse()

```
void GM_MatInverse (
    GMtype_Matrix Matrix,
    GMtype_Matrix * InvertedMatrix )
```

4.15.2.13 GM_MatMultiply()

```
void GM_MatMultiply (
    GMtype_Matrix MatrixA,
    GMtype_Matrix MatrixB,
    GMtype_Matrix * MatrixC )
```

4.15.2.14 GM_MatTranspose()

```
void GM_MatTranspose (
    GMtype_Matrix Matrix,
    GMtype_Matrix * TMatrix )
```

4.15.2.15 GM_Mean()

```
double GM_Mean (
    GMtype_Data Data )
```

4.15.2.16 GM_Median()

```
void GM_Median (
    GMtype_Data Data,
    double * upper,
    double * lower )
```

4.15.2.17 GM_PoleLocation()

```
void GM_PoleLocation (
    GMtype_Model Model,
    GMtype_Pole * Pole )
```

4.15.2.18 GM_PolyFit()

```
void GM_PolyFit (
    GMtype_Data DataX,
    GMtype_Data DataY,
    GMtype_Polynomial * Polynomial )
```

4.15.2.19 GM_Pow()

```
double GM_Pow (
    double x,
    int y )
```

4.15.2.20 GM_PrintMatrix()

```
void GM_PrintMatrix (
    GMtype_Matrix X )
```

4.15.2.21 GM_PrintUserData()

```
void GM_PrintUserData (
    GMtype_CoordGeodetic location,
    GMtype_Date date,
    GMtype_CoordDipole DipLocation )
```

4.15.2.22 GM_ScanIGRF()

```
void GM_ScanIGRF (
    GMtype_Data * G0,
    GMtype_Data * G1,
    GMtype_Data * H1 )
```

4.15.2.23 GM_SetEllipsoid()

```
void GM_SetEllipsoid (
    GMtype_Ellipsoid * Ellip )
```

4.15.2.24 GM_SolvePolynomial()

```
double GM_SolvePolynomial (
    GMtype_Polynomial Polynomial,
    double x )
```

4.15.2.25 GM_Sort()

```
void GM_Sort (
    GMtype_Data * Data )
```

4.15.2.26 GM_SphericalToCartesian()

```
void GM_SphericalToCartesian (
    GMtype_CoordSpherical CoordSpherical,
    GMtype_CoordCartesian * CoordCartesian )
```

4.15.2.27 GM_StandardDeviation()

```
double GM_StandardDeviation (
    GMtype_Data Data )
```

4.15.2.28 GM_Swap()

```
void GM_Swap (
    double * x,
    double * y )
```

4.15.2.29 GM_SwapRows()

```
void GM_SwapRows (
    GMtype_Matrix * Matrix,
    int Row1,
    int Row2 )
```

4.15.2.30 GM_TimeAdjustCoefs()

```
void GM_TimeAdjustCoefs (
    GMtype_Date Date,
    GMtype_Data g0d,
    GMtype_Data g1d,
    GMtype_Data h1d,
    GMtype_Model * Model )
```

4.16 GMHeader.h File Reference

Data Structures

- struct [GMtype_Date](#)
- struct [GMtype_CoordGeodetic](#)
- struct [GMtype_CoordSpherical](#)
- struct [GMtype_CoordDipole](#)
- struct [GMtype_CoordCartesian](#)
- struct [GMtype_Ellipsoid](#)
- struct [GMtype_Polynomial](#)
- struct [GMtype_Model](#)
- struct [GMtype_Pole](#)
- struct [GMtype_Matrix](#)
- struct [GMtype_Data](#)

Macros

- #define [M_PI](#) ((2)*(acos(0.0)))
- #define [GM_STARTYEAR](#) 1900
- #define [RAD2DEG](#)(rad) ((rad)*(180.0L/M_PI))
- #define [DEG2RAD](#)(deg) ((deg)*(M_PI/180.0L))
- #define [ATanH](#)(x) (0.5 * log((1 + x) / (1 - x)))
- #define [MU_0](#) 4*M_PI / 10000000
- #define [R_e](#) 6.371 * 1000000
- #define [TRUE](#) ((int)1)
- #define [FALSE](#) ((int)0)

Functions

- void [GM_CartesianToSpherical](#) ([GMtype_CoordCartesian](#) CoordCartesian, [GMtype_CoordSpherical](#) *CoordSpherical)
- void [GM_CORD](#) ([GMtype_CoordGeodetic](#) location, [GMtype_Date](#) *date, [GMtype_Ellipsoid](#) Ellip, [GMtype_Data](#) g0d, [GMtype_Data](#) g1d, [GMtype_Data](#) h1d, [GMtype_CoordDipole](#) *CoordDipole)
- int [GM_DateToYear](#) ([GMtype_Date](#) *Date)
- void [GM_EarthCartToDipoleCartCD](#) ([GMtype_Pole](#) Pole, [GMtype_CoordCartesian](#) EarthCoord, [GMtype_CoordCartesian](#) *DipoleCoords)
- void [GM_GeodeticToSpherical](#) ([GMtype_Ellipsoid](#) Ellip, [GMtype_CoordGeodetic](#) CoordGeodetic, [GMtype_CoordSpherical](#) *CoordSpherical)
- void [GM_GetUserInput](#) ([GMtype_CoordGeodetic](#) *location, [GMtype_Date](#) *date)
- void [GM_PoleLocation](#) ([GMtype_Model](#) Model, [GMtype_Pole](#) *Pole)
- void [GM_PrintUserData](#) ([GMtype_CoordGeodetic](#) location, [GMtype_Date](#) date, [GMtype_CoordDipole](#) Dip↔ Location)
- void [GM_ScanIGRF](#) ([GMtype_Data](#) *G0, [GMtype_Data](#) *G1, [GMtype_Data](#) *H1)
- void [GM_SetEllipsoid](#) ([GMtype_Ellipsoid](#) *Ellip)
- void [GM_SphericalToCartesian](#) ([GMtype_CoordSpherical](#) CoordSpherical, [GMtype_CoordCartesian](#) *CoordCartesian)
- void [GM_TimeAdjustCoefs](#) ([GMtype_Date](#) Date, [GMtype_Data](#) g0d, [GMtype_Data](#) g1d, [GMtype_Data](#) h1d, [GMtype_Model](#) *Model)
- double [GM_DotProduct](#) ([GMtype_Data](#) VectorA, [GMtype_Data](#) VectorB)
- double [GM_LinearInterpolation](#) (double x1, double x2, double y1, double y2, double x)
- void [GM_LUDecomposition](#) ([GMtype_Matrix](#) A, [GMtype_Matrix](#) *L, [GMtype_Matrix](#) *U, [GMtype_Matrix](#) *P)
- void [GM_LUSolve](#) ([GMtype_Matrix](#) L, [GMtype_Matrix](#) U, [GMtype_Matrix](#) P, [GMtype_Matrix](#) *x, [GMtype_Matrix](#) b)
- double [GM_MatDet](#) ([GMtype_Matrix](#) Matrix)
- void [GM_MatMultiply](#) ([GMtype_Matrix](#) MatrixA, [GMtype_Matrix](#) MatrixB, [GMtype_Matrix](#) *MatrixC)
- void [GM_MatInverse](#) ([GMtype_Matrix](#) Matrix, [GMtype_Matrix](#) *InvertedMatrix)
- void [GM_MatTranspose](#) ([GMtype_Matrix](#) Matrix, [GMtype_Matrix](#) *TMatrix)
- double [GM_Mean](#) ([GMtype_Data](#) Data)
- void [GM_Median](#) ([GMtype_Data](#) Data, double *upper, double *lower)
- void [GM_PolyFit](#) ([GMtype_Data](#) DataX, [GMtype_Data](#) DataY, [GMtype_Polynomial](#) *Polynomial)
- double [GM_Pow](#) (double x, int y)
- void [GM_PrintMatrix](#) ([GMtype_Matrix](#) X)
- double [GM_SolvePolynomial](#) ([GMtype_Polynomial](#) Polynomial, double x)
- void [GM_Sort](#) ([GMtype_Data](#) *Data)
- double [GM_StandardDeviation](#) ([GMtype_Data](#) Data)
- void [GM_Swap](#) (double *x, double *y)
- void [GM_SwapRows](#) ([GMtype_Matrix](#) *Matrix, int Row1, int Row2)

4.16.1 Macro Definition Documentation

4.16.1.1 ATanH

```
#define ATanH(  
    x ) (0.5 * log((1 + x) / (1 - x)))
```

4.16.1.2 DEG2RAD

```
#define DEG2RAD(  
    deg ) ((deg)*(M_PI/180.0L))
```

4.16.1.3 FALSE

```
#define FALSE ((int)0)
```

4.16.1.4 GM_STARTYEAR

```
#define GM_STARTYEAR 1900
```

4.16.1.5 M_PI

```
#define M_PI ((2)*(acos(0.0)))
```

4.16.1.6 MU_0

```
#define MU_0 4*M_PI / 10000000
```

4.16.1.7 R_e

```
#define R_e 6.371 * 1000000
```

4.16.1.8 RAD2DEG

```
#define RAD2DEG(  
    rad ) ((rad)*(180.0L/M_PI))
```

4.16.1.9 TRUE

```
#define TRUE ((int)1)
```

4.16.2 Function Documentation

4.16.2.1 GM_CartesianToSpherical()

```
void GM_CartesianToSpherical (
    GMtype_CoordCartesian CoordCartesian,
    GMtype_CoordSpherical * CoordSpherical )
```

4.16.2.2 GM_CORD()

```
void GM_CORD (
    GMtype_CoordGeodetic location,
    GMtype_Date * date,
    GMtype_Ellipsoid Ellip,
    GMtype_Data g0d,
    GMtype_Data g1d,
    GMtype_Data h1d,
    GMtype_CoordDipole * CoordDipole )
```

4.16.2.3 GM_DateToYear()

```
int GM_DateToYear (
    GMtype_Date * Date )
```

4.16.2.4 GM_DotProduct()

```
double GM_DotProduct (
    GMtype_Data VectorA,
    GMtype_Data VectorB )
```

4.16.2.5 GM_EarthCartToDipoleCartCD()

```
void GM_EarthCartToDipoleCartCD (
    GMtype_Pole Pole,
    GMtype_CoordCartesian EarthCoord,
    GMtype_CoordCartesian * DipoleCoords )
```

4.16.2.6 GM_GeodeticToSpherical()

```
void GM_GeodeticToSpherical (
    GMtype_Ellipsoid Ellip,
    GMtype_CoordGeodetic CoordGeodetic,
    GMtype_CoordSpherical * CoordSpherical )
```

4.16.2.7 GM_GetUserInput()

```
void GM_GetUserInput (
    GMtype_CoordGeodetic * location,
    GMtype_Date * date )
```

4.16.2.8 GM_LinearInterpolation()

```
double GM_LinearInterpolation (
    double x1,
    double x2,
    double y1,
    double y2,
    double x )
```

4.16.2.9 GM_LUDecomposition()

```
void GM_LUDecomposition (
    GMtype_Matrix A,
    GMtype_Matrix * L,
    GMtype_Matrix * U,
    GMtype_Matrix * P )
```

4.16.2.10 GM_LUSolve()

```
void GM_LUSolve (
    GMtype_Matrix L,
    GMtype_Matrix U,
    GMtype_Matrix P,
    GMtype_Matrix * x,
    GMtype_Matrix b )
```

4.16.2.11 GM_MatDet()

```
double GM_MatDet (
    GMtype_Matrix Matrix )
```

4.16.2.12 GM_MatInverse()

```
void GM_MatInverse (
    GMtype_Matrix Matrix,
    GMtype_Matrix * InvertedMatrix )
```

4.16.2.13 GM_MatMultiply()

```
void GM_MatMultiply (
    GMtype_Matrix MatrixA,
    GMtype_Matrix MatrixB,
    GMtype_Matrix * MatrixC )
```

4.16.2.14 GM_MatTranspose()

```
void GM_MatTranspose (
    GMtype_Matrix Matrix,
    GMtype_Matrix * TMatrix )
```

4.16.2.15 GM_Mean()

```
double GM_Mean (
    GMtype_Data Data )
```

4.16.2.16 GM_Median()

```
void GM_Median (
    GMtype_Data Data,
    double * upper,
    double * lower )
```

4.16.2.17 GM_PoleLocation()

```
void GM_PoleLocation (
    GMtype_Model Model,
    GMtype_Pole * Pole )
```

4.16.2.18 GM_PolyFit()

```
void GM_PolyFit (
    GMtype_Data DataX,
    GMtype_Data DataY,
    GMtype_Polynomial * Polynomial )
```

4.16.2.19 GM_Pow()

```
double GM_Pow (
    double x,
    int y )
```

4.16.2.20 GM_PrintMatrix()

```
void GM_PrintMatrix (
    GMtype_Matrix X )
```

4.16.2.21 GM_PrintUserData()

```
void GM_PrintUserData (
    GMtype_CoordGeodetic location,
    GMtype_Date date,
    GMtype_CoordDipole DipLocation )
```

4.16.2.22 GM_ScanIGRF()

```
void GM_ScanIGRF (
    GMtype_Data * G0,
    GMtype_Data * G1,
    GMtype_Data * H1 )
```

4.16.2.23 GM_SetEllipsoid()

```
void GM_SetEllipsoid (
    GMtype_Ellipsoid * Ellip )
```

4.16.2.24 GM_SolvePolynomial()

```
double GM_SolvePolynomial (
    GMtype_Polynomial Polynomial,
    double x )
```

4.16.2.25 GM_Sort()

```
void GM_Sort (
    GMtype_Data * Data )
```

4.16.2.26 GM_SphericalToCartesian()

```
void GM_SphericalToCartesian (
    GMtype_CoordSpherical CoordSpherical,
    GMtype_CoordCartesian * CoordCartesian )
```

4.16.2.27 GM_StandardDeviation()

```
double GM_StandardDeviation (
    GMtype_Data Data )
```

4.16.2.28 GM_Swap()

```
void GM_Swap (
    double * x,
    double * y )
```

4.16.2.29 GM_SwapRows()

```
void GM_SwapRows (
    GMtype_Matrix * Matrix,
    int Row1,
    int Row2 )
```

4.16.2.30 GM_TimeAdjustCoefs()

```
void GM_TimeAdjustCoefs (
    GMtype_Date Date,
    GMtype_Data g0d,
    GMtype_Data g1d,
    GMtype_Data h1d,
    GMtype_Model * Model )
```

4.17 iic_DAC_LTC2657.c File Reference

```
#include "iic_DAC_LTC2657.h"
```

Functions

- int [DAC_LTC2657_initialize](#) (void)
Initialize the device I2C to communicate with the DAC.
- int [DAC_LTC2657_SetChannelVoltage](#) (int channel, float voltage)
Set the voltage of a channel of the DAC.

Variables

- Xlic [I2cInstance](#)
Instance of the I2C device.

4.17.1 Detailed Description

Author

Anthony Schluchin

Date

15th December 2018

Version

0.0

4.17.2 Function Documentation

4.17.2.1 DAC_LTC2657_initialize()

```
int DAC_LTC2657_initialize (
    void )
```

Initialize the device I2C to communicate with the DAC.

Parameters

-	
---	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.17.2.2 DAC_LTC2657_SetChannelVoltage()

```
int DAC_LTC2657_SetChannelVoltage (
    int channel,
    float voltage )
```

Set the voltage of a channel of the DAC.

Parameters

<i>channel</i>	which channel to change (see DEFINES)
<i>voltage</i>	voltage to set

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.17.3 Variable Documentation**4.17.3.1 I2cInstance**

XIic I2cInstance

Instance of the I2C device.

4.18 iic_DAC_LTC2657.h File Reference

```
#include "xparameters.h"
#include "xil_cache.h"
#include "xscugic.h"
#include "xil_printf.h"
#include "xtime_l.h"
#include "xiic.h"
#include "global.h"
```

Macros

- `#define IIC_DAC_LTC2657_H` /* by using protection macros */
- `#define IIC_DEVICE_ID` XPAR_AXI_IIC_0_DEVICE_ID
Base address for device I2C (from xparameters.h)
- `#define IIC_SLAVE_ADDRESS` 0x10
I2C address of the DAC.
- `#define CHANNEL_A` 0x00
Channel A register.
- `#define CHANNEL_B` 0x01
Channel B register.
- `#define CHANNEL_C` 0x02
Channel c register.

- #define `CHANNEL_D` 0x03
Channel D register.
- #define `CHANNEL_E` 0x04
Channel E register.
- #define `CHANNEL_F` 0x05
Channel F register.
- #define `CHANNEL_G` 0x06
Channel G register.
- #define `CHANNEL_H` 0x07
Channel H register.
- #define `CHANNEL_ALL` 0x0F
All channels register.
- #define `DAC_GRP_0` `CHANNEL_A`
Channel A register.
- #define `DAC_GRP_1` `CHANNEL_B`
Channel B register.
- #define `DAC_GRP_2` `CHANNEL_C`
Channel C register.
- #define `DAC_GRP_3` `CHANNEL_D`
Channel D register.
- #define `DAC_VPED` `CHANNEL_H`
Channel H register.

Functions

- int `DAC_LTC2657_initialize` ()
Initialize the device I2C to communicate with the DAC.
- int `DAC_LTC2657_SetChannelVoltage` (int channel, float voltage)
Set the voltage of a channel of the DAC.

4.18.1 Detailed Description

Author

Anthony Schluchin

Date

15th December 2018

Version

0.0

4.18.2 Macro Definition Documentation

4.18.2.1 CHANNEL_A

```
#define CHANNEL_A 0x00
```

Channel A register.

4.18.2.2 CHANNEL_ALL

```
#define CHANNEL_ALL 0x0F
```

All channels register.

4.18.2.3 CHANNEL_B

```
#define CHANNEL_B 0x01
```

Channel B register.

4.18.2.4 CHANNEL_C

```
#define CHANNEL_C 0x02
```

Channel c register.

4.18.2.5 CHANNEL_D

```
#define CHANNEL_D 0x03
```

Channel D register.

4.18.2.6 CHANNEL_E

```
#define CHANNEL_E 0x04
```

Channel E register.

4.18.2.7 CHANNEL_F

```
#define CHANNEL_F 0x05
```

Channel F register.

4.18.2.8 CHANNEL_G

```
#define CHANNEL_G 0x06
```

Channel G register.

4.18.2.9 CHANNEL_H

```
#define CHANNEL_H 0x07
```

Channel H register.

4.18.2.10 DAC_GRP_0

```
#define DAC_GRP_0 CHANNEL_A
```

Channel A register.

4.18.2.11 DAC_GRP_1

```
#define DAC_GRP_1 CHANNEL_B
```

Channel B register.

