

Alpaca Power Pong

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b>Hierarchical Index</b>	<b>1</b>
1.1	Class Hierarchy . . . . .	1
<b>2</b>	<b>Class Index</b>	<b>3</b>
2.1	Class List . . . . .	3
<b>3</b>	<b>File Index</b>	<b>5</b>
3.1	File List . . . . .	5
<b>4</b>	<b>Class Documentation</b>	<b>7</b>
4.1	Adafruit_MCP23008 Class Reference . . . . .	7
4.1.1	Member Function Documentation . . . . .	7
4.1.1.1	begin() [1/2] . . . . .	8
4.1.1.2	begin() [2/2] . . . . .	8
4.1.1.3	digitalRead() . . . . .	8
4.1.1.4	digitalWrite() . . . . .	8
4.1.1.5	pinMode() . . . . .	8
4.1.1.6	pullUp() . . . . .	8
4.1.1.7	read8() . . . . .	9
4.1.1.8	readGPIO() . . . . .	9
4.1.1.9	write8() . . . . .	9
4.1.1.10	writeGPIO() . . . . .	9
4.1.2	Member Data Documentation . . . . .	9
4.1.2.1	fd . . . . .	9
4.1.2.2	i2caddr . . . . .	9

4.2	Button Class Reference . . . . .	10
4.2.1	Constructor & Destructor Documentation . . . . .	10
4.2.1.1	Button() . . . . .	10
4.2.1.2	~Button() . . . . .	10
4.2.2	Member Function Documentation . . . . .	10
4.2.2.1	run() . . . . .	11
4.2.3	Member Data Documentation . . . . .	11
4.2.3.1	fd . . . . .	11
4.2.3.2	mq_ . . . . .	11
4.2.3.3	running . . . . .	11
4.2.3.4	value . . . . .	11
4.3	buttonMessage Struct Reference . . . . .	11
4.3.1	Constructor & Destructor Documentation . . . . .	12
4.3.1.1	buttonMessage() . . . . .	12
4.3.2	Member Data Documentation . . . . .	12
4.3.2.1	x . . . . .	12
4.4	cursorCoord Struct Reference . . . . .	12
4.4.1	Detailed Description . . . . .	12
4.4.2	Member Data Documentation . . . . .	13
4.4.2.1	dir_ . . . . .	13
4.4.2.2	line_ . . . . .	13
4.4.2.3	row_ . . . . .	13
4.4.2.4	type_ . . . . .	13
4.5	WebsiteScoreHandling::DATESTRUCT Struct Reference . . . . .	13
4.5.1	Member Data Documentation . . . . .	13
4.5.1.1	day_ . . . . .	14
4.5.1.2	month_ . . . . .	14
4.5.1.3	year_ . . . . .	14
4.6	I2C_reg Class Reference . . . . .	14
4.6.1	Constructor & Destructor Documentation . . . . .	15

4.6.1.1	I2C_reg()	16
4.6.1.2	~I2C_reg()	16
4.6.2	Member Function Documentation	16
4.6.2.1	displayInit()	16
4.6.2.2	displayRead()	16
4.6.2.3	displayWrite()	16
4.6.2.4	getInstance()	17
4.6.2.5	getMsgQueue()	17
4.6.2.6	handleMsg()	17
4.6.2.7	lockI2C()	17
4.6.2.8	psocUpdate()	17
4.6.2.9	run()	17
4.6.2.10	sendArduinoMessage()	18
4.6.2.11	sendPsoc1Message()	18
4.6.2.12	sendPsoc2Message()	18
4.6.2.13	sendPsocBroadcast()	18
4.6.2.14	sendPsocMessage()	18
4.6.2.15	setArduinoMessage()	18
4.6.2.16	setMsgQueueScoreSystem()	19
4.6.2.17	setPsocMessage()	19
4.6.2.18	setUp()	19
4.6.2.19	stopPsocPolling()	19
4.6.2.20	testStatic()	19
4.6.2.21	unlockI2C()	19
4.6.3	Member Data Documentation	20
4.6.3.1	arduinoAdress_	20
4.6.3.2	arduinoMessage_	20
4.6.3.3	cond_	20
4.6.3.4	currentAddr_	20
4.6.3.5	displayMQ_	20

4.6.3.6	fd_ . . . . .	20
4.6.3.7	mq_ . . . . .	20
4.6.3.8	mut_ . . . . .	21
4.6.3.9	psoc1Adress_ . . . . .	21
4.6.3.10	psoc2Adress_ . . . . .	21
4.6.3.11	psocMessage_ . . . . .	21
4.6.3.12	receivingPsoC_ . . . . .	21
4.6.3.13	running_ . . . . .	21
4.6.3.14	ScoreSystemMQ_ . . . . .	21
4.6.3.15	screenAdress_ . . . . .	21
4.6.3.16	state_ . . . . .	22
4.6.3.17	timer1_ . . . . .	22
4.6.3.18	tt_ . . . . .	22
4.7	LCD Class Reference . . . . .	22
4.7.1	Constructor & Destructor Documentation . . . . .	23
4.7.1.1	LCD() . . . . .	23
4.7.2	Member Function Documentation . . . . .	23
4.7.2.1	begin() . . . . .	24
4.7.2.2	blink() . . . . .	24
4.7.2.3	charWrite() . . . . .	24
4.7.2.4	clear() . . . . .	24
4.7.2.5	command() . . . . .	24
4.7.2.6	cursor() . . . . .	24
4.7.2.7	display() . . . . .	25
4.7.2.8	home() . . . . .	25
4.7.2.9	lcdWrite_four_bits() . . . . .	25
4.7.2.10	noBlink() . . . . .	25
4.7.2.11	noCursor() . . . . .	25
4.7.2.12	noDisplay() . . . . .	25
4.7.2.13	readReg() . . . . .	26

4.7.2.14	<a href="#">send()</a>	26
4.7.2.15	<a href="#">setCursor()</a>	26
4.7.2.16	<a href="#">stringWrite()</a>	26
4.7.3	<a href="#">Member Data Documentation</a>	26
4.7.3.1	<a href="#">_data_pins</a>	26
4.7.3.2	<a href="#">_enable_pin</a>	27
4.7.3.3	<a href="#">_rs_pin</a>	27
4.7.3.4	<a href="#">_rw_pin</a>	27
4.7.3.5	<a href="#">currentLine</a>	27
4.7.3.6	<a href="#">displayControl</a>	27
4.7.3.7	<a href="#">displayFunction</a>	27
4.7.3.8	<a href="#">displayMode</a>	27
4.7.3.9	<a href="#">En</a>	27
4.7.3.10	<a href="#">i2c_</a>	28
4.7.3.11	<a href="#">numLines</a>	28
4.7.3.12	<a href="#">Rs</a>	28
4.7.3.13	<a href="#">Rw</a>	28
4.8	<a href="#">Page Class Reference</a>	28
4.8.1	<a href="#">Constructor &amp; Destructor Documentation</a>	29
4.8.1.1	<a href="#">Page()</a>	29
4.8.2	<a href="#">Member Function Documentation</a>	29
4.8.2.1	<a href="#">buttonLeft()</a>	29
4.8.2.2	<a href="#">buttonPressed()</a>	29
4.8.2.3	<a href="#">buttonRight()</a>	30
4.8.2.4	<a href="#">displayScreen()</a>	30
4.8.2.5	<a href="#">resetPage()</a>	30
4.8.3	<a href="#">Member Data Documentation</a>	30
4.8.3.1	<a href="#">currentChar</a>	30
4.8.3.2	<a href="#">cursorPos_</a>	30
4.8.3.3	<a href="#">nrCursorPos_</a>	31

4.8.3.4	pageText	31
4.8.3.5	possibleCursorPos	31
4.8.3.6	selectingChar	31
4.8.3.7	teamEnter_	31
4.8.3.8	teamNameArr	31
4.9	psocUpdateMessage Struct Reference	32
4.9.1	Constructor & Destructor Documentation	32
4.9.1.1	psocUpdateMessage()	32
4.9.2	Member Data Documentation	32
4.9.2.1	val_	32
4.10	ScoreSystemCtrl Class Reference	32
4.10.1	Constructor & Destructor Documentation	33
4.10.1.1	ScoreSystemCtrl()	34
4.10.2	Member Function Documentation	34
4.10.2.1	endGame()	34
4.10.2.2	getIP()	34
4.10.2.3	getMsgQueue()	34
4.10.2.4	handleMsg()	34
4.10.2.5	handlePsocUpdate()	35
4.10.2.6	handleState()	35
4.10.2.7	resetGame()	35
4.10.2.8	run()	35
4.10.3	Member Data Documentation	35
4.10.3.1	btn	35
4.10.3.2	buttonThread	36
4.10.3.3	collectiveDoubleShots_	36
4.10.3.4	currentScreen	36
4.10.3.5	display	36
4.10.3.6	gameTime_	36
4.10.3.7	i2cThread	36



4.10.3.8	ip_ . . . . .	36
4.10.3.9	mq_ . . . . .	36
4.10.3.10	nr_Cups_In_Game_ . . . . .	37
4.10.3.11	pages_ . . . . .	37
4.10.3.12	reArranging_ . . . . .	37
4.10.3.13	reArrangingZone_ . . . . .	37
4.10.3.14	score_Team_1_ . . . . .	37
4.10.3.15	score_Team_2_ . . . . .	37
4.10.3.16	state_ . . . . .	37
4.10.3.17	teamName1_ . . . . .	38
4.10.3.18	teamName2_ . . . . .	38
4.10.3.19	tempString_ . . . . .	38
4.10.3.20	websitePtr_ . . . . .	38
4.10.3.21	zone1State_ . . . . .	38
4.10.3.22	zone2State_ . . . . .	38
4.11	Test Class Reference . . . . .	39
4.11.1	Member Function Documentation . . . . .	39
4.11.1.1	run() . . . . .	39
4.12	Timer Class Reference . . . . .	39
4.12.1	Constructor & Destructor Documentation . . . . .	40
4.12.1.1	Timer() . . . . .	40
4.12.1.2	~Timer() . . . . .	40
4.12.2	Member Function Documentation . . . . .	40
4.12.2.1	run() . . . . .	40
4.12.2.2	stopTimer() . . . . .	40
4.12.3	Member Data Documentation . . . . .	40
4.12.3.1	id_ . . . . .	41
4.12.3.2	mq_ . . . . .	41
4.12.3.3	running_ . . . . .	41
4.12.3.4	timeout_ . . . . .	41

4.13 WebsiteScoreHandling::TIMESTRUCT Struct Reference . . . . .	41
4.13.1 Member Data Documentation . . . . .	41
4.13.1.1 hours_ . . . . .	41
4.13.1.2 minutes_ . . . . .	42
4.13.1.3 seconds_ . . . . .	42
4.14 WebsiteScoreHandling Class Reference . . . . .	42
4.14.1 Detailed Description . . . . .	43
4.14.2 Constructor & Destructor Documentation . . . . .	43
4.14.2.1 WebsiteScoreHandling() . . . . .	43
4.14.3 Member Function Documentation . . . . .	44
4.14.3.1 getCurrentID() . . . . .	44
4.14.3.2 getDate() . . . . .	44
4.14.3.3 getEndTime() . . . . .	44
4.14.3.4 getGameTime() . . . . .	45
4.14.3.5 getScoreTeam() . . . . .	45
4.14.3.6 getStartTime() . . . . .	45
4.14.3.7 getStartTimeDate() . . . . .	45
4.14.3.8 getTeamName() . . . . .	45
4.14.3.9 getTotalDoubleCupShots() . . . . .	45
4.14.3.10 setDate() . . . . .	45
4.14.3.11 setEndTime() . . . . .	46
4.14.3.12 setGameTime() . . . . .	46
4.14.3.13 setScoreTeam() . . . . .	46
4.14.3.14 setStartTime() . . . . .	46
4.14.3.15 setTeamName() . . . . .	46
4.14.3.16 setTotalDoubleCupShots() . . . . .	47
4.14.3.17 writeToCSV() . . . . .	47
4.14.4 Member Data Documentation . . . . .	47
4.14.4.1 currentGameID . . . . .	47
4.14.4.2 currentLine . . . . .	48

4.14.4.3	date_ . . . . .	48
4.14.4.4	errorFs_ . . . . .	48
4.14.4.5	fs_ . . . . .	48
4.14.4.6	gameIDInteger . . . . .	48
4.14.4.7	gameTime_ . . . . .	48
4.14.4.8	headers . . . . .	48
4.14.4.9	ifs_ . . . . .	49
4.14.4.10	lastGameID . . . . .	49
4.14.4.11	lineNumber . . . . .	49
4.14.4.12	lineToCopy . . . . .	49
4.14.4.13	lineToRead . . . . .	49
4.14.4.14	newName . . . . .	49
4.14.4.15	oldName . . . . .	49
4.14.4.16	scoreTeam1_ . . . . .	49
4.14.4.17	scoreTeam2_ . . . . .	50
4.14.4.18	t . . . . .	50
4.14.4.19	teamName1_ . . . . .	50
4.14.4.20	teamName2_ . . . . .	50
4.14.4.21	timeEnd_ . . . . .	50
4.14.4.22	timeKeeper_ . . . . .	50
4.14.4.23	timeStart_ . . . . .	50
4.14.4.24	timeStartDate_ . . . . .	50
4.14.4.25	totalDoubleCupShots_ . . . . .	50

<b>5</b>	<b>File Documentation</b>	<b>51</b>
5.1	<a href="#">inc/osapi/ScoreSystem/Adafruit_MCP23008.hpp File Reference</a>	51
5.1.1	Macro Definition Documentation	51
5.1.1.1	MCP23008_ADDRESS	51
5.1.1.2	MCP23008_DEFVAL	52
5.1.1.3	MCP23008_GPINTEN	52
5.1.1.4	MCP23008_GPIO	52
5.1.1.5	MCP23008_GPPU	52
5.1.1.6	MCP23008_INTCAP	52
5.1.1.7	MCP23008_INTCON	52
5.1.1.8	MCP23008_INTF	52
5.1.1.9	MCP23008_IOCON	52
5.1.1.10	MCP23008_IODIR	53
5.1.1.11	MCP23008_IPOL	53
5.1.1.12	MCP23008_OLAT	53
5.2	<a href="#">inc/osapi/ScoreSystem/Button.hpp File Reference</a>	53
5.2.1	Enumeration Type Documentation	53
5.2.1.1	buttonEvent	53
5.3	<a href="#">inc/osapi/ScoreSystem/I2C_reg.hpp File Reference</a>	54
5.3.1	Enumeration Type Documentation	54
5.3.1.1	i2c_messages	54
5.4	<a href="#">inc/osapi/ScoreSystem/LCD.hpp File Reference</a>	55
5.4.1	Macro Definition Documentation	56
5.4.1.1	HIGH	56
5.4.1.2	INPUT	56
5.4.1.3	LCD_1LINE	56
5.4.1.4	LCD_2LINE	56
5.4.1.5	LCD_4BITMODE	56
5.4.1.6	LCD_5x10DOTS	56
5.4.1.7	LCD_5x8DOTS	56

5.4.1.8	<a href="#">LCD_8BITMODE</a>	57
5.4.1.9	<a href="#">LCD_BACKLIGHT</a>	57
5.4.1.10	<a href="#">LCD_BLINKOFF</a>	57
5.4.1.11	<a href="#">LCD_BLINKON</a>	57
5.4.1.12	<a href="#">LCD_CLEARDISPLAY</a>	57
5.4.1.13	<a href="#">LCD_CURSORMOVE</a>	57
5.4.1.14	<a href="#">LCD_CURSOROFF</a>	57
5.4.1.15	<a href="#">LCD_CURSORON</a>	57
5.4.1.16	<a href="#">LCD_CURSORSHIFT</a>	58
5.4.1.17	<a href="#">LCD_DISPLAYCONTROL</a>	58
5.4.1.18	<a href="#">LCD_DISPLAYMOVE</a>	58
5.4.1.19	<a href="#">LCD_DISPLAYOFF</a>	58
5.4.1.20	<a href="#">LCD_DISPLAYON</a>	58
5.4.1.21	<a href="#">LCD_ENTRYLEFT</a>	58
5.4.1.22	<a href="#">LCD_ENTRYMODESET</a>	58
5.4.1.23	<a href="#">LCD_ENTRYRIGHT</a>	58
5.4.1.24	<a href="#">LCD_ENTRYSHIFTDECREMENT</a>	59
5.4.1.25	<a href="#">LCD_ENTRYSHIFTINCREMENT</a>	59
5.4.1.26	<a href="#">LCD_FUNCTIONSET</a>	59
5.4.1.27	<a href="#">LCD_MOVELEFT</a>	59
5.4.1.28	<a href="#">LCD_MOVERIGHT</a>	59
5.4.1.29	<a href="#">LCD_NOBACKLIGHT</a>	59
5.4.1.30	<a href="#">LCD_RETURNHOME</a>	59
5.4.1.31	<a href="#">LCD_SETCGRAMADDR</a>	59
5.4.1.32	<a href="#">LCD_SETDDRAMADDR</a>	60
5.4.1.33	<a href="#">LOW</a>	60
5.4.1.34	<a href="#">OUTPUT</a>	60
5.5	<a href="#">inc/osapi/ScoreSystem/Page.hpp File Reference</a>	60
5.5.1	<a href="#">Enumeration Type Documentation</a>	60
5.5.1.1	<a href="#">pageEvent</a>	60

