

Agario Clone

Generated by Doxygen 1.8.13



# Оглавление

1	Bublle Game	1
1.1	Introduction . . . . .	1
2	File Index	3
2.1	File List . . . . .	3
3	File Documentation	5
3.1	/home/bohdan/agario_clone/i2cmaster.h File Reference . . . . .	5
3.2	/home/bohdan/agario_clone/lcd.c File Reference . . . . .	5
3.2.1	Function Documentation . . . . .	5
3.2.1.1	clearBuffer() . . . . .	5
3.2.1.2	drawBuffer() . . . . .	6
3.2.1.3	drawCircle() . . . . .	6
3.2.1.4	drawPixel() . . . . .	6
3.2.1.5	lcd_draw_char() . . . . .	6
3.2.1.6	lcd_draw_string() . . . . .	6
3.2.1.7	setup_i2c() . . . . .	6
3.3	/home/bohdan/agario_clone/lcd.h File Reference . . . . .	7
3.3.1	Macro Definition Documentation . . . . .	8
3.3.1.1	DevSSD1306 . . . . .	8
3.3.1.2	SSD1306_CHARGEUMP . . . . .	8
3.3.1.3	SSD1306_COLUMNADDR . . . . .	8
3.3.1.4	SSD1306_COMSCANDEC . . . . .	8
3.3.1.5	SSD1306_COMSCANINC . . . . .	8

3.3.1.6	SSD1306_DISPLAYALLON . . . . .	8
3.3.1.7	SSD1306_DISPLAYALLON_RESUME . . . . .	9
3.3.1.8	SSD1306_DISPLAYOFF . . . . .	9
3.3.1.9	SSD1306_DISPLAYON . . . . .	9
3.3.1.10	SSD1306_DUTY_CYCLE_1_64 . . . . .	9
3.3.1.11	SSD1306_FLIPS_DISPLAY . . . . .	9
3.3.1.12	SSD1306_HEIGHT . . . . .	9
3.3.1.13	SSD1306_INVERTDISPLAY . . . . .	9
3.3.1.14	SSD1306_MEMORYMODE . . . . .	9
3.3.1.15	SSD1306_NO_OFFSET . . . . .	10
3.3.1.16	SSD1306_NOP . . . . .	10
3.3.1.17	SSD1306_NORMALDISPLAY . . . . .	10
3.3.1.18	SSD1306_PAGEADDR . . . . .	10
3.3.1.19	SSD1306_RESISTOR_RATIO . . . . .	10
3.3.1.20	SSD1306_SEGREMAP . . . . .	10
3.3.1.21	SSD1306_SETCOMPINS . . . . .	10
3.3.1.22	SSD1306_SETCONTRAST . . . . .	10
3.3.1.23	SSD1306_SETDISPLAYCLOCKDIV . . . . .	11
3.3.1.24	SSD1306_SETDISPLAYOFFSET . . . . .	11
3.3.1.25	SSD1306_SETHIGHCOLUMN . . . . .	11
3.3.1.26	SSD1306_SETLOWCOLUMN . . . . .	11
3.3.1.27	SSD1306_SETMULTIPLEX . . . . .	11
3.3.1.28	SSD1306_SETPRECHARGE . . . . .	11
3.3.1.29	SSD1306_SETSTARTLINE . . . . .	11
3.3.1.30	SSD1306_SETVCOMDETECT . . . . .	11
3.3.1.31	SSD1306_SWITCHCAPVCC . . . . .	12
3.3.1.32	SSD1306_WIDTH . . . . .	12
3.3.2	Function Documentation . . . . .	12
3.3.2.1	clearBuffer() . . . . .	12
3.3.2.2	drawBuffer() . . . . .	12