4.18.2.12 DAC_GRP_2

```
#define DAC_GRP_2 CHANNEL_C
```

Channel C register.

4.18.2.13 DAC_GRP_3

```
#define DAC_GRP_3 CHANNEL_D
```

Channel D register.

4.18.2.14 DAC_VPED

```
#define DAC_VPED CHANNEL_H
```

Channel H register.

4.18.2.15 IIC_DAC_LTC2657_H

```
#define IIC_DAC_LTC2657_H /* by using protection macros */
```

4.18.2.16 IIC_DEVICE_ID

```
#define IIC_DEVICE_ID XPAR_AXI_IIC_0_DEVICE_ID
```

Base address for device I2C (from xparameters.h)

4.18.2.17 IIC_SLAVE_ADDRESS

```
#define IIC_SLAVE_ADDRESS 0x10
```

I2C address of the DAC.

4.18.3 Function Documentation

4.18.3.1 DAC_LTC2657_initialize()

```
int DAC_LTC2657_initialize (  
    void )
```

Initialize the device I2C to communicate with the DAC.

Parameters

-	
---	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.18.3.2 DAC_LTC2657_SetChannelVoltage()

```
int DAC_LTC2657_SetChannelVoltage (
    int channel,
    float voltage )
```

Set the voltage of a channel of the DAC.

Parameters

<i>channel</i>	which channel to change (see DEFINES)
<i>voltage</i>	voltage to set

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.19 interrupt.c File Reference

```
#include "interrupt.h"
```

Functions

- void [assert_callback](#) (const char8 *File, s32 Line)
Callback for assertion.
- void [timer_scu_callback](#) (XScuTimer *TimerInstance)
Callback for the timer scu.

- void [timer_tcps_callback](#) (XTtcPs *TimerInstance)
Callback for the timer tcps.
- void [axidma_rx_callback](#) (XAxiDma *AxiDmaInst)
Callback when the dma finish a transfer.
- void [wdt_scu_callback](#) (XScuWdt *WdtInstance)
Callback for the timer wdt.
- int [setup_scu_timer_int](#) (void)
Setup the timer scu interrupt.
- int [setup_tcps_timer_int](#) (void)
Setup the timer tcps interrupt.
- int [setup_axidma_int](#) (void)
Setup the axidma interrupt.
- int [setup_scu_wdt_int](#) (void)
Setup the wdt interrupt.
- int [devices_initialization](#) ()
Initiate and setup all the interrupts.
- int [interrupts_initialization](#) (void)
Attach all the interrupt to the system and set the priority.
- void [enable_interrupts](#) ()
Enable the interrupts and start the timers.
- void [cleanup_interrupts](#) ()
Disable all the interrupts and stop the timers.

Variables

- volatile int [count_scu_timer](#)
Counter of the SCU timer.
- XAxiDma [AxiDmaInstance](#)
Instance of AXI-DMA.
- XScuWdt [WdtScuInstance](#)
Instance of the device watchdog.
- struct netif * [echo_netif](#)
Pointer on the network interface.
- volatile bool [flag_tcps_timer](#)
Flag raised when the Triple Timer Counter overflows.
- volatile bool [flag_scu_timer](#)
Flag raised when the SCU timer overflows.
- volatile bool [flag_assertion](#)
Flag raised when an assertion has occurred.
- volatile bool [flag_while_loop](#)
Flag raised when the program has entered the while loop.
- volatile bool [flag_axidma_error](#)
Flag raised when AXI-DMA has an error.
- volatile bool [flag_axidma_rx_done](#)
Flag raised when AXI-DMA has finished an transfer, in OnDemand mode.
- volatile bool [stream_flag](#)
Flag raised when the user send the command "start streaming".
- volatile bool [empty_flag](#)
Flag true when the list is empty (first_element = last_element)
- int [flag_axidma_rx](#) [4]
Array of flag, one for each PMT.
- data_list * [last_element](#)
Pointer on the last element of the list used in trigger mode.

4.19.1 Detailed Description

Author

Anthony Schluchin

Date

24th October 2018

Version

0.0

4.19.2 Function Documentation

4.19.2.1 assert_callback()

```
void assert_callback (
    const char8 * File,
    s32 Line )
```

Callback for assertion.

Parameters

<i>File</i>	filename where the assertion is called
<i>Line</i>	line number in the File

Returns

None

Note

When this callback is called, the filename and line number are stored in the log file, and then the program stops

4.19.2.2 axidma_rx_callback()

```
void axidma_rx_callback (
    XAxiDma * AxiDmaInst )
```

Callback when the dma finish a transfer.

Parameters

<i>AxiDmaInst</i>	pointer on the axidma's instance
-------------------	----------------------------------

Returns

None

Note

-

4.19.2.3 cleanup_interrupts()

```
void cleanup_interrupts ( )
```

Disable all the interrupts and stop the timers.

Parameters

<i>None</i>	
-------------	--

Returns

None

Note

-

4.19.2.4 devices_initialization()

```
int devices_initialization ( )
```

Initiate and setup all the interrupts.

Parameters

<i>None</i>	
-------------	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.19.2.5 enable_interrupts()

```
void enable_interrupts ( )
```

Enable the interrupts and start the timers.

Parameters

None	
------	--

Returns

None

Note

-

4.19.2.6 interrupts_initialization()

```
int interrupts_initialization (
    void )
```

Attach all the interrupt to the system and set the priority.

Parameters

None	
------	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.19.2.7 setup_axidma_int()

```
int setup_axidma_int (  
    void )
```

Setup the axidma interrupt.

Parameters

None	
------	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.19.2.8 setup_scu_timer_int()

```
int setup_scu_timer_int (  
    void )
```

Setup the timer scu interrupt.

Parameters

None	
------	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.19.2.9 setup_scu_wdt_int()

```
int setup_scu_wdt_int (  
    void )
```

Setup the wdt interrupt.

Parameters

<i>None</i>	
-------------	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.19.2.10 setup_ttcps_timer_int()

```
int setup_ttcps_timer_int (  
    void )
```

Setup the timer tcps interrupt.

Parameters

<i>None</i>	
-------------	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.19.2.11 timer_scu_callback()

```
void timer_scu_callback (  
    XScuTimer * TimerInstance )
```

Callback for the timer scu.

Parameters

<i>TimerInstance</i>	pointer on the timer's instance
----------------------	---------------------------------

Returns

None

Note

This callback is called every 250ms

4.19.2.12 timer_ttcps_callback()

```
void timer_ttcps_callback (
    XTtcPs * TimerInstance )
```

Callback for the timer tcps.

Parameters

<i>TimerInstance</i>	pointer on the timer's instance
----------------------	---------------------------------

Returns

None

Note

This callback is called every 1sec

4.19.2.13 wdt_scu_callback()

```
void wdt_scu_callback (
    XScuWdt * WdtInstance )
```

Callback for the timer wdt.

Parameters

<i>WdtInstance</i>	pointer on the timer's instance
--------------------	---------------------------------

Returns

None

Note

This callback is called only if the wdt is used has a timer (not the case here), used only the configure the wdt period of 1sec

4.19.3 Variable Documentation

4.19.3.1 AxiDmaInstance

```
XAxiDma AxiDmaInstance
```

Instance of AXI-DMA.

4.19.3.2 count_scu_timer

```
volatile int count_scu_timer
```

Counter of the SCU timer.

4.19.3.3 echo_netif

```
struct netif* echo_netif
```

Pointer on the network interface.

4.19.3.4 empty_flag

```
volatile bool empty_flag
```

Flag true when the list is empty (first_element = last_element)

4.19.3.5 flag_assertion

```
volatile bool flag_assertion
```

Flag raised when an assertion has occurred.

4.19.3.6 flag_axidma_error

```
volatile bool flag_axidma_error
```

Flag raised when AXI-DMA has an error.

4.19.3.7 flag_axidma_rx

```
int flag_axidma_rx[4]
```

Array of flag, one for each PMT.

4.19.3.8 flag_axidma_rx_done

```
volatile bool flag_axidma_rx_done
```

Flag raised when AXI-DMA has finished an transfer, in OnDemand mode.

4.19.3.9 flag_scu_timer

```
volatile bool flag_scu_timer
```

Flag raised when the SCU timer overflows.

4.19.3.10 flag_ttcps_timer

```
volatile bool flag_ttcps_timer
```

Flag raised when the Triple Timer Counter overflows.

4.19.3.11 flag_while_loop

```
volatile bool flag_while_loop
```

Flag raised when the program has entered the while loop.

4.19.3.12 last_element

```
data_list* last_element
```

Pointer on the last element of the list used in trigger mode.

4.19.3.13 stream_flag

```
volatile bool stream_flag
```

Flag raised when the user send the command "start streaming".

4.19.3.14 WdtScuInstance

```
XScuWdt WdtScuInstance
```

Instance of the device watchdog.

4.20 interrupt.h File Reference

```
#include "xparameters.h"
#include "xparameters_ps.h"
#include "xil_cache.h"
#include "xscugic.h"
#include "lwip/tcp.h"
#include "xil_printf.h"
#include "platform_config.h"
#include "netif/xadapter.h"
#include "xscutimer.h"
#include "xttcps.h"
#include "xaxidma.h"
#include "xscuwdt.h"
#include "udp_peripheral.h"
#include "axis_peripheral.h"
#include "xtime_l.h"
#include "file_hm.h"
```

Data Structures

- struct [TmrCntrSetup_st](#)

Structure containing all the settings to set up the Triple Timer Counter.

Macros

- `#define INTC_DEVICE_ID XPAR_SCUGIC_SINGLE_DEVICE_ID`
Number of device GIC (from xparameters.h)
- `#define TIMER_DEVICE_ID XPAR_SCUTIMER_DEVICE_ID`
Base address for device SCU timer (from xparameters.h)
- `#define INTC_BASE_ADDR XPAR_SCUGIC_0_CPU_BASEADDR`
Base address for device GIC (from xparameters.h)
- `#define INTC_DIST_BASE_ADDR XPAR_SCUGIC_0_DIST_BASEADDR`
Base address for device GIC (from xparameters.h)
- `#define TIMER_IRPT_INTR XPAR_SCUTIMER_INTR`
ID of SCU timer interrupt.
- `#define TTC_TICK_DEVICE_ID XPAR_XTTCPS_0_DEVICE_ID`
Number of device TTC (from xparameters.h)
- `#define TTC_TICK_INTR_ID XPAR_XTTCPS_0_INTR`
ID of TTC interrupt.
- `#define TTCPS_TIMER_FREQ_HZ 1`
Frequency for TTC.
- `#define WDT_DEVICE_ID XPAR_SCUWDT_0_DEVICE_ID`
Base address for device watchdog (from xparameters.h)
- `#define WDT_IRPT_INTR XPAR_SCUWDT_INTR`
ID of watchdog interrupt.
- `#define WDT_LOAD_VALUE 0x27FFFFFFE`
Value to load in watchdog's counter (= 2sec | 0x13FFFFFF = 1sec)
- `#define RESET_RX_CNTR_LIMIT 400`
Value for reset counter of lwIP connection.

Typedefs

- `typedef struct TmrCntrSetup_st TmrCntrSetup`
Structure containing all the settings to set up the Triple Timer Counter.

Functions

- void `assert_callback` (const char8 *File, s32 Line)
Callback for assertion.
- void `timer_scu_callback` (XScuTimer *TimerInstance)
Callback for the timer scu.
- void `timer_ttcps_callback` (XTtcPs *TimerInstance)
Callback for the timer tcps.
- void `axidma_rx_callback` (XAxiDma *AxiDmaInstance)
Callback when the dma finish a transfer.
- void `wdt_scu_callback` (XScuWdt *WdtInstance)
Callback for the timer wdt.
- int `setup_scu_timer_int` (void)
Setup the timer scu interrupt.
- int `setup_ttcps_timer_int` (void)
Setup the timer tcps interrupt.
- int `setup_axidma_int` (void)

- Setup the axidma interrupt.*
 - int [interrupts_initialization](#) (void)
Attach all the interrupt to the system and set the priority.
 - void [enable_interrupts](#) ()
Enable the interrupts and start the timers.
 - int [devices_initialization](#) ()
Initiate and setup all the interrupts.
 - void [cleanup_interrupts](#) ()
Disable all the interrupts and stop the timers.

4.20.1 Detailed Description

Author

Anthony Schluchin

Date

24th October 2018

Version

0.0

4.20.2 Macro Definition Documentation

4.20.2.1 INTC_BASE_ADDR

```
#define INTC_BASE_ADDR XPAR_SCUGIC_0_CPU_BASEADDR
```

Base address for device GIC (from xparameters.h)

4.20.2.2 INTC_DEVICE_ID

```
#define INTC_DEVICE_ID XPAR_SCUGIC_SINGLE_DEVICE_ID
```

Number of device GIC (from xparameters.h)

4.20.2.3 INTC_DIST_BASE_ADDR

```
#define INTC_DIST_BASE_ADDR XPAR_SCUGIC_0_DIST_BASEADDR
```

Base address for device GIC (from xparameters.h)

4.20.2.4 RESET_RX_CNTR_LIMIT

```
#define RESET_RX_CNTR_LIMIT 400
```

Value for reset counter of lwIP connection.

4.20.2.5 TIMER_DEVICE_ID

```
#define TIMER_DEVICE_ID XPAR_SCUTIMER_DEVICE_ID
```

Base address for device SCU timer (from xparameters.h)

4.20.2.6 TIMER_IRPT_INTR

```
#define TIMER_IRPT_INTR XPAR_SCUTIMER_INTR
```

ID of SCU timer interrupt.

4.20.2.7 TTC_TICK_DEVICE_ID

```
#define TTC_TICK_DEVICE_ID XPAR_XTTCPS_0_DEVICE_ID
```

Number of device TTC (from xparameters.h)

4.20.2.8 TTC_TICK_INTR_ID

```
#define TTC_TICK_INTR_ID XPAR_XTTCPS_0_INTR
```

ID of TTC interrupt.

4.20.2.9 TTCPS_TIMER_FREQ_HZ

```
#define TTCPS_TIMER_FREQ_HZ 1
```

Frequency for TTC.

4.20.2.10 WDT_DEVICE_ID

```
#define WDT_DEVICE_ID XPAR_SCUWDT_0_DEVICE_ID
```

Base address for device watchdog (from xparameters.h)

4.20.2.11 WDT_IRPT_INTR

```
#define WDT_IRPT_INTR XPAR_SCUWDT_INTR
```

ID of watchdog interrupt.

4.20.2.12 WDT_LOAD_VALUE

```
#define WDT_LOAD_VALUE 0x27FFFFFFE
```

Value to load in watchdog's counter (= 2sec | 0x13FFFFFF = 1sec)

4.20.3 Typedef Documentation

4.20.3.1 TmrCntrSetup

```
typedef struct TmrCntrSetup_st TmrCntrSetup
```

Structure containing all the settings to set up the Triple Timer Counter.

4.20.4 Function Documentation

4.20.4.1 assert_callback()

```
void assert_callback (
    const char8 * File,
    s32 Line )
```

Callback for assertion.

Parameters

<i>File</i>	filename where the assertion is called
<i>Line</i>	line number in the File

Returns

None

Note

When this callback is called, the filename and line number are stored in the log file, and then the programm stops

4.20.4.2 axidma_rx_callback()

```
void axidma_rx_callback (
    XAxiDma * AxiDmaInst )
```

Callback when the dma finish a transfer.

Parameters

<i>AxiDmaInst</i>	pointer on the axidma's instance
-------------------	----------------------------------

Returns

None

Note

-

4.20.4.3 cleanup_interrupts()

```
void cleanup_interrupts ( )
```

Disable all the interrupts and stop the timers.

Parameters

<i>None</i>	
-------------	--

Returns

None

Note

-

4.20.4.4 devices_initialization()

```
int devices_initialization ( )
```

Initiate and setup all the interrupts.

Parameters

None	
------	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.20.4.5 enable_interrupts()

```
void enable_interrupts ( )
```

Enable the interrupts and start the timers.

Parameters

None	
------	--

Returns

None

Note

-

4.20.4.6 interrupts_initialization()

```
int interrupts_initialization (  
    void )
```

Attach all the interrupt to the system and set the priority.

Parameters

None	
------	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.20.4.7 setup_axidma_int()

```
int setup_axidma_int (  
    void )
```

Setup the axidma interrupt.

Parameters

None	
------	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.20.4.8 setup_scu_timer_int()

```
int setup_scu_timer_int (  
    void )
```

Setup the timer scu interrupt.

Parameters

<i>None</i>	
-------------	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.20.4.9 setup_ttcps_timer_int()

```
int setup_ttcps_timer_int (  
    void )
```

Setup the timer tcps interrupt.

Parameters

<i>None</i>	
-------------	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.20.4.10 timer_scu_callback()

```
void timer_scu_callback (  
    XScuTimer * TimerInstance )
```

Callback for the timer scu.

Parameters

<i>TimerInstance</i>	pointer on the timer's instance
----------------------	---------------------------------

Returns

None

Note

This callback is called every 250ms

4.20.4.11 timer_ttcps_callback()

```
void timer_ttcps_callback (
    XTtcPs * TimerInstance )
```

Callback for the timer tcps.

Parameters

<i>TimerInstance</i>	pointer on the timer's instance
----------------------	---------------------------------

Returns

None

Note

This callback is called every 1sec

4.20.4.12 wdt_scu_callback()

```
void wdt_scu_callback (
    XScuWdt * WdtInstance )
```

Callback for the timer wdt.

Parameters

<i>WdtInstance</i>	pointer on the timer's instance
--------------------	---------------------------------

Returns

None

Note

This callback is called only if the wdt is used has a timer (not the case here), used only the configure the wdt period of 1sec

4.21 main.c File Reference

```
#include <stdint.h>
#include "lwip/init.h"
#include "netif/xadapter.h"
#include "platform_config.h"
#include "xparameters.h"
#include "udp_peripheral.h"
#include "axis_peripheral.h"
#include "file_hm.h"
#include "global.h"
#include "iic_DAC_LTC2657.h"
#include "pedestal.h"
#include "xtime_l.h"
#include "xscuwdt.h"
#include "get_20_windows.h"
#include "get_transfer_fct.h"
#include "transfer_function.h"
```

Typedefs

- typedef enum [clean_state_enum](#) [clean_state_en](#)
This is the enumeration of the process to stop when exiting the program.
- typedef enum [dma_stm_enum](#) [dma_stm_en](#)
This is the enumeration of the state machine.