5.6	<a href="#">inc/osapi/ScoreSystem/ScoreSystemCtrl.hpp File Reference</a>	61
5.6.1	Enumeration Type Documentation	61
5.6.1.1	<a href="#">pSocMessages</a>	62
5.7	<a href="#">inc/osapi/ScoreSystem/test.hpp File Reference</a>	62
5.8	<a href="#">inc/osapi/ScoreSystem/Timer.hpp File Reference</a>	62
5.9	<a href="#">inc/osapi/ScoreSystem/WebsiteScoreHandling.h File Reference</a>	62
5.9.1	Function Documentation	63
5.9.1.1	<a href="#">setNewIPJS()</a>	63
5.10	<a href="#">ScoreSystem/Adafruit_MCP23008.cpp File Reference</a>	63
5.11	<a href="#">ScoreSystem/Screen-ButtonTestCode/Adafruit_MCP23008.cpp File Reference</a>	64
5.12	<a href="#">ScoreSystem/ScreenTestCode/Adafruit_MCP23008.cpp File Reference</a>	64
5.13	<a href="#">ScoreSystem/Button.cpp File Reference</a>	64
5.14	<a href="#">ScoreSystem/Button_Driver/button_drv.c File Reference</a>	64
5.14.1	Macro Definition Documentation	65
5.14.1.1	<a href="#">ERRGOTO</a>	66
5.14.1.2	<a href="#">MAXLEN</a>	66
5.14.1.3	<a href="#">MODULE_DEBUG</a>	66
5.14.2	Function Documentation	66
5.14.2.1	<a href="#">buttonUpdate()</a>	66
5.14.2.2	<a href="#">DECLARE_WAIT_QUEUE_HEAD()</a>	66
5.14.2.3	<a href="#">MODULE_AUTHOR()</a>	67
5.14.2.4	<a href="#">module_exit()</a>	67
5.14.2.5	<a href="#">module_init()</a>	67
5.14.2.6	<a href="#">MODULE_LICENSE()</a>	67
5.14.2.7	<a href="#">plat_drv_exit()</a>	67
5.14.2.8	<a href="#">plat_drv_init()</a>	67
5.14.2.9	<a href="#">plat_drv_probe()</a>	67
5.14.2.10	<a href="#">plat_drv_read()</a>	68
5.14.2.11	<a href="#">plat_drv_remove()</a>	68
5.14.2.12	<a href="#">rotateLeftUpdate()</a>	68

5.14.2.13 rotateRightUpdate()	68
5.14.3 Variable Documentation	68
5.14.3.1 buttonPin	68
5.14.3.2 devno	68
5.14.3.3 leftPin	69
5.14.3.4 leftStatus	69
5.14.3.5 newValue	69
5.14.3.6 of_plat_drv_platform_device_match	69
5.14.3.7 plat_drv_cdev	69
5.14.3.8 plat_drv_class	69
5.14.3.9 plat_drv_fops	70
5.14.3.10 plat_drv_platform_driver	70
5.14.3.11 prevRightStatus	70
5.14.3.12 prevValue	70
5.14.3.13 rightPin	70
5.14.3.14 rightStatus	71
5.14.3.15 value	71
5.14.3.16 valueRead	71
5.15 ScoreSystem/Button_Driver/button_drv.mod.c File Reference	71
5.15.1 Function Documentation	71
5.15.1.1 __attribute__((1/3))	71
5.15.1.2 __attribute__((2/3))	72
5.15.1.3 __attribute__((3/3))	72
5.15.1.4 MODULE_INFO((1/2))	72
5.15.1.5 MODULE_INFO((2/2))	72
5.16 ScoreSystem/buttonTestCode/host/files/main.d File Reference	72
5.17 ScoreSystem/buttonTestCode/target/files/main.d File Reference	72
5.18 ScoreSystem/Screen-ButtonTestCode/host/files/main.d File Reference	72
5.19 ScoreSystem/Screen-ButtonTestCode/target/files/main.d File Reference	72
5.20 ScoreSystem/buttonTestCode/main.cpp File Reference	72

5.20.1	Function Documentation	73
5.20.1.1	main()	73
5.21	ScoreSystem/i2cTestCode/main.cpp File Reference	73
5.21.1	Function Documentation	73
5.21.1.1	main()	73
5.22	ScoreSystem/main.cpp File Reference	73
5.22.1	Function Documentation	74
5.22.1.1	main()	74
5.23	ScoreSystem/Screen-ButtonTestCode/main.cpp File Reference	74
5.23.1	Function Documentation	74
5.23.1.1	main()	74
5.24	ScoreSystem/ScreenTestCode/main.cpp File Reference	74
5.24.1	Function Documentation	74
5.24.1.1	main()	75
5.25	ScoreSystem/TimeTestCode/main.cpp File Reference	75
5.25.1	Function Documentation	75
5.25.1.1	main()	75
5.26	ScoreSystem/I2C_reg.cpp File Reference	75
5.27	ScoreSystem/i2cTestCode/I2C_reg.cpp File Reference	75
5.28	ScoreSystem/Screen-ButtonTestCode/I2C_reg.cpp File Reference	75
5.29	ScoreSystem/ScreenTestCode/I2C_reg.cpp File Reference	76
5.30	ScoreSystem/i2cTestCode/test.cpp File Reference	76
5.31	ScoreSystem/i2cTestCode/Timer.cpp File Reference	76
5.32	ScoreSystem/Screen-ButtonTestCode/Timer.cpp File Reference	76
5.33	ScoreSystem/ScreenTestCode/Timer.cpp File Reference	76
5.34	ScoreSystem/Timer.cpp File Reference	76
5.35	ScoreSystem/LCD.cpp File Reference	76
5.35.1	Macro Definition Documentation	77
5.35.1.1	BV	77
5.36	ScoreSystem/Screen-ButtonTestCode/LCD.cpp File Reference	77



5.36.1 Macro Definition Documentation . . . . .	77
5.36.1.1 BV . . . . .	77
5.37 ScoreSystem/ScreenTestCode/LCD.cpp File Reference . . . . .	77
5.37.1 Macro Definition Documentation . . . . .	78
5.37.1.1 BV . . . . .	78
5.38 ScoreSystem/Page.cpp File Reference . . . . .	78
5.39 ScoreSystem/ScoreSystemCtrl.cpp File Reference . . . . .	78
5.40 ScoreSystem/Screen-ButtonTestCode/target/files/Adafruit_MCP23008.d File Reference . . . . .	78
5.41 ScoreSystem/Screen-ButtonTestCode/target/files/I2C_reg.d File Reference . . . . .	78
5.42 ScoreSystem/Screen-ButtonTestCode/target/files/LCD.d File Reference . . . . .	78
5.43 ScoreSystem/Screen-ButtonTestCode/target/files/Timer.d File Reference . . . . .	78
5.44 ScoreSystem/WebsiteScoreHandling.cpp File Reference . . . . .	78
5.44.1 Function Documentation . . . . .	79
5.44.1.1 setNewIPJS() . . . . .	79
<b>Index</b>	<b>81</b>



# Chapter 1

## Hierarchical Index

### 1.1 Class Hierarchy

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

Adafruit_MCP23008 . . . . .	7
cursorCoord . . . . .	12
WebsiteScoreHandling::DATESTRUCT . . . . .	13
LCD . . . . .	22
Message	
buttonMessage . . . . .	11
psocUpdateMessage . . . . .	32
Page . . . . .	28
ThreadFunctor	
Button . . . . .	10
I2C_reg . . . . .	14
ScoreSystemCtrl . . . . .	32
Test . . . . .	39
Timer . . . . .	39
WebsiteScoreHandling::TIMESTRUCT . . . . .	41
WebsiteScoreHandling . . . . .	42



## Chapter 2

# Class Index

### 2.1 Class List

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

<a href="#">Adafruit_MCP23008</a> . . . . .	7
<a href="#">Button</a> . . . . .	10
<a href="#">buttonMessage</a> . . . . .	11
<a href="#">cursorCoord</a> . . . . .	12
<a href="#">WebsiteScoreHandling::DATESTRUCT</a> . . . . .	13
<a href="#">I2C_reg</a> . . . . .	14
<a href="#">LCD</a> . . . . .	22
<a href="#">Page</a> . . . . .	28
<a href="#">psocUpdateMessage</a> . . . . .	32
<a href="#">ScoreSystemCtrl</a> . . . . .	32
<a href="#">Test</a> . . . . .	39
<a href="#">Timer</a> . . . . .	39
<a href="#">WebsiteScoreHandling::TIMESTRUCT</a> . . . . .	41
<a href="#">WebsiteScoreHandling</a> Global function; Is called from ScoreSystem upon startup . . . . .	42



## Chapter 3

# File Index

### 3.1 File List

Here is a list of all files with brief descriptions:

inc/osapi/ScoreSystem/Adafruit_MCP23008.hpp	51
inc/osapi/ScoreSystem/Button.hpp	53
inc/osapi/ScoreSystem/I2C_reg.hpp	54
inc/osapi/ScoreSystem/LCD.hpp	55
inc/osapi/ScoreSystem/Page.hpp	60
inc/osapi/ScoreSystem/ScoreSystemCtrl.hpp	61
inc/osapi/ScoreSystem/test.hpp	62
inc/osapi/ScoreSystem/Timer.hpp	62
inc/osapi/ScoreSystem/WebsiteScoreHandling.h	62
ScoreSystem/Adafruit_MCP23008.cpp	63
ScoreSystem/Button.cpp	64
ScoreSystem/I2C_reg.cpp	75
ScoreSystem/LCD.cpp	76
ScoreSystem/main.cpp	73
ScoreSystem/Page.cpp	78
ScoreSystem/ScoreSystemCtrl.cpp	78
ScoreSystem/Timer.cpp	76
ScoreSystem/WebsiteScoreHandling.cpp	78
ScoreSystem/Button_Driver/button_drv.c	64
ScoreSystem/Button_Driver/button_drv.mod.c	71
ScoreSystem/buttonTestCode/main.cpp	72
ScoreSystem/buttonTestCode/host/files/main.d	72
ScoreSystem/buttonTestCode/target/files/main.d	72
ScoreSystem/i2cTestCode/I2C_reg.cpp	75
ScoreSystem/i2cTestCode/main.cpp	73
ScoreSystem/i2cTestCode/test.cpp	76
ScoreSystem/i2cTestCode/Timer.cpp	76
ScoreSystem/Screen-ButtonTestCode/Adafruit_MCP23008.cpp	64
ScoreSystem/Screen-ButtonTestCode/I2C_reg.cpp	75
ScoreSystem/Screen-ButtonTestCode/LCD.cpp	77
ScoreSystem/Screen-ButtonTestCode/main.cpp	74
ScoreSystem/Screen-ButtonTestCode/Timer.cpp	76
ScoreSystem/Screen-ButtonTestCode/host/files/main.d	72
ScoreSystem/Screen-ButtonTestCode/target/files/Adafruit_MCP23008.d	78
ScoreSystem/Screen-ButtonTestCode/target/files/I2C_reg.d	78

ScoreSystem/Screen-ButtonTestCode/target/files/LCD.d	78
ScoreSystem/Screen-ButtonTestCode/target/files/main.d	72
ScoreSystem/Screen-ButtonTestCode/target/files/Timer.d	78
ScoreSystem/ScreenTestCode/Adafruit_MCP23008.cpp	64
ScoreSystem/ScreenTestCode/I2C_reg.cpp	76
ScoreSystem/ScreenTestCode/LCD.cpp	77
ScoreSystem/ScreenTestCode/main.cpp	74
ScoreSystem/ScreenTestCode/Timer.cpp	76
ScoreSystem/TimeTestCode/main.cpp	75



## Chapter 4

# Class Documentation

### 4.1 Adafruit\_MCP23008 Class Reference

```
#include <Adafruit_MCP23008.hpp>
```

#### Public Member Functions

- void [begin](#) (uint8\_t addr)
- void [begin](#) (void)
- void [pinMode](#) (uint8\_t p, uint8\_t d)
- void [digitalWrite](#) (uint8\_t p, uint8\_t d)
- void [pullUp](#) (uint8\_t p, uint8\_t d)
- uint8\_t [digitalRead](#) (uint8\_t p)
- uint8\_t [readGPIO](#) (void)
- void [writeGPIO](#) (uint8\_t)

#### Private Member Functions

- uint8\_t [read8](#) (uint8\_t addr)
- void [write8](#) (uint8\_t addr, uint8\_t data)

#### Private Attributes

- int [fd](#)
- uint8\_t [i2caddr](#)

#### 4.1.1 Member Function Documentation

**4.1.1.1 begin()** [1/2]

```
void Adafruit_MCP23008::begin (
    uint8_t addr )
```

**4.1.1.2 begin()** [2/2]

```
void Adafruit_MCP23008::begin (
    void )
```

**4.1.1.3 digitalRead()**

```
uint8_t Adafruit_MCP23008::digitalRead (
    uint8_t p )
```

**4.1.1.4 digitalWrite()**

```
void Adafruit_MCP23008::digitalWrite (
    uint8_t p,
    uint8_t d )
```

**4.1.1.5 pinMode()**

```
void Adafruit_MCP23008::pinMode (
    uint8_t p,
    uint8_t d )
```

**4.1.1.6 pullUp()**

```
void Adafruit_MCP23008::pullUp (
    uint8_t p,
    uint8_t d )
```

#### 4.1.1.7 read8()

```
uint8_t Adafruit_MCP23008::read8 (
    uint8_t addr ) [private]
```

#### 4.1.1.8 readGPIO()

```
uint8_t Adafruit_MCP23008::readGPIO (
    void )
```

#### 4.1.1.9 write8()

```
void Adafruit_MCP23008::write8 (
    uint8_t addr,
    uint8_t data ) [private]
```

#### 4.1.1.10 writeGPIO()

```
void Adafruit_MCP23008::writeGPIO (
    uint8_t gpio )
```

### 4.1.2 Member Data Documentation

#### 4.1.2.1 fd

```
int Adafruit_MCP23008::fd [private]
```

#### 4.1.2.2 i2caddr

```
uint8_t Adafruit_MCP23008::i2caddr [private]
```

The documentation for this class was generated from the following files:

- inc/osapi/ScoreSystem/[Adafruit\\_MCP23008.hpp](#)
- ScoreSystem/[Adafruit\\_MCP23008.cpp](#)

## 4.2 Button Class Reference

```
#include <Button.hpp>
```

Inherits ThreadFunctor.

### Public Member Functions

- [Button](#) (osapi::MsgQueue \*mq)  
*constructor. Opens the [Button](#) node in /dev and sets the message queue*
- [~Button](#) ()

### Private Member Functions

- void [run](#) ()

### Private Attributes

- bool [running](#) = false
- int [fd](#)
- char [value](#) [2]
- osapi::MsgQueue \* [mq\\_](#)

### 4.2.1 Constructor & Destructor Documentation

#### 4.2.1.1 Button()

```
Button::Button (  
    osapi::MsgQueue * mq )
```

constructor. Opens the [Button](#) node in /dev and sets the message queue

#### 4.2.1.2 ~Button()

```
Button::~~Button ( )
```

### 4.2.2 Member Function Documentation

#### 4.2.2.1 run()

```
void Button::run ( ) [private]
```

Tries to read a value in from the button node every 25ms. The read is blocking in the driver so it will only return once there's new data available, and once there is, it returns to the function and it's able to send that data to the main thread via the message queue

### 4.2.3 Member Data Documentation

#### 4.2.3.1 fd

```
int Button::fd [private]
```

#### 4.2.3.2 mq\_

```
osapi::MsgQueue* Button::mq_ [private]
```

#### 4.2.3.3 running

```
bool Button::running = false [private]
```

#### 4.2.3.4 value

```
char Button::value[2] [private]
```

The documentation for this class was generated from the following files:

- [inc/osapi/ScoreSystem/Button.hpp](#)
- [ScoreSystem/Button.cpp](#)
- [ScoreSystem/Page.cpp](#)

## 4.3 buttonMessage Struct Reference

```
#include <Button.hpp>
```

Inherits Message.

## Public Member Functions

- [buttonMessage](#) (uint8\_t b)

## Public Attributes

- [uint8\\_t x](#)

### 4.3.1 Constructor & Destructor Documentation

#### 4.3.1.1 buttonMessage()

```
buttonMessage::buttonMessage (
    uint8_t b ) [inline]
```

### 4.3.2 Member Data Documentation

#### 4.3.2.1 x

```
uint8_t buttonMessage::x
```

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

- inc/osapi/ScoreSystem/[Button.hpp](#)

## 4.4 cursorCoord Struct Reference

```
#include <Page.hpp>
```

### Public Attributes

- [uint8\\_t line\\_](#)
  - [uint8\\_t row\\_](#)
  - [uint8\\_t dir\\_](#)
  - [uint8\\_t type\\_](#)
- 0 for right-array, 1 for left-arrow, 2 for cursor, 3 for noCursor*

#### 4.4.1 Detailed Description

These are written in the .txt file we load in. they explain what cursor is being used, and it's functionality

## 4.4.2 Member Data Documentation

### 4.4.2.1 dir\_

`uint8_t cursorCoord::dir_`

### 4.4.2.2 line\_

`uint8_t cursorCoord::line_`

### 4.4.2.3 row\_

`uint8_t cursorCoord::row_`

### 4.4.2.4 type\_

`uint8_t cursorCoord::type_`

0 for right-arrow, 1 for left-arrow, 2 for cursor, 3 for noCursor

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

- [inc/osapi/ScoreSystem/Page.hpp](#)

## 4.5 WebsiteScoreHandling::DATESTRUCT Struct Reference

### Public Attributes

- `int day_ = 0`
- `int month_ = 0`
- `int year_ = 0`

## 4.5.1 Member Data Documentation

#### 4.5.1.1 day\_

```
int WebsiteScoreHandling::DATESTRUCT::day_ = 0
```

#### 4.5.1.2 month\_

```
int WebsiteScoreHandling::DATESTRUCT::month_ = 0
```

#### 4.5.1.3 year\_

```
int WebsiteScoreHandling::DATESTRUCT::year_ = 0
```

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

- inc/osapi/ScoreSystem/[WebsiteScoreHandling.h](#)

## 4.6 I2C\_reg Class Reference

```
#include <I2C_reg.hpp>
```

Inherits ThreadFunctor.

