Agario Clone

Generated by Doxygen 1.8.13

Оглавление

Глава 1

Bublle Game

Author

Bohdan Buinich

Version

1.0

Date

2018-08-25

Warning

use only good wire for I2C interface

1.1 Introduction

This code was developed to GLBaseCamp

Глава 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Глава 3

File Documentation

3.1 /home/bohdan/agario_clone/i2cmaster.h File Reference

#include <avr/io.h>
Include dependency graph for i2cmaster.h:

3.2 /home/bohdan/agario_clone/lcd.c File Reference

#include "lcd.h"
#include <string.h>
#include "i2cmaster.h"
#include "myFont.h"
Include dependency graph for lcd.c:

Functions

- void setup_i2c ()
- void drawPixel (int16_t x, int16_t y)
- void clearBuffer (uint8_t *buff)
- void drawCircle (int16_t x0, int16_t y0, int16_t r)
- void drawBuffer (uint8_t column_address, uint8_t page_address, uint8_t *buff)
- void lcd_draw_char (unsigned char column, unsigned char page, unsigned char letter, uint8_t *buff)
- void lcd_draw_string (uint8_t column, uint8_t page, char *string, uint8_t *buff)

3.2.1 Function Documentation

3.2.1.1 clearBuffer()

void clearBuffer (
uint8_t * buff)

3.2.1.2 drawBuffer()

```
void drawBuffer (
                uint8_t column_address,
                uint8_t page_address,
                uint8_t * buff )
```

3.2.1.3 drawCircle()

```
void drawCircle (
                int16_t x0,
                int16_t y0,
                int16_t r )
```

3.2.1.4 drawPixel()

```
void drawPixel (
                int16_t x,
                int16_t y )
```

3.2.1.5 lcd_draw_char()

```
void lcd_draw_char (
                unsigned char column,
                unsigned char page,
                unsigned char letter,
                uint8_t * buff )
```

3.2.1.6 lcd_draw_string()

```
void lcd_draw_string (
                uint8_t column,
                uint8_t page,
                char * string,
                uint8_t * buff )
```

3.2.1.7 setup_i2c()

```
void setup_i2c ( )
```

3.3 /home/bohdan/agario_clone/lcd.h File Reference

#include <stdint.h>
Include dependency graph for lcd.h: This graph shows which files directly or indirectly include this file:

Macros

- #define DevSSD1306 0x78
- #define SSD1306_DUTY_CYCLE_1_64 0x3F
- #define SSD1306_FLIPS_DISPLAY 0xCF
- #define SSD1306_SETCONTRAST 0x81
- #define SSD1306_DISPLAYALLON_RESUME 0xA4
- #define SSD1306_DISPLAYALLON 0xA5
- #define SSD1306_NORMALDISPLAY 0xA6
- #define SSD1306_INVERTDISPLAY 0xA7
- #define SSD1306_DISPLAYOFF 0xAE
- #define SSD1306_DISPLAYON 0xAF
- #define SSD1306_SETDISPLAYOFFSET 0xD3
- #define SSD1306_SETCOMPINS 0xDA
- #define SSD1306_SETVCOMDETECT 0xDB
- #define SSD1306_SETDISPLAYCLOCKDIV 0xD5
- #define SSD1306_SETPRECHARGE 0xD9
- #define SSD1306_SETMULTIPLEX 0xA8
- #define SSD1306_SETLOWCOLUMN 0x00
- #define SSD1306_SETHIGHCOLUMN 0x10
- #define SSD1306_SETSTARTLINE 0x40
- #define SSD1306_MEMORYMODE 0x20
- #define SSD1306_COLUMNADDR 0x21
- #define SSD1306_PAGEADDR 0x22
- #define SSD1306_COMSCANINC 0xC0
- #define SSD1306_COMSCANDEC 0xC8
- #define SSD1306_SEGREMAP 0xA1
- #define SSD1306_CHARGEPUMP 0x8D
- #define SSD1306_SWITCHCAPVCC 0x2
- #define SSD1306_NOP 0xE3
- #define SSD1306_NO_OFFSET 0x0
- #define SSD1306_RESISTOR_RATIO 0x80
- #define SSD1306_WIDTH 128
- #define SSD1306_HEIGHT 64

