

Yavin IV Death Star Tracker
V1.2

Generated by Doxygen 1.8.8

Tue Nov 4 2014 15:19:12

Contents

1	Todo List	1
2	Data Structure Index	3
2.1	Data Structures	3
3	File Index	5
3.1	File List	5
4	Data Structure Documentation	7
4.1	circularBuffer Struct Reference	7
4.1.1	Detailed Description	7
4.1.2	Field Documentation	7
4.1.2.1	data	7
4.1.2.2	head	7
4.1.2.3	tail	7
4.2	Delay Struct Reference	7
4.2.1	Detailed Description	8
4.2.2	Field Documentation	8
4.2.2.1	AzimuthDelay	8
4.2.2.2	InclinationDelay	8
4.3	Direction Struct Reference	8
4.3.1	Detailed Description	8
4.3.2	Field Documentation	9
4.3.2.1	azimuth	9
4.3.2.2	elevation	9
4.4	menuStruct Struct Reference	9
4.4.1	Detailed Description	9
4.4.2	Field Documentation	10
4.4.2.1	confirmFunction	10
4.4.2.2	increment	10
4.4.2.3	LcdDisplayFunction	10
4.4.2.4	LcdTitleMessage	10

4.4.2.5	maxVal	10
4.4.2.6	menuID	10
4.4.2.7	minVal	10
4.4.2.8	returnToPrevious	10
4.4.2.9	serialDisplayFunction	10
4.4.2.10	serialMessage	10
4.5	SubMenu Struct Reference	10
4.5.1	Field Documentation	10
4.5.1.1	defaultFunction	10
4.5.1.2	inptOptions	11
4.5.1.3	LcdMessage	11
4.5.1.4	numericFunction	11
4.5.1.5	serialMessage	11
4.5.1.6	serialOptions	11
4.5.1.7	subMenus	11
4.6	systemState Struct Reference	11
4.6.1	Detailed Description	11
4.6.2	Field Documentation	11
4.6.2.1	current	11
4.6.2.2	previous	11
4.7	TargetStateData Struct Reference	11
4.7.1	Detailed Description	12
4.7.2	Field Documentation	12
4.7.2.1	bad_dirs	12
4.7.2.2	centre	12
4.7.2.3	good_tracks	12
4.7.2.4	out_of_ir	12
4.8	TrackingData Struct Reference	12
4.8.1	Detailed Description	12
4.8.2	Field Documentation	13
4.8.2.1	azimuth	13
4.8.2.2	elevation	13
4.8.2.3	range	13
5	File Documentation	15
5.1	Yavin4DefenceSystem/Code/CircularBuffers.h File Reference	15
5.1.1	Macro Definition Documentation	16
5.1.1.1	BUFFERLENGTH	16
5.1.1.2	BUFFERS_H	16
5.1.1.3	empty	16

5.1.1.4	full	16
5.1.1.5	incMod	16
5.1.1.6	init	16
5.1.1.7	peek	16
5.1.1.8	pop	16
5.1.1.9	push	16
5.2	Yavin4DefenceSystem/Code/Common.h File Reference	16
5.2.1	Macro Definition Documentation	18
5.2.1.1	ADC_DIAL_READ	18
5.2.1.2	ADC_INT	18
5.2.1.3	ADC_IR_READ	18
5.2.1.4	ADC_TEMP_READ	18
5.2.1.5	BCL_INT	18
5.2.1.6	CCP1_CLEAR	18
5.2.1.7	CCP1_INT	18
5.2.1.8	CCP2_CLEAR	18
5.2.1.9	CCP2_INT	19
5.2.1.10	CLOCK	19
5.2.1.11	COMMON_H	19
5.2.1.12	DIV_1024	19
5.2.1.13	DIV_128	19
5.2.1.14	DIV_16	19
5.2.1.15	DIV_2	19
5.2.1.16	DIV_256	19
5.2.1.17	DIV_32	19
5.2.1.18	DIV_4	19
5.2.1.19	DIV_4096	19
5.2.1.20	DIV_512	19
5.2.1.21	DIV_64	19
5.2.1.22	DIV_65536	19
5.2.1.23	DIV_8	19
5.2.1.24	FOSC_4	19
5.2.1.25	INT0_INT	19
5.2.1.26	INT1_INT	19
5.2.1.27	INT2_INT	19
5.2.1.28	INT_SETUP	19
5.2.1.29	LVD_INT	20
5.2.1.30	MNML	20
5.2.1.31	NEXT_STATE	20
5.2.1.32	NEXT_STATE_PTR	20

5.2.1.33	RB_INT	20
5.2.1.34	RC_INT	20
5.2.1.35	SSP_INT	20
5.2.1.36	SWAP	20
5.2.1.37	TMR0_INT	20
5.2.1.38	TMR1_INT	20
5.2.1.39	TMR2_INT	20
5.2.1.40	TMR3_INT	20
5.2.1.41	TX_INT	20
5.2.2	Enumeration Type Documentation	20
5.2.2.1	escSqnce	20
5.2.2.2	PanTiltSettings	21
5.2.2.3	possible_states	21
5.2.2.4	RangeSettings	21
5.2.2.5	TargetState	21
5.3	Yavin4DefenceSystem/Code/ConfigRegs18f4520.h File Reference	22
5.3.1	Detailed Description	22
5.4	Yavin4DefenceSystem/Code/EEPROM.c File Reference	23
5.5	Yavin4DefenceSystem/Code/HardUltest.c File Reference	23
5.5.1	Macro Definition Documentation	24
5.5.1.1	ADCON0_set	24
5.5.1.2	ADCON1_patch	24
5.5.2	Function Documentation	24
5.5.2.1	ADconfig	25
5.5.2.2	ADgo	25
5.5.2.3	INTconfig	25
5.5.2.4	ISR_high	26
5.5.2.5	ISR_high_vect	27
5.5.2.6	ISR_low	27
5.5.2.7	ISR_low_vect	27
5.5.2.8	LEDselect	28
5.5.2.9	main	28
5.5.2.10	PORTconfig	28
5.5.2.11	setup	29
5.5.3	Variable Documentation	29
5.5.3.1	LEDson	29
5.5.3.2	LEDsonpntr	29
5.5.3.3	LEDtest	29
5.5.3.4	LEDtestpntr	29
5.6	Yavin4DefenceSystem/Code/HardUltest.h File Reference	29

5.7 Yavin4DefenceSystem/Code/Interrupts.c File Reference	29
5.7.1 Function Documentation	30
5.7.1.1 highISR	30
5.7.1.2 highVector	31
5.7.1.3 lowISR	31
5.7.1.4 lowVector	32
5.8 Yavin4DefenceSystem/Code/jEEP.h File Reference	33
5.8.1 Macro Definition Documentation	33
5.8.1.1 EEPADDRESS	33
5.8.2 Function Documentation	33
5.8.2.1 readEep	33
5.8.2.2 sendEep	33
5.9 Yavin4DefenceSystem/Code/LCD.c File Reference	33
5.9.1 Function Documentation	34
5.9.1.1 configLCD	34
5.9.1.2 delay	35
5.9.1.3 lcdBusy	35
5.9.1.4 lcdWrite	36
5.9.1.5 lcdWriteChar	37
5.9.1.6 lcdWriteString	37
5.10 Yavin4DefenceSystem/Code/LCD.h File Reference	38
5.10.1 Macro Definition Documentation	38
5.10.1.1 LCD_H	38
5.10.2 Function Documentation	38
5.10.2.1 configLCD	38
5.10.2.2 lcdWriteChar	39
5.10.2.3 lcdWriteString	39
5.11 Yavin4DefenceSystem/Code/LCD_defs.h File Reference	40
5.11.1 Macro Definition Documentation	42
5.11.1.1 B_BLINKOFF	42
5.11.1.2 B_BLINKON	42
5.11.1.3 BF_BUSY	42
5.11.1.4 BF_READY	42
5.11.1.5 C_CURSOFF	42
5.11.1.6 C_CURSORON	42
5.11.1.7 D_DISP OFF	42
5.11.1.8 D_DISP ON	42
5.11.1.9 DISPCLR	42
5.11.1.10 DISPOPT	42
5.11.1.11 DISPSHIFTCURS	42

5.11.1.12 DL_4BIT	42
5.11.1.13 DL_8BIT	42
5.11.1.14 F_5X10DOT	42
5.11.1.15 F_5X7DOT	42
5.11.1.16 ID_CURSL	42
5.11.1.17 ID_CURSR	42
5.11.1.18 LCD_CLKHIGH	42
5.11.1.19 LCD_CLKLOW	42
5.11.1.20 LCD_D0	42
5.11.1.21 LCD_D0_DIR	42
5.11.1.22 LCD_D1	42
5.11.1.23 LCD_D1_DIR	42
5.11.1.24 LCD_D2	42
5.11.1.25 LCD_D2_DIR	42
5.11.1.26 LCD_D3	42
5.11.1.27 LCD_D3_DIR	42
5.11.1.28 LCD_D4	43
5.11.1.29 LCD_D4_DIR	43
5.11.1.30 LCD_D5	43
5.11.1.31 LCD_D5_DIR	43
5.11.1.32 LCD_D6	43
5.11.1.33 LCD_D6_DIR	43
5.11.1.34 LCD_D7	43
5.11.1.35 LCD_D7_DIR	43
5.11.1.36 LCD_DATA	43
5.11.1.37 LCD_DATA_LINE	43
5.11.1.38 LCD_DATA_LINE_DIR	43
5.11.1.39 LCD_E	43
5.11.1.40 LCD_E_DIR	43
5.11.1.41 LCD_INPUT	43
5.11.1.42 LCD_INS	43
5.11.1.43 LCD_OFF	43
5.11.1.44 LCD_ON	43
5.11.1.45 LCD_OUTPUT	43
5.11.1.46 LCD_PIN_INPUT	43
5.11.1.47 LCD_PIN_OUTPUT	43
5.11.1.48 LCD_READ	43
5.11.1.49 LCD_RS	43
5.11.1.50 LCD_RS_DIR	43
5.11.1.51 LCD_RW	43

5.11.1.52 LCD_RW_DIR	43
5.11.1.53 LCD_WRITE	43
5.11.1.54 LINE1	43
5.11.1.55 LINE2	43
5.11.1.56 LINEEND	44
5.11.1.57 LINESTART	44
5.11.1.58 N_1LINE	44
5.11.1.59 N_2LINE	44
5.11.1.60 RL_LFTSHFT	44
5.11.1.61 RL_RGTSHFT	44
5.11.1.62 RTNHOME	44
5.11.1.63 SC_CURSMOVE	44
5.11.1.64 SC_DISPMOVE	44
5.11.1.65 SETCGRAMADD	44
5.11.1.66 SETDDRAMADD	44
5.11.1.67 SETDISPFXN	44
5.11.1.68 SETENTRYMODE	44
5.11.1.69 SH_DISPNSHIFT	44
5.11.1.70 SH_DISPSHIFT	44
5.12 Yavin4DefenceSystem/Code/Menu_String.h File Reference	44
5.12.1 Macro Definition Documentation	46
5.12.1.1 STRINGS_H	46
5.12.2 Variable Documentation	46
5.12.2.1 and	46
5.12.2.2 angleStr	46
5.12.2.3 azMenu	46
5.12.2.4 azMenuLcd	46
5.12.2.5 azOption1	46
5.12.2.6 azOption2	46
5.12.2.7 azOption3	46
5.12.2.8 azOption4	46
5.12.2.9 calibrateAngle1	46
5.12.2.10 CHOOSE	46
5.12.2.11 currentMinAngleStr	46
5.12.2.12 elMenu	46
5.12.2.13 elMenuLcd	46
5.12.2.14 elOption1	46
5.12.2.15 elOption2	46
5.12.2.16 elOption3	46
5.12.2.17 elOption4	46

5.12.2.18 errNumOutOfRange	46
5.12.2.19 gotoAngle2	46
5.12.2.20 gotoAzAngle	46
5.12.2.21 gotoAzAngleLCD	46
5.12.2.22 gotoElAngle	47
5.12.2.23 gotoELAngleLCD	47
5.12.2.24 goUp	47
5.12.2.25 goUp2	47
5.12.2.26 maxAz1	47
5.12.2.27 maxAz3	47
5.12.2.28 maxAzSetStr	47
5.12.2.29 maxAzStr	47
5.12.2.30 maxEl1	47
5.12.2.31 maxEl3	47
5.12.2.32 maxElSetStr	47
5.12.2.33 maxElStr	47
5.12.2.34 maxRngSerialStr	47
5.12.2.35 maxRngSetStr	47
5.12.2.36 maxRngStr	47
5.12.2.37 menuPrefix3	47
5.12.2.38 menuPrefix4	47
5.12.2.39 menuPrefix5	47
5.12.2.40 menuPrefix8	47
5.12.2.41 minAz1	47
5.12.2.42 minAz3	47
5.12.2.43 minAzSetStr	47
5.12.2.44 minAzStr	47
5.12.2.45 minEl1	47
5.12.2.46 minEl3	47
5.12.2.47 minElSetStr	47
5.12.2.48 minElStr	47
5.12.2.49 minRngSerialStr	47
5.12.2.50 minRngSetStr	48
5.12.2.51 minRngStr	48
5.12.2.52 newLine	48
5.12.2.53 rngMenu	48
5.12.2.54 rngMenuLcd	48
5.12.2.55 rngOption1	48
5.12.2.56 rngOption2	48
5.12.2.57 rngOption3	48

5.12.2.58 rngOption4	48
5.12.2.59 rngOption5	48
5.12.2.60 rngOption6	48
5.12.2.61 rngOption7	48
5.12.2.62 showTemp1	48
5.12.2.63 showTemp2	48
5.12.2.64 showTempLCD	48
5.12.2.65 showTempLCDTitle	48
5.12.2.66 topOption1	48
5.12.2.67 topOption2	48
5.12.2.68 topOption3	48
5.12.2.69 topOption4	48
5.12.2.70 topOption5	48
5.12.2.71 topOptionCalTemp	48
5.12.2.72 topOptionFactory	48
5.12.2.73 topOptionLocal	48
5.12.2.74 topOptionRemote	48
5.12.2.75 topOptionRemoteLCD	48
5.12.2.76 welcome	48
5.12.2.77 welcomeLcd	48
5.13 Yavin4DefenceSystem/Code/MenuDefs.h File Reference	49
5.13.1 Macro Definition Documentation	52
5.13.1.1 CLEAR_LINE_STRING	52
5.13.1.2 CLEAR_SCREEN_STRING	52
5.13.1.3 ERR_NO_NUMBER	52
5.13.1.4 ERR_NOT_NUMERIC	52
5.13.1.5 ERR_NUM_OUT_OF_RANGE	52
5.13.1.6 ESC_PRESSED	52
5.13.1.7 FACTORY_SWITCH	52
5.13.1.8 height	52
5.13.1.9 MAX_ANGLE_INFIMUM	52
5.13.1.10 MAX_ANGLE_SUPREMUM	52
5.13.1.11 MAX_RANGE_INFIMUM	52
5.13.1.12 MAX_RANGE_SUPREMUM	52
5.13.1.13 MENU_ISR	52
5.13.1.14 MIN_ANGLE_INFIMUM	52
5.13.1.15 MIN_ANGLE_SUPREMUM	52
5.13.1.16 MIN_RANGE_INFIMUM	52
5.13.1.17 MIN_RANGE_SUPREMUM	52
5.13.1.18 MINUS_CHAR	52

5.13.1.19 SAMPLE_PER_AVG_MAX	52
5.13.1.20 SAMPLE_PER_AVG_MIN	53
5.13.1.21 SERIAL_CURSOR_OFF	53
5.13.1.22 SERIAL_CURSOR_ON	53
5.13.1.23 width	53
5.13.2 Typedef Documentation	53
5.13.2.1 numericInputFunction	53
5.13.2.2 voidFunction	53
5.13.3 Enumeration Type Documentation	53
5.13.3.1 menuState	53
5.13.3.2 userState	53
5.13.4 Variable Documentation	54
5.13.4.1 and	54
5.13.4.2 angleStr	54
5.13.4.3 autoLcdTitle	54
5.13.4.4 autoSearching	54
5.13.4.5 autoSerialMessage	54
5.13.4.6 azMenu	54
5.13.4.7 azMenuLcd	54
5.13.4.8 azOption1	54
5.13.4.9 azOption2	54
5.13.4.10 azOption3	54
5.13.4.11 azOption4	54
5.13.4.12 AzStr	54
5.13.4.13 calibrateAngle1	54
5.13.4.14 calRangeConfirm	54
5.13.4.15 calRangeStr	54
5.13.4.16 calRangeTitle	54
5.13.4.17 CHOOSE	54
5.13.4.18 CHOOSE2	54
5.13.4.19 currentMinAngleStr	54
5.13.4.20 currentValueStr	54
5.13.4.21 elMenu	54
5.13.4.22 elMenuLcd	54
5.13.4.23 elOption1	54
5.13.4.24 elOption2	54
5.13.4.25 elOption3	54
5.13.4.26 elOption4	54
5.13.4.27 ElStr	54
5.13.4.28 errStr	55

5.13.4.29 goto_str	55
5.13.4.30 gotoAngle2	55
5.13.4.31 gotoAzAngle	55
5.13.4.32 gotoAzAngleLCD	55
5.13.4.33 gotoElAngle	55
5.13.4.34 gotoELAngleLCD	55
5.13.4.35 goUp	55
5.13.4.36 goUp2	55
5.13.4.37 hz	55
5.13.4.38 inputNumRangeStr	55
5.13.4.39 maxAz1	55
5.13.4.40 maxAz3	55
5.13.4.41 maxAzSetStr	55
5.13.4.42 maxAzStr	55
5.13.4.43 maxEl1	55
5.13.4.44 maxEl3	55
5.13.4.45 maxElSetStr	55
5.13.4.46 maxElStr	55
5.13.4.47 maxRngSerialStr	55
5.13.4.48 maxRngSetStr	55
5.13.4.49 maxRngStr	55
5.13.4.50 menuPrefix3	55
5.13.4.51 menuPrefix4	55
5.13.4.52 menuPrefix5	55
5.13.4.53 menuPrefix8	55
5.13.4.54 menuStruct	55
5.13.4.55 minAz1	55
5.13.4.56 minAz3	56
5.13.4.57 minAzSetStr	56
5.13.4.58 minAzStr	56
5.13.4.59 minEl1	56
5.13.4.60 minEl3	56
5.13.4.61 minElSetStr	56
5.13.4.62 minElStr	56
5.13.4.63 minRngSerialStr	56
5.13.4.64 minRngSetStr	56
5.13.4.65 minRngStr	56
5.13.4.66 mmStr	56
5.13.4.67 newLine	56
5.13.4.68 numPerSample	56

5.13.4.69 numSamplesStr	56
5.13.4.70 numSampleTitle	56
5.13.4.71 Options	56
5.13.4.72 range_str	56
5.13.4.73 rawRangeStr	56
5.13.4.74 rawRangeTitle	56
5.13.4.75 rngMenu	56
5.13.4.76 rngMenuLcd	56
5.13.4.77 rngOption1	56
5.13.4.78 rngOption2	56
5.13.4.79 rngOption3	56
5.13.4.80 rngOption4	56
5.13.4.81 rngOption5	56
5.13.4.82 rngOption7	56
5.13.4.83 rngOption8	56
5.13.4.84 RngStr	57
5.13.4.85 sampleRate	57
5.13.4.86 set	57
5.13.4.87 showTemp1	57
5.13.4.88 showTemp2	57
5.13.4.89 showTempLCD	57
5.13.4.90 showTempLCDTitle	57
5.13.4.91 tab	57
5.13.4.92 title	57
5.13.4.93 topOption1	57
5.13.4.94 topOption2	57
5.13.4.95 topOption3	57
5.13.4.96 topOption4	57
5.13.4.97 topOption5	57
5.13.4.98 topOptionCalTemp	57
5.13.4.99 topOptionForFactory	57
5.13.4.100topOptionLocal	57
5.13.4.101topOptionRemote	57
5.13.4.102topOptionRemoteLCD	57
5.13.4.103usSampleRateStr	57
5.13.4.104usSampleRateTitle	57
5.13.4.105welcomeLcd	57
5.14 Yavin4DefenceSystem/Code/Menusystem.c File Reference	57
5.14.1 Function Documentation	60
5.14.1.1 autodisp	60

5.14.1.2 checkForSerialInput	60
5.14.1.3 clearScreen	60
5.14.1.4 configureTimer0	61
5.14.1.5 dispAzOptions	61
5.14.1.6 dispEIOptions	62
5.14.1.7 displayMenuSerial	62
5.14.1.8 dispLCDAzMenu	63
5.14.1.9 dispLCDEIMenu	63
5.14.1.10 dispLCDNum	63
5.14.1.11 dispLCDRngMenu	64
5.14.1.12 dispLCDTopMenu	64
5.14.1.13 dispRawRange	64
5.14.1.14 dispRngOptions	64
5.14.1.15 dispSearching	65
5.14.1.16 dispSerialMessage	65
5.14.1.17 dispSetValueMessage	66
5.14.1.18 dispTempSerialMessage	67
5.14.1.19 dispTopOptions	68
5.14.1.20 dispTrack	68
5.14.1.21 errOutOfRange	69
5.14.1.22 exitFromTracking	70
5.14.1.23 filler	70
5.14.1.24 getLocalInputMenu	71
5.14.1.25 getSerialNumericInput	71
5.14.1.26 initialiseMenu	72
5.14.1.27 intToAscii	73
5.14.1.28 navigateAzimuthMenu	74
5.14.1.29 navigateElevationMenu	75
5.14.1.30 navigateRangeMenu	75
5.14.1.31 navigateTopMenu	76
5.14.1.32 noFunction	76
5.14.1.33 noFunctionNumeric	76
5.14.1.34 parseNumeric	76
5.14.1.35 returnToAzMenu	77
5.14.1.36 returnToEIMenu	77
5.14.1.37 returnToRngMenu	78
5.14.1.38 returnToTopMenu	78
5.14.1.39 sendNewLine	78
5.14.1.40 sendROM	79
5.14.1.41 serviceMenu	81

5.14.1.42 setCalibrateRange	81
5.14.1.43 setMenu	82
5.14.1.44 setValue	83
5.14.2 Variable Documentation	83
5.14.2.1 AzGoto	83
5.14.2.2 AzMax	84
5.14.2.3 AzMenu	84
5.14.2.4 AzMin	84
5.14.2.5 calibrateRangeMenu	84
5.14.2.6 ElGoto	84
5.14.2.7 ElMax	84
5.14.2.8 ElMenu	84
5.14.2.9 ElMin	84
5.14.2.10 m_currentMenu	84
5.14.2.11 m_trackingState	84
5.14.2.12 m_userMode	84
5.14.2.13 NumSamples	84
5.14.2.14 RangeMenu	84
5.14.2.15 RawRange	84
5.14.2.16 RngMax	84
5.14.2.17 RngMin	84
5.14.2.18 ShowTemp	84
5.14.2.19 topMenu	85
5.14.2.20 Tracking	85
5.14.2.21 UsSampleRate	85
5.15 Yavin4DefenceSystem/Code/Menusystem.h File Reference	85
5.15.1 Function Documentation	87
5.15.1.1 checkForSerialInput	87
5.15.1.2 dispRawRange	87
5.15.1.3 dispSearching	89
5.15.1.4 dispTrack	89
5.15.1.5 initialiseMenu	90
5.15.1.6 serviceMenu	91
5.15.2 Variable Documentation	92
5.15.2.1 and	92
5.15.2.2 angleStr	92
5.15.2.3 azMenu	93
5.15.2.4 azMenuLcd	93
5.15.2.5 azOption1	93
5.15.2.6 azOption2	93

5.15.2.7 azOption3	93
5.15.2.8 azOption4	93
5.15.2.9 calibrateAngle1	93
5.15.2.10 CHOOSE	93
5.15.2.11 currentMinAngleStr	93
5.15.2.12 elMenu	93
5.15.2.13 elMenuLcd	93
5.15.2.14 elOption1	93
5.15.2.15 elOption2	93
5.15.2.16 elOption3	93
5.15.2.17 elOption4	93
5.15.2.18 gotoAngle2	93
5.15.2.19 gotoAzAngle	93
5.15.2.20 gotoAzAngleLCD	93
5.15.2.21 gotoElAngle	93
5.15.2.22 gotoELAngleLCD	93
5.15.2.23 goUp	93
5.15.2.24 goUp2	93
5.15.2.25 hz	93
5.15.2.26 inputNumRangeStr	93
5.15.2.27 maxAz1	93
5.15.2.28 maxAz3	93
5.15.2.29 maxAzSetStr	93
5.15.2.30 maxAzStr	93
5.15.2.31 maxEl1	94
5.15.2.32 maxEl3	94
5.15.2.33 maxElSetStr	94
5.15.2.34 maxElStr	94
5.15.2.35 maxRngSerialStr	94
5.15.2.36 maxRngSetStr	94
5.15.2.37 maxRngStr	94
5.15.2.38 menuPrefix3	94
5.15.2.39 menuPrefix4	94
5.15.2.40 menuPrefix5	94
5.15.2.41 menuPrefix8	94
5.15.2.42 minAz1	94
5.15.2.43 minAz3	94
5.15.2.44 minAzSetStr	94
5.15.2.45 minAzStr	94
5.15.2.46 minEl1	94

5.15.2.47 minEl3	94
5.15.2.48 minElSetStr	94
5.15.2.49 minElStr	94
5.15.2.50 minRngSerialStr	94
5.15.2.51 minRngSetStr	94
5.15.2.52 minRngStr	94
5.15.2.53 newLine	94
5.15.2.54 rngMenu	94
5.15.2.55 rngMenuLcd	94
5.15.2.56 rngOption1	94
5.15.2.57 rngOption2	94
5.15.2.58 rngOption3	94
5.15.2.59 rngOption4	95
5.15.2.60 rngOption5	95
5.15.2.61 rngOption6	95
5.15.2.62 rngOption7	95
5.15.2.63 showTemp1	95
5.15.2.64 showTemp2	95
5.15.2.65 showTempLCD	95
5.15.2.66 showTempLCDTitle	95
5.15.2.67 tab	95
5.15.2.68 title	95
5.15.2.69 topOption1	95
5.15.2.70 topOption2	95
5.15.2.71 topOption3	95
5.15.2.72 topOption4	95
5.15.2.73 topOption5	95
5.15.2.74 topOptionCalTemp	95
5.15.2.75 topOptionForFactory	95
5.15.2.76 topOptionLocal	95
5.15.2.77 topOptionRemote	95
5.15.2.78 topOptionRemoteLCD	95
5.15.2.79 welcomeLcd	95
5.16 Yavin4DefenceSystem/Code/Menusystem2.c File Reference	95
5.16.1 Macro Definition Documentation	97
5.16.1.1 MAX_LCD_MSG_LEN	97
5.16.1.2 MAX_NUM_OPTIONS	97
5.16.1.3 MAX_SER_MSG_LEN	97
5.16.2 Typedef Documentation	97
5.16.2.1 SubMenu	97

5.16.3 Enumeration Type Documentation	97
5.16.3.1 MenuLevel	97
5.16.4 Function Documentation	97
5.16.4.1 initialiseMenu	97
5.16.4.2 menuISR	98
5.16.4.3 parseInput	98
5.16.4.4 parseNumeric	98
5.16.4.5 serviceMenu	99
5.16.4.6 stateEntry	100
5.16.5 Variable Documentation	101
5.16.5.1 AutoTrack	101
5.16.5.2 Az	101
5.16.5.3 Elev	101
5.16.5.4 level	101
5.16.5.5 ManTrack	101
5.16.5.6 Max	101
5.16.5.7 menus	101
5.16.5.8 Min	101
5.16.5.9 Root	101
5.16.5.10 test	101
5.17 Yavin4DefenceSystem/Code/newfile.h File Reference	101
5.18 Yavin4DefenceSystem/Code/newmain.c File Reference	101
5.18.1 Function Documentation	101
5.18.1.1 initialization	102
5.18.1.2 main	102
5.19 Yavin4DefenceSystem/Code/p18f4520.h File Reference	103
5.19.1 Macro Definition Documentation	127
5.19.1.1 ACCESS	127
5.19.1.2 BANKED	127
5.19.1.3 ClrWdt	127
5.19.1.4 INTSAVELOCS	127
5.19.1.5 Nop	127
5.19.1.6 Reset	127
5.19.1.7 Rlcf	127
5.19.1.8 Rlncf	127
5.19.1.9 Rrcf	127
5.19.1.10 Rrnccf	127
5.19.1.11 Sleep	127
5.19.1.12 Swapf	127
5.19.2 Variable Documentation	127

5.19.2.1	<u>__pad0__</u>	127
5.19.2.2	<u>__pad1__</u>	127
5.19.2.3	A	127
5.19.2.4	ABDEN	127
5.19.2.5	ABDOVF	127
5.19.2.6	ACKDT	127
5.19.2.7	ACKEN	127
5.19.2.8	ACKSTAT	128
5.19.2.9	ACQT	128
5.19.2.10	ACQT0	128
5.19.2.11	ACQT1	128
5.19.2.12	ACQT2	128
5.19.2.13	ADCON0	128
5.19.2.14	ADCON0bits	128
5.19.2.15	ADCON1	128
5.19.2.16	ADCON1bits	128
5.19.2.17	ADCON2	128
5.19.2.18	ADCON2bits	128
5.19.2.19	ADCS	128
5.19.2.20	ADCS0	128
5.19.2.21	ADCS1	128
5.19.2.22	ADCS2	128
5.19.2.23	ADDEN	128
5.19.2.24	ADEN	128
5.19.2.25	ADFM	128
5.19.2.26	ADIE	128
5.19.2.27	ADIF	128
5.19.2.28	ADIP	128
5.19.2.29	ADMSK1	128
5.19.2.30	ADMSK2	128
5.19.2.31	ADMSK3	128
5.19.2.32	ADMSK4	128
5.19.2.33	ADMSK5	128
5.19.2.34	ADON	128
5.19.2.35	ADRES	128
5.19.2.36	ADRESH	129
5.19.2.37	ADRESL	129
5.19.2.38	AN0	129
5.19.2.39	AN1	129
5.19.2.40	AN10	129

5.19.2.41 AN11	129
5.19.2.42 AN12	129
5.19.2.43 AN2	129
5.19.2.44 AN3	129
5.19.2.45 AN4	129
5.19.2.46 AN5	129
5.19.2.47 AN6	129
5.19.2.48 AN7	129
5.19.2.49 AN8	129
5.19.2.50 AN9	129
5.19.2.51 BAUDCON	129
5.19.2.52 BAUDCONbits	129
5.19.2.53 BAUDCTL	129
5.19.2.54 BAUDCTLbits	129
5.19.2.55 BCLIE	129
5.19.2.56 BCLIF	129
5.19.2.57 BCLIP	129
5.19.2.58 BF	129
5.19.2.59 BGST	129
5.19.2.60 BOR	129
5.19.2.61 BRG16	129
5.19.2.62 BRGH	129
5.19.2.63 BSR	129
5.19.2.64 C	130
5.19.2.65 C1INV	130
5.19.2.66 C1OUT	130
5.19.2.67 C2INV	130
5.19.2.68 C2OUT	130
5.19.2.69 CCP1	130
5.19.2.70 CCP1CON	130
5.19.2.71 CCP1CONbits	130
5.19.2.72 CCP1IE	130
5.19.2.73 CCP1IF	130
5.19.2.74 CCP1IP	130
5.19.2.75 CCP1M	130
5.19.2.76 CCP1M0	130
5.19.2.77 CCP1M1	130
5.19.2.78 CCP1M2	130
5.19.2.79 CCP1M3	130
5.19.2.80 CCP1X	130

5.19.2.81 CCP1Y	130
5.19.2.82 CCP2	130
5.19.2.83 CCP2CON	130
5.19.2.84 CCP2CONbits	130
5.19.2.85 CCP2IE	130
5.19.2.86 CCP2IF	130
5.19.2.87 CCP2IP	130
5.19.2.88 CCP2M	130
5.19.2.89 CCP2M0	130
5.19.2.90 CCP2M1	130
5.19.2.91 CCP2M2	130
5.19.2.92 CCP2M3	131
5.19.2.93 CCP2X	131
5.19.2.94 CCP2Y	131
5.19.2.95 CCPR1	131
5.19.2.96 CCPR1H	131
5.19.2.97 CCPR1L	131
5.19.2.98 CCPR2	131
5.19.2.99 CCPR2H	131
5.19.2.100CCPR2L	131
5.19.2.101CFG8	131
5.19.2.102CHS	131
5.19.2.103CHS0	131
5.19.2.104CHS1	131
5.19.2.105CHS2	131
5.19.2.106CHS3	131
5.19.2.107CIS	131
5.19.2.108CK	131
5.19.2.109CKE	131
5.19.2.110CKP	131
5.19.2.111CLKI	131
5.19.2.112CLKO	131
5.19.2.113CM	131
5.19.2.114CM0	131
5.19.2.115CM1	131
5.19.2.116CM2	131
5.19.2.117CMCON	131
5.19.2.118CMCONbits	131
5.19.2.119CMIE	131
5.19.2.120CMIF	132

5.19.2.121CMIP	132
5.19.2.122CREN	132
5.19.2.123CS	132
5.19.2.124CSRC	132
5.19.2.125CVR	132
5.19.2.126CVR0	132
5.19.2.127CVR1	132
5.19.2.128CVR2	132
5.19.2.129CVR3	132
5.19.2.130CVRCON	132
5.19.2.131CVRCONbits	132
5.19.2.132CVREF	132
5.19.2.133CVREN	132
5.19.2.134CVROE	132
5.19.2.135CVRR	132
5.19.2.136CVRSS	132
5.19.2.137D	132
5.19.2.138D_A	132
5.19.2.139D_NOT_A	132
5.19.2.140DC	132
5.19.2.141DC1B	132
5.19.2.142DC1B0	132
5.19.2.143DC1B1	132
5.19.2.144DC2B	132
5.19.2.145DC2B0	132
5.19.2.146DC2B1	132
5.19.2.147DDRA	132
5.19.2.148DDRAbits	133
5.19.2.149DDRB	133
5.19.2.150DDRBBits	133
5.19.2.151DDRC	133
5.19.2.152DDRCBbits	133
5.19.2.153DDRD	133
5.19.2.154DDRDBits	133
5.19.2.155DDRE	133
5.19.2.156DDREbits	133
5.19.2.157DONE	133
5.19.2.158DT	133
5.19.2.159ECCP1AS	133
5.19.2.160ECCP1ASbits	133

5.19.2.161ECCP1DEL	133
5.19.2.162ECCP1DELbits	133
5.19.2.163ECCPAS	133
5.19.2.164ECCPAS0	133
5.19.2.165ECCPAS1	133
5.19.2.166ECCPAS2	133
5.19.2.167ECCPASbits	133
5.19.2.168ECCPASE	133
5.19.2.169EEADDR	133
5.19.2.170EECON1	133
5.19.2.171EECON1bits	133
5.19.2.172EECON2	133
5.19.2.173EEDATA	133
5.19.2.174EEIE	133
5.19.2.175EEIF	133
5.19.2.176EEIP	134
5.19.2.177EEPGD	134
5.19.2.178FERR	134
5.19.2.179FLT0	134
5.19.2.180FLTS	134
5.19.2.181FREE	134
5.19.2.182FSR0	134
5.19.2.183FSR0H	134
5.19.2.184FSR0L	134
5.19.2.185FSR1	134
5.19.2.186FSR1H	134
5.19.2.187FSR1L	134
5.19.2.188FSR2	134
5.19.2.189FSR2H	134
5.19.2.190FSR2L	134
5.19.2.191GCEN	134
5.19.2.192GIE	134
5.19.2.193GIE_GIEH	134
5.19.2.194GIEH	134
5.19.2.195GIEL	134
5.19.2.196GO	134
5.19.2.197GO_DONE	134
5.19.2.198GO_NOT_DONE	134
5.19.2.199HLVDCON	134
5.19.2.200HLVDCONbits	134

5.19.2.201HLVDEN	134
5.19.2.202HLVDIE	134
5.19.2.203HLVDIF	134
5.19.2.204HLVDIN	135
5.19.2.205HLVDIP	135
5.19.2.206HLVDL	135
5.19.2.207HLVDL0	135
5.19.2.208HLVDL1	135
5.19.2.209HLVDL2	135
5.19.2.210HLVDL3	135
5.19.2.211BF	135
5.19.2.212BOV	135
5.19.2.213DLEN	135
5.19.2.214NDF0	135
5.19.2.215NDF1	135
5.19.2.216NDF2	135
5.19.2.217NT0	135
5.19.2.218NT0E	135
5.19.2.219NT0F	135
5.19.2.220NT0IE	135
5.19.2.221INT0IF	135
5.19.2.222NT1	135
5.19.2.223NT1E	135
5.19.2.224INT1F	135
5.19.2.225NT1IE	135
5.19.2.226NT1IF	135
5.19.2.227NT1IP	135
5.19.2.228NT1P	135
5.19.2.229NT2	135
5.19.2.230NT2E	135
5.19.2.231INT2F	135
5.19.2.232NT2IE	136
5.19.2.233NT2IF	136
5.19.2.234INT2IP	136
5.19.2.235NT2P	136
5.19.2.236NTCON	136
5.19.2.237NTCON2	136
5.19.2.238NTCON2bits	136
5.19.2.239NTCON3	136
5.19.2.240NTCON3bits	136

5.19.2.241INTCONbits	136
5.19.2.242NTEDG0	136
5.19.2.243NTEDG1	136
5.19.2.244NTEDG2	136
5.19.2.245NTSRC	136
5.19.2.246OFS	136
5.19.2.247PEN	136
5.19.2.248PR1	136
5.19.2.249PR1bits	136
5.19.2.250PR2	136
5.19.2.251PR2bits	136
5.19.2.252RCF	136
5.19.2.253RCF0	136
5.19.2.254RCF1	136
5.19.2.255RCF2	136
5.19.2.256RVST	136
5.19.2.257VRST	136
5.19.2.258KBI0	136
5.19.2.259KBI1	136
5.19.2.260KBI2	137
5.19.2.261KBI3	137
5.19.2.262LATA	137
5.19.2.263LATA0	137
5.19.2.264LATA1	137
5.19.2.265LATA2	137
5.19.2.266LATA3	137
5.19.2.267LATA4	137
5.19.2.268LATA5	137
5.19.2.269LATA6	137
5.19.2.270LATA7	137
5.19.2.271LATAbits	137
5.19.2.272LATB	137
5.19.2.273LATB0	137
5.19.2.274LATB1	137
5.19.2.275LATB2	137
5.19.2.276LATB3	137
5.19.2.277LATB4	137
5.19.2.278LATB5	137
5.19.2.279LATB6	137
5.19.2.280LATB7	137

5.19.2.281LATBbits	137
5.19.2.282LATC	137
5.19.2.283LATC0	137
5.19.2.284LATC1	137
5.19.2.285LATC2	137
5.19.2.286LATC3	137
5.19.2.287LATC4	137
5.19.2.288LATC5	138
5.19.2.289LATC6	138
5.19.2.290LATC7	138
5.19.2.291LATCbits	138
5.19.2.292LATD	138
5.19.2.293LATD0	138
5.19.2.294LATD1	138
5.19.2.295LATD2	138
5.19.2.296LATD3	138
5.19.2.297LATD4	138
5.19.2.298LATD5	138
5.19.2.299LATD6	138
5.19.2.300LATD7	138
5.19.2.301LATDbits	138
5.19.2.302LATE	138
5.19.2.303LATE0	138
5.19.2.304LATE1	138
5.19.2.305LATE2	138
5.19.2.306LATEbits	138
5.19.2.307LVDCON	138
5.19.2.308LVDCONbits	138
5.19.2.309LVDEN	138
5.19.2.310LVDIE	138
5.19.2.311LVDIF	138
5.19.2.312LVDIN	138
5.19.2.313LVDIP	138
5.19.2.314LVDL0	138
5.19.2.315LVDL1	138
5.19.2.316LVDL2	139
5.19.2.317LVDL3	139
5.19.2.318LVV0	139
5.19.2.319LVV1	139
5.19.2.320LVV2	139

5.19.2.321LVV3	139
5.19.2.322MCLR	139
5.19.2.323N	139
5.19.2.324NOT_A	139
5.19.2.325NOT_ADDRESS	139
5.19.2.326NOT_BOR	139
5.19.2.327NOT_CS	139
5.19.2.328NOT_DONE	139
5.19.2.329NOT_MCLR	139
5.19.2.330NOT_PD	139
5.19.2.331NOT_POR	139
5.19.2.332NOT_RBPU	139
5.19.2.333NOT_RD	139
5.19.2.334NOT_RI	139
5.19.2.335NOT_SS	139
5.19.2.336NOT_T1SYNC	139
5.19.2.337NOT_T3SYNC	139
5.19.2.338NOT_TO	139
5.19.2.339NOT_W	139
5.19.2.340NOT_WR	139
5.19.2.341NOT_WRITE	139
5.19.2.342OBF	139
5.19.2.343OERR	139
5.19.2.344OSC1	140
5.19.2.345OSC2	140
5.19.2.346OSCCON	140
5.19.2.347OSCCONbits	140
5.19.2.348OSCFIE	140
5.19.2.349OSCFIF	140
5.19.2.350OSCFIP	140
5.19.2.351OSCTUNE	140
5.19.2.352OSCTUNEbits	140
5.19.2.353OSTS	140
5.19.2.354OV	140
5.19.2.355P	140
5.19.2.356P1A	140
5.19.2.357P1B	140
5.19.2.358P1C	140
5.19.2.359P1D	140
5.19.2.360P1M	140

5.19.2.361P1M0	140
5.19.2.362P1M1	140
5.19.2.363PC	140
5.19.2.364PCFG	140
5.19.2.365PCFG0	140
5.19.2.366PCFG1	140
5.19.2.367PCFG2	140
5.19.2.368PCFG3	140
5.19.2.369PCL	140
5.19.2.370PCLATH	140
5.19.2.371PCLATU	140
5.19.2.372PD	141
5.19.2.373PDC	141
5.19.2.374PDC0	141
5.19.2.375PDC1	141
5.19.2.376PDC2	141
5.19.2.377PDC3	141
5.19.2.378PDC4	141
5.19.2.379PDC5	141
5.19.2.380PDC6	141
5.19.2.381PEIE	141
5.19.2.382PEIE_GIEL	141
5.19.2.383PEN	141
5.19.2.384PGC	141
5.19.2.385PGD	141
5.19.2.386PGM	141
5.19.2.387PIE1	141
5.19.2.388PIE1bits	141
5.19.2.389PIE2	141
5.19.2.390PIE2bits	141
5.19.2.391PIR1	141
5.19.2.392PIR1bits	141
5.19.2.393PIR2	141
5.19.2.394PIR2bits	141
5.19.2.395PLLEN	141
5.19.2.396PLUSW0	141
5.19.2.397PLUSW1	141
5.19.2.398PLUSW2	141
5.19.2.399POR	141
5.19.2.400PORTA	142

5.19.2.401PORTAbits	142
5.19.2.402PORTB	142
5.19.2.403PORTBbits	142
5.19.2.404PORTC	142
5.19.2.405PORTCbits	142
5.19.2.406PORTD	142
5.19.2.407PORTDbits	142
5.19.2.408PORTE	142
5.19.2.409PORTEbits	142
5.19.2.410POSTDEC0	142
5.19.2.411POSTDEC1	142
5.19.2.412POSTDEC2	142
5.19.2.413POSTINC0	142
5.19.2.414POSTINC1	142
5.19.2.415POSTINC2	142
5.19.2.416PR2	142
5.19.2.417PREINC0	142
5.19.2.418PREINC1	142
5.19.2.419PREINC2	142
5.19.2.420PROD	142
5.19.2.421PRODH	142
5.19.2.422PRODL	142
5.19.2.423PRSEN	142
5.19.2.424PSA	142
5.19.2.425PSP0	142
5.19.2.426PSP1	142
5.19.2.427PSP2	142
5.19.2.428PSP3	143
5.19.2.429PSP4	143
5.19.2.430PSP5	143
5.19.2.431PSP6	143
5.19.2.432PSP7	143
5.19.2.433PSPIE	143
5.19.2.434PSPIF	143
5.19.2.435PSPIP	143
5.19.2.436PSPMODE	143
5.19.2.437PSSAC	143
5.19.2.438PSSAC0	143
5.19.2.439PSSAC1	143
5.19.2.440PSSBD	143

5.19.2.441PSSBD0	143
5.19.2.442PSSBD1	143
5.19.2.443PWM1CON	143
5.19.2.444PWM1CONbits	143
5.19.2.445R	143
5.19.2.446R_NOT_W	143
5.19.2.447R_W	143
5.19.2.448RA0	143
5.19.2.449RA1	143
5.19.2.450RA2	143
5.19.2.451RA3	143
5.19.2.452RA4	143
5.19.2.453RA5	143
5.19.2.454RA6	143
5.19.2.455RA7	143
5.19.2.456RB0	144
5.19.2.457RB1	144
5.19.2.458RB2	144
5.19.2.459RB3	144
5.19.2.460RB4	144
5.19.2.461RB5	144
5.19.2.462RB6	144
5.19.2.463RB7	144
5.19.2.464RBIE	144
5.19.2.465RBIF	144
5.19.2.466RBIP	144
5.19.2.467RBPU	144
5.19.2.468RC0	144
5.19.2.469RC1	144
5.19.2.470RC2	144
5.19.2.471RC3	144
5.19.2.472RC4	144
5.19.2.473RC5	144
5.19.2.474RC6	144
5.19.2.475RC7	144
5.19.2.476RCEN	144
5.19.2.477RCIDL	144
5.19.2.478RCIE	144
5.19.2.479RCIF	144
5.19.2.480RCIP	144

5.19.2.481RCMT	144
5.19.2.482RCON	144
5.19.2.483RCONbits	144
5.19.2.484RCREG	145
5.19.2.485RCSTA	145
5.19.2.486RCSTAbits	145
5.19.2.487RD	145
5.19.2.488RD0	145
5.19.2.489RD1	145
5.19.2.490RD16	145
5.19.2.491RD2	145
5.19.2.492RD3	145
5.19.2.493RD4	145
5.19.2.494RD5	145
5.19.2.495RD6	145
5.19.2.496RD7	145
5.19.2.497RE0	145
5.19.2.498RE1	145
5.19.2.499RE2	145
5.19.2.500RE3	145
5.19.2.501RI	145
5.19.2.502RSEN	145
5.19.2.503RX	145
5.19.2.504RX9	145
5.19.2.505RX9D	145
5.19.2.506RXDTP	145
5.19.2.507S	145
5.19.2.508SBOREN	145
5.19.2.509SCK	145
5.19.2.510SCKP	145
5.19.2.511SCL	145
5.19.2.512SCS	146
5.19.2.513SCS0	146
5.19.2.514SCS1	146
5.19.2.515SDA	146
5.19.2.516SDI	146
5.19.2.517SDO	146
5.19.2.518SEN	146
5.19.2.519SEND B	146
5.19.2.520SMP	146

5.19.2.521SP0	146
5.19.2.522SP1	146
5.19.2.523SP2	146
5.19.2.524SP3	146
5.19.2.525SP4	146
5.19.2.526SPBRG	146
5.19.2.527SPBRGH	146
5.19.2.528SPEN	146
5.19.2.529SREN	146
5.19.2.530SS	146
5.19.2.531SSPADD	146
5.19.2.532SSPBUF	146
5.19.2.533SSPCON1	146
5.19.2.534SSPCON1bits	146
5.19.2.535SSPCON2	146
5.19.2.536SSPCON2bits	146
5.19.2.537SSPEN	146
5.19.2.538SSPIE	146
5.19.2.539SSPIF	146
5.19.2.540SSPIP	147
5.19.2.541SSPM	147
5.19.2.542SSPM0	147
5.19.2.543SSPM1	147
5.19.2.544SSPM2	147
5.19.2.545SSPM3	147
5.19.2.546SSPOV	147
5.19.2.547SSPSTAT	147
5.19.2.548SSPSTATbits	147
5.19.2.549STATUS	147
5.19.2.550STATUSbits	147
5.19.2.551STKFUL	147
5.19.2.552STKOVF	147
5.19.2.553STKPTR	147
5.19.2.554STKPTRbits	147
5.19.2.555STKUNF	147
5.19.2.556SWDTE	147
5.19.2.557SWDTEN	147
5.19.2.558SYNC	147
5.19.2.559T016BIT	147
5.19.2.560T08BIT	147

5.19.2.561T0CKI	147
5.19.2.562T0CON	147
5.19.2.563T0CONbits	147
5.19.2.564T0CS	147
5.19.2.565T0IE	147
5.19.2.566T0IF	147
5.19.2.567T0PS	147
5.19.2.568T0PS0	148
5.19.2.569T0PS1	148
5.19.2.570T0PS2	148
5.19.2.571T0PS3	148
5.19.2.572T0SE	148
5.19.2.573T13CKI	148
5.19.2.574T1CKI	148
5.19.2.575T1CKPS	148
5.19.2.576T1CKPS0	148
5.19.2.577T1CKPS1	148
5.19.2.578T1CON	148
5.19.2.579T1CONbits	148
5.19.2.580T1OSCEN	148
5.19.2.581T1OSI	148
5.19.2.582T1OSO	148
5.19.2.583T1RUN	148
5.19.2.584T1SYNC	148
5.19.2.585T2CKPS	148
5.19.2.586T2CKPS0	148
5.19.2.587T2CKPS1	148
5.19.2.588T2CON	148
5.19.2.589T2CONbits	148
5.19.2.590T2OUTPS	148
5.19.2.591T2OUTPS0	148
5.19.2.592T2OUTPS1	148
5.19.2.593T2OUTPS2	148
5.19.2.594T2OUTPS3	148
5.19.2.595T3CCP1	148
5.19.2.596T3CCP2	149
5.19.2.597T3CKPS	149
5.19.2.598T3CKPS0	149
5.19.2.599T3CKPS1	149
5.19.2.600T3CON	149

5.19.2.601T3CONbits	149
5.19.2.602T3SYNC	149
5.19.2.603TABLAT	149
5.19.2.604TBLPTR	149
5.19.2.605TBLPTRH	149
5.19.2.606TBLPTRL	149
5.19.2.607TBLPTRU	149
5.19.2.608TMR0H	149
5.19.2.609TMR0IE	149
5.19.2.610TMR0IF	149
5.19.2.611TMR0IP	149
5.19.2.612TMR0L	149
5.19.2.613TMR0ON	149
5.19.2.614TMR1CS	149
5.19.2.615TMR1H	149
5.19.2.616TMR1IE	149
5.19.2.617TMR1IF	149
5.19.2.618TMR1IP	149
5.19.2.619TMR1L	149
5.19.2.620TMR1ON	149
5.19.2.621TMR2	149
5.19.2.622TMR2IE	149
5.19.2.623TMR2IF	149
5.19.2.624TMR2IP	150
5.19.2.625TMR2ON	150
5.19.2.626TMR3CS	150
5.19.2.627TMR3H	150
5.19.2.628TMR3IE	150
5.19.2.629TMR3IF	150
5.19.2.630TMR3IP	150
5.19.2.631TMR3L	150
5.19.2.632TMR3ON	150
5.19.2.633TO	150
5.19.2.634TOS	150
5.19.2.635TOSH	150
5.19.2.636TOSL	150
5.19.2.637TOSU	150
5.19.2.638TOUTPS0	150
5.19.2.639TOUTPS1	150
5.19.2.640TOUTPS2	150

5.19.2.641TOUTPS3	150
5.19.2.642TRISA	150
5.19.2.643TRISA0	150
5.19.2.644TRISA1	150
5.19.2.645TRISA2	150
5.19.2.646TRISA3	150
5.19.2.647TRISA4	150
5.19.2.648TRISA5	150
5.19.2.649TRISA6	150
5.19.2.650TRISA7	150
5.19.2.651TRISAbits	150
5.19.2.652TRISB	151
5.19.2.653TRISB0	151
5.19.2.654TRISB1	151
5.19.2.655TRISB2	151
5.19.2.656TRISB3	151
5.19.2.657TRISB4	151
5.19.2.658TRISB5	151
5.19.2.659TRISB6	151
5.19.2.660TRISB7	151
5.19.2.661TRISBbits	151
5.19.2.662TRISC	151
5.19.2.663TRISCO	151
5.19.2.664TRISC1	151
5.19.2.665TRISC2	151
5.19.2.666TRISC3	151
5.19.2.667TRISC4	151
5.19.2.668TRISC5	151
5.19.2.669TRISC6	151
5.19.2.670TRISC7	151
5.19.2.671TRISCbits	151
5.19.2.672TRISD	151
5.19.2.673TRISD0	151
5.19.2.674TRISD1	151
5.19.2.675TRISD2	151
5.19.2.676TRISD3	151
5.19.2.677TRISD4	151
5.19.2.678TRISD5	151
5.19.2.679TRISD6	151
5.19.2.680TRISD7	152

5.19.2.681TRISDbits	152
5.19.2.682TRISE	152
5.19.2.683TRISE0	152
5.19.2.684TRISE1	152
5.19.2.685TRISE2	152
5.19.2.686TRISEbits	152
5.19.2.687TRMT	152
5.19.2.688TUN	152
5.19.2.689TUN0	152
5.19.2.690TUN1	152
5.19.2.691TUN2	152
5.19.2.692TUN3	152
5.19.2.693TUN4	152
5.19.2.694TX	152
5.19.2.695TX9	152
5.19.2.696TX9D	152
5.19.2.697TXCKP	152
5.19.2.698TXEN	152
5.19.2.699TXIE	152
5.19.2.700TXIF	152
5.19.2.701TXIP	152
5.19.2.702TXREG	152
5.19.2.703TXSTA	152
5.19.2.704TXSTAbits	152
5.19.2.705UA	152
5.19.2.706VCFG	152
5.19.2.707VCFG0	152
5.19.2.708VCFG1	153
5.19.2.709DIRMAG	153
5.19.2.710VPP	153
5.19.2.711VREFN	153
5.19.2.712VREFP	153
5.19.2.713W	153
5.19.2.714WCOL	153
5.19.2.715WDTCON	153
5.19.2.716WDTCONbits	153
5.19.2.717WR	153
5.19.2.718WREG	153
5.19.2.719WREN	153
5.19.2.720WRERR	153

5.19.2.721WUE	153
5.19.2.722Z	153
5.20 Yavin4DefenceSystem/Code/PanTilt.c File Reference	153
5.20.1 Macro Definition Documentation	155
5.20.1.1 AZ_PWM_PIN	155
5.20.1.2 DUTY_CYCLE_TIME	155
5.20.1.3 IN_PWM_PIN	155
5.20.1.4 LATENCY	155
5.20.1.5 PWM_HALF_PERIOD	155
5.20.1.6 PWM_PERIOD	155
5.20.1.7 SERVO_INIT	155
5.20.2 Function Documentation	155
5.20.2.1 calibratePanTilt	155
5.20.2.2 calibratePanTiltRange	155
5.20.2.3 configureBase	155
5.20.2.4 delay2Direction	156
5.20.2.5 direction2Delay	156
5.20.2.6 getDir	157
5.20.2.7 getMaxAzimuthAngle	158
5.20.2.8 getMaxElevationAngle	158
5.20.2.9 getMinAzimuthAngle	159
5.20.2.10 getMinElevationAngle	159
5.20.2.11 increment	160
5.20.2.12 incrementFine	160
5.20.2.13 move	161
5.20.2.14 panTiltISR	162
5.20.2.15 rawDir	162
5.20.2.16 setMaxAzimuthAngle	163
5.20.2.17 setMaxElevationAngle	163
5.20.2.18 setMinAzimuthAngle	163
5.20.2.19 setMinElevationAngle	164
5.20.2.20 updated	164
5.20.2.21 validate	164
5.20.3 Variable Documentation	165
5.20.3.1 arcRange	165
5.20.3.2 azimuth_angle_max	165
5.20.3.3 azimuth_angle_min	165
5.20.3.4 calibration_offset	165
5.20.3.5 changed	165
5.20.3.6 current_direction	165