### Public Member Functions

- void [testStatic](#) ()
- void [setMsgQueueScoreSystem](#) (osapi::MsgQueue \*smq)
- osapi::MsgQueue \* [getMsgQueue](#) ()
- void [setUp](#) (int cups)  
*Function which starts the timers to get updates from the psocs, and sets nr. of cups on the psocs.*
- void [setArduinoMessage](#) (uint8\_t message)  
*Function to send a message to the arduino.*
- void [setPsocMessage](#) (uint8\_t message, uint8\_t psoc=0)  
*Function to send a message to either both psocs, by leaving it blank, or a specific psoc by writing 1 or 2.*
- void [displayWrite](#) (uint8\_t addr, uint8\_t data)  
*Function to be used by the [LCD](#) to write to the display.*
- uint8\_t [displayRead](#) (uint8\_t addr)  
*Function to be used by the [LCD](#) to read from display registers.*
- void [displayInit](#) ()  
*Function to be used by the [LCD](#) to initialize the display.*
- void [lockI2C](#) ()  
*Function to lock the mutex. Used by the [LCD](#) class.*
- void [unlockI2C](#) ()  
*Function to unlock the mutex. Used by the [LCD](#) class.*
- void [stopPsocPolling](#) ()  
*Function that disables the timer polling the psocs for updates.*



## Static Public Member Functions

- static [I2C\\_reg](#) & [getInstance](#) ()

## Private Member Functions

- [I2C\\_reg](#) ()  
*Constructor. Opens /dev/i2c-1. Creates the Message queue.*
- [~I2C\\_reg](#) ()  
*Constructor. Closes /dev/i2c-1. Deletes the Message queue.*
- void [run](#) ()  
*ThreadFuncor function. Thread function.*
- void [handleMsg](#) (unsigned long id, [osapi::Message](#) \*msg)  
*Function to handle messages received in the message queue.*
- void [psocUpdate](#) ()  
*Private function which gets updates from the psocs.*
- void [sendPsocBroadcast](#) ()  
*Function which sends messages to both psocs. private.*
- void [sendPsocMessage](#) (uint8\_t psocNr)  
*Function which sends messages to a specific psoc. private.*
- void [sendPsoc1Message](#) ()  
*Function which sends a message to psoc 1. private.*
- void [sendPsoc2Message](#) ()  
*Function which sends a message to psoc 2. private.*
- void [sendArduinoMessage](#) ()  
*Function which sends a message to the arduino. private.*

## Private Attributes

- int [fd\\_](#)
- unsigned int [state\\_](#)
- bool [running\\_](#) = true
- uint8\_t [currentAddr\\_](#) = 0
- uint8\_t [arduinoAdress\\_](#) = 0x06
- uint8\_t [psoc1Adress\\_](#) = 0x10
- uint8\_t [psoc2Adress\\_](#) = 0x11
- uint8\_t [screenAdress\\_](#) = 0x20
- uint8\_t [arduinoMessage\\_](#) = 0
- uint8\_t [psocMessage\\_](#) = 0
- uint8\_t [receivingPsoC\\_](#) = 0
- [Timer](#) \* [timer1\\_](#)
- [osapi::Thread](#) \* [tt\\_](#)
- [osapi::MsgQueue](#) \* [mq\\_](#)
- [osapi::MsgQueue](#) \* [ScoreSystemMQ\\_](#)
- [osapi::MsgQueue](#) \* [displayMQ\\_](#)
- [osapi::Mutex](#) [mut\\_](#)
- [osapi::Conditional](#) [cond\\_](#)

### 4.6.1 Constructor & Destructor Documentation

#### 4.6.1.1 I2C\_reg()

```
I2C_reg::I2C_reg ( ) [private]
```

Constructor. Opens /dev/i2c-1. Creates the Message queue.

#### 4.6.1.2 ~I2C\_reg()

```
I2C_reg::~I2C_reg ( ) [private]
```

Constructor. Closes /dev/i2c-1. Deletes the Message queue.

### 4.6.2 Member Function Documentation

#### 4.6.2.1 displayInit()

```
void I2C_reg::displayInit ( )
```

Function to be used by the [LCD](#) to initialize the display.

#### 4.6.2.2 displayRead()

```
uint8_t I2C_reg::displayRead (
    uint8_t addr )
```

Function to be used by the [LCD](#) to read from display registers.

#### 4.6.2.3 displayWrite()

```
void I2C_reg::displayWrite (
    uint8_t addr,
    uint8_t data )
```

Function to be used by the [LCD](#) to write to the display.

#### 4.6.2.4 getInstance()

```
static I2C_reg& I2C_reg::getInstance ( ) [inline], [static]
```

#### 4.6.2.5 getMsgQueue()

```
osapi::MsgQueue* I2C_reg::getMsgQueue ( ) [inline]
```

#### 4.6.2.6 handleMsg()

```
void I2C_reg::handleMsg (
    unsigned long id,
    osapi::Message * msg ) [private]
```

Function to handle messages received in the message queue.

#### 4.6.2.7 lockI2C()

```
void I2C_reg::lockI2C ( )
```

Function to lock the mutex. Used by the [LCD](#) class.

#### 4.6.2.8 psocUpdate()

```
void I2C_reg::psocUpdate ( ) [private]
```

Private function which gets updates from the psocs.

#### 4.6.2.9 run()

```
void I2C_reg::run ( ) [private]
```

ThreadFunctor function. Thread function.

#### 4.6.2.10 sendArduinoMessage()

```
void I2C_reg::sendArduinoMessage ( ) [private]
```

Function which sends a message to the arduino. private.

#### 4.6.2.11 sendPsoc1Message()

```
void I2C_reg::sendPsoc1Message ( ) [private]
```

Function which sends a message to psoc 1. private.

#### 4.6.2.12 sendPsoc2Message()

```
void I2C_reg::sendPsoc2Message ( ) [private]
```

Function which sends a message to psoc 2. private.

#### 4.6.2.13 sendPsocBroadcast()

```
void I2C_reg::sendPsocBroadcast ( ) [private]
```

Function which sends messages to both psocs. private.

#### 4.6.2.14 sendPsocMessage()

```
void I2C_reg::sendPsocMessage (
    uint8_t psocNr ) [private]
```

Function which sends messages to a specific psoc. private.

#### 4.6.2.15 setArduinoMessage()

```
void I2C_reg::setArduinoMessage (
    uint8_t message )
```

Function to send a message to the arduino.

#### 4.6.2.16 setMsgQueueScoreSystem()

```
void I2C_reg::setMsgQueueScoreSystem (
    osapi::MsgQueue * smq ) [inline]
```

#### 4.6.2.17 setPsocMessage()

```
void I2C_reg::setPsocMessage (
    uint8_t message,
    uint8_t psoc = 0 )
```

Function to send a message to either both psocs, by leaving it blank, or a specific psoc by writing 1 or 2.

#### 4.6.2.18 setUP()

```
void I2C_reg::setUP (
    int cups )
```

Function which starts the timers to get updates from the psocs, and sets nr. of cups on the psocs.

#### 4.6.2.19 stopPsocPolling()

```
void I2C_reg::stopPsocPolling ( )
```

Function that disables the timer polling the psocs for updates.

#### 4.6.2.20 testStatic()

```
void I2C_reg::testStatic ( )
```

#### 4.6.2.21 unlockI2C()

```
void I2C_reg::unlockI2C ( )
```

Function to unlock the mutex. Used by the [LCD](#) class.

### 4.6.3 Member Data Documentation

#### 4.6.3.1 `arduinoAdress_`

```
uint8_t I2C_reg::arduinoAdress_ = 0x06 [private]
```

#### 4.6.3.2 `arduinoMessage_`

```
uint8_t I2C_reg::arduinoMessage_ = 0 [private]
```

#### 4.6.3.3 `cond_`

```
osapi::Conditional I2C_reg::cond_ [private]
```

#### 4.6.3.4 `currentAddr_`

```
uint8_t I2C_reg::currentAddr_ = 0 [private]
```

#### 4.6.3.5 `displayMQ_`

```
osapi::MsgQueue* I2C_reg::displayMQ_ [private]
```

#### 4.6.3.6 `fd_`

```
int I2C_reg::fd_ [private]
```

#### 4.6.3.7 `mq_`

```
osapi::MsgQueue* I2C_reg::mq_ [private]
```

#### 4.6.3.8 mut\_

```
osapi::Mutex I2C_reg::mut_ [private]
```

#### 4.6.3.9 psoc1Adress\_

```
uint8_t I2C_reg::psoc1Adress_ = 0x10 [private]
```

#### 4.6.3.10 psoc2Adress\_

```
uint8_t I2C_reg::psoc2Adress_ = 0x11 [private]
```

#### 4.6.3.11 psocMessage\_

```
uint8_t I2C_reg::psocMessage_ = 0 [private]
```

#### 4.6.3.12 receivingPsoC\_

```
uint8_t I2C_reg::receivingPsoC_ = 0 [private]
```

#### 4.6.3.13 running\_

```
bool I2C_reg::running_ = true [private]
```

#### 4.6.3.14 ScoreSystemMQ\_

```
osapi::MsgQueue* I2C_reg::ScoreSystemMQ_ [private]
```

#### 4.6.3.15 screenAdress\_

```
uint8_t I2C_reg::screenAdress_ = 0x20 [private]
```

#### 4.6.3.16 state\_

```
unsigned int I2C_reg::state_ [private]
```

#### 4.6.3.17 timer1\_

```
Timer* I2C_reg::timer1_ [private]
```

#### 4.6.3.18 tt\_

```
osapi::Thread* I2C_reg::tt_ [private]
```

The documentation for this class was generated from the following files:

- [inc/osapi/ScoreSystem/I2C\\_reg.hpp](#)
- [ScoreSystem/I2C\\_reg.cpp](#)

## 4.7 LCD Class Reference

```
#include <LCD.hpp>
```

### Public Member Functions

- [LCD](#) ()  
*Default constructor, sets up registers.*
- void [begin](#) ()  
*Sets up the display to the correct modes according to the datasheet for the HD44780.*
- void [lcdWrite\\_four\\_bits](#) (uint8\_t [command](#))  
*Writing a four bit value to the screen. Making several I2C requests meanwhile.*
- void [command](#) (uint8\_t [value](#))  
*send a command to the screen. Used to set blink, cursor etc.*
- void [setCursor](#) (uint8\_t col, uint8\_t row)  
*Set cursor position.*
- void [cursor](#) ()  
*enables the buttom cursor*
- void [noCursor](#) ()  
*disables the buttom cursor*
- void [display](#) ()  
*turn display on*
- void [noDisplay](#) ()  
*turn display off*
- void [blink](#) ()  
*starts blinking at the current position*



- void `noBlink` ()  
*stops the current position from blinking*
- void `clear` ()  
*clears the display*
- void `home` ()  
*returns cursor to the home position*
- void `send` (uint8\_t `value`, uint8\_t `mode`)  
*Wrapper function, so that it can be passed an 8bit value and send it using 2x write\_four\_bits.*
- uint8\_t `readReg` ()  
*read the current GPIO values.*
- void `stringWrite` (string `str`)  
*Writes a string to the screen.*
- void `charWrite` (uint8\_t `value`)  
*Writes a char on the screen.*

### Private Attributes

- uint8\_t `En` = 0b00000100
- uint8\_t `Rw` = 0b00000010
- uint8\_t `Rs` = 0b00000001
- uint8\_t `_rs_pin` = 1
- uint8\_t `_rw_pin` = 255
- uint8\_t `_enable_pin` = 2
- uint8\_t `_data_pins` [4]
- uint8\_t `displayFunction` = 0x00
- uint8\_t `displayControl` = 0x00
- uint8\_t `displayMode` = 0x00
- uint8\_t `numLines` = 0
- uint8\_t `currentLine` = 0
- `Adafruit_MCP23008 i2c_`

## 4.7.1 Constructor & Destructor Documentation

### 4.7.1.1 LCD()

```
LCD::LCD ( )
```

Default constructor, sets up registers.

## 4.7.2 Member Function Documentation

#### 4.7.2.1 begin()

```
void LCD::begin (
    void )
```

Sets up the display to the correct modes according to the datasheet for the HD44780.

#### 4.7.2.2 blink()

```
void LCD::blink ( )
```

starts blinking at the current position

#### 4.7.2.3 charWrite()

```
void LCD::charWrite (
    uint8_t value )
```

Writes a char on the screen.

#### 4.7.2.4 clear()

```
void LCD::clear ( )
```

clears the display

#### 4.7.2.5 command()

```
void LCD::command (
    uint8_t value )
```

send a command to the screen. Used to set blink, cursor etc.

#### 4.7.2.6 cursor()

```
void LCD::cursor ( )
```

enables the buttom cursor

#### 4.7.2.7 display()

```
void LCD::display ( )
```

turn display on

#### 4.7.2.8 home()

```
void LCD::home ( )
```

returns cursor to the home position

#### 4.7.2.9 lcdWrite\_four\_bits()

```
void LCD::lcdWrite_four_bits (
    uint8_t command )
```

Writing a four bit value to the screen. Making several I2C requests meanwhile.

#### 4.7.2.10 noBlink()

```
void LCD::noBlink ( )
```

stops the current position from blinking

#### 4.7.2.11 noCursor()

```
void LCD::noCursor ( )
```

disables the bottom cursor

#### 4.7.2.12 noDisplay()

```
void LCD::noDisplay ( )
```

turn display off

#### 4.7.2.13 readReg()

```
uint8_t LCD::readReg ( )
```

read the current GPIO values.

#### 4.7.2.14 send()

```
void LCD::send (
    uint8_t value,
    uint8_t mode )
```

Wrapper function, so that it can be passed an 8bit value and send it using 2x write\_four\_bits.

#### 4.7.2.15 setCursor()

```
void LCD::setCursor (
    uint8_t col,
    uint8_t row )
```

Set cursor position.

#### 4.7.2.16 stringWrite()

```
void LCD::stringWrite (
    string str )
```

Writes a string to the screen.

### 4.7.3 Member Data Documentation

#### 4.7.3.1 \_data\_pins

```
uint8_t LCD::_data_pins[4] [private]
```

#### 4.7.3.2 \_enable\_pin

```
uint8_t LCD::_enable_pin = 2 [private]
```

#### 4.7.3.3 \_rs\_pin

```
uint8_t LCD::_rs_pin = 1 [private]
```

#### 4.7.3.4 \_rw\_pin

```
uint8_t LCD::_rw_pin = 255 [private]
```

#### 4.7.3.5 currentLine

```
uint8_t LCD::currentLine = 0 [private]
```

#### 4.7.3.6 displayControl

```
uint8_t LCD::displayControl = 0x00 [private]
```

#### 4.7.3.7 displayFunction

```
uint8_t LCD::displayFunction = 0x00 [private]
```

#### 4.7.3.8 displayMode

```
uint8_t LCD::displayMode = 0x00 [private]
```

#### 4.7.3.9 En

```
uint8_t LCD::En = 0b00000100 [private]
```

#### 4.7.3.10 i2c\_

```
Adafruit_MCP23008 LCD::i2c_ [private]
```

#### 4.7.3.11 numLines

```
uint8_t LCD::numLines = 0 [private]
```

#### 4.7.3.12 Rs

```
uint8_t LCD::Rs = 0b00000001 [private]
```

#### 4.7.3.13 Rw

```
uint8_t LCD::Rw = 0b00000010 [private]
```

The documentation for this class was generated from the following files:

- inc/osapi/ScoreSystem/[LCD.hpp](#)
- ScoreSystem/[LCD.cpp](#)

## 4.8 Page Class Reference

```
#include <Page.hpp>
```

### Public Member Functions

- [Page](#) (std::string filename)
- void [buttonPressed](#) ([LCD](#) \*display, string &returnString, unsigned int &state\_)
- void [buttonRight](#) ([LCD](#) \*display)
- void [buttonLeft](#) ([LCD](#) \*display)
- void [displayScreen](#) ([LCD](#) \*display, string \*name1=nullptr, string \*name2=nullptr)
- void [resetPage](#) ()

*Resets the page. Cursorposition and teamname is reset.*

## Private Attributes

- `std::vector< std::string > pageText`
- `int8_t cursorPos_ = 0`  
*Page Text to be displayed.*
- `uint8_t nrCursorPos_`  
*Current cursor position.*
- `char teamNameArr [16]`
- `bool teamEnter_ = false`  
*Used for team naming schemes.*
- `bool selectingChar = false`  
*Used for team naming schemes.*
- `char currentChar = 'a'`  
*Used for team naming schemes.*
- `std::vector< struct cursorCoord > possibleCursorPos`  
*Used for team naming schemes.*

## 4.8.1 Constructor & Destructor Documentation

### 4.8.1.1 Page()

```
Page::Page (
    std::string filename ) [inline]
```

Constructor for [Page](#). It opens up the specified file, and reads it in. File needs to be a specific style, where the cursorcoords is in the first line, comma seperated with a colon indicating the end.

## 4.8.2 Member Function Documentation

### 4.8.2.1 buttonLeft()

```
void Page::buttonLeft (
    LCD * display ) [inline]
```

Handles when the button is rotated left. Here it checks if a char is being selected, since it either needs to rotate the cursor or change the char appropriately

### 4.8.2.2 buttonPressed()

```
void Page::buttonPressed (
    LCD * display,
    string & returnString,
    unsigned int & state_ ) [inline]
```

Checks the current button position and checks what action it's supposed to do, according to the type. It needs to check if we're scrolling through chars, since it then needs to deselect it. Updated after accepttest, since a stray line of code was commented out, enabling the possibility of reaching unwanted places when deleting

#### 4.8.2.3 buttonRight()

```
void Page::buttonRight (
    LCD * display ) [inline]
```

Handles when the button is rotated right. Here it checks if a char is being selected, since it either needs to rotate the cursor or change the char appropriately

#### 4.8.2.4 displayScreen()

```
void Page::displayScreen (
    LCD * display,
    string * name1 = nullptr,
    string * name2 = nullptr ) [inline]
```

Displays the screen. Scrolls through the pageText and writes it to the screen. Takes 2 string arguments, which is either both team names, or the IP adress, and displays them accordingly. Remember to check if the page has space for the strings before you pass the strings.