Functions

- void drawPixel (int16_t x, int16_t y)
- void clearBuffer (uint8_t *buff)
- void drawCircle (int16_t x0, int16_t y0, int16_t r)
- void drawBuffer (uint8_t column_address, uint8_t page_address, uint8_t *buff)
- void lcd_draw_char (unsigned char column, unsigned char page, unsigned char letter, uint8_t *buff)
- void lcd_draw_string (uint8_t column, uint8_t page, char *string, uint8_t *buff)
- void setup_i2c ()

Variables

- uint8_t buffer [(128 ∗64)/8]

3.3.1 Macro Definition Documentation

3.3.1.1 DevSSD1306

#define DevSSD1306 0x78

3.3.1.2 SSD1306_CHARGEPUMP

#define SSD1306_CHARGEPUMP 0x8D

3.3.1.3 SSD1306_COLUMNADDR

#define SSD1306_COLUMNADDR 0x21

3.3.1.4 SSD1306_COMSCANDEC

#define SSD1306_COMSCANDEC 0xC8

3.3.1.5 SSD1306_COMSCANINC

#define SSD1306_COMSCANINC 0xC0

3.3.1.6 SSD1306_DISPLAYALLON

#define SSD1306_DISPLAYALLON 0xA5

3.3.1.7 SSD1306_DISPLAYALLON_RESUME

```
#define SSD1306_DISPLAYALLON_RESUME 0xA4
```

3.3.1.8 SSD1306_DISPLAYOFF

```
#define SSD1306_DISPLAYOFF 0xAE
```

3.3.1.9 SSD1306_DISPLAYON

```
#define SSD1306_DISPLAYON 0xAF
```

3.3.1.10 SSD1306_DUTY_CYCLE_1_64

```
#define SSD1306_DUTY_CYCLE_1_64 0x3F
```

3.3.1.11 SSD1306_FLIPS_DISPLAY

```
#define SSD1306_FLIPS_DISPLAY 0xCF
```

3.3.1.12 SSD1306_HEIGHT

```
#define SSD1306_HEIGHT 64
```

3.3.1.13 SSD1306_INVERTDISPLAY

```
#define SSD1306_INVERTDISPLAY 0xA7
```

3.3.1.14 SSD1306_MEMORYMODE

```
#define SSD1306_MEMORYMODE 0x20
```

3.3.1.15 SSD1306_NO_OFFSET

```
#define SSD1306_NO_OFFSET 0x0
```

3.3.1.16 SSD1306_NOP

```
#define SSD1306_NOP 0xE3
```

3.3.1.17 SSD1306_NORMALDISPLAY

```
#define SSD1306_NORMALDISPLAY 0xA6
```

3.3.1.18 SSD1306_PAGEADDR

```
#define SSD1306_PAGEADDR 0x22
```

3.3.1.19 SSD1306_RESISTOR_RATIO

```
#define SSD1306_RESISTOR_RATIO 0x80
```

3.3.1.20 SSD1306_SEGREMAP

```
#define SSD1306_SEGREMAP 0xA1
```

3.3.1.21 SSD1306_SETCOMPINS

```
#define SSD1306_SETCOMPINS 0xDA
```

3.3.1.22 SSD1306_SETCONTRAST

```
#define SSD1306_SETCONTRAST 0x81
```

3.3.1.23 SSD1306_SETDISPLAYCLOCKDIV

```
#define SSD1306_SETDISPLAYCLOCKDIV 0xD5
```

3.3.1.24 SSD1306_SETDISPLAYOFFSET

```
#define SSD1306_SETDISPLAYOFFSET 0xD3
```

3.3.1.25 SSD1306_SETHIGHCOLUMN

```
#define SSD1306_SETHIGHCOLUMN 0x10
```

3.3.1.26 SSD1306_SETLOWCOLUMN

```
#define SSD1306_SETLOWCOLUMN 0x00
```

3.3.1.27 SSD1306_SETMULTIPLEX

```
#define SSD1306_SETMULTIPLEX 0xA8
```

3.3.1.28 SSD1306_SETPRECHARGE

```
#define SSD1306_SETPRECHARGE 0xD9
```

3.3.1.29 SSD1306_SETSTARTLINE

```
#define SSD1306_SETSTARTLINE 0x40
```

3.3.1.30 SSD1306_SETVCOMDETECT

```
#define SSD1306_SETVCOMDETECT 0xDB
```

3.3.1.31 SSD1306_SWITCHCAPVCC

```