5.20.3.7	elevation_angle_max	165
5.20.3.8	elevation_angle_min	165
5.20.3.9	global_delay	165
5.21	Yavin4DefenceSystem/Code/PanTilt.h File Reference	165
5.21.1	Macro Definition Documentation	166
5.21.1.1	PAN_TILT_ISR	166
5.21.1.2	PANTILT_H	166
5.21.2	Function Documentation	166
5.21.2.1	calibratePanTilt	166
5.21.2.2	calibratePanTiltRange	166
5.21.2.3	configureBase	167
5.21.2.4	getDir	167
5.21.2.5	getMaxAzimuthAngle	168
5.21.2.6	getMaxElevationAngle	168
5.21.2.7	getMinAzimuthAngle	169
5.21.2.8	getMinElevationAngle	169
5.21.2.9	increment	170
5.21.2.10	incrementFine	170
5.21.2.11	move	171
5.21.2.12	panTiltISR	172
5.21.2.13	rawDir	172
5.21.2.14	setMaxAzimuthAngle	173
5.21.2.15	setMaxElevationAngle	173
5.21.2.16	setMinAzimuthAngle	173
5.21.2.17	setMinElevationAngle	174
5.21.2.18	updated	174
5.22	Yavin4DefenceSystem/Code/Range.c File Reference	174
5.22.1	Macro Definition Documentation	176
5.22.1.1	CCP1_INPT	176
5.22.1.2	INIT_PIN	176
5.22.1.3	INIT_TRIS	176
5.22.1.4	IR_CONV	176
5.22.1.5	range_IR	176
5.22.1.6	range_US	176
5.22.1.7	ULTRA_CONV	176
5.22.2	Function Documentation	176
5.22.2.1	beginUS	176
5.22.2.2	calibrateRange	177
5.22.2.3	configureAD	178
5.22.2.4	configureRange	178

5.22.2.5	<code>fuseRange</code>	179
5.22.2.6	<code>getLastRange</code>	180
5.22.2.7	<code>getMaxRange</code>	180
5.22.2.8	<code>getMinRange</code>	180
5.22.2.9	<code>getNumSamples</code>	181
5.22.2.10	<code>getTargetState</code>	181
5.22.2.11	<code>getUsSampleRate</code>	181
5.22.2.12	<code>range</code>	182
5.22.2.13	<code>rangelSR</code>	183
5.22.2.14	<code>rangeUltrasonic</code>	183
5.22.2.15	<code>rangeUS</code>	184
5.22.2.16	<code>rawRangeIR</code>	185
5.22.2.17	<code>rawRangeUS</code>	185
5.22.2.18	<code>readTargetState</code>	185
5.22.2.19	<code>setMaxRange</code>	186
5.22.2.20	<code>setMinRange</code>	187
5.22.2.21	<code>setNumSamples</code>	187
5.22.2.22	<code>setUsSampleRate</code>	188
5.22.2.23	<code>transRange</code>	188
5.22.3	Variable Documentation	188
5.22.3.1	<code>calibration_offset_IR</code>	188
5.22.3.2	<code>calibration_offset_US</code>	188
5.22.3.3	<code>ccp_value</code>	188
5.22.3.4	<code>current_target_state</code>	189
5.22.3.5	<code>lastIRRange</code>	189
5.22.3.6	<code>lastRange</code>	189
5.22.3.7	<code>lastUSRange</code>	189
5.22.3.8	<code>m_maxRange</code>	189
5.22.3.9	<code>m_minRange</code>	189
5.22.3.10	<code>measuringUS</code>	189
5.22.3.11	<code>numSamples</code>	189
5.22.3.12	<code>rateIR</code>	189
5.22.3.13	<code>rateUS</code>	189
5.23	Yavin4DefenceSystem/Code/Range.h File Reference	189
5.23.1	Macro Definition Documentation	190
5.23.1.1	<code>RANGE_H</code>	190
5.23.1.2	<code>RANGE_INT</code>	190
5.23.2	Function Documentation	190
5.23.2.1	<code>calibrateRange</code>	190
5.23.2.2	<code>configureAD</code>	190

5.23.2.3	configureRange	191
5.23.2.4	getMaxRange	192
5.23.2.5	getMinRange	192
5.23.2.6	getNumSamples	193
5.23.2.7	getTargetState	193
5.23.2.8	getUsSampleRate	193
5.23.2.9	lastRange	194
5.23.2.10	range	194
5.23.2.11	rangelSR	195
5.23.2.12	rawRangeIR	195
5.23.2.13	rawRangeUS	196
5.23.2.14	readTargetState	196
5.23.2.15	setMaxRange	197
5.23.2.16	setMinRange	198
5.23.2.17	setNumSamples	198
5.23.2.18	setUsSampleRate	199
5.24	Yavin4DefenceSystem/Code/ROM2RAM.c File Reference	199
5.24.1	Function Documentation	199
5.24.1.1	strcpypgm2ram	200
5.25	Yavin4DefenceSystem/Code/Serial.c File Reference	200
5.25.1	Macro Definition Documentation	201
5.25.1.1	BS	201
5.25.1.2	CR	201
5.25.1.3	ESC	201
5.25.1.4	NL	201
5.25.1.5	RC_INT_CLEAR	201
5.25.1.6	RC_INT_DISABLE	201
5.25.1.7	RC_INT_ENABLE	201
5.25.1.8	TAB	202
5.25.1.9	TX_INT_CLEAR	202
5.25.1.10	TX_INT_DISABLE	202
5.25.1.11	TX_INT_ENABLE	202
5.25.2	Function Documentation	202
5.25.2.1	clearReceive	202
5.25.2.2	configureSerial	202
5.25.2.3	popEsc	203
5.25.2.4	readString	203
5.25.2.5	receiveCR	204
5.25.2.6	receiveEmpty	204
5.25.2.7	receiveEsc	205

5.25.2.8 receivePeek	205
5.25.2.9 receivePop	205
5.25.2.10 serialISR	206
5.25.2.11 transChar	206
5.25.2.12 transmit	207
5.25.2.13 transmitComplete	208
5.25.3 Variable Documentation	209
5.25.3.1 carriageReturn	209
5.25.3.2 escPressed	209
5.25.3.3 receive_buffer	209
5.25.3.4 transmit_buffer	209
5.26 Yavin4DefenceSystem/Code/Serial.h File Reference	209
5.26.1 Macro Definition Documentation	210
5.26.1.1 SERIAL_H	210
5.26.1.2 SERIAL_INT	210
5.26.2 Function Documentation	210
5.26.2.1 clearReceive	210
5.26.2.2 configureSerial	210
5.26.2.3 popEsc	211
5.26.2.4 readString	211
5.26.2.5 receiveCR	212
5.26.2.6 receiveEmpty	212
5.26.2.7 receiveEsc	213
5.26.2.8 receivePeek	213
5.26.2.9 receivePop	213
5.26.2.10 serialISR	214
5.26.2.11 transChar	214
5.26.2.12 transmit	215
5.26.2.13 transmitComplete	216
5.27 Yavin4DefenceSystem/Code/Temp.c File Reference	217
5.27.1 Function Documentation	217
5.27.1.1 calibrateTemp	218
5.27.1.2 configureTemp	218
5.27.1.3 getTemp	218
5.27.1.4 rawTemp	218
5.27.1.5 readTemp	219
5.27.1.6 readTempx2	219
5.27.2 Variable Documentation	220
5.27.2.1 calibration_offset	220
5.27.2.2 lastTempx2	220

5.28 Yavin4DefenceSystem/Code/Temp.h File Reference	220
5.28.1 Macro Definition Documentation	221
5.28.1.1 TEMP_H	221
5.28.2 Function Documentation	221
5.28.2.1 calibrateTemp	221
5.28.2.2 configureTemp	221
5.28.2.3 getTemp	222
5.28.2.4 rawTemp	222
5.28.2.5 readTemp	222
5.28.2.6 readTempx2	223
5.29 Yavin4DefenceSystem/Code/Tracking.c File Reference	224
5.29.1 Macro Definition Documentation	224
5.29.1.1 diff	224
5.29.1.2 sampleTargetState	225
5.29.1.3 TARGET_RAD	225
5.29.1.4 TIMER	225
5.29.2 Function Documentation	225
5.29.2.1 configureTracking	225
5.29.2.2 newAngle	225
5.29.2.3 prediction	226
5.29.2.4 search	226
5.29.2.5 track	227
5.29.2.6 trackingISR	227
5.30 Yavin4DefenceSystem/Code/Tracking.h File Reference	228
5.30.1 Macro Definition Documentation	228
5.30.1.1 TRACK_H	228
5.30.1.2 TRACK_INT	228
5.30.2 Function Documentation	228
5.30.2.1 configureTracking	228
5.30.2.2 search	229
5.30.2.3 track	230
5.30.2.4 trackingISR	231
5.31 Yavin4DefenceSystem/Code/usart.h File Reference	231
5.31.1 Macro Definition Documentation	233
5.31.1.1 BAUD_IDLE_RX_PIN_STATE_HIGH	233
5.31.1.2 BAUD_IDLE_RX_PIN_STATE_LOW	233
5.31.1.3 BAUD_IDLE_TX_PIN_STATE_HIGH	233
5.31.1.4 BAUD_IDLE_TX_PIN_STATE_LOW	233
5.31.1.5 MEM_MODEL	233
5.31.1.6 USART_ADDEN_OFF	233

5.31.1.7 USART_ADDEN_ON	233
5.31.1.8 USART_ASYNCH_MODE	233
5.31.1.9 USART_BRGH_HIGH	233
5.31.1.10 USART_BRGH_LOW	233
5.31.1.11 USART_CONT_RX	233
5.31.1.12 USART_EIGHT_BIT	233
5.31.1.13 USART_NINE_BIT	233
5.31.1.14 USART_RX_INT_OFF	233
5.31.1.15 USART_RX_INT_ON	233
5.31.1.16 USART_SINGLE_RX	233
5.31.1.17 USART_SYNC_MASTER	233
5.31.1.18 USART_SYNC_SLAVE	233
5.31.1.19 USART_SYNCH_MODE	233
5.31.1.20 USART_TX_INT_OFF	233
5.31.1.21 USART_TX_INT_ON	233
5.32 Yavin4DefenceSystem/Code/User_Interface.c File Reference	233
5.32.1 Macro Definition Documentation	234
5.32.1.1 ADC_MAX	234
5.32.1.2 BACK_PRESS	234
5.32.1.3 CONFIRM_PRESS	234
5.32.2 Function Documentation	235
5.32.2.1 configUSER	235
5.32.2.2 display	235
5.32.2.3 readDial	235
5.32.2.4 readDialForMenu	236
5.32.2.5 userEmpty	236
5.32.2.6 userISR	237
5.32.2.7 userPeek	237
5.32.2.8 userPop	237
5.32.3 Variable Documentation	238
5.32.3.1 receive	238
5.33 Yavin4DefenceSystem/Code/User_Interface.h File Reference	238
5.33.1 Macro Definition Documentation	238
5.33.1.1 BACK_CHAR	238
5.33.1.2 CONFIRM_CHAR	238
5.33.1.3 USER_H	238
5.33.1.4 USER_INT	238
5.33.2 Function Documentation	238
5.33.2.1 configUSER	238
5.33.2.2 display	239

5.33.2.3	readDial	239
5.33.2.4	userEmpty	240
5.33.2.5	userISR	240
5.33.2.6	userPeek	241
5.33.2.7	userPop	241

Chapter 1

Todo List

File [ConfigRegs18f4520.h](#)

Consider if Watchdog Timer should be enabled for production code, and add WDT management code if so.

Consider if Stack Overflow Reset should be enabled for production code. It may be better not to, as there is no re-entrant or recursive code, or dynamic memory allocation here - if the code loads statically it should run.

Change startup code c016iz.c so that everything (including initialised variables) correctly re-starts if [main\(\)](#) ever exits or a BOR or WDT reset occurs.

Global [delay \(unsigned int t\)](#)

Complete, test, debug and document this code!!!

Global [range \(void\)](#)

Read in temperature for US calculation

Global [readTempx2 \(void\)](#)

Test and debug this function

Test and debug this function

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

<code>circularBuffer</code>		
Stores bytes of data in a circular buffer		7
<code>Delay</code>	7
<code>Direction</code>	Stores an inclination and azimuth	8
<code>menuStruct</code>		
Defines a sub menu		9
<code>SubMenu</code>	10
<code>systemState</code>		
Stores the current state of the system		11
<code>TargetStateData</code>		
Stores data about the last target track		11
<code>TrackingData</code>		
Stores the current target information		12

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Yavin4DefenceSystem/Code/CircularBuffers.h	15
Yavin4DefenceSystem/Code/Common.h	16
Yavin4DefenceSystem/Code/ConfigRegs18f4520.h Include file to set the Configuration Bits of a PIC18F4520	22
Yavin4DefenceSystem/Code/EEPROM.c	23
Yavin4DefenceSystem/Code/HardUltest.c	23
Yavin4DefenceSystem/Code/HardUltest.h	29
Yavin4DefenceSystem/Code/Interrupts.c	29
Yavin4DefenceSystem/Code/jEEP.h	33
Yavin4DefenceSystem/Code/LCD.c	33
Yavin4DefenceSystem/Code/LCD.h	38
Yavin4DefenceSystem/Code/LCD_defs.h	40
Yavin4DefenceSystem/Code/Menu.Strings.h	44
Yavin4DefenceSystem/Code/MenuDefs.h	49
Yavin4DefenceSystem/Code/MenuSystem.c	57
Yavin4DefenceSystem/Code/MenuSystem.h	85
Yavin4DefenceSystem/Code/MenuSystem2.c	95
Yavin4DefenceSystem/Code/newfile.h	101
Yavin4DefenceSystem/Code/newmain.c	101
Yavin4DefenceSystem/Code/p18f4520.h	103
Yavin4DefenceSystem/Code/PanTilt.c	153
Yavin4DefenceSystem/Code/PanTilt.h	165
Yavin4DefenceSystem/Code/Range.c	174
Yavin4DefenceSystem/Code/Range.h	189
Yavin4DefenceSystem/Code/ROM2RAM.c	199
Yavin4DefenceSystem/Code/Serial.c	200
Yavin4DefenceSystem/Code/Serial.h	209
Yavin4DefenceSystem/Code/Temp.c	217
Yavin4DefenceSystem/Code/Temp.h	220
Yavin4DefenceSystem/Code/Tracking.c	224
Yavin4DefenceSystem/Code/Tracking.h	228
Yavin4DefenceSystem/Code/usart.h	231
Yavin4DefenceSystem/Code/User_Interface.c	233
Yavin4DefenceSystem/Code/User_Interface.h	238

Chapter 4

Data Structure Documentation

4.1 circularBuffer Struct Reference

Stores bytes of data in a circular buffer.

```
#include <CircularBuffers.h>
```

Data Fields

- unsigned char `head`
- unsigned char `tail`
- unsigned char `data [BUFFERLENGTH]`

4.1.1 Detailed Description

Stores bytes of data in a circular buffer.

typedef of `circularBuffer` struct

Length: BUFFERLENGTH

Description: Defines a circular buffer in which bytes of data can be placed, or removed at any time in the correct order that they were pushed. The accompanying functionality facilitates all necessary circular buffer related operations. This implementation reduces the buffer length by 1 to differentiate between full and empty states. BUFFERLENGTH can be redefined in individual files to get different buffer lengths

4.1.2 Field Documentation

4.1.2.1 unsigned char data[BUFFERLENGTH]

4.1.2.2 unsigned char head

4.1.2.3 unsigned char tail

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

- Yavin4DefenceSystem/Code/CircularBuffers.h

4.2 Delay Struct Reference

Data Fields

- signed int [AzimuthDelay](#)
- signed int [InclinationDelay](#)

4.2.1 Detailed Description

File: ADC.c Author: Grant

Description: Contains all the functionality for the Pan Tilt module. All variables and settings concerning the pan tilt module including the current direction, PDM delay info, min and max settings are private to this module. The interface functions allow all valid access to the module.

Duties: -Software interface to the Pan Tilt Module -Moves Pan Tilt -Reads current Pan Tilt position (based on PDM's) -Generate PDM signals

Functions: Local: void [validate\(unsigned int *delay\)](#); Direction [delay2Direction\(Delay dly\)](#); Delay [direction2Delay\(← Direction dir\)](#);

Public Interface: void [configureBase\(void\)](#); void [move\(Direction destination\)](#); void [increment\(Direction difference\)](#); void [incrementFine\(Direction difference\)](#); Direction [getDir\(void\)](#); void [calibratePanTilt\(Direction reference\)](#); Direction [rawDir\(void\)](#); char [updated\(void\)](#); void [panTiltISR\(void\)](#); char [getMaxAzimuthAngle\(void\)](#); char [getMinAzimuthAngle\(void\)](#); char [getMaxElevationAngle\(void\)](#); char [getMinElevationAngle\(void\)](#); void [setMaxAzimuthAngle\(char p_angle\)](#); void [setMinAzimuthAngle\(char p_angle\)](#); void [setMaxElevationAngle\(char p_angle\)](#); void [setMinElevationAngle\(char p_angle\)](#);

Created on 16 September 2014, 6:47 PM

4.2.2 Field Documentation

4.2.2.1 signed int AzimuthDelay

4.2.2.2 signed int InclinationDelay

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

- [Yavin4DefenceSystem/Code/PanTilt.c](#)

4.3 Direction Struct Reference

Stores an inclination and azimuth.

```
#include <Common.h>
```

Data Fields

- int [azimuth](#)
- int [elevation](#)

4.3.1 Detailed Description

Stores an inclination and azimuth.

typedef of [Direction](#) struct

Description: A fully defined direction in which to point the pan tilt actuator or any other such purpose

Elements: -Azimuth: Contains the azimuth component of the direction (generally degrees) -Inclination: Contains the inclination component of the direction (generally degrees)

4.3.2 Field Documentation

4.3.2.1 int azimuth

4.3.2.2 int elevation

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

- Yavin4DefenceSystem/Code/Common.h

4.4 menuStruct Struct Reference

Defines a sub menu.

```
#include <MenuDefs.h>
```

Data Fields

- menuState menuID
- rom char * serialMessage
- rom char * lcdTitleMessage
- int minVal
- int maxVal
- char increment
- voidFunction serialDisplayFunction
- numericInputFunction confirmFunction
- numericInputFunction lcdDisplayFunction
- voidFunction returnToPrevious

4.4.1 Detailed Description

Defines a sub menu.

typedef of [menuStruct](#) struct

Description: Stores all data required to define a submenu

Elements:

- menuNum:
- SerialMessage: The message to be displayed on entering the state
- LcdTitleMessage: LCD Message to be displayed on entering the state
- MinVal: The minimum value accepted by the state
- MaxVal: The maximum value accepted by the state
- increment:
- serialDisplayFunction: Function to display result over serial
- confirmFunction: Function to confirm entry
- lcdDisplayFunction: Function to display result on LCD
- returnToPrevious: Function to return to previous menu state

4.4.2 Field Documentation

- 4.4.2.1 `numericInputFunction confirmFunction`
- 4.4.2.2 `char increment`
- 4.4.2.3 `numericInputFunction lcdDisplayFunction`
- 4.4.2.4 `rom char* lcdTitleMessage`
- 4.4.2.5 `int maxVal`
- 4.4.2.6 `menuState menuID`
- 4.4.2.7 `int minVal`
- 4.4.2.8 `voidFunction returnToPrevious`
- 4.4.2.9 `voidFunction serialDisplayFunction`
- 4.4.2.10 `rom char* serialMessage`

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

- Yavin4DefenceSystem/Code/MenuDefs.h

4.5 SubMenu Struct Reference

Collaboration diagram for SubMenu:



Data Fields

- `char serialMessage [MAX_SER_MSG_LEN]`
- `char lcdMessage [MAX_LCD_MSG_LEN]`
- `char serialOptions [MAX_NUM_OPTIONS]`
- `char inptOptions [MAX_NUM_OPTIONS]`
- `void(* numericFunction)(int)`
- `void(* defaultFunction)(void)`
- `struct SubMenu * subMenus`

4.5.1 Field Documentation

- 4.5.1.1 `void(* defaultFunction)(void)`

- 4.5.1.2 char inptOptions[MAX_NUM_OPTIONS]
- 4.5.1.3 char lcdMessage[MAX_LCD_MSG_LEN]
- 4.5.1.4 void(* numericFunction)(int)
- 4.5.1.5 char serialMessage[MAX_SER_MSG_LEN]
- 4.5.1.6 char serialOptions[MAX_NUM_OPTIONS]
- 4.5.1.7 struct SubMenu* subMenus

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

- Yavin4DefenceSystem/Code/[Menusem2.c](#)

4.6 systemState Struct Reference

Stores the current state of the system.

```
#include <Common.h>
```

Data Fields

- [possible_states current](#)
- [possible_states previous](#)

4.6.1 Detailed Description

Stores the current state of the system.

typedef of [systemState](#) struct

Description: Stores which state the system is currently in, as defined by the possible_states enumeration. Also stores the previous system state so the system knows if this is the first iteration of the state

4.6.2 Field Documentation

4.6.2.1 [possible_states current](#)

4.6.2.2 [possible_states previous](#)

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

- Yavin4DefenceSystem/Code/[Common.h](#)

4.7 TargetStateData Struct Reference

Stores data about the last target track.

Data Fields

- unsigned `bad_dirs`: 2
- unsigned `out_of_ir`: 2
- unsigned `good_tracks`: 2
- unsigned `centre`: 2

4.7.1 Detailed Description

Stores data about the last target track.

typedef of `targetStateData` struct

Description: Stores the number of samples of each type sampled in the previous target track

Elements: -`bad_dirs` - stores the number of BAD_DIR's sampled - only 3 bits
 -`out_of_ir` - Stores the number of OUT_OF_IR's sampled - only 3 bits
 -`good_tracks` - Stores the number of GOOD_TRACK's sampled - only 3 bits

4.7.2 Field Documentation

4.7.2.1 unsigned `bad_dirs`

4.7.2.2 unsigned `centre`

4.7.2.3 unsigned `good_tracks`

4.7.2.4 unsigned `out_of_ir`

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

- Yavin4DefenceSystem/Code/[Tracking.c](#)

4.8 TrackingData Struct Reference

Stores the current target information.

`#include <Common.h>`

Data Fields

- unsigned int `range`
- int `azimuth`
- int `elevation`

4.8.1 Detailed Description

Stores the current target information.

typedef of `TrackingData` struct

Description: Stores distance, azimuth and inclination tracking data to the target. This struct is used by the tracking function and others to communicate the position of the target

4.8.2 Field Documentation

4.8.2.1 int azimuth

4.8.2.2 int elevation

4.8.2.3 unsigned int range

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

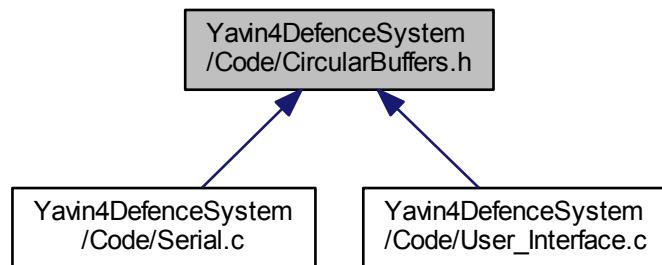
- Yavin4DefenceSystem/Code/[Common.h](#)

Chapter 5

File Documentation

5.1 Yavin4DefenceSystem/Code/CircularBuffers.h File Reference

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



Data Structures

- struct `circularBuffer`

Stores bytes of data in a circular buffer.

Macros

- `#define incMod(ptr) if(ptr==BUFFERLENGTH-1) ptr = 0; else ptr++;`
Buffer related macro functionality.
- `#define empty(buf) (buf.tail == buf.head)`
- `#define full(buf) (buf.head == (buf.tail + 1) % BUFFERLENGTH)`
- `#define peek(buf) buf.data[buf.head]`
- `#define push(byte, buf) buf.data[buf.tail] = byte; if(full(buf)) incMod(buf.head); incMod(buf.tail)`
- `#define pop(buf) buf.data[buf.head]; if (!empty(buf)) incMod(buf.head)`
- `#define init(buf) buf.tail = 0; buf.head = 0`
- `#define BUFFERLENGTH 80`
- `#define BUFFERS_H 0`

5.1.1 Macro Definition Documentation

5.1.1.1 #define BUFFERLENGTH 80

5.1.1.2 #define BUFFERS_H 0

5.1.1.3 #define empty(buf) (buf.tail == buf.head)

5.1.1.4 #define full(buf) (buf.head == (buf.tail + 1) % BUFFERLENGTH)

5.1.1.5 #define incMod(ptr) if(ptr==BUFFERLENGTH-1) ptr = 0; else ptr++;

Buffer related macro functionality.

File: [CircularBuffers.h](#) Author: Grant

Description: Contains Circular Buffer functionality and structure definitions. This allows Circular Buffers to be used in any file by simply including this header. Furthermore, circular buffer functionality can be altered on a program level.

Contains: -Definition of CircularBuffer type -Peek, pop, push and init functionality -Empty, full, queries

Created on 8 October 2014, 10:30 AM

5.1.1.6 #define init(buf) buf.tail = 0; buf.head = 0

5.1.1.7 #define peek(buf) buf.data[buf.head]

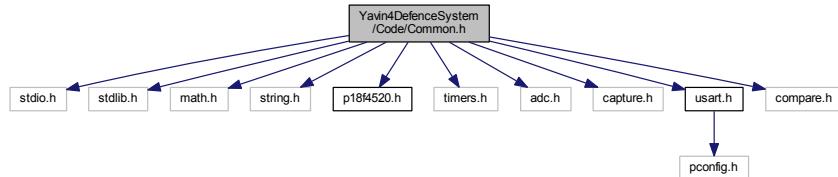
5.1.1.8 #define pop(buf) buf.data[buf.head]; if (!empty(buf)) incMod(buf.head)

5.1.1.9 #define push(byte, buf) buf.data[buf.tail] = byte; if(full(buf)) incMod(buf.head); incMod(buf.tail)

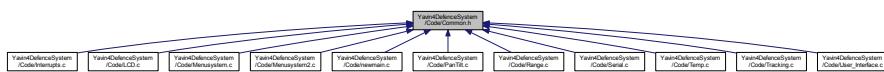
5.2 Yavin4DefenceSystem/Code/Common.h File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>
#include <p18f4520.h>
#include <timers.h>
#include <adc.h>
#include <capture.h>
#include <usart.h>
#include <compare.h>
```

Include dependency graph for Common.h:



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



Data Structures

- struct [Direction](#)
Stores an inclination and azimuth.
- struct [TrackingData](#)
Stores the current target information.
- struct [systemState](#)
Stores the current state of the system.

Macros

- #define [MNML](#)
- #define [NEXT_STATE](#)(s, state) state.previous = state.current; state.current = s
Macros to change the state of the system.
- #define [NEXT_STATE_PTR](#)(s, state) state->previous = state->current; state->current = s
- #define [DIV_2](#)(v) ((v) >> 1)
Efficient Division macros based on bit shifting, Cannot be used on negative numbers.
- #define [DIV_4](#)(v) ((v) >> 2)
- #define [DIV_8](#)(v) ((v) >> 3)
- #define [DIV_16](#)(v) ((v) >> 4)
- #define [DIV_32](#)(v) ((v) >> 5)
- #define [DIV_64](#)(v) ((v) >> 6)
- #define [DIV_128](#)(v) ((v) >> 7)
- #define [DIV_256](#)(v) ((v) >> 8)
- #define [DIV_512](#)(v) ((v) >> 9)
- #define [DIV_1024](#)(v) ((v) >> 10)
- #define [DIV_4096](#)(v) ((v) >> 12)
- #define [DIV_65536](#)(v) ((v) >> 16)
- #define [SWAP](#)(x, y) (y = (y ^ (x = (x ^ (y = (x ^ y))))))
Swaps the values in two variables.
- #define [ADC_IR_READ](#) 0x01
ADC Channel Select macros.
- #define [ADC_TEMP_READ](#) 0x05
- #define [ADC_DIAL_READ](#) 0x09
- #define [TX_INT](#) (PIR1bits.TXIF && PIE1bits.TXIE)
Interrupt macros.
- #define [RC_INT](#) (PIR1bits.RCIF && PIE1bits.RCIE)
- #define [CCP1_INT](#) (PIR1bits.CCP1IF && PIE1bits.CCP1IE)
- #define [CCP2_INT](#) (PIR2bits.CCP2IF && PIE2bits.CCP2IE)
- #define [INT0_INT](#) (INTCONbits.INT0IF && INTCONbits.INT0IE)
- #define [INT1_INT](#) (INTCON3bits.INT1IF && INTCON3bits.INT1IE)
- #define [INT2_INT](#) (INTCON3bits.INT2IF && INTCON3bits.INT2IE)
- #define [RB_INT](#) (INTCONbits.RBIF && INTCONbits.RBIE)
- #define [TMR2_INT](#) (PIR1bits.TMR2IF && PIE1bits.TMR2IE)
- #define [TMR0_INT](#) (INTCONbits.TMR0IF && INTCONbits.TMR0IE)

- #define TMR1_INT (PIR1bits.TMR1IF && PIE1bits.TMR1IE)
- #define TMR3_INT (PIR2bits.TMR3IF && PIE2bits.TMR3IE)
- #define ADC_INT (PIR1bits.ADIF && PIE1bits.ADIE)
- #define SSP_INT (PIR1bits.SSPIF && PIE1bits.SSPIE)
- #define BCL_INT (PIR2bits.BCLIF && PIE2bits.BCLIE)
- #define LVD_INT (PIR2bits.LVDIF && PIE2bits.LVDIE)
- #define CCP1_CLEAR (PIR1bits.CCP1IF = 0)
- #define CCP2_CLEAR (PIR2bits.CCP2IF = 0)
- #define CLOCK 10000000
Define the clock rate and FOSC_4.
- #define FOSC_4 2500000
- #define INT_SETUP()
- #define COMMON_H

Enumerations

- enum TargetState {
 NO_TARGET, OUT_OF_IR, BAD_DIR, GOOD_TRACK,
CLOSE_RANGE }
- The different types of tracking possible depending on which sensors observe the target.*
- enum PanTiltSettings { MAX_AZ, MIN_AZ, MAX_EL, MIN_EL }
The different PanTilt settings that can be changed.
- enum RangeSettings {
 MAX_RANGE, MIN_RANGE, IR_RATE, US_RATE,
IR_SAMPLES, US_SAMPLES }
- The different Range settings that can be changed.*
- enum escSqnce { CLR_SCN, ERS_LN }
Available ANSI escape Sequences.
- enum possible_states {
 UNDEF, INIT, SRCH, TRCK,
MENU, RAW_RANGE_STATE }
- Define the possible states the system can be in.*

5.2.1 Macro Definition Documentation

5.2.1.1 #define ADC_DIAL_READ 0x09

5.2.1.2 #define ADC_INT (PIR1bits.ADIF && PIE1bits.ADIE)

5.2.1.3 #define ADC_IR_READ 0x01

ADC Channel Select macros.

5.2.1.4 #define ADC_TEMP_READ 0x05

5.2.1.5 #define BCL_INT (PIR2bits.BCLIF && PIE2bits.BCLIE)

5.2.1.6 #define CCP1_CLEAR (PIR1bits.CCP1IF = 0)

5.2.1.7 #define CCP1_INT (PIR1bits.CCP1IF && PIE1bits.CCP1IE)

5.2.1.8 #define CCP2_CLEAR (PIR2bits.CCP2IF = 0)

5.2.1.9 #define CCP2_INT (PIR2bits.CCP2IF && PIE2bits.CCP2IE)

5.2.1.10 #define CLOCK 10000000

Define the clock rate and FOSC_4.

5.2.1.11 #define COMMON_H

5.2.1.12 #define DIV_1024(v) ((v) >> 10)

5.2.1.13 #define DIV_128(v) ((v) >> 7)

5.2.1.14 #define DIV_16(v) ((v) >> 4)

5.2.1.15 #define DIV_2(v) ((v) >> 1)

Efficient Division macros based on bit shifting, Cannot be used on negative numbers.

5.2.1.16 #define DIV_256(v) ((v) >> 8)

5.2.1.17 #define DIV_32(v) ((v) >> 5)

5.2.1.18 #define DIV_4(v) ((v) >> 2)

5.2.1.19 #define DIV_4096(v) ((v) >> 12)

5.2.1.20 #define DIV_512(v) ((v) >> 9)

5.2.1.21 #define DIV_64(v) ((v) >> 6)

5.2.1.22 #define DIV_65536(v) ((v) >> 16)

5.2.1.23 #define DIV_8(v) ((v) >> 3)

5.2.1.24 #define FOSC_4 2500000

5.2.1.25 #define INT0_INT (INTCONbits.INT0IF && INTCONbits.INT0IE)

5.2.1.26 #define INT1_INT (INTCON3bits.INT1IF && INTCON3bits.INT1IE)

5.2.1.27 #define INT2_INT (INTCON3bits.INT2IF && INTCON3bits.INT2IE)

5.2.1.28 #define INT_SETUP()

Value:

```
INTCONbits.GIEH = 1; \
INTCONbits.GIEL = 1; \
PIR1 = 0; \
PIR2 = 0; \
RCONbits.IPEN = 1; \
IPR1 = 0; \
IPR2 = 0; \
IPR1bits.TXIP = 1; \
IPR2bits.CCP2IP = 1; \
IPR1bits.CCP1IP = 1;
```

5.2.1.29 #define LVD_INT (PIR2bits.LVDIF && PIE2bits.LVDIE)

5.2.1.30 #define MNML

File: [Common.h](#) Author: Grant

Description: Contains all program scope definitions, declarations and inclusions. This header should be included in all source files by default.

Contains: -PIC18F family library headers -Direction struct typedef -TrackingData struct typedef -systemState struct typedef -system state macro functionality -TargetState enumeration -Interrupt flag macros -Division macros -Clock frequency definitions -SWAP macro functionality

Created on 11 September 2014, 12:24 PM

5.2.1.31 #define NEXT_STATE(s, state) state.previous = state.current; state.current = s

Macros to change the state of the system.

5.2.1.32 #define NEXT_STATE_PTR(s, state) state->previous = state->current; state->current = s

5.2.1.33 #define RB_INT (INTCONbits.RBIF && INTCONbits.RBIE)

5.2.1.34 #define RC_INT (PIR1bits.RCIF && PIE1bits.RCIE)

5.2.1.35 #define SSP_INT (PIR1bits.SSPIF && PIE1bits.SSPIE)

5.2.1.36 #define SWAP(x, y)(y = (y ^ (x = (x ^ (y = (x ^ y))))))

Swaps the values in two variables.

5.2.1.37 #define TMR0_INT (INTCONbits.TMR0IF && INTCONbits.TMR0IE)

5.2.1.38 #define TMR1_INT (PIR1bits.TMR1IF && PIE1bits.TMR1IE)

5.2.1.39 #define TMR2_INT (PIR1bits.TMR2IF && PIE1bits.TMR2IE)

5.2.1.40 #define TMR3_INT (PIR2bits.TMR3IF && PIE2bits.TMR3IE)

5.2.1.41 #define TX_INT (PIR1bits.TXIF && PIE1bits.TXIE)

Interrupt macros.

5.2.2 Enumeration Type Documentation

5.2.2.1 enum escSqnce

Available ANSI escape Sequences.

Enumerator

CLR_SCN

ERS_LN

5.2.2.2 enum PanTiltSettings

The different PanTilt settings that can be changed.

Enumerator

MAX_AZ
MIN_AZ
MAX_EL
MIN_EL

5.2.2.3 enum possible_states

Define the possible states the system can be in.

Enumerator

UNDEF
INIT
SRCH
TRCK
MENU
RAW_RANGE_STATE

5.2.2.4 enum RangeSettings

The different Range settings that can be changed.

Enumerator

MAX_RANGE
MIN_RANGE
IR_RATE
US_RATE
IR_SAMPLES
US_SAMPLES

5.2.2.5 enum TargetState

The different types of tracking possible depending on which sensors observe the target.

Enumerator

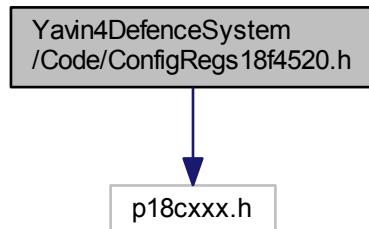
NO_TARGET
OUT_OF_IR
BAD_DIR
GOOD_TRACK
CLOSE_RANGE

5.3 Yavin4DefenceSystem/Code/ConfigRegs18f4520.h File Reference

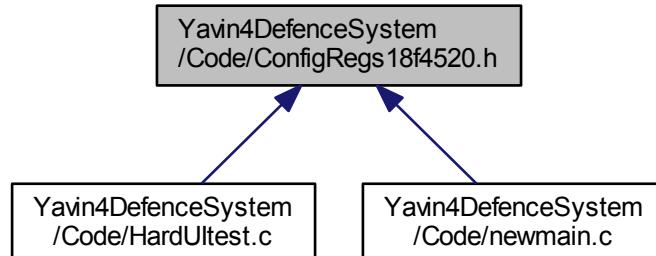
Include file to set the Configuration Bits of a PIC18F4520.

```
#include <p18cxx.h>
```

Include dependency graph for ConfigRegs18f4520.h:



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



5.3.1 Detailed Description

Include file to set the Configuration Bits of a PIC18F4520.

If the macro `__DEBUG` is defined, the bits are set as appropriate for development and debugging:

- HS Oscillator; Oscillator Switch disabled; Power-On Timer;
- Brown-out Reset disabled;
- Watchdog Timer disabled;
- CCP2 Multiplex disabled;
- Stack Overflow Reset;
- Low-voltage Programming disabled, Debug mode enabled;

- No protection bits set.

If the macro __DEBUG is NOT defined, the bits are set as appropriate for production code release:

- HS Oscillator; Oscillator Switch disabled; Power-On Timer;
- Brown-out Reset enabled at 4.2V;
- Watchdog Timer disabled;
- CCP2 Multiplex disabled;
- Stack Overflow Reset;
- Low-voltage Programming and Debug mode disabled;
- No protection bits set.

Version

0.1 - derived from ConfigRegs18F452.h

Date

28-Aug-2014

Author

David Rye

Note

This file was generated by the compiler from the command line: mcc18 -p18f4520 –help-config > configReg18F4520.h

Todo Consider if Watchdog Timer should be enabled for production code, and add WDT management code if so.

Todo Consider if Stack Overflow Reset should be enabled for production code. It may be better not to, as there is no re-entrant or recursive code, or dynamic memory allocation here - if the code loads statically it should run.

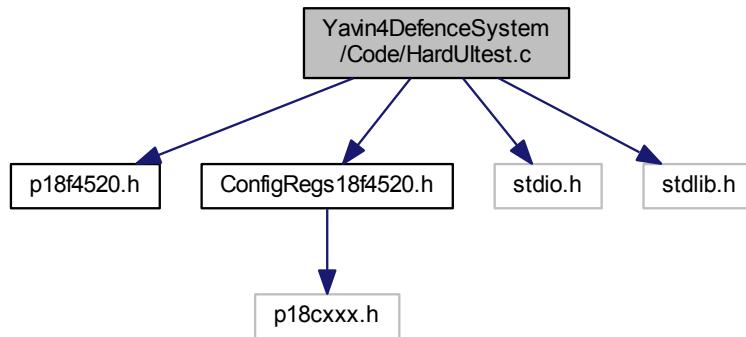
Todo Change startup code c016iz.c so that everything (including initialised variables) correctly re-starts if [main\(\)](#) ever exits or a BOR or WDT reset occurs.

5.4 Yavin4DefenceSystem/Code/EEPROM.c File Reference

5.5 Yavin4DefenceSystem/Code/HardUltest.c File Reference

```
#include <p18f4520.h>
#include "ConfigRegs18f4520.h"
#include <stdio.h>
#include <stdlib.h>
```

Include dependency graph for HardUlttest.c:



Macros

- #define ADCON0_set 0x41
- #define ADCON1_patch 0x80

Functions

- void PORTconfig (void)
- void LEDselect (char val)
- void setup (void)
- void INTconfig (void)
- void ADconfig (void)
- void ADgo (void)
- void ISR_high_vect (void)
- void ISR_high (void)
- void ISR_low_vect (void)
- void ISR_low (void)
- void main (void)

Variables

- char LEDtest [] ={0x33, 0x66, 0x9A, 0xCE, 0xFF}
- char LEDson [] ={0x00, 0x08, 0x0C, 0x0E, 0x0F}
- char * LEDtestptr
- char * LEDsonptr

5.5.1 Macro Definition Documentation

5.5.1.1 #define ADCON0_set 0x41

5.5.1.2 #define ADCON1_patch 0x80

5.5.2 Function Documentation

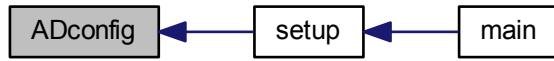
5.5.2.1 void ADconfig (void)

Patch until error is resolved

Configure port configuration, all set to digital I/O, except AN0 use analogue and use standard vrefs

FOSC/8 and enable conversions

Here is the caller graph for this function:



5.5.2.2 void ADgo (void)

Assign pointer to beginning of array

Assign pointer for LED choice

Begin a conversion

Poll the conversion flag until complete

Loop through the options

If the converted number is less than the stored reference, turn on corresponding LED

If the correct bin is found, get out of loop

Increment pointers for next case

Here is the caller graph for this function:



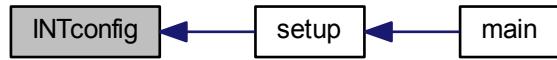
5.5.2.3 void INTconfig (void)

enable RB port change interrupt

Check RBIF for change (at least one) (software cleared=> read or write)

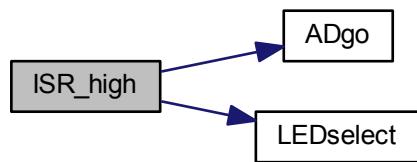
Set pin change as high interrupt

Here is the caller graph for this function:



5.5.2.4 void ISR_high (void)

Here is the call graph for this function:

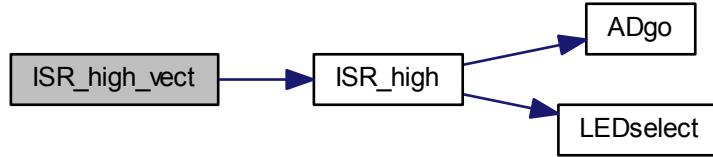


Here is the caller graph for this function:



5.5.2.5 void ISR_high_vect (void)

Here is the call graph for this function:

**5.5.2.6 void ISR_low (void)**

Here is the caller graph for this function:

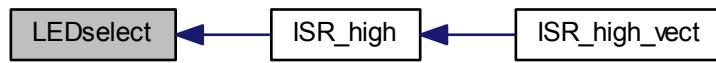
**5.5.2.7 void ISR_low_vect (void)**

Here is the call graph for this function:



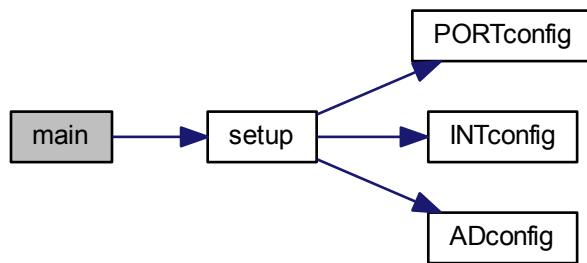
5.5.2.8 void LEDselect (char val)

Here is the caller graph for this function:



5.5.2.9 void main (void)

Here is the call graph for this function:



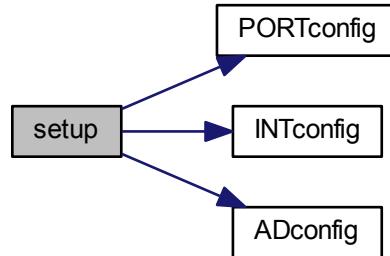
5.5.2.10 void PORTconfig (void)

Here is the caller graph for this function:



5.5.2.11 void setup (void)

Here is the call graph for this function:



Here is the caller graph for this function:



5.5.3 Variable Documentation

5.5.3.1 char LEDson[] ={0x00, 0x08, 0x0C, 0x0E, 0x0F}

5.5.3.2 char* LEDsonpnt

5.5.3.3 char LEDtest[] ={0x33, 0x66, 0x9A, 0xCE, 0xFF}

5.5.3.4 char* LEDtestpnt

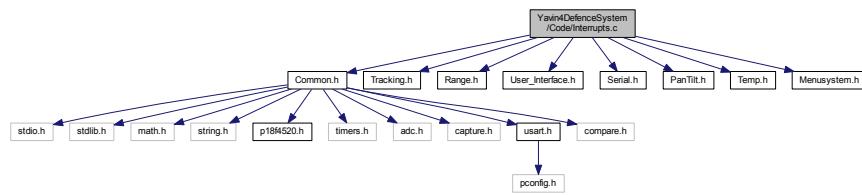
5.6 Yavin4DefenceSystem/Code/HardUltest.h File Reference

5.7 Yavin4DefenceSystem/Code/Interrupts.c File Reference

```

#include "Common.h"
#include "Tracking.h"
#include "Range.h"
#include "User_Interface.h"
#include "Serial.h"
#include "PanTilt.h"
#include "Temp.h"
#include "Menusystem.h"
  
```

Include dependency graph for Interrupts.c:



Functions

- void [lowISR](#) (void)
- void [highISR](#) (void)
- void [highVector](#) (void)
- void [lowVector](#) (void)

5.7.1 Function Documentation

5.7.1.1 void [highISR](#) (void)

Function: [highISR\(void\)](#)

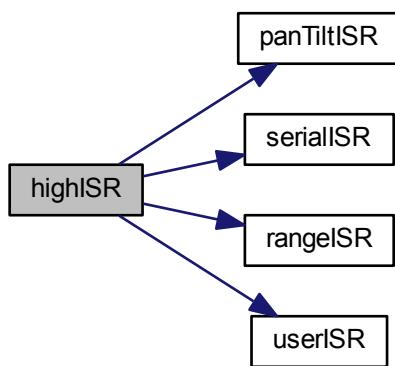
Include: Local to [Interrupts.c](#)

Description: Interrupt Service Routine to check what condition initiated a high priority interrupt call, and perform the nessicary action

Arguments: None

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.1.2 void highVector (void)

Function: [highVector\(void\)](#)

Include: Local to [Interrupts.c](#)

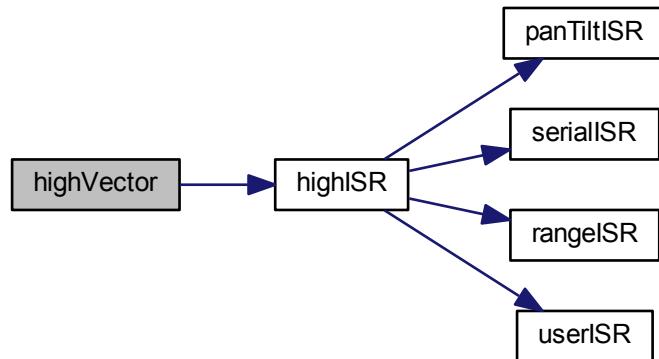
Description: Sends program control to the high priority ISR

Arguments: None

Returns: None

Remarks: This is an interrupt vector, placing a goto in the high priority interrupt table to call the high priority ISR

Here is the call graph for this function:



5.7.1.3 void lowISR (void)

File: [Interrupts.c](#) Author: Grant

Description: Contains the interrupt service routines for the system.

Duties: -Handle all interrupts called by the system -Call relevant module service routines

Created on 8 October 2014, 11:27 AM

Function: [lowISR\(void\)](#)

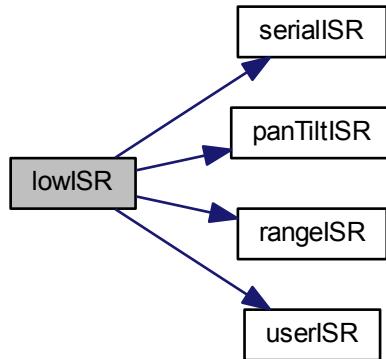
Include: Local to [Interrupts.c](#)

Description: Interrupt Service Routine to check what condition initiated a low priority interrupt call, and perform the nessicary action

Arguments: None

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.1.4 void lowVector(void)

Function: `lowVector(void)`

Include: Local to [Interrupts.c](#)

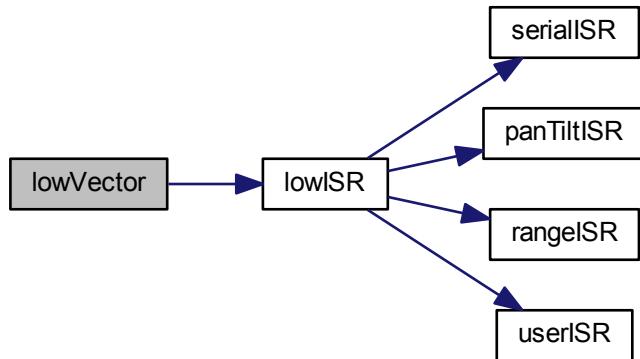
Description: Sends program control to the low priority ISR

Arguments: None

Returns: None

Remarks: This is an interrupt vector, placing a goto in the low priority interrupt table to call the low priority ISR

Here is the call graph for this function:



5.8 Yavin4DefenceSystem/Code/jEEP.h File Reference

Macros

- `#define EEPADDRESS 0x00`

Functions

- `void sendEep (int number, int address)`
- `int readEep (int address)`

5.8.1 Macro Definition Documentation

5.8.1.1 `#define EEPADDRESS 0x00`

5.8.2 Function Documentation

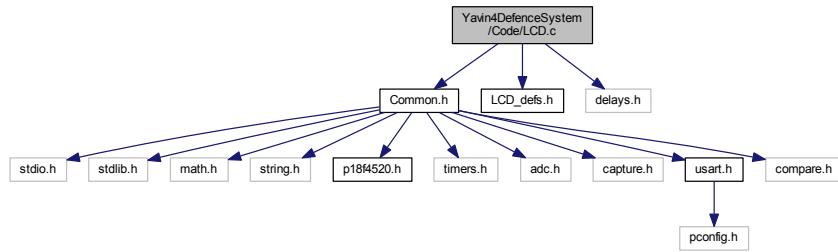
5.8.2.1 `int readEep (int address)`

5.8.2.2 `void sendEep (int number, int address)`

5.9 Yavin4DefenceSystem/Code/LCD.c File Reference