Enumerations

- enum [clean_state_enum](#) { [GLOBAL_VAR](#) =0x1, [INTERRUPT](#) =0x2, [UDP](#) =0x4 }
 - enum [dma_stm_enum](#) { [IDLE](#), [STREAM](#), [GET_TRANSFER_FCT](#), [GET_20_WINDOWS](#) }
- This is the enumeration of the state machine.*

Functions

- void [end_main](#) ([clean_state_en](#) state)
- int [main](#) ()

Variables

- struct netif * [echo_netif](#)
Pointer on the network interface.
- volatile bool [run_flag](#)
Flag reset when the user send the command "stop uC".
- volatile bool [stream_flag](#)
Flag raised when the user send the command "start streaming".
- volatile bool [flag_ttcs_timer](#)
Flag raised when the Triple Timer Counter overflows.
- volatile bool [flag_scu_timer](#)

- Flag raised when the SCU timer overflows.*

 - XScuWdt [WdtScuInstance](#)

Instance of the device watchdog.
 - volatile bool [flag_assertion](#)

Flag raised when an assertion has occurred.
 - volatile bool [flag_while_loop](#)

Flag raised when the program has entered the while loop.
 - int [flag_axidma_rx](#) [4]

Array of flag, one for each PMT.
 - int * [regptr](#)

Array containing registers of AXI-lite.
 - volatile bool [get_transfer_fct_flag](#)

Flag raised when the user send the command "get transfer function".
 - volatile bool [get_20_windows_flag](#)

Flag raised when the user send the command "get 20 windows".
 - volatile bool [empty_flag](#)

Flag true when the list is empty (first_element = last_element)
 - [data_list](#) * [first_element](#)

Pointer on the first element of the list used in trigger mode.

4.21.1 Detailed Description

Author

Anthony Schluchin

Date

16th November 2018

Version

0.0

4.21.2 Typedef Documentation

4.21.2.1 `clean_state_en`

```
typedef enum clean\_state\_enum clean\_state\_en
```

This is the enumeration of the process to stop when exiting the program.

4.21.2.2 dma_stm_en

```
typedef enum dma_stm_enum dma_stm_en
```

This is the enumeration of the state machine.

4.21.3 Enumeration Type Documentation

4.21.3.1 clean_state_enum

```
enum clean_state_enum
```

This is the enumeration of the process to stop when exiting the program.

Enumerator

GLOBAL_VAR	Free the global variable reserved in function init_global_var
INTERRUPT	Stop the interrupt
UDP	Close both of the UDP communications

4.21.3.2 dma_stm_enum

```
enum dma_stm_enum
```

This is the enumeration of the state machine.

Enumerator

IDLE	No data to send, waiting on a command
STREAM	System in mode streaming
GET_TRANSFER_FCT	System sending the data for the transfer function in response to the corresponding command
GET_20_WINDOWS	System sending the data 20 consecutive windows in response to the corresponding command

4.21.4 Function Documentation

4.21.4.1 end_main()

```
void end_main (  
    clean_state_en state )
```

4.21.4.2 main()

```
int main ( )
```

4.21.5 Variable Documentation

4.21.5.1 echo_netif

```
struct netif* echo_netif
```

Pointer on the network interface.

4.21.5.2 empty_flag

```
volatile bool empty_flag
```

Flag true when the list is empty (first_element = last_element)

4.21.5.3 first_element

```
data_list* first_element
```

Pointer on the first element of the list used in trigger mode.

4.21.5.4 flag_assertion

```
volatile bool flag_assertion
```

Flag raised when an assertion has occurred.

4.21.5.5 flag_axidma_rx

```
int flag_axidma_rx[4]
```

Array of flag, one for each PMT.

4.21.5.6 flag_scu_timer

```
volatile bool flag_scu_timer
```

Flag raised when the SCU timer overflows.

4.21.5.7 flag_ttcps_timer

```
volatile bool flag_ttcps_timer
```

Flag raised when the Triple Timer Counter overflows.

4.21.5.8 flag_while_loop

```
volatile bool flag_while_loop
```

Flag raised when the program has entered the while loop.

4.21.5.9 get_20_windows_flag

```
volatile bool get_20_windows_flag
```

Flag raised when the user send the command "get 20 windows".

4.21.5.10 get_transfer_fct_flag

```
volatile bool get_transfer_fct_flag
```

Flag raised when the user send the command "get transfer function".

4.21.5.11 regptr

```
int* regptr
```

Array containing registers of AXI-lite.

4.21.5.12 run_flag

```
volatile bool run_flag
```

Flag reset when the user send the command "stop uC".

4.21.5.13 stream_flag

```
volatile bool stream_flag
```

Flag raised when the user send the command "start streaming".

4.21.5.14 WdtScuInstance

```
XScuWdt WdtScuInstance
```

Instance of the device watchdog.

4.22 pedestal.c File Reference

```
#include "pedestal.h"
```

Functions

- int [init_pedestals](#) (void)

Calculate the pedestal value for every memory location in the TARGET C.

Variables

- `int * regptr`
Array containing registers of AXI-lite.
- `uint16_t pedestal [512][16][32]`
Array containing the pedestal correction for every sample.
- `volatile bool flag_axidma_error`
Flag raised when AXI-DMA has an error.
- `volatile bool flag_axidma_rx_done`
Flag raised when AXI-DMA has finished an transfer, in OnDemand mode.
- `volatile bool flag_ttcps_timer`
Flag raised when the Triple Timer Counter overflows.
- `volatile bool flag_scu_timer`
Flag raised when the SCU timer overflows.
- `XScuWdt WdtScuInstance`
Instance of the device watchdog.

4.22.1 Detailed Description

Author

Anthony Schluchin

Date

18th December 2018

Version

0.0

4.22.2 Function Documentation

4.22.2.1 `init_pedestals()`

```
int init_pedestals (
    void )
```

Calculate the pedestal value for every memory location in the TARGET C.

Parameters

-	
---	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.22.3 Variable Documentation**4.22.3.1 flag_axidma_error**

```
volatile bool flag_axidma_error
```

Flag raised when AXI-DMA has an error.

4.22.3.2 flag_axidma_rx_done

```
volatile bool flag_axidma_rx_done
```

Flag raised when AXI-DMA has finished an transfer, in OnDemand mode.

4.22.3.3 flag_scu_timer

```
volatile bool flag_scu_timer
```

Flag raised when the SCU timer overflows.

4.22.3.4 flag_ttcps_timer

```
volatile bool flag_ttcps_timer
```

Flag raised when the Triple Timer Counter overflows.

4.22.3.5 pedestal

```
uint16_t pedestal[512][16][32]
```

Array containing the pedestal correction for every sample.

4.22.3.6 regptr

```
int* regptr
```

Array containing registers of AXI-lite.

4.22.3.7 WdtScuInstance

```
XScuWdt WdtScuInstance
```

Instance of the device watchdog.

4.23 pedestal.h File Reference

```
#include "data_analysis.h"
#include "axis_peripheral.h"
#include "xil_types.h"
#include "xstatus.h"
#include "TARGETC_RegisterMap.h"
#include "file_hm.h"
#include "xscuwdt.h"
```

Functions

- int [init_pedestals](#) (void)
Calculate the pedestal value for every memory location in the TARGET C.

4.23.1 Detailed Description

Author

Anthony Schluchin

Date

18th December 2018

Version

0.0

4.23.2 Function Documentation

4.23.2.1 init_pedestals()

```
int init_pedestals (
    void )
```

Calculate the pedestal value for every memory location in the TARGET C.

Parameters

-	
---	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.24 platform_config.h File Reference

Macros

- `#define PLATFORM_EMAC_BASEADDR XPAR_XEMACPS_0_BASEADDR`
- `#define PLATFORM_ZYNQ`

4.24.1 Macro Definition Documentation

4.24.1.1 PLATFORM_EMAC_BASEADDR

```
#define PLATFORM_EMAC_BASEADDR XPAR_XEMACPS_0_BASEADDR
```

4.24.1.2 PLATFORM_ZYNQ

```
#define PLATFORM_ZYNQ
```

4.25 platform_mb.c File Reference

4.26 platform_ppc.c File Reference

4.27 sfp.c File Reference

```
#include "xparameters.h"
```

4.27.1 Detailed Description

This file programs sfp phy chip.

MODIFICATION HISTORY:

Ver	Who	Date	Changes
1.0	srt	10/19/13	Initial Version

4.28 si5324.c File Reference

```
#include "xparameters.h"
```

4.28.1 Detailed Description

This file programs si5324 chip which generates clock for the peripherals.

Please refer to Si5324 Datasheet for more information <http://www.silabs.com/Support%20Documents/TechnicalDocs/Si5324.pdf>

Tested on Zynq ZC706 platform

MODIFICATION HISTORY:

Ver	Who	Date	Changes
1.0	srt	10/19/13	Initial Version

4.29 TARGETC_RegisterMap.c File Reference

```
#include "TARGETC_RegisterMap.h"
```

Functions

- void [SetTargetCRegisters](#) (void)
Set the TargetC Registers to default value using AXI Lite control.
- void [GetTargetCStatus](#) ()
In VERBOSE mode, print the status bit.
- void [GetTargetCControl](#) ()
In VERBOSE mode, print the control bit.
- void [ControlRegisterWrite](#) (int mask, int actionID)
Change a bit in the control register.
- void [WriteRegister](#) (int regID, int regData)
Change the value of a TARGET register.
- void [WriteReadBackRegister](#) (int regID, int regData)
Change the value of a TARGET register and reads it back.

Variables

- int * [regptr](#)
Array containing registers of AXI-lite.

4.29.1 Detailed Description

Author

Jonathan Hendriks

Date

14th November 2018

Version

0.0

4.29.2 Function Documentation

4.29.2.1 ControlRegisterWrite()

```
void ControlRegisterWrite (  
    int mask,  
    int actionID )
```

Change a bit in the control register.

Parameters

<i>mask</i>	which to change
<i>actionID</i>	set with ENABLE, reset with DISABLE, initialized with INIT

Returns

-

Note

-

4.29.2.2 GetTargetCControl()

```
void GetTargetCControl ( )
```

In VERBOSE mode, print the control bit.

Parameters

-	
---	--

Returns

-

Note

-

4.29.2.3 GetTargetCStatus()

```
void GetTargetCStatus ( )
```

In VERBOSE mode, print the status bit.

Parameters

-	
---	--

Returns

-

Note

-

4.29.2.4 SetTargetCRegisters()

```
void SetTargetCRegisters (
    void )
```

Set the TargetC Registers to default value using AXI Lite control.

Parameters

-	
---	--

Returns

status

Note

-

4.29.2.5 WriteReadBackRegister()

```
void WriteReadBackRegister (
    int regID,
    int regData )
```

Change the value of a TARGET register and reads it back.

Parameters

<i>regID</i>	register ID
<i>regData</i>	new value to set

Returns

-

Note

Result is printed

4.29.2.6 WriteRegister()

```
void WriteRegister (
    int  regID,
    int  regData )
```

Change the value of a TARGET register.

Parameters

<i>regID</i>	register ID
<i>regData</i>	new value to set

Returns

-

Note

-

4.29.3 Variable Documentation**4.29.3.1 regptr**

```
int* regptr
```

Array containing registers of AXI-lite.

4.30 TARGETC_RegisterMap.h File Reference

```
#include <stdio.h>
#include "xil_printf.h"
#include "global.h"
#include "utility.h"
```

Macros

- #define [TARGETC_REGISTERMAP_H](#) /* by using protection macros */
- #define [TC_VDLYTUNE_REG](#) 1
DAC Fine tune for delay cells 1 to 62, TC_VDLYTUNE_REG is the base address.
- #define [TC_SSTOUTFB_REG](#) 65
TARGETC Timing Generator parameter for SSTOUT Feedback.
- #define [TC_SSPIN_LE_REG](#) 66
TARGETC Timing Generator parameter for SSPIN Leading Edge (LE)
- #define [TC_SSPIN_TE_REG](#) 67
TARGETC Timing Generator parameter for SSPIN Trailing Edge (TE)
- #define [TC_WR_STRB2_LE_REG](#) 68
TARGETC Timing Generator parameter for Write Strobe 2 Leading Edge (LE)
- #define [TC_WR_STRB2_TE_REG](#) 69
TARGETC Timing Generator parameter for Write Strobe 2 Trailing Edge (TE)
- #define [TC_WR2_ADDR_LE_REG](#) 70
TARGETC Timing Generator parameter for Write 2 Address Leading Edge (LE)
- #define [TC_WR2_ADDR_TE_REG](#) 71
TARGETC Timing Generator parameter for Write 2 Address Trailing Edge (TE)
- #define [TC_WR_STRB1_LE_REG](#) 72
TARGETC Timing Generator parameter for Write Strobe 1 Leading Edge (LE)
- #define [TC_WR_STRB1_TE_REG](#) 73
TARGETC Timing Generator parameter for Write Strobe 1 Trailing Edge (TE)
- #define [TC_WR1_ADDR_LE_REG](#) 74
TARGETC Timing Generator parameter for Write 2 Address Leading Edge (LE)
- #define [TC_WR1_ADDR_TE_REG](#) 75
TARGETC Timing Generator parameter for Write 2 Address Trailing Edge (TE)
- #define [TC_MONTIMING_REG](#) 76
Monitor Timing Register.
- #define [TC_MT_PASS_MASK](#) 0x00000004
Monitor Timing Output Pin selection mask : PASS.
- #define [TC_MT_SSPOUT_MASK](#) 0x00000000
Monitor Timing Output Pin selection mask : SSPOUT.
- #define [TC_MT_SSTOUT_MASK](#) 0x00000010
Monitor Timing Output Pin selection mask : SSTOUT.
- #define [TC_MT_SSTOUTFB_MASK](#) 0x00000020
Monitor Timing Output Pin selection mask : SSTOUTFB.
- #define [TC_MT_SSPIN_MASK](#) 0x00000030
Monitor Timing Output Pin selection mask : SSPIN.
- #define [TC_MT_WR_STRB1_MASK](#) 0x00000040
Monitor Timing Output Pin selection mask : WR_STRB1.
- #define [TC_MT_WR1_ADDR_SYNC_MASK](#) 0x00000050
Monitor Timing Output Pin selection mask : WR1_ADDR.
- #define [TC_MT_WR_STRB2_MASK](#) 0x00000060
Monitor Timing Output Pin selection mask : WR_STRB2.
- #define [TC_MT_WR2_ADDR_SYNC_MASK](#) 0x00000070
Monitor Timing Output Pin selection mask : WR2_ADDR.
- #define [TC_MT_VDD_MASK](#) 0x00000080
Monitor Timing Output Pin selection mask : VDD (set High)
- #define [TC_VQBUFF_REG](#) 77
DAC Voltage Bias for QBIAS, VTRIMT and VBIAS.

- #define [TC_QBIAS_REG](#) 78
DAC Voltage Bias for the Delay Lock Loop (DLL)
- #define [TC_VTRIMT_REG](#) 79
DAC Voltage for fine tune of SSTOUIFB signal.
- #define [TC_VBIAS_REG](#) 80
DAC Voltage for global fine tune of the Vdly1 to Vdly64, with base address TC_VDLYTUNE_REG.
- #define [TC_VAPBUFF_REG](#) 81
DAC Voltage bias for VADJP, 0 = disable.
- #define [TC_VADJP_REG](#) 82
DAC Voltage for the DLL.
- #define [TC_VANBUFF_REG](#) 83
DAC Voltage bias for VADJN, 0 = disable.
- #define [TC_VADJN_REG](#) 84
DAC Voltage for the DLL.
- #define [TC_SBBIAS_REG](#) 85
DAC Voltage bias for Super Buffer and Registers.
- #define [TC_VDISCH_REG](#) 86
DAC Voltage bias for the discharge of the Wilkinson compator capacitor.
- #define [TC_ISEL_REG](#) 87
DAC Voltage bias for increasing/decreasing the slope charge of the Wilkinson capacitor.
- #define [TC_DBBIAS_REG](#) 88
DAC Voltage Bias for SSBIAS, VDISCH and ISEL.
- #define [TC_CMPBIAS2_REG](#) 89
DAC Voltage Bias for 2nd MosFET stage.
- #define [TC_PUBIAS_REG](#) 90
DAC Voltage Bias for the Pull-Up MosFET.
- #define [TC_CMPBIASIN_REG](#) 91
DAC Voltage Bias for the current source of the Wilkinson Comparator.
- #define [TC_MISCDIG_REG](#) 92
Miscellaneous register.
- #define [TC_TPG_REG](#) 128
Test pattern generator Register.
- #define [TC_CONTROL_REG](#) 129
Programmable Logic (PL) control register.
- #define [WRITE_MASK](#) 0x00000001
PL Control Mask : Write to TARGETC register.
- #define [REGCLR_MASK](#) 0x00000020
PL Control Mask : control over TARGETC Register Clear input (RegCLR)
- #define [SS_TPG_MASK](#) 0x00000080
PL Control Mask : control over TARGETC Sample Any Select input (Sampl_Any), 0 = TPG or 1 = Sample.
- #define [WINDOW_MASK](#) 0x00000400
PL Control Mask : Sample and Readout of windows depending on the arguments in TC_FSTWINDOW_REG and TC_NBRWINDOW_REG.
- #define [SWRESET_MASK](#) 0x00001000
PL Control Mask : Software reset for PL side, 0=enable, 1 disable.
- #define [SMODE_MASK](#) 0x00002000
PL Control Mask : Development bit for selection between interrupt mode, 0= each sample redout or 1=AXI-DMA.
- #define [TESTSTREAM_MASK](#) 0x00004000
PL Control Mask : Test the stream by sending dummy data from AXI-Stream component.
- #define [TESTFIFO_MASK](#) 0x00008000
PL Control Mask : Test the FIFO manager by filling it with dummy data.