#define SSD1306_SWITCHCAPVCC 0x2
```

3.3.1.32 SSD1306_WIDTH

```
#define SSD1306_WIDTH 128
```

3.3.2 Function Documentation

3.3.2.1 clearBuffer()

```
void clearBuffer (
                uint8_t * buff )
```

3.3.2.2 drawBuffer()

```
void drawBuffer (
                uint8_t column_address,
                uint8_t page_address,
                uint8_t * buff )
```

3.3.2.3 drawCircle()

```
void drawCircle (
                int16_t x0,
                int16_t y0,
                int16_t r )
```

3.3.2.4 drawPixel()

```
void drawPixel (
                int16_t x,
                int16_t y )
```

3.3.2.5 lcd_draw_char()

```
void lcd_draw_char (
                unsigned char column,
                unsigned char page,
                unsigned char letter,
                uint8_t * buff )
```

3.3.2.6 lcd_draw_string()

```
void lcd_draw_string (
                uint8_t column,
                uint8_t page,
                char * string,
                uint8_t * buff )
```

3.3.2.7 setup_i2c()

```
void setup_i2c ( )
```

3.3.3 Variable Documentation

3.3.3.1 buffer

```
uint8_t buffer[(128 *64)/8]
```

3.4 /home/bohdan/agario_clone/myFont.c File Reference

Variables

- unsigned char Ascii_1 [97][5]

3.4.1 Variable Documentation

3.4.1.1 Ascii_1

```
unsigned char Ascii_1[97][5]
```

3.5 /home/bohdan/agario_clone/myFont.h File Reference

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

Variables

- unsigned char Ascii_1 [97][5]

3.5.1 Variable Documentation

3.5.1.1 Ascii_1

unsigned char Ascii_1[97][5]

3.6 /home/bohdan/agario_clone/test_i2cmaster.c File Reference

#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <util/atomic.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include "i2cmaster.h"
#include "lcd.h"
Include dependency graph for test_i2cmaster.c:

Macros

- #define max(a, b) ({typeof (a) _a = (a); typeof (b) _b = (b); _a > _b ? _a : _b; })
  A macro that returns the maximum of a and b.
- #define min(a, b) ({typeof (a) _a = (a); typeof (b) _b = (b); _a < _b ? _a : _b; })
  A macro that returns the minimum of a and b.
- #define constrain(v, lo, hi) (max(min(v, hi), lo))
  A macro constrains a number to be within a range.
- #define CTC_MATCH_OVERFLOW ((F_CPU / 1000) / 8)
  Calculate the value needed for the CTC match value in OCR1A.
- #define playAREAX 512
  X-axis game area.
- #define playAREAY 512
  Y-axis game area.
- #define OLEDX 128
  Oled size on the X-axis.
- #define OLEDY 64
  Oled size on the Y-axis.
- #define enemyCOUNT 30
  Enemy count in the game.
- #define particleCOUNT 10
  Particle count in the game.
- #define DevSSD1306 0x78
  device address of SSD1306 OLED, uses 8-bit address (0x3c)!!!