```
#include "Common.h"
#include "LCD_defs.h"
#include <delays.h>
```

Include dependency graph for LCD.c:



Functions

- void [delay](#) (unsigned int t)

Include:
- char [lcdBusy](#) (void)

Include:
- void [lcdWrite](#) (unsigned char byte, unsigned char mode)

Include: LCD.h.
- void [configLCD](#) (void)

Initialises the LCD so that it can be used.
- void [lcdWriteString](#) (char *string, char line)

Include: LCD.h.
- void [lcdWriteChar](#) (char byte, char line, char column)

Feed character, line (1 or 2), and column(1-16)

5.9.1 Function Documentation

5.9.1.1 void configLCD (void)

Initialises the LCD so that it can be used.

Feed character, line (1 or 2), and column(1-16)

Function: [configLCD\(void\)](#)

Include: [LCD.h](#)

Description: Initialises the LCD hardware so that data can be displayed on the LCD in local interface mode

Arguments: None

Returns: None Set data lines to output

Set control lines low

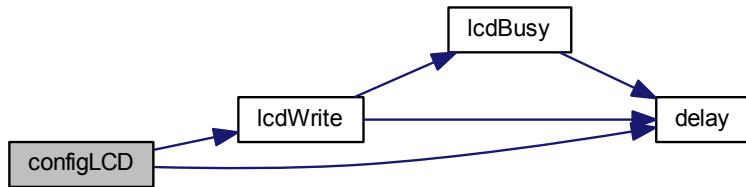
8 bits, 2 lines, 5x7 dots, 0x38

Display on, cursor off, blinking off (Z, 0x0F)

Cursor moves right?, display not shift (Z, 0x04)

Clear display

Here is the call graph for this function:



5.9.1.2 void delay (unsigned int t)

Include:

File: [User_Interface.c](#) Author:

Description: Contains all the interface to the LCD hardware

Duties: -Interfaces with and controls the LCD -Displays data on the LCD

Todo Complete, test, debug and document this code!!!

Created on 15 September 2014, 1:21 PM

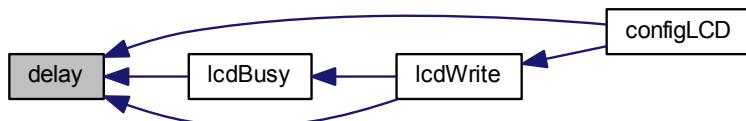
Function: `delay(void)`

Description:

Arguments: `t` -

Returns: None

Here is the caller graph for this function:



5.9.1.3 char lcdBusy (void)

Include:

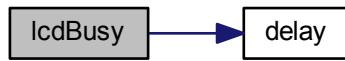
Function: [lcdBusy\(void\)](#)

Description:

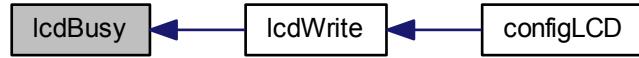
Arguments: None

Returns: None Set data lines to input
Set the enable low, ensure clock signal is low
Set RW to read read from LCD
Set the register select to low for instruction
Set the enable high, ensure clock signal is high to allow data transmission
Set control lines to output
Set data lines to output
Set enable low to allow data to flow through
Set control lines to output
Set data lines to output
Set enable low to allow data to flow through

Here is the call graph for this function:



Here is the caller graph for this function:



5.9.1.4 void `LcdWrite(unsigned char byte, unsigned char mode)`

Include: [LCD.h](#).

Function: [`LcdWrite\(unsigned char byte, unsigned char mode\)`](#)

Description:

Arguments: byte - mode -

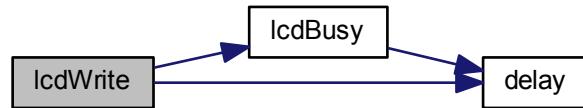
Returns: None Set data lines to output

Write data to data lines

Set clock high

Set clock low

Here is the call graph for this function:



Here is the caller graph for this function:



5.9.1.5 void LcdWriteChar (char byte, char line, char column)

Feed character, line (1 or 2), and column(1-16)

Feed character string, and line (1 or 2)

Function: LcdWriteChar(unsigned char byte, unsigned char line, unsigned char column)

Include: [LCD.h](#)

Description: Writes a character to the LCD at a given location

Arguments: byte - byte to write line - line to write to column - column to write in

Returns: NoneFeed character, line (1 or 2), and column(1-16)

5.9.1.6 void LcdWriteString (char * string, char line)

Include: [LCD.h](#).

Function: LcdWriteString(char *string, unsigned char line)

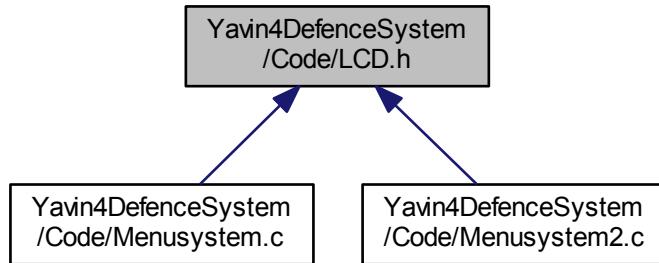
Description: Writes a string to the LCD at the given line

Arguments: string - The string to write line - the line to write to

Returns: NoneFeed character string, and line (1 or 2) Also include information about which row

5.10 Yavin4DefenceSystem/Code/LCD.h File Reference

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



Macros

- `#define LCD_H`

Functions

- `void LcdWriteString (char *string, char line)`
Include: LCD.h.
- `void LcdWriteChar (char byte, char line, char column)`
Feed character string, and line (1 or 2)
- `void configLCD (void)`
Feed character, line (1 or 2), and column(1-16)

5.10.1 Macro Definition Documentation

5.10.1.1 `#define LCD_H`

5.10.2 Function Documentation

5.10.2.1 `void configLCD (void)`

Feed character, line (1 or 2), and column(1-16)

Function: configLCD(void)

Initialises the LCD so that it can be used

Include: LCD.h

Description: Initialises the LCD hardware so that data can be displayed on the LCD in local interface mode

Arguments: None

Returns: None

Feed character, line (1 or 2), and column(1-16)

Function: configLCD(void)

Include: [LCD.h](#)

Description: Initialises the LCD hardware so that data can be displayed on the LCD in local interface mode

Arguments: None

Returns: None Set data lines to output

Set control lines low

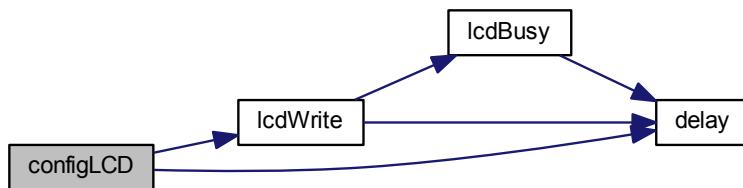
8 bits, 2 lines, 5x7 dots, 0x38

Display on, cursor off, blinking off (Z, 0x0F)

Cursor moves right?, display not shift (Z, 0x04)

Clear display

Here is the call graph for this function:



5.10.2.2 void lcdWriteChar(char byte, char line, char column)

Feed character string, and line (1 or 2)

Function: `lcdWriteChar(unsigned char byte, unsigned char line, unsigned char column)`

Include: [LCD.h](#)

Description: Writes a character to the LCD at a given location

Arguments: byte - byte to write line - line to write to column - column to write in

Returns: None

Feed character string, and line (1 or 2)

Function: `lcdWriteChar(unsigned char byte, unsigned char line, unsigned char column)`

Include: [LCD.h](#)

Description: Writes a character to the LCD at a given location

Arguments: byte - byte to write line - line to write to column - column to write in

Returns: NoneFeed character, line (1 or 2), and column(1-16)

5.10.2.3 void lcdWriteString(char * string, char line)

Include: [LCD.h](#).

File: [LCD.h](#) Author: Grant

Description: Contains the public interface for the LCD module.

Created on 17 September 2014, 9:01 PM

Function: `LcdWriteString(char *string, unsigned char line)`

Description: Writes a string to the LCD at the given line

Arguments: `string` - The string to write `line` - the line to write to

Returns: None

Function: `LcdWriteString(char *string, unsigned char line)`

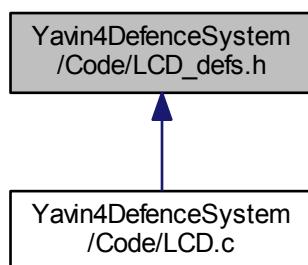
Description: Writes a string to the LCD at the given line

Arguments: `string` - The string to write `line` - the line to write to

Returns: NoneFeed character string, and line (1 or 2) Also include information about which row

5.11 Yavin4DefenceSystem/Code/LCD_defs.h File Reference

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



Macros

- `#define LCD_D0 PORTDbits.RD0`
- `#define LCD_D1 PORTDbits.RD1`
- `#define LCD_D2 PORTDbits.RD2`
- `#define LCD_D3 PORTDbits.RD3`
- `#define LCD_D4 PORTDbits.RD4`
- `#define LCD_D5 PORTDbits.RD5`
- `#define LCD_D6 PORTDbits.RD6`
- `#define LCD_D7 PORTDbits.RD7`
- `#define LCD_D0_DIR TRISDbits.RD0`
- `#define LCD_D1_DIR TRISDbits.RD1`
- `#define LCD_D2_DIR TRISDbits.RD2`
- `#define LCD_D3_DIR TRISDbits.RD3`
- `#define LCD_D4_DIR TRISDbits.RD4`
- `#define LCD_D5_DIR TRISDbits.RD5`
- `#define LCD_D6_DIR TRISDbits.RD6`
- `#define LCD_D7_DIR TRISDbits.RD7`
- `#define LCD_DATA_LINE PORTD`
- `#define LCD_DATA_LINE_DIR TRISD`

- #define LCD_RS PORTCbits.RC4
- #define LCD_RW PORTCbits.RC5
- #define LCD_E PORTAbits.RA4
- #define LCD_RS_DIR TRISCbits.RC4
- #define LCD_RW_DIR TRISCbits.RC5
- #define LCD_E_DIR TRISAbits.RA4
- #define LCD_INS 0
- #define LCD_DATA 1
- #define LCD_WRITE 0
- #define LCD_READ 1
- #define LCD_CLKLOW 0
- #define LCD_CLKHIGH 1
- #define LCD_OFF 0
- #define LCD_ON 1
- #define LCD_INPUT 0xFF
- #define LCD_OUTPUT 0x00
- #define LCD_PIN_INPUT 1
- #define LCD_PIN_OUTPUT 0
- #define DISPCLR 0x01
- #define RTNHOME 0x02
- #define SETENTRYMODE 0x04
- #define ID_CURSL 0x00
- #define ID_CURSR 0x02
- #define SH_DISPNSHIFT 0x00
- #define SH_DISPSHIFT 0x01
- #define DISPOPT 0x08
- #define D_DISPOFF 0x00
- #define D_DISPON 0x04
- #define C_CURSOFF 0x00
- #define C_CURSOR 0x02
- #define B_BLINKOFF 0x00
- #define B_BLINKON 0x01
- #define DISPSHIFTCURS 0x10
- #define SC_CURSMOVE 0x00
- #define SC_DISPMOVE 0x08
- #define RL_LFTSHFT 0x00
- #define RL_RGTSHFT 0x04
- #define SETDISPFXN 0x20
- #define DL_4BIT 0x00
- #define DL_8BIT 0x10
- #define N_1LINE 0x00
- #define N_2LINE 0x08
- #define F_5X7DOT 0x00
- #define F_5X10DOT 0x04
- #define SETCGRAMADD 0x40
- #define SETDDRAMADD 0x80
- #define BF_READY 0x00
- #define BF_BUSY 1
- #define LINE1 0x00
- #define LINE2 0x40
- #define LINESTART 0x00
- #define LINEEND 0x0F

5.11.1 Macro Definition Documentation

```
5.11.1.1 #define B_BLINKOFF 0x00  
  
5.11.1.2 #define B_BLINKON 0x01  
  
5.11.1.3 #define BF_BUSY 1  
  
5.11.1.4 #define BF_READY 0x00  
  
5.11.1.5 #define C_CURSOFF 0x00  
  
5.11.1.6 #define C_CURSON 0x02  
  
5.11.1.7 #define D_DISPOFF 0x00  
  
5.11.1.8 #define D_DISPON 0x04  
  
5.11.1.9 #define DISPCLR 0x01  
  
5.11.1.10 #define DISPOPT 0x08  
  
5.11.1.11 #define DISPSHIFTCURS 0x10  
  
5.11.1.12 #define DL_4BIT 0x00  
  
5.11.1.13 #define DL_8BIT 0x10  
  
5.11.1.14 #define F_5X10DOT 0x04  
  
5.11.1.15 #define F_5X7DOT 0x00  
  
5.11.1.16 #define ID_CURSL 0x00  
  
5.11.1.17 #define ID_CURSR 0x02  
  
5.11.1.18 #define LCD_CLKHIGH 1  
  
5.11.1.19 #define LCD_CLKLOW 0  
  
5.11.1.20 #define LCD_D0 PORTDbits.RD0  
  
5.11.1.21 #define LCD_D0_DIR TRISDbits.RD0  
  
5.11.1.22 #define LCD_D1 PORTDbits.RD1  
  
5.11.1.23 #define LCD_D1_DIR TRISDbits.RD1  
  
5.11.1.24 #define LCD_D2 PORTDbits.RD2  
  
5.11.1.25 #define LCD_D2_DIR TRISDbits.RD2  
  
5.11.1.26 #define LCD_D3 PORTDbits.RD3  
  
5.11.1.27 #define LCD_D3_DIR TRISDbits.RD3
```

```
5.11.1.28 #define LCD_D4 PORTDbits.RD4
5.11.1.29 #define LCD_D4_DIR TRISDbits.RD4
5.11.1.30 #define LCD_D5 PORTDbits.RD5
5.11.1.31 #define LCD_D5_DIR TRISDbits.RD5
5.11.1.32 #define LCD_D6 PORTDbits.RD6
5.11.1.33 #define LCD_D6_DIR TRISDbits.RD6
5.11.1.34 #define LCD_D7 PORTDbits.RD7
5.11.1.35 #define LCD_D7_DIR TRISDbits.RD7
5.11.1.36 #define LCD_DATA 1
5.11.1.37 #define LCD_DATA_LINE PORTD
5.11.1.38 #define LCD_DATA_LINE_DIR TRISD
5.11.1.39 #define LCD_E PORTAbits.RA4
5.11.1.40 #define LCD_E_DIR TRISAbits.RA4
5.11.1.41 #define LCD_INPUT 0xFF
5.11.1.42 #define LCD_INS 0
5.11.1.43 #define LCD_OFF 0
5.11.1.44 #define LCD_ON 1
5.11.1.45 #define LCD_OUTPUT 0x00
5.11.1.46 #define LCD_PIN_INPUT 1
5.11.1.47 #define LCD_PIN_OUTPUT 0
5.11.1.48 #define LCD_READ 1
5.11.1.49 #define LCD_RS PORTCbits.RC4
5.11.1.50 #define LCD_RS_DIR TRISCbits.RC4
5.11.1.51 #define LCD_RW PORTCbits.RC5
5.11.1.52 #define LCD_RW_DIR TRISCbits.RC5
5.11.1.53 #define LCD_WRITE 0
5.11.1.54 #define LINE1 0x00
5.11.1.55 #define LINE2 0x40
```

```

5.11.1.56 #define LINEEND 0x0F
5.11.1.57 #define LINESTART 0x00
5.11.1.58 #define N_1LINE 0x00
5.11.1.59 #define N_2LINE 0x08
5.11.1.60 #define RL_LFTSHFT 0x00
5.11.1.61 #define RL_RGTSHFT 0x04
5.11.1.62 #define RTNHOME 0x02
5.11.1.63 #define SC_CURSMOVE 0x00
5.11.1.64 #define SC_DISPMOVE 0x08
5.11.1.65 #define SETCGRAMADD 0x40
5.11.1.66 #define SETDDRAMADD 0x80
5.11.1.67 #define SETDISPFXN 0x20
5.11.1.68 #define SETENTRYMODE 0x04
5.11.1.69 #define SH_DISPNSHIFT 0x00
5.11.1.70 #define SH_DISPSHIFT 0x01

```

5.12 Yavin4DefenceSystem/Code/Menu.Strings.h File Reference

Macros

- `#define STRINGS_H 0`

Variables

- `static const rom char newLine [] = "\r\n";`
- `static const rom char CHOOSE [] = "\n\n\tPlease select your option (eg. 2): \r\n";`
- `static const rom char menuPrefix3 [] = "\t3:";`
- `static const rom char menuPrefix4 [] = "\t4:";`
- `static const rom char menuPrefix5 [] = "\t5:";`
- `static const rom char menuPrefix8 [] = "\t8:";`
- `static const rom char goUp [] = "\tReturn to previous menu\r\n";`
- `static const rom char goUp2 [] = "Press Esc to return to the previous menu.\r\n";`
- `static const rom char errNumOutOfRange [] = " Error: Please enter a number between ";`
- `static const rom char and [] = " and ";`
- `static const rom char welcome [] = " Welcome to Yavin IV Defence System ";`
- `static const rom char welcomeLcd [] = " Welcome!";`
- `static const far rom char topOption1 [] = "\t1:\tAutomatic Tracking\r\n";`
- `static const far rom char topOption2 [] = "\t2:\tAzimuth Options\r\n";`
- `static const far rom char topOption3 [] = "\t3:\tElevation Options\r\n";`
- `static const far rom char topOption4 [] = "\t4:\tRange Options\r\n";`

- static const far rom char `topOption5` [] = "\t5:\tDisplay Current Temperature\r\n"
- static const far rom char `topOptionCalTemp` [] = "\t6:\tCalibrate the Temperature sensor\r\n"
- static const far rom char `topOptionLocal` [] = "\t6:\tSwitch to Local mode\r\n"
- static const far rom char `topOptionFactory` [] = "\t7:\tSwitch to Factory mode\r\n"
- static const far rom char `topOptionRemote` [] = "\t6:\tSwitch to Remote Serial mode\r\n"
- static const far rom char `topOptionRemoteLCD` [] = "Switch to Remote"
- static const rom char `azMenu` [] = "\r\nAzimuth Configuration\r\n"
- static const rom char `azMenuLcd` [] = "Azimuth Options"
- static const far rom char `azOption1` [] = "\t1:\tGo to a specified azimuth angle\r\n"
- static const far rom char `azOption2` [] = "\t2:\tSet the minimum azimuth angle\r\n"
- static const far rom char `azOption3` [] = "\t3:\tSet the maximum azimuth angle\r\n"
- static const far rom char `azOption4` [] = "\t4:\tCalibrate the azimuth motor\r\n"
- static const rom char `elMenu` [] = "\r\nElevation Configuration\r\n"
- static const rom char `elMenuLcd` [] = "Elevation Options"
- static const far rom char `elOption1` [] = "\t1:\tGo to a specified elevation angle\r\n"
- static const far rom char `elOption2` [] = "\t2:\tSet the minimum elevation angle\r\n"
- static const far rom char `elOption3` [] = "\t3:\tSet the maximum elevation angle\r\n"
- static const far rom char `elOption4` [] = "\t4:\tCalibrate the elevation motor\r\n"
- static const rom char `rngMenu` [] = "\r\nRange Configuration\r\n"
- static const rom char `rngMenuLcd` [] = "Range Options"
- static const far rom char `rngOption1` [] = "\t1:\tSet the minimum system range\r\n"
- static const far rom char `rngOption2` [] = "\t2:\tSet the maximum system range\r\n"
- static const far rom char `rngOption3` [] = "\t3:\tView the raw ultrasound **and** infrared sensor readings\r\n"
- static const far rom char `rngOption4` [] = "\t4:\tSet the ultrasound sample rate\r\n"
- static const far rom char `rngOption5` [] = "\t5:\tSet the number of ultrasound samples used per estimate\r\n"
- static const far rom char `rngOption6` [] = "\t6:\tSet the infrared sample rate\r\n"
- static const far rom char `rngOption7` [] = "\t7:\tSet the number of infrared samples used per estimate\r\n"
- static const far rom char `showTempLCDTitle` [] = "Display Temp"
- static const far rom char `showTempLCD` [] = "Temp = "
- static const far rom char `showTemp1` [] = "\r\n The current temperature is "
- static const far rom char `showTemp2` [] = " degrees Celcius. \r\n\r\n Press Esc to return.\r\n"
- static const far rom char `gotoAzAngle` [] = "\r\n Enter the azimuth angle in degrees. \r\n"
- static const far rom char `gotoAzAngleLCD` [] = "Go to Azimuth"
- static const far rom char `gotoElAngle` [] = "\r\n Enter the elevation angle in degrees. \r\n"
- static const far rom char `gotoELAngleLCD` [] = "Go to Elevation"
- static const far rom char `gotoAngle2` [] = " The current minimum **and** maximum angles are "
- static const far rom char `angleStr` [] = "Angle"
- static const far rom char `maxAzStr` [] = "Max Azimuth"
- static const far rom char `maxAzSetStr` [] = "Set Max Azimuth"
- static const far rom char `maxAz1` [] = " Enter a new maximum azimuth angle: \r\n"
- static const far rom char `maxAz3` [] = "\r\n Maximum azimuth angle **set** to "
- static const far rom char `currentMinAngleStr` [] = "\r\n The current minimum angle is "
- static const far rom char `minAzStr` [] = "Min Azimuth"
- static const far rom char `minAzSetStr` [] = "Set Min Azimuth"
- static const far rom char `minAz1` [] = " Enter a new minimum azimuth angle: \r\n"
- static const far rom char `minAz3` [] = "\r\n Minimum azimuth angle **set** to "
- static const far rom char `calibrateAngle1` [] = "\r\n Enter the true angle of the current position \r\n"
- static const far rom char `maxElStr` [] = "Max Elevation"
- static const far rom char `maxElSetStr` [] = "Set Max Elevation"
- static const far rom char `maxEl1` [] = " Enter a new maximum elevation angle: \r\n"
- static const far rom char `maxEl3` [] = "\r\n Maximum elevation angle **set** to "
- static const far rom char `minElStr` [] = "Min Elevation"
- static const far rom char `minElSetStr` [] = "Set Min Elevation"
- static const far rom char `minEl1` [] = " Enter a new minimum elevation angle: \r\n"
- static const far rom char `minEl3` [] = "\r\n Minimum elevation angle **set** to "

- static const far rom char `minRngStr` [] = "Min Range"
- static const far rom char `minRngSetStr` [] = "Set **Min** Range"
- static const far rom char `minRngSerialStr` [] = "Enter a new minimum range: \r\n"
- static const far rom char `maxRngStr` [] = "Max Range"
- static const far rom char `maxRngSetStr` [] = "Set **Max** Range"
- static const far rom char `maxRngSerialStr` [] = "Enter a new maximum range: \r\n"

5.12.1 Macro Definition Documentation

5.12.1.1 `#define STRINGS_H 0`

5.12.2 Variable Documentation

5.12.2.1 `const rom char and[] = " and "` [static]

5.12.2.2 `const far rom char angleStr[] = "Angle"` [static]

5.12.2.3 `const rom char azMenu[] = "\r\nAzimuth Configuration\r\n"` [static]

5.12.2.4 `const rom char azMenuLcd[] = "Azimuth Options"` [static]

5.12.2.5 `const far rom char azOption1[] = "\t1:\tGo to a specified azimuth angle\r\n"` [static]

5.12.2.6 `const far rom char azOption2[] = "\t2:\tSet the minimum azimuth angle\r\n"` [static]

5.12.2.7 `const far rom char azOption3[] = "\t3:\tSet the maximum azimuth angle\r\n"` [static]

5.12.2.8 `const far rom char azOption4[] = "\t4:\tCalibrate the azimuth motor\r\n"` [static]

5.12.2.9 `const far rom char calibrateAngle1[] = "\r\n Enter the true angle of the current position \r\n"` [static]

5.12.2.10 `const rom char CHOOSE[] = "\n\n\tPlease select your option (eg. 2):\r\n"` [static]

5.12.2.11 `const far rom char currentMinAngleStr[] = "\r\n The current minimum angle is "` [static]

5.12.2.12 `const rom char elMenu[] = "\r\nElevation Configuration\r\n"` [static]

5.12.2.13 `const rom char elMenuLcd[] = "Elevation Options"` [static]

5.12.2.14 `const far rom char elOption1[] = "\t1:\tGo to a specified elevation angle\r\n"` [static]

5.12.2.15 `const far rom char elOption2[] = "\t2:\tSet the minimum elevation angle\r\n"` [static]

5.12.2.16 `const far rom char elOption3[] = "\t3:\tSet the maximum elevation angle\r\n"` [static]

5.12.2.17 `const far rom char elOption4[] = "\t4:\tCalibrate the elevation motor\r\n"` [static]

5.12.2.18 `const rom char errNumOutOfRange[] = " Error: Please enter a number between "` [static]

5.12.2.19 `const far rom char gotoAngle2[] = " The current minimum and maximum angles are "` [static]

5.12.2.20 `const far rom char gotoAzAngle[] = "\r\n Enter the azimuth angle in degrees. \r\n"` [static]

5.12.2.21 `const far rom char gotoAzAngleLCD[] = "Go to Azimuth"` [static]

```
5.12.2.22 const far rom char gotoElAngle[] = "\r\n Enter the elevation angle in degrees. \r\n" [static]
5.12.2.23 const far rom char gotoELAngleLCD[] = "Go to Elevation" [static]
5.12.2.24 const rom char goUp[] = "\tReturn to previous menu\r\n" [static]
5.12.2.25 const rom char goUp2[] = "Press Esc to return to the previous menu.\r\n" [static]
5.12.2.26 const far rom char maxAz1[] = " Enter a new maximum azimuth angle: \r\n" [static]
5.12.2.27 const far rom char maxAz3[] = "\r\n Maximum azimuth angle set to " [static]
5.12.2.28 const far rom char maxAzSetStr[] = "Set Max Azimuth" [static]
5.12.2.29 const far rom char maxAzStr[] = "Max Azimuth" [static]
5.12.2.30 const far rom char maxEl1[] = " Enter a new maximum elevation angle: \r\n" [static]
5.12.2.31 const far rom char maxEl3[] = "\r\n Maximum elevation angle set to " [static]
5.12.2.32 const far rom char maxElSetStr[] = "Set Max Elevation" [static]
5.12.2.33 const far rom char maxElStr[] = "Max Elevation" [static]
5.12.2.34 const far rom char maxRngSerialStr[] = "Enter a new minimum range: \r\n" [static]
5.12.2.35 const far rom char maxRngSetStr[] = "Set Max Range" [static]
5.12.2.36 const far rom char maxRngStr[] = "Max Range" [static]
5.12.2.37 const rom char menuPrefix3[] = "\t3:" [static]
5.12.2.38 const rom char menuPrefix4[] = "\t4:" [static]
5.12.2.39 const rom char menuPrefix5[] = "\t5:" [static]
5.12.2.40 const rom char menuPrefix8[] = "\t8:" [static]
5.12.2.41 const far rom char minAz1[] = " Enter a new minimum azimuth angle: \r\n" [static]
5.12.2.42 const far rom char minAz3[] = "\r\n Minimum azimuth angle set to " [static]
5.12.2.43 const far rom char minAzSetStr[] = "Set Min Azimuth" [static]
5.12.2.44 const far rom char minAzStr[] = "Min Azimuth" [static]
5.12.2.45 const far rom char minEl1[] = " Enter a new minimum elevation angle: \r\n" [static]
5.12.2.46 const far rom char minEl3[] = "\r\n Minimum elevation angle set to " [static]
5.12.2.47 const far rom char minElSetStr[] = "Set Min Elevation" [static]
5.12.2.48 const far rom char minElStr[] = "Min Elevation" [static]
5.12.2.49 const far rom char minRngSerialStr[] = "Enter a new minimum range: \r\n" [static]
```

5.12.2.50 const far rom char minRngSetStr[] = "Set Min Range" [static]

5.12.2.51 const far rom char minRngStr[] = "Min Range" [static]

5.12.2.52 const rom char newLine[] = "\r\n" [static]

5.12.2.53 const rom char rngMenu[] = "\r\nRange Configuration\r\n" [static]

5.12.2.54 const rom char rngMenuLcd[] = "Range Options" [static]

5.12.2.55 const far rom char rngOption1[] = "\t1:\tSet the minimum system range\r\n" [static]

5.12.2.56 const far rom char rngOption2[] = "\t2:\tSet the maximum system range\r\n" [static]

5.12.2.57 const far rom char rngOption3[] = "\t3:\tView the raw ultrasound and infrared sensor readings\r\n" [static]

5.12.2.58 const far rom char rngOption4[] = "\t4:\tSet the ultrasound sample rate\r\n" [static]

5.12.2.59 const far rom char rngOption5[] = "\t5:\tSet the number of ultrasound samples used per estimate\r\n" [static]

5.12.2.60 const far rom char rngOption6[] = "\t6:\tSet the infrared sample rate\r\n" [static]

5.12.2.61 const far rom char rngOption7[] = "\t7:\tSet the number of infrared samples used per estimate\r\n" [static]

5.12.2.62 const far rom char showTemp1[] = "\r\n The current temperature is " [static]

5.12.2.63 const far rom char showTemp2[] = " degrees Celcius. \r\n\r\n Press Esc to return.\r\n" [static]

5.12.2.64 const far rom char showTempLCD[] = "Temp = " [static]

5.12.2.65 const far rom char showTempLCDTitle[] = "Display Temp" [static]

5.12.2.66 const far rom char topOption1[] = "\t1:\tAutomatic Tracking\r\n" [static]

5.12.2.67 const far rom char topOption2[] = "\t2:\tAzimuth Options\r\n" [static]

5.12.2.68 const far rom char topOption3[] = "\t3:\tElevation Options\r\n" [static]

5.12.2.69 const far rom char topOption4[] = "\t4:\tRange Options\r\n" [static]

5.12.2.70 const far rom char topOption5[] = "\t5:\tDisplay Current Temperature\r\n" [static]

5.12.2.71 const far rom char topOptionCalTemp[] = "\t6:\tCalibrate the Temperature sensor\r\n" [static]

5.12.2.72 const far rom char topOptionFactory[] = "\t7:\tSwitch to Factory mode\r\n" [static]

5.12.2.73 const far rom char topOptionLocal[] = "\t6:\tSwitch to Local mode\r\n" [static]

5.12.2.74 const far rom char topOptionRemote[] = "\t6:\tSwitch to Remote Serial mode\r\n" [static]

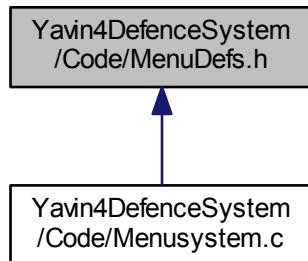
5.12.2.75 const far rom char topOptionRemoteLCD[] = "Switch to Remote" [static]

5.12.2.76 const rom char welcome[] = " Welcome to Yavin IV Defence System " [static]

5.12.2.77 const rom char welcomeLcd[] = " Welcome!" [static]

5.13 Yavin4DefenceSystem/Code/MenuDefs.h File Reference

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



Data Structures

- struct `menuStruct`
Defines a sub menu.

Macros

- #define FACTORY_SWITCH PORTBbits.RB2
- #define MENU_ISR 0
- #define width 80
- #define height 24
- #define MAX_ANGLE_SUPREMUM 45
- #define MAX_ANGLE_INFIMUM 20
- #define MIN_ANGLE_SUPREMUM -20
- #define MIN_ANGLE_INFIMUM -45
- #define MAX_RANGE_SUPREMUM 3000
- #define MAX_RANGE_INFIMUM 1000
- #define MIN_RANGE_SUPREMUM 600
- #define MIN_RANGE_INFIMUM 300
- #define SAMPLE_PER_AVG_MIN 1
- #define SAMPLE_PER_AVG_MAX 5
- #define CLEAR_SCREEN_STRING "\033[2J\033[0;0H"
- #define CLEAR_LINE_STRING "\033[2K"
- #define SERIAL_CURSOR_OFF "\033[?25l"
- #define SERIAL_CURSOR_ON "\033[?25h"
- #define ERR_NUM_OUT_OF_RANGE -1000
- #define ERR_NOT_NUMERIC -2000
- #define ERR_NO_NUMBER -3000
- #define ESC_PRESSED -4000
- #define MINUS_CHAR 0x2D

Typedefs

- `typedef void(* numericInputFunction)(int)`
- `typedef void(* voidFunction)(void)`

Enumerations

- `enum menuState {
TOP_LEVEL, TRACKING, AZ_MENU, AZ_GOTO,
AZ_MAX, AZ_MIN, AZ_CALIBRATE, EL_MENU,
EL_GOTO, EL_MAX, EL_MIN, EL_CALIBRATE,
RANGE_MENU, RANGE_MAX, RANGE_MIN, US_SAMPLE_RATE,
US_SAMPLE_AVG, IR_SAMPLE_RATE, IR_SAMPLE_AVG, RANGE_RAW,
CALIBRATE_RANGE, SHOW_TEMP, CALIBRATE_TEMP }`
- `enum userState { LOCAL, REMOTE, FACTORY }`

Variables

- `struct menuStruct menuStruct`
- `const rom char newLine [] = "\r\n"`
- `const rom char CHOOSE [] = "\tPlease select option (eg. 2)"`
- `const rom char CHOOSE2 [] = "\tPress Enter to confirm, or Esc to return"`
- `const rom char menuPrefix3 [] = "3:"`
- `const rom char menuPrefix4 [] = "4:"`
- `const rom char menuPrefix5 [] = "5:"`
- `const rom char menuPrefix8 [] = "8:"`
- `const rom char goUp [] = "Return to previous menu"`
- `const rom char goUp2 [] = "Press Esc to return to the previous menu."`
- `const rom char errStr [] = "Error: "`
- `const rom char inputNumRangeStr [] = "Please enter a number between "`
- `const rom char currentValueStr [] = "The current value is: "`
- `const rom char and [] = " and "`
- `const rom char AzStr [] = "Azimuth"`
- `const rom char ElStr [] = "Elevation"`
- `const rom char RngStr [] = "Range"`
- `const rom char mmStr [] = "Range (mm)"`
- `const rom char sampleRate [] = "Sampling Frequency (Hz)"`
- `const rom char numPerSample [] = "Number of samples per estimate"`
- `const rom char hz [] = "Hz"`
- `const rom char tab [] = "\t"`
- `const rom char title [] = "Welcome to the Yavin IV Defence System!"`
- `const rom char welcomeLcd [] = "Home Menu"`
- `const rom char topOption1 [] = "\t1:\tAutomatic Tracking"`
- `const rom char topOption2 [] = "\t2:\tAzimuth Options"`
- `const rom char topOption3 [] = "\t3:\tElevation Options"`
- `const rom char topOption4 [] = "\t4:\tRange Options"`
- `const rom char topOption5 [] = "\t5:\tDisplay Current Temperature"`
- `const rom char topOptionCalTemp [] = "\t6:\tCalibrate the Temperature sensor"`
- `const rom char topOptionLocal [] = "\t6:\tSwitch to Local mode"`
- `const rom char topOptionForFactory [] = "\t7:\tSwitch to Factory mode"`
- `const rom char topOptionRemote [] = "\t6:\tSwitch to Remote Serial mode"`
- `const rom char topOptionRemoteLCD [] = "Switch to Remote"`
- `const rom char autoSerialMessage [] = "Automatic tracking mode initiated."`
- `const rom char autoLcdTitle [] = "Auto-tracking"`

- const rom char `autoSearching` [] = "Searching..."
- const rom char `set` [] = "Set"
- const rom char `range_str` [] = "Range"
- const rom char `goto_str` [] = "Go to"
- const rom char `Options` [] = "Options"
- const rom char `azMenu` [] = "\r\nAzimuth Configuration\r\n"
- const rom char `azMenuLcd` [] = "Azimuth Options"
- const rom char `azOption1` [] = "\t1:\tGo to a specified azimuth angle"
- const rom char `azOption2` [] = "\t2:\tSet the min azimuth angle"
- const rom char `azOption3` [] = "\t3:\tSet the max azimuth angle"
- const rom char `azOption4` [] = "\t4:\tCalibrate the azimuth motor"
- const rom char `eIMenu` [] = "Elevation Configuration"
- const rom char `eIMenuLcd` [] = "Elevation Options"
- const rom char `eIOption1` [] = "\t1:\tGo to a specified elevation angle"
- const rom char `eIOption2` [] = "\t2:\tSet min elevation angle"
- const rom char `eIOption3` [] = "\t3:\tSet max elevation angle"
- const rom char `eIOption4` [] = "\t4:\tCalibrate elevation servo"
- const rom char `rngMenu` [] = "Range Configuration"
- const rom char `rngMenuLcd` [] = "Range Options"
- const rom char `rngOption1` [] = "\t1:\tSet min system range"
- const rom char `rngOption2` [] = "\t2:\tSet max system range"
- const rom char `rngOption3` [] = "\t3:\tView raw ultrasound and IR sensor data"
- const rom char `rngOption4` [] = "\t4:\tSet the ultrasound sample rate"
- const rom char `rngOption5` [] = "\t5:\tSet number of ultrasound samples per estimate"
- const rom char `rngOption7` [] = "\t6:\tSet number of IR samples per estimate"
- const rom char `rngOption8` [] = "\t7:\tCalibrate to a set range"
- const rom char `showTempLCDTitle` [] = "Display Temp"
- const rom char `showTempLCD` [] = "Temp = "
- const rom char `showTemp1` [] = "Temp: "
- const rom char `showTemp2` [] = " deg Celcius. \r\n\r\n\tPress Esc to return"
- const rom char `gotoAzAngle` [] = "Enter azimuth angle in degrees"
- const rom char `gotoAzAngleLCD` [] = "Go to Azimuth"
- const rom char `gotoElAngle` [] = "Enter elevation angle in degrees"
- const rom char `gotoELAngleLCD` [] = "Go to Elevation"
- const rom char `gotoAngle2` [] = " Current min & max angles: "
- const rom char `angleStr` [] = "Angle"
- const rom char `maxAzStr` [] = "Max Azimuth"
- const rom char `maxAzSetStr` [] = "Set Max Azimuth"
- const rom char `maxAz1` [] = "Enter a new max azimuth: "
- const rom char `maxAz3` [] = "Max azimuth set to "
- const rom char `currentMinAngleStr` [] = "Current min "
- const rom char `minAzStr` [] = "Min Azimuth"
- const rom char `minAzSetStr` [] = "Set Min Azimuth"
- const rom char `minAz1` [] = "Enter a new min azimuth: "
- const rom char `minAz3` [] = "Min azimuth set to "
- const rom char `calibrateAngle1` [] = "Enter Calibration angles: "
- const rom char `maxElStr` [] = "Max Elevation"
- const rom char `maxElSetStr` [] = "Set Max Elevation"
- const rom char `maxEl1` [] = "Enter a new max elevation: "
- const rom char `maxEl3` [] = "Max elevation set to "
- const rom char `minElStr` [] = "Min Elevation"
- const rom char `minElSetStr` [] = "Set Min Elevation"
- const rom char `minEl1` [] = "Enter a new min elevation: "
- const rom char `minEl3` [] = "Min elevation set to "
- const rom char `minRngStr` [] = "Min Range"

- const rom char `minRngSetStr` [] = "Set Min Range"
- const rom char `minRngSerialStr` [] = "Enter a new min range: (mm)"
- const rom char `maxRngStr` [] = "Max Range"
- const rom char `maxRngSetStr` [] = "Set Max Range"
- const rom char `maxRngSerialStr` [] = "Enter a new max `range` (mm):"
- const rom char `rawRangeStr` [] = "Displaying raw `range` in mm:"
- const rom char `rawRangeTitle` [] = "Raw Range"
- const rom char `calRangeStr` [] = "Place an object 500mm away from the sensor and press enter."
- const rom char `calRangeTitle` [] = "Calibrate Range"
- const rom char `calRangeConfirm` [] = "Range calibrated"
- const rom char `usSampleRateStr` [] = "Please enter a sample"
- const rom char `usSampleRateTitle` [] = "Ultrasound Sample Rate"
- const rom char `numSamplesStr` [] = "Enter a number of samples to take."
- const rom char `numSampleTitle` [] = "Num. of Samples"

5.13.1 Macro Definition Documentation

```

5.13.1.1 #define CLEAR_LINE_STRING "\033[2K"

5.13.1.2 #define CLEAR_SCREEN_STRING "\033[2J\033[0;0H"

5.13.1.3 #define ERR_NO_NUMBER -3000

5.13.1.4 #define ERR_NOT_NUMERIC -2000

5.13.1.5 #define ERR_NUM_OUT_OF_RANGE -1000

5.13.1.6 #define ESC_PRESSED -4000

5.13.1.7 #define FACTORY_SWITCH PORTBbits.RB2

5.13.1.8 #define height 24

5.13.1.9 #define MAX_ANGLE_INFIMUM 20

5.13.1.10 #define MAX_ANGLE_SUPREMUM 45

5.13.1.11 #define MAX_RANGE_INFIMUM 1000

5.13.1.12 #define MAX_RANGE_SUPREMUM 3000

5.13.1.13 #define MENU_ISR 0

5.13.1.14 #define MIN_ANGLE_INFIMUM -45

5.13.1.15 #define MIN_ANGLE_SUPREMUM -20

5.13.1.16 #define MIN_RANGE_INFIMUM 300

5.13.1.17 #define MIN_RANGE_SUPREMUM 600

5.13.1.18 #define MINUS_CHAR 0x2D

5.13.1.19 #define SAMPLE_PER_AVG_MAX 5

```

```
5.13.1.20 #define SAMPLE_PER_AVG_MIN 1  
5.13.1.21 #define SERIAL_CURSOR_OFF "\033[?25l"  
5.13.1.22 #define SERIAL_CURSOR_ON "\033[?25h"  
5.13.1.23 #define width 80
```

5.13.2 Typedef Documentation

```
5.13.2.1 typedef void(* numericInputFunction)(int)  
5.13.2.2 typedef void(* voidFunction)(void)
```

5.13.3 Enumeration Type Documentation

5.13.3.1 enum menuState

Enumerator

TOP_LEVEL
TRACKING
AZ_MENU
AZ_GOTO
AZ_MAX
AZ_MIN
AZ_CALIBRATE
EL_MENU
EL_GOTO
EL_MAX
EL_MIN
EL_CALIBRATE
RANGE_MENU
RANGE_MAX
RANGE_MIN
US_SAMPLE_RATE
US_SAMPLE_AVG
IR_SAMPLE_RATE
IR_SAMPLE_AVG
RANGE_RAW
CALIBRATE_RANGE
SHOW_TEMP
CALIBRATE_TEMP

5.13.3.2 enum userState

Enumerator

LOCAL
REMOTE
FACTORY

5.13.4 Variable Documentation

- 5.13.4.1 const rom char and[] = " and "
- 5.13.4.2 const rom char angleStr[] = "Angle"
- 5.13.4.3 const rom char autoLcdTitle[] = "Auto-tracking"
- 5.13.4.4 const rom char autoSearching[] = "Searching..."
- 5.13.4.5 const rom char autoSerialMessage[] = "Automatic tracking mode initiated."
- 5.13.4.6 const rom char azMenu[] = "\r\nAzimuth Configuration\r\n"
- 5.13.4.7 const rom char azMenuLcd[] = "Azimuth Options"
- 5.13.4.8 const rom char azOption1[] = "\t1:\tGo to a specified azimuth angle"
- 5.13.4.9 const rom char azOption2[] = "\t2:\tSet the min azimuth angle"
- 5.13.4.10 const rom char azOption3[] = "\t3:\tSet the max azimuth angle"
- 5.13.4.11 const rom char azOption4[] = "\t4:\tCalibrate the azimuth motor"
- 5.13.4.12 const rom char AzStr[] = "Azimuth"
- 5.13.4.13 const rom char calibrateAngle1[] = "Enter Calibration angles: "
- 5.13.4.14 const rom char calRangeConfirm[] = "Range calibrated"
- 5.13.4.15 const rom char calRangeStr[] = "Place an object 500mm away from the sensor and press enter."
- 5.13.4.16 const rom char calRangeTitle[] = "Calibrate Range"
- 5.13.4.17 const rom char CHOOSE[] = "\tPlease select option (eg. 2)"
- 5.13.4.18 const rom char CHOOSE2[] = "\tPress Enter to confirm, or Esc to return"
- 5.13.4.19 const rom char currentMinAngleStr[] = "Current min "
- 5.13.4.20 const rom char currentValueStr[] = "The current value is: "
- 5.13.4.21 const rom char elMenu[] = "Elevation Configuration"
- 5.13.4.22 const rom char elMenuLcd[] = "Elevation Options"
- 5.13.4.23 const rom char elOption1[] = "\t1:\tGo to a specified elevation angle"
- 5.13.4.24 const rom char elOption2[] = "\t2:\tSet min elevation angle"
- 5.13.4.25 const rom char elOption3[] = "\t3:\tSet max elevation angle"
- 5.13.4.26 const rom char elOption4[] = "\t4:\tCalibrate elevation servo"
- 5.13.4.27 const rom char ElStr[] = "Elevation"

```
5.13.4.28 const rom char errStr[] = "Error: "  
5.13.4.29 const rom char goto_str[] = "Go to"  
5.13.4.30 const rom char gotoAngle2[] = " Current min & max angles: "  
5.13.4.31 const rom char gotoAzAngle[] = "Enter azimuth angle in degrees"  
5.13.4.32 const rom char gotoAzAngleLCD[] = "Go to Azimuth"  
5.13.4.33 const rom char gotoElAngle[] = "Enter elevation angle in degrees"  
5.13.4.34 const rom char gotoELAngleLCD[] = "Go to Elevation"  
5.13.4.35 const rom char goUp[] = "Return to previous menu"  
5.13.4.36 const rom char goUp2[] = "Press Esc to return to the previous menu."  
5.13.4.37 const rom char hz[] = "Hz"  
5.13.4.38 const rom char inputNumRangeStr[] = "Please enter a number between "  
5.13.4.39 const rom char maxAz1[] = "Enter a new max azimuth: "  
5.13.4.40 const rom char maxAz3[] = "Max azimuth set to "  
5.13.4.41 const rom char maxAzSetStr[] = "Set Max Azimuth"  
5.13.4.42 const rom char maxAzStr[] = "Max Azimuth"  
5.13.4.43 const rom char maxEl1[] = "Enter a new max elevation: "  
5.13.4.44 const rom char maxEl3[] = "Max elevation set to "  
5.13.4.45 const rom char maxElSetStr[] = "Set Max Elevation"  
5.13.4.46 const rom char maxElStr[] = "Max Elevation"  
5.13.4.47 const rom char maxRngSerialStr[] = "Enter a new max range (mm):"  
5.13.4.48 const rom char maxRngSetStr[] = "Set Max Range"  
5.13.4.49 const rom char maxRngStr[] = "Max Range"  
5.13.4.50 const rom char menuPrefix3[] = "3:"  
5.13.4.51 const rom char menuPrefix4[] = "4:"  
5.13.4.52 const rom char menuPrefix5[] = "5:"  
5.13.4.53 const rom char menuPrefix8[] = "8:"  
5.13.4.54 struct menuStruct menuStruct  
5.13.4.55 const rom char minAz1[] = "Enter a new min azimuth: "
```

5.13.4.56 const rom char minAz3[] = "Min azimuth set to "

5.13.4.57 const rom char minAzSetStr[] = "Set Min Azimuth"

5.13.4.58 const rom char minAzStr[] = "Min Azimuth"

5.13.4.59 const rom char minEl1[] = "Enter a new min elevation: "

5.13.4.60 const rom char minEl3[] = "Min elevation set to "

5.13.4.61 const rom char minElSetStr[] = "Set Min Elevation"

5.13.4.62 const rom char minElStr[] = "Min Elevation"

5.13.4.63 const rom char minRngSerialStr[] = "Enter a new min range: (mm)"

5.13.4.64 const rom char minRngSetStr[] = "Set Min Range"

5.13.4.65 const rom char minRngStr[] = "Min Range"

5.13.4.66 const rom char mmStr[] = "Range (mm)"

5.13.4.67 const rom char newLine[] = "\r\n"

5.13.4.68 const rom char numPerSample[] = "Number of samples per estimate"

5.13.4.69 const rom char numSamplesStr[] = "Enter a number of samples to take."

5.13.4.70 const rom char numSampleTitle[] = "Num. of Samples"

5.13.4.71 const rom char Options[] = "Options"

5.13.4.72 const rom char range_str[] = "Range"

5.13.4.73 const rom char rawRangeStr[] = "Displaying raw range in mm:"

5.13.4.74 const rom char rawRangeTitle[] = "Raw Range"

5.13.4.75 const rom char rngMenu[] = "Range Configuration"

5.13.4.76 const rom char rngMenuLcd[] = "Range Options"

5.13.4.77 const rom char rngOption1[] = "\t1:\tSet min system range"

5.13.4.78 const rom char rngOption2[] = "\t2:\tSet max system range"

5.13.4.79 const rom char rngOption3[] = "\t3:\tView raw ultrasound and IR sensor data"

5.13.4.80 const rom char rngOption4[] = "\t4:\tSet the ultrasound sample rate"

5.13.4.81 const rom char rngOption5[] = "\t5:\tSet number of ultrasound samples per estimate"

5.13.4.82 const rom char rngOption7[] = "\t6:\tSet number of IR samples per estimate"

5.13.4.83 const rom char rngOption8[] = "\t7:\tCalibrate to a set range"

5.13.4.84 const rom char RngStr[] = "Range"

5.13.4.85 const rom char sampleRate[] = "Sampling Frequency (Hz)"

5.13.4.86 const rom char set[] = "Set"

5.13.4.87 const rom char showTemp1[] = "Temp: "

5.13.4.88 const rom char showTemp2[] = " deg Celcius. \r\n\n\tPress Esc to return"

5.13.4.89 const rom char showTempLCD[] = "Temp = "

5.13.4.90 const rom char showTempLCDTitle[] = "Display Temp"

5.13.4.91 const rom char tab[] = "\t"

5.13.4.92 const rom char title[] = "Welcome to the Yavin IV Defence System!"

5.13.4.93 const rom char topOption1[] = "\t1:\tAutomatic Tracking"

5.13.4.94 const rom char topOption2[] = "\t2:\tAzimuth Options"

5.13.4.95 const rom char topOption3[] = "\t3:\tElevation Options"

5.13.4.96 const rom char topOption4[] = "\t4:\tRange Options"

5.13.4.97 const rom char topOption5[] = "\t5:\tDisplay Current Temperature"

5.13.4.98 const rom char topOptionCalTemp[] = "\t6:\tCalibrate the Temperature sensor"

5.13.4.99 const rom char topOptionForFactory[] = "\t7:\tSwitch to Factory mode"

5.13.4.100 const rom char topOptionLocal[] = "\t6:\tSwitch to Local mode"

5.13.4.101 const rom char topOptionRemote[] = "\t6:\tSwitch to Remote Serial mode"

5.13.4.102 const rom char topOptionRemoteLCD[] = "Switch to Remote"

5.13.4.103 const rom char usSampleRateStr[] = "Please enter a sample"

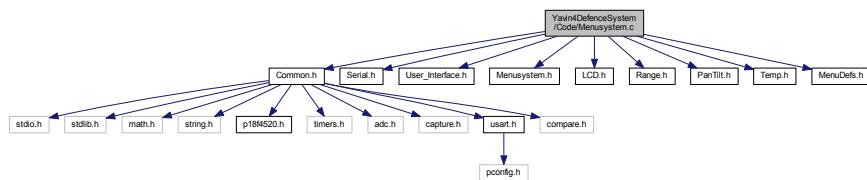
5.13.4.104 const rom char usSampleRateTitle[] = "Ultrasound Sample Rate"

5.13.4.105 const rom char welcomeLcd[] = "Home Menu"

5.14 Yavin4DefenceSystem/Code/Menusystem.c File Reference