- #define [PSBUSY_MASK](#) 0x00010000
PL Control Mask : processing system busy mask, to diable new transfer before it is ready.
- #define [CPUMODE_MASK](#) 0x00020000
PL Control Mask : PL Running mode, 0=User Mode or 1 = Trigger mode.
- #define [TC_STATUS_REG](#) 130
Programmable Logic (PL) status register.
- #define [BUSY_MASK](#) 0x00000001
PL Status Mask : Write Register action status.
- #define [LOCKED_MASK](#) 0x00000002
PL Status Mask : Clock management system MMCM lock feedback.
- #define [STORAGE_MASK](#) 0x00000004
PL Status Mask : Feedback of the sample and readout of Windows action (WINDOW_MASK in control register)
- #define [SSVALID_MASK](#) 0x00000008
PL Status Mask : Sample Select valid signal, monitors new sample availability.
- #define [WINDOWBUSY_MASK](#) 0x00000010
PL Status Mask : Monitor busy bit for the readout, digitization and sample readout processes.
- #define [TC_ADDR_REG](#) 131
Address of register to be update in the TARGETC for a write register operation.
- #define [TC_DATA_OUT_REG](#) 132
read back register for write register operation
- #define [TC_eDO_CH0_REG](#) 133
Sample readout for Channel 0 using SMODE=0.
- #define [TC_eDO_CH1_REG](#) 134
Sample readout for Channel 1 using SMODE=0.
- #define [TC_eDO_CH2_REG](#) 135
Sample readout for Channel 2 using SMODE=0.
- #define [TC_eDO_CH3_REG](#) 136
Sample readout for Channel 3 using SMODE=0.
- #define [TC_eDO_CH4_REG](#) 137
Sample readout for Channel 4 using SMODE=0.
- #define [TC_eDO_CH5_REG](#) 138
Sample readout for Channel 5 using SMODE=0.
- #define [TC_eDO_CH6_REG](#) 139
Sample readout for Channel 6 using SMODE=0.
- #define [TC_eDO_CH7_REG](#) 140
Sample readout for Channel 7 using SMODE=0.
- #define [TC_eDO_CH8_REG](#) 141
Sample readout for Channel 8 using SMODE=0.
- #define [TC_eDO_CH9_REG](#) 142
Sample readout for Channel 9 using SMODE=0.
- #define [TC_eDO_CH10_REG](#) 143
Sample readout for Channel 10 using SMODE=0.
- #define [TC_eDO_CH11_REG](#) 144
Sample readout for Channel 11 using SMODE=0.
- #define [TC_eDO_CH12_REG](#) 145
Sample readout for Channel 12 using SMODE=0.
- #define [TC_eDO_CH13_REG](#) 146
Sample readout for Channel 13 using SMODE=0.
- #define [TC_eDO_CH14_REG](#) 147
Sample readout for Channel 14 using SMODE=0.
- #define [TC_eDO_CH15_REG](#) 148

- Sample readout for Channel 15 using SMODE=0.*
- #define [TC_FSTWINDOW_REG](#) 151
For window operation in User Mode (CPUMODE=0), this register specifies the first window from 0 to 511.
- #define [TC_NBRWINDOW_REG](#) 152
For window operation in User Mode (CPUMODE=0), this register specifies the number of window to be readout consecutively, MAXIMUM is 15 windows.
- #define [LAST_REGISTER_ADDR](#) 153
Last register.
- #define [ENABLE](#) 1
PL Control Register Mask enable action.
- #define [DISABLE](#) 0
PL Control Register Mask disable action.
- #define [INIT](#) 2
PL Control Register Mask INIT action.

Functions

- void [SetTargetCRegisters](#) (void)
Set the TargetC Registers to default value using AXI Lite control.
- void [GetTargetCStatus](#) ()
In VERBOSE mode, print the status bit.
- void [GetTargetCControl](#) ()
In VERBOSE mode, print the control bit.
- void [ControlRegisterWrite](#) (int mask, int actionID)
Change a bit in the control register.
- void [WriteRegister](#) (int regID, int regData)
Change the value of a TARGET register.
- void [WriteReadBackRegister](#) (int regID, int regData)
Change the value of a TARGET register and reads it back.

4.30.1 Detailed Description

Author

Jonathan Hendriks

Date

14th November 2018

Version

0.0

4.30.2 Macro Definition Documentation

4.30.2.1 BUSY_MASK

```
#define BUSY_MASK 0x00000001
```

PL Status Mask : Write Register action status.

4.30.2.2 CPUMODE_MASK

```
#define CPUMODE_MASK 0x00020000
```

PL Control Mask : PL Running mode, 0=User Mode or 1 = Trigger mode.

4.30.2.3 DISABLE

```
#define DISABLE 0
```

PL Control Register Mask disable action.

4.30.2.4 ENABLE

```
#define ENABLE 1
```

PL Control Register Mask enable action.

Utility define

4.30.2.5 INIT

```
#define INIT 2
```

PL Control Register Mask INIT action.

4.30.2.6 LAST_REGISTER_ADDR

```
#define LAST_REGISTER_ADDR 153
```

Last register.

4.30.2.7 LOCKED_MASK

```
#define LOCKED_MASK 0x00000002
```

PL Status Mask : Clock management system MMCM lock feedback.

4.30.2.8 PSBUSY_MASK

```
#define PSBUSY_MASK 0x00010000
```

PL Control Mask : processing system busy mask, to diable new transfer before it is ready.

4.30.2.9 REGCLR_MASK

```
#define REGCLR_MASK 0x00000020
```

PL Control Mask : control over TARGETC Register Clear input (RegCLR)

4.30.2.10 SMODE_MASK

```
#define SMODE_MASK 0x00002000
```

PL Control Mask : Development bit for selection between interrupt mode, 0= each sample redout or 1=AXI-DMA.

4.30.2.11 SS_TPG_MASK

```
#define SS_TPG_MASK 0x00000080
```

PL Control Mask : control over TARGETC Sample Any Select input (Sampl_Any), 0 = TPG or 1 = Sample.

4.30.2.12 SSVALID_MASK

```
#define SSVALID_MASK 0x00000008
```

PL Status Mask : Sample Select valid signal, monitors new sample availability.

4.30.2.13 STORAGE_MASK

```
#define STORAGE_MASK 0x00000004
```

PL Status Mask : Feedback of the sample and readout of Windows action (WINDOW_MASK in control register)

4.30.2.14 SWRESET_MASK

```
#define SWRESET_MASK 0x00001000
```

PL Control Mask : Software reset for PL side, 0=enable, 1 disable.

4.30.2.15 TARGETC_REGISTERMAP_H

```
#define TARGETC_REGISTERMAP_H /* by using protection macros */
```

4.30.2.16 TC_ADDR_REG

```
#define TC_ADDR_REG 131
```

Address of register to be update in the TARGETC for a write register operation.

4.30.2.17 TC_CMPBIAS2_REG

```
#define TC_CMPBIAS2_REG 89
```

DAC Voltage Bias for 2nd MosFET stage.

4.30.2.18 TC_CMPBIASIN_REG

```
#define TC_CMPBIASIN_REG 91
```

DAC Voltage Bias for the current source of the Wilkinson Comparator.

4.30.2.19 TC_CONTROL_REG

```
#define TC_CONTROL_REG 129
```

Programmable Logic (PL) control register.

PL-PS User registers

4.30.2.20 TC_DATA_OUT_REG

```
#define TC_DATA_OUT_REG 132
```

read back register for write register operation

4.30.2.21 TC_DBBIAS_REG

```
#define TC_DBBIAS_REG 88
```

DAC Voltage Bias for SSBIAS, VDISCH and ISEL.

4.30.2.22 TC_eDO_CH0_REG

```
#define TC_eDO_CH0_REG 133
```

Sample readout for Channel 0 using SMODE=0.

4.30.2.23 TC_eDO_CH10_REG

```
#define TC_eDO_CH10_REG 143
```

Sample readout for Channel 10 using SMODE=0.

4.30.2.24 TC_eDO_CH11_REG

```
#define TC_eDO_CH11_REG 144
```

Sample readout for Channel 11 using SMODE=0.

4.30.2.25 TC_eDO_CH12_REG

```
#define TC_eDO_CH12_REG 145
```

Sample readout for Channel 12 using SMODE=0.

4.30.2.26 TC_eDO_CH13_REG

```
#define TC_eDO_CH13_REG 146
```

Sample readout for Channel 13 using SMODE=0.

4.30.2.27 TC_eDO_CH14_REG

```
#define TC_eDO_CH14_REG 147
```

Sample readout for Channel 14 using SMODE=0.

4.30.2.28 TC_eDO_CH15_REG

```
#define TC_eDO_CH15_REG 148
```

Sample readout for Channel 15 using SMODE=0.

4.30.2.29 TC_eDO_CH1_REG

```
#define TC_eDO_CH1_REG 134
```

Sample readout for Channel 1 using SMODE=0.

4.30.2.30 TC_eDO_CH2_REG

```
#define TC_eDO_CH2_REG 135
```

Sample readout for Channel 2 using SMODE=0.

4.30.2.31 TC_eDO_CH3_REG

```
#define TC_eDO_CH3_REG 136
```

Sample readout for Channel 3 using SMODE=0.

4.30.2.32 TC_eDO_CH4_REG

```
#define TC_eDO_CH4_REG 137
```

Sample readout for Channel 4 using SMODE=0.

4.30.2.33 TC_eDO_CH5_REG

```
#define TC_eDO_CH5_REG 138
```

Sample readout for Channel 5 using SMODE=0.

4.30.2.34 TC_eDO_CH6_REG

```
#define TC_eDO_CH6_REG 139
```

Sample readout for Channel 6 using SMODE=0.

4.30.2.35 TC_eDO_CH7_REG

```
#define TC_eDO_CH7_REG 140
```

Sample readout for Channel 7 using SMODE=0.

4.30.2.36 TC_eDO_CH8_REG

```
#define TC_eDO_CH8_REG 141
```

Sample readout for Channel 8 using SMODE=0.

4.30.2.37 TC_eDO_CH9_REG

```
#define TC_eDO_CH9_REG 142
```

Sample readout for Channel 9 using SMODE=0.

4.30.2.38 TC_FSTWINDOW_REG

```
#define TC_FSTWINDOW_REG 151
```

For windowr operation in User Mode (CPUMODE=0), this register specifies the first window from 0 to 511.

4.30.2.39 TC_ISEL_REG

```
#define TC_ISEL_REG 87
```

DAC Voltage bias for increasing/decreasing the slope charge of the Wilkinson capacitor.

4.30.2.40 TC_MISCDIG_REG

```
#define TC_MISCDIG_REG 92
```

Miscellaneous register.

4.30.2.41 TC_MONTIMING_REG

```
#define TC_MONTIMING_REG 76
```

Monitor Timing Register.

4.30.2.42 TC_MT_PASS_MASK

```
#define TC_MT_PASS_MASK 0x00000004
```

Monitor Timing Output Pin selection mask : PASS.

4.30.2.43 TC_MT_SSPIN_MASK

```
#define TC_MT_SSPIN_MASK 0x00000030
```

Monitor Timing Output Pin selection mask : SSPIN.

4.30.2.44 TC_MT_SSPOUT_MASK

```
#define TC_MT_SSPOUT_MASK 0x00000000
```

Monitor Timing Output Pin selection mask : SSPOUT.

4.30.2.45 TC_MT_SSTOUT_MASK

```
#define TC_MT_SSTOUT_MASK 0x00000010
```

Monitor Timing Output Pin selection mask : SSTOUT.

4.30.2.46 TC_MT_SSTOUTFB_MASK

```
#define TC_MT_SSTOUTFB_MASK 0x00000020
```

Monitor Timing Output Pin selection mask : SSTOUTFB.

4.30.2.47 TC_MT_VDD_MASK

```
#define TC_MT_VDD_MASK 0x00000080
```

Monitor Timing Output Pin selection mask : VDD (set High)

4.30.2.48 TC_MT_WR1_ADDR_SYNC_MASK

```
#define TC_MT_WR1_ADDR_SYNC_MASK 0x00000050
```

Monitor Timing Output Pin selection mask : WR1_ADDR.

4.30.2.49 TC_MT_WR2_ADDR_SYNC_MASK

```
#define TC_MT_WR2_ADDR_SYNC_MASK 0x00000070
```

Monitor Timing Output Pin selection mask : WR2_ADDR.

4.30.2.50 TC_MT_WR_STRB1_MASK

```
#define TC_MT_WR_STRB1_MASK 0x00000040
```

Monitor Timing Output Pin selection mask : WR_STRB1.

4.30.2.51 TC_MT_WR_STRB2_MASK

```
#define TC_MT_WR_STRB2_MASK 0x00000060
```

Monitor Timing Output Pin selection mask : WR_STRB2.

4.30.2.52 TC_NBRWINDOW_REG

```
#define TC_NBRWINDOW_REG 152
```

For window operation in User Mode (CPUMODE=0), this register specifies the number of window to be readout consecutively, MAXIMUM is 15 windows.

4.30.2.53 TC_PUBIAS_REG

```
#define TC_PUBIAS_REG 90
```

DAC Voltage Bias for the Pull-Up MosFET.

4.30.2.54 TC_QBIAS_REG

```
#define TC_QBIAS_REG 78
```

DAC Voltage Bias for the Delay Lock Loop (DLL)

4.30.2.55 TC_SBBIAS_REG

```
#define TC_SBBIAS_REG 85
```

DAC Voltage bias for Super Buffer and Registers.

4.30.2.56 TC_SSPIN_LE_REG

```
#define TC_SSPIN_LE_REG 66
```

TARGETC Timing Generator parameter for SSPIN Leading Edge (LE)

4.30.2.57 TC_SSPIN_TE_REG

```
#define TC_SSPIN_TE_REG 67
```

TARGETC Timing Generator parameter for SSPIN Trailing Edge (TE)

4.30.2.58 TC_SSTOUTFB_REG

```
#define TC_SSTOUTFB_REG 65
```

TARGETC Timing Generator parameter for SSTOUT Feedback.

4.30.2.59 TC_STATUS_REG

```
#define TC_STATUS_REG 130
```

Programmable Logic (PL) status register.

4.30.2.60 TC_TPG_REG

```
#define TC_TPG_REG 128
```

Test pattern generator Register.

4.30.2.61 TC_VADJN_REG

```
#define TC_VADJN_REG 84
```

DAC Voltage for the DLL.

4.30.2.62 TC_VADJP_REG

```
#define TC_VADJP_REG 82
```

DAC Voltage for the DLL.

4.30.2.63 TC_VANBUFF_REG

```
#define TC_VANBUFF_REG 83
```

DAC Voltage bias for VADJN, 0 = disable.

4.30.2.64 TC_VAPBUFF_REG

```
#define TC_VAPBUFF_REG 81
```

DAC Voltage bias for VADJP, 0 = disable.

4.30.2.65 TC_VBIAS_REG

```
#define TC_VBIAS_REG 80
```

DAC Voltage for global fine tune of the Vdly1 to Vdly64, with base address TC_VDLYTUNE_REG.

4.30.2.66 TC_VDISCH_REG

```
#define TC_VDISCH_REG 86
```

DAC Voltage bias for the discharge of the Wilkinson compator capacitor.

4.30.2.67 TC_VDLYTUNE_REG

```
#define TC_VDLYTUNE_REG 1
```

DAC Fine tune for delay cells 1 to 62, TC_VDLYTUNE_REG is the base address.

Definition of Registers

4.30.2.68 TC_VQBUFF_REG

```
#define TC_VQBUFF_REG 77
```

DAC Voltage Bias for QBIAS, VTRIMT and VBIAS.

4.30.2.69 TC_VTRIMT_REG

```
#define TC_VTRIMT_REG 79
```

DAC Voltage for fine tune of SSTOUBF signal.

4.30.2.70 TC_WR1_ADDR_LE_REG

```
#define TC_WR1_ADDR_LE_REG 74
```

TARGETC Timing Generator parameter for Write 2 Address Leading Edge (LE)

4.30.2.71 TC_WR1_ADDR_TE_REG

```
#define TC_WR1_ADDR_TE_REG 75
```

TARGETC Timing Generator parameter for Write 2 Address Trailing Edge (TE)

4.30.2.72 TC_WR2_ADDR_LE_REG

```
#define TC_WR2_ADDR_LE_REG 70
```

TARGETC Timing Generator parameter for Write 2 Address Leading Edge (LE)

4.30.2.73 TC_WR2_ADDR_TE_REG

```
#define TC_WR2_ADDR_TE_REG 71
```

TARGETC Timing Generator parameter for Write 2 Address Trailing Edge (TE)

4.30.2.74 TC_WR_STRB1_LE_REG

```
#define TC_WR_STRB1_LE_REG 72
```

TARGETC Timing Generator parameter for Write Strobe 1 Leading Edge (LE)

4.30.2.75 TC_WR_STRB1_TE_REG

```
#define TC_WR_STRB1_TE_REG 73
```

TARGETC Timing Generator parameter for Write Strobe 1 Trailing Edge (TE)

4.30.2.76 TC_WR_STRB2_LE_REG

```
#define TC_WR_STRB2_LE_REG 68
```

TARGETC Timing Generator parameter for Write Strobe 2 Leading Edge (LE)

4.30.2.77 TC_WR_STRB2_TE_REG

```
#define TC_WR_STRB2_TE_REG 69
```

TARGETC Timing Generator parameter for Write Strobe 2 Trailing Edge (TE)

4.30.2.78 TESTFIFO_MASK

```
#define TESTFIFO_MASK 0x00008000
```

PL Control Mask : Test the FIFO manager by filling it with dummy data.

4.30.2.79 TESTSTREAM_MASK

```
#define TESTSTREAM_MASK 0x00004000
```

PL Control Mask : Test the stream by sending dummy data from AXI-Stream component.

4.30.2.80 WINDOW_MASK

```
#define WINDOW_MASK 0x00000400
```

PL Control Mask : Sample and Readout of windows depending on the arguments in TC_FSTWINDOW_REG and TC_NBRWINDOW_REG.

4.30.2.81 WINDOWBUSY_MASK

```
#define WINDOWBUSY_MASK 0x00000010
```

PL Status Mask : Monitor busy bit for the readout, digitization and sample readout processes.

4.30.2.82 WRITE_MASK

```
#define WRITE_MASK 0x00000001
```

PL Control Mask : Write to TARGETC register.

4.30.3 Function Documentation

4.30.3.1 ControlRegisterWrite()

```
void ControlRegisterWrite (  
    int mask,  
    int actionID )
```

Change a bit in the control register.

Parameters

<i>mask</i>	which to change
<i>actionID</i>	set with ENABLE, reset with DISABLE, initialized with INIT

Returns

-

Note

-

4.30.3.2 GetTargetCControl()

```
void GetTargetCControl ( )
```

In VERBOSE mode, print the control bit.

Parameters

-	
---	--

Returns

-

Note

-

4.30.3.3 GetTargetCStatus()

```
void GetTargetCStatus ( )
```

In VERBOSE mode, print the status bit.

Parameters

-	
---	--

Returns

-

Note

-

4.30.3.4 SetTargetCRegisters()

```
void SetTargetCRegisters (
    void )
```

Set the TargetC Registers to default value using AXI Lite control.

Parameters

-	
---	--

Returns

status

Note

-

4.30.3.5 WriteReadBackRegister()

```
void WriteReadBackRegister (
    int regID,
    int regData )
```

Change the value of a TARGET register and reads it back.

Parameters

<i>regID</i>	register ID
<i>regData</i>	new value to set

Returns

-

Note

Result is printed

4.30.3.6 WriteRegister()

```
void WriteRegister (
    int regID,
    int regData )
```

Change the value of a TARGET register.