Functions

- void Start ()
  Start function.
- ISR (INT0_vect)
  Interrupt on the button for start game.
- void init_TIMER ()
- ISR (TIMER1_COMPA_vect)
  Interrupt on timer, add 1 for millils.
- unsigned long millis ()
  Return millis.
- void init_ADC ()
- void ADC_data ()
  Return ADC value.
- void init_interrupt ()
- uint32_t map (uint32_t x, uint32_t in_min, uint32_t in_max, uint32_t out_min, uint32_t out_max)
  Map function.
- uint16_t makeRandom (uint16_t upper)
  makeRandom function
- uint16_t checkCONTACT (uint8_t spriteNUM, uint8_t spriteX, uint8_t spriteY, uint8_t spriteR)
  Functiom check contact with enemy sprite.
- int checkPCONTACT (uint8_t spriteNUM, uint8_t spriteX, uint8_t spriteY)
  Functiom check contact with particle sprite.
- int randint (uint16_t min, uint16_t max)
  Functiom get random integers in a range min to max.
- void deadANIMATION ()
  Functiom dead Animation when player lose game.
- void enemyDEAD (uint8_t enemyNUM)
  Functiom ememy dead.
- void youWIN ()
  Functiom win game.
- void gameMode ()
  Game algoritm.
- int main (void)

Variables

- volatile unsigned long timer1_millis
- long milliseconds_since
- uint16_t XN = 0
  Joystick position on the X-axis.
- uint16_t YN = 0
  Joystick position on the Y-axis.
- uint8_t charR = 5
  Size of the Character.
- uint8_t charX = 128
  Character location X.
- uint8_t charY = 128
  Character location Y.

- float charACCX = 0

  Character Momentum X.
- float charACCY = 0

  Character Momentum Y.
- uint16_t enemySTAT [enemyCOUNT][5]

  enemy location and stats
- uint16_t JOYXPOS = 0
- uint16_t JOYYPOS = 0
- uint16_t particle [particleCOUNT][3]
- float friction = 0.05
- uint8_t eatPART = 0

  Count of the eaten elements.
- uint8_t buffer [(128 ∗64)/8]
- uint8_t gameStart = 0

3.6.1 Macro Definition Documentation

3.6.1.1 constrain

```
#define constrain(
                v,
                lo,
                hi ) (max(min(v, hi), lo))
```

A macro constrains a number to be within a range.

Parameters

v	The number to constrain, all data types.
lo	The lower end of the range, all data types.
hi	The upper end of the range, all data types.

3.6.1.2 CTC_MATCH_OVERFLOW

```
#define CTC_MATCH_OVERFLOW ((F_CPU / 1000) / 8)
```

Calculate the value needed for the CTC match value in OCR1A.

3.6.1.3 DevSSD1306

```
#define DevSSD1306 0x78
```

device address of SSD1306 OLED, uses 8-bit address (0x3c)!!!

3.6.1.4 enemyCOUNT

```
#define enemyCOUNT 30
```

Enemy count in the game.

3.6.1.5 max

```
#define max(
        a,
        b ) ({typeof (a) _a = (a); typeof (b) _b = (b); _a > _b ? _a : _b; })
```

A macro that returns the maximum of a and b.

3.6.1.6 min

```
#define min(
        a,
        b ) ({typeof (a) _a = (a); typeof (b) _b = (b); _a < _b ? _a : _b; })
```

A macro that returns the minimum of a and b.

3.6.1.7 OLEDX

```
#define OLEDX 128
```

Oled size on the X-axis.

3.6.1.8 OLEDY

```
#define OLEDY 64
```

Oled size on the Y-axis.

3.6.1.9 particleCOUNT

```
#define particleCOUNT 10
```

Particle count in the game.