```
#include "Common.h"
#include "Serial.h"
#include "User_Interface.h"
#include "Menusystem.h"
#include "LCD.h"
#include "Range.h"
#include "PanTilt.h"
#include "Temp.h"
#include "MenuDefs.h"
```

Include dependency graph for Menusystem.c:



Functions

- static void **sendROM** (const rom char *romchar)

Include: Local to Menusystem.c.
- static void **clearScreen** ()

Clears the Serial Display.
- static void **send.NewLine** (char length)

Include: Local to Menusystem.c.
- static void **filler** (char length)
- static void **errOutOfRange** (int lowerBound, int upperBound)
- static char * **intToAscii** (int num)
- static void **autodisp** (void)
- int **parseNumeric** (char *number)

parses user input string into a number
- static int **getSerialNumericInput** ()

Waits for input and parses number.
- static void **configureTimer0** (void)
- static int **getLocalInputMenu** (int maxStates, int(*function)(int))
- static void **setValue** (int input)
- static void **setMenu** (struct menuStruct menu)
- static void **navigateTopMenu** (int inputResult)
- static void **noFunctionNumeric** (int input)
- static void **setCalibrateRange** (int num)
- static void **noFunction** (void)
- static void **returnToTopMenu** (void)
- static void **exitFromTracking** (void)
- static void **returnToAzMenu** (void)
- static void **returnToElMenu** (void)
- static void **returnToRngMenu** (void)
- static void **navigateAzimuthMenu** (int inputResult)
- static void **navigateElevationMenu** (int inputResult)
- static void **navigateRangeMenu** (int inputResult)
- static void **dispTopOptions** (void)
- static void **displayMenuSerial** ()
- static void **dispAzOptions** (void)
- static void **dispElOptions** (void)
- static void **dispRngOptions** (void)
- static void **dispTempSerialMessage** (void)
- static void **dispSerialMessage** (void)
- static void **dispSetValueMessage** (void)
- static void **dispLCDTopMenu** (int option)

LCD Display Functions.

- static void `dispLCDAAzMenu` (int option)
- static void `dispLCDEIMenu` (int option)
- static void `dispLCDRngMenu` (int option)
- static void `dispLCDNum` (int option)
- void `initialiseMenu` (`systemState` *state)

Initialises the menu system.
- char `checkForSerialInput` (void)
- void `serviceMenu` (void)

services any user interface with the menu
- void `dispTrack` (`TrackingData` target)
- void `dispRawRange` ()
- void `dispSearching` ()

Variables

- struct `menuStruct topMenu` = {`TOP_LEVEL`, title, welcomeLcd, 1, 6, 1, `dispTopOptions`, `navigateTopMenu`, 0, `noFunction`}
- struct `menuStruct AzMenu` = {`AZ_MENU`, azMenu, azMenuLcd, 1, 6, 1, `dispAzOptions`, `navigateAzimuthMenu`, 0, `returnToTopMenu`}
- struct `menuStruct ElMenu` = {`EL_MENU`, elMenu, elMenuLcd, 1, 6, 1, `dispElOptions`, `navigateElevationMenu`, 0, `returnToTopMenu`}
- struct `menuStruct RangeMenu` = {`RANGE_MENU`, rngMenu, rngMenuLcd, 1, 6, 1, `dispRngOptions`, `navigateRangeMenu`, 0, `returnToTopMenu`}
- struct `menuStruct AzGoto` = {`AZ_GOTO`, gotoAzAngle, gotoAzAngleLCD, `MIN_ANGLE_INFIMUM`, `MAX_ANGLE_SUPREMUM`, 5, `dispSetValueMessage`, `setValue`, `noFunctionNumeric`, `returnToAzMenu`}

Remote/Local Functions.

- struct `menuStruct ElGoto` = {`EL_GOTO`, gotoElAngle, gotoELAngleLCD, `MIN_ANGLE_INFIMUM`, `MAX_ANGLE_SUPREMUM`, 5, `dispSetValueMessage`, `setValue`, `noFunctionNumeric`, `returnToElMenu`}
- struct `menuStruct AzMin` = {`AZ_MIN`, minAz1, minAzSetStr, `MIN_ANGLE_INFIMUM`, `MIN_ANGLE_SUPREMUM`, 5, `dispSetValueMessage`, `setValue`, `noFunctionNumeric`, `returnToAzMenu`}
- struct `menuStruct AzMax` = {`AZ_MAX`, maxAz1, maxAzSetStr, `MAX_ANGLE_INFIMUM`, `MAX_ANGLE_SUPREMUM`, 5, `dispSetValueMessage`, `setValue`, `noFunctionNumeric`, `returnToAzMenu`}
- struct `menuStruct ElMin` = {`EL_MIN`, minEl1, minElSetStr, `MIN_ANGLE_INFIMUM`, `MIN_ANGLE_SUPREMUM`, 5, `dispSetValueMessage`, `setValue`, `noFunctionNumeric`, `returnToElMenu`}
- struct `menuStruct ElMax` = {`EL_MAX`, maxEl1, maxElSetStr, `MAX_ANGLE_INFIMUM`, `MAX_ANGLE_SUPREMUM`, 5, `dispSetValueMessage`, `setValue`, `noFunctionNumeric`, `returnToElMenu`}
- struct `menuStruct RngMin` = {`RANGE_MIN`, minRngSerialStr, minRngSetStr, `MIN_RANGE_INFIMUM`, `MIN_RANGE_SUPREMUM`, 50, `dispSetValueMessage`, `setValue`, `noFunctionNumeric`, `returnToRngMenu`}
- struct `menuStruct RngMax` = {`RANGE_MAX`, maxRngSerialStr, maxRngSetStr, `MAX_RANGE_INFIMUM`, `MAX_RANGE_SUPREMUM`, 50, `dispSetValueMessage`, `setValue`, `noFunctionNumeric`, `returnToRngMenu`}
- struct `menuStruct ShowTemp` = {`SHOW_TEMP`, showTempLCDTitle, showTempLCDTitle, 0, 0, 0, `dispTempSerialMessage`, `noFunctionNumeric`, `noFunctionNumeric`, `returnToTopMenu`}
- struct `menuStruct Tracking` = {`TRACKING`, autoSerialMessage, autoLcdTitle, 0, 0, 0, `dispSerialMessage`, `noFunctionNumeric`, `noFunctionNumeric`, `exitFromTracking`}
- struct `menuStruct RawRange`
- struct `menuStruct NumSamples`
- struct `menuStruct UsSampleRate`
- struct `menuStruct calibrateRangeMenu`
- struct `menuStruct m_currentMenu`
- static `userState m_userMode`
- static `systemState * m_trackingState`

5.14.1 Function Documentation

5.14.1.1 static void autodisp(void) [static]

5.14.1.2 char checkForSerialInput(void)

Function: [checkForSerialInput\(void\)](#)

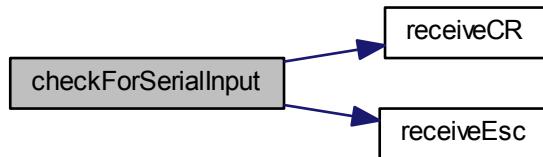
Include: Menusystem

Description: Checks the serial/local buffers for inputs

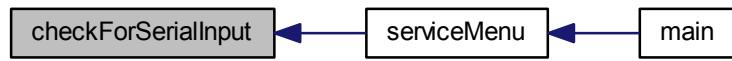
Arguments: None

Returns: 1 if input has been received, 0 otherwise Wait until the receive buffer is no longer empty Indicating that a command has been passed

Here is the call graph for this function:



Here is the caller graph for this function:



5.14.1.3 static void clearScreen() [static]

Clears the Serial Display.

Clears the Serial display

Function: [clearScreen\(void\)](#)

Include: Local to [Menusystem.c](#)

: Transmits the given string from ROM over serial

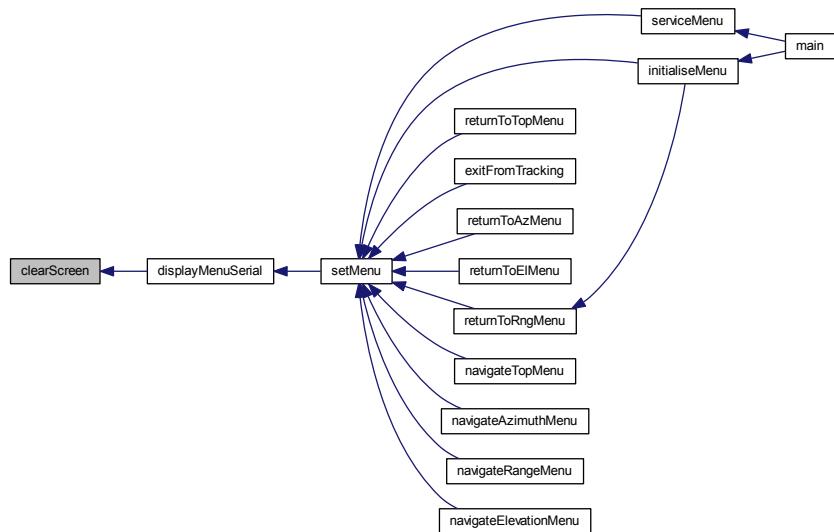
The string to transmit

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:

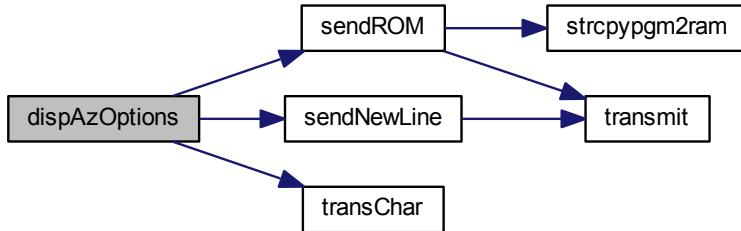


5.14.1.4 static void configureTimer0(void) [static]

5.14.1.5 static void dispAzOptions() [static]

Display the user options for the Azimuth menu

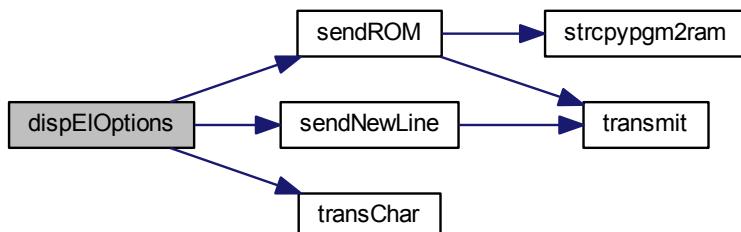
Here is the call graph for this function:



5.14.1.6 static void dispElOptions() [static]

Display the user options for the Azimuth menu

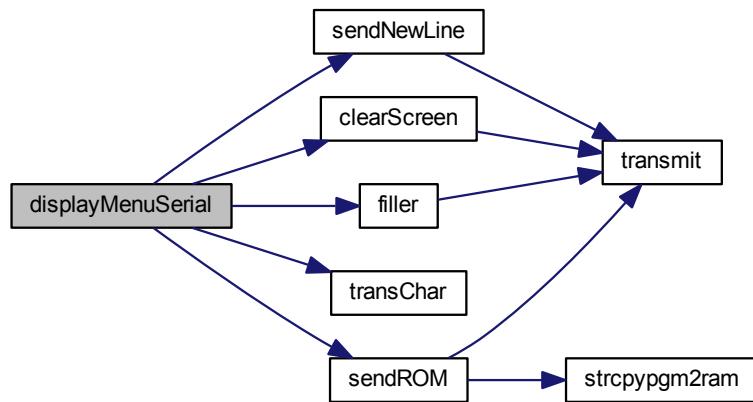
Here is the call graph for this function:



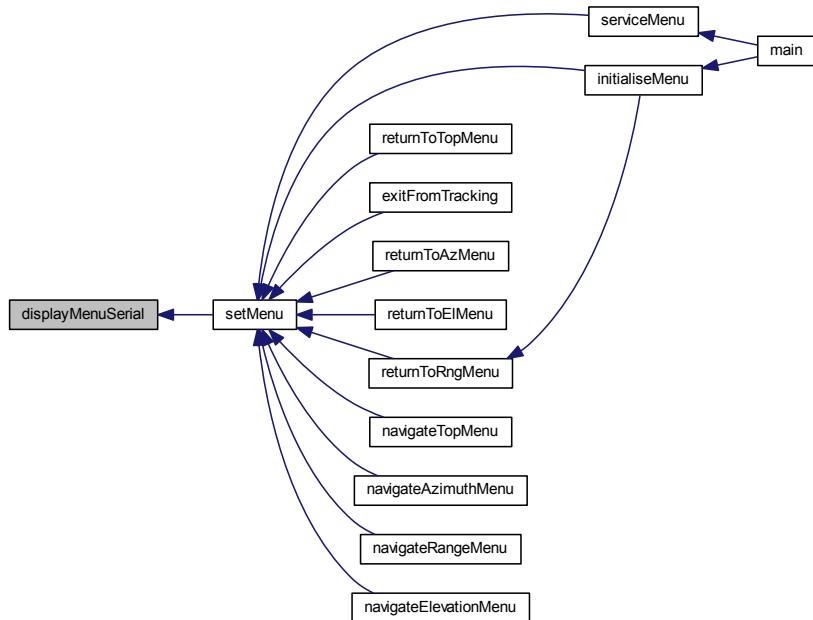
5.14.1.7 static void displayMenuSerial() [static]

Display the current menu Title and other information over serial

Here is the call graph for this function:



Here is the caller graph for this function:



5.14.1.8 static void dispLCDAzMenu (int option) [static]

5.14.1.9 static void dispLCDElMenu (int option) [static]

5.14.1.10 static void dispLCDNum (int option) [static]

5.14.1.11 static void dispLCDRngMenu (int *option*) [static]

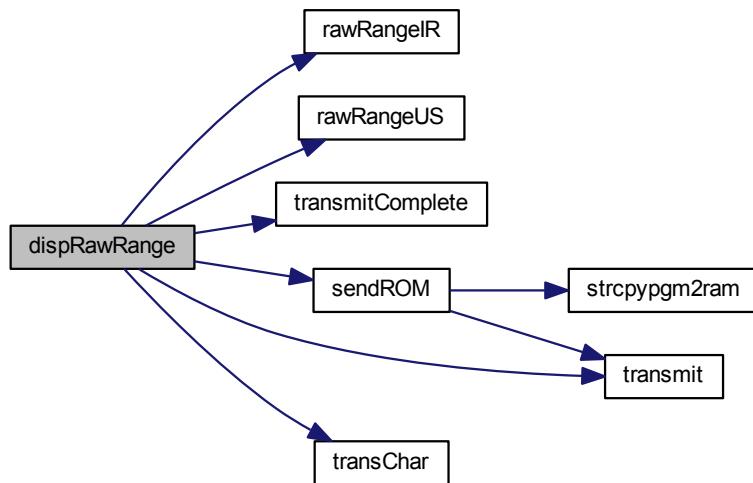
5.14.1.12 static void dispLCDTopMenu (int *option*) [static]

LCD Display Functions.

5.14.1.13 void dispRawRange (void)

Displays the Tracking data over serial

Here is the call graph for this function:



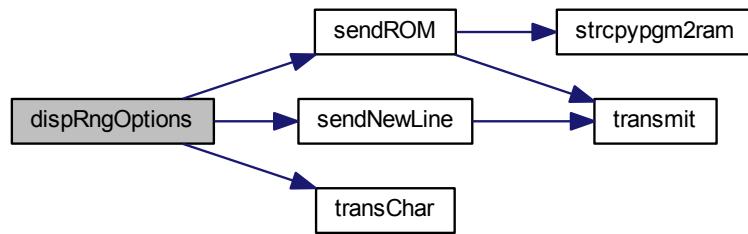
Here is the caller graph for this function:



5.14.1.14 static void dispRngOptions () [static]

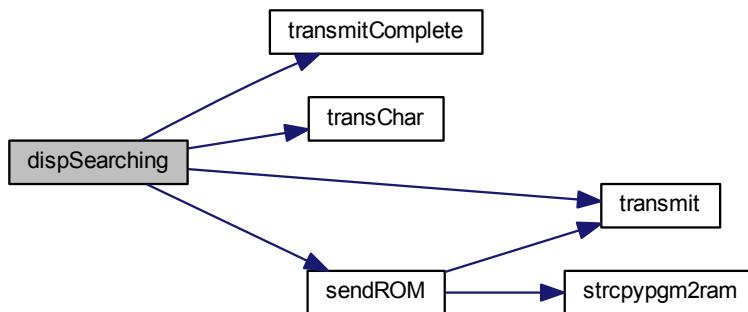
Display the user options for the Azimuth menu

Here is the call graph for this function:



5.14.1.15 void dispSearching (void)

Here is the call graph for this function:



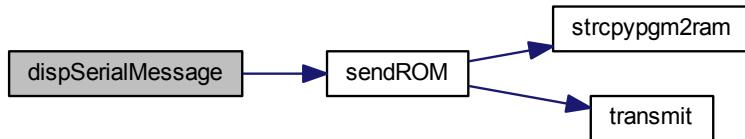
Here is the caller graph for this function:



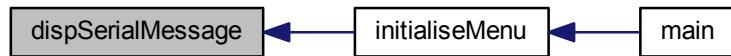
5.14.1.16 static void dispSerialMessage (void) [static]

Display the menu serial message

Here is the call graph for this function:



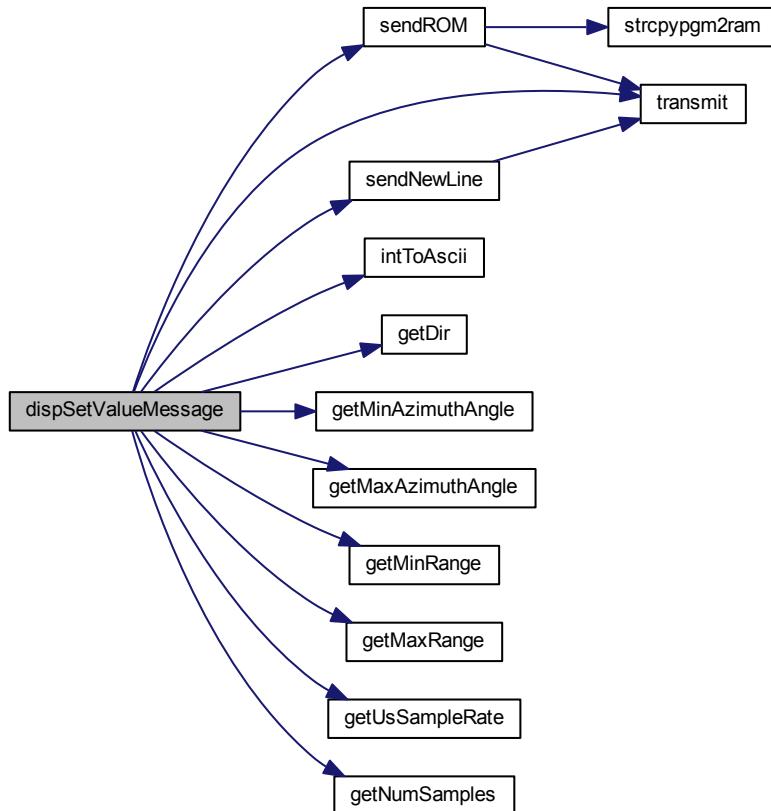
Here is the caller graph for this function:



5.14.1.17 static void dispSetValueMessage (void) [static]

Display the menu serial message with the maximum, minimum and current values

Here is the call graph for this function:



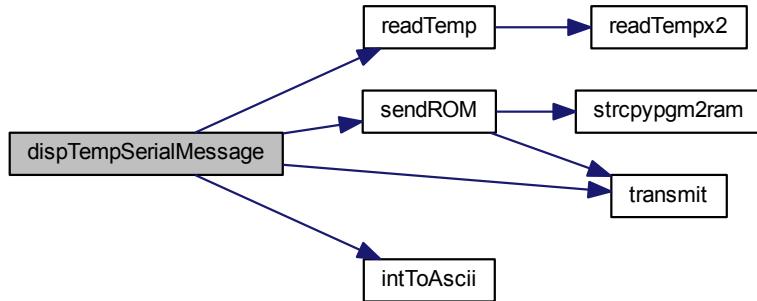
Here is the caller graph for this function:



5.14.1.18 static void dispTempSerialMessage(void) [static]

Display the Show Temperature serial message

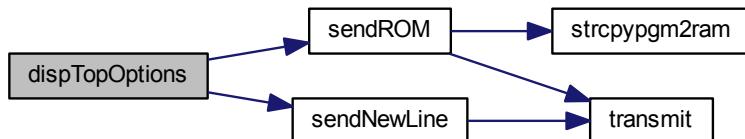
Here is the call graph for this function:



5.14.1.19 static void dispTopOptions (void) [static]

Display the user options for the top level home menu

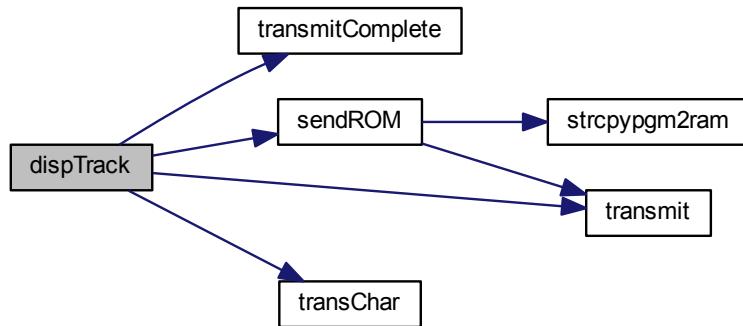
Here is the call graph for this function:



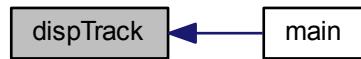
5.14.1.20 void dispTrack (TrackingData target)

Displays the Tracking data over serial

Here is the call graph for this function:



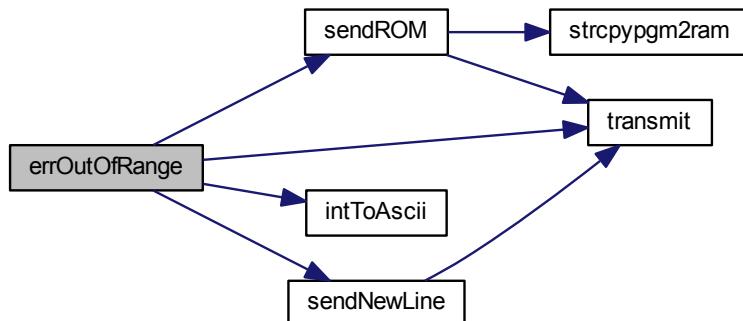
Here is the caller graph for this function:



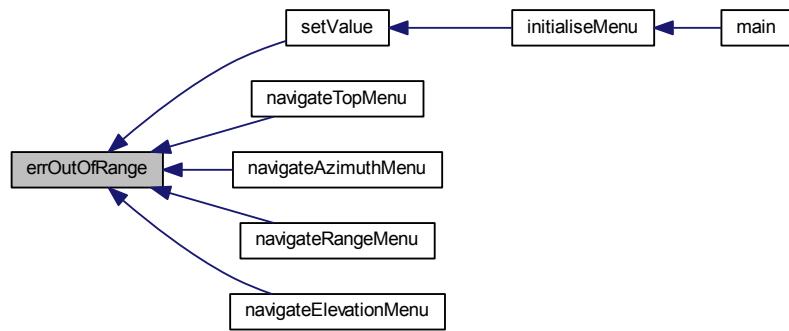
5.14.1.21 static void errOutOfRange (int lowerBound, int upperBound) [static]

Description: Displays a number out of range error

Here is the call graph for this function:

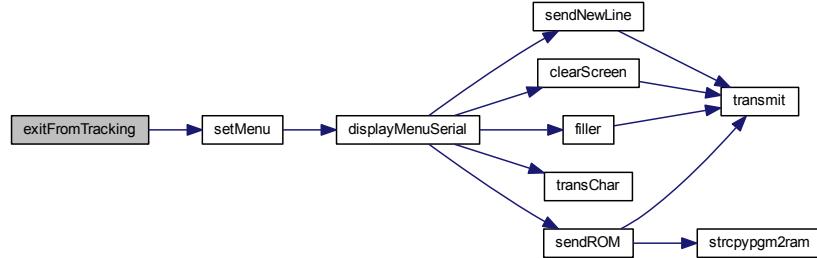


Here is the caller graph for this function:



5.14.1.22 static void exitFromTracking (void) [static]

Here is the call graph for this function:

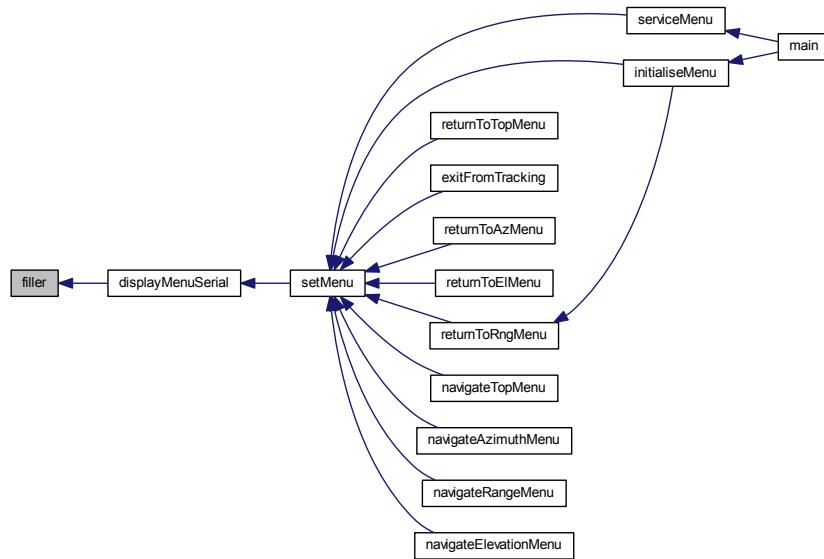


5.14.1.23 static void filler (char length) [static]

Here is the call graph for this function:



Here is the caller graph for this function:



5.14.1.24 static int getLocalInputMenu (int *maxStates*, int(*)(int) *function*) [static]

5.14.1.25 static int getSerialNumericInput () [static]

Waits for input and parses number.

Function: `waitForNumericInput`

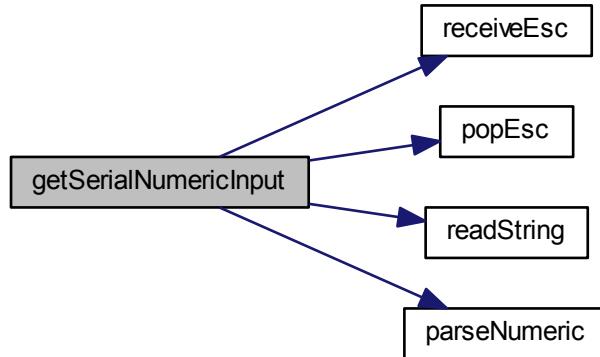
Include:

Description: Waits in a loop for a user command ended with a new line character. Once a command is received, it is converted into the appropriate numeric value that the user has given.

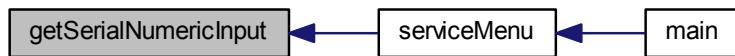
Arguments: None

Returns: The integer result of the string input `ERR_NOT_NUMERIC` for any non-numeric input `ERR_NUM_OUT_OF_RANGE` for 0 digits or 5+ digits `ESC_PRESSED` if escape was pressed Get the input string and store it in

Here is the call graph for this function:



Here is the caller graph for this function:



5.14.1.26 void initialiseMenu (`systemState * state`)

Initialises the menu system.

Function: [initialiseMenu\(void\)](#)

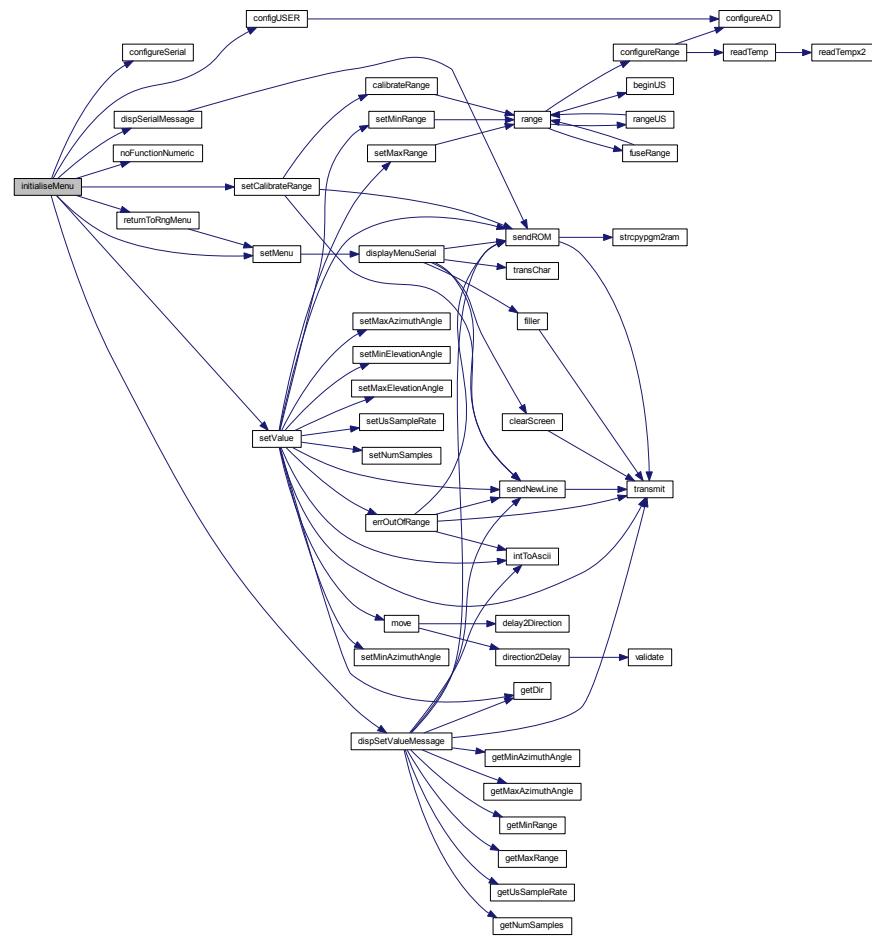
Include: [Menusestem.h](#)

Description: initialises the menu system so that it is fully operational

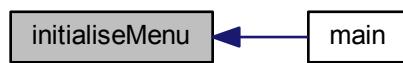
Arguments: None

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:

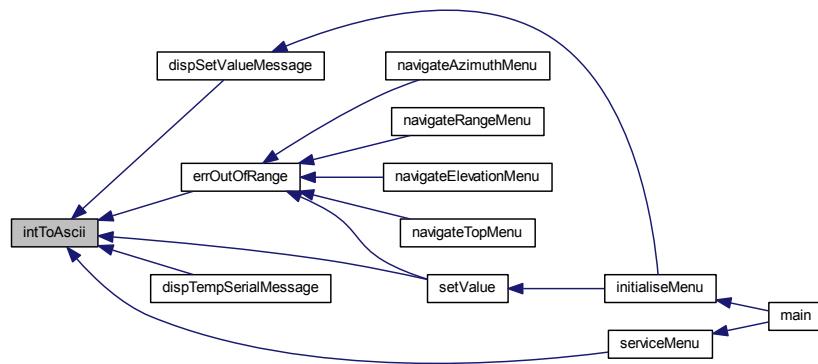


5.14.1.27 static char * intToAscii (int num) [static]

Description: Returns the value of the potentiometer on the user interface, given a maximum and minimum value, and the interval between values (eg 10-100 in multiples of 10).

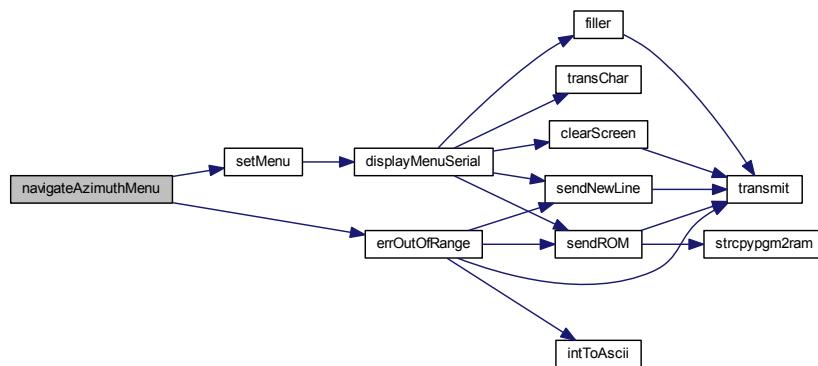
Description: Converts a number to a string Can only print numbers under 8 digits

Here is the caller graph for this function:



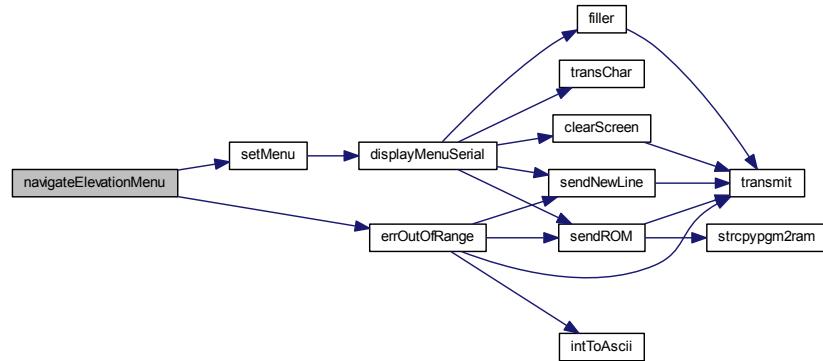
5.14.1.28 static void navigateAzimuthMenu (int *inputResult*) [static]

Here is the call graph for this function:

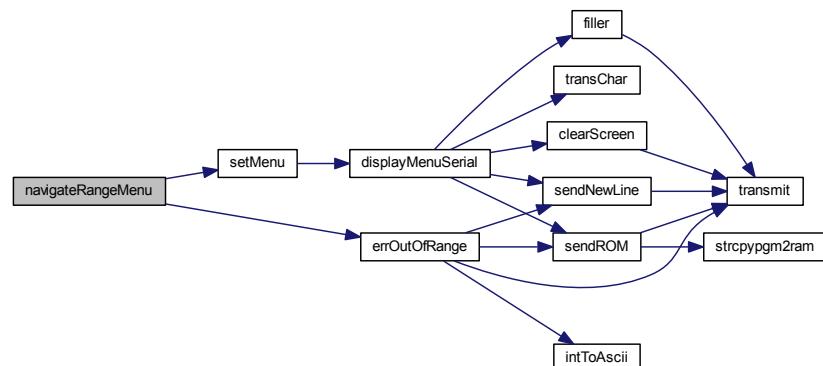


5.14.1.29 static void navigateElevationMenu (int *inputResult*) [static]

Here is the call graph for this function:

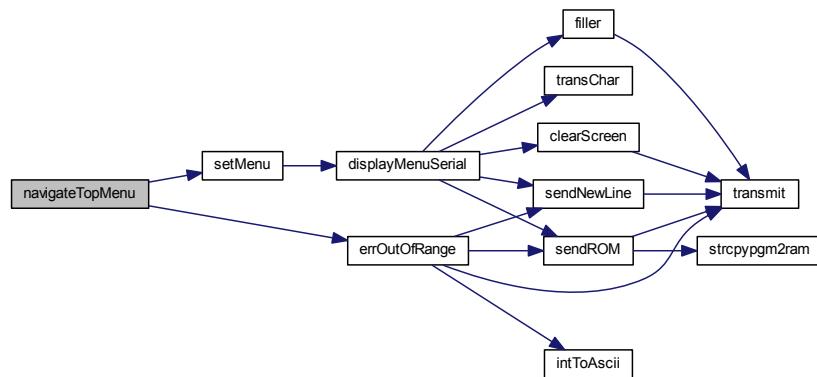
5.14.1.30 static void navigateRangeMenu (int *inputResult*) [static]

Here is the call graph for this function:



5.14.1.31 static void navigateTopMenu (int *inputResult*) [static]

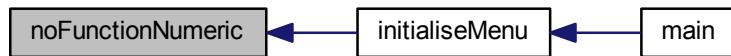
Here is the call graph for this function:



5.14.1.32 static void noFunction (void) [static]

5.14.1.33 static void noFunctionNumeric (int *input*) [static]

Here is the caller graph for this function:



5.14.1.34 static int parseNumeric (char * *number*)

parses user input string into a number

Function: [parseNumeric\(char *number\)](#)

Include:

Description: Converts ASCII input to a number, and records an error for non-numeric input, or if the number is larger than 4 digits. No number used by the user in this program will be larger than 4 digits.

Arguments: The ASCII string to decode

Returns: The integer result of the string `ERR_NOT_NUMERIC` for any non-numeric input `ERR_NUM_OUT_OF_RANGE` for 0 digits or 5+ digits

Function: [parseNumeric\(char *number\)](#)

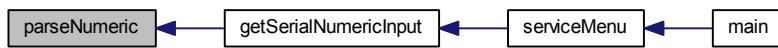
Include:

Description: Calls the function which matches the user input

Arguments: None

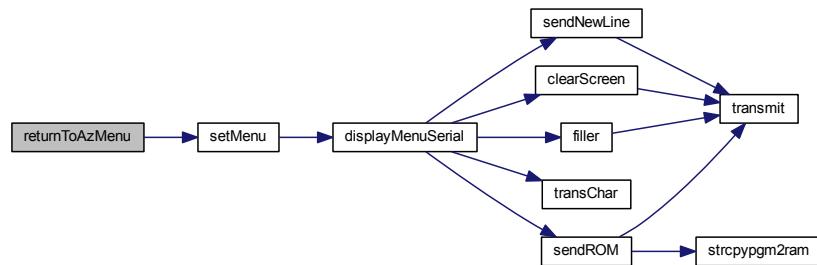
Returns: None

Here is the caller graph for this function:



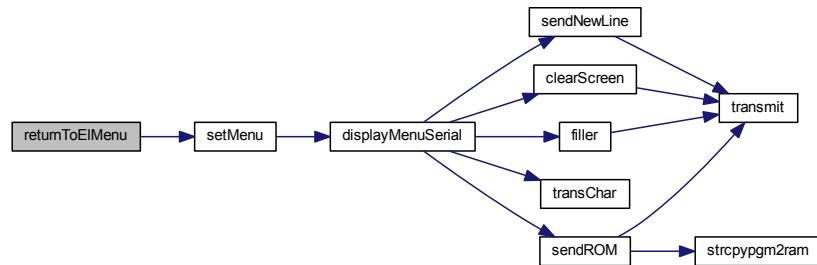
5.14.1.35 static void returnToAzMenu (void) [static]

Here is the call graph for this function:



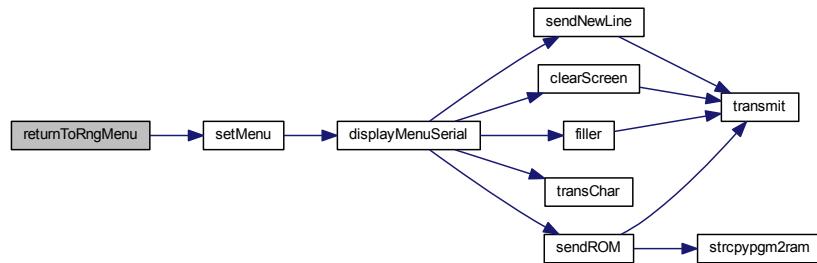
5.14.1.36 static void returnToElMenu (void) [static]

Here is the call graph for this function:

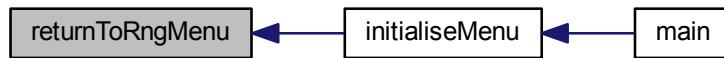


5.14.1.37 static void returnToRngMenu (void) [static]

Here is the call graph for this function:

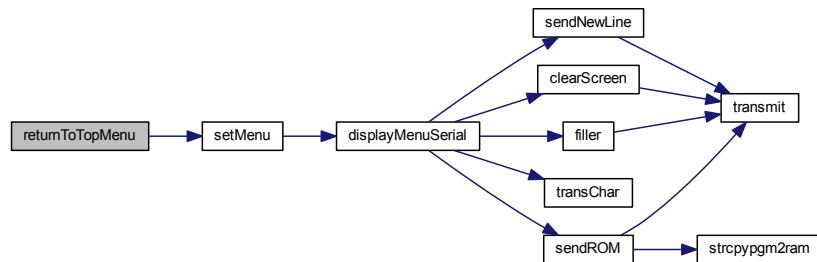


Here is the caller graph for this function:



5.14.1.38 static void returnToTopMenu (void) [static]

Here is the call graph for this function:



5.14.1.39 static void sendNewLine(char length) [static]

Include: Local to [Menuseystem.c](#).

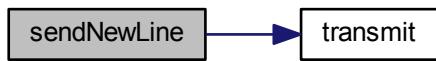
Function: [sendNewLine\(char length\)](#)

: Prints a number of new line () characters.

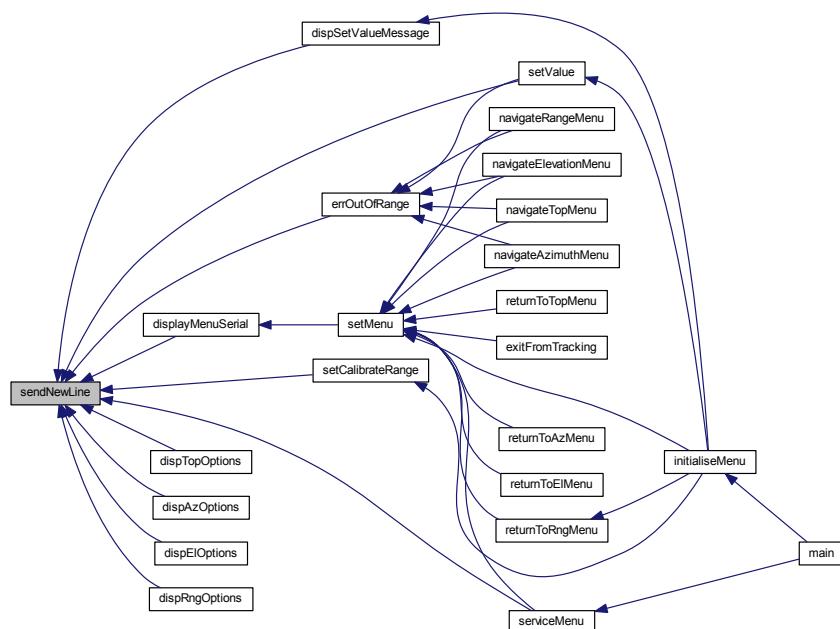
The number of new lines to print

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.14.1.40 static void sendROM (const rom char * romchar) [static]

Include: Local to [Menusystem.c](#).

File: [Menusystem.c](#) Author:

Description:

Duties:

Functions:

Created on 16 September 2014, 6:47 PM

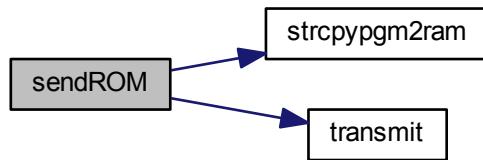
Function: sendROM(void)

: Transmits the given string from ROM over serial

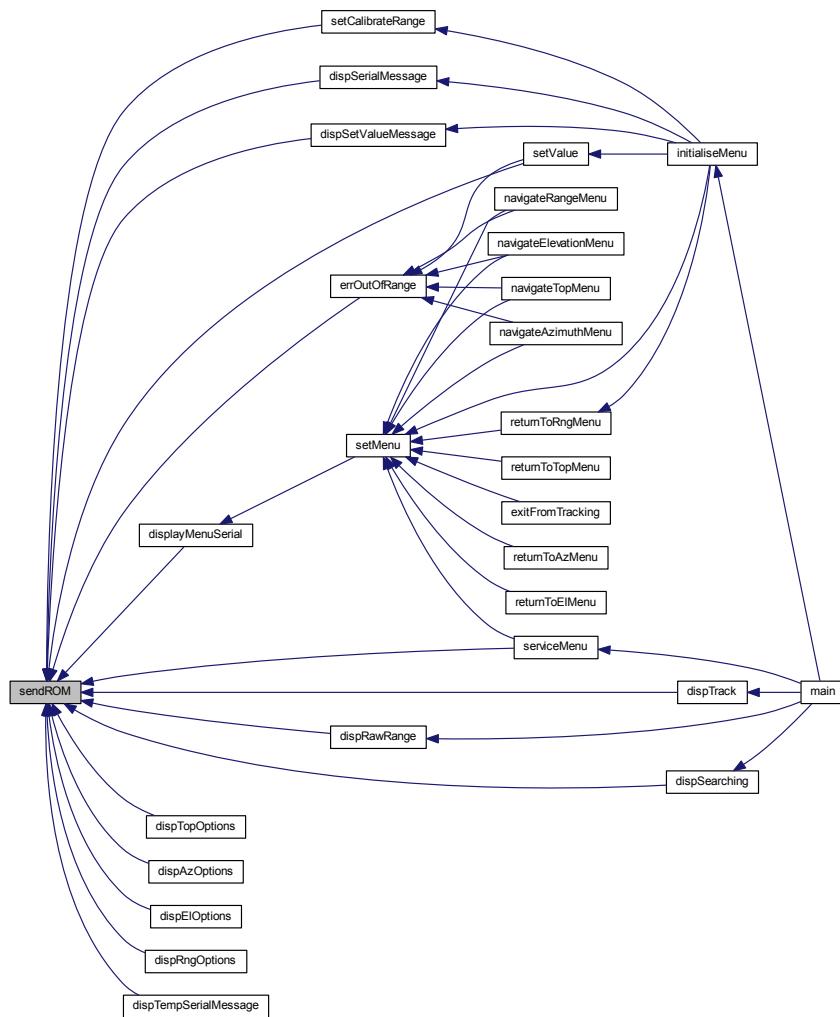
The string to transmit

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.14.1.41 void serviceMenu (void)

services any user interface with the menu

Function: [serviceMenu\(void\)](#)

Include:

Description: Checks if the user has made any inputs to the system. If not the function simply returns. If they have then it services the inputs, displays the correct outputs and performs the specified actions

Arguments: None

Returns: None If Esc or Back button pressed, return

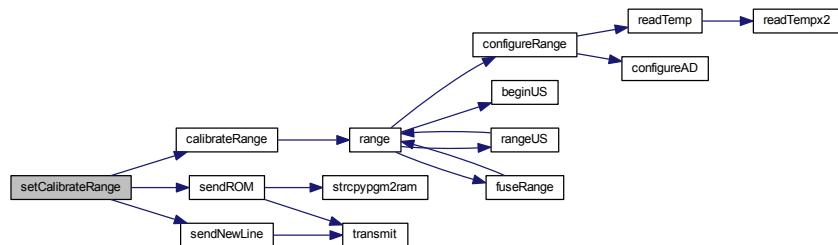
Otherwise Confirm the selection

Here is the caller graph for this function:

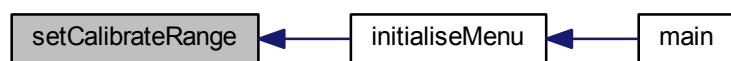


5.14.1.42 static void setCalibrateRange (int num) [static]

Here is the call graph for this function:

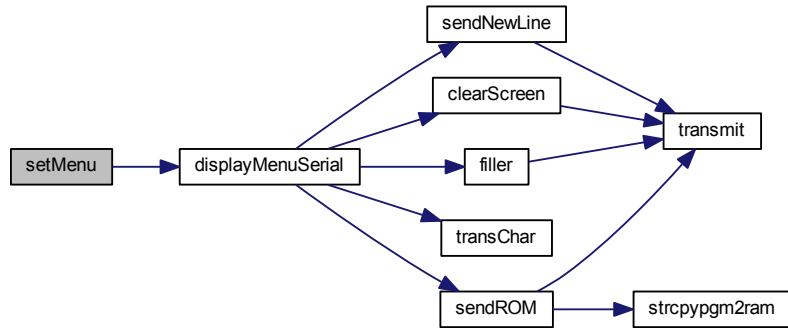


Here is the caller graph for this function:

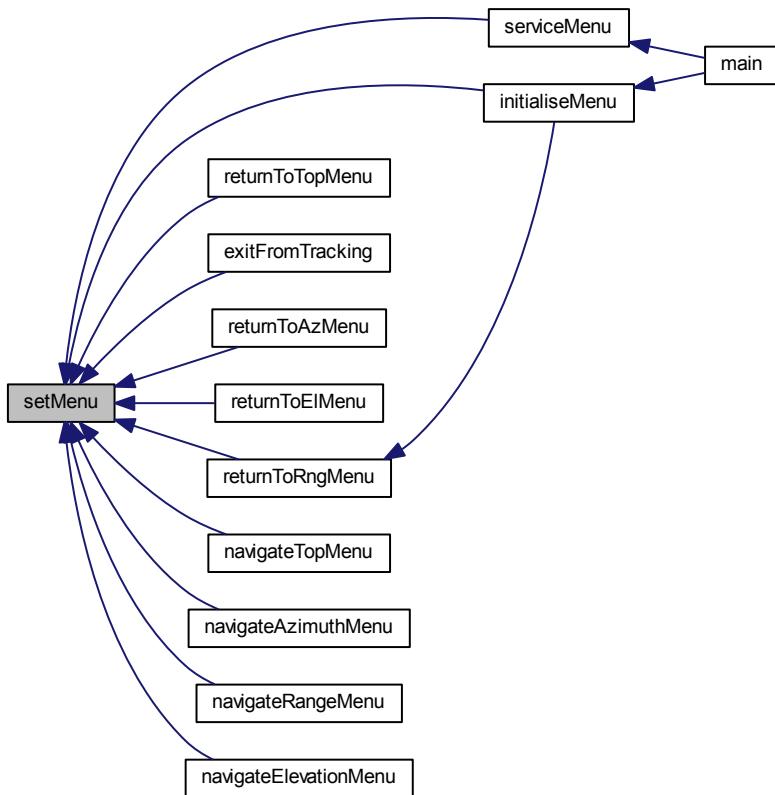


5.14.1.43 static void setMenu (struct menuStruct menu) [static]

Here is the call graph for this function:



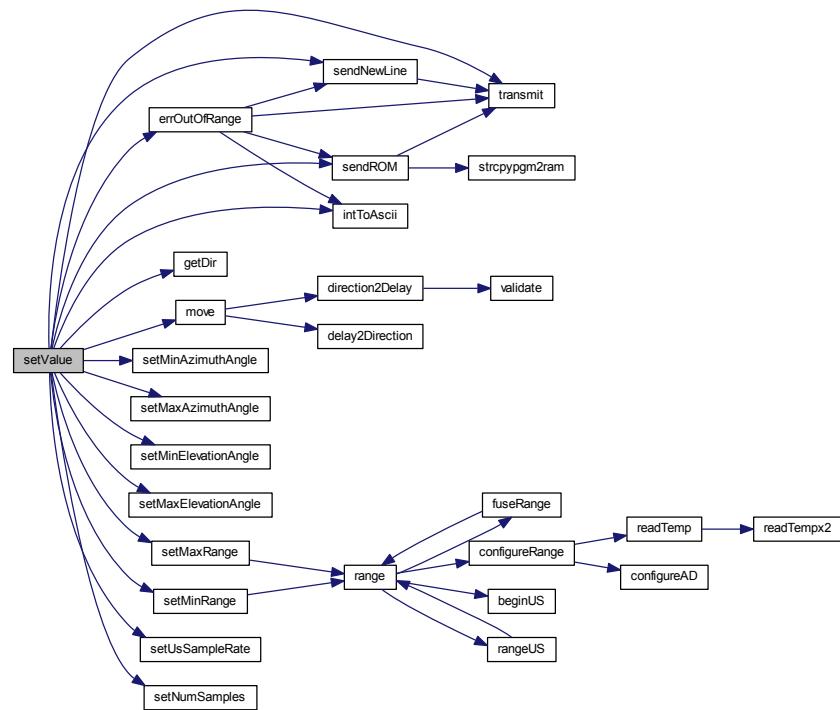
Here is the caller graph for this function:



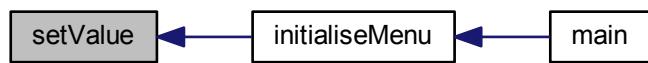
5.14.1.44 static void setValue (int *input*) [static]

Description: General function for menus which set values (Such as Set Max Range). This calls the appropriate function, and transmits user messages.

Here is the call graph for this function:



Here is the caller graph for this function:



5.14.2 Variable Documentation

```
5.14.2.1 struct menuStruct AzGoto = {AZ_GOTO, gotoAzAngle, gotoAzAngleLCD,MIN_ANGLE_INFIMUM,
MAX_ANGLE_SUPREMUM, 5,dispSetValueMessage, setValue, noFunctionNumeric,
returnToAzMenu}
```

Remote/Local Functions.

- 5.14.2.2 struct menuStruct AzMax = {AZ_MAX, maxAz1, maxAzSetStr,MAX_ANGLE_INFIMUM,
MAX_ANGLE_SUPREMUM, 5, dispSetValueMessage, setValue, noFunctionNumeric,
returnToAzMenu}
- 5.14.2.3 struct menuStruct AzMenu = {AZ_MENU, azMenu, azMenuLcd,1, 6, 1, dispAzOptions,
navigateAzimuthMenu, 0, returnToTopMenu}
- 5.14.2.4 struct menuStruct AzMin = {AZ_MIN, minAz1, minAzSetStr,MIN_ANGLE_INFIMUM,
MIN_ANGLE_SUPREMUM, 5, dispSetValueMessage, setValue, noFunctionNumeric,
returnToAzMenu}
- 5.14.2.5 struct menuStruct calibrateRangeMenu
- 5.14.2.6 struct menuStruct ElGoto = {EL_GOTO, gotoElAngle, gotoELAngleLCD,MIN_ANGLE_INFIMUM,
MAX_ANGLE_SUPREMUM, 5, dispSetValueMessage, setValue, noFunctionNumeric,
returnToElMenu}
- 5.14.2.7 struct menuStruct ElMax = {EL_MAX, maxEl1, maxElSetStr,MAX_ANGLE_INFIMUM,
MAX_ANGLE_SUPREMUM, 5, dispSetValueMessage, setValue, noFunctionNumeric,
returnToElMenu}
- 5.14.2.8 struct menuStruct ElMenu = {EL_MENU, elMenu, elMenuLcd,1, 6, 1, dispElOptions,
navigateElevationMenu, 0, returnToTopMenu}
- 5.14.2.9 struct menuStruct ElMin = {EL_MIN, minEl1, minElSetStr,MIN_ANGLE_INFIMUM,
MIN_ANGLE_SUPREMUM, 5, dispSetValueMessage, setValue, noFunctionNumeric,
returnToElMenu}
- 5.14.2.10 struct menuStruct m_currentMenu
- Global variable with the current menu position
- 5.14.2.11 systemState* m_trackingState [static]
- 5.14.2.12 userState m_userMode [static]
- Global variable with the current user mode: Local, remote or factory
- 5.14.2.13 struct menuStruct NumSamples
- 5.14.2.14 struct menuStruct RangeMenu = {RANGE_MENU, rngMenu, rngMenuLcd,1, 6, 1, dispRngOptions,
navigateRangeMenu, 0, returnToTopMenu}
- 5.14.2.15 struct menuStruct RawRange
- 5.14.2.16 struct menuStruct RngMax = {RANGE_MAX, maxRngSerialStr, maxRngSetStr,
MAX_RANGE_INFIMUM, MAX_RANGE_SUPREMUM, 50, dispSetValueMessage, setValue,
noFunctionNumeric, returnToRngMenu}
- 5.14.2.17 struct menuStruct RngMin = {RANGE_MIN, minRngSerialStr, minRngSetStr, MIN_RANGE_INFIMUM,
MIN_RANGE_SUPREMUM, 50, dispSetValueMessage, setValue, noFunctionNumeric,
returnToRngMenu}
- 5.14.2.18 struct menuStruct ShowTemp = {SHOW_TEMP, showTempLCDTitle, showTempLCDTitle, 0, 0, 0,
dispTempSerialMessage, noFunctionNumeric, noFunctionNumeric, returnToTopMenu}

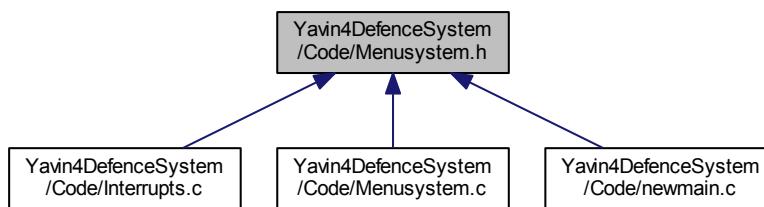
```
5.14.2.19 struct menuStruct topMenu = {TOP_LEVEL, title, welcomeLcd, 1, 6, 1, dispTopOptions,
                                         navigateTopMenu, 0, noFunction}
```

```
5.14.2.20 struct menuStruct Tracking = {TRACKING, autoSerialMessage, autoLcdTitle, 0, 0, 0,
                                         dispSerialMessage, noFunctionNumeric, noFunctionNumeric, exitFromTracking}
```