Parameters

<i>regID</i>	register ID
<i>regData</i>	new value to set

Returns

-

Note

-

4.31 time_hm.c File Reference

```
#include "time_hm.h"
```

Functions

- void [gettime_hm](#) ([time_cplt](#) *t)
Get the time (year, month, day, hour,...) depending on the offset.
- void [settime_hm](#) ([time_cplt](#) *t)
Set the time (year, month, day, hour,...) meaning set the offset.
- bool [isALeapYear](#) (int year)
Return if the year is leap or not.
- void [addtime](#) (void)
Add 2.5sec to the clock time.
- void [stringtotime](#) (char *time_str, [time_cplt](#) *time)
Convert string to time_cplt.

Variables

- [time_cplt offset_time](#)
Variable which contain the offset time. It can be changed with [settime_hmr](#).
- [uint64_t offset_counter](#) = 0
Variable which contain the offset of the General Timer when the time was set.
- int [day_per_month](#) [13] = {(int)NULL, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}

4.31.1 Detailed Description

Author

Anthony Schluchin

Date

9th November 2018

Version

0.0

4.31.2 Function Documentation

4.31.2.1 addtime()

```
void addtime (
    void )
```

Add 2.5sec to the clock time.

Parameters

<i>None</i>	
-------------	--

Returns

None

Note

Specially used to compensate the delay due to a reboot with the wdt

4.31.2.2 gettime_hm()

```
void gettime_hm (
    time_cplt * t )
```

Get the time (year, month, day, hour,...) depending on the offset.

Parameters

<i>t</i>	Pointer to the time_cplt structure wich will return the current time.
----------	-----------------------------------------------------------------------

Returns

None.

Note

None.

4.31.2.3 isALeapYear()

```
bool isALeapYear (  
    int year )
```

Return if the year is leap or not.

Parameters

<i>year</i>	the year to be tested
-------------	-----------------------

Returns

- True: if the year is leap
 - False: if the year is not leap

Note

None.

4.31.2.4 settime_hm()

```
void settime_hm (  
    time_cplt * t )
```

Set the time (year, month, day, hour,...) meaning set the offset.

Parameters

<i>t</i>	Pointer to the time_cplt structure wich will be written in the global offset value
----------	------------------------------------------------------------------------------------

Returns

None.

Note

When this function, we need to save the state of the Global Timer Counter Register (XTime_GetTime)

4.31.2.5 stringtotime()

```
void stringtotime (  
    char * time_str,  
    time_cplt * time )
```

Convert string to time_cplt.

Parameters

<i>time_str</i>	the time in string to convert
<i>time</i>	pointer to the time converted

Returns

None

Note

The string to convert should be like this: "dd.mm.yyyy @ hh:mm:ss"

4.31.3 Variable Documentation

4.31.3.1 day_per_month

```
int day_per_month[13] = {(int)NULL, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}
```

@bried "Constant" wich contain the number of day for every month (ex: january = day_per_mont[1])

4.31.3.2 offset_counter

```
uint64_t offset_counter = 0
```

Variable which contain the offset of the General Timer when the time was set.

4.31.3.3 offset_time

```
time_cplt offset_time
```

Initial value:

```
= {  
    .year = 2000,  
    .month = 1,  
    .day = 1,  
    .hour = 0,  
    .minute = 0,  
    .second = 0,  
    .milisecond = 0  
}
```

Variable which contain the offset time. It can be changed with settime_hmr.

4.32 time_hm.h File Reference

```
#include <stdbool.h>  
#include "xtime_l.h"
```

Data Structures

- struct `time_cplt_st`
Structure of the timestamp.

Typedefs

- typedef struct `time_cplt_st` `time_cplt`
Structure of the timestamp.

Functions

- void `gettime_hm` (`time_cplt` *t)
Get the time (year, month, day, hour,...) depending on the offset.
- void `settime_hm` (`time_cplt` *t)
Set the time (year, month, day, hour,...) meaning set the offset.
- bool `isALeapYear` (int year)
Return if the year is leap or not.
- void `addtime` (void)
Add 2.5sec to the clock time.
- void `stringtotime` (char *time_str, `time_cplt` *time)
Convert string to time_cplt.

4.32.1 Detailed Description

Author

Anthony Schluchin

Date

9th November 2018

Version

0.0

4.32.2 Typedef Documentation

4.32.2.1 `time_cplt`

```
typedef struct time_cplt_st time_cplt
```

Structure of the timestamp.

4.32.3 Function Documentation

4.32.3.1 `addtime()`

```
void addtime (  
    void )
```

Add 2.5sec to the clock time.

Parameters

<i>None</i>	
-------------	--

Returns

None

Note

Specially used to compensate the delay due to a reboot with the wdt

4.32.3.2 `gettime_hm()`

```
void gettime_hm (
    time_cplt * t )
```

Get the time (year, month, day, hour,...) depending on the offset.

Parameters

<i>t</i>	Pointer to the time_cplt structure wich will return the current time.
----------	-----------------------------------------------------------------------

Returns

None.

Note

None.

4.32.3.3 `isALeapYear()`

```
bool isALeapYear (
    int year )
```

Return if the year is leap or not.

Parameters

<i>year</i>	the year to be tested
-------------	-----------------------

Returns

- True: if the year is leap
 - False: if the year is not leap

Note

None.

4.32.3.4 settime_hm()

```
void settime_hm (  
    time_cplt * t )
```

Set the time (year, month, day, hour,...) meaning set the offset.

Parameters

<i>t</i>	Pointer to the time_cplt structure wich will be written in the global offset value
----------	------------------------------------------------------------------------------------

Returns

None.

Note

When this function, we need to save the state of the Global Timer Counter Register (XTime_GetTime)

4.32.3.5 stringtotime()

```
void stringtotime (  
    char * time_str,  
    time_cplt * time )
```

Convert string to time_cplt.

Parameters

<i>time_str</i>	the time in string to convert
<i>time</i>	pointer to the time converted

Returns

None

Note

The string to convert should be like this: "dd.mm.yyyy @ hh:mm:ss"

4.33 transfer_function.c File Reference

```
#include "transfer_function.h"
```

Functions

- int [init_transfer_function](#) (void)
Calculate the transfer function lookup table.

Variables

- int * [regptr](#)
Array containing registers of AXI-lite.
- volatile bool [flag_axidma_error](#)
Flag raised when AXI-DMA has an error.
- volatile bool [flag_axidma_rx_done](#)
Flag raised when AXI-DMA has finished an transfer, in OnDemand mode.
- uint16_t [pedestal](#) [512][16][32]
Array containing the pedestal correction for every sample.
- uint16_t [lookup_table](#) [2048]
Lookup table to correct the transfer function.
- volatile bool [flag_ttcps_timer](#)
Flag raised when the Triple Timer Counter overflows.
- volatile bool [flag_scu_timer](#)
Flag raised when the SCU timer overflows.
- XScuWdt [WdtScuInstance](#)
Instance of the device watchdog.

4.33.1 Detailed Description

Author

Anthony Schluchin

Date

7th January 2019

Version

0.0

Author

Anthony Schluchin

Date

16th January 2019

Version

0.0

4.33.2 Function Documentation

4.33.2.1 init_transfer_function()

```
int init_transfer_function (
    void )
```

Calculate the transfer function lookup table.

Parameters

-	
---	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.33.3 Variable Documentation

4.33.3.1 flag_axidma_error

```
volatile bool flag_axidma_error
```

Flag raised when AXI-DMA has an error.

4.33.3.2 flag_axidma_rx_done

```
volatile bool flag_axidma_rx_done
```

Flag raised when AXI-DMA has finished an transfer, in OnDemand mode.

4.33.3.3 flag_scu_timer

```
volatile bool flag_scu_timer
```

Flag raised when the SCU timer overflows.

4.33.3.4 flag_ttcps_timer

```
volatile bool flag_ttcps_timer
```

Flag raised when the Triple Timer Counter overflows.

4.33.3.5 lookup_table

```
uint16_t lookup_table[2048]
```

Lookup table to correct the transfer function.

4.33.3.6 pedestal

```
uint16_t pedestal[512][16][32]
```

Array containing the pedestal correction for every sample.

4.33.3.7 regptr

```
int* regptr
```

Array containing registers of AXI-lite.

4.33.3.8 WdtScuInstance

```
XScuWdt WdtScuInstance
```

Instance of the device watchdog.

4.34 transfer_function.h File Reference

```
#include "xstatus.h"
#include "data_analysis.h"
#include "xil_types.h"
#include "axis_peripheral.h"
#include "TARGETC_RegisterMap.h"
#include "iic_DAC_LTC2657.h"
#include "GMHeader.h"
#include "file_hm.h"
#include "xscuwdt.h"
```


Functions

- int [init_transfer_function](#) (void)
Calculate the transfer fuction lookup table.

4.34.1 Detailed Description

Author

Anthony Schluchin

Date

7th January 2019

Version

0.0

Author

Anthony Schluchin

Date

16th January 2019

Version

0.0

4.34.2 Function Documentation

4.34.2.1 init_transfer_function()

```
int init_transfer_function (  
    void )
```

Calculate the transfer fuction lookup table.

Parameters

-	
---	--

Returns

XST_SUCCESS or XST_FAILURE (defined in xstatus.h)

Note

-

4.35 udp_peripheral.c File Reference

```
#include "udp_peripheral.h"
```

Functions

- `err_t transfer_data` (char *frame, uint16_t length)
Send a frame through UDP.
- `err_t transfer_cmd` (char *frame, uint16_t length)
Send a frame through UDP.
- `void udp_cmd_rcv` (void *arg, struct udp_pcb *pcb, struct pbuf *p, const ip_addr_t *addr, u16_t port)
callback when a command is received
- `int command_parser` (struct pbuf *p, char *return_buf)
Process the data received and report a command.
- `int setup_udp_settings` (ip_addr_t pc_ipaddr)
Setup all the settings for the UDP communication (data and command)
- `int setup_pcb_data` (ip_addr_t pc_ipaddr, uint16_t port)
Setup the UDP Protocol Control Block for the data frame.
- `int setup_pcb_cmd` (ip_addr_t pc_ipaddr, uint16_t port)
Setup the UDP Protocol Control Block for the command frame.
- `void cleanup_udp` (void)
Disable and clean the UDP PCB for the data and the command.
- `void print_ip` (char *msg, ip_addr_t *ip)
Function to print an IP address with a message (used in debug mostly)
- `void print_ip_settings` (ip_addr_t *ip, ip_addr_t *mask, ip_addr_t *gw)
Function to print all the settings of an Internet connection (used in debug mostly)

Variables

- `struct udp_pcb * pcb_data`
UDP Protocol Control Block for data communication.
- `struct udp_pcb * pcb_cmd`
UDP Protocol Control Block for command communication.
- `struct pbuf * buf_data`
Buffer structure used to send data packet.
- `struct pbuf * buf_cmd`
Buffer structure used to send command packet.
- `volatile int count_ttcps_timer`
Counter of the TTC.

- volatile int [count_scu_timer](#)
Instance of AXI-DMA.
- int [nbre_of_bytes](#)
Number of bytes sent during streaming (trigger mode)
- volatile bool [run_flag](#)
Flag reset when the user send the command "stop uC".
- volatile bool [stream_flag](#)
Flag raised when the user send the command "start streaming".
- volatile bool [get_transfer_fct_flag](#)
Flag raised when the user send the command "get transfer function".
- volatile bool [get_20_windows_flag](#)
Flag raised when the user send the command "get 20 windows".
- int * [regptr](#)
Array containing registers of AXI-lite.
- char * [frame_buf_cmd](#)
Buffer used to send the command (50 bytes above it reserved for protocol header)

4.35.1 Detailed Description

Author

Anthony Schluchin

Date

24th October 2018

Version

0.0

4.35.2 Function Documentation

4.35.2.1 cleanup_udp()

```
void cleanup_udp (  
    void )
```

Disable and clean the UDP PCB for the data and the command.

Parameters

None	
------	--

Returns

None

Note

-

4.35.2.2 command_parser()

```
int command_parser (
    struct pbuf * p,
    char * return_buf )
```

Process the data received and report a command.

Parameters

<i>p</i>	pointer to the buffer containing the data received
<i>return_buf</i>	pointer to the data to be echoed back

Returns

-1 if there is a problem or the number of byte in return_bf

Note

-

4.35.2.3 print_ip()

```
void print_ip (
    char * msg,
    ip_addr_t * ip )
```

Function to print an IP address with a message (used in debug mostly)

Parameters

<i>msg</i>	pointer to the message to print before the IP address
<i>ip</i>	pointer to the IP address to print

Returns

None

Note

-

4.35.2.4 print_ip_settings()

```
void print_ip_settings (
    ip_addr_t * ip,
    ip_addr_t * mask,
    ip_addr_t * gw )
```

Function to print all the settings of an Internet connection (used in debug mostly)

Parameters

<i>ip</i>	pointer to the IP address to print
<i>mask</i>	pointer to the mask to print
<i>gw</i>	pointer to the gateway to print

Returns

None

Note

-

4.35.2.5 setup_pcb_cmd()

```
int setup_pcb_cmd (
    ip_addr_t pc_ipaddr,
    uint16_t port )
```

Setup the UDP Protocol Control Block for the command frame.

Parameters

<i>pc_ipaddr</i>	IP address of the computer
<i>port</i>	UDP port of the computer used for the command

Returns

0 if ok, negative values if there is a problem

Note

-

4.35.2.6 setup_pcb_data()

```
int setup_pcb_data (
    ip_addr_t pc_ipaddr,
    uint16_t port )
```

Setup the UDP Protocol Control Block for the data frame.

Parameters

<i>pc_ipaddr</i>	IP address of the computer
<i>port</i>	UDP port of the computer used for the data

Returns

0 if ok, negative values if there is a problem

Note

-

4.35.2.7 setup_udp_settings()

```
int setup_udp_settings (
    ip_addr_t pc_ipaddr )
```

Setup all the settings for the UDP communication (data and command)

Parameters

<i>pc_ipaddr</i>	IP address of the computer
------------------	----------------------------

Returns

0 if ok, negative values if there is a problem

Note

-

4.35.2.8 transfer_cmd()

```
err_t transfer_cmd (
    char * frame,
    uint16_t length )
```

Send a frame trough UDP.

Parameters

<i>frame</i>	pointer to the frame to send
<i>length</i>	size of the frame

Returns

type err_enum_t: enumeration from err.h file

Note

frame must take in consideration the header ex: for a buffer of size 6 `char* test_array = (char *)malloc(6 + BUF_HEADER_SIZE);` and then `transfer_data(&test_array[BUF_HEADER_SIZE], 6)`

4.35.2.9 transfer_data()

```
err_t transfer_data (
    char * frame,
    uint16_t length )
```

Send a frame trough UDP.

Parameters

<i>frame</i>	pointer to the frame to send
<i>length</i>	size of the frame

Returns

type err_enum_t: enumeration from err.h file

Note

frame must take in consideration the header ex: for a buffer of size 6 `char* test_array = (char *)malloc(6 + BUF_HEADER_SIZE);` and then `transfer_data(&test_array[BUF_HEADER_SIZE], 6)`

4.35.2.10 udp_cmd_rcv()

```
void udp_cmd_rcv (
    void * arg,
    struct udp_pcb * pcb,
    struct pbuf * p,
    const ip_addr_t * addr,
    u16_t port )
```

callback when a command is received

Parameters

<i>arg</i>	not used
<i>pcb</i>	UDP Protocol Control Block used to send the data.
<i>p</i>	pointer to the buffer containing the data received
<i>addr</i>	IP address of the destination
<i>port</i>	UDP port of the destination

Returns

None

Note

-

4.35.3 Variable Documentation**4.35.3.1 buf_cmd**

```
struct pbuf* buf_cmd
```

Buffer structure used to send command packet.

4.35.3.2 buf_data

```
struct pbuf* buf_data
```

Buffer structure used to send data packet.

4.35.3.3 count_scu_timer

```
volatile int count_scu_timer
```

Instance of AXI-DMA.

Instance of AXI-DMA.

4.35.3.4 count_ttcps_timer

```
volatile int count_ttcps_timer
```

Counter of the TTC.

4.35.3.5 frame_buf_cmd

```
char* frame_buf_cmd
```

Buffer used to send the command (50 bytes above it reserved for protocol header)

4.35.3.6 get_20_windows_flag

```
volatile bool get_20_windows_flag
```

Flag raised when the user send the command "get 20 windows".

4.35.3.7 get_transfer_fct_flag

```
volatile bool get_transfer_fct_flag
```

Flag raised when the user send the command "get transfer function".

4.35.3.8 nbre_of_bytes

```
int nbre_of_bytes
```

Number of bytes sent during streaming (trigger mode)

4.35.3.9 pcb_cmd

```
struct udp_pcb* pcb_cmd
```

UDP Protocol Control Block for command communication.

4.35.3.10 pcb_data

```
struct udp_pcb* pcb_data
```

UDP Protocol Control Block for data communication.

4.35.3.11 regptr

```
int* regptr
```

Array containing registers of AXI-lite.

4.35.3.12 run_flag

```
volatile bool run_flag
```

Flag reset when the user send the command "stop uC".

4.35.3.13 stream_flag

```
volatile bool stream_flag
```

Flag raised when the user send the command "start streaming".

4.36 udp_peripheral.h File Reference

```
#include <stdio.h>
#include <string.h>
#include "lwip/err.h"
#include "lwip/udp.h"
#include "stdbool.h"
#include "time_hm.h"
#include "interrupt.h"
#include "TARGETC_RegisterMap.h"
```

Macros

- `#define BUF_HEADER_SIZE 50`
Length of protocol header (in bytes)
- `#define MAX_DATA_SIZE CHANNEL*SAMPLE*2+15`
Length maximum of frame data (in bytes)
- `#define REGMAP_SIZE_UDP 128`
Number of register send/received by UDP.
- `#define MAX_CMD_SIZE 2*REGMAP_SIZE_UDP+20`
Length maximum of frame command (in bytes)
- `#define PORT_CMD 7`
Port used for the command packet transmission.
- `#define PORT_DATA 8`
Port used for the data packet transmission.