3.6.1.10 playAREAX

#define playAREAX 512

X-axis game area.

3.6.1.11 playAREAY

#define playAREAY 512

Y-axis game area.

3.6.2 Function Documentation

3.6.2.1 ADC_data()

void ADC_data ( )

Return ADC value.

Parameters

XN,value	from A0
YN,value	from A1

3.6.2.2 checkCONTACT()

uint16_t checkCONTACT (
uint8_t spriteNUM,
uint8_t spriteX,
uint8_t spriteY,
uint8_t spriteR)

Functiom check contact with enemy sprite.

Parameters

spriteNUM	number enemy sprite
spriteX	position on the X-axis
spriteY	position on the Y-axis
spriteR	radius

Return values

0	if no contact
1	if contact

3.6.2.3 checkPCONTACT()

```
int checkPCONTACT (
                uint8_t spriteNUM,
                uint8_t spriteX,
                uint8_t spriteY )
```

Functiom check contact with particle sprite.

Parameters

spriteNUM	number enemy sprite
spriteX	position on the X-axis
spriteY	position on the Y-axis

Return values

0	if no contact
1	if contact

3.6.2.4 deadANIMATION()

```
void deadANIMATION ( )
```

Functiom dead Animation when player lose game.

3.6.2.5 enemyDEAD()

```
void enemyDEAD (
                uint8_t enemyNUM )
```

Functiom ememy dead.

Parameters

enemyNUM	numer dead enemy

3.6.2.6 gameMode()

void gameMode ( )

Game algoritm.

3.6.2.7 init_ADC()

void init_ADC ( )

3.6.2.8 init_interrupt()

void init_interrupt ( )

3.6.2.9 init_TIMER()

void init_TIMER ( )

3.6.2.10 ISR() [1/2]

ISR (
INT0_vect)

Interrupt on the button for start game.

See also

Start()

Parameters

INT0_vect	

3.6.2.11 ISR() [2/2]

```
ISR (
        TIMER1_COMPA_vect  )
```

Interrupt on timer, add 1 for millils.

Parameters

TIMER1_COMPA_vect	

3.6.2.12 main()

```
int main (
        void  )
```

3.6.2.13 makeRandom()

```
uint16_t makeRandom (
        uint16_t upper )
```

makeRandom function

Functiom get random integers in a certain range

Parameters

upper	value in which range must be new random value

Returns

Random integers in a certain range

3.6.2.14 map()

```
uint32_t map (
        uint32_t x,
        uint32_t in_min,
        uint32_t in_max,
        uint32_t out_min,
        uint32_t out_max )
```

Map function.

Re-maps a number from one range to another. That is, a value of fromLow would get mapped to toLow, a value of fromHigh to toHigh, values in-between to values in-between, etc.

Parameters

x	The number to map.
in_min	The lower bound of the value's current range.
in_max	The upper bound of the value's current range.
out_min	The lower bound of the value's target range.
out_max	The upper bound of the value's target range.

Returns

Re-maps value

3.6.2.15 millis()

```
unsigned long millis (
        void  )
```

Return millis.

Returns

millis

3.6.2.16 randint()

```
int randint (
        uint16_t min,
        uint16_t max )
```

Functiom get random integers in a range min to max.

Parameters

max	value in which range must be new random value
min	value in which range must be new random value

Returns

Random integers in a range

3.6.2.17 Start()

void Start ( )

Start function.

See also

Start()

3.6.2.18 youWIN()

void youWIN ( )

Functiom win game.

3.6.3 Variable Documentation

3.6.3.1 buffer

uint8_t buffer[(128 ∗64)/8]

3.6.3.2 charACCX

float charACCX = 0

Character Momentum X.

3.6.3.3 charACCY

float charACCY = 0

Character Momentum Y.

3.6.3.4 charR

uint8_t charR = 5

Size of the Character.

3.6.3.5 charX

uint8_t charX = 128

Character location X.