```
5.14.2.21 struct menuStruct UsSampleRate
```

5.15 Yavin4DefenceSystem/Code/Menusystem.h File Reference

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



Functions

- void `initialiseMenu (systemState *state)`
Initialises the menu system.
- char `checkForSerialInput (void)`
- void `serviceMenu (void)`
services any user interface with the menu
- void `dispTrack (TrackingData target)`
- void `dispSearching (void)`
- void `dispRawRange (void)`

Variables

- const rom char `newLine []`
- const rom char `CHOOSE []`
- const rom char `menuPrefix3 []`
- const rom char `menuPrefix4 []`
- const rom char `menuPrefix5 []`
- const rom char `menuPrefix8 []`
- const rom char `goUp []`
- const rom char `goUp2 []`
- const rom char `inputNumRangeStr []`
- const rom char `and []`
- const rom char `tab []`
- const rom char `hz []`
- const rom char `title []`
- const rom char `welcomeLcd []`
- const rom char `topOption1 []`

- const rom char `topOption2` []
- const rom char `topOption3` []
- const rom char `topOption4` []
- const rom char `topOption5` []
- const rom char `topOptionCalTemp` []
- const rom char `topOptionLocal` []
- const rom char `topOptionForFactory` []
- const rom char `topOptionRemote` []
- const rom char `topOptionRemoteLCD` []
- const rom char `azMenu` []
- const rom char `azMenuLcd` []
- const rom char `azOption1` []
- const rom char `azOption2` []
- const rom char `azOption3` []
- const rom char `azOption4` []
- const rom char `elMenu` []
- const rom char `elMenuLcd` []
- const rom char `elOption1` []
- const rom char `elOption2` []
- const rom char `elOption3` []
- const rom char `elOption4` []
- const rom char `rngMenu` []
- const rom char `rngMenuLcd` []
- const rom char `rngOption1` []
- const rom char `rngOption2` []
- const rom char `rngOption3` []
- const rom char `rngOption4` []
- const rom char `rngOption5` []
- const rom char `rngOption6` []
- const rom char `rngOption7` []
- const rom char `showTempLCDTitle` []
- const rom char `showTempLCD` []
- const rom char `showTemp1` []
- const rom char `showTemp2` []
- const rom char `gotoAzAngle` []
- const rom char `gotoAzAngleLCD` []
- const rom char `gotoElAngle` []
- const rom char `gotoELAngleLCD` []
- const rom char `gotoAngle2` []
- const rom char `angleStr` []
- const rom char `maxAzStr` []
- const rom char `maxAzSetStr` []
- const rom char `maxAz1` []
- const rom char `maxAz3` []
- const rom char `currentMinAngleStr` []
- const rom char `minAzStr` []
- const rom char `minAzSetStr` []
- const rom char `minAz1` []
- const rom char `minAz3` []
- const rom char `calibrateAngle1` []
- const rom char `maxElStr` []
- const rom char `maxElSetStr` []
- const rom char `maxEl1` []
- const rom char `maxEl3` []
- const rom char `minElStr` []

- const rom char `minElSetStr []`
- const rom char `minEl1 []`
- const rom char `minEl3 []`
- const rom char `minRngStr []`
- const rom char `minRngSetStr []`
- const rom char `minRngSerialStr []`
- const rom char `maxRngStr []`
- const rom char `maxRngSetStr []`
- const rom char `maxRngSerialStr []`

5.15.1 Function Documentation

5.15.1.1 `char checkForSerialInput (void)`

Function: `checkForSerialInput(void)`

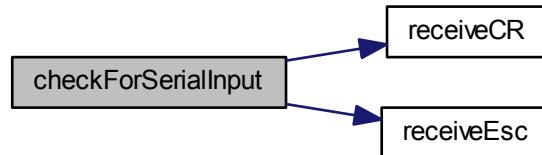
Include: Menusystem

Description: Checks the serial/local buffers for inputs

Arguments: None

Returns: 1 if input has been received, 0 otherwise Wait until the receive buffer is no longer empty Indicating that a command has been passed

Here is the call graph for this function:



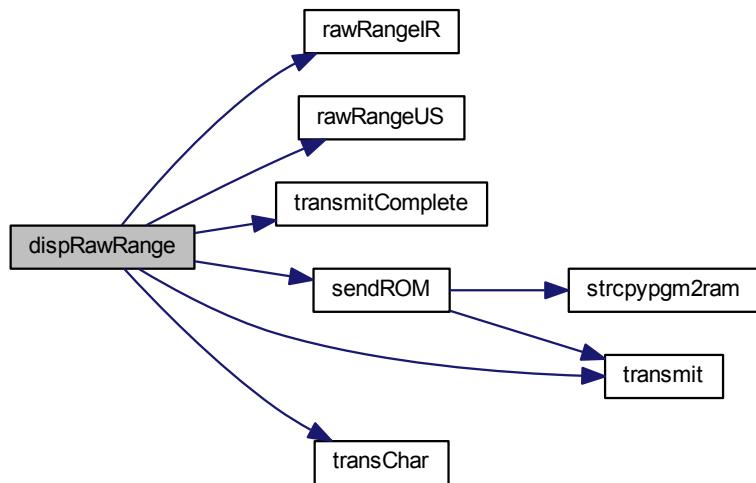
Here is the caller graph for this function:



5.15.1.2 `void dispRawRange (void)`

Displays the Tracking data over serial

Here is the call graph for this function:

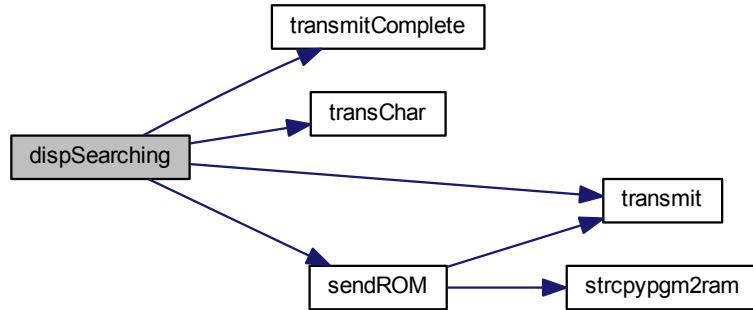


Here is the caller graph for this function:



5.15.1.3 void dispSearching (void)

Here is the call graph for this function:



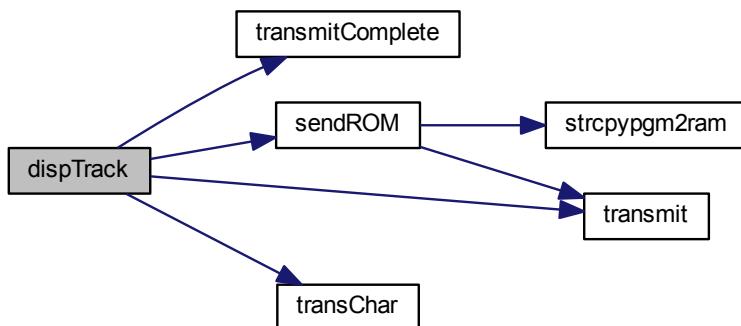
Here is the caller graph for this function:



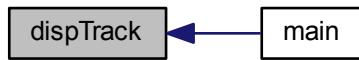
5.15.1.4 void dispTrack (TrackingData target)

Displays the Tracking data over serial

Here is the call graph for this function:



Here is the caller graph for this function:



5.15.1.5 void initialiseMenu (**systemState** * state)

Initialises the menu system.

Function: [initialiseMenu\(systemState *state\)](#)

Include: [Menusystem.h](#)

Description: initialises the menu system so that it is fully operational

Arguments: state - The current system state

Returns: None

Function: [initialiseMenu\(void\)](#)

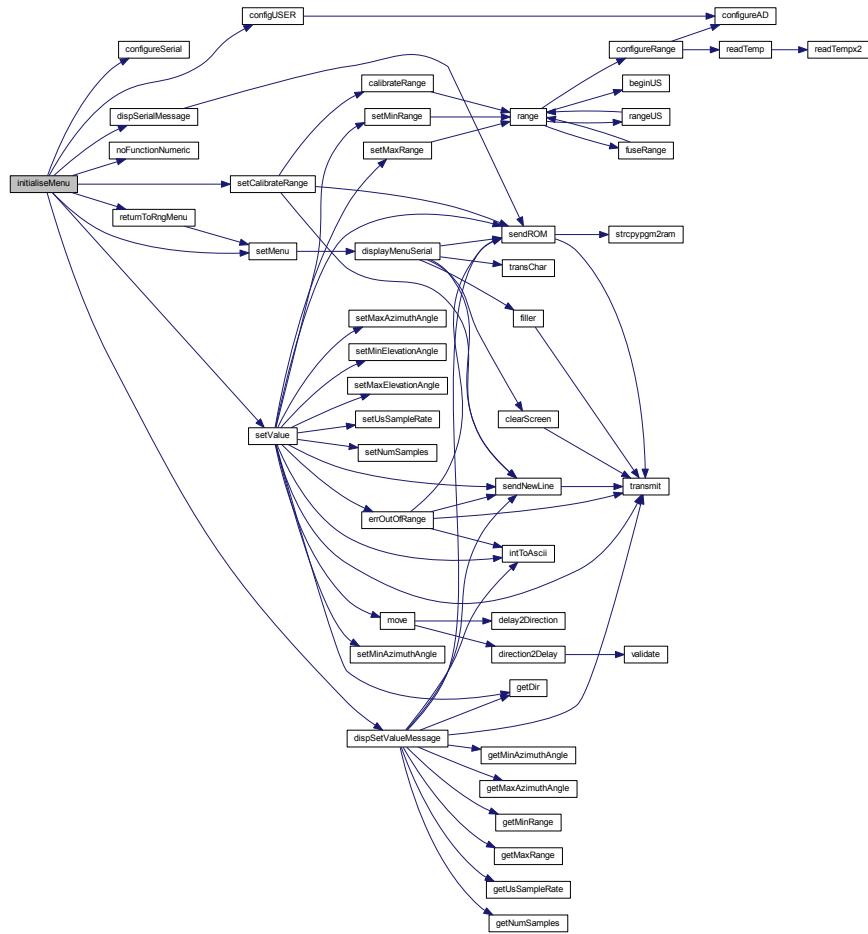
Include: [Menusystem.h](#)

Description: initialises the menu system so that it is fully operational

Arguments: None

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.15.1.6 void serviceMenu (void)

services any user interface with the menu

Function: [serviceMenu\(void\)](#)

Include:

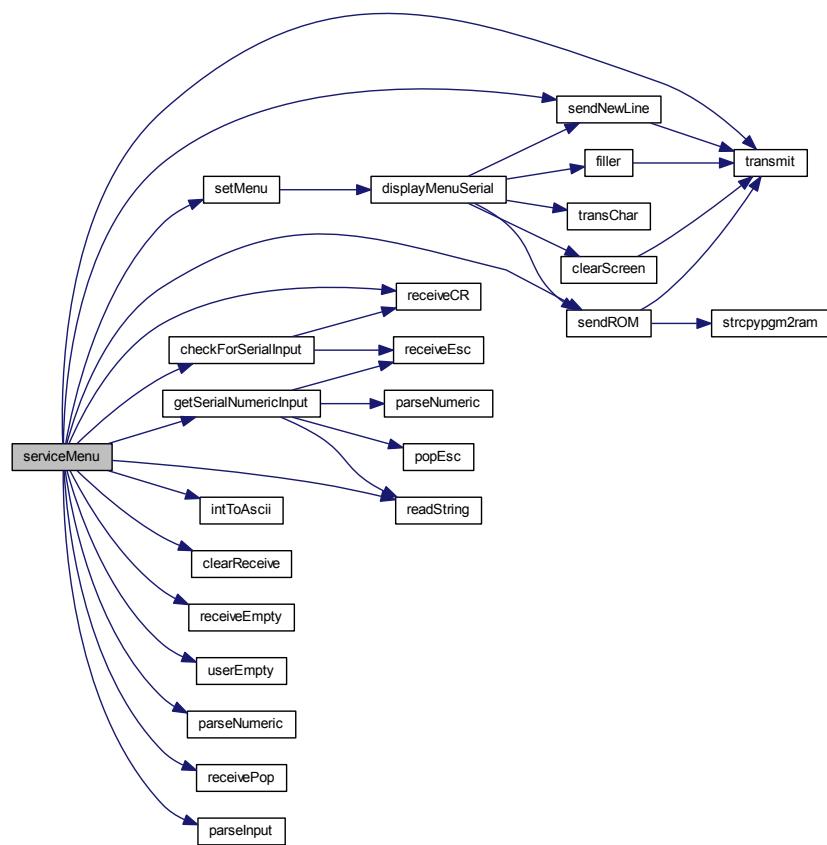
Description: Checks if the user has made any inputs to the system. If not the function simply returns. If they have then it services the inputs, displays the correct outputs and performs the specified actions

Arguments: None

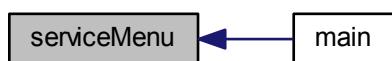
Returns: None If Esc or Back button pressed, return

Otherwise Confirm the selection

Here is the call graph for this function:



Here is the caller graph for this function:



5.15.2 Variable Documentation

5.15.2.1 const rom char and[]

5.15.2.2 const rom char angleStr[]

5.15.2.3 const rom char azMenu[]

5.15.2.4 const rom char azMenuLcd[]

5.15.2.5 const rom char azOption1[]

5.15.2.6 const rom char azOption2[]

5.15.2.7 const rom char azOption3[]

5.15.2.8 const rom char azOption4[]

5.15.2.9 const rom char calibrateAngle1[]

5.15.2.10 const rom char CHOOSE[]

5.15.2.11 const rom char currentMinAngleStr[]

5.15.2.12 const rom char elMenu[]

5.15.2.13 const rom char elMenuLcd[]

5.15.2.14 const rom char elOption1[]

5.15.2.15 const rom char elOption2[]

5.15.2.16 const rom char elOption3[]

5.15.2.17 const rom char elOption4[]

5.15.2.18 const rom char gotoAngle2[]

5.15.2.19 const rom char gotoAzAngle[]

5.15.2.20 const rom char gotoAzAngleLCD[]

5.15.2.21 const rom char gotoElAngle[]

5.15.2.22 const rom char gotoELAngleLCD[]

5.15.2.23 const rom char goUp[]

5.15.2.24 const rom char goUp2[]

5.15.2.25 const rom char hz[]

5.15.2.26 const rom char inputNumRangeStr[]

5.15.2.27 const rom char maxAz1[]

5.15.2.28 const rom char maxAz3[]

5.15.2.29 const rom char maxAzSetStr[]

5.15.2.30 const rom char maxAzStr[]

- 5.15.2.31 const rom char maxEl1[]
- 5.15.2.32 const rom char maxEl3[]
- 5.15.2.33 const rom char maxElSetStr[]
- 5.15.2.34 const rom char maxElStr[]
- 5.15.2.35 const rom char maxRngSerialStr[]
- 5.15.2.36 const rom char maxRngSetStr[]
- 5.15.2.37 const rom char maxRngStr[]
- 5.15.2.38 const rom char menuPrefix3[]
- 5.15.2.39 const rom char menuPrefix4[]
- 5.15.2.40 const rom char menuPrefix5[]
- 5.15.2.41 const rom char menuPrefix8[]
- 5.15.2.42 const rom char minAz1[]
- 5.15.2.43 const rom char minAz3[]
- 5.15.2.44 const rom char minAzSetStr[]
- 5.15.2.45 const rom char minAzStr[]
- 5.15.2.46 const rom char minEl1[]
- 5.15.2.47 const rom char minEl3[]
- 5.15.2.48 const rom char minElSetStr[]
- 5.15.2.49 const rom char minElStr[]
- 5.15.2.50 const rom char minRngSerialStr[]
- 5.15.2.51 const rom char minRngSetStr[]
- 5.15.2.52 const rom char minRngStr[]
- 5.15.2.53 const rom char newLine[]
- 5.15.2.54 const rom char rngMenu[]
- 5.15.2.55 const rom char rngMenuLcd[]
- 5.15.2.56 const rom char rngOption1[]
- 5.15.2.57 const rom char rngOption2[]
- 5.15.2.58 const rom char rngOption3[]

5.15.2.59 const rom char rngOption4[]

5.15.2.60 const rom char rngOption5[]

5.15.2.61 const rom char rngOption6[]

5.15.2.62 const rom char rngOption7[]

5.15.2.63 const rom char showTemp1[]

5.15.2.64 const rom char showTemp2[]

5.15.2.65 const rom char showTempLCD[]

5.15.2.66 const rom char showTempLCDTitle[]

5.15.2.67 const rom char tab[]

5.15.2.68 const rom char title[]

5.15.2.69 const rom char topOption1[]

5.15.2.70 const rom char topOption2[]

5.15.2.71 const rom char topOption3[]

5.15.2.72 const rom char topOption4[]

5.15.2.73 const rom char topOption5[]

5.15.2.74 const rom char topOptionCalTemp[]

5.15.2.75 const rom char topOptionForFactory[]

5.15.2.76 const rom char topOptionLocal[]

5.15.2.77 const rom char topOptionRemote[]

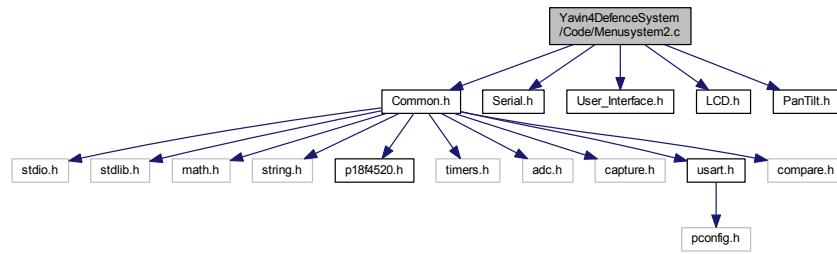
5.15.2.78 const rom char topOptionRemoteLCD[]

5.15.2.79 const rom char welcomeLcd[]

5.16 Yavin4DefenceSystem/Code/Menusystem2.c File Reference

```
#include "Common.h"
#include "Serial.h"
#include "User_Interface.h"
#include "LCD.h"
#include "PanTilt.h"
```

Include dependency graph for Menusystem2.c:



Data Structures

- struct [SubMenu](#)

Macros

- #define MAX_SER_MSG_LEN 30
- #define MAX_LCD_MSG_LEN 20
- #define MAX_NUM_OPTIONS 5

TypeDefs

- typedef struct [SubMenu](#) SubMenu

Enumerations

- enum [MenuLevel](#) { ROOT, SUB, FUNC }

Functions

- static void [parseInput](#) (char input)
parse the user input
- static int [parseNumeric](#) (char *number)
parses user input string into a number
- static void [stateEntry](#) (void)
Performs default actions on first entering the state.
- void [initialiseMenu](#) (void)
Initialises the menu system.
- void [serviceMenu](#) (void)
services any user interface with the menu
- void [menuISR](#) (void)
ISR function for the menu subsystem.

Variables

- static [SubMenu Min](#)
- static [SubMenu Max](#)
- static [SubMenu AutoTrack](#)
- static [SubMenu ManTrack](#)
- static [SubMenu Az](#)
- static [SubMenu Elev](#)
- static [SubMenu Root](#)
- static [MenuLevel level = ROOT](#)
- static [SubMenu * test = &Root](#)
- static [SubMenu menus \[3\]](#)

5.16.1 Macro Definition Documentation

5.16.1.1 `#define MAX_LCD_MSG_LEN 20`

5.16.1.2 `#define MAX_NUM_OPTIONS 5`

5.16.1.3 `#define MAX_SER_MSG_LEN 30`

5.16.2 Typedef Documentation

5.16.2.1 `typedef struct SubMenu SubMenu`

5.16.3 Enumeration Type Documentation

5.16.3.1 `enum MenuLevel`

Enumerator

ROOT

SUB

FUNC

5.16.4 Function Documentation

5.16.4.1 `void initialiseMenu(void)`

Initialises the menu system.

Function: [initialiseMenu\(void\)](#)

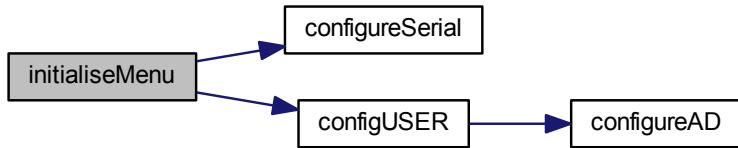
Include: [Menusystem.h](#)

Description: initialises the menu system so that it is fully operational

Arguments: None

Returns: None

Here is the call graph for this function:



5.16.4.2 void menuISR (void)

ISR function for the menu subsystem.

Function: [menuISR\(void\)](#)

Include: [Menusystem.h](#)

Description: services any interrupts associated with the menu system

Arguments: None

Returns: None

5.16.4.3 static void parseInput (char input) [static]

parse the user input

Function: [parseInput\(char input\)](#)

Include: [Menusystem.h](#)

Description: services any interrupts associated with the menu system

Arguments: None

Returns: None

Here is the caller graph for this function:



5.16.4.4 static int parseNumeric (char * number) [static]

parses user input string into a number

Function: [parseNumeric\(char *number\)](#)

Include:

Description: Calls the function which matches the user input

Arguments: None

Returns: None

Here is the caller graph for this function:



5.16.4.5 void serviceMenu (void)

services any user interface with the menu

Function: [serviceMenu\(void\)](#)

Include:

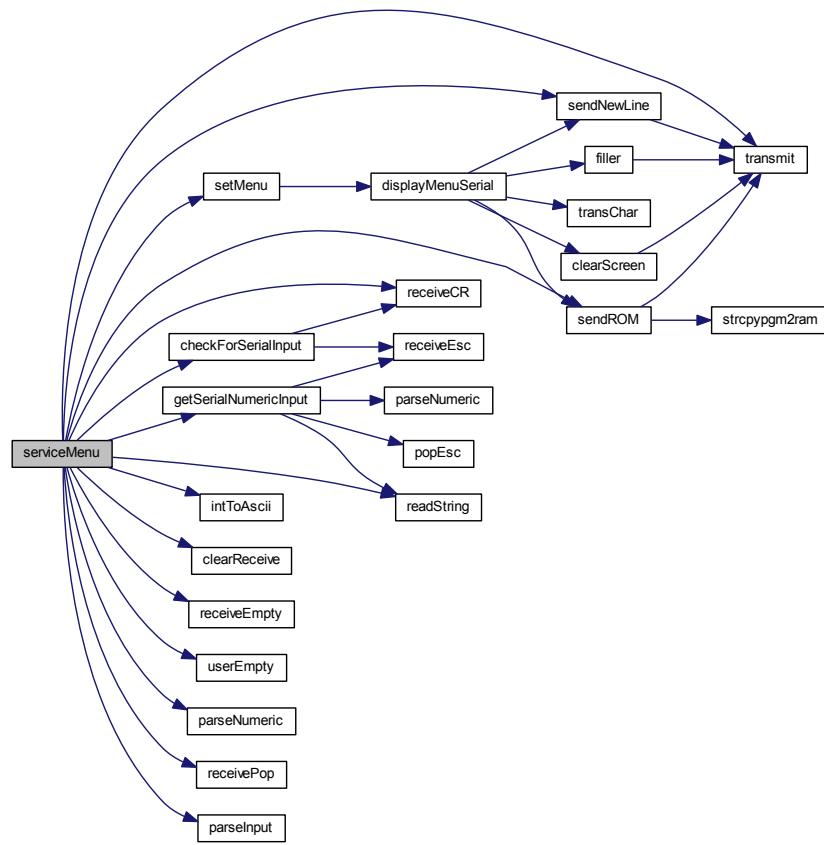
Description: Checks if the user has made any inputs to the system. If not the function simply returns. If they have then it services the inputs, displays the correct outputs and performs the specified actions

Arguments: None

Returns: None If Esc or Back button pressed, return

Otherwise Confirm the selection

Here is the call graph for this function:



5.16.4.6 static void stateEntry(void) [static]

Performs default actions on first entering the state.

Function: [stateEntry\(void\)](#)

Include:

Description:

Arguments: None

Returns: None

Here is the call graph for this function:



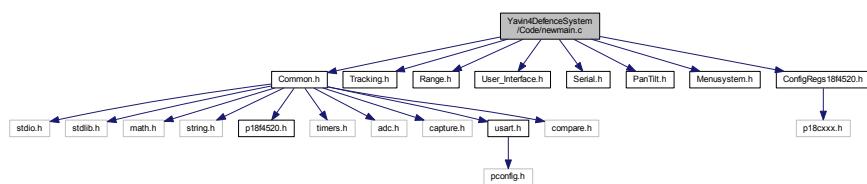
5.16.5 Variable Documentation

- 5.16.5.1 **SubMenu AutoTrack** [static]
- 5.16.5.2 **SubMenu Az** [static]
- 5.16.5.3 **SubMenu Elev** [static]
- 5.16.5.4 **MenuLevel level = ROOT** [static]
- 5.16.5.5 **SubMenu ManTrack** [static]
- 5.16.5.6 **SubMenu Max** [static]
- 5.16.5.7 **SubMenu menus[3]** [static]
- 5.16.5.8 **SubMenu Min** [static]
- 5.16.5.9 **SubMenu Root** [static]
- 5.16.5.10 **SubMenu* test = &Root** [static]

5.17 Yavin4DefenceSystem/Code/newfile.h File Reference

5.18 Yavin4DefenceSystem/Code/newmain.c File Reference

```
#include "Common.h"
#include "Tracking.h"
#include "Range.h"
#include "User_Interface.h"
#include "Serial.h"
#include "PanTilt.h"
#include "Menusystem.h"
#include "ConfigRegs18f4520.h"
Include dependency graph for newmain.c:
```



Functions

- static void `initialization (systemState *state)`
- void `main ()`

Program entry point.

5.18.1 Function Documentation

5.18.1.1 static void initialization (`systemState * state`) [static]

File: [newmain.c](#)

Author: Grant, Bas, Ayush, Zhenning, Jacob, Alvaro

Description: Controls the main system state of the product based on a state transition type template.

Created on 7 September 2014, 4:12 PM

Function: [initialization\(systemState *state\)](#)

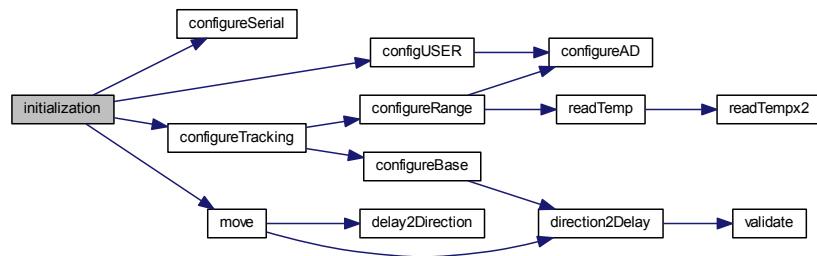
Include: Local to [newmain.c](#)

Description: Initializes the system, turns on the sensors and checks if they are ready to begin working

Arguments: state - The current state of the system

Returns: The next system state - At the moment always just transitions to CHECK

Here is the call graph for this function:



Here is the caller graph for this function:



5.18.1.2 void main (void)

Program entry point.

Function: [main\(void\)](#)

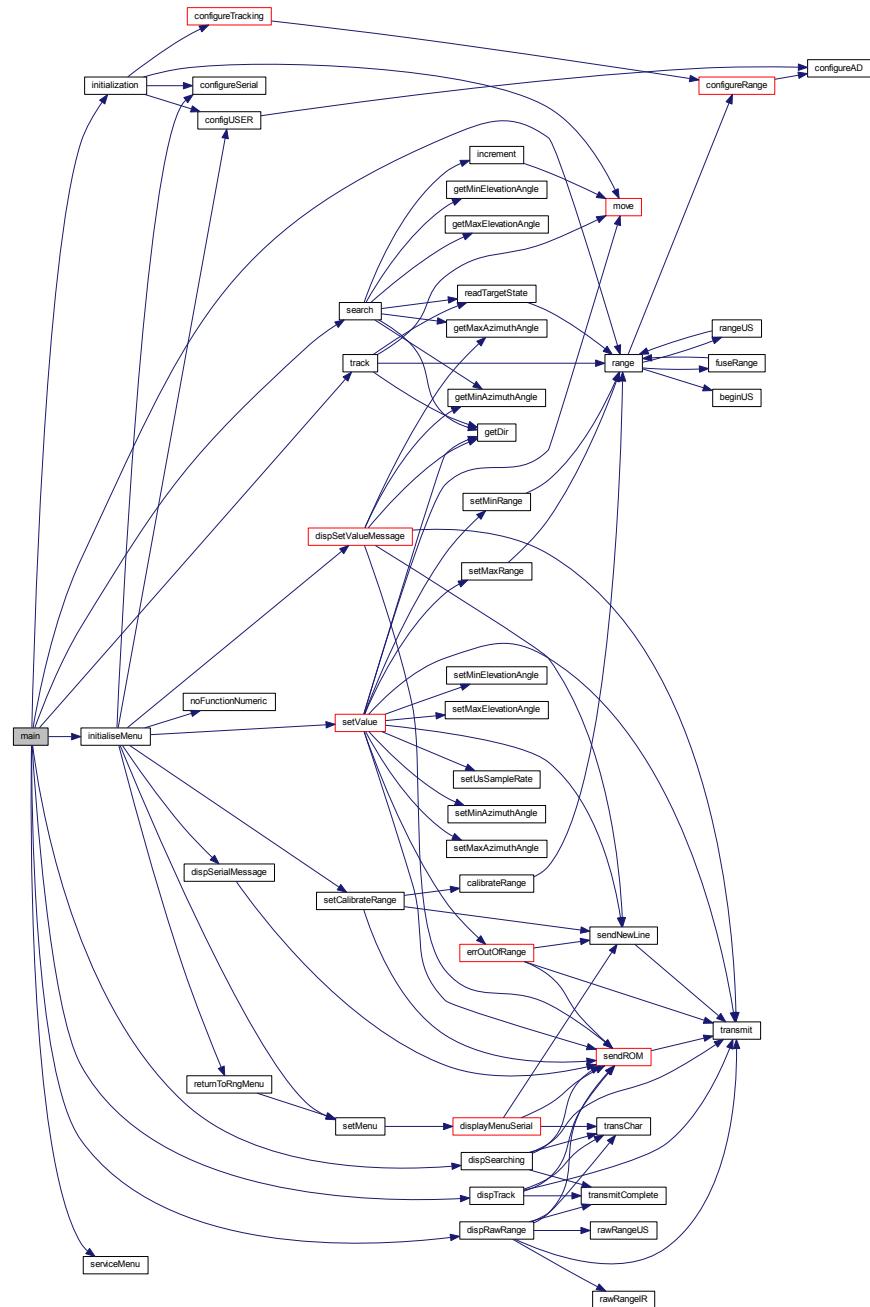
Include: Local to [newmain.c](#)

Description: stores the current system state and manages all transitions

Arguments: None

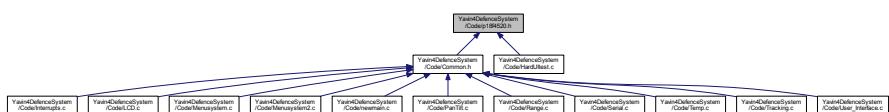
Returns: None

Here is the call graph for this function:



5.19 Yavin4DefenceSystem/Code/p18f4520.h File Reference

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



Macros

- #define ACCESS 0
- #define BANKED 1
- #define Nop() {_asm nop _endasm}
- #define ClrWdt() {_asm clrwdt _endasm}
- #define Sleep() {_asm sleep _endasm}
- #define Reset() {_asm reset _endasm}
- #define Rlcf(f, dest, access) {_asm movlb f rlc f,dest,access _endasm}
- #define Rlncf(f, dest, access) {_asm movlb f rlncf f,dest,access _endasm}
- #define Rrcf(f, dest, access) {_asm movlb f rrcf f,dest,access _endasm}
- #define Rrncf(f, dest, access) {_asm movlb f rrncf f,dest,access _endasm}
- #define Swapf(f, dest, access) {_asm movlb f swapf f,dest,access _endasm }
- #define INTSAVELOCS TBLPTR, TABLAT, PROD

Variables

- volatile near unsigned char PORTA
 - union {
 - struct {
 - unsigned RA0:1
 - unsigned RA1:1
 - unsigned RA2:1
 - unsigned RA3:1
 - unsigned RA4:1
 - unsigned RA5:1
 - unsigned RA6:1
 - unsigned RA7:1
 - struct {
 - unsigned AN0:1
 - unsigned AN1:1
 - unsigned AN2:1
 - unsigned AN3:1
 - unsigned T0CKI:1
 - unsigned AN4:1
 - unsigned OSC2:1
 - unsigned OSC1:1
- struct {
 - unsigned __pad0__:2
 - unsigned VREFN:1
 - unsigned VREFP:1
 - unsigned __pad1__:1
 - unsigned SS:1
 - unsigned CLK0:1
 - unsigned CLK1:1
- struct {
 - unsigned __pad0__:2
 - unsigned CVREF:1
 - unsigned __pad1__:2
 - unsigned NOT_SS:1
- struct {
 - unsigned __pad0__:5
 - unsigned LVDIN:1

```

    }
    struct {
        unsigned __pad0__:5
        unsigned HLVDIN:1
    }
    struct {
        unsigned __pad0__:4
        unsigned C1OUT:1
        unsigned C2OUT:1
    }
} PORTAbits

```

- volatile near unsigned char PORTB

```

union {
    struct {
        unsigned RB0:1
        unsigned RB1:1
        unsigned RB2:1
        unsigned RB3:1
        unsigned RB4:1
        unsigned RB5:1
        unsigned RB6:1
        unsigned RB7:1
    }
    struct {
        unsigned INT0:1
        unsigned INT1:1
        unsigned INT2:1
        unsigned CCP2:1
        unsigned KBIO:1
        unsigned KBII:1
        unsigned KBII2:1
        unsigned KBIII:1
    }
    struct {
        unsigned AN12:1
        unsigned AN10:1
        unsigned AN8:1
        unsigned AN9:1
        unsigned AN11:1
        unsigned PGM:1
        unsigned PGC:1
        unsigned PGD:1
    }
    struct {
        unsigned FLT0:1
    }
} PORTBbits

```

- volatile near unsigned char PORTC

```

union {
    struct {
        unsigned RC0:1
        unsigned RC1:1
        unsigned RC2:1
        unsigned RC3:1
        unsigned RC4:1
        unsigned RC5:1
    }
}

```

```

unsigned RC6:1
unsigned RC7:1
}
struct {
    unsigned T1OSO:1
    unsigned T1OSI:1
    unsigned CCP1:1
    unsigned SCK:1
    unsigned SDI:1
    unsigned SDO:1
    unsigned TX:1
    unsigned RX:1
}
struct {
    unsigned T13CKI:1
    unsigned CCP2:1
    unsigned __pad0__:1
    unsigned SCL:1
    unsigned SDA:1
    unsigned __pad1__:1
    unsigned CK:1
    unsigned DT:1
}
struct {
    unsigned T1CKI:1
    unsigned __pad0__:1
    unsigned P1A:1
}
}
PORTCbits

```

- volatile near unsigned char PORTD

```

union {
    struct {
        unsigned RD0:1
        unsigned RD1:1
        unsigned RD2:1
        unsigned RD3:1
        unsigned RD4:1
        unsigned RD5:1
        unsigned RD6:1
        unsigned RD7:1
    }
    struct {
        unsigned PSP0:1
        unsigned PSP1:1
        unsigned PSP2:1
        unsigned PSP3:1
        unsigned PSP4:1
        unsigned PSP5:1
        unsigned PSP6:1
        unsigned PSP7:1
    }
    struct {
        unsigned __pad0__:5
        unsigned P1B:1
        unsigned P1C:1
        unsigned P1D:1
    }
}

```

```
} PORTDbits
```

- volatile near unsigned char PORTE

```
• union {
    struct {
        unsigned RE0:1
        unsigned RE1:1
        unsigned RE2:1
        unsigned RE3:1
    }
    struct {
        unsigned RD:1
        unsigned WR:1
        unsigned CS:1
        unsigned MCLR:1
    }
    struct {
        unsigned NOT_RD:1
        unsigned NOT_WR:1
        unsigned NOT_CS:1
        unsigned NOT_MCLR:1
    }
    struct {
        unsigned AN5:1
        unsigned AN6:1
        unsigned AN7:1
        unsigned VPP:1
    }
}
```

} PORTEbits

- volatile near unsigned char LATA

```
• struct {
    unsigned LATA0:1
    unsigned LATA1:1
    unsigned LATA2:1
    unsigned LATA3:1
    unsigned LATA4:1
    unsigned LATA5:1
    unsigned LATA6:1
    unsigned LATA7:1
}
```

} LATAbits

- volatile near unsigned char LATB

```
• struct {
    unsigned LATB0:1
    unsigned LATB1:1
    unsigned LATB2:1
    unsigned LATB3:1
    unsigned LATB4:1
    unsigned LATB5:1
    unsigned LATB6:1
    unsigned LATB7:1
}
```

} LATBbits

- volatile near unsigned char LATC

```
• struct {
    unsigned LATC0:1
    unsigned LATC1:1
}
```

```
unsigned LATC2:1
unsigned LATC3:1
    unsigned LATC4:1
    unsigned LATC5:1
    unsigned LATC6:1
    unsigned LATC7:1
} LATCbits
```

- volatile near unsigned char LATD

```
struct {
    unsigned LATD0:1
    unsigned LATD1:1
    unsigned LATD2:1
    unsigned LATD3:1
    unsigned LATD4:1
    unsigned LATD5:1
    unsigned LATD6:1
    unsigned LATD7:1
} LATDbits
```

- volatile near unsigned char LATE

```
struct {
    unsigned LATE0:1
    unsigned LATE1:1
    unsigned LATE2:1
} LATEbits
```

- volatile near unsigned char DDRA

```
union {
    struct {
        unsigned TRISA0:1
        unsigned TRISA1:1
        unsigned TRISA2:1
        unsigned TRISA3:1
        unsigned TRISA4:1
        unsigned TRISA5:1
        unsigned TRISA6:1
        unsigned TRISA7:1
    }
    struct {
        unsigned RA0:1
        unsigned RA1:1
        unsigned RA2:1
        unsigned RA3:1
        unsigned RA4:1
        unsigned RA5:1
        unsigned RA6:1
        unsigned RA7:1
    }
} DDRAbits
```

- volatile near unsigned char TRISA

```
union {
    struct {
        unsigned TRISA0:1
        unsigned TRISA1:1
        unsigned TRISA2:1
        unsigned TRISA3:1
    }
}
```

```

    unsigned TRISA4:1
    unsigned TRISA5:1
        unsigned TRISA6:1
        unsigned TRISA7:1
    }
    struct {
        unsigned RA0:1
        unsigned RA1:1
        unsigned RA2:1
        unsigned RA3:1
        unsigned RA4:1
        unsigned RA5:1
        unsigned RA6:1
        unsigned RA7:1
    }
} TRISAbits

```

- volatile near unsigned char DDRB

```

union {
    struct {
        unsigned TRISB0:1
        unsigned TRISB1:1
        unsigned TRISB2:1
        unsigned TRISB3:1
        unsigned TRISB4:1
        unsigned TRISB5:1
        unsigned TRISB6:1
        unsigned TRISB7:1
    }
    struct {
        unsigned RB0:1
        unsigned RB1:1
        unsigned RB2:1
        unsigned RB3:1
        unsigned RB4:1
        unsigned RB5:1
        unsigned RB6:1
        unsigned RB7:1
    }
} DDRBbits

```

- volatile near unsigned char TRISB

```

union {
    struct {
        unsigned TRISB0:1
        unsigned TRISB1:1
        unsigned TRISB2:1
        unsigned TRISB3:1
        unsigned TRISB4:1
        unsigned TRISB5:1
        unsigned TRISB6:1
        unsigned TRISB7:1
    }
    struct {
        unsigned RB0:1
        unsigned RB1:1
        unsigned RB2:1
        unsigned RB3:1
    }
} TRISBbits

```

```

    unsigned RB4:1
    unsigned RB5:1
        unsigned RB6:1
        unsigned RB7:1
    }
} TRISBbits

```

- volatile near unsigned char **DDRC**

```

• union {
    struct {
        unsigned TRISC0:1
        unsigned TRISC1:1
        unsigned TRISC2:1
        unsigned TRISC3:1
        unsigned TRISC4:1
        unsigned TRISC5:1
        unsigned TRISC6:1
        unsigned TRISC7:1
    }
    struct {
        unsigned RC0:1
        unsigned RC1:1
        unsigned RC2:1
        unsigned RC3:1
        unsigned RC4:1
        unsigned RC5:1
        unsigned RC6:1
        unsigned RC7:1
    }
} DDRCbits

```

- volatile near unsigned char **TRISC**

```

• union {
    struct {
        unsigned TRISC0:1
        unsigned TRISC1:1
        unsigned TRISC2:1
        unsigned TRISC3:1
        unsigned TRISC4:1
        unsigned TRISC5:1
        unsigned TRISC6:1
        unsigned TRISC7:1
    }
    struct {
        unsigned RC0:1
        unsigned RC1:1
        unsigned RC2:1
        unsigned RC3:1
        unsigned RC4:1
        unsigned RC5:1
        unsigned RC6:1
        unsigned RC7:1
    }
} TRISCbits

```

- volatile near unsigned char **DDRD**

```

• union {
    struct {

```

```

    unsigned TRISD0:1
    unsigned TRISD1:1
        unsigned TRISD2:1
        unsigned TRISD3:1
        unsigned TRISD4:1
        unsigned TRISD5:1
        unsigned TRISD6:1
        unsigned TRISD7:1
    }
    struct {
        unsigned RD0:1
        unsigned RD1:1
        unsigned RD2:1
        unsigned RD3:1
        unsigned RD4:1
        unsigned RD5:1
        unsigned RD6:1
        unsigned RD7:1
    }
} DDRDbits

```

- volatile near unsigned char TRISD

```

• union {
    struct {
        unsigned TRISD0:1
        unsigned TRISD1:1
        unsigned TRISD2:1
        unsigned TRISD3:1
        unsigned TRISD4:1
        unsigned TRISD5:1
        unsigned TRISD6:1
        unsigned TRISD7:1
    }
    struct {
        unsigned RD0:1
        unsigned RD1:1
        unsigned RD2:1
        unsigned RD3:1
        unsigned RD4:1
        unsigned RD5:1
        unsigned RD6:1
        unsigned RD7:1
    }
} TRISDbits

```

- volatile near unsigned char DDRE

```

• union {
    struct {
        unsigned TRISE0:1
        unsigned TRISE1:1
        unsigned TRISE2:1
        unsigned __pad0__:1
        unsigned PSPMODE:1
        unsigned IBOV:1
        unsigned OBF:1
        unsigned IBF:1
    }
    struct {

```

```

    unsigned RE0:1
    unsigned RE1:1
        unsigned RE2:1
        unsigned RE3:1
    }
} DDREbits

```

- volatile near unsigned char **TRISE**

```

• union {
    struct {
        unsigned TRISE0:1
        unsigned TRISE1:1
        unsigned TRISE2:1
        unsigned __pad0__:1
        unsigned PSPMODE:1
        unsigned IBOV:1
        unsigned OBF:1
        unsigned IBF:1
    }
    struct {
        unsigned RE0:1
        unsigned RE1:1
        unsigned RE2:1
        unsigned RE3:1
    }
} TRISEbits

```

- volatile near unsigned char **OSCTUNE**

```

• union {
    struct {
        unsigned TUN:5
        unsigned __pad0__:1
        unsigned PLLEN:1
        unsigned INTSRC:1
    }
    struct {
        unsigned TUN0:1
        unsigned TUN1:1
        unsigned TUN2:1
        unsigned TUN3:1
        unsigned TUN4:1
    }
} OSCTUNEbits

```

- volatile near unsigned char **PIE1**

```

• struct {
    unsigned TMR1IE:1
    unsigned TMR2IE:1
    unsigned CCP1IE:1
    unsigned SSPIE:1
    unsigned TXIE:1
    unsigned RCIE:1
    unsigned ADIE:1
    unsigned PSPIE:1
} PIE1bits

```

- volatile near unsigned char **PIR1**

```
• struct {
```

```

    unsigned TMR1IF:1
    unsigned TMR2IF:1
    unsigned CCP1IF:1
    unsigned SSP1F:1
    unsigned TXIF:1
    unsigned RCIF:1
    unsigned ADIF:1
    unsigned PSPIF:1
} PIR1bits

```

- volatile near unsigned char IPR1

```

• struct {
    unsigned TMR1IP:1
    unsigned TMR2IP:1
    unsigned CCP1IP:1
    unsigned SSP1P:1
    unsigned TXIP:1
    unsigned RCIP:1
    unsigned ADIP:1
    unsigned PSPIP:1
} IPR1bits

```

- volatile near unsigned char PIE2

```

• union {
    struct {
        unsigned CCP2IE:1
        unsigned TMR3IE:1
        unsigned HLVDIE:1
        unsigned BCLIE:1
        unsigned EEIE:1
        unsigned __pad0__:1
        unsigned CMIE:1
        unsigned OSCFIE:1
    }
    struct {
        unsigned __pad0__:2
        unsigned LVDIE:1
    }
} PIE2bits

```

- volatile near unsigned char PIR2

```

• union {
    struct {
        unsigned CCP2IF:1
        unsigned TMR3IF:1
        unsigned HLVDIF:1
        unsigned BCLIF:1
        unsigned EEIF:1
        unsigned __pad0__:1
        unsigned CMIF:1
        unsigned OSCFIF:1
    }
    struct {
        unsigned __pad0__:2
        unsigned LVDIF:1
    }
} PIR2bits

```

- volatile near unsigned char IPR2
- union {

 struct {

 unsigned CCP2IP:1

 unsigned TMR3IP:1

 unsigned HLVDIP:1

 unsigned BCLIP:1

 unsigned EEIP:1

 unsigned __pad0__:1

 unsigned CMIP:1

 unsigned OSCFIP:1

 }

 struct {

 unsigned __pad0__:2

 unsigned LVDIP:1

 }
 } IPR2bits
- volatile near unsigned char EECON1
- struct {

 unsigned RD:1

 unsigned WR:1

 unsigned WREN:1

 unsigned WRERR:1

 unsigned FREE:1

 unsigned __pad0__:1

 unsigned CFGS:1

 unsigned EEPGD:1
 } EECON1bits
- volatile near unsigned char EECON2
- volatile near unsigned char EEDATA
- volatile near unsigned char EEADR
- volatile near unsigned char RCSTA
- union {

 struct {

 unsigned RX9D:1

 unsigned OERR:1

 unsigned FERR:1

 unsigned ADDEN:1

 unsigned CREN:1

 unsigned SREN:1

 unsigned RX9:1

 unsigned SPEN:1
 }

 struct {

 unsigned __pad0__:3

 unsigned ADEN:1
 }
 } RCSTAbits
- volatile near unsigned char TXSTA
- struct {

 unsigned TX9D:1

 unsigned TRMT:1

 unsigned BRGH:1

 unsigned SENDB:1

 unsigned SYNC:1
 }

```

unsigned TXEN:1
unsigned TX9:1
    unsigned CSRC:1
} TXSTAbits

• volatile near unsigned char TXREG
• volatile near unsigned char RCREG
• volatile near unsigned char SPBRG
• volatile near unsigned char SPBRGH
• volatile near unsigned char T3CON
• union {
    struct {
        unsigned TMR3ON:1
        unsigned TMR3CS:1
        unsigned NOT_T3SYNC:1
        unsigned T3CCP1:1
        unsigned T3CKPS:2
        unsigned T3CCP2:1
        unsigned RD16:1
    }
    struct {
        unsigned __pad0__:2
        unsigned T3SYNC:1
        unsigned __pad1__:1
        unsigned T3CKPS0:1
        unsigned T3CKPS1:1
    }
} T3CONbits

• volatile near unsigned char TMR3L
• volatile near unsigned char TMR3H
• volatile near unsigned char CMCON
• union {
    struct {
        unsigned CM:3
        unsigned CIS:1
        unsigned C1INV:1
        unsigned C2INV:1
        unsigned C1OUT:1
        unsigned C2OUT:1
    }
    struct {
        unsigned CM0:1
        unsigned CM1:1
        unsigned CM2:1
    }
} CMCONbits

• volatile near unsigned char CVRCON
• union {
    struct {
        unsigned CVR:4
        unsigned CVRSS:1
        unsigned CVRR:1
        unsigned CVROE:1
        unsigned CVREN:1
    }
    struct {

```

```

    unsigned CVR0:1
    unsigned CVR1:1
        unsigned CVR2:1
        unsigned CVR3:1
    }
} CVRCONbits

```

- volatile near unsigned char ECCP1AS

```

• union {
    struct {
        unsigned PSSBD:2
        unsigned PSSAC:2
        unsigned ECCPAS:3
        unsigned ECCPASE:1
    }
    struct {
        unsigned PSSBD0:1
        unsigned PSSBD1:1
        unsigned PSSAC0:1
        unsigned PSSAC1:1
        unsigned ECCPAS0:1
        unsigned ECCPAS1:1
        unsigned ECCPAS2:1
    }
} ECCP1ASbits

```

```

• union {
    struct {
        unsigned PSSBD:2
        unsigned PSSAC:2
        unsigned ECCPAS:3
        unsigned ECCPASE:1
    }
    struct {
        unsigned PSSBD0:1
        unsigned PSSBD1:1
        unsigned PSSAC0:1
        unsigned PSSAC1:1
        unsigned ECCPAS0:1
        unsigned ECCPAS1:1
        unsigned ECCPAS2:1
    }
} ECCPASbits

```

- volatile near unsigned char ECCP1DEL

```

• union {
    struct {
        unsigned PDC:7
        unsigned PRSEN:1
    }
    struct {
        unsigned PDC0:1
        unsigned PDC1:1
        unsigned PDC2:1
        unsigned PDC3:1
        unsigned PDC4:1
        unsigned PDC5:1
        unsigned PDC6:1
    }
}

```

```

        }
    } ECCP1DELbits

    • volatile near unsigned char PWM1CON
    • union {
        struct {
            unsigned PDC:7
            unsigned PRSEN:1
        }
        struct {
            unsigned PDC0:1
            unsigned PDC1:1
            unsigned PDC2:1
            unsigned PDC3:1
            unsigned PDC4:1
            unsigned PDC5:1
            unsigned PDC6:1
        }
    } PWM1CONbits

    • volatile near unsigned char BAUDCON
    • union {
        struct {
            unsigned ABDEN:1
            unsigned WUE:1
            unsigned __pad0__:1
            unsigned BRG16:1
            unsigned TXCKP:1
            unsigned RXDTP:1
            unsigned RCIDL:1
            unsigned ABDOVF:1
        }
        struct {
            unsigned __pad0__:4
            unsigned SCKP:1
            unsigned __pad1__:1
            unsigned RCMT:1
        }
    } BAUDCONbits

    • volatile near unsigned char BAUDCTL
    • union {
        struct {
            unsigned ABDEN:1
            unsigned WUE:1
            unsigned __pad0__:1
            unsigned BRG16:1
            unsigned TXCKP:1
            unsigned RXDTP:1
            unsigned RCIDL:1
            unsigned ABDOVF:1
        }
        struct {
            unsigned __pad0__:4
            unsigned SCKP:1
            unsigned __pad1__:1
            unsigned RCMT:1
        }
    }
}

```

```

} BAUDCTLbits

• volatile near unsigned char CCP2CON
• union {
    struct {
        unsigned CCP2M:4
        unsigned DC2B:2
    }
    struct {
        unsigned CCP2M0:1
        unsigned CCP2M1:1
        unsigned CCP2M2:1
        unsigned CCP2M3:1
        unsigned CCP2Y:1
        unsigned CCP2X:1
    }
    struct {
        unsigned __pad0__:4
        unsigned DC2B0:1
        unsigned DC2B1:1
    }
}
} CCP2CONbits

• volatile near unsigned CCPR2
• volatile near unsigned char CCPR2L
• volatile near unsigned char CCPR2H
• volatile near unsigned char CCP1CON
• union {
    struct {
        unsigned CCP1M:4
        unsigned DC1B:2
        unsigned P1M:2
    }
    struct {
        unsigned CCP1M0:1
        unsigned CCP1M1:1
        unsigned CCP1M2:1
        unsigned CCP1M3:1
        unsigned CCP1Y:1
        unsigned CCP1X:1
        unsigned P1M0:1
        unsigned P1M1:1
    }
    struct {
        unsigned __pad0__:4
        unsigned DC1B0:1
        unsigned DC1B1:1
    }
}
} CCP1CONbits

• volatile near unsigned CCPR1
• volatile near unsigned char CCPR1L
• volatile near unsigned char CCPR1H
• volatile near unsigned char ADCON2
• union {
    struct {
        unsigned ADCS:3
        unsigned ACQT:3
    }
}
}
```

```

    unsigned __pad0__:1
    unsigned ADFM:1
    }
    struct {
        unsigned ADCS0:1
        unsigned ADCS1:1
        unsigned ADCS2:1
        unsigned ACQT0:1
        unsigned ACQT1:1
        unsigned ACQT2:1
    }
} ADCON2bits

```

- volatile near unsigned char ADCON1

```

union {
    struct {
        unsigned PCFG:4
        unsigned VCFG:2
    }
    struct {
        unsigned PCFG0:1
        unsigned PCFG1:1
        unsigned PCFG2:1
        unsigned PCFG3:1
        unsigned VCFG0:1
        unsigned VCFG1:1
    }
} ADCON1bits

```

- volatile near unsigned char ADCON0

```

union {
    struct {
        unsigned ADON:1
        unsigned GO_NOT_DONE:1
        unsigned CHS:4
    }
    struct {
        unsigned __pad0__:1
        unsigned GO:1
        unsigned CHS0:1
        unsigned CHS1:1
        unsigned CHS2:1
        unsigned CHS3:1
    }
    struct {
        unsigned __pad0__:1
        unsigned DONE:1
    }
    struct {
        unsigned __pad0__:1
        unsigned NOT_DONE:1
    }
    struct {
        unsigned __pad0__:1
        unsigned GO_DONE:1
    }
} ADCON0bits

```

- volatile near unsigned **ADRES**
- volatile near unsigned char **ADRESL**
- volatile near unsigned char **ADRESH**
- volatile near unsigned char **SSPCON2**
- union {

 struct {

 unsigned **SEN**:1

 unsigned **RSEN**:1

 unsigned **PEN**:1

 unsigned **RCEN**:1

 unsigned **ACKEN**:1

 unsigned **ACKDT**:1

 unsigned **ACKSTAT**:1

 unsigned **GCEN**:1

 }

 struct {

 unsigned **__pad0__**:1

 unsigned **ADMSK1**:1

 unsigned **ADMSK2**:1

 unsigned **ADMSK3**:1

 unsigned **ADMSK4**:1

 unsigned **ADMSK5**:1

 }
 }
 } **SSPCON2bits**
- volatile near unsigned char **SSPCON1**
- union {

 struct {

 unsigned **SSPM**:4

 unsigned **CKP**:1

 unsigned **SSPEN**:1

 unsigned **SSPOV**:1

 unsigned **WCOL**:1

 }

 struct {

 unsigned **SSPM0**:1

 unsigned **SSPM1**:1

 unsigned **SSPM2**:1

 unsigned **SSPM3**:1

 }
 } **SSPCON1bits**
- volatile near unsigned char **SSPSTAT**
- union {

 struct {

 unsigned **BF**:1

 unsigned **UA**:1

 unsigned **R_NOT_W**:1

 unsigned **S**:1

 unsigned **P**:1

 unsigned **D_NOT_A**:1

 unsigned **CKE**:1

 unsigned **SMP**:1

 }

 struct {

 unsigned **__pad0__**:2

 unsigned **R**:1

 unsigned **__pad1__**:2
 }
 }

```

        unsigned D:1
    }
    struct {
        unsigned __pad0__:2
        unsigned W:1
        unsigned __pad1__:2
        unsigned A:1
    }
    struct {
        unsigned __pad0__:2
        unsigned NOT_W:1
        unsigned __pad1__:2
        unsigned NOT_A:1
    }
    struct {
        unsigned __pad0__:2
        unsigned R_W:1
        unsigned __pad1__:2
        unsigned D_A:1
    }
    struct {
        unsigned __pad0__:2
        unsigned NOT_WRITE:1
        unsigned __pad1__:2
        unsigned NOT_ADDRESS:1
    }
} SSPSTATbits

```

- volatile near unsigned char SSPADD
- volatile near unsigned char SSPBUF
- volatile near unsigned char T2CON
- union {
 struct {
 unsigned T2CKPS:2
 unsigned TMR2ON:1
 unsigned T2OUTPS:4
 }
 struct {
 unsigned T2CKPS0:1
 unsigned T2CKPS1:1
 unsigned __pad0__:1
 unsigned T2OUTPS0:1
 unsigned T2OUTPS1:1
 unsigned T2OUTPS2:1
 unsigned T2OUTPS3:1
 }
 struct {
 unsigned __pad0__:3
 unsigned TOUTPS0:1
 unsigned TOUTPS1:1
 unsigned TOUTPS2:1
 unsigned TOUTPS3:1
 }
 }
} T2CONbits

- volatile near unsigned char PR2
- volatile near unsigned char TMR2
- volatile near unsigned char T1CON

- union {
 struct {
 unsigned TMR1ON:1
 unsigned TMR1CS:1
 unsigned NOT_T1SYNC:1
 unsigned T1OSCEN:1
 unsigned T1CKPS:2
 unsigned T1RUN:1
 unsigned RD16:1
 }
 struct {
 unsigned __pad0__:2
 unsigned T1SYNC:1
 unsigned __pad1__:1
 unsigned T1CKPS0:1
 unsigned T1CKPS1:1
 }
 } T1CONbits

- volatile near unsigned char TMR1L
- volatile near unsigned char TMR1H
- volatile near unsigned char RCON
- union {
 struct {
 unsigned NOT_BOR:1
 unsigned NOT_POR:1
 unsigned NOT_PD:1
 unsigned NOT_TO:1
 unsigned NOT_RI:1
 unsigned __pad0__:1
 unsigned SBOREN:1
 unsigned IPEN:1
 }
 struct {
 unsigned BOR:1
 unsigned POR:1
 unsigned PD:1
 unsigned TO:1
 unsigned RI:1
 }
 } RCONbits

- volatile near unsigned char WDTCON
- union {
 struct {
 unsigned SWDTEN:1
 }
 struct {
 unsigned SWDTE:1
 }
 } WDTCONbits

- volatile near unsigned char HLVDCON
- union {
 struct {
 unsigned HLVDL:4
 unsigned HLVDEN:1
 unsigned IVRST:1
 }
 }

```

        unsigned __pad0__:1
        unsigned VDIRMAG:1
    }
    struct {
        unsigned HLVDL0:1
        unsigned HLVDL1:1
        unsigned HLVDL2:1
        unsigned HLVDL3:1
    }
    struct {
        unsigned LVDL0:1
        unsigned LVDL1:1
        unsigned LVDL2:1
        unsigned LVDL3:1
        unsigned LVDEN:1
        unsigned IRVST:1
    }
    struct {
        unsigned LVV0:1
        unsigned LVV1:1
        unsigned LVV2:1
        unsigned LVV3:1
        unsigned __pad0__:1
        unsigned BGST:1
    }
}
HLVDCONbits

```

- volatile near unsigned char LVDCON

```

union {
    struct {
        unsigned HLVDL:4
        unsigned HLVDEN:1
        unsigned IRVST:1
        unsigned __pad0__:1
        unsigned VDIRMAG:1
    }
    struct {
        unsigned HLVDL0:1
        unsigned HLVDL1:1
        unsigned HLVDL2:1
        unsigned HLVDL3:1
    }
    struct {
        unsigned LVDL0:1
        unsigned LVDL1:1
        unsigned LVDL2:1
        unsigned LVDL3:1
        unsigned LVDEN:1
        unsigned IRVST:1
    }
    struct {
        unsigned LVV0:1
        unsigned LVV1:1
        unsigned LVV2:1
        unsigned LVV3:1
        unsigned __pad0__:1
        unsigned BGST:1
    }
}

```

```

} LVDCONbits

• volatile near unsigned char OSCCON
• union {
    struct {
        unsigned SCS:2
        unsigned IOFS:1
        unsigned OSTS:1
        unsigned IRCF:3
        unsigned IDLEN:1
    }
    struct {
        unsigned SCS0:1
        unsigned SCS1:1
        unsigned FLTS:1
        unsigned __pad0__:1
        unsigned IRCF0:1
        unsigned IRCF1:1
        unsigned IRCF2:1
    }
}
} OSCCONbits

• volatile near unsigned char T0CON
• union {
    struct {
        unsigned T0PS:3
        unsigned PSA:1
        unsigned T0SE:1
        unsigned T0CS:1
        unsigned T08BIT:1
        unsigned TMR0ON:1
    }
    struct {
        unsigned T0PS0:1
        unsigned T0PS1:1
        unsigned T0PS2:1
        unsigned T0PS3:1
        unsigned __pad0__:2
        unsigned T016BIT:1
    }
}
} T0CONbits

• volatile near unsigned char TMR0L
• volatile near unsigned char TMR0H
• near unsigned char STATUS
• struct {
    unsigned C:1
    unsigned DC:1
    unsigned Z:1
    unsigned OV:1
    unsigned N:1
}
} STATUSbits