3.3.2.3	<a href="#">drawCircle()</a>	12
3.3.2.4	<a href="#">drawPixel()</a>	12
3.3.2.5	<a href="#">lcd_draw_char()</a>	13
3.3.2.6	<a href="#">lcd_draw_string()</a>	13
3.3.2.7	<a href="#">setup_i2c()</a>	13
3.3.3	<a href="#">Variable Documentation</a>	13
3.3.3.1	<a href="#">buffer</a>	13
3.4	<a href="#">/home/bohdan/agario_clone/myFont.c File Reference</a>	13
3.4.1	<a href="#">Variable Documentation</a>	13
3.4.1.1	<a href="#">Ascii_1</a>	13
3.5	<a href="#">/home/bohdan/agario_clone/myFont.h File Reference</a>	14
3.5.1	<a href="#">Variable Documentation</a>	14
3.5.1.1	<a href="#">Ascii_1</a>	14
3.6	<a href="#">/home/bohdan/agario_clone/test_i2cmaster.c File Reference</a>	14
3.6.1	<a href="#">Macro Definition Documentation</a>	16
3.6.1.1	<a href="#">constrain</a>	16
3.6.1.2	<a href="#">CTC_MATCH_OVERFLOW</a>	16
3.6.1.3	<a href="#">DevSSD1306</a>	16
3.6.1.4	<a href="#">enemyCOUNT</a>	17
3.6.1.5	<a href="#">max</a>	17
3.6.1.6	<a href="#">min</a>	17
3.6.1.7	<a href="#">OLEDX</a>	17
3.6.1.8	<a href="#">OLEDY</a>	17
3.6.1.9	<a href="#">particleCOUNT</a>	17
3.6.1.10	<a href="#">playAREAX</a>	18
3.6.1.11	<a href="#">playAREAY</a>	18
3.6.2	<a href="#">Function Documentation</a>	18
3.6.2.1	<a href="#">ADC_data()</a>	18
3.6.2.2	<a href="#">checkCONTACT()</a>	18
3.6.2.3	<a href="#">checkPCONTACT()</a>	19

3.6.2.4	<code>deadANIMATION()</code>	19
3.6.2.5	<code>enemyDEAD()</code>	19
3.6.2.6	<code>gameMode()</code>	20
3.6.2.7	<code>init_ADC()</code>	20
3.6.2.8	<code>init_interrupt()</code>	20
3.6.2.9	<code>init_TIMER()</code>	20
3.6.2.10	<code>ISR()</code> [1/2]	20
3.6.2.11	<code>ISR()</code> [2/2]	21
3.6.2.12	<code>main()</code>	21
3.6.2.13	<code>makeRandom()</code>	21
3.6.2.14	<code>map()</code>	21
3.6.2.15	<code>millis()</code>	22
3.6.2.16	<code>randint()</code>	22
3.6.2.17	<code>Start()</code>	23
3.6.2.18	<code>youWIN()</code>	23
3.6.3	Variable Documentation	23
3.6.3.1	<code>buffer</code>	23
3.6.3.2	<code>charACCX</code>	23
3.6.3.3	<code>charACCY</code>	23
3.6.3.4	<code>charR</code>	24
3.6.3.5	<code>charX</code>	24
3.6.3.6	<code>charY</code>	24
3.6.3.7	<code>eatPART</code>	24
3.6.3.8	<code>enemySTAT</code>	24
3.6.3.9	<code>friction</code>	24
3.6.3.10	<code>gameStart</code>	25
3.6.3.11	<code>JOYXPOS</code>	25
3.6.3.12	<code>JOYYPOS</code>	25
3.6.3.13	<code>milliseconds_since</code>	25
3.6.3.14	<code>particle</code>	25

---

3.6.3.15	<a href="#">timer1_millis</a>	25
3.6.3.16	<a href="#">XN</a>	25
3.6.3.17	<a href="#">YN</a>	26
3.7	<a href="#">/home/bohdan/agario_clone/twimaster.c File Reference</a>	26
3.7.1	<a href="#">Macro Definition Documentation</a>	26
3.7.1.1	<a href="#">SCL_CLOCK</a>	26
3.7.2	<a href="#">Function Documentation</a>	26
3.7.2.1	<a href="#">i2c_init()</a>	26
3.7.2.2	<a href="#">i2c_readAck()</a>	27
3.7.2.3	<a href="#">i2c_readNak()</a>	27
3.7.2.4	<a href="#">i2c_rep_start()</a>	27
3.7.2.5	<a href="#">i2c_start()</a>	28
3.7.2.6	<a href="#">i2c_start_wait()</a>	28
3.7.2.7	<a href="#">i2c_stop()</a>	28
3.7.2.8	<a href="#">i2c_write()</a>	29
	<a href="#">Index</a>	31