#### 4.8.2.5 resetPage()

```
void Page::resetPage ( ) [inline]
```

Resets the page. Cursorposition and teamname is reset.

### 4.8.3 Member Data Documentation

#### 4.8.3.1 currentChar

```
char Page::currentChar = 'a' [private]
```

Used for team naming schemes.

#### 4.8.3.2 cursorPos\_

```
int8_t Page::cursorPos_ = 0 [private]
```

[Page](#) Text to be displayed.



#### 4.8.3.3 nrCursorPos\_

```
uint8_t Page::nrCursorPos_ [private]
```

Current cursor position.

#### 4.8.3.4 pageText

```
std::vector<std::string> Page::pageText [private]
```

#### 4.8.3.5 possibleCursorPos

```
std::vector<struct cursorCoord> Page::possibleCursorPos [private]
```

Used for team naming schemes.

#### 4.8.3.6 selectingChar

```
bool Page::selectingChar = false [private]
```

Used for team naming schemes.

#### 4.8.3.7 teamEnter\_

```
bool Page::teamEnter_ = false [private]
```

Used for team naming schemes.

#### 4.8.3.8 teamNameArr

```
char Page::teamNameArr[16] [private]
```

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

- [inc/osapi/ScoreSystem/Page.hpp](#)

## 4.9 psocUpdateMessage Struct Reference

```
#include <I2C_reg.hpp>
```

Inherits Message.

### Public Member Functions

- [psocUpdateMessage](#) (uint8\_t val)

### Public Attributes

- uint8\_t [val\\_](#)

### 4.9.1 Constructor & Destructor Documentation

#### 4.9.1.1 psocUpdateMessage()

```
psocUpdateMessage::psocUpdateMessage (
    uint8_t val ) [inline]
```

### 4.9.2 Member Data Documentation

#### 4.9.2.1 val\_

```
uint8_t psocUpdateMessage::val_
```

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

- [inc/osapi/ScoreSystem/I2C\\_reg.hpp](#)

## 4.10 ScoreSystemCtrl Class Reference

```
#include <ScoreSystemCtrl.hpp>
```

Inherits ThreadFunctor.

## Public Member Functions

- [ScoreSystemCtrl](#) ()
- [osapi::MsgQueue \\* getMsgQueue](#) ()

## Private Member Functions

- void [run](#) ()
- void [handleMsg](#) (unsigned int id, [osapi::Message \\*msgPtr](#))
- void [handleState](#) ()
- void [handlePsocUpdate](#) (uint8\_t psoc, [osapi::Message \\*msgPtr](#))
- void [resetGame](#) ()
- void [endGame](#) (int winner=1)
- string [getIP](#) ()

## Private Attributes

- string [ip\\_](#)
- string [tempString\\_](#)
- unsigned int [state\\_](#) = [pageEvent::noUpdate](#)
- uint8\_t [currentScreen](#) = 0
- unsigned long [gameTime\\_](#) = 0
- uint8\_t [nr\\_Cups\\_In\\_Game\\_](#) = 10
- string [teamName1\\_](#) = "moon moon"
- string [teamName2\\_](#) = "red pandas"
- *DEFAULT VALUES.*
- enum [pSocMessages zone1State\\_](#) = [pSocMessages::NO\\_CHANGE](#)
- *DEFAULT VALUES.*
- enum [pSocMessages zone2State\\_](#) = [pSocMessages::NO\\_CHANGE](#)
- *DEFAULT VALUES.*
- uint8\_t [score\\_Team\\_1\\_](#) = 0
- *DEFAULT VALUES.*
- uint8\_t [score\\_Team\\_2\\_](#) = 0
- uint8\_t [collectiveDoubleShots\\_](#) = 0
- bool [reArranging\\_](#) = false
- uint8\_t [reArrangingZone\\_](#) = 0
- vector< [Page](#) \* > [pages\\_](#)
- [LCD](#) \* [display](#)
- [Button](#) \* [btn](#)
- [WebsiteScoreHandling](#) \* [websitePtr\\_](#)
- [osapi::Thread](#) \* [i2cThread](#)
- [osapi::Thread](#) \* [buttonThread](#)
- [osapi::MsgQueue](#) \* [mq\\_](#)

### 4.10.1 Constructor & Destructor Documentation

#### 4.10.1.1 ScoreSystemCtrl()

```
ScoreSystemCtrl::ScoreSystemCtrl ( )
```

----- DEFAULT CONSTRUCTOR ----- SETS UP VARIABLES, LOADS IN PAGES ----- STARTING I2C AND BUTTON THREADS AND STARTS THE DISPLAY -

### 4.10.2 Member Function Documentation

#### 4.10.2.1 endGame()

```
void ScoreSystemCtrl::endGame (
    int winner = 1 ) [private]
```

----- END GAME ----- ENDS GAME, UPLOADS TO WEBSITE AND RESETS TO INITIAL STATE --- Waits 15 seconds before returning to the welcome screen

#### 4.10.2.2 getIP()

```
string ScoreSystemCtrl::getIP ( ) [private]
```

----- GET IP ----- LOADS IN IP FROM FILE ---

#### 4.10.2.3 getMsgQueue()

```
osapi::MsgQueue * ScoreSystemCtrl::getMsgQueue ( )
```

----- GET MESSAGEQUEUE ----- RETURNS POINTER TO MESSAGEQUEUE ---

#### 4.10.2.4 handleMsg()

```
void ScoreSystemCtrl::handleMsg (
    unsigned int id,
    osapi::Message * msgPtr ) [private]
```

----- HANDLEMSG ----- HANDLES MESSAGES IN OUR MESSAGEQUEUE --- Handles the received message. If the button is pressed, then page->buttonPressed is called. Same happens with rotateLeft & rotateRight. It also handles receiving updates from the 2 psocs, where it calls handlePsocUpdate

## 4.10.2.5 handlePsocUpdate()

```
void ScoreSystemCtrl::handlePsocUpdate (
    uint8_t psoc,
    osapi::Message * msgPtr ) [private]
```

----- HANDLE PSOC UPDATE ----- — HANDLES THE VALUES RETURNED FROM THE PSOC —  
 Handles the updates received from the psocs, by casting the Message pointer to a [psocUpdateMessage](#), where an uint8\_t value can be retrieved from. It will then discard the update, if its received during rearrange, which means, that no team can finish, while 1 is rearranging. It will also ready the game, when initially setting up the cups, making it possible to press play game.

Otherwise it checks if the current state is of less value than the newly received state, which will cause a change of state. This happens until a cupZoneReady state is received.

The teams scores are also updated in this function.

## 4.10.2.6 handleState()

```
void ScoreSystemCtrl::handleState ( ) [private]
```

----- HANDLE STATE ----- — HANDLES STATES RETURNED BY OUR PAGES — Handles the state received from the buttonPressed function. It then takes action according to the state received. Most commonly it increases the currentScreen variable and displays the next screen. It will also save teamNames and send information to the PsoC's/Arduino

CONNECTING TO INTERNET, NEEDS TO WAIT FOR CONNMANCTL TO START, HENCE THE DELAY

## 4.10.2.7 resetGame()

```
void ScoreSystemCtrl::resetGame ( ) [private]
```

----- RESET GAME ----- — RESETS THE TABLE TO INITIAL STATE — Resets pages, psocs, Arduino, names, sates etc. to initial values.

## 4.10.2.8 run()

```
void ScoreSystemCtrl::run ( ) [private]
```

— RUN — — THREAD METHOD — Gets message from the Message Queue, and handles the received message

## 4.10.3 Member Data Documentation

## 4.10.3.1 btn

```
Button* ScoreSystemCtrl::btn [private]
```

#### 4.10.3.2 buttonThread

```
osapi::Thread* ScoreSystemCtrl::buttonThread [private]
```

#### 4.10.3.3 collectiveDoubleShots\_

```
uint8_t ScoreSystemCtrl::collectiveDoubleShots_ = 0 [private]
```

#### 4.10.3.4 currentScreen

```
uint8_t ScoreSystemCtrl::currentScreen = 0 [private]
```

#### 4.10.3.5 display

```
LCD* ScoreSystemCtrl::display [private]
```

#### 4.10.3.6 gameTime\_

```
unsigned long ScoreSystemCtrl::gameTime_ = 0 [private]
```

#### 4.10.3.7 i2cThread

```
osapi::Thread* ScoreSystemCtrl::i2cThread [private]
```

#### 4.10.3.8 ip\_

```
string ScoreSystemCtrl::ip_ [private]
```

#### 4.10.3.9 mq\_

```
osapi::MsgQueue* ScoreSystemCtrl::mq_ [private]
```

#### 4.10.3.10 nr\_Cups\_In\_Game\_

```
uint8_t ScoreSystemCtrl::nr_Cups_In_Game_ = 10 [private]
```

#### 4.10.3.11 pages\_

```
vector<Page*> ScoreSystemCtrl::pages_ [private]
```

#### 4.10.3.12 reArranging\_

```
bool ScoreSystemCtrl::reArranging_ = false [private]
```

#### 4.10.3.13 reArrangingZone\_

```
uint8_t ScoreSystemCtrl::reArrangingZone_ = 0 [private]
```

#### 4.10.3.14 score\_Team\_1\_

```
uint8_t ScoreSystemCtrl::score_Team_1_ = 0 [private]
```

DEFAULT VALUES.

#### 4.10.3.15 score\_Team\_2\_

```
uint8_t ScoreSystemCtrl::score_Team_2_ = 0 [private]
```

#### 4.10.3.16 state\_

```
unsigned int ScoreSystemCtrl::state_ = pageEvent::noUpdate [private]
```

#### 4.10.3.17 teamName1\_

```
string ScoreSystemCtrl::teamName1_ = "moon moon" [private]
```

#### 4.10.3.18 teamName2\_

```
string ScoreSystemCtrl::teamName2_ = "red pandas" [private]
```

DEFAULT VALUES.

#### 4.10.3.19 tempString\_

```
string ScoreSystemCtrl::tempString_ [private]
```

#### 4.10.3.20 websitePtr\_

```
WebsiteScoreHandling* ScoreSystemCtrl::websitePtr_ [private]
```

#### 4.10.3.21 zone1State\_

```
enum pSocMessages ScoreSystemCtrl::zone1State_ = pSocMessages::NO\_CHANGE [private]
```

DEFAULT VALUES.

#### 4.10.3.22 zone2State\_

```
enum pSocMessages ScoreSystemCtrl::zone2State_ = pSocMessages::NO\_CHANGE [private]
```

DEFAULT VALUES.

The documentation for this class was generated from the following files:

- [inc/osapi/ScoreSystem/ScoreSystemCtrl.hpp](#)
- [ScoreSystem/ScoreSystemCtrl.cpp](#)



## 4.11 Test Class Reference

```
#include <test.hpp>
```

Inherits ThreadFunctor.

### Private Member Functions

- void [run](#) ()

### 4.11.1 Member Function Documentation

#### 4.11.1.1 [run\(\)](#)

```
void Test::run ( ) [private]
```

The documentation for this class was generated from the following files:

- inc/osapi/ScoreSystem/[test.hpp](#)
- ScoreSystem/i2cTestCode/[test.cpp](#)

## 4.12 Timer Class Reference

```
#include <Timer.hpp>
```

Inherits ThreadFunctor.

### Public Member Functions

- [Timer](#) (unsigned long timeout, unsigned long id, osapi::MsgQueue \*mq)
- virtual [~Timer](#) ()
- void [stopTimer](#) ()

*Stops the timer, so that it can be joined and deleted.*

### Private Member Functions

- void [run](#) ()

*ThreadFunctor function, Sends a message every X ms.*

## Private Attributes

- `osapi::MsgQueue * mq_ = nullptr`
- `unsigned long id_ = 0`
- `unsigned long timeout_ = 0`
- `bool running_ = true`

## 4.12.1 Constructor & Destructor Documentation

### 4.12.1.1 Timer()

```
Timer::Timer (
    unsigned long timeout,
    unsigned long id,
    osapi::MsgQueue * mq )
```

Normal constructor. Sets the time of which it is to overflow, and what id it's supposed to pass along. it also takes what message queue to place the message in.

### 4.12.1.2 ~Timer()

```
Timer::~~Timer ( ) [virtual]
```

## 4.12.2 Member Function Documentation

### 4.12.2.1 run()

```
void Timer::run ( ) [private]
```

ThreadFunctor function, Sends a message every X ms.

### 4.12.2.2 stopTimer()

```
void Timer::stopTimer ( )
```

Stops the timer, so that it can be joined and deleted.

## 4.12.3 Member Data Documentation

#### 4.12.3.1 id\_

```
unsigned long Timer::id_ = 0 [private]
```

#### 4.12.3.2 mq\_

```
osapi::MsgQueue* Timer::mq_ = nullptr [private]
```

#### 4.12.3.3 running\_

```
bool Timer::running_ = true [private]
```

#### 4.12.3.4 timeout\_

```
unsigned long Timer::timeout_ = 0 [private]
```

The documentation for this class was generated from the following files:

- inc/osapi/ScoreSystem/[Timer.hpp](#)
- ScoreSystem/i2cTestCode/[Timer.cpp](#)

## 4.13 WebsiteScoreHandling::TIMESTRUCT Struct Reference

### Public Attributes

- int [seconds\\_](#) = 0
- int [minutes\\_](#) = 0
- int [hours\\_](#) = 0

### 4.13.1 Member Data Documentation

#### 4.13.1.1 hours\_

```
int WebsiteScoreHandling::TIMESTRUCT::hours_ = 0
```

#### 4.13.1.2 minutes\_

```
int WebsiteScoreHandling::Timestruct::minutes_ = 0
```

#### 4.13.1.3 seconds\_

```
int WebsiteScoreHandling::Timestruct::seconds_ = 0
```

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

- inc/osapi/ScoreSystem/[WebsiteScoreHandling.h](#)

## 4.14 WebsiteScoreHandling Class Reference

Global function; Is called from ScoreSystem upon startup.

```
#include <WebsiteScoreHandling.h>
```

### Classes

- struct [DATESTRUCT](#)
- struct [Timestruct](#)

### Public Member Functions

- [WebsiteScoreHandling](#) (std::string teamName1, std::string teamName2, uint8\_t scoreTeam1, uint8\_t scoreTeam2, unsigned long gameTime, uint8\_t doubleShots)

### Private Member Functions

- int [getScoreTeam](#) (int teamNumber)
- int [getTotalDoubleCupShots](#) ()
- std::string [getTeamName](#) (int teamNumber)
- std::string [getGameTime](#) ()
- std::string [getDate](#) ()
- std::string [getStartTimeDate](#) ()
- std::string [getStartTime](#) ()
- *Printing StartTime while filling out zeroes when number is lower than 10.*
- std::string [getEndTime](#) ()
- void [setScoreTeam](#) (int teamNumber, int score)
- void [setTotalDoubleCupShots](#) (int amountOfShots)
- void [setTeamName](#) (int teamNumber, std::string teamName)
- void [setGameTime](#) (int gameTime)
- void [setDate](#) ()
- void [setStartTime](#) ()
- *Turning back time from the EndTime point to the StartTime point.*
- void [setEndTime](#) ()
- void [writeToCSV](#) ()
- void [getCurrentID](#) ()

## Private Attributes

- `std::fstream fs_`
- `std::fstream errorFs_`
- `std::ifstream ifs_`
- `std::string headers` = "Spil ID,Hold 1,Hold 2,Score Hold 1,Score Hold 2,Varighed,Starttidspunkt,Sluttidspunkt,Dobbelt skud"
- `std::string lastGameID` = "000000"
- `std::string currentGameID` = "000000"
- `std::string oldName` = ""
- `std::string newName` = ""
- `std::string lineToRead` = ""
- `std::string lineToCopy` = ""
- `unsigned int gameIdInteger` = 0
- `unsigned int currentLine` = 0
- `unsigned int lineNumber` = 0
- `std::string teamName1_` = ""
- `std::string teamName2_` = ""
- `unsigned int scoreTeam1_` = 0
- `unsigned int scoreTeam2_` = 0
- `unsigned int totalDoubleCupShots_` = 0
- `DATESTRUCT date_`
- `TIMESTRUCT gameTime_`
- `TIMESTRUCT timeStart_`
- `DATESTRUCT timeStartDate_`
- `TIMESTRUCT timeEnd_`
- `time_t t` = time(NULL)
- `struct tm * timeKeeper_`

### 4.14.1 Detailed Description

Global function; Is called from ScoreSystem upon startup.