• near unsigned FSR2
• near unsigned char FSR2L
• near unsigned char FSR2H
• volatile near unsigned char PLUSW2
• volatile near unsigned char PREINC2

```

- volatile near unsigned char POSTDEC2
- volatile near unsigned char POSTINC2
- near unsigned char INDF2
- near unsigned char BSR
- near unsigned FSR1
- near unsigned char FSR1L
- near unsigned char FSR1H
- volatile near unsigned char PLUSW1
- volatile near unsigned char PREINC1
- volatile near unsigned char POSTDEC1
- volatile near unsigned char POSTINC1
- near unsigned char INDF1
- near unsigned char WREG
- near unsigned FSR0
- near unsigned char FSR0L
- near unsigned char FSR0H
- volatile near unsigned char PLUSW0
- volatile near unsigned char PREINC0
- volatile near unsigned char POSTDECO
- volatile near unsigned char POSTINC0
- near unsigned char INDF0
- volatile near unsigned char INTCON3
- union {


```
        struct {
          unsigned INT1IF:1;
          unsigned INT2IF:1;
          unsigned __pad0__:1;
          unsigned INT1IE:1;
          unsigned INT2IE:1;
          unsigned __pad1__:1;
          unsigned INT1IP:1;
          unsigned INT2IP:1;
        }
        struct {
          unsigned INT1F:1;
          unsigned INT2F:1;
          unsigned __pad0__:1;
          unsigned INT1E:1;
          unsigned INT2E:1;
          unsigned __pad1__:1;
          unsigned INT1P:1;
          unsigned INT2P:1;
        }
      } INTCON3bits
```
- volatile near unsigned char INTCON2
- union {


```
        struct {
          unsigned RBIP:1;
          unsigned __pad0__:1;
          unsigned TMR0IP:1;
          unsigned __pad1__:1;
          unsigned INTEDG2:1;
          unsigned INTEDG1:1;
          unsigned INTEDG0:1;
          unsigned NOT_RBPU:1;
        }
```

```
struct {
    unsigned __pad0__:7
    unsigned RBPU:1
}
} INTCON2bits
```

- volatile near unsigned char INTCON
- union {
 struct {
 unsigned RBIF:1
 unsigned INTOIF:1
 unsigned TMR0IF:1
 unsigned RBIE:1
 unsigned INTOIE:1
 unsigned TMR0IE:1
 unsigned PEIE_GIEL:1
 unsigned GIE_GIEH:1
 }
 struct {
 unsigned __pad0__:1
 unsigned INT0F:1
 unsigned TOIF:1
 unsigned __pad1__:1
 unsigned INT0E:1
 unsigned TOIE:1
 unsigned PEIE:1
 unsigned GIE:1
 }
 struct {
 unsigned __pad0__:6
 unsigned GIEL:1
 unsigned GIEH:1
 }
 } INTCONbits
- near unsigned PROD
- near unsigned char PRODL
- near unsigned char PRODH
- volatile near unsigned char TABLAT
- volatile near unsigned short long TBLPTR
- volatile near unsigned char TBLPTRL
- volatile near unsigned char TBLPTRH
- volatile near unsigned char TBLPTRU
- volatile near unsigned short long PC
- volatile near unsigned char PCL
- volatile near unsigned char PCLATH
- volatile near unsigned char PCLATU
- volatile near unsigned char STKPTR
- union {
 struct {
 unsigned STKPTR:5
 unsigned __pad0__:1
 unsigned STKUNF:1
 unsigned STKFUL:1
 }
 struct {
 unsigned SP0:1
 }
 }

```

    unsigned SP1:1
    unsigned SP2:1
    unsigned SP3:1
        unsigned SP4:1
        unsigned __pad0__:2
        unsigned STKOVF:1
    }
} STKPTRbits

```

- near unsigned short long TOS
- near unsigned char TOSL
- near unsigned char TOSH
- near unsigned char TOSU

5.19.1 Macro Definition Documentation

5.19.1.1 #define ACCESS 0

5.19.1.2 #define BANKED 1

5.19.1.3 #define ClrWdt() {__asm clrvwdt __endasm}

5.19.1.4 #define INTSAVELOCS TBLPTR, TABLAT, PROD

5.19.1.5 #define Nop() {__asm nop __endasm}

5.19.1.6 #define Reset() {__asm reset __endasm}

5.19.1.7 #define Rlcf(f, dest, access) {__asm movlb f rlc f,dest,access __endasm}

5.19.1.8 #define Rlncf(f, dest, access) {__asm movlb f rlnc f,dest,access __endasm}

5.19.1.9 #define Rrcf(f, dest, access) {__asm movlb f rrc f,dest,access __endasm}

5.19.1.10 #define Rrncf(f, dest, access) {__asm movlb f rrnc f,dest,access __endasm}

5.19.1.11 #define Sleep() {__asm sleep __endasm}

5.19.1.12 #define Swapf(f, dest, access) {__asm movlb f swap f,dest,access __endasm }

5.19.2 Variable Documentation

5.19.2.1 unsigned __pad0__

5.19.2.2 unsigned __pad1__

5.19.2.3 unsigned A

5.19.2.4 unsigned ABDEN

5.19.2.5 unsigned ABDOVF

5.19.2.6 unsigned ACKDT

5.19.2.7 unsigned ACKEN

- 5.19.2.8 unsigned ACKSTAT
- 5.19.2.9 unsigned ACQT
- 5.19.2.10 unsigned ACQTO
- 5.19.2.11 unsigned ACQT1
- 5.19.2.12 unsigned ACQT2
- 5.19.2.13 volatile near unsigned char ADCON0
- 5.19.2.14 volatile { ... } ADCON0bits
- 5.19.2.15 volatile near unsigned char ADCON1
- 5.19.2.16 volatile { ... } ADCON1bits
- 5.19.2.17 volatile near unsigned char ADCON2
- 5.19.2.18 volatile { ... } ADCON2bits
- 5.19.2.19 unsigned ADCS
- 5.19.2.20 unsigned ADCS0
- 5.19.2.21 unsigned ADCS1
- 5.19.2.22 unsigned ADCS2
- 5.19.2.23 unsigned ADDEN
- 5.19.2.24 unsigned ADEN
- 5.19.2.25 unsigned ADFM
- 5.19.2.26 unsigned ADIE
- 5.19.2.27 unsigned ADIF
- 5.19.2.28 unsigned ADIP
- 5.19.2.29 unsigned ADMSK1
- 5.19.2.30 unsigned ADMSK2
- 5.19.2.31 unsigned ADMSK3
- 5.19.2.32 unsigned ADMSK4
- 5.19.2.33 unsigned ADMSK5
- 5.19.2.34 unsigned ADON
- 5.19.2.35 volatile near unsigned ADRES

5.19.2.36 volatile near unsigned char ADRESH

5.19.2.37 volatile near unsigned char ADRESL

5.19.2.38 unsigned AN0

5.19.2.39 unsigned AN1

5.19.2.40 unsigned AN10

5.19.2.41 unsigned AN11

5.19.2.42 unsigned AN12

5.19.2.43 unsigned AN2

5.19.2.44 unsigned AN3

5.19.2.45 unsigned AN4

5.19.2.46 unsigned AN5

5.19.2.47 unsigned AN6

5.19.2.48 unsigned AN7

5.19.2.49 unsigned AN8

5.19.2.50 unsigned AN9

5.19.2.51 volatile near unsigned char BAUDCON

5.19.2.52 volatile { ... } BAUDCONbits

5.19.2.53 volatile near unsigned char BAUDCTL

5.19.2.54 volatile { ... } BAUDCTLbits

5.19.2.55 unsigned BCLIE

5.19.2.56 unsigned BCLIF

5.19.2.57 unsigned BCLIP

5.19.2.58 unsigned BF

5.19.2.59 unsigned BGST

5.19.2.60 unsigned BOR

5.19.2.61 unsigned BRG16

5.19.2.62 unsigned BRGH

5.19.2.63 near unsigned char BSR

5.19.2.64 unsigned C
5.19.2.65 unsigned C1INV
5.19.2.66 unsigned C1OUT
5.19.2.67 unsigned C2INV
5.19.2.68 unsigned C2OUT
5.19.2.69 unsigned CCP1
5.19.2.70 volatile near unsigned char CCP1CON
5.19.2.71 volatile { ... } CCP1CONbits
5.19.2.72 unsigned CCP1IE
5.19.2.73 unsigned CCP1IF
5.19.2.74 unsigned CCP1IP
5.19.2.75 unsigned CCP1M
5.19.2.76 unsigned CCP1M0
5.19.2.77 unsigned CCP1M1
5.19.2.78 unsigned CCP1M2
5.19.2.79 unsigned CCP1M3
5.19.2.80 unsigned CCP1X
5.19.2.81 unsigned CCP1Y
5.19.2.82 unsigned CCP2
5.19.2.83 volatile near unsigned char CCP2CON
5.19.2.84 volatile { ... } CCP2CONbits
5.19.2.85 unsigned CCP2IE
5.19.2.86 unsigned CCP2IF
5.19.2.87 unsigned CCP2IP
5.19.2.88 unsigned CCP2M
5.19.2.89 unsigned CCP2M0
5.19.2.90 unsigned CCP2M1
5.19.2.91 unsigned CCP2M2

5.19.2.92 unsigned CCP2M3
5.19.2.93 unsigned CCP2X
5.19.2.94 unsigned CCP2Y
5.19.2.95 volatile near unsigned CCPR1
5.19.2.96 volatile near unsigned char CCPR1H
5.19.2.97 volatile near unsigned char CCPR1L
5.19.2.98 volatile near unsigned CCPR2
5.19.2.99 volatile near unsigned char CCPR2H
5.19.2.100 volatile near unsigned char CCPR2L
5.19.2.101 unsigned CFGS
5.19.2.102 unsigned CHS
5.19.2.103 unsigned CHS0
5.19.2.104 unsigned CHS1
5.19.2.105 unsigned CHS2
5.19.2.106 unsigned CHS3
5.19.2.107 unsigned CIS
5.19.2.108 unsigned CK
5.19.2.109 unsigned CKE
5.19.2.110 unsigned CKP
5.19.2.111 unsigned CLKI
5.19.2.112 unsigned CLKO
5.19.2.113 unsigned CM
5.19.2.114 unsigned CM0
5.19.2.115 unsigned CM1
5.19.2.116 unsigned CM2
5.19.2.117 volatile near unsigned char CMCON
5.19.2.118 volatile { ... } CMCONbits
5.19.2.119 unsigned CMIE

5.19.2.120 unsigned CMIF
5.19.2.121 unsigned CMIP
5.19.2.122 unsigned CREN
5.19.2.123 unsigned CS
5.19.2.124 unsigned CSRC
5.19.2.125 unsigned CVR
5.19.2.126 unsigned CVR0
5.19.2.127 unsigned CVR1
5.19.2.128 unsigned CVR2
5.19.2.129 unsigned CVR3
5.19.2.130 volatile near unsigned char CVRCON
5.19.2.131 volatile { ... } CVRCONbits
5.19.2.132 unsigned CVREF
5.19.2.133 unsigned CVREN
5.19.2.134 unsigned CVROE
5.19.2.135 unsigned CVRR
5.19.2.136 unsigned CVRSS
5.19.2.137 unsigned D
5.19.2.138 unsigned D_A
5.19.2.139 unsigned D_NOT_A
5.19.2.140 unsigned DC
5.19.2.141 unsigned DC1B
5.19.2.142 unsigned DC1B0
5.19.2.143 unsigned DC1B1
5.19.2.144 unsigned DC2B
5.19.2.145 unsigned DC2B0
5.19.2.146 unsigned DC2B1
5.19.2.147 volatile near unsigned char DDRA

5.19.2.148 volatile { ... } DDRAbits
5.19.2.149 volatile near unsigned char DDRB
5.19.2.150 volatile { ... } DDRBbits
5.19.2.151 volatile near unsigned char DDRC
5.19.2.152 volatile { ... } DDRCbits
5.19.2.153 volatile near unsigned char DDRD
5.19.2.154 volatile { ... } DDRDbits
5.19.2.155 volatile near unsigned char DDRE
5.19.2.156 volatile { ... } DDREbits
5.19.2.157 unsigned DONE
5.19.2.158 unsigned DT
5.19.2.159 volatile near unsigned char ECCP1AS
5.19.2.160 volatile { ... } ECCP1ASbits
5.19.2.161 volatile near unsigned char ECCP1DEL
5.19.2.162 volatile { ... } ECCP1DELbits
5.19.2.163 volatile near unsigned char ECCPAS
5.19.2.164 unsigned ECCPAS0
5.19.2.165 unsigned ECCPAS1
5.19.2.166 unsigned ECCPAS2
5.19.2.167 volatile { ... } ECCPASbits
5.19.2.168 unsigned ECCPASE
5.19.2.169 volatile near unsigned char EEADR
5.19.2.170 volatile near unsigned char EECON1
5.19.2.171 volatile { ... } EECON1bits
5.19.2.172 volatile near unsigned char EECON2
5.19.2.173 volatile near unsigned char EEDATA
5.19.2.174 unsigned EEIE
5.19.2.175 unsigned EEIF

5.19.2.176 unsigned EEIP
5.19.2.177 unsigned EEPGD
5.19.2.178 unsigned FERR
5.19.2.179 unsigned FLT0
5.19.2.180 unsigned FLTS
5.19.2.181 unsigned FREE
5.19.2.182 near unsigned FSR0
5.19.2.183 near unsigned char FSR0H
5.19.2.184 near unsigned char FSR0L
5.19.2.185 near unsigned FSR1
5.19.2.186 near unsigned char FSR1H
5.19.2.187 near unsigned char FSR1L
5.19.2.188 near unsigned FSR2
5.19.2.189 near unsigned char FSR2H
5.19.2.190 near unsigned char FSR2L
5.19.2.191 unsigned GCEN
5.19.2.192 unsigned GIE
5.19.2.193 unsigned GIE_GIEH
5.19.2.194 unsigned GIEH
5.19.2.195 unsigned GIEL
5.19.2.196 unsigned GO
5.19.2.197 unsigned GO_DONE
5.19.2.198 unsigned GO_NOT_DONE
5.19.2.199 volatile near unsigned char HLVDCON
5.19.2.200 volatile { ... } HLVDCONbits
5.19.2.201 unsigned HLVDEN
5.19.2.202 unsigned HLVDIE
5.19.2.203 unsigned HLVDIF

5.19.2.204 unsigned HLV DIN
5.19.2.205 unsigned HLV DIP
5.19.2.206 unsigned HLV DL
5.19.2.207 unsigned HLV DL0
5.19.2.208 unsigned HLV DL1
5.19.2.209 unsigned HLV DL2
5.19.2.210 unsigned HLV DL3
5.19.2.211 unsigned IBF
5.19.2.212 unsigned IBOV
5.19.2.213 unsigned IDLEN
5.19.2.214 near unsigned char INDF0
5.19.2.215 near unsigned char INDF1
5.19.2.216 near unsigned char INDF2
5.19.2.217 unsigned INT0
5.19.2.218 unsigned INT0E
5.19.2.219 unsigned INT0F
5.19.2.220 unsigned INT0IE
5.19.2.221 unsigned INT0IF
5.19.2.222 unsigned INT1
5.19.2.223 unsigned INT1E
5.19.2.224 unsigned INT1F
5.19.2.225 unsigned INT1IE
5.19.2.226 unsigned INT1IF
5.19.2.227 unsigned INT1IP
5.19.2.228 unsigned INT1P
5.19.2.229 unsigned INT2
5.19.2.230 unsigned INT2E
5.19.2.231 unsigned INT2F

5.19.2.232 unsigned INT2IE
5.19.2.233 unsigned INT2IF
5.19.2.234 unsigned INT2IP
5.19.2.235 unsigned INT2P
5.19.2.236 volatile near unsigned char INTCON
5.19.2.237 volatile near unsigned char INTCON2
5.19.2.238 volatile { ... } INTCON2bits
5.19.2.239 volatile near unsigned char INTCON3
5.19.2.240 volatile { ... } INTCON3bits
5.19.2.241 volatile { ... } INTCONbits
5.19.2.242 unsigned INTEDG0
5.19.2.243 unsigned INTEDG1
5.19.2.244 unsigned INTEDG2
5.19.2.245 unsigned INTSRC
5.19.2.246 unsigned IOFS
5.19.2.247 unsigned IPEN
5.19.2.248 volatile near unsigned char IPR1
5.19.2.249 volatile { ... } IPR1bits
5.19.2.250 volatile near unsigned char IPR2
5.19.2.251 volatile { ... } IPR2bits
5.19.2.252 unsigned IRCF
5.19.2.253 unsigned IRCFO
5.19.2.254 unsigned IRCF1
5.19.2.255 unsigned IRCF2
5.19.2.256 unsigned IRVST
5.19.2.257 unsigned IVRST
5.19.2.258 unsigned KBI0
5.19.2.259 unsigned KBI1

5.19.2.260 unsigned KBI2
5.19.2.261 unsigned KBI3
5.19.2.262 volatile near unsigned char LATA
5.19.2.263 unsigned LATA0
5.19.2.264 unsigned LATA1
5.19.2.265 unsigned LATA2
5.19.2.266 unsigned LATA3
5.19.2.267 unsigned LATA4
5.19.2.268 unsigned LATA5
5.19.2.269 unsigned LATA6
5.19.2.270 unsigned LATA7
5.19.2.271 volatile { ... } LATAbits
5.19.2.272 volatile near unsigned char LATB
5.19.2.273 unsigned LATB0
5.19.2.274 unsigned LATB1
5.19.2.275 unsigned LATB2
5.19.2.276 unsigned LATB3
5.19.2.277 unsigned LATB4
5.19.2.278 unsigned LATB5
5.19.2.279 unsigned LATB6
5.19.2.280 unsigned LATB7
5.19.2.281 volatile { ... } LATBbits
5.19.2.282 volatile near unsigned char LATC
5.19.2.283 unsigned LATC0
5.19.2.284 unsigned LATC1
5.19.2.285 unsigned LATC2
5.19.2.286 unsigned LATC3
5.19.2.287 unsigned LATC4

5.19.2.288 unsigned LATC5

5.19.2.289 unsigned LATC6

5.19.2.290 unsigned LATC7

5.19.2.291 volatile { ... } LATCbits

5.19.2.292 volatile near unsigned char LATD

5.19.2.293 unsigned LATD0

5.19.2.294 unsigned LATD1

5.19.2.295 unsigned LATD2

5.19.2.296 unsigned LATD3

5.19.2.297 unsigned LATD4

5.19.2.298 unsigned LATD5

5.19.2.299 unsigned LATD6

5.19.2.300 unsigned LATD7

5.19.2.301 volatile { ... } LATDbits

5.19.2.302 volatile near unsigned char LATE

5.19.2.303 unsigned LATE0

5.19.2.304 unsigned LATE1

5.19.2.305 unsigned LATE2

5.19.2.306 volatile { ... } LATEbits

5.19.2.307 volatile near unsigned char LVDCON

5.19.2.308 volatile { ... } LVDCONbits

5.19.2.309 unsigned LVDEN

5.19.2.310 unsigned LVDIE

5.19.2.311 unsigned LVDIF

5.19.2.312 unsigned LVDIN

5.19.2.313 unsigned LVDIP

5.19.2.314 unsigned LVDL0

5.19.2.315 unsigned LVDL1

5.19.2.316 unsigned LVDL2
5.19.2.317 unsigned LVDL3
5.19.2.318 unsigned LVV0
5.19.2.319 unsigned LVV1
5.19.2.320 unsigned LVV2
5.19.2.321 unsigned LVV3
5.19.2.322 unsigned MCLR
5.19.2.323 unsigned N
5.19.2.324 unsigned NOT_A
5.19.2.325 unsigned NOT_ADDRESS
5.19.2.326 unsigned NOT_BOR
5.19.2.327 unsigned NOT_CS
5.19.2.328 unsigned NOT_DONE
5.19.2.329 unsigned NOT_MCLR
5.19.2.330 unsigned NOT_PD
5.19.2.331 unsigned NOT_POR
5.19.2.332 unsigned NOT_RBPU
5.19.2.333 unsigned NOT_RD
5.19.2.334 unsigned NOT_RI
5.19.2.335 unsigned NOT_SS
5.19.2.336 unsigned NOT_T1SYNC
5.19.2.337 unsigned NOT_T3SYNC
5.19.2.338 unsigned NOT_TO
5.19.2.339 unsigned NOT_W
5.19.2.340 unsigned NOT_WR
5.19.2.341 unsigned NOT_WRITE
5.19.2.342 unsigned OBF
5.19.2.343 unsigned OERR

5.19.2.344 unsigned OSC1
5.19.2.345 unsigned OSC2
5.19.2.346 volatile near unsigned char OSCCON
5.19.2.347 volatile { ... } OSCCONbits
5.19.2.348 unsigned OSCFIE
5.19.2.349 unsigned OSCFIF
5.19.2.350 unsigned OSCFIP
5.19.2.351 volatile near unsigned char OSCTUNE
5.19.2.352 volatile { ... } OSCTUNEbits
5.19.2.353 unsigned OSTS
5.19.2.354 unsigned OV
5.19.2.355 unsigned P
5.19.2.356 unsigned P1A
5.19.2.357 unsigned P1B
5.19.2.358 unsigned P1C
5.19.2.359 unsigned P1D
5.19.2.360 unsigned P1M
5.19.2.361 unsigned P1M0
5.19.2.362 unsigned P1M1
5.19.2.363 volatile near unsigned short long PC
5.19.2.364 unsigned PCFG
5.19.2.365 unsigned PCFG0
5.19.2.366 unsigned PCFG1
5.19.2.367 unsigned PCFG2
5.19.2.368 unsigned PCFG3
5.19.2.369 volatile near unsigned char PCL
5.19.2.370 volatile near unsigned char PCLATH
5.19.2.371 volatile near unsigned char PCLATU

5.19.2.372 unsigned PD
5.19.2.373 unsigned PDC
5.19.2.374 unsigned PDC0
5.19.2.375 unsigned PDC1
5.19.2.376 unsigned PDC2
5.19.2.377 unsigned PDC3
5.19.2.378 unsigned PDC4
5.19.2.379 unsigned PDC5
5.19.2.380 unsigned PDC6
5.19.2.381 unsigned PEIE
5.19.2.382 unsigned PEIE_GIEL
5.19.2.383 unsigned PEN
5.19.2.384 unsigned PGC
5.19.2.385 unsigned PGD
5.19.2.386 unsigned PGM
5.19.2.387 volatile near unsigned char PIE1
5.19.2.388 volatile { ... } PIE1bits
5.19.2.389 volatile near unsigned char PIE2
5.19.2.390 volatile { ... } PIE2bits
5.19.2.391 volatile near unsigned char PIR1
5.19.2.392 volatile { ... } PIR1bits
5.19.2.393 volatile near unsigned char PIR2
5.19.2.394 volatile { ... } PIR2bits
5.19.2.395 unsigned PLLEN
5.19.2.396 volatile near unsigned char PLUSW0
5.19.2.397 volatile near unsigned char PLUSW1
5.19.2.398 volatile near unsigned char PLUSW2
5.19.2.399 unsigned POR

- 5.19.2.400 volatile near unsigned char PORTA
- 5.19.2.401 volatile { ... } PORTAbits
- 5.19.2.402 volatile near unsigned char PORTB
- 5.19.2.403 volatile { ... } PORTBbits
- 5.19.2.404 volatile near unsigned char PORTC
- 5.19.2.405 volatile { ... } PORTCbits
- 5.19.2.406 volatile near unsigned char PORTD
- 5.19.2.407 volatile { ... } PORTDbits
- 5.19.2.408 volatile near unsigned char PORTE
- 5.19.2.409 volatile { ... } PORTEbits
- 5.19.2.410 volatile near unsigned char POSTDEC0
- 5.19.2.411 volatile near unsigned char POSTDEC1
- 5.19.2.412 volatile near unsigned char POSTDEC2
- 5.19.2.413 volatile near unsigned char POSTINC0
- 5.19.2.414 volatile near unsigned char POSTINC1
- 5.19.2.415 volatile near unsigned char POSTINC2
- 5.19.2.416 volatile near unsigned char PR2
- 5.19.2.417 volatile near unsigned char PREINC0
- 5.19.2.418 volatile near unsigned char PREINC1
- 5.19.2.419 volatile near unsigned char PREINC2
- 5.19.2.420 near unsigned PROD
- 5.19.2.421 near unsigned char PRODH
- 5.19.2.422 near unsigned char PRODL
- 5.19.2.423 unsigned PRSEN
- 5.19.2.424 unsigned PSA
- 5.19.2.425 unsigned PSP0
- 5.19.2.426 unsigned PSP1
- 5.19.2.427 unsigned PSP2

5.19.2.428 unsigned PSP3
5.19.2.429 unsigned PSP4
5.19.2.430 unsigned PSP5
5.19.2.431 unsigned PSP6
5.19.2.432 unsigned PSP7
5.19.2.433 unsigned PSPIE
5.19.2.434 unsigned PSPIF
5.19.2.435 unsigned PSPIP
5.19.2.436 unsigned PSPMODE
5.19.2.437 unsigned PSSAC
5.19.2.438 unsigned PSSAC0
5.19.2.439 unsigned PSSAC1
5.19.2.440 unsigned PSSBD
5.19.2.441 unsigned PSSBD0
5.19.2.442 unsigned PSSBD1
5.19.2.443 volatile near unsigned char PWM1CON
5.19.2.444 volatile { ... } PWM1CONbits
5.19.2.445 unsigned R
5.19.2.446 unsigned R_NOT_W
5.19.2.447 unsigned R_W
5.19.2.448 unsigned RA0
5.19.2.449 unsigned RA1
5.19.2.450 unsigned RA2
5.19.2.451 unsigned RA3
5.19.2.452 unsigned RA4
5.19.2.453 unsigned RA5
5.19.2.454 unsigned RA6
5.19.2.455 unsigned RA7

5.19.2.456 unsigned RB0
5.19.2.457 unsigned RB1
5.19.2.458 unsigned RB2
5.19.2.459 unsigned RB3
5.19.2.460 unsigned RB4
5.19.2.461 unsigned RB5
5.19.2.462 unsigned RB6
5.19.2.463 unsigned RB7
5.19.2.464 unsigned RBIE
5.19.2.465 unsigned RBIF
5.19.2.466 unsigned RBIP
5.19.2.467 unsigned RBPU
5.19.2.468 unsigned RC0
5.19.2.469 unsigned RC1
5.19.2.470 unsigned RC2
5.19.2.471 unsigned RC3
5.19.2.472 unsigned RC4
5.19.2.473 unsigned RC5
5.19.2.474 unsigned RC6
5.19.2.475 unsigned RC7
5.19.2.476 unsigned RCEN
5.19.2.477 unsigned RCIDL
5.19.2.478 unsigned RCIE
5.19.2.479 unsigned RCIF
5.19.2.480 unsigned RCIP
5.19.2.481 unsigned RCMT
5.19.2.482 volatile near unsigned char RCON
5.19.2.483 volatile { ... } RCONbits

5.19.2.484 volatile near unsigned char RCREG

5.19.2.485 volatile near unsigned char RCSTA

5.19.2.486 volatile { ... } RCSTAbits

5.19.2.487 unsigned RD

5.19.2.488 unsigned RD0

5.19.2.489 unsigned RD1

5.19.2.490 unsigned RD16

5.19.2.491 unsigned RD2

5.19.2.492 unsigned RD3

5.19.2.493 unsigned RD4

5.19.2.494 unsigned RD5

5.19.2.495 unsigned RD6

5.19.2.496 unsigned RD7

5.19.2.497 unsigned RE0

5.19.2.498 unsigned RE1

5.19.2.499 unsigned RE2

5.19.2.500 unsigned RE3

5.19.2.501 unsigned RI

5.19.2.502 unsigned RSEN

5.19.2.503 unsigned RX

5.19.2.504 unsigned RX9

5.19.2.505 unsigned RX9D

5.19.2.506 unsigned RXDTP

5.19.2.507 unsigned S

5.19.2.508 unsigned SBOREN

5.19.2.509 unsigned SCK

5.19.2.510 unsigned SCKP

5.19.2.511 unsigned SCL

5.19.2.512 unsigned SCS
5.19.2.513 unsigned SCS0
5.19.2.514 unsigned SCS1
5.19.2.515 unsigned SDA
5.19.2.516 unsigned SDI
5.19.2.517 unsigned SDO
5.19.2.518 unsigned SEN
5.19.2.519 unsigned SENDB
5.19.2.520 unsigned SMP
5.19.2.521 unsigned SP0
5.19.2.522 unsigned SP1
5.19.2.523 unsigned SP2
5.19.2.524 unsigned SP3
5.19.2.525 unsigned SP4
5.19.2.526 volatile near unsigned char SPBRG
5.19.2.527 volatile near unsigned char SPBRGH
5.19.2.528 unsigned SPEN
5.19.2.529 unsigned SREN
5.19.2.530 unsigned SS
5.19.2.531 volatile near unsigned char SSPADD
5.19.2.532 volatile near unsigned char SSPBUF
5.19.2.533 volatile near unsigned char SSPCON1
5.19.2.534 volatile { ... } SSPCON1bits
5.19.2.535 volatile near unsigned char SSPCON2
5.19.2.536 volatile { ... } SSPCON2bits
5.19.2.537 unsigned SSPEN
5.19.2.538 unsigned SSPIE
5.19.2.539 unsigned SSPIF

5.19.2.540 unsigned SSPIP
5.19.2.541 unsigned SSPM
5.19.2.542 unsigned SSPMO
5.19.2.543 unsigned SSPM1
5.19.2.544 unsigned SSPM2
5.19.2.545 unsigned SSPM3
5.19.2.546 unsigned SSPOV
5.19.2.547 volatile near unsigned char SSPSTAT
5.19.2.548 volatile { ... } SSPSTATbits
5.19.2.549 near unsigned char STATUS
5.19.2.550 near { ... } STATUSbits
5.19.2.551 unsigned STKFUL
5.19.2.552 unsigned STKOVF
5.19.2.553 unsigned STKPTR
5.19.2.554 volatile { ... } STKPTRbits
5.19.2.555 unsigned STKUNF
5.19.2.556 unsigned SWDTE
5.19.2.557 unsigned SWDTEN
5.19.2.558 unsigned SYNC
5.19.2.559 unsigned T016BIT
5.19.2.560 unsigned T08BIT
5.19.2.561 unsigned T0CKI
5.19.2.562 volatile near unsigned char T0CON
5.19.2.563 volatile { ... } T0CONbits
5.19.2.564 unsigned T0CS
5.19.2.565 unsigned T0IE
5.19.2.566 unsigned T0IF
5.19.2.567 unsigned T0PS

5.19.2.568 unsigned T0PS0
5.19.2.569 unsigned T0PS1
5.19.2.570 unsigned T0PS2
5.19.2.571 unsigned T0PS3
5.19.2.572 unsigned T0SE
5.19.2.573 unsigned T13CKI
5.19.2.574 unsigned T1CKI
5.19.2.575 unsigned T1CKPS
5.19.2.576 unsigned T1CKPS0
5.19.2.577 unsigned T1CKPS1
5.19.2.578 volatile near unsigned char T1CON
5.19.2.579 volatile { ... } T1CONbits
5.19.2.580 unsigned T1OSCEN
5.19.2.581 unsigned T1OSI
5.19.2.582 unsigned T1OSO
5.19.2.583 unsigned T1RUN
5.19.2.584 unsigned T1SYNC
5.19.2.585 unsigned T2CKPS
5.19.2.586 unsigned T2CKPS0
5.19.2.587 unsigned T2CKPS1
5.19.2.588 volatile near unsigned char T2CON
5.19.2.589 volatile { ... } T2CONbits
5.19.2.590 unsigned T2OUTPS
5.19.2.591 unsigned T2OUTPS0
5.19.2.592 unsigned T2OUTPS1
5.19.2.593 unsigned T2OUTPS2
5.19.2.594 unsigned T2OUTPS3
5.19.2.595 unsigned T3CCP1

5.19.2.596 unsigned T3CCP2
5.19.2.597 unsigned T3CKPS
5.19.2.598 unsigned T3CKPS0
5.19.2.599 unsigned T3CKPS1
5.19.2.600 volatile near unsigned char T3CON
5.19.2.601 volatile { ... } T3CONbits
5.19.2.602 unsigned T3SYNC
5.19.2.603 volatile near unsigned char TABLAT
5.19.2.604 volatile near unsigned short long TBLPTR
5.19.2.605 volatile near unsigned char TBLPTRH
5.19.2.606 volatile near unsigned char TBLPTRL
5.19.2.607 volatile near unsigned char TBLPTRU
5.19.2.608 volatile near unsigned char TMROH
5.19.2.609 unsigned TMROIE
5.19.2.610 unsigned TMROIF
5.19.2.611 unsigned TMROIP
5.19.2.612 volatile near unsigned char TMROL
5.19.2.613 unsigned TMROON
5.19.2.614 unsigned TMR1CS
5.19.2.615 volatile near unsigned char TMR1H
5.19.2.616 unsigned TMR1IE
5.19.2.617 unsigned TMR1IF
5.19.2.618 unsigned TMR1IP
5.19.2.619 volatile near unsigned char TMR1L
5.19.2.620 unsigned TMR1ON
5.19.2.621 volatile near unsigned char TMR2
5.19.2.622 unsigned TMR2IE
5.19.2.623 unsigned TMR2IF

5.19.2.624 unsigned TMR2IP
5.19.2.625 unsigned TMR2ON
5.19.2.626 unsigned TMR3CS
5.19.2.627 volatile near unsigned char TMR3H
5.19.2.628 unsigned TMR3IE
5.19.2.629 unsigned TMR3IF
5.19.2.630 unsigned TMR3IP
5.19.2.631 volatile near unsigned char TMR3L
5.19.2.632 unsigned TMR3ON
5.19.2.633 unsigned TO
5.19.2.634 near unsigned short long TOS
5.19.2.635 near unsigned char TOSH
5.19.2.636 near unsigned char TOSL
5.19.2.637 near unsigned char TOSU
5.19.2.638 unsigned TOUTPS0
5.19.2.639 unsigned TOUTPS1
5.19.2.640 unsigned TOUTPS2
5.19.2.641 unsigned TOUTPS3
5.19.2.642 volatile near unsigned char TRISA
5.19.2.643 unsigned TRISA0
5.19.2.644 unsigned TRISA1
5.19.2.645 unsigned TRISA2
5.19.2.646 unsigned TRISA3
5.19.2.647 unsigned TRISA4
5.19.2.648 unsigned TRISA5
5.19.2.649 unsigned TRISA6
5.19.2.650 unsigned TRISA7
5.19.2.651 volatile { ... } TRISAbits

5.19.2.652 volatile near unsigned char TRISB

5.19.2.653 unsigned TRISB0

5.19.2.654 unsigned TRISB1

5.19.2.655 unsigned TRISB2

5.19.2.656 unsigned TRISB3

5.19.2.657 unsigned TRISB4

5.19.2.658 unsigned TRISB5

5.19.2.659 unsigned TRISB6

5.19.2.660 unsigned TRISB7

5.19.2.661 volatile { ... } TRISBbits

5.19.2.662 volatile near unsigned char TRISC

5.19.2.663 unsigned TRISC0

5.19.2.664 unsigned TRISC1

5.19.2.665 unsigned TRISC2

5.19.2.666 unsigned TRISC3

5.19.2.667 unsigned TRISC4

5.19.2.668 unsigned TRISC5

5.19.2.669 unsigned TRISC6

5.19.2.670 unsigned TRISC7

5.19.2.671 volatile { ... } TRISCbits

5.19.2.672 volatile near unsigned char TRISD

5.19.2.673 unsigned TRISD0

5.19.2.674 unsigned TRISD1

5.19.2.675 unsigned TRISD2

5.19.2.676 unsigned TRISD3

5.19.2.677 unsigned TRISD4

5.19.2.678 unsigned TRISD5

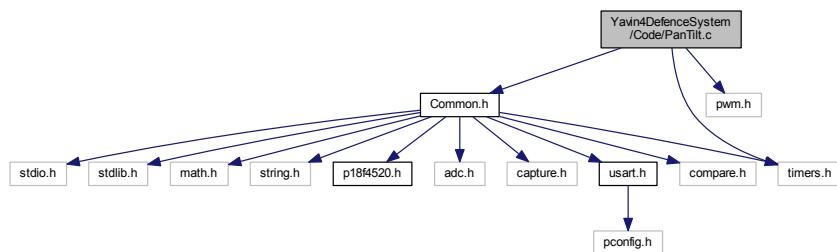
5.19.2.679 unsigned TRISD6

5.19.2.680 unsigned TRISD7
5.19.2.681 volatile { ... } TRISDbits
5.19.2.682 volatile near unsigned char TRISE
5.19.2.683 unsigned TRISE0
5.19.2.684 unsigned TRISE1
5.19.2.685 unsigned TRISE2
5.19.2.686 volatile { ... } TRISEbits
5.19.2.687 unsigned TRMT
5.19.2.688 unsigned TUN
5.19.2.689 unsigned TUN0
5.19.2.690 unsigned TUN1
5.19.2.691 unsigned TUN2
5.19.2.692 unsigned TUN3
5.19.2.693 unsigned TUN4
5.19.2.694 unsigned TX
5.19.2.695 unsigned TX9
5.19.2.696 unsigned TX9D
5.19.2.697 unsigned TXCKP
5.19.2.698 unsigned TXEN
5.19.2.699 unsigned TXIE
5.19.2.700 unsigned TXIF
5.19.2.701 unsigned TXIP
5.19.2.702 volatile near unsigned char TXREG
5.19.2.703 volatile near unsigned char TXSTA
5.19.2.704 volatile { ... } TXSTAbits
5.19.2.705 unsigned UA
5.19.2.706 unsigned VCFG
5.19.2.707 unsigned VCFG0

- 5.19.2.708 unsigned VCFG1
- 5.19.2.709 unsigned VDIRMAG
- 5.19.2.710 unsigned VPP
- 5.19.2.711 unsigned VREFN
- 5.19.2.712 unsigned VREFP
- 5.19.2.713 near unsigned char W
- 5.19.2.714 unsigned WCOL
- 5.19.2.715 volatile near unsigned char WDTCON
- 5.19.2.716 volatile { ... } WDTCONbits
- 5.19.2.717 unsigned WR
- 5.19.2.718 near unsigned char WREG
- 5.19.2.719 unsigned WREN
- 5.19.2.720 unsigned WRERR
- 5.19.2.721 unsigned WUE
- 5.19.2.722 unsigned Z

5.20 Yavin4DefenceSystem/Code/PanTilt.c File Reference

```
#include "Common.h"
#include <pwm.h>
#include <timers.h>
Include dependency graph for PanTilt.c:
```



Data Structures

- struct Delay

Macros

- #define AZ_PWM_PIN PORTCbits.RC0
- #define IN_PWM_PIN PORTCbits.RC1
- #define DUTY_CYCLE_TIME 2500
- #define PWM_PERIOD 50000
- #define PWM_HALF_PERIOD 25000
- #define LATENCY 340
- #define SERVO_INIT() TRISCbits.RC0 = 0; TRISCbits.RC1 = 0; PORTCbits.RC0 = 0; PORTCbits.RC1 = 0

Functions

- static void validate (unsigned int *delay)
- static Direction delay2Direction (Delay dly)
- static Delay direction2Delay (Direction dir)
- void configureBase (void)
- void move (Direction destination)
- void increment (Direction difference)
- void incrementFine (Direction difference)
- Direction getDir (void)
- char getMaxAzimuthAngle (void)
- char getMinAzimuthAngle (void)
- char getMaxElevationAngle (void)
- char getMinElevationAngle (void)
- void setMaxAzimuthAngle (char p_angle)
- void setMinAzimuthAngle (char p_angle)
- void setMaxElevationAngle (char p_angle)
- void setMinElevationAngle (char p_angle)
- void calibratePanTilt (Direction reference)
- void calibratePanTiltRange (Direction reference)
- Direction rawDir (void)
- void panTiltISR (void)
- char updated (void)

Variables

- static Direction calibration_offset = { 0, 3 }
- static Direction arcRange = { 94, 103 }
- static Delay global_delay
- static signed char azimuth_angle_max
- static signed char azimuth_angle_min
- static signed char elevation_angle_max
- static signed char elevation_angle_min
- static Direction current_direction
- static volatile char changed = 0

5.20.1 Macro Definition Documentation

5.20.1.1 `#define AZ_PWM_PIN PORTCbits.RC0`

5.20.1.2 `#define DUTY_CYCLE_TIME 2500`

5.20.1.3 `#define IN_PWM_PIN PORTCbits.RC1`

5.20.1.4 `#define LATENCY 340`

5.20.1.5 `#define PWM_HALF_PERIOD 25000`

5.20.1.6 `#define PWM_PERIOD 50000`

5.20.1.7 `#define SERVO_INIT() TRISCbits.RC0 = 0; TRISCbits.RC1 = 0; PORTCbits.RC0 = 0; PORTCbits.RC1 = 0`

5.20.2 Function Documentation

5.20.2.1 `void calibratePanTilt (Direction reference)`

Function: [calibratePanTilt\(Direction reference\)](#)

Include: [PanTilt.h](#)

Description: Calibrates the pan tile mechanism offset so that any future reference to move to the reference value specified in the function call will move the pan tilt back to the current position.

Arguments: reference - A struct containing the azimuth and inclinaion you wish to define as this position.

Returns: None

5.20.2.2 `void calibratePanTiltRange (Direction reference)`

Function: [calibratePanTiltRange\(Direction reference\)](#)

Include: [PanTilt.h](#)

Description: Calibrates the pan tile mechanism's range so that any future reference to move to the reference value specified in the function call will move the pan tilt back to the current position.

Arguments: reference - A struct containing the azimuth and inclinaion you wish to define as this position.

Returns: None

5.20.2.3 `void configureBase (void)`

Function: [configureBase\(void\)](#)

Include: [PanTilt.h](#)

Description: Configures the Pan Tilt mechanism for operation

Arguments: None

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.20.2.4 static Direction delay2Direction (Delay dly) [static]

Function: direction2Delay(DirectionState dir)

Include: Local to [PanTilt.c](#)

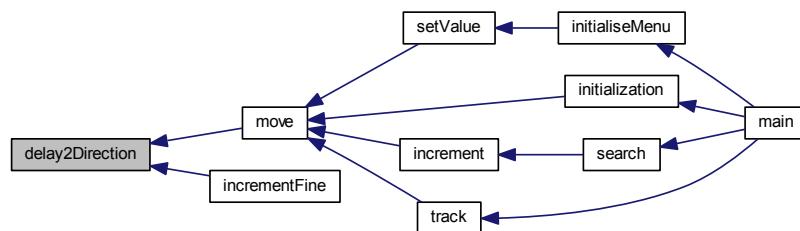
Description: Converts an azimuth and inclination direction into a pwm period

Arguments: dir - Locally defined struct containing the desired Azimuth and Inclination

Returns: [Delay](#) - The required PDM delay to move the servos to the given direction

Remark: This function relies on the `ARC_RANGE` macro being set correctly. This should hold the value of the maximum angle that the servos can be commanded

Here is the caller graph for this function:



5.20.2.5 static Delay direction2Delay (Direction dir) [static]

Function: direction2Delay(DirectionState dir)

Include: Local to [PanTilt.c](#)

Description: Converts an azimuth and inclination direction into a pwm period

Arguments: dir - Locally defined struct containing the desired Azimuth and Inclination

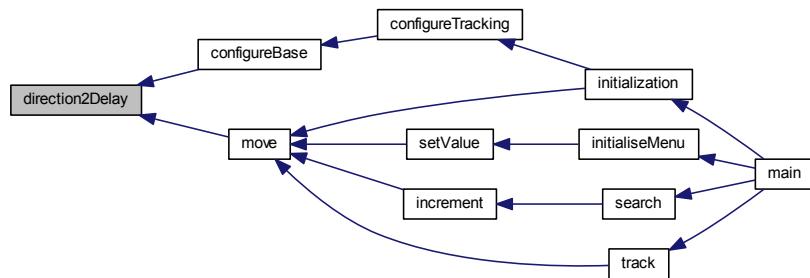
Returns: [Delay](#) - The required PDM delay to move the servos to the given direction

Remark: This function relies on the ARC_RANGE macro being set correctly. This should hold the value of the maximum angle that the servos can be commanded

Here is the call graph for this function:



Here is the caller graph for this function:



5.20.2.6 Direction getDir(void)

Function: [getDir\(void\)](#)

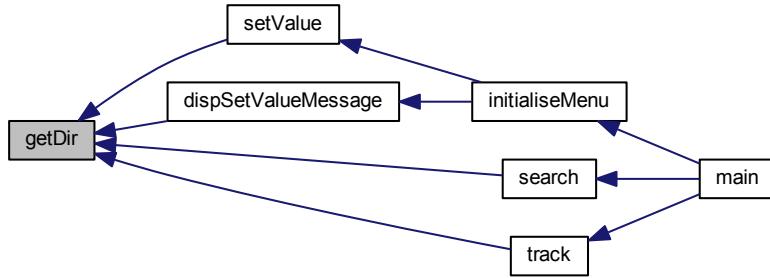
Include: [PanTilt.h](#)

Description: returns the current position of the pan tilt mechanism

Arguments: None

Returns: A struct containing the azimuth and inclination

Here is the caller graph for this function:



5.20.2.7 char getMaxAzimuthAngle (void)

Function: [getMaxAzimuthAngle\(void\)](#)

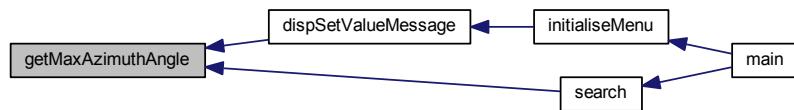
Include: [PanTilt.h](#)

Description: returns the maximum angle of the azimuth servo

Arguments: None

Returns: A char with the maximum azimuth angle.

Here is the caller graph for this function:



5.20.2.8 char getMaxElevationAngle (void)

Function: [getMaxElevationAngle\(void\)](#)

Include: [PanTilt.h](#)

Description: returns the maximum angle of the elevation servo

Arguments: None

Returns: A char with the maximum elevation angle.

Here is the caller graph for this function:



5.20.2.9 char getMinAzimuthAngle (void)

Function: [getMinAzimuthAngle\(void\)](#)

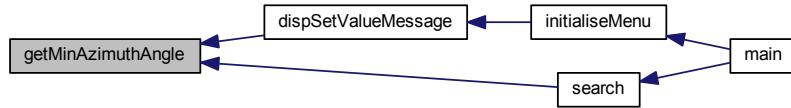
Include: [PanTilt.h](#)

Description: returns the minimum angle of the azimuth servo

Arguments: None

Returns: A char with the minimum azimuth angle.

Here is the caller graph for this function:



5.20.2.10 char getMinElevationAngle (void)

Function: [getMinElevationAngle\(void\)](#)

Include: [PanTilt.h](#)

Description: returns the minimum angle of the elevation servo

Arguments: None

Returns: A char with the minimum elevation angle.

Here is the caller graph for this function:



5.20.2.11 void increment (*Direction difference*)

Function: [increment\(Direction difference\)](#)

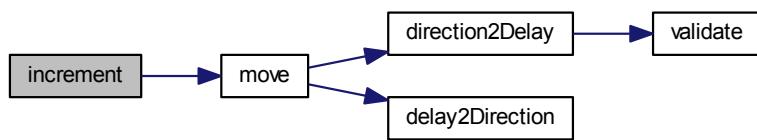
Include: [PanTilt.h](#)

Description: Moves the pan tilt actuator to the specified destination

Arguments: destination - A struct containing the desired azimuth and inclination

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.20.2.12 void incrementFine (*Direction difference*)

Function: [incrementFine\(Direction difference\)](#)

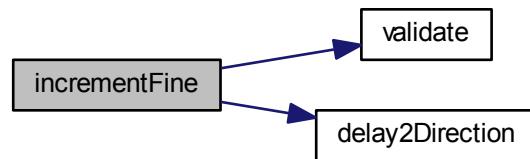
Include: [PanTilt.h](#)

Description: Moves the pan tilt actuator to the specified (Relative) destination

Arguments: destination - A struct containing the desired azimuth and inclination

Returns: None

Here is the call graph for this function:



5.20.2.13 void move (Direction *destination*)

Function: [move\(Direction destination\)](#)

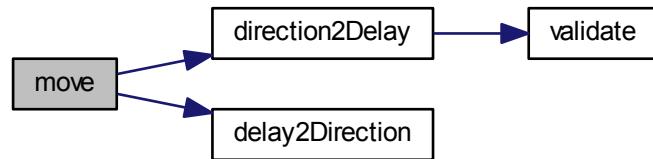
Include: [PanTilt.h](#)

Description: Moves the pan tilt actuator to the specified destination

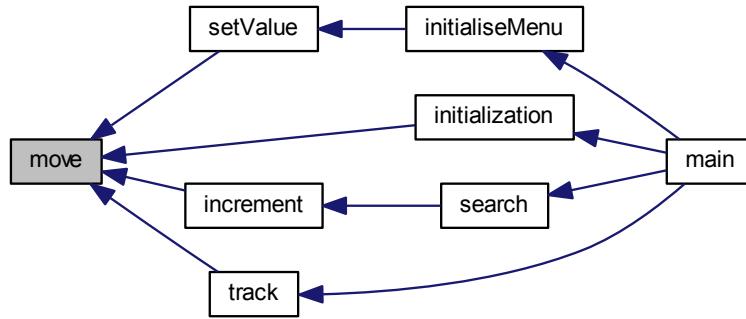
Arguments: destination - A struct containing the desired azimuth and inclination

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.20.2.14 void panTiltISR (void)

Function: [panTiltISR\(void\)](#)

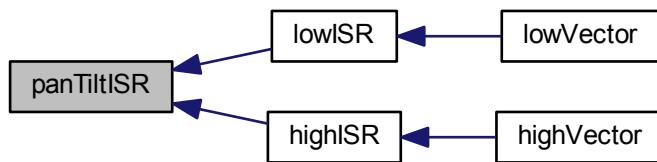
Include: [PanTilt.h](#)

Description: Acts as the ISR for the PanTilt module

Arguments: None

Returns: None

Here is the caller graph for this function:



5.20.2.15 Direction rawDir (void)

Function: [rawDir\(void\)](#)

Include: [PanTilt.h](#)

Description: returns the current PanTilt position without calibrating

Arguments: None

Returns: The position of the pan tilt without any calibration

5.20.2.16 void setMaxAzimuthAngle (char *p_angle*)

Function: setMaxAzimuthAngle(void)

Include: [PanTilt.h](#)

Description: sets the maximum angle of the azimuth servo

Arguments: The maximum angle (as char) to set for the azimuth servo

Returns: None.

Here is the caller graph for this function:

**5.20.2.17 void setMaxElevationAngle (char *p_angle*)**

Function: setMaxElevationAngle(void)

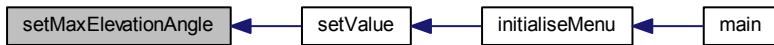
Include: [PanTilt.h](#)

Description: sets the maximum angle of the elevation servo

Arguments: The maximum angle (as char) to set for the elevation servo

Returns: None.

Here is the caller graph for this function:

**5.20.2.18 void setMinAzimuthAngle (char *p_angle*)**

Function: setMinAzimuthAngle(void)

Include: [PanTilt.h](#)

Description: sets the minimum angle of the azimuth servo

Arguments: The minimum angle (as char) to set for the azimuth servo

Returns: None.

Here is the caller graph for this function:



5.20.2.19 void setMinElevationAngle (char p_angle)

Function: `setMinElevationAngle(void)`

Include: [PanTilt.h](#)

Description: sets the minimum angle of the elevation servo

Arguments: The minimum angle (as char) to set for the elevation servo

Returns: None.

Here is the caller graph for this function:



5.20.2.20 char updated (void)

Function: [updated\(void\)](#)

Include: [PanTilt.h](#)

Description: returns true if the last move or increment or incrementFine function has taken effect. The new direction is only loaded in at the end of the PDM, so it could take up to 0.02 seconds for the change to take effect.

Arguments: delay - a pointer to the delay variable

Returns: None

5.20.2.21 static void validate (unsigned int * delay) [static]

Function: [validate\(unsigned int *delay\)](#)

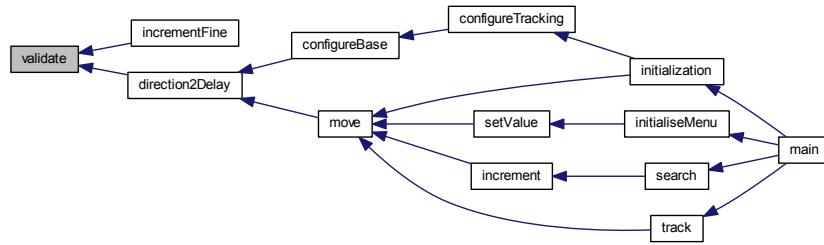
Include: Local to [PanTilt.c](#)

Description: Limits the duration of the PDM to between 1000us and 2000us

Arguments: delay - a pointer to the delay variable

Returns: None

Here is the caller graph for this function:



5.20.3 Variable Documentation

5.20.3.1 **Direction arcRange = { 94, 103 }** [static]

5.20.3.2 **signed char azimuth_angle_max** [static]

5.20.3.3 **signed char azimuth_angle_min** [static]

5.20.3.4 **Direction calibration_offset = { 0, 3 }** [static]

5.20.3.5 **volatile char changed = 0** [static]

5.20.3.6 **Direction current_direction** [static]

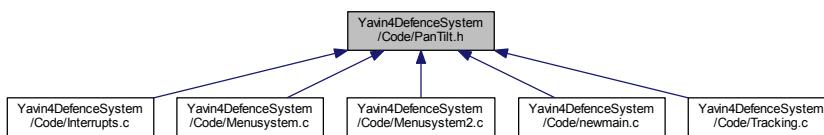
5.20.3.7 **signed char elevation_angle_max** [static]

5.20.3.8 **signed char elevation_angle_min** [static]

5.20.3.9 **Delay global_delay** [static]

5.21 Yavin4DefenceSystem/Code/PanTilt.h File Reference

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



Macros

- #define PAN_TILT_ISR CCP2_INT
- #define PANTILT_H

Functions

- void `configureBase` (void)
- void `move` (`Direction` destination)
- void `increment` (`Direction` difference)
- void `incrementFine` (`Direction` difference)
- `Direction getDir` (void)
- void `calibratePanTilt` (`Direction` reference)
- void `calibratePanTiltRange` (`Direction` reference)
- `Direction rawDir` (void)
- char `updated` (void)
- void `panTiltISR` (void)
- char `getMaxAzimuthAngle` (void)
- char `getMinAzimuthAngle` (void)
- char `getMaxElevationAngle` (void)
- char `getMinElevationAngle` (void)
- void `setMaxAzimuthAngle` (char p_angle)
- void `setMinAzimuthAngle` (char p_angle)
- void `setMaxElevationAngle` (char p_angle)
- void `setMinElevationAngle` (char p_angle)

5.21.1 Macro Definition Documentation

5.21.1.1 #define PAN_TILT_ISR CCP2_INT

File: `PanTilt.h` Author: Grant

Description: This file contains the public interface for the PanTile module

Created on 16 September 2014, 6:33 PM

5.21.1.2 #define PANTILT_H

5.21.2 Function Documentation

5.21.2.1 void calibratePanTilt (`Direction` reference)

Function: `calibratePanTilt(Direction reference)`

Include: `PanTilt.h`

Description: Calibrates the pan tile mechanism offset so that any future reference to move to the reference value specified in the function call will move the pan tilt back to the current position.

Arguments: reference - A struct containing the azimuth and inclinaion you wish to define as this position.

Returns: None

5.21.2.2 void calibratePanTiltRange (`Direction` reference)

Function: `calibratePanTiltRange(Direction reference)`

Include: `PanTilt.h`

Description: Calibrates the pan tile mechanism's range so that any future reference to move to the reference value specified in the function call will move the pan tilt back to the current position.

Arguments: reference - A struct containing the azimuth and inclinaion you wish to define as this position.

Returns: None

5.21.2.3 void configureBase (void)

Function: [configureBase\(void\)](#)

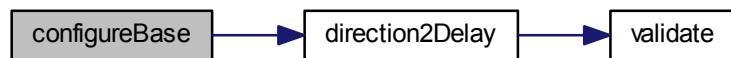
Include: [PanTilt.h](#)

Description: Configures the Pan Tilt mechanism for operation

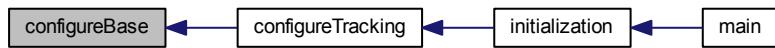
Arguments: None

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.21.2.4 Direction getDir (void)

Function: [getDir\(void\)](#)

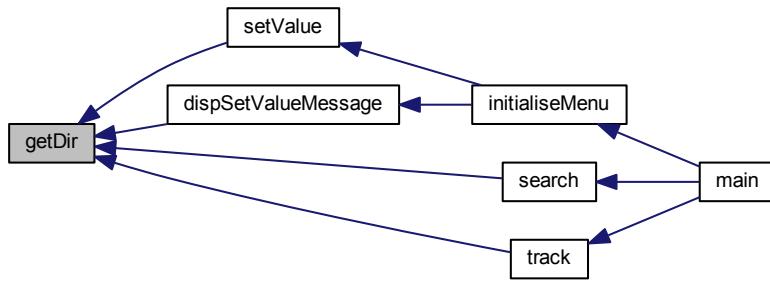
Include: [PanTilt.h](#)

Description: returns the current position of the pan tilt mechanism

Arguments: None

Returns: A struct containing the azimuth and inclination

Here is the caller graph for this function:



5.21.2.5 char getMaxAzimuthAngle (void)

Function: [getMaxAzimuthAngle\(void\)](#)

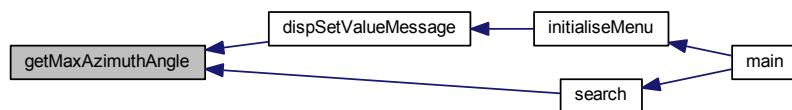
Include: [PanTilt.h](#)

Description: returns the maximum angle of the azimuth servo

Arguments: None

Returns: A char with the maximum azimuth angle.

Here is the caller graph for this function:



5.21.2.6 char getMaxElevationAngle (void)

Function: [getMaxElevationAngle\(void\)](#)

Include: [PanTilt.h](#)

Description: returns the maximum angle of the elevation servo

Arguments: None

Returns: A char with the maximum elevation angle.

Here is the caller graph for this function:



5.21.2.7 char getMinAzimuthAngle (void)

Function: [getMinAzimuthAngle\(void\)](#)

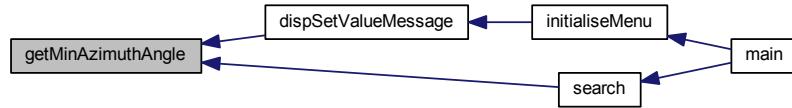
Include: [PanTilt.h](#)

Description: returns the minimum angle of the azimuth servo

Arguments: None

Returns: A char with the minimum azimuth angle.

Here is the caller graph for this function:



5.21.2.8 char getMinElevationAngle (void)

Function: [getMinElevationAngle\(void\)](#)

Include: [PanTilt.h](#)

Description: returns the minimum angle of the elevation servo

Arguments: None

Returns: A char with the minimum elevation angle.

Here is the caller graph for this function:



5.21.2.9 void increment (Direction difference)

Function: [increment\(Direction difference\)](#)

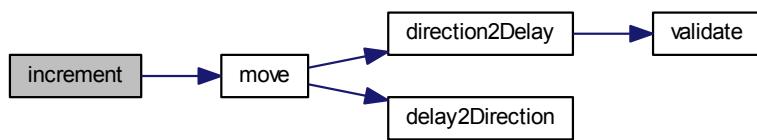
Include: [PanTilt.h](#)

Description: Moves the pan tilt actuator to the specified destination

Arguments: destination - A struct containing the desired azimuth and inclination

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.21.2.10 void incrementFine (Direction difference)

Function: [incrementFine\(Direction difference\)](#)

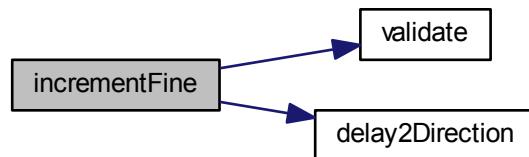
Include: [PanTilt.h](#)

Description: Moves the pan tilt actuator to the specified (Relative) destination

Arguments: destination - A struct containing the desired azimuth and inclination

Returns: None

Here is the call graph for this function:



5.21.2.11 void move (Direction *destination*)

Function: [move\(Direction destination\)](#)

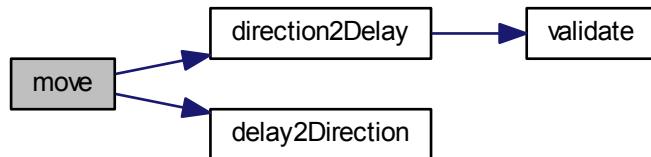
Include: [PanTilt.h](#)

Description: Moves the pan tilt actuator to the specified destination

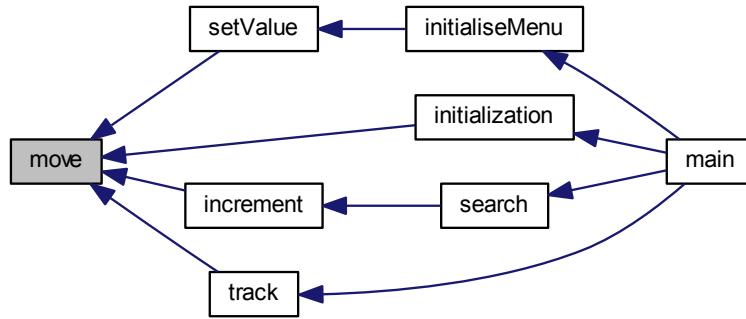
Arguments: destination - A struct containing the desired azimuth and inclination

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.21.2.12 void panTiltISR (void)

Function: [panTiltISR\(void\)](#)

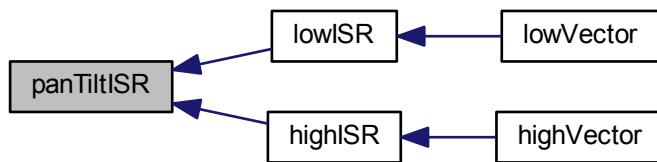
Include: [PanTilt.h](#)

Description: Acts as the ISR for the PanTilt module

Arguments: None

Returns: None

Here is the caller graph for this function:



5.21.2.13 Direction rawDir (void)

Function: [rawDir\(void\)](#)

Include: [PanTilt.h](#)

Description: returns the current PanTilt position without calibrating

Arguments: None

Returns: The position of the pan tilt without any calibration

5.21.2.14 void setMaxAzimuthAngle (char *p_angle*)

Function: setMaxAzimuthAngle(void)

Include: [PanTilt.h](#)

Description: sets the maximum angle of the azimuth servo

Arguments: The maximum angle (as char) to set for the azimuth servo

Returns: None.

Here is the caller graph for this function:



5.21.2.15 void setMaxElevationAngle (char *p_angle*)

Function: setMaxElevationAngle(void)

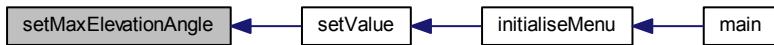
Include: [PanTilt.h](#)

Description: sets the maximum angle of the elevation servo

Arguments: The maximum angle (as char) to set for the elevation servo

Returns: None.

Here is the caller graph for this function:



5.21.2.16 void setMinAzimuthAngle (char *p_angle*)

Function: setMinAzimuthAngle(void)

Include: [PanTilt.h](#)

Description: sets the minimum angle of the azimuth servo

Arguments: The minimum angle (as char) to set for the azimuth servo

Returns: None.

Here is the caller graph for this function:



5.21.2.17 void setMinElevationAngle (char p_angle)

Function: setMinElevationAngle(void)

Include: [PanTilt.h](#)

Description: sets the minimum angle of the elevation servo

Arguments: The minimum angle (as char) to set for the elevation servo

Returns: None.

Here is the caller graph for this function:



5.21.2.18 char updated (void)

Function: [updated\(void\)](#)

Include: [PanTilt.h](#)

Description: returns true if the last move or increment or incrementFine function has taken effect. The new direction is only loaded in at the end of the PDM, so it could take up to 0.02 seconds for the change to take effect.

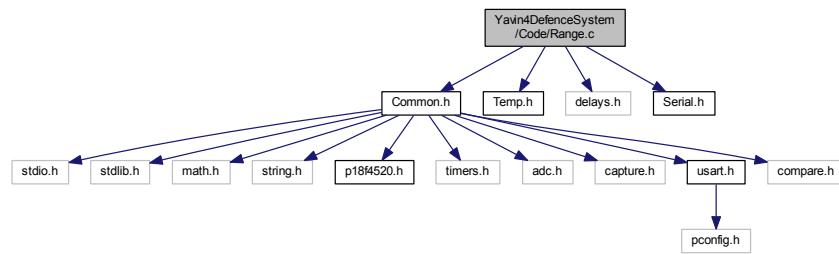
Arguments: delay - a pointer to the delay variable

Returns: None

5.22 Yavin4DefenceSystem/Code/Range.c File Reference