# Глава 1

## Bublle Game

Author

Bohdan Buinich

Version

1.0

Date

2018-08-25

Warning

use only good wire for I2C interface

Copyright

GNU Public License

### 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:

/home/bohdan/agario_clone/ <a href="#">i2cmaster.h</a> . . . . .	5
/home/bohdan/agario_clone/ <a href="#">lcd.c</a> . . . . .	5
/home/bohdan/agario_clone/ <a href="#">lcd.h</a> . . . . .	7
/home/bohdan/agario_clone/ <a href="#">myFont.c</a> . . . . .	13
/home/bohdan/agario_clone/ <a href="#">myFont.h</a> . . . . .	14
/home/bohdan/agario_clone/ <a href="#">test_i2cmaster.c</a> . . . . .	14
/home/bohdan/agario_clone/ <a href="#">twimaster.c</a> . . . . .	26



## 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\_CHARGEUMP

```
#define SSD1306_CHARGEUMP 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 millis.
- 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)  
Function check contact with enemy sprite.
- int `checkPCONTACT` (uint8\_t spriteNUM, uint8\_t spriteX, uint8\_t spriteY)  
Function check contact with particle sprite.
- int `randint` (uint16\_t min, uint16\_t max)  
Function get random integers in a range min to max.
- void `deadANIMATION` ()  
Function dead Animation when player lose game.
- void `enemyDEAD` (uint8\_t enemyNUM)  
Function enemy dead.
- void `youWIN` ()  
Function win game.
- void `gameMode` ()  
Game algorithm.
- 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 )
```

Function 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 )
```

Function 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 ( )
```

Function dead Animation when player lose game.