### 4.14.2 Constructor & Destructor Documentation

#### 4.14.2.1 WebsiteScoreHandling()

```
WebsiteScoreHandling::WebsiteScoreHandling (
    std::string teamName1,
    std::string teamName2,
    uint8_t scoreTeam1,
    uint8_t scoreTeam2,
    unsigned long gameTime,
    uint8_t doubleShots )
```

Author: Søren Skieller Setting timeinfo for class

Setting game stats

Getting time info

Setting up website

### 4.14.3 Member Function Documentation

#### 4.14.3.1 `getCurrentID()`

```
void WebsiteScoreHandling::getCurrentID ( ) [private]
```

If not opened beforehand, open now

If file not created, create standard file

Closing read/write

Opening read

Resetting ifstream flags

If able to open file

Reading from start of file

Discarding the lines we read but do not need

Positioning the read pointer to second line

Get 2nd line

Getting first element, 6 chars long

If file is empty

Closing read

Closing read/write

Opening read

Resetting ifstream flags

New gameId

Deciding amount of extra numbers for ID (MAX ID: 999999)

#### 4.14.3.2 `getDate()`

```
std::string WebsiteScoreHandling::getDate ( ) [private]
```

#### 4.14.3.3 `getEndTime()`

```
std::string WebsiteScoreHandling::getEndTime ( ) [private]
```

Do nothing

#### 4.14.3.4 getGameTime()

```
std::string WebsiteScoreHandling::getGameTime ( ) [private]
```

Do nothing, only removing the zeroes for the hour counter

#### 4.14.3.5 getScoreTeam()

```
int WebsiteScoreHandling::getScoreTeam (
    int teamNumber ) [private]
```

#### 4.14.3.6 getStartTime()

```
std::string WebsiteScoreHandling::getStartTime ( ) [private]
```

Printing StartTime while filling out zeroes when number is lower than 10.

#### 4.14.3.7 getStartTimeDate()

```
std::string WebsiteScoreHandling::getStartTimeDate ( ) [private]
```

#### 4.14.3.8 getTeamName()

```
std::string WebsiteScoreHandling::getTeamName (
    int teamNumber ) [private]
```

#### 4.14.3.9 getTotalDoubleCupShots()

```
int WebsiteScoreHandling::getTotalDoubleCupShots ( ) [private]
```

#### 4.14.3.10 setDate()

```
void WebsiteScoreHandling::setDate ( ) [private]
```

Setting default start date

#### 4.14.3.11 setEndTime()

```
void WebsiteScoreHandling::setEndTime ( ) [private]
```

Since RPI is GMT

#### 4.14.3.12 setGameTime()

```
void WebsiteScoreHandling::setGameTime (
    int gameTime ) [private]
```

Setting seconds

Find seconds

Find minutes

Find hours

#### 4.14.3.13 setScoreTeam()

```
void WebsiteScoreHandling::setScoreTeam (
    int teamNumber,
    int score ) [private]
```

#### 4.14.3.14 setStartTime()

```
void WebsiteScoreHandling::setStartTime ( ) [private]
```

Turning back time from the EndTime point to the StartTime point.

Removing gametime from current timer

If the seconds go below 0

Minutes

Hours

Days

12 and 0 = december

Months

#### 4.14.3.15 setTeamName()

```
void WebsiteScoreHandling::setTeamName (
    int teamNumber,
    std::string teamName ) [private]
```



## 4.14.3.16 setTotalDoubleCupShots()

```
void WebsiteScoreHandling::setTotalDoubleCupShots (
    int amountOfShots ) [private]
```

## 4.14.3.17 writeToCSV()

```
void WebsiteScoreHandling::writeToCSV ( ) [private]
```

Creating new file temp, using app to add added content to end

Using flush to write nothing to file making sure it is created

Getting ID from last game from final.csv "Spil ID, Hold 1, Hold 2, Score Hold 1, Score Hold 2, Varighed, Starttidspunkt, ← Sluttidspunkt, Dobbelt skud";

Spil ID //!< Hold 1 //!< Hold 2

Score Hold 1 //!< Score Hold 2 //!< Varighed

Starttidspunkt //!< Sluttidspunkt //!< Dobbelt skud

Opening final.csv for overwriting

Discarding the lines we read but do not need

Positioning the read pointer to second line

Reading the rest of the file one line at the time and writing that to fs\_. Since eof flag does not work when the final.csv is placed in the same folder as the website solution, we are here checking ourselves whether the end of document is reached.

Closing both file streams

===== Delete old final.csv, replace it with new temp.csv =====

Finally flush the stream

and close it if not closed yet

## 4.14.4 Member Data Documentation

## 4.14.4.1 currentGameID

```
std::string WebsiteScoreHandling::currentGameID = "000000" [private]
```

#### 4.14.4.2 `currentLine`

```
unsigned int WebsiteScoreHandling::currentLine = 0 [private]
```

#### 4.14.4.3 `date_`

```
DATESTRUCT WebsiteScoreHandling::date_ [private]
```

#### 4.14.4.4 `errorFs_`

```
std::fstream WebsiteScoreHandling::errorFs_ [private]
```

#### 4.14.4.5 `fs_`

```
std::fstream WebsiteScoreHandling::fs_ [private]
```

#### 4.14.4.6 `gameIDInteger`

```
unsigned int WebsiteScoreHandling::gameIDInteger = 0 [private]
```

#### 4.14.4.7 `gameTime_`

```
TIMESTRUCT WebsiteScoreHandling::gameTime_ [private]
```

#### 4.14.4.8 `headers`

```
std::string WebsiteScoreHandling::headers = "Spil ID, Hold 1, Hold 2, Score Hold 1, Score Hold 2, Varighed, Starttidspunkt, Sluttidspunkt, Dobbelt skud" [private]
```

#### 4.14.4.9 ifs\_

```
std::ifstream WebsiteScoreHandling::ifs_ [private]
```

#### 4.14.4.10 lastGameID

```
std::string WebsiteScoreHandling::lastGameID = "000000" [private]
```

#### 4.14.4.11 lineNumber

```
unsigned int WebsiteScoreHandling::lineNumber = 0 [private]
```

#### 4.14.4.12 lineToCopy

```
std::string WebsiteScoreHandling::lineToCopy = "" [private]
```

#### 4.14.4.13 lineToRead

```
std::string WebsiteScoreHandling::lineToRead = "" [private]
```

#### 4.14.4.14 newName

```
std::string WebsiteScoreHandling::newName = "" [private]
```

#### 4.14.4.15 oldName

```
std::string WebsiteScoreHandling::oldName = "" [private]
```

#### 4.14.4.16 scoreTeam1\_

```
unsigned int WebsiteScoreHandling::scoreTeam1_ = 0 [private]
```

#### 4.14.4.17 scoreTeam2\_

```
unsigned int WebsiteScoreHandling::scoreTeam2_ = 0 [private]
```

#### 4.14.4.18 t

```
time_t WebsiteScoreHandling::t = time(NULL) [private]
```

#### 4.14.4.19 teamName1\_

```
std::string WebsiteScoreHandling::teamName1_ = "" [private]
```

#### 4.14.4.20 teamName2\_

```
std::string WebsiteScoreHandling::teamName2_ = "" [private]
```

#### 4.14.4.21 timeEnd\_

```
TIMESTRUCT WebsiteScoreHandling::timeEnd_ [private]
```

#### 4.14.4.22 timeKeeper\_

```
struct tm* WebsiteScoreHandling::timeKeeper_ [private]
```

#### 4.14.4.23 timeStart\_

```
TIMESTRUCT WebsiteScoreHandling::timeStart_ [private]
```

#### 4.14.4.24 timeStartDate\_

```
DATESTRUCT WebsiteScoreHandling::timeStartDate_ [private]
```

#### 4.14.4.25 totalDoubleCupShots\_

```
unsigned int WebsiteScoreHandling::totalDoubleCupShots_ = 0 [private]
```

The documentation for this class was generated from the following files:

- inc/osapi/ScoreSystem/[WebsiteScoreHandling.h](#)
- ScoreSystem/[WebsiteScoreHandling.cpp](#)

## Chapter 5

# File Documentation

### 5.1 inc/osapi/ScoreSystem/Adafruit\_MCP23008.hpp File Reference

```
#include <osapi/ScoreSystem/I2C_reg.hpp>
```

#### Classes

- class [Adafruit\\_MCP23008](#)

#### Macros

- #define [MCP23008\\_ADDRESS](#) 0x20
- #define [MCP23008\\_IODIR](#) 0x00
- #define [MCP23008\\_IPOL](#) 0x01
- #define [MCP23008\\_GPINTEN](#) 0x02
- #define [MCP23008\\_DEFVAL](#) 0x03
- #define [MCP23008\\_INTCON](#) 0x04
- #define [MCP23008\\_IOCON](#) 0x05
- #define [MCP23008\\_GPPU](#) 0x06
- #define [MCP23008\\_INTF](#) 0x07
- #define [MCP23008\\_INTCAP](#) 0x08
- #define [MCP23008\\_GPIO](#) 0x09
- #define [MCP23008\\_OLAT](#) 0x0A

#### 5.1.1 Macro Definition Documentation

##### 5.1.1.1 MCP23008\_ADDRESS

```
#define MCP23008_ADDRESS 0x20
```

#### 5.1.1.2 MCP23008\_DEFVAL

```
#define MCP23008_DEFVAL 0x03
```

#### 5.1.1.3 MCP23008\_GPINTEN

```
#define MCP23008_GPINTEN 0x02
```

#### 5.1.1.4 MCP23008\_GPIO

```
#define MCP23008_GPIO 0x09
```

#### 5.1.1.5 MCP23008\_GPPU

```
#define MCP23008_GPPU 0x06
```

#### 5.1.1.6 MCP23008\_INTCAP

```
#define MCP23008_INTCAP 0x08
```

#### 5.1.1.7 MCP23008\_INTCON

```
#define MCP23008_INTCON 0x04
```

#### 5.1.1.8 MCP23008\_INTF

```
#define MCP23008_INTF 0x07
```

#### 5.1.1.9 MCP23008\_IOCON

```
#define MCP23008_IOCON 0x05
```

#### 5.1.1.10 MCP23008\_IODIR

```
#define MCP23008_IODIR 0x00
```

#### 5.1.1.11 MCP23008\_IPOL

```
#define MCP23008_IPOL 0x01
```

#### 5.1.1.12 MCP23008\_OLAT

```
#define MCP23008_OLAT 0x0A
```

## 5.2 inc/osapi/ScoreSystem/Button.hpp File Reference

```
#include <osapi/ThreadFunctor.hpp>
#include <osapi/MsgQueue.hpp>
#include <osapi/Message.hpp>
#include <fcntl.h>
#include <unistd.h>
#include <errno.h>
```

### Classes

- struct [buttonMessage](#)
- class [Button](#)

### Enumerations

- enum [buttonEvent](#) { [btnPressed](#), [btnRight](#), [btnLeft](#) }

### 5.2.1 Enumeration Type Documentation

#### 5.2.1.1 buttonEvent

```
enum buttonEvent
```

## Enumerator

btnPressed	
btnRight	
btnLeft	

## 5.3 inc/osapi/ScoreSystem/I2C\_reg.hpp File Reference

```
#include <sys/ioctl.h>
#include <fcntl.h>
#include <unistd.h>
#include <osapi/ThreadFunctor.hpp>
#include <osapi/MsgQueue.hpp>
#include <osapi/Message.hpp>
#include <osapi/Conditional.hpp>
#include <osapi/Mutex.hpp>
#include <osapi/Utility.hpp>
#include <osapi/Thread.hpp>
#include <osapi/ScoreSystem/Timer.hpp>
```

## Classes

- struct [psocUpdateMessage](#)
- class [I2C\\_reg](#)

## Enumerations

- enum [i2c\\_messages](#) {  
[TIMER\\_OUT](#), [ARDUINOMESSGE](#), [PSOCBROADCAST](#), [PSOC1MESSAGE](#),  
[PSOC2MESSAGE](#), [PSOC1UPDATE](#) = 50, [PSOC2UPDATE](#) = 60 }

### 5.3.1 Enumeration Type Documentation

#### 5.3.1.1 i2c\_messages

enum [i2c\\_messages](#)

## Enumerator

TIMER_OUT	
ARDUINOMESSGE	
PSOCBROADCAST	
PSOC1MESSAGE	
PSOC2MESSAGE	
PSOC1UPDATE	
PSOC2UPDATE	



## 5.4 inc/osapi/ScoreSystem/LCD.hpp File Reference

```
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>
#include <sys/ioctl.h>
#include <errno.h>
#include <pthread.h>
#include <iostream>
#include <string>
#include <osapi/ScoreSystem/Adafruit_MCP23008.hpp>
```

### Classes

- class [LCD](#)

### Macros

- #define [HIGH](#) 1
- #define [INPUT](#) 1
- #define [LOW](#) 0
- #define [OUTPUT](#) 0
- #define [LCD\\_CLEARDISPLAY](#) 0x01
- #define [LCD\\_RETURNHOME](#) 0x02
- #define [LCD\\_ENTRYMODESET](#) 0x04
- #define [LCD\\_DISPLAYCONTROL](#) 0x08
- #define [LCD\\_CURSORSHIFT](#) 0x10
- #define [LCD\\_FUNCTIONSET](#) 0x20
- #define [LCD\\_SETCGRAMADDR](#) 0x40
- #define [LCD\\_SETDDRAMADDR](#) 0x80
- #define [LCD\\_ENTRYRIGHT](#) 0x00
- #define [LCD\\_ENTRYLEFT](#) 0x02
- #define [LCD\\_ENTRYSHIFTINCREMENT](#) 0x01
- #define [LCD\\_ENTRYSHIFTDECREMENT](#) 0x00
- #define [LCD\\_DISPLAYON](#) 0x04
- #define [LCD\\_DISPLAYOFF](#) 0x00
- #define [LCD\\_CURSORON](#) 0x02
- #define [LCD\\_CURSOROFF](#) 0x00
- #define [LCD\\_BLINKON](#) 0x01
- #define [LCD\\_BLINKOFF](#) 0x00
- #define [LCD\\_DISPLAYMOVE](#) 0x08
- #define [LCD\\_CURSORMOVE](#) 0x00
- #define [LCD\\_MOVERIGHT](#) 0x04
- #define [LCD\\_MOVELEFT](#) 0x00
- #define [LCD\\_8BITMODE](#) 0x10
- #define [LCD\\_4BITMODE](#) 0x00
- #define [LCD\\_2LINE](#) 0x08
- #define [LCD\\_1LINE](#) 0x00
- #define [LCD\\_5x10DOTS](#) 0x04
- #define [LCD\\_5x8DOTS](#) 0x00
- #define [LCD\\_BACKLIGHT](#) 0x08
- #define [LCD\\_NOBACKLIGHT](#) 0x00

## 5.4.1 Macro Definition Documentation

### 5.4.1.1 HIGH

```
#define HIGH 1
```

### 5.4.1.2 INPUT

```
#define INPUT 1
```

### 5.4.1.3 LCD\_1LINE

```
#define LCD_1LINE 0x00
```

### 5.4.1.4 LCD\_2LINE

```
#define LCD_2LINE 0x08
```

### 5.4.1.5 LCD\_4BITMODE

```
#define LCD_4BITMODE 0x00
```

### 5.4.1.6 LCD\_5x10DOTS

```
#define LCD_5x10DOTS 0x04
```

### 5.4.1.7 LCD\_5x8DOTS

```
#define LCD_5x8DOTS 0x00
```

#### 5.4.1.8 LCD\_8BITMODE

```
#define LCD_8BITMODE 0x10
```

#### 5.4.1.9 LCD\_BACKLIGHT

```
#define LCD_BACKLIGHT 0x08
```

#### 5.4.1.10 LCD\_BLINKOFF

```
#define LCD_BLINKOFF 0x00
```

#### 5.4.1.11 LCD\_BLINKON

```
#define LCD_BLINKON 0x01
```

#### 5.4.1.12 LCD\_CLEARDISPLAY

```
#define LCD_CLEARDISPLAY 0x01
```

#### 5.4.1.13 LCD\_CURSORMOVE

```
#define LCD_CURSORMOVE 0x00
```

#### 5.4.1.14 LCD\_CURSOROFF

```
#define LCD_CURSOROFF 0x00
```

#### 5.4.1.15 LCD\_CURSORON

```
#define LCD_CURSORON 0x02
```

#### 5.4.1.16 LCD\_CURSORSHIFT

```
#define LCD_CURSORSHIFT 0x10
```

#### 5.4.1.17 LCD\_DISPLAYCONTROL

```
#define LCD_DISPLAYCONTROL 0x08
```

#### 5.4.1.18 LCD\_DISPLAYMOVE

```
#define LCD_DISPLAYMOVE 0x08
```

#### 5.4.1.19 LCD\_DISPLAYOFF

```
#define LCD_DISPLAYOFF 0x00
```

#### 5.4.1.20 LCD\_DISPLAYON

```
#define LCD_DISPLAYON 0x04
```

#### 5.4.1.21 LCD\_ENTRYLEFT

```
#define LCD_ENTRYLEFT 0x02
```

#### 5.4.1.22 LCD\_ENTRYMODESET

```
#define LCD_ENTRYMODESET 0x04
```

#### 5.4.1.23 LCD\_ENTRYRIGHT

```
#define LCD_ENTRYRIGHT 0x00
```

#### 5.4.1.24 LCD\_ENTRYSHIFTDECREMENT

```
#define LCD_ENTRYSHIFTDECREMENT 0x00
```

#### 5.4.1.25 LCD\_ENTRYSHIFTINCREMENT

```
#define LCD_ENTRYSHIFTINCREMENT 0x01
```

#### 5.4.1.26 LCD\_FUNCTIONSET

```
#define LCD_FUNCTIONSET 0x20
```

#### 5.4.1.27 LCD\_MOVELEFT

```
#define LCD_MOVELEFT 0x00
```

#### 5.4.1.28 LCD\_MOVERIGHT

```
#define LCD_MOVERIGHT 0x04
```

#### 5.4.1.29 LCD\_NOBACKLIGHT

```
#define LCD_NOBACKLIGHT 0x00
```

#### 5.4.1.30 LCD\_RETURNHOME

```
#define LCD_RETURNHOME 0x02
```

#### 5.4.1.31 LCD\_SETCGRAMADDR

```
#define LCD_SETCGRAMADDR 0x40
```

#### 5.4.1.32 LCD\_SETDDRAMADDR

```
#define LCD_SETDDRAMADDR 0x80
```

#### 5.4.1.33 LOW

```
#define LOW 0
```

#### 5.4.1.34 OUTPUT

```
#define OUTPUT 0
```

### 5.5 inc/osapi/ScoreSystem/Page.hpp File Reference

```
#include <iostream>
#include <fstream>
#include <vector>
```

#### Classes

- struct [cursorCoord](#)
- class [Page](#)

#### Enumerations

- enum [pageEvent](#) {  
    [noUpdate](#) = 0, [nextPage](#) = 1, [teamNameEntered](#) = 2, [syncMusic](#) = 3,  
    [fullGame](#) = 4, [halfGame](#) = 5, [placeCupsExit](#) = 6, [startGame](#) = 7,  
    [quickPlay](#) = 8, [reArrangeCups](#) = 9, [team1Rearrange](#) = 10, [team2Rearrange](#) = 11,  
    [doneRearrange](#) = 12, [calibrate](#) = 13 }

*Enum for what state is returned to [ScoreSystemCtrl](#). It's handled in the `handleState()` function.*

#### 5.5.1 Enumeration Type Documentation

##### 5.5.1.1 pageEvent

```
enum pageEvent
```

Enum for what state is returned to [ScoreSystemCtrl](#). It's handled in the `handleState()` function.