```
#include "Common.h"
#include "Temp.h"
#include <delays.h>
#include "Serial.h"
```

Include dependency graph for Range.c:



Macros

- `#define IR_CONV(ad) ((unsigned long)135174 / (ad) - 28)`
Converts AD reading into IR range.
- `#define ULTRA_CONV(tme, T) DIV_65536(tme * (unsigned long)(DIV_65536((unsigned long)519078 * T) + (unsigned long)4362)) - 18`
- `#define INIT_PIN PORTCbits.RC3`
- `#define INIT_TRIS TRISCbits.RC3`
- `#define CCP1_INPT TRISCbits.RC2`
- `#define range_IR sumIR`
- `#define range_US sumUS`

Functions

- `static void transRange (unsigned int us, unsigned int ir)`
- `static void beginUS (void)`
- `static unsigned int rangeUS (unsigned char temp)`
- `static unsigned int fuseRange (unsigned int us, unsigned int ir)`
- `void configureRange (void)`
- `unsigned int range (void)`
Samples the range.
- `void configureAD (void)`
- `void rangeISR (void)`
- `unsigned int rangeUltrasonic (void)`
- `void calibrateRange (unsigned int reference)`
- `unsigned int rawRangeIR (void)`
- `unsigned int rawRangeUS (void)`
- `unsigned int getLastRange (void)`
Samples the range.
- `TargetState getTargetState (void)`
- `TargetState readTargetState (void)`
- `int getMaxRange (void)`
- `int getMinRange (void)`
- `void setMaxRange (int range)`
- `void setMinRange (int range)`
- `void setNumSamples (char samples)`
- `char getNumSamples ()`
- `void setUsSampleRate (char sampleRate)`
- `char getUsSampleRate ()`

Variables

- static volatile char `measuringUS` = 0
- static int `m_minRange` = 300
- static int `m_maxRange` = 1000
- static unsigned int `lastRange` = 0
- static unsigned int `lastUSRange` = 0
- static unsigned int `lastIRRange` = 0
- static signed int `calibration_offset_IR` = 0
- static signed int `calibration_offset_US` = 0
- static char `numSamples` = 1
- static unsigned char `rateUS` = 15
- static unsigned int `rateIR` = 200
- static `TargetState current_target_state`
- static volatile unsigned int `ccp_value`

5.22.1 Macro Definition Documentation

5.22.1.1 `#define CCP1_INPT TRISCbits.RC2`

5.22.1.2 `#define INIT_PIN PORTCbits.RC3`

5.22.1.3 `#define INIT_TRIS TRISCbits.RC3`

5.22.1.4 `#define IR_CONV(ad) ((unsigned long)135174 / (ad) - 28)`

Converts AD reading into IR range.

5.22.1.5 `#define range_IR sumIR`

5.22.1.6 `#define range_US sumUS`

5.22.1.7 `#define ULTRA_CONV(tme, T) DIV_65536(tme * (unsigned long)(DIV_65536((unsigned long)519078 * T) + (unsigned long)4362)) - 18`

Converts a time delay (in clock cycles) and a temperature into an ultrasonic distance. Uses a linear approximation of the speed of sound to temperature relation, which changes the gradient of the Ultrasonic calibration

5.22.2 Function Documentation

5.22.2.1 `static void beginUS(void) [static]`

Function: `beginUS(void)`

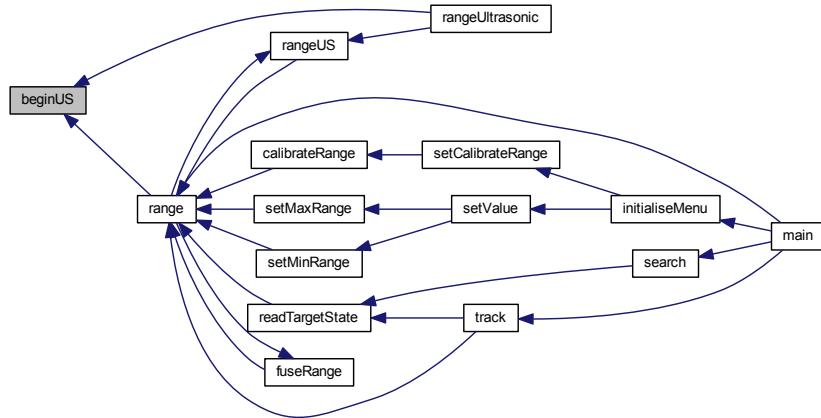
Include: `ultrasonic.h`

Description: starts a scan on the ultrasonic sensor

Arguments: None

Returns: None

Here is the caller graph for this function:



5.22.2.2 void calibrateRange (unsigned int reference)

Function: [calibrateRange\(unsigned int reference\)](#)

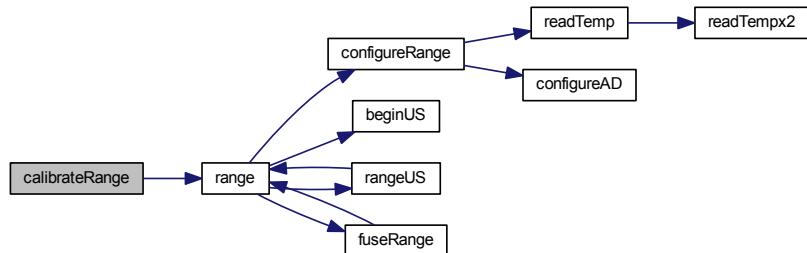
Include:

Description: Calibrates the range of both the IR and ultrasonic sensors based on a given range.

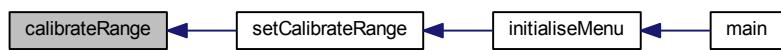
Arguments: reference - The distance in mm to calibrate the current measurements from

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.22.2.3 void configureAD (void)

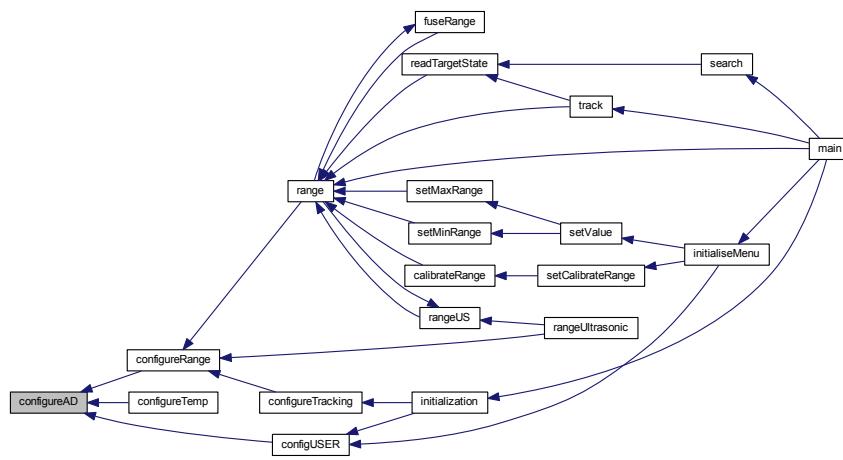
Function: [configureAD\(void\)](#)

Include: [Range.h](#)

Description: Configures the ADC, In ADCON1, we set right-justified mode, and select AN0 as the input channel. In ADCON0, we set a sample rate of Fosc/8, select AN0, and enable the ADC. Arguments: None

Returns: None

Here is the caller graph for this function:



5.22.2.4 void configureRange (void)

Function: [configureRange\(void\)](#)

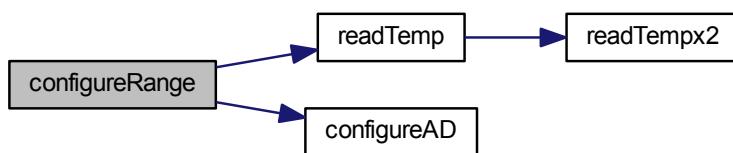
Include: [Range.h](#)

Description: Configures the Range module

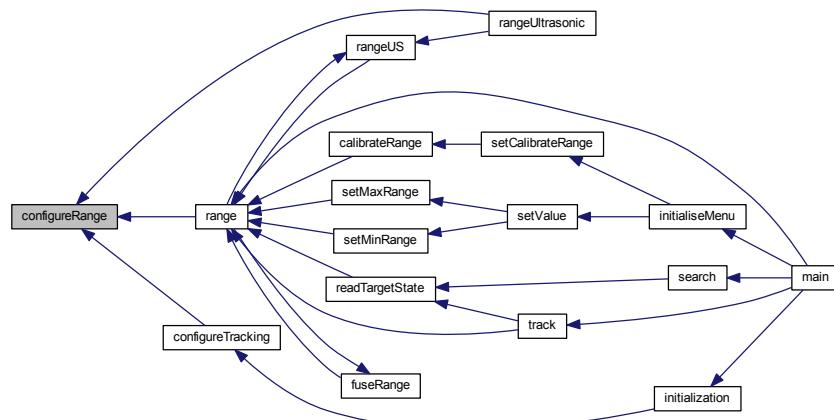
Arguments: None

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.22.2.5 static unsigned int fuseRange (unsigned int us, unsigned int ir) [static]

Function: [fuseRange\(unsigned int us, unsigned int ir\)](#)

Include: [Range.h](#)

Description: Fuses the IR and Ultrasonic ranges, and sets the target state

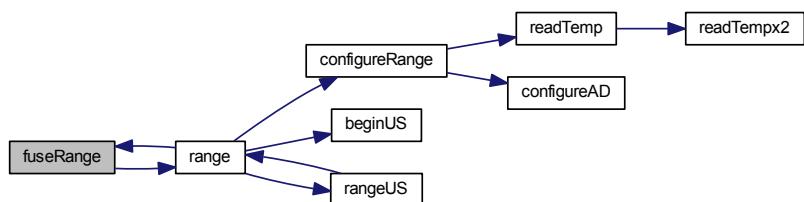
Arguments: us - the Ultrasonic range (mm) ir - the IR range (mm)

Returns: the fused range

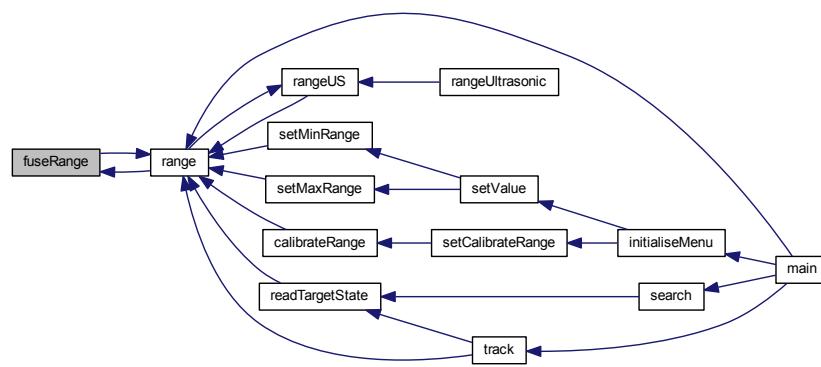
Note: Also sets the current target state based on the reading from both the IR and ultrasonic sensors Implement IR in here a little?

: Report Error?

Here is the call graph for this function:



Here is the caller graph for this function:



5.22.2.6 unsigned int getLastRange (void)

Samples the range.

Function: [getLastRange\(void\)](#)

Include: [Range.h](#)

Description: Returns the last range sample, so the data can be used without re-sampling the range finding sensors

Arguments: None

Returns: the last Range sample data

5.22.2.7 int getMaxRange (void)

Function: [getMaxRange\(\)](#)

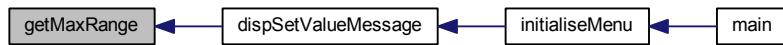
Include: [Range.h](#)

Description: Gets the maximum range of the device

Arguments:

Returns: The max range in mm

Here is the caller graph for this function:



5.22.2.8 int getMinRange (void)

Function: [getMinRange\(\)](#)

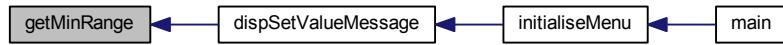
Include: [Range.h](#)

Description: Gets the minimum range of the device

Arguments:

Returns: The min range in mm

Here is the caller graph for this function:



5.22.2.9 char getNumSamples ()

Function: [getNumSamples\(\)](#)

Include: [Range.h](#)

Description: Gets the number of samples per range estimate

Arguments: None

Returns: The number of samples

Here is the caller graph for this function:



5.22.2.10 TargetState getTargetState (void)

Function: [getTargetState\(void\)](#)

Include: [Range.h](#)

Description: Returns the target state from the last range reading. E.g. Good track, or direction not quite correct as US returned, but IR didn't and was within IR range etc.

Arguments: None

Returns: the target state

5.22.2.11 char getUsSampleRate ()

Function: [getUsSampleRate\(\)](#)

Include: [Range.h](#)

Description: Gets the sampling rate of the ultrasonic sensor

Arguments:

Returns: The sampling rate (In Hz)

Here is the caller graph for this function:



5.22.2.12 unsigned int range(void)

Samples the range.

Function: [range\(void\)](#)

Include: [Range.h](#)

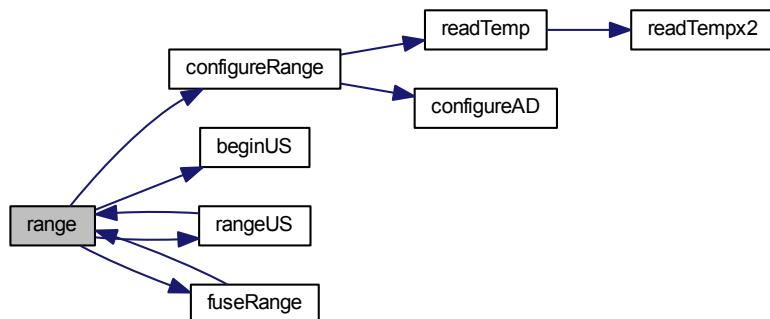
Description: Takes a number of samples of the ultrasonic sensor at a specified rate. Continues to sample the IR sensor at a different rate while sampling the ultrasonic. Then combines the ranges and sets the target state

Arguments: None

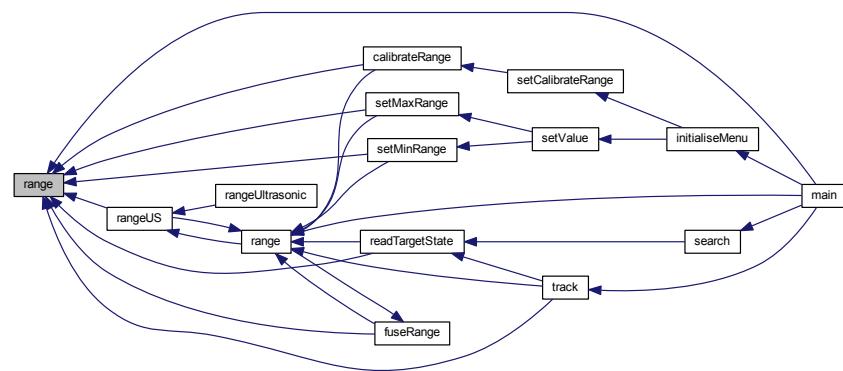
Returns: the range Standard room temperature for now

Todo Read in temperature for US calculation

Here is the call graph for this function:



Here is the caller graph for this function:



5.22.2.13 void rangelSR (void)

Function: [rangelSR\(void\)](#)

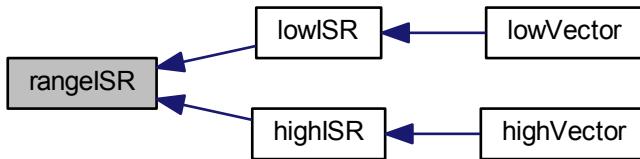
Include: [range.h](#)

Description: Called when an range related interrupt is fired, acts as the service routine for the rangefinding module.

Arguments: None

Returns: None

Here is the caller graph for this function:



5.22.2.14 unsigned int rangeUltrasonic (void)

Function: [rangeUltrasonic\(void\)](#)

Include:

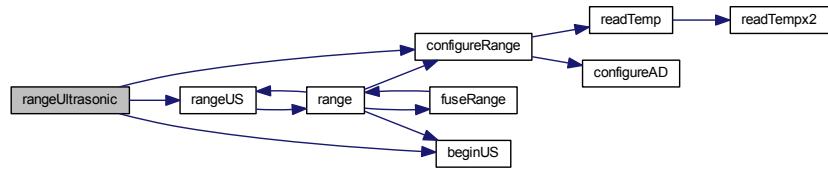
Description: performs an ultrasonic range reading. Pins:

Arguments: None

Returns: the average of the samples

todo remove this function?

Here is the call graph for this function:



5.22.2.15 static unsigned int rangeUS (unsigned char temp) [static]

Function: [rangeUS\(unsigned char temp\)](#)

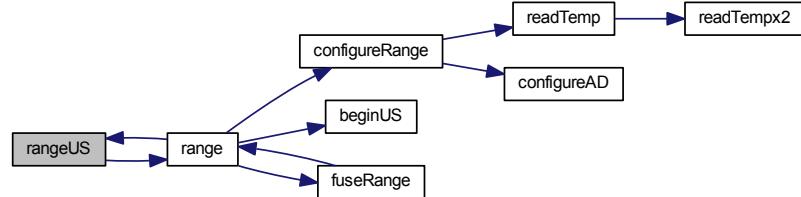
Include: ultrasonic.h

Description: Returns the result of the ultrasonic read (zero if no target found). Will poll until measurement is complete.

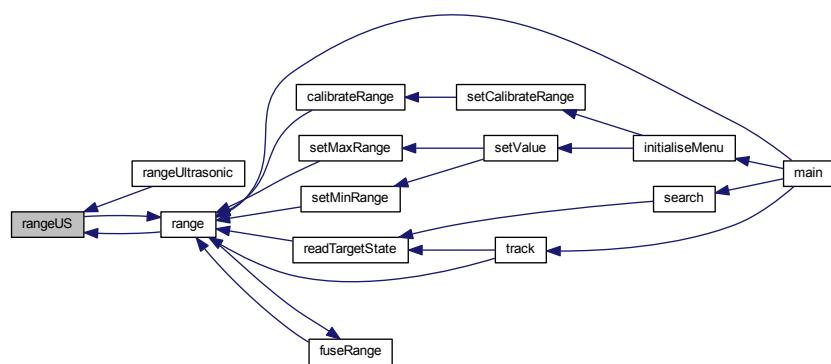
Arguments: tempx2 - 2x the temperature in deg Celsius

Returns: Distance in mm (unsigned int)

Here is the call graph for this function:



Here is the caller graph for this function:



5.22.2.16 unsigned int rawRangeIR (void)

Function: [rawRangeIR\(void\)](#)

Include: [Range.h](#)

Description: Returns the last IR range reading

Arguments: None

Returns: the range in mm

Here is the caller graph for this function:

**5.22.2.17 unsigned int rawRangeUS (void)**

Function: [rawRangeUS\(void\)](#)

Include: [Range.h](#)

Description: Returns the last US range reading

Arguments: None

Returns: the range in mm

Here is the caller graph for this function:

**5.22.2.18 TargetState readTargetState (void)**

Function: [readTargetState\(void\)](#)

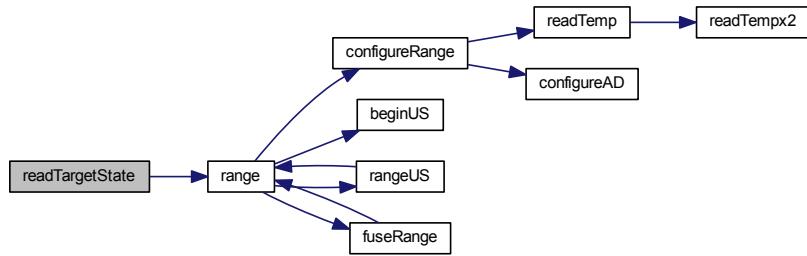
Include: [Range.h](#)

Description: Does the same thing as getTargetState, but actually performs a [range\(\)](#) read

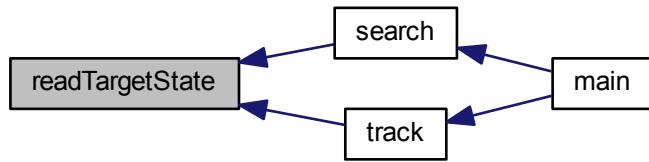
Arguments: None

Returns: the target state

Here is the call graph for this function:



Here is the caller graph for this function:



5.22.2.19 void setMaxRange(int range)

Function: [setMaxRange\(\)](#)

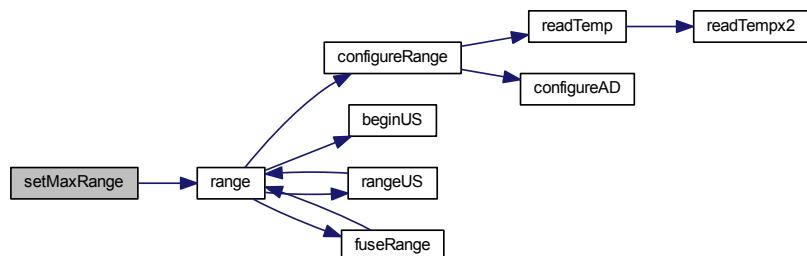
Include: [Range.h](#)

Description: Sets the maximum range of the device

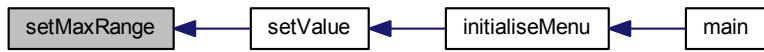
Arguments: range - The max range to set in mm

Returns: N/A

Here is the call graph for this function:



Here is the caller graph for this function:



5.22.2.20 void setMinRange (int range)

Function: [setMinRange\(\)](#)

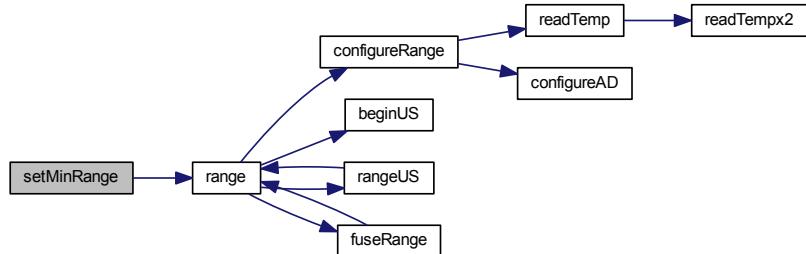
Include: [Range.h](#)

Description: Sets the minimum range of the device

Arguments: range - The max range to set in mm

Returns: N/A

Here is the call graph for this function:



Here is the caller graph for this function:



5.22.2.21 void setNumSamples (char samples)

Function: [setNumSamples\(\)](#)

Include: [Range.h](#)

Description: sets the number of samples per range estimage

Arguments: the number of samples

Returns: None

Here is the caller graph for this function:



5.22.2.22 void setUsSampleRate (char sampleRate)

Function: [setUsSampleRate\(char sampleRate\)](#)

Include: [Range.h](#)

Description: Sets the ultrasonic sampling rate

Arguments: the sampling rate in Hz

Returns: None

Here is the caller graph for this function:



5.22.2.23 static void transRange (unsigned int us, unsigned int ir) [static]

File: [Range.c](#) Author: Grant

Description: Contains all the functionality for the range module. All variables and settings concerning the range module including the the max distance, timeouts etc are private to this module. The interface functions allow all valid access to the module.

Duties: -Interface to the IR and Ultrasonic sensors -Read the range from the IR and ultrasonic sensors -Fuse distances from the IR and ultrasonic sensors -Calibrate the IR and ultrasonic sensors

Functions:

Created on 15 September 2014, 11:27 AM

5.22.3 Variable Documentation

5.22.3.1 signed int calibration_offset_IR = 0 [static]

5.22.3.2 signed int calibration_offset_US = 0 [static]

5.22.3.3 volatile unsigned int ccp_value [static]

5.22.3.4 `TargetState current_target_state [static]`

5.22.3.5 `unsigned int lastIRRange = 0 [static]`

5.22.3.6 `unsigned int lastRange = 0 [static]`

5.22.3.7 `unsigned int lastUSRRange = 0 [static]`

5.22.3.8 `int m_maxRange = 1000 [static]`

5.22.3.9 `int m_minRange = 300 [static]`

5.22.3.10 `volatile char measuringUS = 0 [static]`

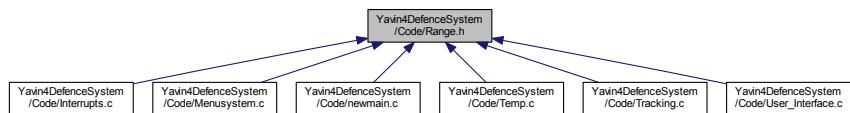
5.22.3.11 `char numSamples = 1 [static]`

5.22.3.12 `unsigned int rateIR = 200 [static]`

5.22.3.13 `unsigned char rateUS = 15 [static]`

5.23 Yavin4DefenceSystem/Code/Range.h File Reference

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



Macros

- `#define RANGE_INT CCP1_INT | TMR1_INT`
- `#define RANGE_H`

Functions

- `void configureRange (void)`
- `void configureAD (void)`
- `unsigned int range (void)`
 - Samples the range.*
- `unsigned int lastRange (void)`
 - Samples the range.*
- `void rangeISR (void)`
- `void calibrateRange (signed int distance)`
- `unsigned int rawRangeIR (void)`
- `unsigned int rawRangeUS (void)`
- `TargetState getTargetState (void)`
- `TargetState readTargetState (void)`
- `void setMaxRange (int range)`
- `void setMinRange (int range)`
- `int getMaxRange (void)`

- int [getMinRange](#) (void)
- char [getUsSampleRate](#) ()
- void [setUsSampleRate](#) (char samples)
- char [getNumSamples](#) ()
- void [setNumSamples](#) (char sampleRate)

5.23.1 Macro Definition Documentation

5.23.1.1 `#define RANGE_H`

5.23.1.2 `#define RANGE_INT CCP1_INT | TMR1_INT`

File: [Range.h](#) Author: Grant

Description: Contains the public interface for the ultrasonic module. This file contains all the external declarations, macros and global variables for using and Interfacing with the Range module.

Created on 15 September 2014, 11:24 AM

5.23.2 Function Documentation

5.23.2.1 `void calibrateRange (signed int distance)`

Function: [calibrateRange\(unsigned int reference\)](#)

Include:

Description: Calibrates the range of both the IR and ultrasonic sensors based on a given range.

Arguments: reference - The distance in mm to calibrate the current measurements from

Returns: None

5.23.2.2 `void configureAD (void)`

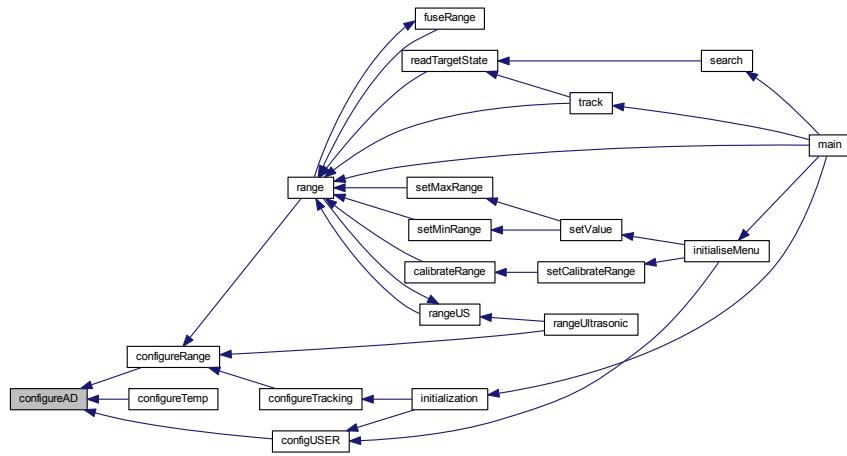
Function: [configureAD\(void\)](#)

Include: [Range.h](#)

Description: Configures the ADC, In ADCON1, we set right-justified mode, and select AN0 as the input channel. In ADCON0, we set a sample rate of Fosc/8, select AN0, and enable the ADC. Arguments: None

Returns: None

Here is the caller graph for this function:



5.23.2.3 void configureRange (void)

Function: `configureRange(void)`

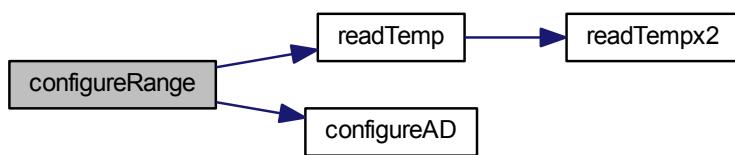
Include: Range.h

Description: Configures the Range module

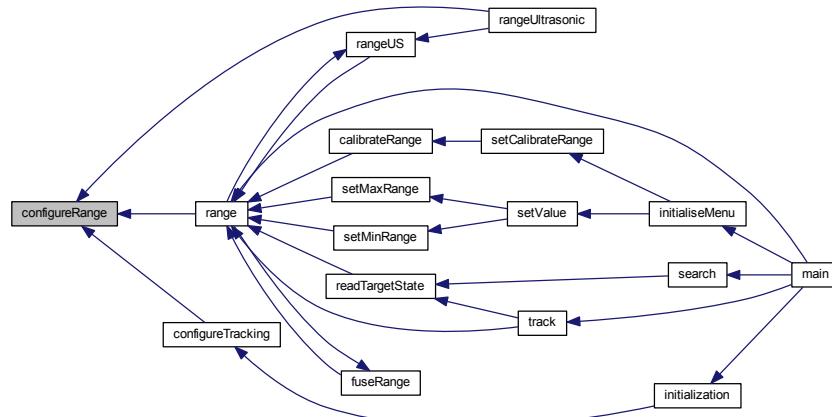
Arguments: None

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.23.2.4 int getMaxRange (void)

Function: [getMaxRange\(\)](#)

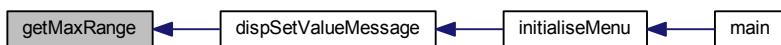
Include: [Range.h](#)

Description: Gets the maximum range of the device

Arguments:

Returns: The max range in mm

Here is the caller graph for this function:



5.23.2.5 int getMinRange (void)

Function: [getMinRange\(\)](#)

Include: [Range.h](#)

Description: Gets the minimum range of the device

Arguments:

Returns: The min range in mm

Here is the caller graph for this function:



5.23.2.6 char getNumSamples()

Function: [getNumSamples\(\)](#)

Include: [Range.h](#)

Description: Gets the number of samples per range estimate

Arguments: None

Returns: The number of samples

Here is the caller graph for this function:



5.23.2.7 TargetState getTargetState(void)

Function: [getTargetState\(void\)](#)

Include: [Range.h](#)

Description: Returns the target state from the last range reading. E.g. Good track, or direction not quite correct as US returned, but IR didn't and was within IR range etc.

Arguments: None

Returns: the target state

5.23.2.8 char getUsSampleRate()

Function: [getUsSampleRate\(\)](#)

Include: [Range.h](#)

Description: Gets the sampling rate of the ultrasonic sensor

Arguments:

Returns: The sampling rate (In Hz)

Here is the caller graph for this function:



5.23.2.9 unsigned int lastRange(void)

Samples the range.

Function: [getLastRange\(void\)](#)

Include: [Range.h](#)

Description: Returns the last range sample, so the data can be used without re-sampling the range finding sensors

Arguments: None

Returns: the last Range sample data

5.23.2.10 unsigned int range(void)

Samples the range.

Function: [range\(void\)](#)

Include: [Range.h](#)

Description: Takes a number of samples of the ultrasonic sensor at a specified rate. Continues to sample the IR sensor at a different rate while sampling the ultrasonic. Then combines the ranges and sets the target state

Arguments: None

Returns: the range to the target in mm

Function: [range\(void\)](#)

Include: [Range.h](#)

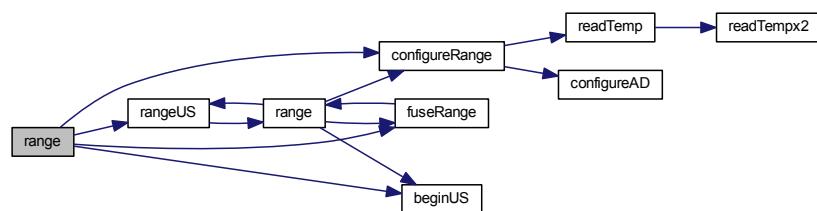
Description: Takes a number of samples of the ultrasonic sensor at a specified rate. Continues to sample the IR sensor at a different rate while sampling the ultrasonic. Then combines the ranges and sets the target state

Arguments: None

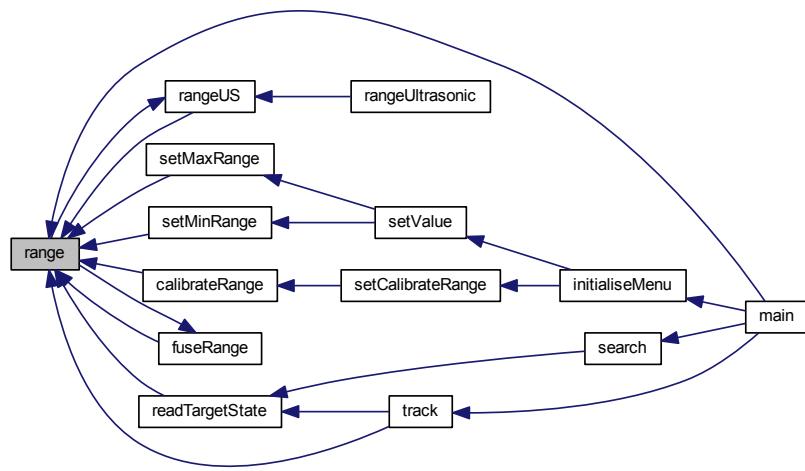
Returns: the range Standard room temperature for now

Todo Read in temperature for US calculation

Here is the call graph for this function:



Here is the caller graph for this function:



5.23.2.11 void rangeISR (void)

Function: [rangeISR\(void\)](#)

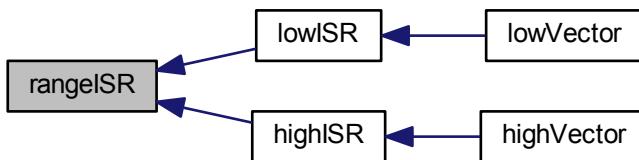
Include: [range.h](#)

Description: Called when an range related interrupt is fired, acts as the service routine for the rangefinding module.

Arguments: None

Returns: None

Here is the caller graph for this function:



5.23.2.12 unsigned int rawRangeIR (void)

Function: [rawRangeIR\(void\)](#)

Include: [Range.h](#)

Description: Returns the last IR range reading

Arguments: None

Returns: the range in mm

Here is the caller graph for this function:



5.23.2.13 unsigned int rawRangeUS (void)

Function: [rawRangeUS\(void\)](#)

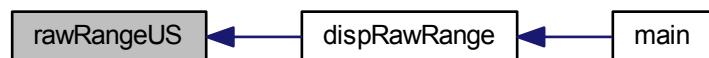
Include: [Range.h](#)

Description: Returns the last US range reading

Arguments: None

Returns: the range in mm

Here is the caller graph for this function:



5.23.2.14 TargetState readTargetState (void)

Function: [readTargetState\(void\)](#)

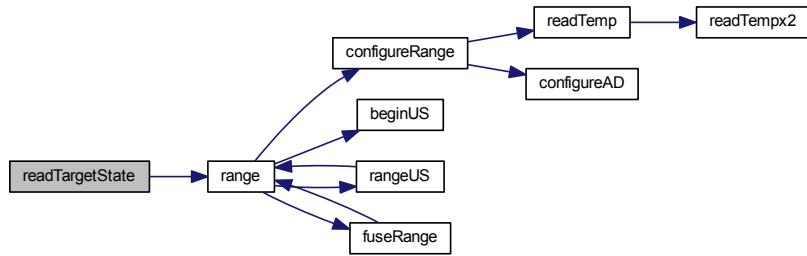
Include: [Range.h](#)

Description: Does the same thing as getTargetState, but actually performs a [range\(\)](#) read

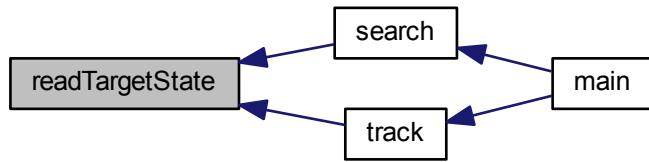
Arguments: None

Returns: the target state

Here is the call graph for this function:



Here is the caller graph for this function:



5.23.2.15 void setMaxRange(int range)

Function: [setMaxRange\(\)](#)

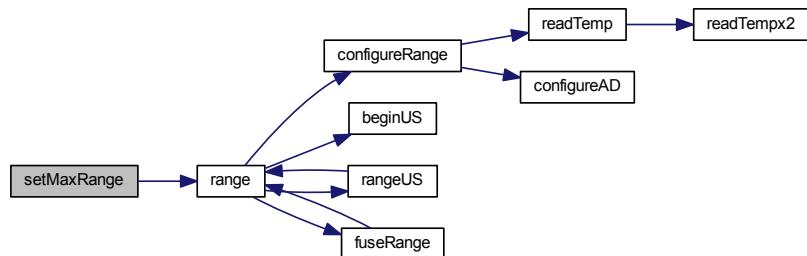
Include: [Range.h](#)

Description: Sets the maximum range of the device

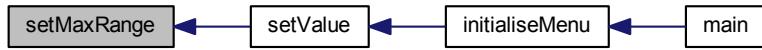
Arguments: range - The max range to set in mm

Returns: N/A

Here is the call graph for this function:



Here is the caller graph for this function:



5.23.2.16 void setMinRange (int range)

Function: [setMinRange\(\)](#)

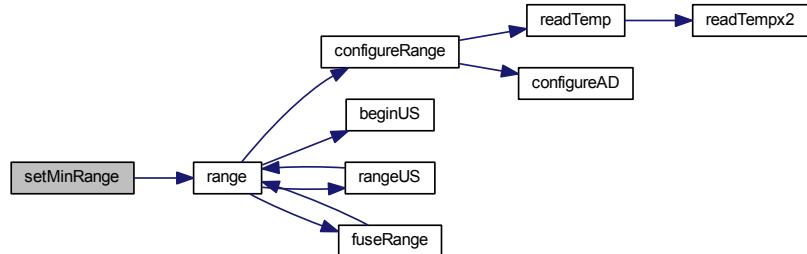
Include: [Range.h](#)

Description: Sets the minimum range of the device

Arguments: range - The max range to set in mm

Returns: N/A

Here is the call graph for this function:



Here is the caller graph for this function:



5.23.2.17 void setNumSamples (char samples)

Function: [setNumSamples\(\)](#)

Include: [Range.h](#)

Description: sets the number of samples per range estimage

Arguments: the number of samples

Returns: None

Here is the caller graph for this function:



5.23.2.18 void setUsSampleRate (char sampleRate)

Function: [setUsSampleRate\(char sampleRate\)](#)

Include: [Range.h](#)

Description: Sets the ultrasonic sampling rate

Arguments: the sampling rate in Hz

Returns: None

Here is the caller graph for this function:



5.24 Yavin4DefenceSystem/Code/ROM2RAM.c File Reference

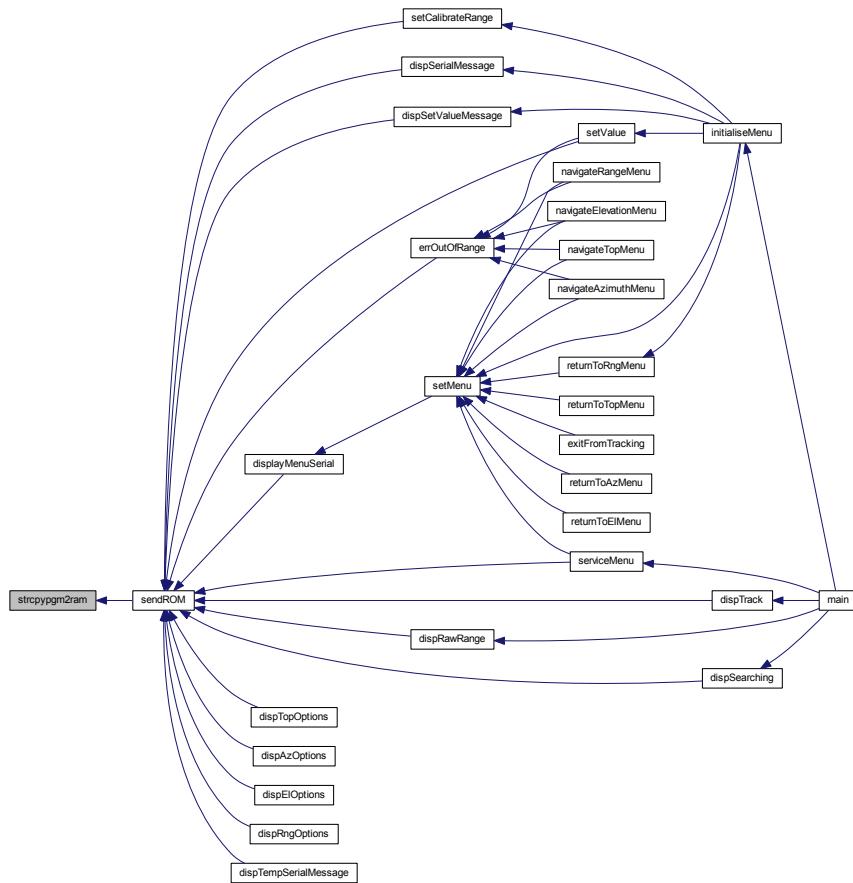
Functions

- `char * strcpypgm2ram (auto char *s1, auto const rom char *s2)`

5.24.1 Function Documentation

5.24.1.1 `char * strcpypgm2ram (auto char * s1, auto const rom char * s2)`

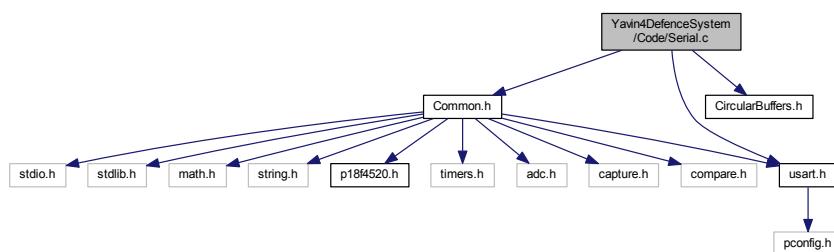
Here is the caller graph for this function:



5.25 Yavin4DefenceSystem/Code/Serial.c File Reference