### 3.6.2.5 enemyDEAD()

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

Function enemy 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

Function 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 )
```

Function 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 ( )

Function 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

- /home/bohdan/agario\_clone/i2cmaster.h, 5
  - /home/bohdan/agario\_clone/lcd.c, 5
  - /home/bohdan/agario\_clone/lcd.h, 7
  - /home/bohdan/agario\_clone/myFont.c, 13
  - /home/bohdan/agario\_clone/myFont.h, 14
  - /home/bohdan/agario\_clone/test\_i2cmaster.c, 14
  - /home/bohdan/agario\_clone/twimaster.c, 26
- ADC\_data
  - test\_i2cmaster.c, 18
- Ascii\_1
  - myFont.c, 13
  - myFont.h, 14
- buffer
  - lcd.h, 13
  - test\_i2cmaster.c, 23
- CTC\_MATCH\_OVERFLOW
  - test\_i2cmaster.c, 16
- charACCX
  - test\_i2cmaster.c, 23
- charACCY
  - test\_i2cmaster.c, 23
- charR
  - test\_i2cmaster.c, 23
- charX
  - test\_i2cmaster.c, 24
- charY
  - test\_i2cmaster.c, 24
- checkCONTACT
  - test\_i2cmaster.c, 18
- checkPCONTACT
  - test\_i2cmaster.c, 19
- clearBuffer
  - lcd.c, 5
  - lcd.h, 12
- constrain
  - test\_i2cmaster.c, 16
- deadANIMATION
  - test\_i2cmaster.c, 19
- DevSSD1306
  - lcd.h, 8
  - test\_i2cmaster.c, 16
- drawBuffer
  - lcd.c, 5
  - lcd.h, 12
- drawCircle
  - lcd.c, 6
- lcd.h, 12
- drawPixel
  - lcd.c, 6
  - lcd.h, 12
- eatPART
  - test\_i2cmaster.c, 24
- enemyCOUNT
  - test\_i2cmaster.c, 16
- enemyDEAD
  - test\_i2cmaster.c, 19
- enemySTAT
  - test\_i2cmaster.c, 24
- friction
  - test\_i2cmaster.c, 24
- gameMode
  - test\_i2cmaster.c, 20
- gameStart
  - test\_i2cmaster.c, 24
- i2c\_init
  - twimaster.c, 26
- i2c\_readAck
  - twimaster.c, 27
- i2c\_readNak
  - twimaster.c, 27
- i2c\_rep\_start
  - twimaster.c, 27
- i2c\_start
  - twimaster.c, 28
- i2c\_start\_wait
  - twimaster.c, 28
- i2c\_stop
  - twimaster.c, 28
- i2c\_write
  - twimaster.c, 29
- ISR
  - test\_i2cmaster.c, 20
- init\_ADC
  - test\_i2cmaster.c, 20
- init\_TIMER
  - test\_i2cmaster.c, 20
- init\_interrupt
  - test\_i2cmaster.c, 20
- JOYXPOS
  - test\_i2cmaster.c, 25
- JOYYPOS

- test\_i2cmaster.c, 25
- lcd.c
  - clearBuffer, 5
  - drawBuffer, 5
  - drawCircle, 6
  - drawPixel, 6
  - lcd\_draw\_char, 6
  - lcd\_draw\_string, 6
  - setup\_i2c, 6
- lcd.h
  - buffer, 13
  - clearBuffer, 12
  - DevSSD1306, 8
  - drawBuffer, 12
  - drawCircle, 12
  - drawPixel, 12
  - lcd\_draw\_char, 12
  - lcd\_draw\_string, 13
  - SSD1306\_CHARGE\_PUMP, 8
  - SSD1306\_COLUMNADDR, 8
  - SSD1306\_COMSCANDEC, 8
  - SSD1306\_COMSCANINC, 8
  - SSD1306\_DISPLAYALLON\_RESUME, 8
  - SSD1306\_DISPLAYALLON, 8
  - SSD1306\_DISPLAYOFF, 9
  - SSD1306\_DISPLAYON, 9
  - SSD1306\_DUTY\_CYCLE\_1\_64, 9
  - SSD1306\_FLIPS\_DISPLAY, 9
  - SSD1306\_HEIGHT, 9
  - SSD1306\_INVERTDISPLAY, 9
  - SSD1306\_MEMORYMODE, 9
  - SSD1306\_NO\_OFFSET, 9
  - SSD1306\_NORMALDISPLAY, 10
  - SSD1306\_NOP, 10
  - SSD1306\_PAGEADDR, 10
  - SSD1306\_RESISTOR\_RATIO, 10
  - SSD1306\_SEGREMAP, 10
  - SSD1306\_SETCOMPINS, 10
  - SSD1306\_SETCONTRAST, 10
  - SSD1306\_SETDISPLAYCLOCKDIV, 10
  - SSD1306\_SETDISPLAYOFFSET, 11
  - SSD1306\_SETHIGHCOLUMN, 11
  - SSD1306\_SETLOWCOLUMN, 11
  - SSD1306\_SETMULTIPLEX, 11
  - SSD1306\_SETPRECHARGE, 11
  - SSD1306\_SETSTARTLINE, 11
  - SSD1306\_SETVCOMDETECT, 11
  - SSD1306\_SWITCHCAPVCC, 11
  - SSD1306\_WIDTH, 12
  - setup\_i2c, 13
- lcd\_draw\_char
  - lcd.c, 6