Functions

- `err_t transfer_data (char *frame, uint16_t length)`
Send a frame through UDP.
- `void udp_cmd_recv (void *arg, struct udp_pcb *pcb, struct pbuf *p, const ip_addr_t *addr, u16_t port)`
callback when a command is received
- `int command_parser (struct pbuf *p, char *return_buf)`
Process the data received and report a command.
- `int setup_udp_settings (ip_addr_t pc_ipaddr)`
Setup all the settings for the UDP communication (data and command)
- `void cleanup_udp (void)`
Disable and clean the UDP PCB for the data and the command.
- `void print_ip (char *msg, ip_addr_t *ip)`
Function to print an IP address with a message (used in debug mostly)
- `void print_ip_settings (ip_addr_t *ip, ip_addr_t *mask, ip_addr_t *gw)`
Function to print all the settings of an Internet connection (used in debug mostly)
- `void tcp_fasttmr (void)`
- `void tcp_slowtmr (void)`
- `int setup_pcb_data (ip_addr_t pc_ipaddr, uint16_t port)`
Setup the UDP Protocol Control Block for the data frame.
- `int setup_pcb_cmd (ip_addr_t pc_ipaddr, uint16_t port)`
Setup the UDP Protocol Control Block for the command frame.

4.36.1 Macro Definition Documentation

4.36.1.1 BUF_HEADER_SIZE

```
#define BUF_HEADER_SIZE 50
```

Length of protocol header (in bytes)

4.36.1.2 MAX_CMD_SIZE

```
#define MAX_CMD_SIZE 2*REGMAP_SIZE_UDP+20
```

Length maximum of frame command (in bytes)

4.36.1.3 MAX_DATA_SIZE

```
#define MAX_DATA_SIZE CHANNEL*SAMPLE*2+15
```

Length maximum of frame data (in bytes)

4.36.1.4 PORT_CMD

```
#define PORT_CMD 7
```

Port used for the command packet transmission.

4.36.1.5 PORT_DATA

```
#define PORT_DATA 8
```

Port used for the data packet transmission.

4.36.1.6 REGMAP_SIZE_UDP

```
#define REGMAP_SIZE_UDP 128
```

Number of register send/received by UDP.

4.36.2 Function Documentation

4.36.2.1 cleanup_udp()

```
void cleanup_udp (  
    void )
```

Disable and clean the UDP PCB for the data and the command.

Parameters

<i>None</i>	
-------------	--

Returns

None

Note

-

4.36.2.2 command_parser()

```
int command_parser (
    struct pbuf * p,
    char * return_buf )
```

Process the data received and report a command.

Parameters

<i>p</i>	pointer to the buffer containing the data received
<i>return_buf</i>	pointer to the data to be echoed back

Returns

-1 if there is a problem or the number of byte in return_buf

Note

-

4.36.2.3 print_ip()

```
void print_ip (
    char * msg,
    ip_addr_t * ip )
```

Function to print an IP address with a message (used in debug mostly)

Parameters

<i>msg</i>	pointer to the message to print before the IP address
<i>ip</i>	pointer to the IP address to print

Returns

None

Note

-

4.36.2.4 print_ip_settings()

```
void print_ip_settings (
    ip_addr_t * ip,
    ip_addr_t * mask,
    ip_addr_t * gw )
```

Function to print all the settings of an Internet connection (used in debug mostly)

Parameters

<i>ip</i>	pointer to the IP address to print
<i>mask</i>	pointer to the mask to print
<i>gw</i>	pointer to the gateway to print

Returns

None

Note

-

4.36.2.5 setup_pcb_cmd()

```
int setup_pcb_cmd (
    ip_addr_t pc_ipaddr,
    uint16_t port )
```

Setup the UDP Protocol Control Block for the command frame.

Parameters

<i>pc_ipaddr</i>	IP address of the computer
<i>port</i>	UDP port of the computer used for the command

Returns

0 if ok, negative values if there is a problem

Note

-

4.36.2.6 setup_pcb_data()

```
int setup_pcb_data (
    ip_addr_t pc_ipaddr,
    uint16_t port )
```

Setup the UDP Protocol Control Block for the data frame.

Parameters

<i>pc_ipaddr</i>	IP address of the computer
<i>port</i>	UDP port of the computer used for the data

Returns

0 if ok, negative values if there is a problem

Note

-

4.36.2.7 setup_udp_settings()

```
int setup_udp_settings (
    ip_addr_t pc_ipaddr )
```

Setup all the settings for the UDP communication (data and command)

Parameters

<i>pc_ipaddr</i>	IP address of the computer
------------------	----------------------------

Returns

0 if ok, negative values if there is a problem

Note

-

4.36.2.8 tcp_fasttmr()

```
void tcp_fasttmr (  
    void )
```

4.36.2.9 tcp_slowtmr()

```
void tcp_slowtmr (  
    void )
```

4.36.2.10 transfer_data()

```
err_t transfer_data (  
    char * frame,  
    uint16_t length )
```

Send a frame through UDP.

Parameters

<i>frame</i>	pointer to the frame to send
<i>length</i>	size of the frame

Returns

type `err_enum_t`: enumeration from `err.h` file

Note

frame must take in consideration the header ex: for a buffer of size 6 `char* test_array = (char *)malloc(6 + BUF_HEADER_SIZE);` and then `transfer_data(&test_array[BUF_HEADER_SIZE], 6)`

4.36.2.11 udp_cmd_rcv()

```
void udp_cmd_rcv (  
    void * arg,
```



```
struct udp_pcb * pcb,  
struct pbuf * p,  
const ip_addr_t * addr,  
u16_t port )
```

callback when a command is received

Parameters

<i>arg</i>	not used
<i>pcb</i>	UDP Protocol Control Block used to send the data.
<i>p</i>	pointer to the buffer containing the data received
<i>addr</i>	IP address of the destination
<i>port</i>	UDP port of the destination

Returns

None

Note

-

4.37 utility.c File Reference

```
#include "utility.h"
```

Functions

- void [decToHexa](#) (int n)
Function to print a decimal value in hexadecimal.
- void [decToBin](#) (unsigned int n)

4.37.1 Detailed Description

Author

Anthony Schluchin

Date

24th October 2018

Version

0.0

4.37.2 Function Documentation

4.37.2.1 decToBin()

```
void decToBin (
    unsigned int n )
```

4.37.2.2 decToHexa()

```
void decToHexa (
    int n )
```

Function to print a decimal value in hexadecimal.

Parameters

<i>n</i>	decimal value to print
----------	------------------------

Returns

None

Note

-

4.38 utility.h File Reference

```
#include "xil_printf.h"
```

Functions

- void [decToHexa](#) (int n)
Function to print a decimal value in hexadecimal.
- void [decToBin](#) (unsigned int n)

4.38.1 Detailed Description

Author

Anthony Schluchin

Date

24th October 2018

Version

0.0

4.38.2 Function Documentation

4.38.2.1 decToBin()

```
void decToBin (
    unsigned int n )
```

4.38.2.2 decToHexa()

```
void decToHexa (
    int n )
```

Function to print a decimal value in hexadecimal.

Parameters

<i>n</i>	decimal value to print
----------	------------------------

Returns

None

Note

-

Index

a

- GMtype_Ellipsoid, [14](#)
- addtime
 - time_hm.c, [139](#)
 - time_hm.h, [143](#)
- amplitude
 - features_ext_st, [9](#)
- assert_callback
 - interrupt.c, [85](#)
 - interrupt.h, [97](#)
- ATanH
 - GMHeader.h, [69](#)
- axidma_rx_callback
 - interrupt.c, [85](#)
 - interrupt.h, [98](#)
- AxiDmaInstance
 - global.c, [55](#)
 - interrupt.c, [91](#)
- axis_peripheral.c, [23](#)
 - dma_received_data, [24](#)
 - empty_flag, [25](#)
 - first_element, [25](#)
 - flag_axidma_error, [26](#)
 - flag_axidma_rx_done, [26](#)
 - flag_scu_timer, [26](#)
 - flag_ttcs_timer, [26](#)
 - frame_buf, [26](#)
 - last_element, [26](#)
 - regptr, [27](#)
 - test_TPG, [24](#)
 - WdtSculInstance, [27](#)
 - XAxiDma_SimpleTransfer_hm, [25](#)
- axis_peripheral.h, [27](#)
 - dma_received_data, [28](#)
 - FEATURES_ID, [28](#)
 - FULL_WAVEFORM_ID, [28](#)
 - test_TPG, [29](#)
 - XAxiDma_SimpleTransfer_hm, [29](#)

b

- GMtype_Ellipsoid, [14](#)
- buf_cmd
 - udp_peripheral.c, [156](#)
- buf_data
 - udp_peripheral.c, [156](#)
- BUF_HEADER_SIZE
 - udp_peripheral.h, [160](#)
- BUSY_MASK
 - TARGETC_RegisterMap.h, [121](#)

CHANNEL

- data_analysis.h, [33](#)
- CHANNEL_A
 - iic_DAC_LTC2657.h, [79](#)
- CHANNEL_ALL
 - iic_DAC_LTC2657.h, [80](#)
- CHANNEL_B
 - iic_DAC_LTC2657.h, [80](#)
- CHANNEL_C
 - iic_DAC_LTC2657.h, [80](#)
- CHANNEL_D
 - iic_DAC_LTC2657.h, [80](#)
- CHANNEL_E
 - iic_DAC_LTC2657.h, [80](#)
- CHANNEL_F
 - iic_DAC_LTC2657.h, [80](#)
- CHANNEL_G
 - iic_DAC_LTC2657.h, [81](#)
- CHANNEL_H
 - iic_DAC_LTC2657.h, [81](#)
- clean_state_en
 - main.c, [104](#)
- clean_state_enum
 - main.c, [105](#)
- cleanup_global_var
 - global.c, [54](#)
 - global.h, [60](#)
- cleanup_interrupts
 - interrupt.c, [86](#)
 - interrupt.h, [98](#)
- cleanup_udp
 - udp_peripheral.c, [151](#)
 - udp_peripheral.h, [160](#)
- coef
 - GMtype_Polynomial, [18](#)
- columns
 - GMtype_Matrix, [15](#)
- command_parser
 - udp_peripheral.c, [152](#)
 - udp_peripheral.h, [161](#)
- ControlRegisterWrite
 - TARGETC_RegisterMap.c, [114](#)
 - TARGETC_RegisterMap.h, [135](#)
- coordinates
 - data_analysis.h, [35](#)
- coordinates_st, [5](#)
 - x, [5](#)
 - y, [5](#)
- correct_data

- data_analysis.c, [30](#)
- data_analysis.h, [36](#)
- count_scu_timer
 - global.c, [55](#)
 - interrupt.c, [91](#)
 - udp_peripheral.c, [157](#)
- count_ttcs_timer
 - global.c, [56](#)
 - udp_peripheral.c, [157](#)
- CPUMODE_MASK
 - TARGETC_RegisterMap.h, [122](#)
- create_logfile
 - file_hm.c, [39](#)
 - file_hm.h, [43](#)
- create_timefile
 - file_hm.c, [40](#)
 - file_hm.h, [43](#)
- DAC_GRP_0
 - iic_DAC_LTC2657.h, [81](#)
- DAC_GRP_1
 - iic_DAC_LTC2657.h, [81](#)
- DAC_GRP_2
 - iic_DAC_LTC2657.h, [81](#)
- DAC_GRP_3
 - iic_DAC_LTC2657.h, [81](#)
- DAC_LTC2657_initialize
 - iic_DAC_LTC2657.c, [77](#)
 - iic_DAC_LTC2657.h, [82](#)
- DAC_LTC2657_SetChannelVoltage
 - iic_DAC_LTC2657.c, [77](#)
 - iic_DAC_LTC2657.h, [83](#)
- DAC_VPED
 - iic_DAC_LTC2657.h, [82](#)
- data
 - data_axi_st, [6](#)
 - data_list_st, [8](#)
- data_analysis.c, [30](#)
 - correct_data, [30](#)
 - extract_features, [31](#)
 - lookup_table, [31](#)
 - pedestal, [31](#)
- data_analysis.h, [32](#)
 - CHANNEL, [33](#)
 - coordinates, [35](#)
 - correct_data, [36](#)
 - data_axi, [35](#)
 - data_axi_un, [35](#)
 - data_list, [36](#)
 - extract_features, [37](#)
 - features_ext, [36](#)
 - LAST_SHIFT, [33](#)
 - MASK_INFO, [33](#)
 - MAX_WINDOW, [33](#)
 - SAMPLE, [34](#)
 - SIZE_DATA_ARRAY, [34](#)
 - SIZE_DATA_ARRAY_BYT, [34](#)
 - THRESHOLD_CMP, [34](#)
 - THRESHOLD_PULSE, [34](#)
 - time_un, [36](#)
 - TOO_LONG_SHIFT, [34](#)
 - TRIG_SHIFT, [35](#)
 - VPED_ANALOG, [35](#)
 - VPED_DIGITAL, [35](#)
- data_array
 - data_axi_union, [7](#)
- data_axi
 - data_analysis.h, [35](#)
- data_axi_st, [6](#)
 - data, [6](#)
 - info, [6](#)
 - PL_spare, [6](#)
 - wdo_id, [6](#)
 - wdo_time, [6](#)
- data_axi_un
 - data_analysis.h, [35](#)
- data_axi_union, [7](#)
 - data_array, [7](#)
 - data_struct, [7](#)
- data_list
 - data_analysis.h, [36](#)
- data_list_st, [8](#)
 - data, [8](#)
 - next, [8](#)
 - previous, [8](#)
- data_struct
 - data_axi_union, [7](#)
- data_test.c, [37](#)
 - made_frame, [38](#)
- data_test.h, [38](#)
 - made_frame, [38](#)
- Day
 - GMtype_Date, [13](#)
- day
 - time_cplt_st, [19](#)
- day_per_month
 - time_hm.c, [142](#)
- DayNumber
 - GMtype_Date, [13](#)
- DecimalYear
 - GMtype_Date, [13](#)
- decToBin
 - utility.c, [166](#)
 - utility.h, [167](#)
- decToHexa
 - utility.c, [166](#)
 - utility.h, [167](#)
- DEG2RAD
 - GMHeader.h, [69](#)
- degree
 - GMtype_Polynomial, [18](#)
- devices_initialization
 - interrupt.c, [86](#)
 - interrupt.h, [99](#)
- DISABLE
 - TARGETC_RegisterMap.h, [122](#)
- dma_received_data

- axis_peripheral.c, 24
- axis_peripheral.h, 28
- dma_stm_en
 - main.c, 104
- dma_stm_enum
 - main.c, 105
- echo_netif
 - global.c, 56
 - interrupt.c, 91
 - main.c, 106
- element
 - GMtype_Data, 12
 - GMtype_Matrix, 15
- empty_flag
 - axis_peripheral.c, 25
 - global.c, 56
 - interrupt.c, 91
 - main.c, 106
- ENABLE
 - TARGETC_RegisterMap.h, 122
- enable_interrupts
 - interrupt.c, 87
 - interrupt.h, 99
- end_main
 - main.c, 105
- eps
 - GMtype_Ellipsoid, 14
- epssq
 - GMtype_Ellipsoid, 15
- extract_features
 - data_analysis.c, 31
 - data_analysis.h, 37
- FALSE
 - GMHeader.h, 70
- features_ext
 - data_analysis.h, 36
- features_ext_st, 8
 - amplitude, 9
 - time, 9
- FEATURES_ID
 - axis_peripheral.h, 28
- file_hm.c, 39
 - create_logfile, 39
 - create_timefile, 40
 - log_event, 40
 - log_wdtevent, 41
 - mount_sd_card, 41
 - Path, 42
 - update_timefile, 42
- file_hm.h, 42
 - create_logfile, 43
 - create_timefile, 43
 - log_event, 44
 - log_wdtevent, 44
 - mount_sd_card, 45
 - update_timefile, 45
- first_element
 - axis_peripheral.c, 25
 - global.c, 56
 - main.c, 106
- fla
 - GMtype_Ellipsoid, 15
- flag_assertion
 - global.c, 56
 - interrupt.c, 91
 - main.c, 106
- flag_axidma_error
 - axis_peripheral.c, 26
 - get_20_windows.c, 47
 - get_transfer_fct.c, 50
 - global.c, 56
 - interrupt.c, 91
 - pedestal.c, 110
 - transfer_function.c, 147
- flag_axidma_rx
 - global.c, 57
 - interrupt.c, 92
 - main.c, 106
- flag_axidma_rx_done
 - axis_peripheral.c, 26
 - get_20_windows.c, 47
 - get_transfer_fct.c, 50
 - global.c, 57
 - interrupt.c, 92
 - pedestal.c, 110
 - transfer_function.c, 147
- flag_scu_timer
 - axis_peripheral.c, 26
 - get_20_windows.c, 47
 - get_transfer_fct.c, 51
 - global.c, 57
 - interrupt.c, 92
 - main.c, 107
 - pedestal.c, 110
 - transfer_function.c, 147
- flag_ttcps_timer
 - axis_peripheral.c, 26
 - get_20_windows.c, 47
 - get_transfer_fct.c, 51
 - global.c, 57
 - interrupt.c, 92
 - main.c, 107
 - pedestal.c, 110
 - transfer_function.c, 147
- flag_while_loop
 - global.c, 57
 - interrupt.c, 92
 - main.c, 107
- frame_buf
 - axis_peripheral.c, 26
 - get_20_windows.c, 47
 - get_transfer_fct.c, 51
 - global.c, 57
- frame_buf_cmd
 - global.c, 58