```
#include "Common.h"
#include <usart.h>
#include "CircularBuffers.h"
```

Include dependency graph for Serial.c:



Macros

- #define TX_INT_CLEAR() PIR1bits.TXIF = 0
- #define RC_INT_CLEAR() PIR1bits.RCIF = 0
- #define TX_INT_ENABLE() TX_INT_CLEAR(); PIE1bits.TXIE = 1
- #define RC_INT_ENABLE() RC_INT_CLEAR(); PIE1bits.RCIE = 1
- #define TX_INT_DISABLE() PIE1bits.TXIE = 0
- #define RC_INT_DISABLE() PIE1bits.RCIE = 0
- #define CR 0x0D
- #define NL 0x0A
- #define ESC 0x1B
- #define TAB 0x09
- #define BS 0x07

Functions

- void **configureSerial** (void)
- void **transmit** (char *string)
- void **transChar** (char c)
- char **receiveEmpty** (void)
- char **receivePeek** (void)
- char **receivePop** (void)
- char **receiveCR** (void)
- char **receiveEsc** (void)
- void **popEsc** (void)
- void **readString** (char *string)
- char **transmitComplete** (void)
- void **serialISR** (void)
- void **clearReceive** (void)

Variables

- static volatile **circularBuffer** receive_buffer
- static volatile **circularBuffer** transmit_buffer
- static volatile char carriageReturn = 0
- static volatile char escPressed = 0

5.25.1 Macro Definition Documentation

5.25.1.1 #define BS 0x07

5.25.1.2 #define CR 0x0D

5.25.1.3 #define ESC 0x1B

5.25.1.4 #define NL 0x0A

5.25.1.5 #define RC_INT_CLEAR() PIR1bits.RCIF = 0

5.25.1.6 #define RC_INT_DISABLE() PIE1bits.RCIE = 0

5.25.1.7 #define RC_INT_ENABLE() RC_INT_CLEAR(); PIE1bits.RCIE = 1

5.25.1.8 #define TAB 0x09

5.25.1.9 #define TX_INT_CLEAR() PIR1bits.TXIF = 0

File: [newmain.c](#) Author: Grant

Description: Contains the functionality for the Serial module. All variables and settings concerning the serial module, such as the receive and transmit circular buffers are private to this module. The interface functions allow all valid access to the module.

Duties: -Stores an received characters in the received buffer -Stores any characters to be transmitted -Transmits anything in transmit buffer via interrupts -Accessor functions for using and querying buffers

Functions:

Created on 7 September 2014, 4:12 PM

5.25.1.10 #define TX_INT_DISABLE() PIE1bits.TXIE = 0

5.25.1.11 #define TX_INT_ENABLE() TX_INT_CLEAR(); PIE1bits.TXIE = 1

5.25.2 Function Documentation

5.25.2.1 void clearReceive (void)

Function: [clearReceive\(void\)](#)

Include: [Serial.h](#)

Description: Clears the receive buffer

Arguments: None

Returns: None

Here is the caller graph for this function:



5.25.2.2 void configureSerial (void)

Function: [configureSerial\(void\)](#)

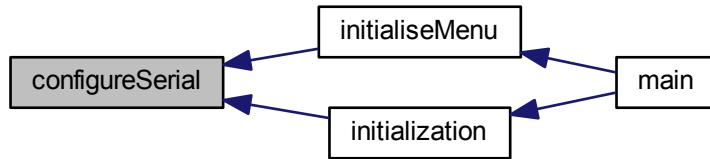
Include: [Serial.h](#)

Description: Configures the serial ready for communication

Arguments: None

Returns: None

Here is the caller graph for this function:



5.25.2.3 void popEsc(void)

Function: [popEsc\(void\)](#)

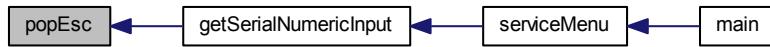
Include:

Description: Processes the Esc command and removes any input before the Esc command.

Arguments: None

Returns: Non-zero if received escape character

Here is the caller graph for this function:



5.25.2.4 void readString(char * string)

Function: [readString\(char *string\)](#)

Include: [Serial.h](#)

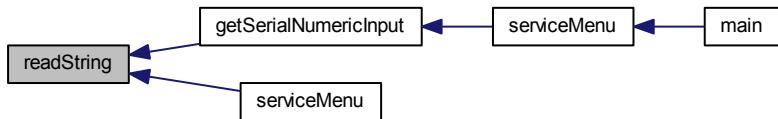
Description: Writes all received data up to a carriage return into given location.

Arguments: `string` - Pointer to location to store received data

Returns: Received data including the carriage return

Remarks: Make sure that you reserve at least `BUFFERLENGTH` elements at the location pointed to by `string` before calling this function.

Here is the caller graph for this function:



5.25.2.5 char receiveCR (void)

Function: [receiveCR\(void\)](#)

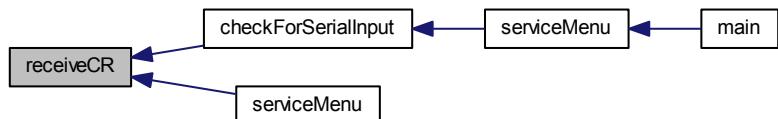
Include: [Serial.h](#)

Description: Indicates whether a Carriage Return has been received

Arguments: None

Returns: non-zero if CR has been received, zero otherwise

Here is the caller graph for this function:



5.25.2.6 char receiveEmpty (void)

Function: [receiveEmpty\(void\)](#)

Include: [Serial.h](#)

Description: Indicates if the receive buffer is empty

Arguments: None

Returns: returns true if the receive buffer is empty

Here is the caller graph for this function:



5.25.2.7 char receiveEsc (void)

Function: [receiveEsc\(void\)](#)

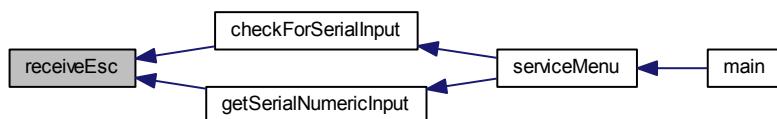
Include:

Description: Indicates whether an Escape character has been received

Arguments: None

Returns: non-zero if the Esc has been received, zero otherwise

Here is the caller graph for this function:



5.25.2.8 char receivePeek (void)

Function: [receivePeek\(void\)](#)

Include: [Serial.h](#)

Description: Returns the next character in the receive buffer without removing it from the buffer

Arguments: None

Returns: The next received character

5.25.2.9 char receivePop (void)

Function: [receivePop\(void\)](#)

Include: [Serial.h](#)

Description: Pops the next received character from the received buffer

Arguments: None

Returns: The next character from the receive buffer

Here is the caller graph for this function:



5.25.2.10 void serialISR (void)

Function: [serialISR\(void\)](#)

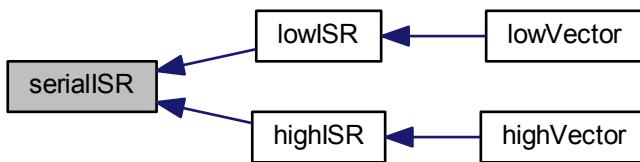
Include: [Serial.h](#)

Description: Acts as the interrupt service routine for the serial module

Arguments: None

Returns: None

Here is the caller graph for this function:



5.25.2.11 void transChar (char c)

Function: [transChar\(char c\)](#)

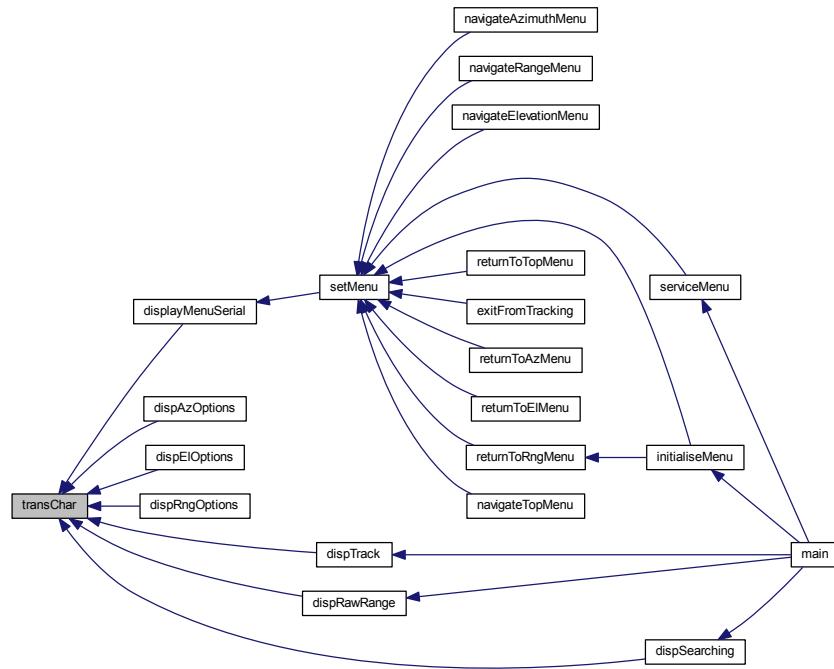
Include: [Serial.h](#)

Description: Transmits a single character

Arguments: `c` - character to transmit

Returns: None

Here is the caller graph for this function:



5.25.2.12 void transmit(*char * string*)

Function: [transmit\(char *string\)](#)

Include: [Serial.h](#)

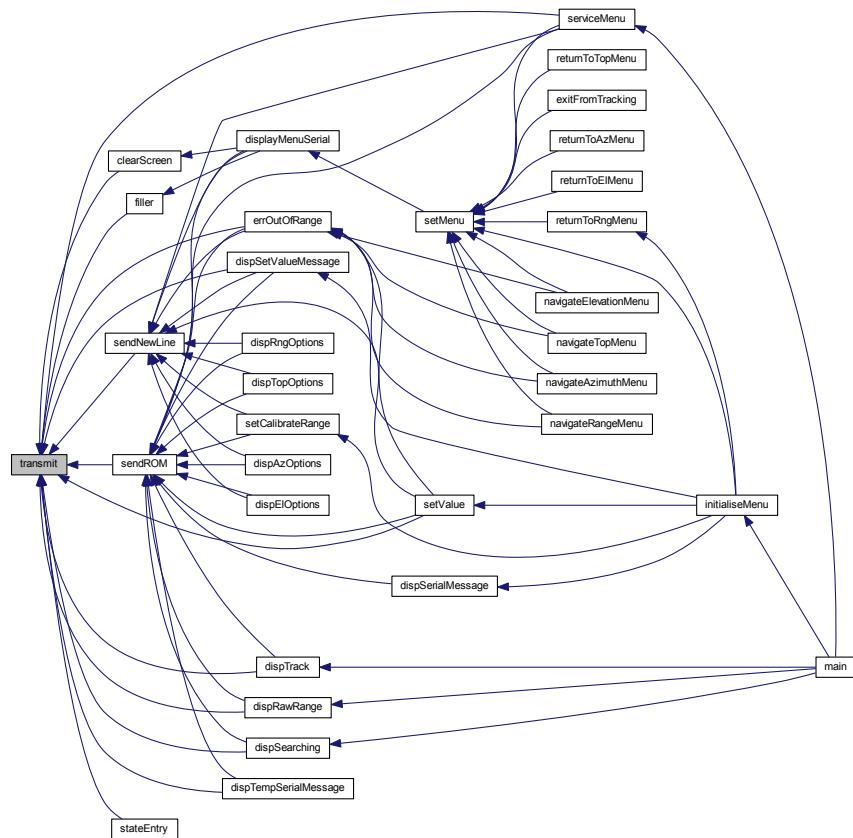
Description: Begins transmitting the string over serial (interrupt driven)

Arguments: string - pointer to the beginning of the string to transmit

Returns: None

NOTE: Must be Null Terminated! Cannot receive a literal.

Here is the caller graph for this function:



5.25.2.13 char transmitComplete(void)

Function: [transmitComplete\(void\)](#)

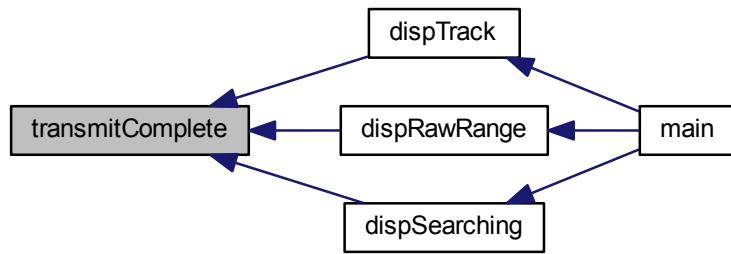
Include: [Serial.h](#)

Description: returns non-zero if the message has been completely transmitted e.g. if the transmit buffer is empty

Arguments: None

Returns: non-zero if all messages have been transmitted

Here is the caller graph for this function:



5.25.3 Variable Documentation

5.25.3.1 volatile char carriageReturn = 0 [static]

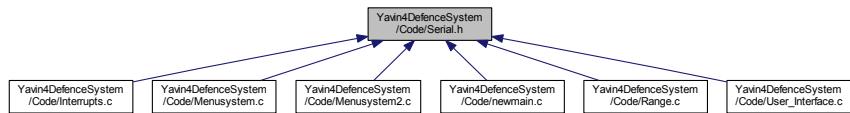
5.25.3.2 volatile char escPressed = 0 [static]

5.25.3.3 volatile circularBuffer receive_buffer [static]

5.25.3.4 volatile circularBuffer transmit_buffer [static]

5.26 Yavin4DefenceSystem/Code/Serial.h File Reference

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



Macros

- #define SERIAL_INT (TX_INT || RC_INT)
- #define SERIAL_H

Functions

- void **configureSerial** (void)
- void **serialISR** (void)
- void **transmit** (char *string)
- char **transmitComplete** (void)
- void **transChar** (char c)
- char **receiveEmpty** (void)
- char **receivePeek** (void)

- char `receivePop` (void)
- char `receiveCR` (void)
- char `receiveEsc` (void)
- void `readString` (char *string)
- void `popEsc` (void)
- void `clearReceive` (void)

5.26.1 Macro Definition Documentation

5.26.1.1 `#define SERIAL_H`

5.26.1.2 `#define SERIAL_INT (TX_INT || RC_INT)`

File: [Serial.h](#) Author: Grant

Description: Contains the public interface for the serial module. This file contains all the external declarations, macros and global variables for using and Interfacing with the serial module.

Created on 17 September 2014, 3:27 PM

5.26.2 Function Documentation

5.26.2.1 void `clearReceive` (void)

Function: [clearReceive\(void\)](#)

Include: [Serial.h](#)

Description: Clears the receive buffer

Arguments: None

Returns: None

Here is the caller graph for this function:



5.26.2.2 void `configureSerial` (void)

Function: [configureSerial\(void\)](#)

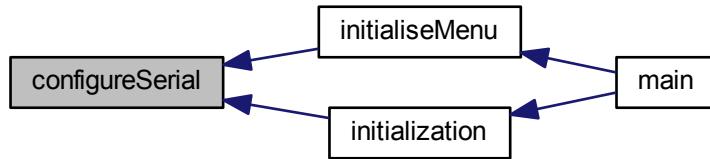
Include: [Serial.h](#)

Description: Configures the serial ready for communication

Arguments: None

Returns: None

Here is the caller graph for this function:



5.26.2.3 void popEsc(void)

Function: [popEsc\(void\)](#)

Include:

Description: Processes the Esc command and removes any input before the Esc command.

Arguments: None

Returns: Non-zero if received escape character

Here is the caller graph for this function:



5.26.2.4 void readString(char * string)

Function: [readString\(char *string\)](#)

Include: [Serial.h](#)

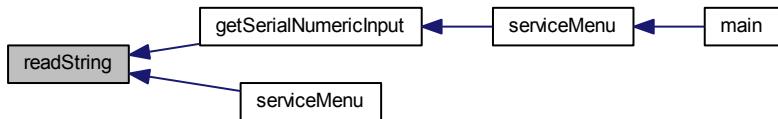
Description: Writes all received data up to a carriage return into given location.

Arguments: string - Pointer to location to store received data

Returns: Received data including the carriage return

Remarks: Make sure that you reserve at least BUFFERLENGTH elements at the location pointed to by string before calling this function.

Here is the caller graph for this function:



5.26.2.5 char receiveCR (void)

Function: [receiveCR\(void\)](#)

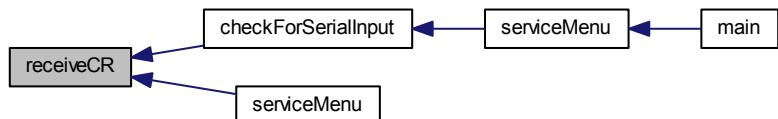
Include: [Serial.h](#)

Description: Indicates whether a Carriage Return has been received

Arguments: None

Returns: non-zero if CR has been received, zero otherwise

Here is the caller graph for this function:



5.26.2.6 char receiveEmpty (void)

Function: [receiveEmpty\(void\)](#)

Include: [Serial.h](#)

Description: Indicates if the receive buffer is empty

Arguments: None

Returns: returns true if the receive buffer is empty

Here is the caller graph for this function:



5.26.2.7 char receiveEsc (void)

Function: [receiveEsc\(void\)](#)

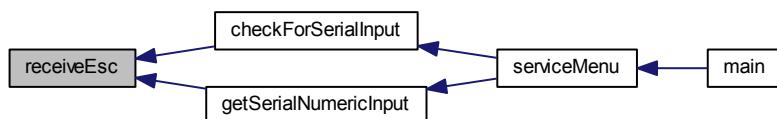
Include:

Description: Indicates whether an Escape character has been received

Arguments: None

Returns: non-zero if the Esc has been received, zero otherwise

Here is the caller graph for this function:



5.26.2.8 char receivePeek (void)

Function: [receivePeek\(void\)](#)

Include: [Serial.h](#)

Description: Returns the next character in the receive buffer without removing it from the buffer

Arguments: None

Returns: The next received character

5.26.2.9 char receivePop (void)

Function: [receivePop\(void\)](#)

Include: [Serial.h](#)

Description: Pops the next received character from the received buffer

Arguments: None

Returns: The next character from the receive buffer

Here is the caller graph for this function:



5.26.2.10 void serialISR (void)

Function: [serialISR\(void\)](#)

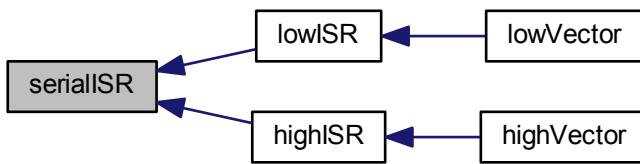
Include: [Serial.h](#)

Description: Acts as the interrupt service routine for the serial module

Arguments: None

Returns: None

Here is the caller graph for this function:



5.26.2.11 void transChar (char c)

Function: [transChar\(char c\)](#)

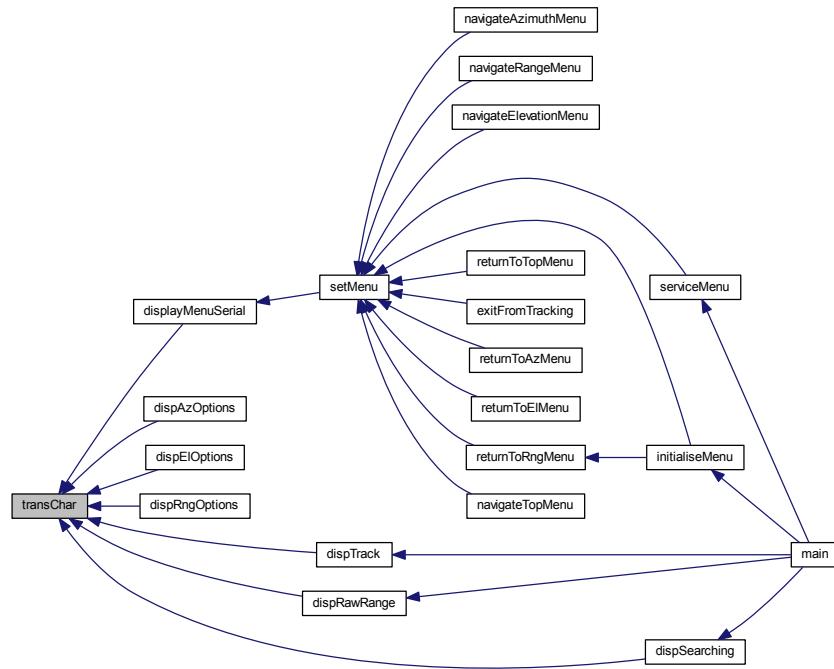
Include: [Serial.h](#)

Description: Transmits a single character

Arguments: c - character to transmit

Returns: None

Here is the caller graph for this function:



5.26.2.12 void transmit (*char * string*)

Function: [transmit\(char *string\)](#)

Include: [Serial.h](#)

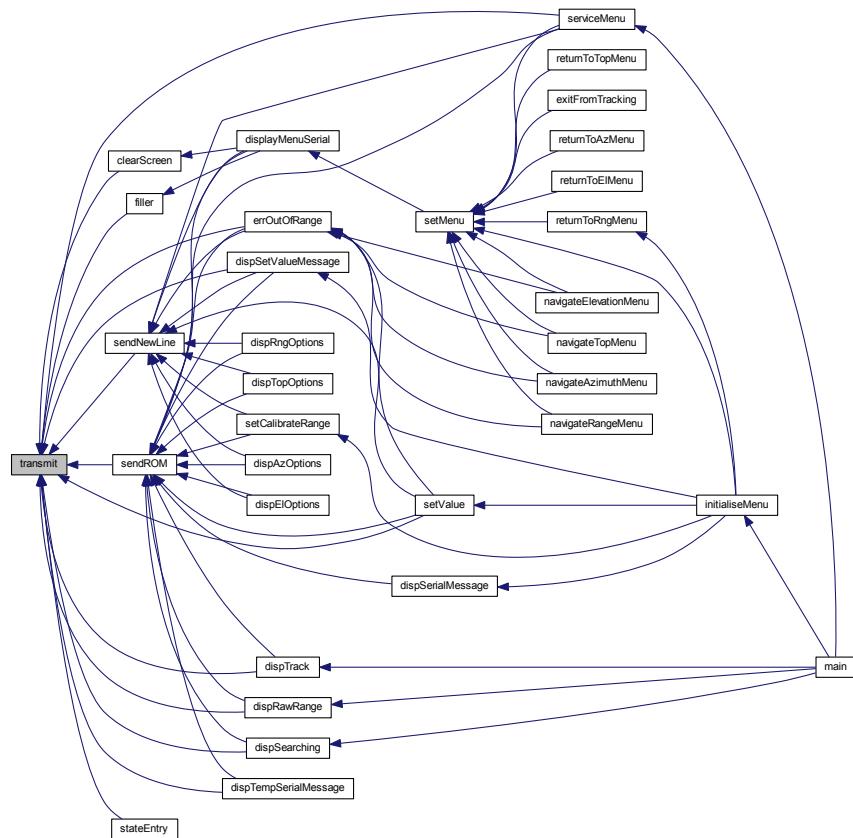
Description: Begins transmitting the string over serial (interrupt driven)

Arguments: string - pointer to the beginning of the string to transmit

Returns: None

NOTE: Must be Null Terminated! Cannot receive a literal.

Here is the caller graph for this function:



5.26.2.13 char transmitComplete(void)

Function: [transmitComplete\(void\)](#)

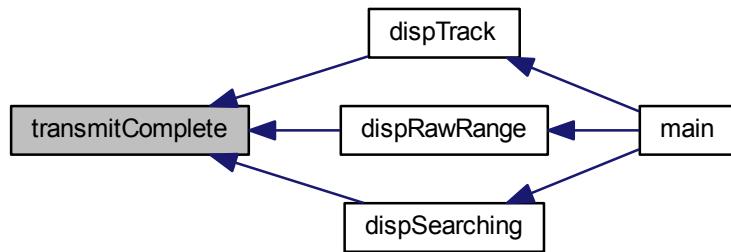
Include: [Serial.h](#)

Description: returns non-zero if the message has been completely transmitted e.g. if the transmit buffer is empty

Arguments: None

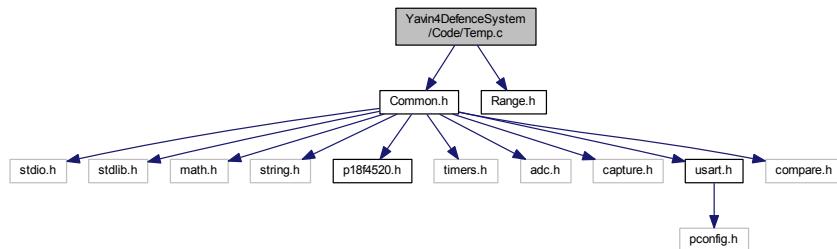
Returns: non-zero if all messages have been transmitted

Here is the caller graph for this function:



5.27 Yavin4DefenceSystem/Code/Temp.c File Reference

```
#include "Common.h"
#include "Range.h"
Include dependency graph for Temp.c:
```



Functions

- void [configureTemp](#) (void)
- unsigned char [readTempx2](#) (void)
- unsigned char [readTemp](#) (void)
- unsigned char [rawTemp](#) (void)
- void [calibrateTemp](#) (unsigned char reference)
- unsigned char [getTemp](#) (void)

Variables

- static signed char [calibration_offset](#) = 0
- static unsigned char [lastTempx2](#)

5.27.1 Function Documentation

5.27.1.1 void calibrateTemp (unsigned char *reference*)

Function: [calibrateTemp\(unsigned char reference\)](#)

Include: [Temp.h](#)

Description: calibrates the temperature sensor by updating the calibration offset variable

Arguments: *reference* - Reference temperature in deg C

Returns: None

Note: This function does not perform a temperature read, but uses the last value. This is because the *readTemp* function automatically calibrates.

5.27.1.2 void configureTemp (void)

Function: [configureTemp\(void\)](#)

Include: [Temp](#)

Description: Configures the temperature module for use

Arguments: None

Returns: None

Here is the call graph for this function:



5.27.1.3 unsigned char getTemp (void)

Function: [getTemp\(void\)](#)

Include: [Temp.h](#)

Description: calibrates the temperature sensor by updating the calibration offset variable

Arguments: *reference* - Reference temperature in deg C

Returns: None

5.27.1.4 unsigned char rawTemp (void)

Function: [rawTemp\(void\)](#)

Include: [Temp.h](#)

Description: Returns the raw (uncalibrated temperature)

Arguments: None

Returns: Temp (in deg celsius) as an unsigned char

5.27.1.5 `unsigned char readTemp (void)`

Function: [readTemp\(void\)](#)

Include: [Temp.h](#)

Description: Reads the temperature from the TEMP sensor

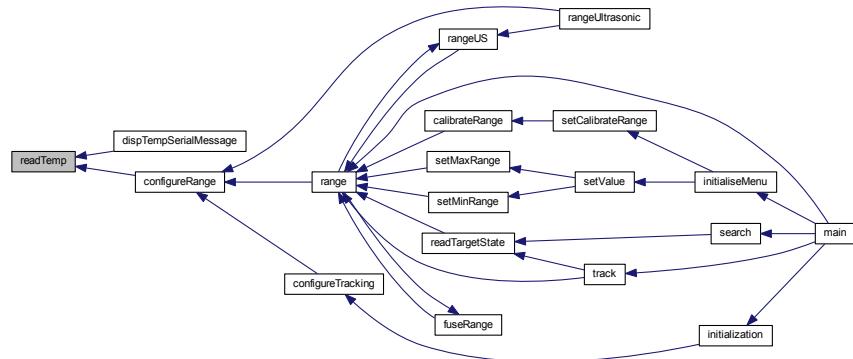
Arguments: None

Returns: Temp (in deg celsius) as an unsigned char

Here is the call graph for this function:



Here is the caller graph for this function:

5.27.1.6 `unsigned char readTempx2 (void)`

Function: [readTempx2\(void\)](#)

Include: [Temp.h](#)

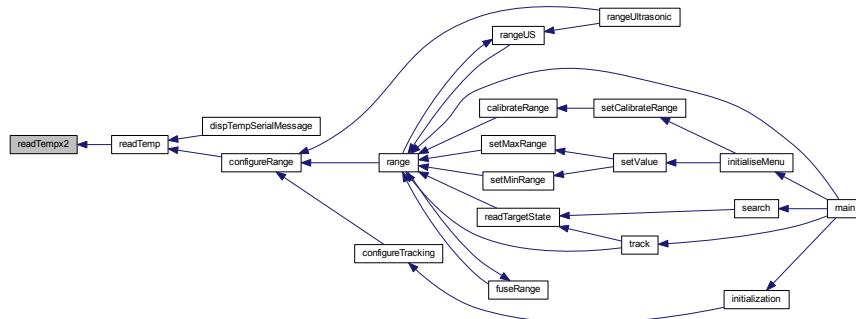
Description: Reads the temperature from the Temp sensor

Arguments: None

Returns: temp x 2 (in deg celsius) as an unsigned char

Todo Test and debug this function

Here is the caller graph for this function:



5.27.2 Variable Documentation

5.27.2.1 signed char calibration_offset = 0 [static]

File: [Temp.c](#) Author: Grant

Description: Contains all the functionality for the Temp module.

Duties: -Samples the temperature sensor -Stores the last temperature value -Calibrates the temperature sensor

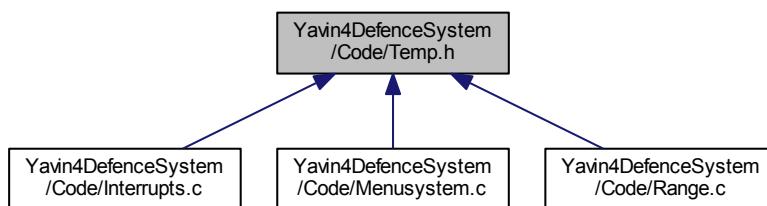
Functions:

Created on 7 September 2014, 4:12 PM

5.27.2.2 unsigned char lastTempx2 [static]

5.28 Yavin4DefenceSystem/Code/Temp.h File Reference

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



Macros

- #define TEMP_H

Functions

- void [configureTemp](#) (void)

- `unsigned char rawTemp (void)`
- `unsigned char readTemp (void)`
- `unsigned char readTempx2 (void)`
- `void calibrateTemp (unsigned char reference)`
- `unsigned char getTemp (void)`

5.28.1 Macro Definition Documentation

5.28.1.1 `#define TEMP_H`

5.28.2 Function Documentation

5.28.2.1 `void calibrateTemp (unsigned char reference)`

Function: `calibrateTemp(unsigned char reference)`

Include: `Temp.h`

Description: calibrates the temperature sensor by updating the calibration offset variable

Arguments: `reference` - Reference temperature in deg C

Returns: None

Note: This function does not perform a temperature read, but uses the last value. This is because the `readTemp` function automatically calibrates.

5.28.2.2 `void configureTemp (void)`

File: `Temp.h` Author: Grant

Description: Contains all the functionality and variables for the temp module. All unnecessary functions and variables should be shielded from the external program, and interfaced with accessor and mutator functions

Contains: -Configure function -Read temperature functions -Get (last) temperature (read) function -Calibrate Temperature function

Created on 17 September 2014, 2:16 PM

Function: `configureTemp(void)`

Include: `Temp`

Description: Configures the temperature module for use

Arguments: None

Returns: None

Function: `configureTemp(void)`

Include: `Temp`

Description: Configures the temperature module for use

Arguments: None

Returns: None

Here is the call graph for this function:



5.28.2.3 unsigned char getTemp (void)

Function: [calibrationTemp\(unsigned char reference\)](#)

Include: [Temp.h](#)

Description: calibrates the temperature sensor by updating the calibration offset variable

Arguments: reference - Reference temperature in deg C

Returns: None

5.28.2.4 unsigned char rawTemp (void)

Function: [rawTemp\(void\)](#)

Include: [Temp.h](#)

Description: Returns the raw (uncalibrated temperature)

Arguments: None

Returns: Temp (in deg celsius) as an unsigned char

5.28.2.5 unsigned char readTemp (void)

Function: [readTemp\(void\)](#)

Include: [Temp.h](#)

Description: Reads the temperature from the TEMP sensor

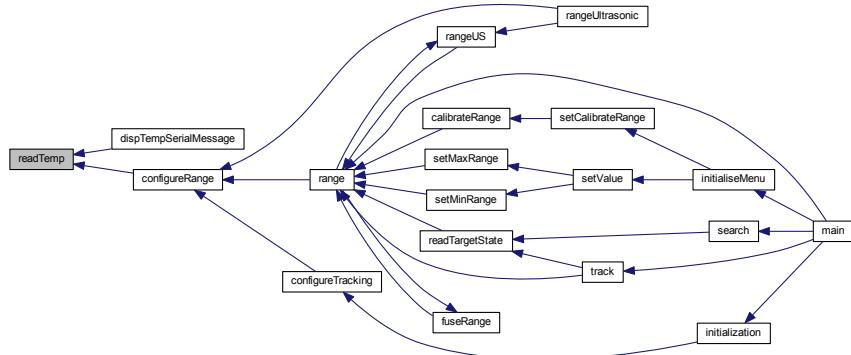
Arguments: None

Returns: Temp (in deg celsius) as an unsigned char

Here is the call graph for this function:



Here is the caller graph for this function:



5.28.2.6 unsigned char readTempx2 (void)

Function: `readTempx2(void)`

Include: Temp.h

Description: Reads the temperature from the Temp sensor

Arguments: None

Returns: temp x 2 (in deg celsius) as an unsigned char

Todo Test and debug this function

Function: `readTempx2(void)`

Include: Temp.h

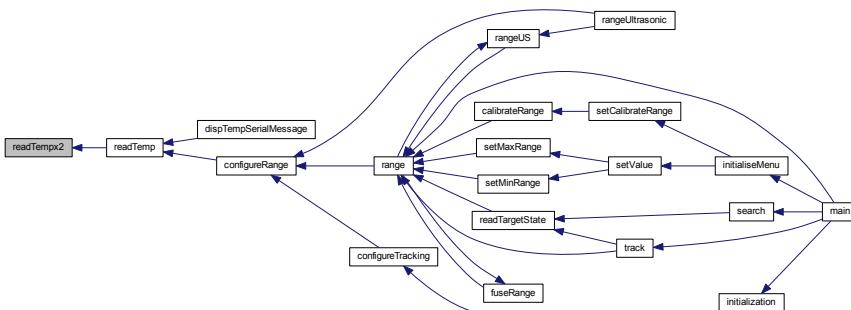
Description: Reads the temperature from the Temp sensor

Arguments: None

Returns: temp x 2 (in deg celsius) as an unsigned char

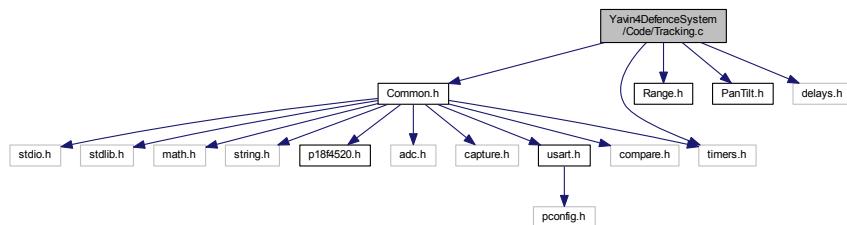
Todo Test and debug this function

Here is the caller graph for this function:



5.29 Yavin4DefenceSystem/Code/Tracking.c File Reference

```
#include "Common.h"
#include "Range.h"
#include "PanTilt.h"
#include <delays.h>
#include <timers.h>
Include dependency graph for Tracking.c:
```



Data Structures

- struct [TargetStateData](#)
Stores data about the last target track.

Macros

- #define diff 8
- #define TARGET_RAD 30
- #define sampleTargetState weight
- #define TIMER TMR2

Functions

- static char [newAngle](#) (char angle, [TargetStateData](#) target_data)
- void [configureTracking](#) (void)
- void [search](#) ([systemState](#) *state)
- void [trackingISR](#) (void)
- [TrackingData](#) [track](#) ([systemState](#) *state)
- static [Direction](#) [prediction](#) ([Direction](#) current)

5.29.1 Macro Definition Documentation

5.29.1.1 #define diff 8

File: [Tracking.c](#) Author: Grant

Description: Contains all the functionality for the Tracking Module. This module contains the functionality and algorithms used to search for, and track the target. This module makes use of PanTilt, Range and Temp modules, and configures them automatically from a call to `configureTrack()`.

Duties: -Control and coordinate Pan Tilt and range sensors -Implement searching and tracking algorithms -Predict next position of target???

Functions:

Created on 15 September 2014, 1:42 PM

5.29.1.2 #define sampleTargetState weight

5.29.1.3 #define TARGET_RAD 30

5.29.1.4 #define TIMER TMR2

5.29.2 Function Documentation

5.29.2.1 void configureTracking (void)

Function: [configureTracking\(void\)](#)

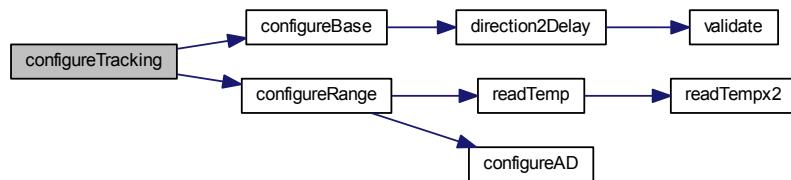
Include: [Tracking.h](#)

Description: Configures the System to begin tracking/searching

Arguments: None

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.29.2.2 static char newAngle (char angle, TargetStateData target_data) [static]

Function: [newAngle\(void\)](#)

Include: Local to [Tracking.c](#)

Description: Calculates the new angle offset - So when the target position is better known the sampling locations are more fine. Not currently implemented

Arguments: angle - The previous angle

`target_data` - The data from the last target sample

Returns: Angle - The new angle to offset the direction

5.29.2.3 static Direction prediction (`Direction current`) [static]

Function: [prediction\(Direction current\)](#)

Include: Local to [Tracking.c](#)

Description: Predicts where the object is likely to be found based on previous movement and prediction algorithms.
Not currently implemented

Arguments: `current` - The current position of the target

Returns: [Direction](#) - The Predicted likely location

5.29.2.4 void search (`systemState * state`)

Function: [search\(void\)](#)

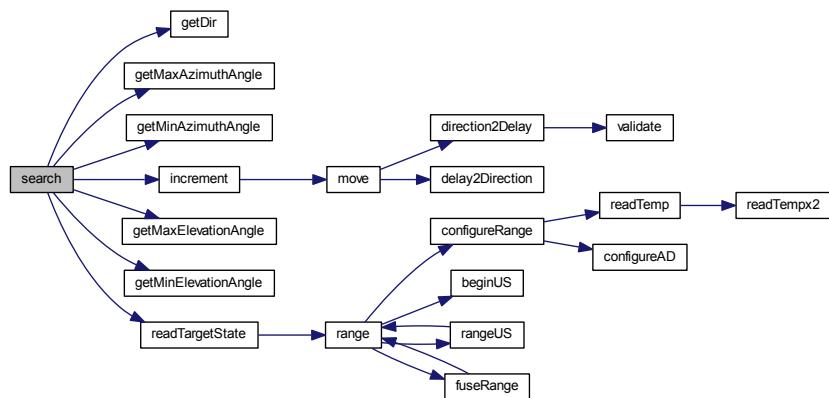
Include: [Tracking.h](#)

Description: Performs an incremental change in position. Local variables store previous movements to create a raster-like search pattern. Then samples the range sensors and determines the next system state.

Arguments: `state` - a pointer to the current system state

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.29.2.5 TrackingData track (systemState * state)

Function: track(void)

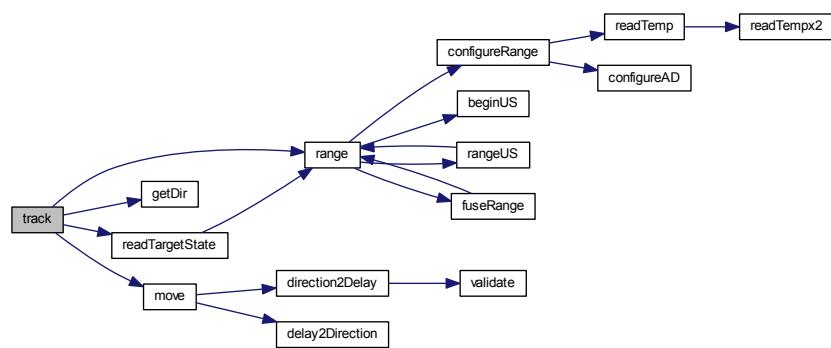
Include: [Tracking.h](#)

Description: Takes a number of samples at the previous target location, and several discrete locations around the target location and takes a weighted average based on the sampled signal return

Arguments: state - A pointer to the current system state

Returns: TargetData - The current target information (Azimuth, inclination and range to the target). This information is then used for the Display in the User interface module.

Here is the call graph for this function:



Here is the caller graph for this function:



5.29.2.6 void trackingISR (void)

Function: [trackingISR\(void\)](#)

Include: [Tracking.h](#)

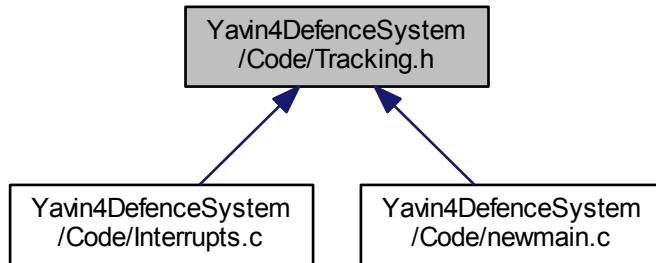
Description: Acts as the Interrupt Service Routine for the Tracking module Not currently implemented

Arguments: None

Returns: None

5.30 Yavin4DefenceSystem/Code/Tracking.h File Reference

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



Macros

- `#define TRACK_INT 0`
- `#define TRACK_H`

Functions

- `void configureTracking (void)`
- `void search (systemState *state)`
- `TrackingData track (systemState *state)`
- `void trackingISR (void)`

5.30.1 Macro Definition Documentation

5.30.1.1 `#define TRACK_H`

5.30.1.2 `#define TRACK_INT 0`

File: [Tracking.h](#) Author: Grant

Description: Public interface function for the Tracking Module. This module contains the functionality and algorithms used to search for, and track the target. This module makes use of PanTilt, Range and Temp modules, and configures them automatically from a call to `configureTrack()`.

Contains: -Configure tracking -Searching algorithm -Track algorithm

Created on 15 September 2014, 1:41 PM

5.30.2 Function Documentation

5.30.2.1 `void configureTracking (void)`

Function: [configureTracking\(void\)](#)

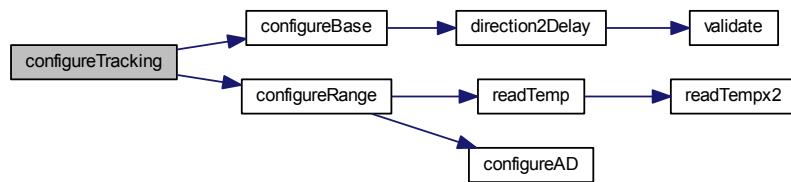
Include: [Tracking.h](#)

Description: Configures the System to begin tracking/searching

Arguments: None

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.30.2.2 void search (systemState * state)

Function: `search(void)`

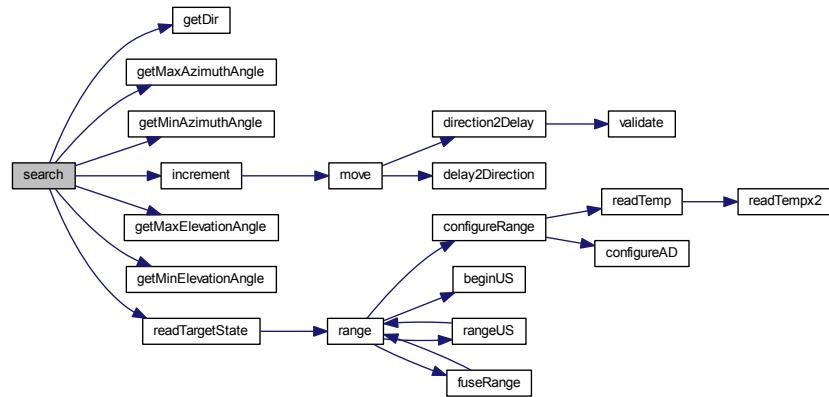
Include: [Tracking.h](#)

Description: Performs an incremental change in position. Local variables store previous movements to create a raster-like search pattern. Then samples the range sensors and determines the next system state.

Arguments: `state` - a pointer to the current system state

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.30.2.3 TrackingData track(`systemState * state`)

Function: `track(void)`

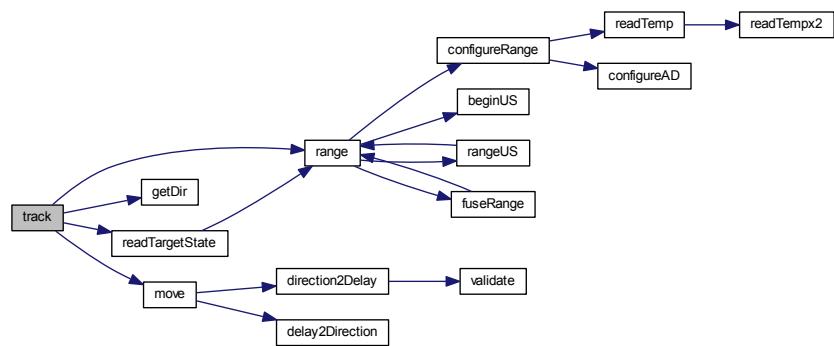
Include: [Tracking.h](#)

Description: Takes a number of samples at the previous target location, and several discrete locations around the target location and takes a weighted average based on the sampled signal return

Arguments: `state` - A pointer to the current system state

Returns: `TargetData` - The current target information (Azimuth, inclination and range to the target). This information is then used for the Display in the User interface module.

Here is the call graph for this function:



Here is the caller graph for this function:



5.30.2.4 void trackingISR (void)

Function: [trackingISR\(void\)](#)

Include: [Tracking.h](#)

Description: Acts as the Interrupt Service Routine for the Tracking module Not currently implemented

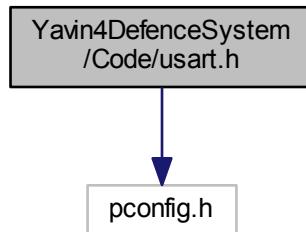
Arguments: None

Returns: None

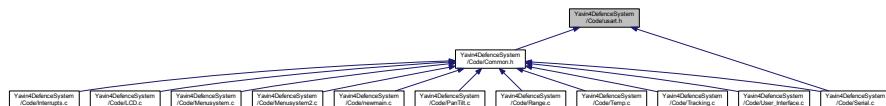
5.31 Yavin4DefenceSystem/Code/usart.h File Reference

```
#include <pconfig.h>
```

Include dependency graph for usart.h:



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



Macros

- #define **MEM_MODEL** far
- #define **USART_TX_INT_ON** 0b11111111
- #define **USART_TX_INT_OFF** 0b01111111
- #define **USART_RX_INT_ON** 0b11111111
- #define **USART_RX_INT_OFF** 0b10111111
- #define **USART_BRGH_HIGH** 0b11111111
- #define **USART_BRGH_LOW** 0b11101111
- #define **USART_CONT_RX** 0b11111111
- #define **USART_SINGLE_RX** 0b11110111
- #define **USART_SYNC_MASTER** 0b11111111
- #define **USART_SYNC_SLAVE** 0b11111011
- #define **USART_NINE_BIT** 0b11111111
- #define **USART_EIGHT_BIT** 0b11111101
- #define **USART_SYNCH_MODE** 0b11111111
- #define **USART_ASYNCH_MODE** 0b11111110
- #define **USART_ADDEN_ON** 0b11111111
- #define **USART_ADDEN_OFF** 0b11011111
- #define **BAUD_IDLE_RX_PIN_STATE_HIGH** 0b11011111
- #define **BAUD_IDLE_RX_PIN_STATE_LOW** 0b11111111
- #define **BAUD_IDLE_TX_PIN_STATE_HIGH** 0b11101111
- #define **BAUD_IDLE_TX_PIN_STATE_LOW** 0b11111111

5.31.1 Macro Definition Documentation

5.31.1.1 `#define BAUD_IDLE_RX_PIN_STATE_HIGH 0b11011111`

5.31.1.2 `#define BAUD_IDLE_RX_PIN_STATE_LOW 0b11111111`

5.31.1.3 `#define BAUD_IDLE_TX_PIN_STATE_HIGH 0b11101111`

5.31.1.4 `#define BAUD_IDLE_TX_PIN_STATE_LOW 0b11111111`

5.31.1.5 `#define MEM_MODEL far`

5.31.1.6 `#define USART_ADDEN_OFF 0b11011111`

5.31.1.7 `#define USART_ADDEN_ON 0b11111111`

5.31.1.8 `#define USART_ASYNCH_MODE 0b11111110`

5.31.1.9 `#define USART_BRGH_HIGH 0b11111111`

5.31.1.10 `#define USART_BRGH_LOW 0b11101111`

5.31.1.11 `#define USART_CONT_RX 0b11111111`

5.31.1.12 `#define USART_EIGHT_BIT 0b11111101`

5.31.1.13 `#define USART_NINE_BIT 0b11111111`

5.31.1.14 `#define USART_RX_INT_OFF 0b10111111`

5.31.1.15 `#define USART_RX_INT_ON 0b11111111`

5.31.1.16 `#define USART_SINGLE_RX 0b11110111`

5.31.1.17 `#define USART_SYNC_MASTER 0b11111111`

5.31.1.18 `#define USART_SYNC_SLAVE 0b11111011`

5.31.1.19 `#define USART_SYNCH_MODE 0b11111111`

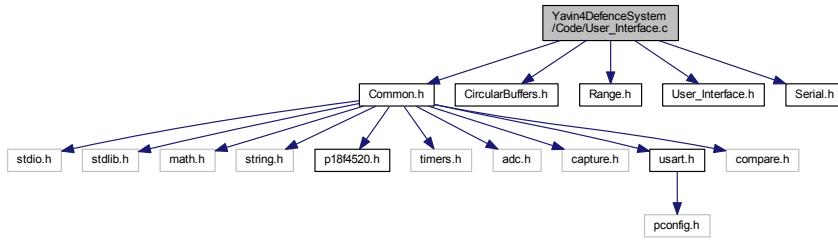
5.31.1.20 `#define USART_TX_INT_OFF 0b01111111`

5.31.1.21 `#define USART_TX_INT_ON 0b11111111`

5.32 Yavin4DefenceSystem/Code/User_Interface.c File Reference

```
#include "Common.h"
#include "CircularBuffers.h"
#include "Range.h"
#include "User_Interface.h"
#include "Serial.h"
```

Include dependency graph for User_Interface.c:



Macros

- #define CONFIRM_PRESS INTCONbits.INT0IF
- #define BACK_PRESS INTCON3bits.INT1F
- #define ADC_MAX 1023

Functions

- void `display` (TrackingData data)
- void `userISR` (void)
- void `configUSER` (void)
- unsigned int `readDial` (unsigned int max)
- unsigned int `readDialForMenu` (unsigned int max)
- char `userEmpty` (void)
- char `userPop` (void)
- char `userPeek` (void)

Variables

- `circularBuffer receive`

5.32.1 Macro Definition Documentation

5.32.1.1 #define ADC_MAX 1023

5.32.1.2 #define BACK_PRESS INTCON3bits.INT1F

5.32.1.3 #define CONFIRM_PRESS INTCONbits.INT0IF

File: [User_Interface.c](#) Author: Grant

Description: Contains all the functionality for the User_Interface module. Works the same way as the serial module, but for the user interface.

Duties: -Stores any local user input in a receive buffer -Sends display data to the LCD

Created on 15 September 2014, 1:21 PM

5.32.2 Function Documentation

5.32.2.1 void configUSER (void)

Function: [configUSER\(void\)](#)

Include: [User_Interface.h](#)

Description: Configures the user interface for use

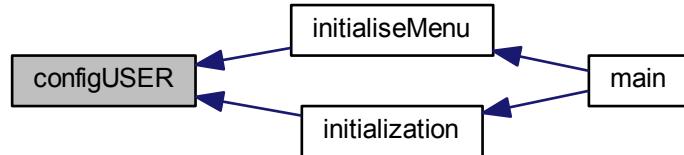
Arguments: None

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.32.2.2 void display (TrackingData data)

Function: [display\(TrackingData data\)](#)

Include: [User_Interface.h](#)

Description: displays the current tracking information

Arguments: data - Struct which contains all known data concerning the The position of the target. A Range of 0 indicates that no target was observed.

Returns: None

5.32.2.3 unsigned int readDial (unsigned int max)

Function: [readDial\(void\)](#)

Include:

Description: Read the position of the dial by splitting the value returned from the ADC into even discrete steps based on the given argument. For instance, an input of 7 will return a value from 0 to 7 based on the value found from the potentiometer

Arguments: int max - The number of maximum state to split values into

Returns: The dial position scaled by the maximum number

Here is the caller graph for this function:



5.32.2.4 unsigned int readDialForMenu (unsigned int max)

Function: `readDialForMEnu(void)`

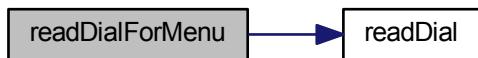
Include:

Description: Read the position of the dial for Navigation menus. Currently unused.

Arguments: None

Returns: Returns a value from 1 to max used in Navigation menus

Here is the call graph for this function:



5.32.2.5 char userEmpty (void)

Function: `userEmpty(void)`

Include: `User_Interface.h`

Description: returns non-zero if the user interface buffer is empty (i.e. no user input has been detected)

Arguments: None

Returns: if the user input buffer is empty

Here is the caller graph for this function:



5.32.2.6 void userISR(void)

Function: [userISR\(void\)](#)

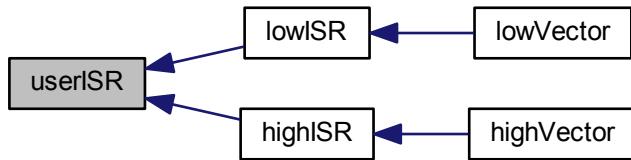
Include: [User_Interface.h](#)

Description: Acts as the ISR for the User_interface module

Arguments: None

Returns: None

Here is the caller graph for this function:



5.32.2.7 char userPeek(void)

Function: [userPeek\(void\)](#)

Include: [User_Interface.h](#)

Description: returns a character in the user_Interface buffer without removing it from the buffer

Arguments: None

Returns: a character in the user interface buffer

5.32.2.8 char userPop(void)

Function: [userPop\(void\)](#)

Include: [User_Interface.h](#)

Description: pops a character from the user interface receive buffer

Arguments: None

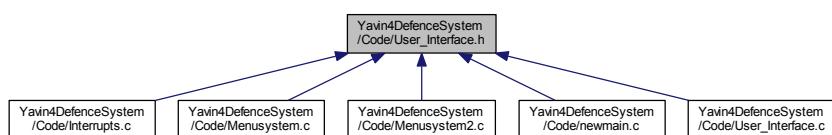
Returns: a character popped from the user Interface buffer

5.32.3 Variable Documentation

5.32.3.1 `circularBuffer receive`

5.33 Yavin4DefenceSystem/Code/User_Interface.h File Reference

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



Macros

- `#define USER_INT (RB_INT || INTO_INT || INT1_INT)`
- `#define CONFIRM_CHAR 0x0D`
- `#define BACK_CHAR 0x1B`
- `#define USER_H`

Functions

- `void display (TrackingData data)`
- `void userISR (void)`
- `void configUSER (void)`
- `char userEmpty (void)`
- `char userPop (void)`
- `char userPeek (void)`
- `unsigned int readDial (unsigned int max)`

5.33.1 Macro Definition Documentation

5.33.1.1 `#define BACK_CHAR 0x1B`

5.33.1.2 `#define CONFIRM_CHAR 0x0D`

5.33.1.3 `#define USER_H`

5.33.1.4 `#define USER_INT (RB_INT || INTO_INT || INT1_INT)`

5.33.2 Function Documentation

5.33.2.1 `void configUSER (void)`

Function: `configUSER(void)`

Include: `User_Interface.h`

Description: Configures the user interface for use

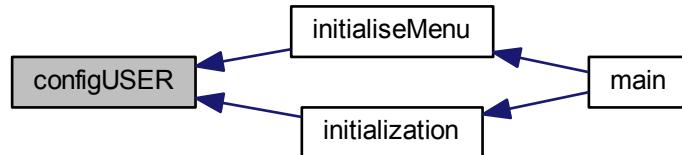
Arguments: None

Returns: None

Here is the call graph for this function:



Here is the caller graph for this function:



5.33.2.2 void display (TrackingData data)

Function: [display\(TrackingData data\)](#)

Include: [User_Interface.h](#)

Description: displays the current tracking information

Arguments: data - Struct which contains all known data concerning the The position of the target. A Range of 0 indicates that no target was observed.

Returns: None

5.33.2.3 unsigned int readDial (unsigned int max)

Function: [readDial\(void\)](#)

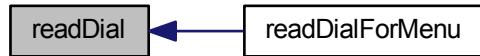
Include:

Description: Read the position of the dial by splitting the value returned from the ADC into even discrete steps based on the given argument. For instance, an input of 7 will return a value from 0 to 7 based on the value found from the potentiometer

Arguments: int max - The number of maximum state to split values into

Returns: The dial position scaled by the maximum number

Here is the caller graph for this function:



5.33.2.4 char userEmpty(void)

Function: [userEmpty\(void\)](#)

Include: [User_Interface.h](#)

Description: returns non-zero if the user interface buffer is empty (i.e. no user input has been detected)

Arguments: None

Returns: if the user input buffer is empty

Here is the caller graph for this function:



5.33.2.5 void userISR(void)

Function: [userISR\(void\)](#)

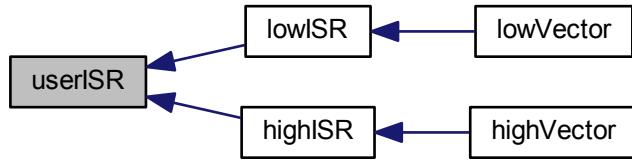
Include: [User_Interface.h](#)

Description: Acts as the ISR for the User_interface module

Arguments: None

Returns: None

Here is the caller graph for this function:



5.33.2.6 char userPeek(void)

Function: [userPeek\(void\)](#)

Include: [User_Interface.h](#)

Description: returns a character in the user_Interface buffer without removing it from the buffer

Arguments: None

Returns: a character in the user interface buffer

5.33.2.7 char userPop(void)

Function: [userPop\(void\)](#)

Include: [User_Interface.h](#)

Description: pops a character from the user interface receive buffer

Arguments: None

Returns: a character poped from the user Interface buffer