Alpaca Power Pong

Generated by Doxygen 1.8.13

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

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 avaliable, 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-arroy, 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-arroy, 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 ()

  ThreadFunctor 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 buttom 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 ScoreSystemCrtl Class Reference

```
#include <ScoreSystemCrtl.hpp>
```

Inherits ThreadFunctor.

Public Member Functions

- ScoreSystemCrtl ()
- 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 ScoreSystemCrtl()

```
ScoreSystemCrtl::ScoreSystemCrtl ( )
```

----------— DEFAULT CONSTRUCTOR ----------— ------— SETS UP VARIABLES, LOADS IN PAGES ----— – STAR↩
TING I2C AND BUTTON THREADS AND STARTS THE DISPLAY –

4.10.2 Member Function Documentation

4.10.2.1 endGame()

```
void ScoreSystemCrtl::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 ScoreSystemCrtl::getIP ( )  [private]
```

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

4.10.2.3 getMsgQueue()

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

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

4.10.2.4 handleMsg()

```
void ScoreSystemCrtl::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 ScoreSystemCrtl::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 retrived 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 untill a cupZoneReady state is received.

The teams scores are also updated in this function.

4.10.2.6 handleState()

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

----—— HANDLE STATE -----—— -— HANDLES STATES RETURNED BY OUR PAGES -— Handles the state received from the buttonPressed function. It then takes action accouring 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 ScoreSystemCrtl::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 ScoreSystemCrtl::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* ScoreSystemCrtl::btn  [private]
```

4.10.3.2 buttonThread

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

4.10.3.3 collectiveDoubleShots_

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

4.10.3.4 currentScreen

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

4.10.3.5 display

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

4.10.3.6 gameTime_

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

4.10.3.7 i2cThread

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

4.10.3.8 ip_

```
string ScoreSystemCrtl::ip_  [private]
```

4.10.3.9 mq_

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

4.10.3.10 nr_Cups_In_Game_

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

4.10.3.11 pages_

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

4.10.3.12 reArranging_

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

4.10.3.13 reArrangingZone_

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

4.10.3.14 score_Team_1_

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

DEFAULT VALUES.

4.10.3.15 score_Team_2_

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

4.10.3.16 state_

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

4.10.3.17 teamName1_

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

4.10.3.18 teamName2_

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

DEFAULT VALUES.

4.10.3.19 tempString_

```
string ScoreSystemCrtl::tempString_  [private]
```

4.10.3.20 websitePtr_

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

4.10.3.21 zone1State_

```
enum pSocMessages ScoreSystemCrtl::zone1State_ = pSocMessages::NO_CHANGE  [private]
```

DEFAULT VALUES.

4.10.3.22 zone2State_

```
enum pSocMessages ScoreSystemCrtl::zone2State_ = pSocMessages::NO_CHANGE  [private]
```

DEFAULT VALUES.

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

- inc/osapi/ScoreSystem/ScoreSystemCrtl.hpp
- ScoreSystem/ScoreSystemCrtl.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 score←Team2, 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 ScoreSystemCrtl. 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 ScoreSystemCrtl. 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/ScoreSystemCrtl.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 ScoreSystemCrtl

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/ScoreSystemCrtl.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/ScoreSystemCrtl.cpp File Reference

```
#include <osapi/ScoreSystem/ScoreSystemCrtl.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