3.6.3.6 charY

uint8_t charY = 128

Character location Y.

3.6.3.7 eatPART

uint8_t eatPART = 0

Count of the eaten elements.

3.6.3.8 enemySTAT

uint16_t enemySTAT[enemyCOUNT][5]

enemy location and stats

3.6.3.9 friction

float friction = 0.05

3.6.3.10 gameStart

uint8_t gameStart = 0

3.6.3.11 JOYXPOS

uint16_t JOYXPOS = 0

3.6.3.12 JOYYPOS

uint16_t JOYYPOS = 0

3.6.3.13 milliseconds_since

long milliseconds_since

3.6.3.14 particle

uint16_t particle[particleCOUNT][3]

3.6.3.15 timer1_millis

volatile unsigned long timer1_millis

3.6.3.16 XN

uint16_t XN = 0

Joystick position on the X-axis.

3.6.3.17 YN

uint16_t YN = 0

Joystick position on the Y-axis.

3.7 /home/bohdan/agario_clone/twimaster.c File Reference

#include <inttypes.h>
#include <compat/twi.h>
#include "i2cmaster.h"
Include dependency graph for twimaster.c:

Macros

- #define SCL_CLOCK 400000L

Functions

- void i2c_init (void)
  initialize the I2C master interace. Need to be called only once
- unsigned char i2c_start (unsigned char address)
  Issues a start condition and sends address and transfer direction.
- void i2c_start_wait (unsigned char address)
  Issues a start condition and sends address and transfer direction.
- unsigned char i2c_rep_start (unsigned char address)
  Issues a repeated start condition and sends address and transfer direction.
- void i2c_stop (void)
  Terminates the data transfer and releases the I2C bus.
- unsigned char i2c_write (unsigned char data)
  Send one byte to I2C device.
- unsigned char i2c_readAck (void)
  read one byte from the I2C device, request more data from device
- unsigned char i2c_readNak (void)
  read one byte from the I2C device, read is followed by a stop condition

3.7.1 Macro Definition Documentation

3.7.1.1 SCL_CLOCK

#define SCL_CLOCK 400000L

3.7.2 Function Documentation

3.7.2.1 i2c_init()

void i2c_init (
void)

initialize the I2C master interace. Need to be called only once

Parameters

void	

Returns

none

3.7.2.2 i2c_readAck()

```
unsigned char i2c_readAck (
                void  )
```

read one byte from the I2C device, request more data from device

Returns

byte read from I2C device

3.7.2.3 i2c_readNak()

```
unsigned char i2c_readNak (
                void  )
```

read one byte from the I2C device, read is followed by a stop condition

Returns

byte read from I2C device

3.7.2.4 i2c_rep_start()

```
unsigned char i2c_rep_start (
                unsigned char addr )
```

Issues a repeated start condition and sends address and transfer direction.

Parameters

addr	address and transfer direction of I2C device

Return values

0	device accessible
1	failed to access device

3.7.2.5 i2c_start()

```
unsigned char i2c_start (
                unsigned char addr )
```

Issues a start condition and sends address and transfer direction.

Parameters

addr	address and transfer direction of I2C device

Return values

0	device accessible
1	failed to access device

3.7.2.6 i2c_start_wait()

```
void i2c_start_wait (
                unsigned char addr )
```

Issues a start condition and sends address and transfer direction.

If device is busy, use ack polling to wait until device ready

Parameters

addr	address and transfer direction of I2C device

Returns

none

3.7.2.7 i2c_stop()

```
void i2c_stop (
                void  )
```

Terminates the data transfer and releases the I2C bus.

Parameters

void	

Returns

none

3.7.2.8 i2c_write()

```
unsigned char i2c_write (
                unsigned char data )
```

Send one byte to I2C device.

Parameters

data	byte to be transfered

Return values

0	write successful
1	write failed

Предметный указатель