## Enumerator

noUpdate	
nextPage	
teamNameEntered	
syncMusic	
fullGame	
halfGame	
placeCupsExit	
startGame	
quickPlay	
reArrangeCups	
team1Rearrange	
team2Rearrange	
doneRearrange	
calibrate	

## 5.6 inc/osapi/ScoreSystem/ScoreSystemCtrl.hpp File Reference

```
#include <osapi/ScoreSystem/LCD.hpp>
#include <osapi/ScoreSystem/Page.hpp>
#include <osapi/ScoreSystem/Button.hpp>
#include <osapi/ScoreSystem/I2C_reg.hpp>
#include <osapi/ScoreSystem/WebsiteScoreHandling.hpp>
#include <string>
#include <vector>
#include <osapi/ClockTime.hpp>
#include <osapi/Time.hpp>
#include <osapi/MsgQueue.hpp>
#include <osapi/Message.hpp>
#include <osapi/ThreadFunctor.hpp>
#include <osapi/Thread.hpp>
```

## Classes

- class [ScoreSystemCtrl](#)

## Enumerations

- enum [pSocMessages](#) {  
[ONE\\_BALL\\_ONE\\_CUP](#) = 0x01, [ONE\\_BALL\\_TWO\\_CUPS](#) = 0x02, [TWO\\_BALLS\\_ONE\\_CUP](#) = 0x03, [ALL\\_CUPS\\_PLACED](#) = 0x04,  
[CUP\\_ZONE\\_READY](#) = 0x05, [EMPTY\\_CUPZONE](#) = 0x06, [CALIBRATE](#) = 0x07, [NO\\_CHANGE](#) = 0xFF }  
*RECEIVABLE PSOC MESSAGES.*

## 5.6.1 Enumeration Type Documentation

### 5.6.1.1 pSocMessages

enum [pSocMessages](#)

RECEIVABLE PSOC MESSAGES.

Enumerator

ONE_BALL_ONE_CUP	
ONE_BALL_TWO_CUPS	
TWO_BALLS_ONE_CUP	
ALL_CUPS_PLACED	
CUP_ZONE_READY	
EMPTY_CUPZONE	
CALIBRATE	
NO_CHANGE	

## 5.7 inc/osapi/ScoreSystem/test.hpp File Reference

```
#include <osapi/ThreadFunctor.hpp>
#include <osapi/ScoreSystem/I2C_reg.hpp>
```

### Classes

- class [Test](#)

## 5.8 inc/osapi/ScoreSystem/Timer.hpp File Reference

```
#include <osapi/ThreadFunctor.hpp>
#include <osapi/MsgQueue.hpp>
```

### Classes

- class [Timer](#)

## 5.9 inc/osapi/ScoreSystem/WebsiteScoreHandling.h File Reference

```
#include <iostream>
#include <string>
#include <fstream>
#include <limits>
#include <time.h>
```



## Classes

- class [WebsiteScoreHandling](#)  
*Global function; Is called from ScoreSystem upon startup.*
- struct [WebsiteScoreHandling::DATESTRUCT](#)
- struct [WebsiteScoreHandling::Timestruct](#)

## Functions

- void [setNewIPJS](#) ()

### 5.9.1 Function Documentation

#### 5.9.1.1 setNewIPJS()

```
void setNewIPJS ( )
```

Author: Søren Skieller Accessing file with info about current IP address

Splitting string to only get IP

Skip equal sign

Remove comma in end

Cleanup

Opening functionality.js and tempFunc for overwriting new IP

Writing javascript before IP to temp file

Writing the new IP to the temp file

Skipping ahead of the IP line

Writing rest of functionality.js to temp file

Closing both file streams

Deleting old functionality.js, replace with tempfunc.js

## 5.10 ScoreSystem/Adafruit\_MCP23008.cpp File Reference

```
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <inttypes.h>
#include <string.h>
#include <sys/ioctl.h>
#include <errno.h>
#include <pthread.h>
#include <iostream>
#include <osapi/ScoreSystem/Adafruit_MCP23008.hpp>
#include <osapi/ScoreSystem/I2C_reg.hpp>
```

### 5.11 ScoreSystem/Screen-ButtonTestCode/Adafruit\_MCP23008.cpp File Reference

```
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <inttypes.h>
#include <string.h>
#include <sys/ioctl.h>
#include <errno.h>
#include <pthread.h>
#include <iostream>
#include <osapi/ScoreSystem/Adafruit_MCP23008.hpp>
#include <osapi/ScoreSystem/I2C_reg.hpp>
```

### 5.12 ScoreSystem/ScreenTestCode/Adafruit\_MCP23008.cpp File Reference

```
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <inttypes.h>
#include <string.h>
#include <sys/ioctl.h>
#include <errno.h>
#include <pthread.h>
#include <iostream>
#include <osapi/ScoreSystem/Adafruit_MCP23008.hpp>
#include <osapi/ScoreSystem/I2C_reg.hpp>
```

### 5.13 ScoreSystem/Button.cpp File Reference

```
#include <osapi/ScoreSystem/Button.hpp>
```

### 5.14 ScoreSystem/Button\_Driver/button\_drv.c File Reference

```
#include <linux/cdev.h>
#include <asm/uaccess.h>
#include <linux/module.h>
#include <linux/platform_device.h>
#include <linux/gpio.h>
#include <linux/of_gpio.h>
#include <linux/wait.h>
#include <linux/sched.h>
#include <linux/interrupt.h>
```

## Macros

- `#define MAXLEN 32`
- `#define MODULE_DEBUG 1`
- `#define ERRGOTO(label, ...)`

## Functions

- static `DECLARE_WAIT_QUEUE_HEAD` (wtqueue)
- static `irqreturn_t buttonUpdate` (int irq, void \*dev)
- static `irqreturn_t rotateLeftUpdate` (int irq, void \*dev)
- static `irqreturn_t rotateRightUpdate` (int irq, void \*dev)
- static `int __init plat_drv_init` (void)
- static `void __exit plat_drv_exit` (void)
- `ssize_t plat_drv_read` (struct file \*filep, char \_\_user \*ubuf, size\_t count, loff\_t \*f\_pos)
- static `int plat_drv_probe` (struct platform\_device \*pdev)
- static `int plat_drv_remove` (struct platform\_device \*pdev)
- `module_init` (plat\_drv\_init)
- `module_exit` (plat\_drv\_exit)
- `MODULE_AUTHOR` ("Jonas Agger Joergensen")
- `MODULE_LICENSE` ("GPL")

## Variables

- static struct platform\_driver `plat_drv_platform_driver`
- struct file\_operations `plat_drv_fops`
- static struct class \* `plat_drv_class`
- static `dev_t devno`
- static struct cdev `plat_drv_cdev`
- `int leftPin` = 9
- `int rightPin` = 11
- `int buttonPin` = 10
- static `int valueRead` = 0
- static `int newValue` = 0
- `int value` = 0
- `int prevValue` = 0
- `int rightStatus`
- `int leftStatus`
- `int prevRightStatus`
- static const struct of\_device\_id `of_plat_drv_platform_device_match` []

### 5.14.1 Macro Definition Documentation

#### 5.14.1.1 ERRGOTO

```
#define ERRGOTO(  
    label,  
    ... )
```

##### Value:

```
{  
    printk (__VA_ARGS__);  
    goto label;  
} while (0)
```

\\

#### 5.14.1.2 MAXLEN

```
#define MAXLEN 32
```

#### 5.14.1.3 MODULE\_DEBUG

```
#define MODULE_DEBUG 1
```

### 5.14.2 Function Documentation

#### 5.14.2.1 buttonUpdate()

```
static irqreturn_t buttonUpdate (  
    int irq,  
    void * dev ) [static]
```

#### 5.14.2.2 DECLARE\_WAIT\_QUEUE\_HEAD()

```
static DECLARE_WAIT_QUEUE_HEAD (  
    wtqueue ) [static]
```

#### 5.14.2.3 MODULE\_AUTHOR()

```
MODULE_AUTHOR (
    "Jonas Agger Joergensen" )
```

#### 5.14.2.4 module\_exit()

```
module_exit (
    plat_drv_exit )
```

#### 5.14.2.5 module\_init()

```
module_init (
    plat_drv_init )
```

#### 5.14.2.6 MODULE\_LICENSE()

```
MODULE_LICENSE (
    "GPL" )
```

#### 5.14.2.7 plat\_drv\_exit()

```
static void __exit plat_drv_exit (
    void ) [static]
```

#### 5.14.2.8 plat\_drv\_init()

```
static int __init plat_drv_init (
    void ) [static]
```

#### 5.14.2.9 plat\_drv\_probe()

```
static int plat_drv_probe (
    struct platform_device * pdev ) [static]
```

#### 5.14.2.10 plat\_drv\_read()

```
ssize_t plat_drv_read (
    struct file * filep,
    char __user * ubuf,
    size_t count,
    loff_t * f_pos )
```

#### 5.14.2.11 plat\_drv\_remove()

```
static int plat_drv_remove (
    struct platform_device * pdev ) [static]
```

#### 5.14.2.12 rotateLeftUpdate()

```
static irqreturn_t rotateLeftUpdate (
    int irq,
    void * dev ) [static]
```

#### 5.14.2.13 rotateRightUpdate()

```
static irqreturn_t rotateRightUpdate (
    int irq,
    void * dev ) [static]
```

### 5.14.3 Variable Documentation

#### 5.14.3.1 buttonPin

```
int buttonPin = 10
```

#### 5.14.3.2 devno

```
dev_t devno [static]
```

#### 5.14.3.3 leftPin

```
int leftPin = 9
```

#### 5.14.3.4 leftStatus

```
int leftStatus
```

#### 5.14.3.5 newValue

```
int newValue = 0 [static]
```

#### 5.14.3.6 of\_plat\_drv\_platform\_device\_match

```
const struct of_device_id of_plat_drv_platform_device_match[] [static]
```

**Initial value:**

```
= {  
    { .compatible = "ase, knap", }, {}  
}
```

#### 5.14.3.7 plat\_drv\_cdev

```
struct cdev plat_drv_cdev [static]
```

#### 5.14.3.8 plat\_drv\_class

```
struct class* plat_drv_class [static]
```

#### 5.14.3.9 plat\_drv\_fops

```
struct file_operations plat_drv_fops
```

**Initial value:**

```
=  
{  
    .owner    = THIS_MODULE,  
    .read     = plat_drv_read,  
}
```

#### 5.14.3.10 plat\_drv\_platform\_driver

```
static struct platform_driver plat_drv_platform_driver [static]
```

**Initial value:**

```
= {  
    .probe      = plat_drv_probe,  
    .remove     = plat_drv_remove,  
    .driver     = {  
        .name    = "knap",  
        .of_match_table = of_plat_drv_platform_device_match,  
        .owner   = THIS_MODULE,  
    },  
}
```

#### 5.14.3.11 prevRightStatus

```
int prevRightStatus
```

#### 5.14.3.12 prevValue

```
int prevValue = 0
```

#### 5.14.3.13 rightPin

```
int rightPin = 11
```



#### 5.14.3.14 rightStatus

```
int rightStatus
```

#### 5.14.3.15 value

```
int value = 0
```

#### 5.14.3.16 valueRead

```
int valueRead = 0 [static]
```

## 5.15 ScoreSystem/Button\_Driver/button\_drv.mod.c File Reference

```
#include <linux/module.h>
#include <linux/vermagic.h>
#include <linux/compiler.h>
```

### Functions

- [MODULE\\_INFO](#) (vermagic, VERMAGIC\_STRING)
- `__visible struct module __this_module` [\\_\\_attribute\\_\\_](#) ((section(".gnu.linkonce.this\_module")))
- `static const struct modversion_info ____versions[]` [\\_\\_used](#) [\\_\\_attribute\\_\\_](#) ((section("\_\_versions")))
- `static const char __module_depends[]` [\\_\\_used](#) [\\_\\_attribute\\_\\_](#) ((section(".modinfo")))
- [MODULE\\_INFO](#) (srcversion, "F78163A0CC945A5E0B2E434")

### 5.15.1 Function Documentation

#### 5.15.1.1 `__attribute__`() [1/3]

```
__visible struct module __this_module __attribute__ (
    (section(".gnu.linkonce.this_module")) )
```

**5.15.1.2** `__attribute__()` [2/3]

```
static const struct modversion_info ____versions [] __used __attribute__ (
    (section("__versions"))) ) [static]
```

**5.15.1.3** `__attribute__()` [3/3]

```
static const char __module_depends [] __used __attribute__ (
    (section(".modinfo"))) ) [static]
```

**5.15.1.4** `MODULE_INFO()` [1/2]

```
MODULE_INFO (
    vermagic ,
    VERMAGIC_STRING )
```

**5.15.1.5** `MODULE_INFO()` [2/2]

```
MODULE_INFO (
    srcversion ,
    "F78163A0CC945A5E0B2E434" )
```

**5.16** `ScoreSystem/buttonTestCode/host/files/main.d` File Reference**5.17** `ScoreSystem/buttonTestCode/target/files/main.d` File Reference**5.18** `ScoreSystem/Screen-ButtonTestCode/host/files/main.d` File Reference**5.19** `ScoreSystem/Screen-ButtonTestCode/target/files/main.d` File Reference**5.20** `ScoreSystem/buttonTestCode/main.cpp` File Reference

```
#include <fcntl.h>
#include <unistd.h>
#include <errno.h>
```

## Functions

- int [main](#) ()

### 5.20.1 Function Documentation

#### 5.20.1.1 main()

```
int main ( )
```

## 5.21 ScoreSystem/i2cTestCode/main.cpp File Reference

```
#include <osapi/ScoreSystem/I2C_reg.hpp>
#include <iostream>
#include <unistd.h>
#include <osapi/Thread.hpp>
#include <osapi/ScoreSystem/test.hpp>
```

## Functions

- int [main](#) ()

### 5.21.1 Function Documentation

#### 5.21.1.1 main()

```
int main ( )
```

## 5.22 ScoreSystem/main.cpp File Reference

```
#include <osapi/ScoreSystem/ScoreSystemCtrl.hpp>
#include <osapi/Thread.hpp>
#include <sys/types.h>
#include <unistd.h>
```

## Functions

- int [main](#) ()

### 5.22.1 Function Documentation

#### 5.22.1.1 main()

```
int main ( )
```

## 5.23 ScoreSystem/Screen-ButtonTestCode/main.cpp File Reference

```
#include <fcntl.h>
#include <unistd.h>
#include <errno.h>
#include <iostream>
#include <osapi/ScoreSystem/LCD.hpp>
```

### Functions

- int [main](#) ()

### 5.23.1 Function Documentation

#### 5.23.1.1 main()

```
int main ( )
```

## 5.24 ScoreSystem/ScreenTestCode/main.cpp File Reference

```
#include <osapi/ScoreSystem/LCD.hpp>
```

### Functions

- int [main](#) ()

### 5.24.1 Function Documentation

#### 5.24.1.1 main()

```
int main ( )
```

## 5.25 ScoreSystem/TimeTestCode/main.cpp File Reference

```
#include <osapi/ClockTime.hpp>
#include <iostream>
#include <unistd.h>
#include <string>
```

### Functions

- int [main](#) ()

#### 5.25.1 Function Documentation

##### 5.25.1.1 main()

```
int main ( )
```

## 5.26 ScoreSystem/I2C\_reg.cpp File Reference

```
#include <osapi/ScoreSystem/I2C_reg.hpp>
#include <iostream>
```

## 5.27 ScoreSystem/i2cTestCode/I2C\_reg.cpp File Reference

```
#include <osapi/ScoreSystem/I2C_reg.hpp>
#include <iostream>
```

## 5.28 ScoreSystem/Screen-ButtonTestCode/I2C\_reg.cpp File Reference

```
#include <osapi/ScoreSystem/I2C_reg.hpp>
#include <iostream>
```

### 5.29 ScoreSystem/ScreenTestCode/I2C\_reg.cpp File Reference

```
#include <osapi/ScoreSystem/I2C_reg.hpp>
#include <iostream>
```

### 5.30 ScoreSystem/i2cTestCode/test.cpp File Reference

```
#include <osapi/ScoreSystem/test.hpp>
#include <unistd.h>
```

### 5.31 ScoreSystem/i2cTestCode/Timer.cpp File Reference

```
#include <osapi/ScoreSystem/Timer.hpp>
#include <unistd.h>
```

### 5.32 ScoreSystem/Screen-ButtonTestCode/Timer.cpp File Reference

```
#include <osapi/ScoreSystem/Timer.hpp>
#include <unistd.h>
```