  - lcd.h, 12
- lcd\_draw\_string
  - lcd.c, 6
  - lcd.h, 13
- main
  - test\_i2cmaster.c, 21
- makeRandom
  - test\_i2cmaster.c, 21
- map
  - test\_i2cmaster.c, 21
- max
  - test\_i2cmaster.c, 17
- millis
  - test\_i2cmaster.c, 22
- milliseconds\_since
  - test\_i2cmaster.c, 25
- min
  - test\_i2cmaster.c, 17
- myFont.c
  - Ascii\_1, 13
- myFont.h
  - Ascii\_1, 14
- OLEDX
  - test\_i2cmaster.c, 17
- OLEDY
  - test\_i2cmaster.c, 17
- particle
  - test\_i2cmaster.c, 25
- particleCOUNT
  - test\_i2cmaster.c, 17
- playAREAX
  - test\_i2cmaster.c, 17
- playAREAY
  - test\_i2cmaster.c, 18
- randint
  - test\_i2cmaster.c, 22
- SCL\_CLOCK
  - twimaster.c, 26
- SSD1306\_CHARGE\_PUMP
  - lcd.h, 8
- SSD1306\_COLUMNADDR
  - lcd.h, 8
- SSD1306\_COMSCANDEC
  - lcd.h, 8
- SSD1306\_COMSCANINC
  - lcd.h, 8
- SSD1306\_DISPLAYALLON\_RESUME
  - lcd.h, 8
- SSD1306\_DISPLAYALLON
  - lcd.h, 8
- SSD1306\_DISPLAYOFF
  - lcd.h, 9
- SSD1306\_DISPLAYON
  - lcd.h, 9
- SSD1306\_DUTY\_CYCLE\_1\_64
  - lcd.h, 9
- SSD1306\_FLIPS\_DISPLAY
  - lcd.h, 9
- SSD1306\_HEIGHT
  - lcd.h, 9



- SSD1306\_INVERTDISPLAY
  - lcd.h, [9](#)
- SSD1306\_MEMORYMODE
  - lcd.h, [9](#)
- SSD1306\_NO\_OFFSET
  - lcd.h, [9](#)
- SSD1306\_NORMALDISPLAY
  - lcd.h, [10](#)
- SSD1306\_NOP
  - lcd.h, [10](#)
- SSD1306\_PAGEADDR
  - lcd.h, [10](#)
- SSD1306\_RESISTOR\_RATIO
  - lcd.h, [10](#)
- SSD1306\_SEGREMAP
  - lcd.h, [10](#)
- SSD1306\_SETCOMPINS
  - lcd.h, [10](#)
- SSD1306\_SETCONTRAST
  - lcd.h, [10](#)
- SSD1306\_SETDISPLAYCLOCKDIV
  - lcd.h, [10](#)
- SSD1306\_SETDISPLAYOFFSET
  - lcd.h, [11](#)
- SSD1306\_SETHIGHCOLUMN
  - lcd.h, [11](#)
- SSD1306\_SETLOWCOLUMN
  - lcd.h, [11](#)
- SSD1306\_SETMULTIPLEX
  - lcd.h, [11](#)
- SSD1306\_SETPRECHARGE
  - lcd.h, [11](#)
- SSD1306\_SETSTARTLINE
  - lcd.h, [11](#)
- SSD1306\_SETVCOMDETECT
  - lcd.h, [11](#)
- SSD1306\_SWITCHCAPVCC
  - lcd.h, [11](#)
- SSD1306\_WIDTH
  - lcd.h, [12](#)
- setup\_i2c
  - lcd.c, [6](#)
  - lcd.h, [13](#)
- Start
  - test\_i2cmaster.c, [22](#)
- test\_i2cmaster.c
  - ADC\_data, [18](#)
  - buffer, [23](#)
  - CTC\_MATCH\_OVERFLOW, [16](#)
  - charACCX, [23](#)
  - charACCY, [23](#)
  - charR, [23](#)
  - charX, [24](#)
  - charY, [24](#)
  - checkCONTACT, [18](#)
  - checkPCONTACT, [19](#)
  - constrain, [16](#)
  - deadANIMATION, [19](#)
  - DevSSD1306, [16](#)
  - eatPART, [24](#)
  - enemyCOUNT, [16](#)
  - enemyDEAD, [19](#)
  - enemySTAT, [24](#)
  - friction, [24](#)
  - gameMode, [20](#)
  - gameStart, [24](#)
  - ISR, [20](#)
  - init\_ADC, [20](#)
  - init\_TIMER, [20](#)
  - init\_interrupt, [20](#)
  - JOYXPOS, [25](#)
  - JOYYPOS, [25](#)
  - main, [21](#)
  - makeRandom, [21](#)
  - map, [21](#)
  - max, [17](#)
  - millis, [22](#)
  - milliseconds\_since, [25](#)
  - min, [17](#)
  - OLEDX, [17](#)
  - OLEDY, [17](#)
  - particle, [25](#)
  - particleCOUNT, [17](#)
  - playAREAX, [17](#)
  - playAREAY, [18](#)
  - randint, [22](#)
  - Start, [22](#)
  - timer1\_millis, [25](#)
  - XN, [25](#)
  - YN, [25](#)
  - youWIN, [23](#)
- timer1\_millis
  - test\_i2cmaster.c, [25](#)
- twimaster.c
  - i2c\_init, [26](#)
  - i2c\_readAck, [27](#)
  - i2c\_readNak, [27](#)
  - i2c\_rep\_start, [27](#)
  - i2c\_start, [28](#)
  - i2c\_start\_wait, [28](#)
  - i2c\_stop, [28](#)
  - i2c\_write, [29](#)
  - SCL\_CLOCK, [26](#)
- XN
  - test\_i2cmaster.c, [25](#)
- YN
  - test\_i2cmaster.c, [25](#)
- youWIN
  - test\_i2cmaster.c, [23](#)