- udp_peripheral.c, 157
- frame_buf_cmd_tmp
 - global.c, 58
- frame_buf_tmp
 - global.c, 58
- FULL_WAVEFORM_ID
 - axis_peripheral.h, 28
- g0
 - GMtype_Model, 16
- g1
 - GMtype_Model, 16
- GET_20_WINDOWS
 - main.c, 105
- get_20_windows.c, 46
 - flag_axidma_error, 47
 - flag_axidma_rx_done, 47
 - flag_scu_timer, 47
 - flag_ttcps_timer, 47
 - frame_buf, 47
 - get_20_windows_fct, 46
 - lookup_table, 48
 - pedestal, 48
 - regptr, 48
 - WdtSculInstance, 48
- get_20_windows.h, 48
 - get_20_windows_fct, 49
- get_20_windows_fct
 - get_20_windows.c, 46
 - get_20_windows.h, 49
- get_20_windows_flag
 - global.c, 58
 - main.c, 107
 - udp_peripheral.c, 157
- GET_TRANSFER_FCT
 - main.c, 105
- get_transfer_fct.c, 49
 - flag_axidma_error, 50
 - flag_axidma_rx_done, 50
 - flag_scu_timer, 51
 - flag_ttcps_timer, 51
 - frame_buf, 51
 - lookup_table, 51
 - pedestal, 51
 - regptr, 51
 - send_data_transfer_fct, 50
 - WdtSculInstance, 52
- get_transfer_fct.h, 52
 - send_data_transfer_fct, 52
- get_transfer_fct_flag
 - global.c, 58
 - main.c, 107
 - udp_peripheral.c, 157
- GetTargetCControl
 - TARGETC_RegisterMap.c, 115
 - TARGETC_RegisterMap.h, 136
- GetTargetCStatus
 - TARGETC_RegisterMap.c, 115
 - TARGETC_RegisterMap.h, 136
- gettime_hm
 - time_hm.c, 139
 - time_hm.h, 144
- global.c, 53
 - AxiDmaInstance, 55
 - cleanup_global_var, 54
 - count_scu_timer, 55
 - count_ttcps_timer, 56
 - echo_netif, 56
 - empty_flag, 56
 - first_element, 56
 - flag_assertion, 56
 - flag_axidma_error, 56
 - flag_axidma_rx, 57
 - flag_axidma_rx_done, 57
 - flag_scu_timer, 57
 - flag_ttcps_timer, 57
 - flag_while_loop, 57
 - frame_buf, 57
 - frame_buf_cmd, 58
 - frame_buf_cmd_tmp, 58
 - frame_buf_tmp, 58
 - get_20_windows_flag, 58
 - get_transfer_fct_flag, 58
 - init_global_var, 55
 - last_element, 58
 - lookup_table, 59
 - nbre_of_bytes, 59
 - pedestal, 59
 - regptr, 59
 - run_flag, 59
 - stream_flag, 59
 - WdtSculInstance, 60
- global.h, 60
 - cleanup_global_var, 60
 - init_global_var, 61
- GLOBAL_VAR
 - main.c, 105
- GM_CartesianToSpherical
 - GM_SubLibrary.c, 63
 - GMHeader.h, 71
- GM_CORD
 - GM_SubLibrary.c, 63
 - GMHeader.h, 71
- GM_DateToYear
 - GM_SubLibrary.c, 63
 - GMHeader.h, 71
- GM_DotProduct
 - GM_SubLibrary.c, 63
 - GMHeader.h, 71
- GM_EarthCartToDipoleCartCD
 - GM_SubLibrary.c, 63
 - GMHeader.h, 71
- GM_GeodeticToSpherical
 - GM_SubLibrary.c, 63
 - GMHeader.h, 72
- GM_GetUserInput
 - GM_SubLibrary.c, 64

- GMHeader.h, 72
- GM_LinearInterpolation
 - GM_SubLibrary.c, 64
 - GMHeader.h, 72
- GM_LUDecomposition
 - GM_SubLibrary.c, 64
 - GMHeader.h, 72
- GM_LUSolve
 - GM_SubLibrary.c, 64
 - GMHeader.h, 72
- GM_MatDet
 - GM_SubLibrary.c, 64
 - GMHeader.h, 73
- GM_MatInverse
 - GM_SubLibrary.c, 65
 - GMHeader.h, 73
- GM_MatMultiply
 - GM_SubLibrary.c, 65
 - GMHeader.h, 73
- GM_MatTranspose
 - GM_SubLibrary.c, 65
 - GMHeader.h, 73
- GM_Mean
 - GM_SubLibrary.c, 65
 - GMHeader.h, 73
- GM_Median
 - GM_SubLibrary.c, 65
 - GMHeader.h, 73
- GM_PoleLocation
 - GM_SubLibrary.c, 65
 - GMHeader.h, 74
- GM_PolyFit
 - GM_SubLibrary.c, 66
 - GMHeader.h, 74
- GM_Pow
 - GM_SubLibrary.c, 66
 - GMHeader.h, 74
- GM_PrintMatrix
 - GM_SubLibrary.c, 66
 - GMHeader.h, 74
- GM_PrintUserData
 - GM_SubLibrary.c, 66
 - GMHeader.h, 74
- GM_ScanIGRF
 - GM_SubLibrary.c, 66
 - GMHeader.h, 74
- GM_SetEllipsoid
 - GM_SubLibrary.c, 66
 - GMHeader.h, 75
- GM_SolvePolynomial
 - GM_SubLibrary.c, 67
 - GMHeader.h, 75
- GM_Sort
 - GM_SubLibrary.c, 67
 - GMHeader.h, 75
- GM_SphericalToCartesian
 - GM_SubLibrary.c, 67
 - GMHeader.h, 75
- GM_StandardDeviation
 - GM_SubLibrary.c, 67
 - GMHeader.h, 75
- GM_STARTYEAR
 - GMHeader.h, 70
- GM_SubLibrary.c, 61
 - GM_CartesianToSpherical, 63
 - GM_CORD, 63
 - GM_DateToYear, 63
 - GM_DotProduct, 63
 - GM_EarthCartToDipoleCartCD, 63
 - GM_GeodeticToSpherical, 63
 - GM_GetUserInput, 64
 - GM_LinearInterpolation, 64
 - GM_LUDecomposition, 64
 - GM_LUSolve, 64
 - GM_MatDet, 64
 - GM_MatInverse, 65
 - GM_MatMultiply, 65
 - GM_MatTranspose, 65
 - GM_Mean, 65
 - GM_Median, 65
 - GM_PoleLocation, 65
 - GM_PolyFit, 66
 - GM_Pow, 66
 - GM_PrintMatrix, 66
 - GM_PrintUserData, 66
 - GM_ScanIGRF, 66
 - GM_SetEllipsoid, 66
 - GM_SolvePolynomial, 67
 - GM_Sort, 67
 - GM_SphericalToCartesian, 67
 - GM_StandardDeviation, 67
 - GM_Swap, 67
 - GM_SwapRows, 67
 - GM_TimeAdjustCoefs, 68
- GM_Swap
 - GM_SubLibrary.c, 67
 - GMHeader.h, 75
- GM_SwapRows
 - GM_SubLibrary.c, 67
 - GMHeader.h, 76
- GM_TimeAdjustCoefs
 - GM_SubLibrary.c, 68
 - GMHeader.h, 76
- GMHeader.h, 68
 - ATanH, 69
 - DEG2RAD, 69
 - FALSE, 70
 - GM_CartesianToSpherical, 71
 - GM_CORD, 71
 - GM_DateToYear, 71
 - GM_DotProduct, 71
 - GM_EarthCartToDipoleCartCD, 71
 - GM_GeodeticToSpherical, 72
 - GM_GetUserInput, 72
 - GM_LinearInterpolation, 72
 - GM_LUDecomposition, 72

- GM_LUSolve, 72
- GM_MatDet, 73
- GM_MatInverse, 73
- GM_MatMultiply, 73
- GM_MatTranspose, 73
- GM_Mean, 73
- GM_Median, 73
- GM_PoleLocation, 74
- GM_PolyFit, 74
- GM_Pow, 74
- GM_PrintMatrix, 74
- GM_PrintUserData, 74
- GM_ScanIGRF, 74
- GM_SetEllipsoid, 75
- GM_SolvePolynomial, 75
- GM_Sort, 75
- GM_SphericalToCartesian, 75
- GM_StandardDeviation, 75
- GM_STARTYEAR, 70
- GM_Swap, 75
- GM_SwapRows, 76
- GM_TimeAdjustCoefs, 76
- M_PI, 70
- MU_0, 70
- R_e, 70
- RAD2DEG, 70
- TRUE, 70
- GMtype_CoordCartesian, 9
 - x, 9
 - y, 10
 - z, 10
- GMtype_CoordDipole, 10
 - lambda, 10
 - phi, 10
- GMtype_CoordGeodetic, 11
 - HeightAboveEllipsoid, 11
 - lambda, 11
 - phi, 11
- GMtype_CoordSpherical, 11
 - lambda, 12
 - phig, 12
 - r, 12
- GMtype_Data, 12
 - element, 12
 - size, 12
- GMtype_Date, 13
 - Day, 13
 - DayNumber, 13
 - DecimalYear, 13
 - Month, 13
 - Year, 14
- GMtype_Ellipsoid, 14
 - a, 14
 - b, 14
 - eps, 14
 - epssq, 15
 - fla, 15
 - re, 15
- GMtype_Matrix, 15
 - columns, 15
 - element, 15
 - rows, 16
- GMtype_Model, 16
 - g0, 16
 - g1, 16
 - h1, 16
- GMtype_Pole, 17
 - lambda, 17
 - M, 17
 - phi, 17
- GMtype_Polynomial, 18
 - coef, 18
 - degree, 18
- h1
 - GMtype_Model, 16
- HeightAboveEllipsoid
 - GMtype_CoordGeodetic, 11
- hour
 - time_cp1t_st, 19
- I2cInstance
 - iic_DAC_LTC2657.c, 78
- IDLE
 - main.c, 105
- iic_DAC_LTC2657.c, 76
 - DAC_LTC2657_initialize, 77
 - DAC_LTC2657_SetChannelVoltage, 77
 - I2cInstance, 78
- iic_DAC_LTC2657.h, 78
 - CHANNEL_A, 79
 - CHANNEL_ALL, 80
 - CHANNEL_B, 80
 - CHANNEL_C, 80
 - CHANNEL_D, 80
 - CHANNEL_E, 80
 - CHANNEL_F, 80
 - CHANNEL_G, 81
 - CHANNEL_H, 81
 - DAC_GRP_0, 81
 - DAC_GRP_1, 81
 - DAC_GRP_2, 81
 - DAC_GRP_3, 81
 - DAC_LTC2657_initialize, 82
 - DAC_LTC2657_SetChannelVoltage, 83
 - DAC_VPED, 82
 - IIC_DAC_LTC2657_H, 82
 - IIC_DEVICE_ID, 82
 - IIC_SLAVE_ADDRESS, 82
- IIC_DAC_LTC2657_H
 - iic_DAC_LTC2657.h, 82
- IIC_DEVICE_ID
 - iic_DAC_LTC2657.h, 82
- IIC_SLAVE_ADDRESS
 - iic_DAC_LTC2657.h, 82
- info
 - data_axi_st, 6

- INIT
 - TARGETC_RegisterMap.h, 122
- init_global_var
 - global.c, 55
 - global.h, 61
- init_pedestals
 - pedestal.c, 109
 - pedestal.h, 112
- init_transfer_function
 - transfer_function.c, 147
 - transfer_function.h, 149
- INTC_BASE_ADDR
 - interrupt.h, 95
- INTC_DEVICE_ID
 - interrupt.h, 95
- INTC_DIST_BASE_ADDR
 - interrupt.h, 95
- INTERRUPT
 - main.c, 105
- interrupt.c, 83
 - assert_callback, 85
 - axidma_rx_callback, 85
 - AxiDmaInstance, 91
 - cleanup_interrups, 86
 - count_scu_timer, 91
 - devices_initialization, 86
 - echo_netif, 91
 - empty_flag, 91
 - enable_interrups, 87
 - flag_assertion, 91
 - flag_axidma_error, 91
 - flag_axidma_rx, 92
 - flag_axidma_rx_done, 92
 - flag_scu_timer, 92
 - flag_ttcps_timer, 92
 - flag_while_loop, 92
 - interrups_initialization, 87
 - last_element, 92
 - setup_axidma_int, 87
 - setup_scu_timer_int, 88
 - setup_scu_wdt_int, 88
 - setup_ttcps_timer_int, 89
 - stream_flag, 93
 - timer_scu_callback, 89
 - timer_ttcps_callback, 90
 - wdt_scu_callback, 90
 - WdtSculInstance, 93
- interrupt.h, 93
 - assert_callback, 97
 - axidma_rx_callback, 98
 - cleanup_interrups, 98
 - devices_initialization, 99
 - enable_interrups, 99
 - INTC_BASE_ADDR, 95
 - INTC_DEVICE_ID, 95
 - INTC_DIST_BASE_ADDR, 95
 - interrups_initialization, 99
 - RESET_RX_CNTR_LIMIT, 96
 - setup_axidma_int, 100
 - setup_scu_timer_int, 100
 - setup_ttcps_timer_int, 101
 - TIMER_DEVICE_ID, 96
 - TIMER_IRPT_INTR, 96
 - timer_scu_callback, 101
 - timer_ttcps_callback, 102
 - TmrCntrSetup, 97
 - TTC_TICK_DEVICE_ID, 96
 - TTC_TICK_INTR_ID, 96
 - TTCPS_TIMER_FREQ_HZ, 96
 - WDT_DEVICE_ID, 97
 - WDT_IRPT_INTR, 97
 - WDT_LOAD_VALUE, 97
 - wdt_scu_callback, 102
- interrups_initialization
 - interrupt.c, 87
 - interrupt.h, 99
- Interval
 - TmrCntrSetup_st, 21
- isALeapYear
 - time_hm.c, 139
 - time_hm.h, 144
- lambda
 - GMtype_CoordDipole, 10
 - GMtype_CoordGeodetic, 11
 - GMtype_CoordSpherical, 12
 - GMtype_Pole, 17
- last_element
 - axis_peripheral.c, 26
 - global.c, 58
 - interrupt.c, 92
- LAST_REGISTER_ADDR
 - TARGETC_RegisterMap.h, 122
- LAST_SHIFT
 - data_analysis.h, 33
- LOCKED_MASK
 - TARGETC_RegisterMap.h, 122
- log_event
 - file_hm.c, 40
 - file_hm.h, 44
- log_wdtevent
 - file_hm.c, 41
 - file_hm.h, 44
- lookup_table
 - data_analysis.c, 31
 - get_20_windows.c, 48
 - get_transfer_fct.c, 51
 - global.c, 59
 - transfer_function.c, 148
- M
 - GMtype_Pole, 17
- M_PI
 - GMHeader.h, 70
- made_frame
 - data_test.c, 38
 - data_test.h, 38

- main
 - main.c, [106](#)
- main.c, [103](#)
 - clean_state_en, [104](#)
 - clean_state_enum, [105](#)
 - dma_stm_en, [104](#)
 - dma_stm_enum, [105](#)
 - echo_netif, [106](#)
 - empty_flag, [106](#)
 - end_main, [105](#)
 - first_element, [106](#)
 - flag_assertion, [106](#)
 - flag_axidma_rx, [106](#)
 - flag_scu_timer, [107](#)
 - flag_ttcps_timer, [107](#)
 - flag_while_loop, [107](#)
 - GET_20_WINDOWS, [105](#)
 - get_20_windows_flag, [107](#)
 - GET_TRANSFER_FCT, [105](#)
 - get_transfer_fct_flag, [107](#)
 - GLOBAL_VAR, [105](#)
 - IDLE, [105](#)
 - INTERRUPT, [105](#)
 - main, [106](#)
 - regptr, [107](#)
 - run_flag, [108](#)
 - STREAM, [105](#)
 - stream_flag, [108](#)
 - UDP, [105](#)
 - WdtSculInstance, [108](#)
- MASK_INFO
 - data_analysis.h, [33](#)
- MAX_CMD_SIZE
 - udp_peripheral.h, [160](#)
- MAX_DATA_SIZE
 - udp_peripheral.h, [160](#)
- MAX_WINDOW
 - data_analysis.h, [33](#)
- milisecond
 - time_cplt_st, [19](#)
- minute
 - time_cplt_st, [19](#)
- Month
 - GMtype_Date, [13](#)
- month
 - time_cplt_st, [19](#)
- mount_sd_card
 - file_hm.c, [41](#)
 - file_hm.h, [45](#)
- MU_0
 - GMHeader.h, [70](#)
- nbre_of_bytes
 - global.c, [59](#)
 - udp_peripheral.c, [157](#)
- next
 - data_list_st, [8](#)
- offset_counter
 - time_hm.c, [142](#)
- offset_time
 - time_hm.c, [142](#)
- Options
 - TmrCntrSetup_st, [21](#)
- OutputHz
 - TmrCntrSetup_st, [21](#)
- Path
 - file_hm.c, [42](#)
- pcb_cmd
 - udp_peripheral.c, [158](#)
- pcb_data
 - udp_peripheral.c, [158](#)
- pedestal
 - data_analysis.c, [31](#)
 - get_20_windows.c, [48](#)
 - get_transfer_fct.c, [51](#)
 - global.c, [59](#)
 - pedestal.c, [110](#)
 - transfer_function.c, [148](#)
- pedestal.c, [108](#)
 - flag_axidma_error, [110](#)
 - flag_axidma_rx_done, [110](#)
 - flag_scu_timer, [110](#)
 - flag_ttcps_timer, [110](#)
 - init_pedestals, [109](#)
 - pedestal, [110](#)
 - regptr, [111](#)
 - WdtSculInstance, [111](#)
- pedestal.h, [111](#)
 - init_pedestals, [112](#)
- phi
 - GMtype_CoordDipole, [10](#)
 - GMtype_CoordGeodetic, [11](#)
 - GMtype_Pole, [17](#)
- phig
 - GMtype_CoordSpherical, [12](#)
- PL_spare
 - data_axi_st, [6](#)
- platform_config.h, [112](#)
 - PLATFORM_EMAC_BASEADDR, [112](#)
 - PLATFORM_ZYNQ, [112](#)
- PLATFORM_EMAC_BASEADDR
 - platform_config.h, [112](#)
- platform_mb.c, [113](#)
- platform_ppc.c, [113](#)
- PLATFORM_ZYNQ
 - platform_config.h, [112](#)
- PORT_CMD
 - udp_peripheral.h, [160](#)
- PORT_DATA
 - udp_peripheral.h, [160](#)
- Prescaler
 - TmrCntrSetup_st, [21](#)
- previous
 - data_list_st, [8](#)
- print_ip
 - udp_peripheral.c, [152](#)