### 5.33 ScoreSystem/ScreenTestCode/Timer.cpp File Reference

```
#include <osapi/ScoreSystem/Timer.hpp>
#include <unistd.h>
```

### 5.34 ScoreSystem/Timer.cpp File Reference

```
#include <osapi/ScoreSystem/Timer.hpp>
#include <unistd.h>
```

### 5.35 ScoreSystem/LCD.cpp File Reference

```
#include <osapi/ScoreSystem/LCD.hpp>
```

## Macros

- #define BV(bit) (1 <<(bit))

### 5.35.1 Macro Definition Documentation

#### 5.35.1.1 BV

```
#define BV(  
    bit ) (1 <<(bit))
```

## 5.36 ScoreSystem/Screen-ButtonTestCode/LCD.cpp File Reference

```
#include <osapi/ScoreSystem/LCD.hpp>
```

## Macros

- #define BV(bit) (1 <<(bit))

### 5.36.1 Macro Definition Documentation

#### 5.36.1.1 BV

```
#define BV(  
    bit ) (1 <<(bit))
```

## 5.37 ScoreSystem/ScreenTestCode/LCD.cpp File Reference

```
#include <osapi/ScoreSystem/LCD.hpp>
```

## Macros

- #define BV(bit) (1 <<(bit))

### 5.37.1 Macro Definition Documentation

#### 5.37.1.1 BV

```
#define BV(  
    bit ) (1 <<(bit))
```

### 5.38 ScoreSystem/Page.cpp File Reference

```
#include <osapi/ScoreSystem/Button.hpp>
```

### 5.39 ScoreSystem/ScoreSystemCtrl.cpp File Reference

```
#include <osapi/ScoreSystem/ScoreSystemCtrl.hpp>  
#include <fstream>  
#include <iostream>
```

### 5.40 ScoreSystem/Screen-ButtonTestCode/target/files/Adafruit\_MCP23008.d File Reference

### 5.41 ScoreSystem/Screen-ButtonTestCode/target/files/I2C\_reg.d File Reference

### 5.42 ScoreSystem/Screen-ButtonTestCode/target/files/LCD.d File Reference

### 5.43 ScoreSystem/Screen-ButtonTestCode/target/files/Timer.d File Reference

### 5.44 ScoreSystem/WebsiteScoreHandling.cpp File Reference

```
#include "WebsiteScoreHandling.h"
```

#### Functions

- void [setNewIPJS](#) ()



### 5.44.1 Function Documentation

#### 5.44.1.1 setNewIPJS()

```
void setNewIPJS ( )
```

Author: Søren Skieller Accessing file with info about current IP address

Splitting string to only get IP

Skip equal sign

Remove comma in end

Cleanup

Opening functionality.js and tempFunc for overwriting new IP

Writing javascript before IP to temp file

Writing the new IP to the temp file

Skipping ahead of the IP line

Writing rest of functionality.js to temp file

Closing both file streams

Deleting old functionality.js, replace with tempfunc.js



# Index

- `__attribute__`
    - `button_drv.mod.c`, [71](#), [72](#)
  - `_data_pins`
    - LCD, [26](#)
  - `_enable_pin`
    - LCD, [26](#)
  - `_rs_pin`
    - LCD, [27](#)
  - `_rw_pin`
    - LCD, [27](#)
  - `~Button`
    - Button, [10](#)
  - `~I2C_reg`
    - I2C\_reg, [16](#)
  - `~Timer`
    - Timer, [40](#)
- Adafruit\_MCP23008, [7](#)
  - `begin`, [7](#), [8](#)
  - `digitalRead`, [8](#)
  - `digitalWrite`, [8](#)
  - `fd`, [9](#)
  - `i2caddr`, [9](#)
  - `pinMode`, [8](#)
  - `pullUp`, [8](#)
  - `read8`, [8](#)
  - `readGPIO`, [9](#)
  - `write8`, [9](#)
  - `writeGPIO`, [9](#)
- Adafruit\_MCP23008.hpp
  - `MCP23008_ADDRESS`, [51](#)
  - `MCP23008_DEFVAL`, [51](#)
  - `MCP23008_GPINTEN`, [52](#)
  - `MCP23008_GPIO`, [52](#)
  - `MCP23008_GPPU`, [52](#)
  - `MCP23008_INTCAP`, [52](#)
  - `MCP23008_INTCON`, [52](#)
  - `MCP23008_INTF`, [52](#)
  - `MCP23008_IOCON`, [52](#)
  - `MCP23008_IODIR`, [52](#)
  - `MCP23008_IPOL`, [53](#)
  - `MCP23008_OLAT`, [53](#)
- `arduinoAdress_`
  - I2C\_reg, [20](#)
- `arduinoMessage_`
  - I2C\_reg, [20](#)
- `begin`
  - Adafruit\_MCP23008, [7](#), [8](#)
  - LCD, [23](#)
- `blink`
  - LCD, [24](#)
- `btn`
  - ScoreSystemCtrl, [35](#)
- Button, [10](#)
  - `~Button`, [10](#)
  - Button, [10](#)
  - `fd`, [11](#)
  - `mq_`, [11](#)
  - `run`, [10](#)
  - `running`, [11](#)
  - `value`, [11](#)
- Button.hpp
  - `buttonEvent`, [53](#)
- `button_drv.c`
  - `buttonPin`, [68](#)
  - `buttonUpdate`, [66](#)
  - `DECLARE_WAIT_QUEUE_HEAD`, [66](#)
  - `devno`, [68](#)
  - `ERRGOTO`, [65](#)
  - `leftPin`, [68](#)
  - `leftStatus`, [69](#)
  - `MAXLEN`, [66](#)
  - `MODULE_AUTHOR`, [66](#)
  - `MODULE_DEBUG`, [66](#)
  - `MODULE_LICENSE`, [67](#)
  - `module_exit`, [67](#)
  - `module_init`, [67](#)
  - `newValue`, [69](#)
  - `of_plat_drv_platform_device_match`, [69](#)
  - `plat_drv_cdev`, [69](#)
  - `plat_drv_class`, [69](#)
  - `plat_drv_exit`, [67](#)
  - `plat_drv_fops`, [69](#)
  - `plat_drv_init`, [67](#)
  - `plat_drv_platform_driver`, [70](#)
  - `plat_drv_probe`, [67](#)
  - `plat_drv_read`, [67](#)
  - `plat_drv_remove`, [68](#)
  - `prevRightStatus`, [70](#)
  - `prevValue`, [70](#)
  - `rightPin`, [70](#)
  - `rightStatus`, [70](#)
  - `rotateLeftUpdate`, [68](#)
  - `rotateRightUpdate`, [68](#)
  - `value`, [71](#)
  - `valueRead`, [71](#)
- `button_drv.mod.c`
  - `__attribute__`, [71](#), [72](#)

- MODULE\_INFO, 72
- buttonEvent
  - Button.hpp, 53
- buttonLeft
  - Page, 29
- buttonMessage, 11
  - buttonMessage, 12
  - x, 12
- buttonPin
  - button\_drv.c, 68
- buttonPressed
  - Page, 29
- buttonRight
  - Page, 29
- buttonTestCode/main.cpp
  - main, 73
- buttonThread
  - ScoreSystemCtrl, 35
- buttonUpdate
  - button\_drv.c, 66
- BV
  - LCD.cpp, 77
  - Screen-ButtonTestCode/LCD.cpp, 77
  - ScreenTestCode/LCD.cpp, 78
- charWrite
  - LCD, 24
- clear
  - LCD, 24
- collectiveDoubleShots\_
  - ScoreSystemCtrl, 36
- command
  - LCD, 24
- cond\_
  - I2C\_reg, 20
- currentAddr\_
  - I2C\_reg, 20
- currentChar
  - Page, 30
- currentGameID
  - WebsiteScoreHandling, 47
- currentLine
  - LCD, 27
  - WebsiteScoreHandling, 47
- currentScreen
  - ScoreSystemCtrl, 36
- cursor
  - LCD, 24
- cursorCoord, 12
  - dir\_, 13
  - line\_, 13
  - row\_, 13
  - type\_, 13
- cursorPos\_
  - Page, 30
- DECLARE\_WAIT\_QUEUE\_HEAD
  - button\_drv.c, 66
- date\_
  - WebsiteScoreHandling, 48
- day\_
  - WebsiteScoreHandling::DATESTRUCT, 13
- devno
  - button\_drv.c, 68
- digitalRead
  - Adafruit\_MCP23008, 8
- digitalWrite
  - Adafruit\_MCP23008, 8
- dir\_
  - cursorCoord, 13
- display
  - LCD, 24
  - ScoreSystemCtrl, 36
- displayControl
  - LCD, 27
- displayFunction
  - LCD, 27
- displayInit
  - I2C\_reg, 16
- displayMQ\_
  - I2C\_reg, 20
- displayMode
  - LCD, 27
- displayRead
  - I2C\_reg, 16
- displayScreen
  - Page, 30
- displayWrite
  - I2C\_reg, 16
- ERRGOTO
  - button\_drv.c, 65
- En
  - LCD, 27
- endGame
  - ScoreSystemCtrl, 34
- errorFs\_
  - WebsiteScoreHandling, 48
- fd
  - Adafruit\_MCP23008, 9
  - Button, 11
- fd\_
  - I2C\_reg, 20
- fs\_
  - WebsiteScoreHandling, 48
- gameIDInteger
  - WebsiteScoreHandling, 48
- gameTime\_
  - ScoreSystemCtrl, 36
  - WebsiteScoreHandling, 48
- getCurrentID
  - WebsiteScoreHandling, 44
- getDate
  - WebsiteScoreHandling, 44
- getEndTime
  - WebsiteScoreHandling, 44

- getGameTime
  - WebsiteScoreHandling, [44](#)
- getInstance
  - I2C\_reg, [16](#)
- getIP
  - ScoreSystemCtrl, [34](#)
- getMsgQueue
  - I2C\_reg, [17](#)
  - ScoreSystemCtrl, [34](#)
- getScoreTeam
  - WebsiteScoreHandling, [45](#)
- getStartTime
  - WebsiteScoreHandling, [45](#)
- getStartTimeDate
  - WebsiteScoreHandling, [45](#)
- getTeamName
  - WebsiteScoreHandling, [45](#)
- getTotalDoubleCupShots
  - WebsiteScoreHandling, [45](#)
- HIGH
  - LCD.hpp, [56](#)
- handleMsg
  - I2C\_reg, [17](#)
  - ScoreSystemCtrl, [34](#)
- handlePsocUpdate
  - ScoreSystemCtrl, [34](#)
- handleState
  - ScoreSystemCtrl, [35](#)
- headers
  - WebsiteScoreHandling, [48](#)
- home
  - LCD, [25](#)
- hours\_
  - WebsiteScoreHandling::TIMESTRUCT, [41](#)
- I2C\_reg, [14](#)
  - ~I2C\_reg, [16](#)
  - arduinoAddress\_, [20](#)
  - arduinoMessage\_, [20](#)
  - cond\_, [20](#)
  - currentAddr\_, [20](#)
  - displayInit, [16](#)
  - displayMQ\_, [20](#)
  - displayRead, [16](#)
  - displayWrite, [16](#)
  - fd\_, [20](#)
  - getInstance, [16](#)
  - getMsgQueue, [17](#)
  - handleMsg, [17](#)
  - I2C\_reg, [15](#)
  - lockI2C, [17](#)
  - mq\_, [20](#)
  - mut\_, [20](#)
  - psoc1Adress\_, [21](#)
  - psoc2Adress\_, [21](#)
  - psocMessage\_, [21](#)
  - psocUpdate, [17](#)
  - receivingPsoC\_, [21](#)
  - run, [17](#)
  - running\_, [21](#)
  - ScoreSystemMQ\_, [21](#)
  - screenAdress\_, [21](#)
  - sendArduinoMessage, [17](#)
  - sendPsoc1Message, [18](#)
  - sendPsoc2Message, [18](#)
  - sendPsocBroadcast, [18](#)
  - sendPsocMessage, [18](#)
  - setArduinoMessage, [18](#)
  - setMsgQueueScoreSystem, [18](#)
  - setPsocMessage, [19](#)
  - setUp, [19](#)
  - state\_, [21](#)
  - stopPsocPolling, [19](#)
  - testStatic, [19](#)
  - timer1\_, [22](#)
  - tt\_, [22](#)
  - unlockI2C, [19](#)
- I2C\_reg.hpp
  - i2c\_messages, [54](#)
- i2c\_
  - LCD, [27](#)
- i2c\_messages
  - I2C\_reg.hpp, [54](#)
- i2cTestCode/main.cpp
  - main, [73](#)
- i2cThread
  - ScoreSystemCtrl, [36](#)
- i2caddr
  - Adafruit\_MCP23008, [9](#)
- INPUT
  - LCD.hpp, [56](#)
- id\_
  - Timer, [40](#)
- ifs\_
  - WebsiteScoreHandling, [48](#)
- inc/osapi/ScoreSystem/Adafruit\_MCP23008.hpp, [51](#)
- inc/osapi/ScoreSystem/Button.hpp, [53](#)
- inc/osapi/ScoreSystem/I2C\_reg.hpp, [54](#)
- inc/osapi/ScoreSystem/LCD.hpp, [55](#)
- inc/osapi/ScoreSystem/Page.hpp, [60](#)
- inc/osapi/ScoreSystem/ScoreSystemCtrl.hpp, [61](#)
- inc/osapi/ScoreSystem/Timer.hpp, [62](#)
- inc/osapi/ScoreSystem/WebsiteScoreHandling.h, [62](#)
- inc/osapi/ScoreSystem/test.hpp, [62](#)
- ip\_
  - ScoreSystemCtrl, [36](#)
- LCD.cpp
  - BV, [77](#)
- LCD.hpp
  - HIGH, [56](#)
  - INPUT, [56](#)
  - LCD\_1LINE, [56](#)
  - LCD\_2LINE, [56](#)
  - LCD\_4BITMODE, [56](#)
  - LCD\_5x10DOTS, [56](#)
  - LCD\_5x8DOTS, [56](#)

LCD\_8BITMODE, [56](#)  
 LCD\_BACKLIGHT, [57](#)  
 LCD\_BLINKOFF, [57](#)  
 LCD\_BLINKON, [57](#)  
 LCD\_CLEARDISPLAY, [57](#)  
 LCD\_CURSORMOVE, [57](#)  
 LCD\_CURSOROFF, [57](#)  
 LCD\_CUSORON, [57](#)  
 LCD\_CURSORSHIFT, [57](#)  
 LCD\_DISPLAYCONTROL, [58](#)  
 LCD\_DISPLAYMOVE, [58](#)  
 LCD\_DISPLAYOFF, [58](#)  
 LCD\_DISPLAYON, [58](#)  
 LCD\_ENTRYLEFT, [58](#)  
 LCD\_ENTRYMODESET, [58](#)  
 LCD\_ENTRYRIGHT, [58](#)  
 LCD\_ENTRYSHIFTDECREMENT, [58](#)  
 LCD\_ENTRYSHIFTINCREMENT, [59](#)  
 LCD\_FUNCTIONSET, [59](#)  
 LCD\_MOVELEFT, [59](#)  
 LCD\_MOVERIGHT, [59](#)  
 LCD\_NOBACKLIGHT, [59](#)  
 LCD\_RETURNHOME, [59](#)  
 LCD\_SETCGRAMADDR, [59](#)  
 LCD\_SETDDRAMADDR, [59](#)  
 LOW, [60](#)  
 OUTPUT, [60](#)  
 LCD\_1LINE  
     [LCD.hpp, 56](#)  
 LCD\_2LINE  
     [LCD.hpp, 56](#)  
 LCD\_4BITMODE  
     [LCD.hpp, 56](#)  
 LCD\_5x10DOTS  
     [LCD.hpp, 56](#)  
 LCD\_5x8DOTS  
     [LCD.hpp, 56](#)  
 LCD\_8BITMODE  
     [LCD.hpp, 56](#)  
 LCD\_BACKLIGHT  
     [LCD.hpp, 57](#)  
 LCD\_BLINKOFF  
     [LCD.hpp, 57](#)  
 LCD\_BLINKON  
     [LCD.hpp, 57](#)  
 LCD\_CLEARDISPLAY  
     [LCD.hpp, 57](#)  
 LCD\_CURSORMOVE  
     [LCD.hpp, 57](#)  
 LCD\_CURSOROFF  
     [LCD.hpp, 57](#)  
 LCD\_CUSORON  
     [LCD.hpp, 57](#)  
 LCD\_CURSORSHIFT  
     [LCD.hpp, 57](#)  
 LCD\_DISPLAYCONTROL  
     [LCD.hpp, 58](#)  
 LCD\_DISPLAYMOVE  
     [LCD.hpp, 58](#)  
 LCD\_DISPLAYOFF  
     [LCD.hpp, 58](#)  
 LCD\_DISPLAYON  
     [LCD.hpp, 58](#)  
 LCD\_ENTRYLEFT  
     [LCD.hpp, 58](#)  
 LCD\_ENTRYMODESET  
     [LCD.hpp, 58](#)  
 LCD\_ENTRYRIGHT  
     [LCD.hpp, 58](#)  
 LCD\_ENTRYSHIFTDECREMENT  
     [LCD.hpp, 58](#)  
 LCD\_ENTRYSHIFTINCREMENT  
     [LCD.hpp, 59](#)  
 LCD\_FUNCTIONSET  
     [LCD.hpp, 59](#)  
 LCD\_MOVELEFT  
     [LCD.hpp, 59](#)  
 LCD\_MOVERIGHT  
     [LCD.hpp, 59](#)  
 LCD\_NOBACKLIGHT  
     [LCD.hpp, 59](#)  
 LCD\_RETURNHOME  
     [LCD.hpp, 59](#)  
 LCD\_SETCGRAMADDR  
     [LCD.hpp, 59](#)  
 LCD\_SETDDRAMADDR  
     [LCD.hpp, 59](#)  
 LCD, [22](#)  
     \_data\_pins, [26](#)  
     \_enable\_pin, [26](#)  
     \_rs\_pin, [27](#)  
     \_rw\_pin, [27](#)  
     begin, [23](#)  
     blink, [24](#)  
     charWrite, [24](#)  
     clear, [24](#)  
     command, [24](#)  
     currentLine, [27](#)  
     cursor, [24](#)  
     display, [24](#)  
     displayControl, [27](#)  
     displayFunction, [27](#)  
     displayMode, [27](#)  
     En, [27](#)  
     home, [25](#)  
     i2c\_, [27](#)  
     LCD, [23](#)  
     lcdWrite\_four\_bits, [25](#)  
     noBlink, [25](#)  
     noCursor, [25](#)  
     noDisplay, [25](#)  
     numLines, [28](#)  
     readReg, [25](#)  
     Rs, [28](#)  
     Rw, [28](#)  
     send, [26](#)

- setCursor, [26](#)
  - stringWrite, [26](#)
- LOW
  - LCD.hpp, [60](#)
- lastGameID
  - WebsiteScoreHandling, [49](#)
- lcdWrite\_four\_bits
  - LCD, [25](#)
- leftPin
  - button\_drv.c, [68](#)
- leftStatus
  - button\_drv.c, [69](#)
- line\_
  - cursorCoord, [13](#)
- lineNumber
  - WebsiteScoreHandling, [49](#)
- lineToCopy
  - WebsiteScoreHandling, [49](#)
- lineToRead
  - WebsiteScoreHandling, [49](#)
- lockI2C
  - I2C\_reg, [17](#)
- MAXLEN
  - button\_drv.c, [66](#)
- MCP23008\_ADDRESS
  - Adafruit\_MCP23008.hpp, [51](#)
- MCP23008\_DEFVAL
  - Adafruit\_MCP23008.hpp, [51](#)
- MCP23008\_GPINTEN
  - Adafruit\_MCP23008.hpp, [52](#)
- MCP23008\_GPIO
  - Adafruit\_MCP23008.hpp, [52](#)
- MCP23008\_GPPU
  - Adafruit\_MCP23008.hpp, [52](#)
- MCP23008\_INTCAP
  - Adafruit\_MCP23008.hpp, [52](#)
- MCP23008\_INTCON
  - Adafruit\_MCP23008.hpp, [52](#)
- MCP23008\_INTF
  - Adafruit\_MCP23008.hpp, [52](#)
- MCP23008\_IOCON
  - Adafruit\_MCP23008.hpp, [52](#)
- MCP23008\_IODIR
  - Adafruit\_MCP23008.hpp, [52](#)
- MCP23008\_IPOL
  - Adafruit\_MCP23008.hpp, [53](#)
- MCP23008\_OLAT
  - Adafruit\_MCP23008.hpp, [53](#)
- MODULE\_AUTHOR
  - button\_drv.c, [66](#)
- MODULE\_DEBUG
  - button\_drv.c, [66](#)
- MODULE\_INFO
  - button\_drv.mod.c, [72](#)
- MODULE\_LICENSE
  - button\_drv.c, [67](#)
- main
  - buttonTestCode/main.cpp, [73](#)
  - i2cTestCode/main.cpp, [73](#)
  - main.cpp, [74](#)
  - Screen-ButtonTestCode/main.cpp, [74](#)
  - ScreenTestCode/main.cpp, [74](#)
  - TimeTestCode/main.cpp, [75](#)
- main.cpp
  - main, [74](#)
- minutes\_
  - WebsiteScoreHandling::TIMESTRUCT, [41](#)
- module\_exit
  - button\_drv.c, [67](#)
- module\_init
  - button\_drv.c, [67](#)
- month\_
  - WebsiteScoreHandling::DATESTRUCT, [14](#)
- mq\_
  - Button, [11](#)
  - I2C\_reg, [20](#)
  - ScoreSystemCtrl, [36](#)
  - Timer, [41](#)
- mut\_
  - I2C\_reg, [20](#)
- newName
  - WebsiteScoreHandling, [49](#)
- newValue
  - button\_drv.c, [69](#)
- noBlink
  - LCD, [25](#)
- noCursor
  - LCD, [25](#)
- noDisplay
  - LCD, [25](#)
- nr\_Cups\_In\_Game\_
  - ScoreSystemCtrl, [36](#)
- nrCursorPos\_
  - Page, [30](#)
- numLines
  - LCD, [28](#)
- OUTPUT
  - LCD.hpp, [60](#)
- of\_plat\_drv\_platform\_device\_match
  - button\_drv.c, [69](#)
- oldName
  - WebsiteScoreHandling, [49](#)
- pSocMessages
  - ScoreSystemCtrl.hpp, [61](#)
- Page, [28](#)
  - buttonLeft, [29](#)
  - buttonPressed, [29](#)
  - buttonRight, [29](#)
  - currentChar, [30](#)
  - cursorPos\_, [30](#)
  - displayScreen, [30](#)
  - nrCursorPos\_, [30](#)
  - Page, [29](#)
  - pageText, [31](#)

- possibleCursorPos, [31](#)
- resetPage, [30](#)
- selectingChar, [31](#)
- teamEnter\_, [31](#)
- teamNameArr, [31](#)
- Page.hpp
  - pageEvent, [60](#)
- pageEvent
  - Page.hpp, [60](#)
- pageText
  - Page, [31](#)
- pages\_
  - ScoreSystemCtrl, [37](#)
- pinMode
  - Adafruit\_MCP23008, [8](#)
- plat\_drv\_cdev
  - button\_drv.c, [69](#)
- plat\_drv\_class
  - button\_drv.c, [69](#)
- plat\_drv\_exit
  - button\_drv.c, [67](#)
- plat\_drv\_fops
  - button\_drv.c, [69](#)
- plat\_drv\_init
  - button\_drv.c, [67](#)
- plat\_drv\_platform\_driver
  - button\_drv.c, [70](#)
- plat\_drv\_probe
  - button\_drv.c, [67](#)
- plat\_drv\_read
  - button\_drv.c, [67](#)
- plat\_drv\_remove
  - button\_drv.c, [68](#)
- possibleCursorPos
  - Page, [31](#)
- prevRightStatus
  - button\_drv.c, [70](#)
- prevValue
  - button\_drv.c, [70](#)
- psoc1Adress\_
  - I2C\_reg, [21](#)
- psoc2Adress\_
  - I2C\_reg, [21](#)
- psocMessage\_
  - I2C\_reg, [21](#)
- psocUpdate
  - I2C\_reg, [17](#)
- psocUpdateMessage, [32](#)
  - psocUpdateMessage, [32](#)
  - val\_, [32](#)
- pullUp
  - Adafruit\_MCP23008, [8](#)
- reArranging\_
  - ScoreSystemCtrl, [37](#)
- reArrangingZone\_
  - ScoreSystemCtrl, [37](#)
- read8
  - Adafruit\_MCP23008, [8](#)
- readGPIO
  - Adafruit\_MCP23008, [9](#)
- readReg
  - LCD, [25](#)
- receivingPsoC\_
  - I2C\_reg, [21](#)
- resetGame
  - ScoreSystemCtrl, [35](#)
- resetPage
  - Page, [30](#)
- rightPin
  - button\_drv.c, [70](#)
- rightStatus
  - button\_drv.c, [70](#)
- rotateLeftUpdate
  - button\_drv.c, [68](#)
- rotateRightUpdate
  - button\_drv.c, [68](#)
- row\_
  - cursorCoord, [13](#)
- Rs
  - LCD, [28](#)
- run
  - Button, [10](#)
  - I2C\_reg, [17](#)
  - ScoreSystemCtrl, [35](#)
  - Test, [39](#)
  - Timer, [40](#)
- running
  - Button, [11](#)
- running\_
  - I2C\_reg, [21](#)
  - Timer, [41](#)
- Rw
  - LCD, [28](#)
- score\_Team\_1\_
  - ScoreSystemCtrl, [37](#)
- score\_Team\_2\_
  - ScoreSystemCtrl, [37](#)
- ScoreSystem/Adafruit\_MCP23008.cpp, [63](#)
- ScoreSystem/Button.cpp, [64](#)
- ScoreSystem/Button\_Driver/button\_drv.c, [64](#)
- ScoreSystem/Button\_Driver/button\_drv.mod.c, [71](#)
- ScoreSystem/I2C\_reg.cpp, [75](#)
- ScoreSystem/LCD.cpp, [76](#)
- ScoreSystem/Page.cpp, [78](#)
- ScoreSystem/ScoreSystemCtrl.cpp, [78](#)
- ScoreSystem/Screen-ButtonTestCode/Adafruit\_MC↵
  - P23008.cpp, [64](#)
- ScoreSystem/Screen-ButtonTestCode/I2C\_reg.cpp, [75](#)
- ScoreSystem/Screen-ButtonTestCode/LCD.cpp, [77](#)
- ScoreSystem/Screen-ButtonTestCode/Timer.cpp, [76](#)
- ScoreSystem/Screen-ButtonTestCode/host/files/main.↵
  - d, [72](#)
- ScoreSystem/Screen-ButtonTestCode/main.cpp, [74](#)
- ScoreSystem/Screen-ButtonTestCode/target/files/↵
  - Adafruit\_MCP23008.d, [78](#)



ScoreSystem/Screen-ButtonTestCode/target/files/I2C↔  
     \_reg.d, 78  
 ScoreSystem/Screen-ButtonTestCode/target/files/LC↔  
     D.d, 78  
 ScoreSystem/Screen-ButtonTestCode/target/files/↔  
     Timer.d, 78  
 ScoreSystem/Screen-ButtonTestCode/target/files/main.↔  
     d, 72  
 ScoreSystem/ScreenTestCode/Adafruit\_MCP23008.↔  
     cpp, 64  
 ScoreSystem/ScreenTestCode/I2C\_reg.cpp, 76  
 ScoreSystem/ScreenTestCode/LCD.cpp, 77  
 ScoreSystem/ScreenTestCode/Timer.cpp, 76  
 ScoreSystem/ScreenTestCode/main.cpp, 74  
 ScoreSystem/TimeTestCode/main.cpp, 75  
 ScoreSystem/Timer.cpp, 76  
 ScoreSystem/WebsiteScoreHandling.cpp, 78  
 ScoreSystem/buttonTestCode/host/files/main.d, 72  
 ScoreSystem/buttonTestCode/main.cpp, 72  
 ScoreSystem/buttonTestCode/target/files/main.d, 72  
 ScoreSystem/i2cTestCode/I2C\_reg.cpp, 75  
 ScoreSystem/i2cTestCode/Timer.cpp, 76  
 ScoreSystem/i2cTestCode/main.cpp, 73  
 ScoreSystem/i2cTestCode/test.cpp, 76  
 ScoreSystem/main.cpp, 73  
 ScoreSystemCtrl, 32  
     btn, 35  
     buttonThread, 35  
     collectiveDoubleShots\_, 36  
     currentScreen, 36  
     display, 36  
     endGame, 34  
     gameTime\_, 36  
     getIP, 34  
     getMsgQueue, 34  
     handleMsg, 34  
     handlePsocUpdate, 34  
     handleState, 35  
     i2cThread, 36  
     ip\_, 36  
     mq\_, 36  
     nr\_Cups\_In\_Game\_, 36  
     pages\_, 37  
     reArranging\_, 37  
     reArrangingZone\_, 37  
     resetGame, 35  
     run, 35  
     score\_Team\_1\_, 37  
     score\_Team\_2\_, 37  
     ScoreSystemCtrl, 33  
     state\_, 37  
     teamName1\_, 37  
     teamName2\_, 38  
     tempString\_, 38  
     websitePtr\_, 38  
     zone1State\_, 38  
     zone2State\_, 38  
 ScoreSystemCtrl.hpp  
     pSocMessages, 61  
 ScoreSystemMQ\_  
     I2C\_reg, 21  
 scoreTeam1\_  
     WebsiteScoreHandling, 49  
 scoreTeam2\_  
     WebsiteScoreHandling, 49  
 Screen-ButtonTestCode/LCD.cpp  
     BV, 77  
 Screen-ButtonTestCode/main.cpp  
     main, 74  
 screenAdress\_  
     I2C\_reg, 21  
 ScreenTestCode/LCD.cpp  
     BV, 78  
 ScreenTestCode/main.cpp  
     main, 74  
 seconds\_  
     WebsiteScoreHandling::TIMESTRUCT, 42  
 selectingChar  
     Page, 31  
 send  
     LCD, 26  
 sendArduinoMessage  
     I2C\_reg, 17  
 sendPsoc1Message  
     I2C\_reg, 18  
 sendPsoc2Message  
     I2C\_reg, 18  
 sendPsocBroadcast  
     I2C\_reg, 18  
 sendPsocMessage  
     I2C\_reg, 18  
 setArduinoMessage  
     I2C\_reg, 18  
 setCursor  
     LCD, 26  
 setDate  
     WebsiteScoreHandling, 45  
 setEndTime  
     WebsiteScoreHandling, 45  
 setGameTime  
     WebsiteScoreHandling, 46  
 setMsgQueueScoreSystem  
     I2C\_reg, 18  
 setNewIPJS  
     WebsiteScoreHandling.cpp, 79  
     WebsiteScoreHandling.h, 63  
 setPsocMessage  
     I2C\_reg, 19  
 setScoreTeam  
     WebsiteScoreHandling, 46  
 setStartTime  
     WebsiteScoreHandling, 46  
 setTeamName  
     WebsiteScoreHandling, 46  
 setTotalDoubleCupShots  
     WebsiteScoreHandling, 46

- setUP
  - I2C\_reg, 19
- state\_
  - I2C\_reg, 21
  - ScoreSystemCtrl, 37
- stopPsocPolling
  - I2C\_reg, 19
- stopTimer
  - Timer, 40
- stringWrite
  - LCD, 26
- t
  - WebsiteScoreHandling, 50
- teamEnter\_
  - Page, 31
- teamName1\_
  - ScoreSystemCtrl, 37
  - WebsiteScoreHandling, 50
- teamName2\_
  - ScoreSystemCtrl, 38
  - WebsiteScoreHandling, 50
- teamNameArr
  - Page, 31
- tempString\_
  - ScoreSystemCtrl, 38
- Test, 39
  - run, 39
- testStatic
  - I2C\_reg, 19
- timeEnd\_
  - WebsiteScoreHandling, 50
- timeKeeper\_
  - WebsiteScoreHandling, 50
- timeStart\_
  - WebsiteScoreHandling, 50
- timeStartDate\_
  - WebsiteScoreHandling, 50
- TimeTestCode/main.cpp
  - main, 75
- timeout\_
  - Timer, 41
- Timer, 39
  - ~Timer, 40
  - id\_, 40
  - mq\_, 41
  - run, 40
  - running\_, 41
  - stopTimer, 40
  - timeout\_, 41
  - Timer, 40
- timer1\_
  - I2C\_reg, 22
- totalDoubleCupShots\_
  - WebsiteScoreHandling, 50
- tt\_
  - I2C\_reg, 22
- type\_
  - cursorCoord, 13
- unlockI2C
  - I2C\_reg, 19
- val\_
  - psocUpdateMessage, 32
- value
  - Button, 11
  - button\_drv.c, 71
- valueRead
  - button\_drv.c, 71
- websitePtr\_
  - ScoreSystemCtrl, 38
- WebsiteScoreHandling, 42
  - currentGameID, 47
  - currentLine, 47
  - date\_, 48
  - errorFs\_, 48
  - fs\_, 48
  - gameIDInteger, 48
  - gameTime\_, 48
  - getCurrentID, 44
  - getDate, 44
  - getEndTime, 44
  - getGameTime, 44
  - getScoreTeam, 45
  - getStartTime, 45
  - getStartTimeDate, 45
  - getTeamName, 45
  - getTotalDoubleCupShots, 45
  - headers, 48
  - ifs\_, 48
  - lastGameID, 49
  - lineNumber, 49
  - lineToCopy, 49
  - lineToRead, 49
  - newName, 49
  - oldName, 49
  - scoreTeam1\_, 49
  - scoreTeam2\_, 49
  - setDate, 45
  - setEndTime, 45
  - setGameTime, 46
  - setScoreTeam, 46
  - setStartTime, 46
  - setTeamName, 46
  - setTotalDoubleCupShots, 46
  - t, 50
  - teamName1\_, 50
  - teamName2\_, 50
  - timeEnd\_, 50
  - timeKeeper\_, 50
  - timeStart\_, 50
  - timeStartDate\_, 50
  - totalDoubleCupShots\_, 50
  - WebsiteScoreHandling, 43
  - writeToCSV, 47
- WebsiteScoreHandling.cpp
  - setNewIPJS, 79

- WebsiteScoreHandling.h
  - setNewIPJS, [63](#)
- WebsiteScoreHandling::DATESTRUCT, [13](#)
  - day\_, [13](#)
  - month\_, [14](#)
  - year\_, [14](#)
- WebsiteScoreHandling::TIMESTRUCT, [41](#)
  - hours\_, [41](#)
  - minutes\_, [41](#)
  - seconds\_, [42](#)
- write8
  - Adafruit\_MCP23008, [9](#)
- writeGPIO
  - Adafruit\_MCP23008, [9](#)
- writeToCSV
  - WebsiteScoreHandling, [47](#)
- x
  - buttonMessage, [12](#)
- year\_
  - WebsiteScoreHandling::DATESTRUCT, [14](#)
- zone1State\_
  - ScoreSystemCtrl, [38](#)
- zone2State\_
  - ScoreSystemCtrl, [38](#)