- udp_peripheral.h, 161
- print_ip_settings
 - udp_peripheral.c, 153
 - udp_peripheral.h, 162
- PSBUSY_MASK
 - TARGETC_RegisterMap.h, 123
- r
 - GMtype_CoordSpherical, 12
- R_e
 - GMHeader.h, 70
- RAD2DEG
 - GMHeader.h, 70
- re
 - GMtype_Ellipsoid, 15
- REGCLR_MASK
 - TARGETC_RegisterMap.h, 123
- REGMAP_SIZE_UDP
 - udp_peripheral.h, 160
- regptr
 - axis_peripheral.c, 27
 - get_20_windows.c, 48
 - get_transfer_fct.c, 51
 - global.c, 59
 - main.c, 107
 - pedestal.c, 111
 - TARGETC_RegisterMap.c, 117
 - transfer_function.c, 148
 - udp_peripheral.c, 158
- RESET_RX_CNTR_LIMIT
 - interrupt.h, 96
- rows
 - GMtype_Matrix, 16
- run_flag
 - global.c, 59
 - main.c, 108
 - udp_peripheral.c, 158
- SAMPLE
 - data_analysis.h, 34
- second
 - time_cp1t_st, 19
- send_data_transfer_fct
 - get_transfer_fct.c, 50
 - get_transfer_fct.h, 52
- SetTargetCRegisters
 - TARGETC_RegisterMap.c, 116
 - TARGETC_RegisterMap.h, 136
- settime_hm
 - time_hm.c, 141
 - time_hm.h, 145
- setup_axidma_int
 - interrupt.c, 87
 - interrupt.h, 100
- setup_pcb_cmd
 - udp_peripheral.c, 153
 - udp_peripheral.h, 162
- setup_pcb_data
 - udp_peripheral.c, 154
- udp_peripheral.h, 163
- setup_scu_timer_int
 - interrupt.c, 88
 - interrupt.h, 100
- setup_scu_wdt_int
 - interrupt.c, 88
- setup_ttcps_timer_int
 - interrupt.c, 89
 - interrupt.h, 101
- setup_udp_settings
 - udp_peripheral.c, 154
 - udp_peripheral.h, 163
- sfp.c, 113
- si5324.c, 113
- size
 - GMtype_Data, 12
- SIZE_DATA_ARRAY
 - data_analysis.h, 34
- SIZE_DATA_ARRAY_BYT
 - data_analysis.h, 34
- SMODE_MASK
 - TARGETC_RegisterMap.h, 123
- SS_TPG_MASK
 - TARGETC_RegisterMap.h, 123
- SSVALID_MASK
 - TARGETC_RegisterMap.h, 123
- STORAGE_MASK
 - TARGETC_RegisterMap.h, 123
- STREAM
 - main.c, 105
- stream_flag
 - global.c, 59
 - interrupt.c, 93
 - main.c, 108
 - udp_peripheral.c, 158
- stringtotime
 - time_hm.c, 141
 - time_hm.h, 145
- SWRESET_MASK
 - TARGETC_RegisterMap.h, 124
- TARGETC_RegisterMap.c, 114
 - ControlRegisterWrite, 114
 - GetTargetCControl, 115
 - GetTargetCStatus, 115
 - regptr, 117
 - SetTargetCRegisters, 116
 - WriteReadBackRegister, 116
 - WriteRegister, 117
- TARGETC_RegisterMap.h, 117
 - BUSY_MASK, 121
 - ControlRegisterWrite, 135
 - CPUMODE_MASK, 122
 - DISABLE, 122
 - ENABLE, 122
 - GetTargetCControl, 136
 - GetTargetCStatus, 136
 - INIT, 122
 - LAST_REGISTER_ADDR, 122

LOCKED_MASK, [122](#)
 PSBUSY_MASK, [123](#)
 REGCLR_MASK, [123](#)
 SetTargetCRegisters, [136](#)
 SMODE_MASK, [123](#)
 SS_TPG_MASK, [123](#)
 SSVALID_MASK, [123](#)
 STORAGE_MASK, [123](#)
 SWRESET_MASK, [124](#)
 TARGETC_REGISTERMAP_H, [124](#)
 TC_ADDR_REG, [124](#)
 TC_CMPBIAS2_REG, [124](#)
 TC_CMPBIASIN_REG, [124](#)
 TC_CONTROL_REG, [124](#)
 TC_DATA_OUT_REG, [125](#)
 TC_DBBIAS_REG, [125](#)
 TC_eDO_CH0_REG, [125](#)
 TC_eDO_CH10_REG, [125](#)
 TC_eDO_CH11_REG, [125](#)
 TC_eDO_CH12_REG, [125](#)
 TC_eDO_CH13_REG, [126](#)
 TC_eDO_CH14_REG, [126](#)
 TC_eDO_CH15_REG, [126](#)
 TC_eDO_CH1_REG, [126](#)
 TC_eDO_CH2_REG, [126](#)
 TC_eDO_CH3_REG, [126](#)
 TC_eDO_CH4_REG, [127](#)
 TC_eDO_CH5_REG, [127](#)
 TC_eDO_CH6_REG, [127](#)
 TC_eDO_CH7_REG, [127](#)
 TC_eDO_CH8_REG, [127](#)
 TC_eDO_CH9_REG, [127](#)
 TC_FSTWINDOW_REG, [128](#)
 TC_ISEL_REG, [128](#)
 TC_MISCDIG_REG, [128](#)
 TC_MONTIMING_REG, [128](#)
 TC_MT_PASS_MASK, [128](#)
 TC_MT_SSPIN_MASK, [128](#)
 TC_MT_SSPOUT_MASK, [129](#)
 TC_MT_SSTOUT_MASK, [129](#)
 TC_MT_SSTOUTFB_MASK, [129](#)
 TC_MT_VDD_MASK, [129](#)
 TC_MT_WR1_ADDR_SYNC_MASK, [129](#)
 TC_MT_WR2_ADDR_SYNC_MASK, [129](#)
 TC_MT_WR_STRB1_MASK, [130](#)
 TC_MT_WR_STRB2_MASK, [130](#)
 TC_NBRWINDOW_REG, [130](#)
 TC_PUBIAS_REG, [130](#)
 TC_QBIAS_REG, [130](#)
 TC_SBBIAS_REG, [130](#)
 TC_SSPIN_LE_REG, [131](#)
 TC_SSPIN_TE_REG, [131](#)
 TC_SSTOUTFB_REG, [131](#)
 TC_STATUS_REG, [131](#)
 TC_TPG_REG, [131](#)
 TC_VADJN_REG, [131](#)
 TC_VADJP_REG, [132](#)
 TC_VANBUFF_REG, [132](#)
 TC_VAPBUFF_REG, [132](#)
 TC_VBIAS_REG, [132](#)
 TC_VDISCH_REG, [132](#)
 TC_VDLTYTUNE_REG, [132](#)
 TC_VQBUFF_REG, [133](#)
 TC_VTRIMT_REG, [133](#)
 TC_WR1_ADDR_LE_REG, [133](#)
 TC_WR1_ADDR_TE_REG, [133](#)
 TC_WR2_ADDR_LE_REG, [133](#)
 TC_WR2_ADDR_TE_REG, [133](#)
 TC_WR_STRB1_LE_REG, [134](#)
 TC_WR_STRB1_TE_REG, [134](#)
 TC_WR_STRB2_LE_REG, [134](#)
 TC_WR_STRB2_TE_REG, [134](#)
 TESTFIFO_MASK, [134](#)
 TESTSTREAM_MASK, [134](#)
 WINDOW_MASK, [135](#)
 WINDOWBUSY_MASK, [135](#)
 WRITE_MASK, [135](#)
 WriteReadBackRegister, [137](#)
 WriteRegister, [137](#)
 TARGETC_REGISTERMAP_H
 TARGETC_RegisterMap.h, [124](#)
 TC_ADDR_REG
 TARGETC_RegisterMap.h, [124](#)
 TC_CMPBIAS2_REG
 TARGETC_RegisterMap.h, [124](#)
 TC_CMPBIASIN_REG
 TARGETC_RegisterMap.h, [124](#)
 TC_CONTROL_REG
 TARGETC_RegisterMap.h, [124](#)
 TC_DATA_OUT_REG
 TARGETC_RegisterMap.h, [125](#)
 TC_DBBIAS_REG
 TARGETC_RegisterMap.h, [125](#)
 TC_eDO_CH0_REG
 TARGETC_RegisterMap.h, [125](#)
 TC_eDO_CH10_REG
 TARGETC_RegisterMap.h, [125](#)
 TC_eDO_CH11_REG
 TARGETC_RegisterMap.h, [125](#)
 TC_eDO_CH12_REG
 TARGETC_RegisterMap.h, [125](#)
 TC_eDO_CH13_REG
 TARGETC_RegisterMap.h, [126](#)
 TC_eDO_CH14_REG
 TARGETC_RegisterMap.h, [126](#)
 TC_eDO_CH15_REG
 TARGETC_RegisterMap.h, [126](#)
 TC_eDO_CH1_REG
 TARGETC_RegisterMap.h, [126](#)
 TC_eDO_CH2_REG
 TARGETC_RegisterMap.h, [126](#)
 TC_eDO_CH3_REG
 TARGETC_RegisterMap.h, [126](#)
 TC_eDO_CH4_REG
 TARGETC_RegisterMap.h, [127](#)
 TC_eDO_CH5_REG

- TARGETC_RegisterMap.h, [127](#)
- TC_eDO_CH6_REG
 - TARGETC_RegisterMap.h, [127](#)
- TC_eDO_CH7_REG
 - TARGETC_RegisterMap.h, [127](#)
- TC_eDO_CH8_REG
 - TARGETC_RegisterMap.h, [127](#)
- TC_eDO_CH9_REG
 - TARGETC_RegisterMap.h, [127](#)
- TC_FSTWINDOW_REG
 - TARGETC_RegisterMap.h, [128](#)
- TC_ISEL_REG
 - TARGETC_RegisterMap.h, [128](#)
- TC_MISCDIG_REG
 - TARGETC_RegisterMap.h, [128](#)
- TC_MONTIMING_REG
 - TARGETC_RegisterMap.h, [128](#)
- TC_MT_PASS_MASK
 - TARGETC_RegisterMap.h, [128](#)
- TC_MT_SSPIN_MASK
 - TARGETC_RegisterMap.h, [128](#)
- TC_MT_SSPOUT_MASK
 - TARGETC_RegisterMap.h, [129](#)
- TC_MT_SSTOUT_MASK
 - TARGETC_RegisterMap.h, [129](#)
- TC_MT_SSTOUTFB_MASK
 - TARGETC_RegisterMap.h, [129](#)
- TC_MT_VDD_MASK
 - TARGETC_RegisterMap.h, [129](#)
- TC_MT_WR1_ADDR_SYNC_MASK
 - TARGETC_RegisterMap.h, [129](#)
- TC_MT_WR2_ADDR_SYNC_MASK
 - TARGETC_RegisterMap.h, [129](#)
- TC_MT_WR_STRB1_MASK
 - TARGETC_RegisterMap.h, [130](#)
- TC_MT_WR_STRB2_MASK
 - TARGETC_RegisterMap.h, [130](#)
- TC_NBRWINDOW_REG
 - TARGETC_RegisterMap.h, [130](#)
- TC_PUBIAS_REG
 - TARGETC_RegisterMap.h, [130](#)
- TC_QBIAS_REG
 - TARGETC_RegisterMap.h, [130](#)
- TC_SBBIAS_REG
 - TARGETC_RegisterMap.h, [130](#)
- TC_SSPIN_LE_REG
 - TARGETC_RegisterMap.h, [131](#)
- TC_SSPIN_TE_REG
 - TARGETC_RegisterMap.h, [131](#)
- TC_SSTOUTFB_REG
 - TARGETC_RegisterMap.h, [131](#)
- TC_STATUS_REG
 - TARGETC_RegisterMap.h, [131](#)
- TC_TPG_REG
 - TARGETC_RegisterMap.h, [131](#)
- TC_VADJN_REG
 - TARGETC_RegisterMap.h, [131](#)
- TC_VADJP_REG
 - TARGETC_RegisterMap.h, [132](#)
- TC_VANBUFF_REG
 - TARGETC_RegisterMap.h, [132](#)
- TC_VAPBUFF_REG
 - TARGETC_RegisterMap.h, [132](#)
- TC_VBIAS_REG
 - TARGETC_RegisterMap.h, [132](#)
- TC_VDISCH_REG
 - TARGETC_RegisterMap.h, [132](#)
- TC_VDLYTUNE_REG
 - TARGETC_RegisterMap.h, [132](#)
- TC_VQBUFF_REG
 - TARGETC_RegisterMap.h, [133](#)
- TC_VTRIMT_REG
 - TARGETC_RegisterMap.h, [133](#)
- TC_WR1_ADDR_LE_REG
 - TARGETC_RegisterMap.h, [133](#)
- TC_WR1_ADDR_TE_REG
 - TARGETC_RegisterMap.h, [133](#)
- TC_WR2_ADDR_LE_REG
 - TARGETC_RegisterMap.h, [133](#)
- TC_WR2_ADDR_TE_REG
 - TARGETC_RegisterMap.h, [133](#)
- TC_WR_STRB1_LE_REG
 - TARGETC_RegisterMap.h, [134](#)
- TC_WR_STRB1_TE_REG
 - TARGETC_RegisterMap.h, [134](#)
- TC_WR_STRB2_LE_REG
 - TARGETC_RegisterMap.h, [134](#)
- TC_WR_STRB2_TE_REG
 - TARGETC_RegisterMap.h, [134](#)
- tcp_fasttmr
 - udp_peripheral.h, [164](#)
- tcp_slowtmr
 - udp_peripheral.h, [164](#)
- test_TPG
 - axis_peripheral.c, [24](#)
 - axis_peripheral.h, [29](#)
- TESTFIFO_MASK
 - TARGETC_RegisterMap.h, [134](#)
- TESTSTREAM_MASK
 - TARGETC_RegisterMap.h, [134](#)
- THRESHOLD_CMP
 - data_analysis.h, [34](#)
- THRESHOLD_PULSE
 - data_analysis.h, [34](#)
- time
 - features_ext_st, [9](#)
- time_cplt
 - time_hm.h, [143](#)
- time_cplt_st, [18](#)
 - day, [19](#)
 - hour, [19](#)
 - millisecond, [19](#)
 - minute, [19](#)
 - month, [19](#)
 - second, [19](#)
 - year, [19](#)

- time_fl
 - time_union, 20
- time_hm.c, 138
 - addtime, 139
 - day_per_month, 142
 - gettime_hm, 139
 - isALeapYear, 139
 - offset_counter, 142
 - offset_time, 142
 - settime_hm, 141
 - stringtotime, 141
- time_hm.h, 142
 - addtime, 143
 - gettime_hm, 144
 - isALeapYear, 144
 - settime_hm, 145
 - stringtotime, 145
 - time_cplt, 143
- time_t
 - time_union, 20
- time_un
 - data_analysis.h, 36
- time_union, 20
 - time_fl, 20
 - time_t, 20
- TIMER_DEVICE_ID
 - interrupt.h, 96
- TIMER_IRPT_INTR
 - interrupt.h, 96
- timer_scu_callback
 - interrupt.c, 89
 - interrupt.h, 101
- timer_ttcps_callback
 - interrupt.c, 90
 - interrupt.h, 102
- TmrCntrSetup
 - interrupt.h, 97
- TmrCntrSetup_st, 21
 - Interval, 21
 - Options, 21
 - OutputHz, 21
 - Prescaler, 21
- TOO_LONG_SHIFT
 - data_analysis.h, 34
- transfer_cmd
 - udp_peripheral.c, 155
- transfer_data
 - udp_peripheral.c, 155
 - udp_peripheral.h, 164
- transfer_function.c, 146
 - flag_axidma_error, 147
 - flag_axidma_rx_done, 147
 - flag_scu_timer, 147
 - flag_ttcps_timer, 147
 - init_transfer_function, 147
 - lookup_table, 148
 - pedestal, 148
 - regptr, 148
 - WdtScuInstance, 148
- transfer_function.h, 148
 - init_transfer_function, 149
- TRIG_SHIFT
 - data_analysis.h, 35
- TRUE
 - GMHeader.h, 70
- TTC_TICK_DEVICE_ID
 - interrupt.h, 96
- TTC_TICK_INTR_ID
 - interrupt.h, 96
- TTCPS_TIMER_FREQ_HZ
 - interrupt.h, 96
- UDP
 - main.c, 105
- udp_cmd_recv
 - udp_peripheral.c, 156
 - udp_peripheral.h, 164
- udp_peripheral.c, 150
 - buf_cmd, 156
 - buf_data, 156
 - cleanup_udp, 151
 - command_parser, 152
 - count_scu_timer, 157
 - count_ttcps_timer, 157
 - frame_buf_cmd, 157
 - get_20_windows_flag, 157
 - get_transfer_fct_flag, 157
 - nbre_of_bytes, 157
 - pcb_cmd, 158
 - pcb_data, 158
 - print_ip, 152
 - print_ip_settings, 153
 - regptr, 158
 - run_flag, 158
 - setup_pcb_cmd, 153
 - setup_pcb_data, 154
 - setup_udp_settings, 154
 - stream_flag, 158
 - transfer_cmd, 155
 - transfer_data, 155
 - udp_cmd_recv, 156
- udp_peripheral.h, 159
 - BUF_HEADER_SIZE, 160
 - cleanup_udp, 160
 - command_parser, 161
 - MAX_CMD_SIZE, 160
 - MAX_DATA_SIZE, 160
 - PORT_CMD, 160
 - PORT_DATA, 160
 - print_ip, 161
 - print_ip_settings, 162
 - REGMAP_SIZE_UDP, 160
 - setup_pcb_cmd, 162
 - setup_pcb_data, 163
 - setup_udp_settings, 163
 - tcp_fasttmr, 164
 - tcp_slowtmr, 164

- transfer_data, [164](#)
- udp_cmd_recv, [164](#)
- update_timefile
 - file_hm.c, [42](#)
 - file_hm.h, [45](#)
- utility.c, [165](#)
 - decToBin, [166](#)
 - decToHexa, [166](#)
- utility.h, [166](#)
 - decToBin, [167](#)
 - decToHexa, [167](#)
- VPED_ANALOG
 - data_analysis.h, [35](#)
- VPED_DIGITAL
 - data_analysis.h, [35](#)
- wdo_id
 - data_axi_st, [6](#)
- wdo_time
 - data_axi_st, [6](#)
- WDT_DEVICE_ID
 - interrupt.h, [97](#)
- WDT_IRPT_INTR
 - interrupt.h, [97](#)
- WDT_LOAD_VALUE
 - interrupt.h, [97](#)
- wdt_scu_callback
 - interrupt.c, [90](#)
 - interrupt.h, [102](#)
- WdtSculInstance
 - axis_peripheral.c, [27](#)
 - get_20_windows.c, [48](#)
 - get_transfer_fct.c, [52](#)
 - global.c, [60](#)
 - interrupt.c, [93](#)
 - main.c, [108](#)
 - pedestal.c, [111](#)
 - transfer_function.c, [148](#)
- WINDOW_MASK
 - TARGETC_RegisterMap.h, [135](#)
- WINDOWBUSY_MASK
 - TARGETC_RegisterMap.h, [135](#)
- WRITE_MASK
 - TARGETC_RegisterMap.h, [135](#)
- WriteReadBackRegister
 - TARGETC_RegisterMap.c, [116](#)
 - TARGETC_RegisterMap.h, [137](#)
- WriteRegister
 - TARGETC_RegisterMap.c, [117](#)
 - TARGETC_RegisterMap.h, [137](#)
- x
 - coordinates_st, [5](#)
 - GMtype_CoordCartesian, [9](#)
- XAxiDma_SimpleTransfer_hm
 - axis_peripheral.c, [25](#)
 - axis_peripheral.h, [29](#)
- y
 - coordinates_st, [5](#)
 - GMtype_CoordCartesian, [10](#)
- Year
 - GMtype_Date, [14](#)
- year
 - time_cplt_st, [19](#)
- z
 - GMtype_CoordCartesian, [10](#)