DE2 water system

Generated by Doxygen 1.9.2

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 3

Module Documentation

3.1 English font for Nokia LCD Library <english_font.h>

english_font.h for AVR-GCC.

Variables

- const unsigned char **font6x8** [ ][6]

3.1.1 Detailed Description

english_font.h for AVR-GCC.

```
#include "english_font.h"
```

The library contains definitions of symbols to be displayed on Nokia LCD.

3.2 GPIO Library <gpio.h>

GPIO library for AVR-GCC.

Functions

- void GPIO_config_output (volatile uint8_t *reg_name, uint8_t pin_num)

  Configure one output pin in Data Direction Register.
- void GPIO_config_input_pullup (volatile uint8_t *reg_name, uint8_t pin_num)

  Configure one input pin and enable pull-up.
- void GPIO_write_low (volatile uint8_t *reg_name, uint8_t pin_num)

  Write one pin to a low value.
- void GPIO_write_high (volatile uint8_t *reg_name, uint8_t pin_num)

  Write one pin to a low value.
- void GPIO_toggle (volatile uint8_t *reg_name, uint8_t pin_num)

  Write one pin to a low value.
- void GPIO_config_input_nopull (volatile uint8_t *reg_name, uint8_t pin_num)

  Write one pin to a low value.
- uint8_t GPIO_read (volatile uint8_t *reg_name, uint8_t pin_num)

  Read a value from input pin.

3.2.1 Detailed Description

GPIO library for AVR-GCC.
`#include "gpio.h"`

The library contains functions for controlling AVRs' gpio pin(s).

Note

Based on AVR Libc Reference Manual. Tested on ATmega328P (Arduino Uno), 16 MHz, AVR 8-bit Toolchain 3.6.2.

Author

Tomas Fryza, Dept. of Radio Electronics, Brno University of Technology, Czechia

Copyright

3.2.2 Function Documentation

3.2.2.1 GPIO_config_input_nopull()

```
void GPIO_config_input_nopull (
            volatile uint8_t * reg_name,
            uint8_t pin_num )
```

Write one pin to a low value.

Parameters

reg_name	Address of Port Register, such as &PORTB
pin_num	Pin designation in the interval 0 to 7

Returns

none

3.2.2.2 GPIO_config_input_pullup()

```
void GPIO_config_input_pullup (
            volatile uint8_t * reg_name,
            uint8_t pin_num )
```

Configure one input pin and enable pull-up.

Parameters

reg_name	Address of Data Direction Register, such as &DDRB
pin_num	Pin designation in the interval 0 to 7

Returns

none

3.2.2.3 GPIO_config_output()

```
void GPIO_config_output (
            volatile uint8_t * reg_name,
            uint8_t pin_num )
```

Configure one output pin in Data Direction Register.

Parameters

reg_name	Address of Data Direction Register, such as &DDRB
pin_num	Pin designation in the interval 0 to 7

Returns

none

3.2.2.4 GPIO_read()

```
uint8_t GPIO_read (
            volatile uint8_t * reg_name,
            uint8_t pin_num )
```

Read a value from input pin.

Parameters

reg_name	Address of Pin Register, such as &PINB
pin_num	Pin designation in the interval 0 to 7

Returns

Pin value

3.2.2.5 GPIO_toggle()

```
void GPIO_toggle (
            volatile uint8_t * reg_name,
            uint8_t pin_num )
```

Write one pin to a low value.

Parameters

reg_name	Address of Port Register, such as &PORTB
pin_num	Pin designation in the interval 0 to 7

Returns

none

3.2.2.6 GPIO_write_high()

```
void GPIO_write_high (
            volatile uint8_t * reg_name,
            uint8_t pin_num )
```

Write one pin to a low value.

Parameters

reg_name	Address of Port Register, such as &PORTB
pin_num	Pin designation in the interval 0 to 7

Returns

none

3.2.2.7 GPIO_write_low()

```
void GPIO_write_low (
            volatile uint8_t * reg_name,
            uint8_t pin_num )
```

Write one pin to a low value.

Parameters

reg_name	Address of Port Register, such as &PORTB
pin_num	Pin designation in the interval 0 to 7

Returns

none

3.3 Ultrasound sensor Library <HC-SR04.h>

HC-SR04 for AVR-GCC.

Functions

- void init_ultrasonic_sensor (void)

  Init ultrasound sensor. Setup pins.
- uint32_t get_dist ()

  Return last measured distance in mm.
- uint32_t get_dist_avg ()

  Return average of 10 most recent measured distances in mm.
- uint32_t get_cnt ()

  Return counts of timer of input pulse.

3.3.1 Detailed Description

HC-SR04 for AVR-GCC.

```
#include "HC-SR04.h"
```

The library contains functions for controlling HC-SR04 and getting distance, ticks and average distance.

3.3.2 Function Documentation

3.3.2.1 get_cnt()

```
uint32_t get_cnt ( )
```

Return counts of timer of input pulse.

Returns

distance in mm

3.3.2.2 get_dist()

```
uint32_t get_dist ( )
```

Return last measured distance in mm.

Returns

distance in mm

3.3.2.3 get_dist_avg()

```
uint32_t get_dist_avg ( )
```

Return average of 10 most recent measured distances in mm.

Returns

distance in mm

3.3.2.4 init_ultrasonic_sensor()

```
void init_ultrasonic_sensor (
            void  )
```

Init ultrasound sensor. Setup pins.

Returns

none

3.4 Main static <main.h>

Main file consists of functions and definitions for displaying data, water tank dimensions, calculations and other.

Macros

- #define **F_CPU** 16000000L

3.4.1 Detailed Description

Main file consists of functions and definitions for displaying data, water tank dimensions, calculations and other.

```
#include "main.h"
```

3.5 Nokia 5110 LCD Library <nokia_5110_lcd.h>

LCD Nokia 5110 library for AVR-GCC.

Macros

- #define **F_CPU** 16000000L
- #define **ARDUINO_UNO**
- #define **LCD_RST_PORT** PORTB
- #define **LCD_RST_DDR** DDRB
- #define **LCD_RST_PIN** 5
- #define **LCD_CE_PORT** PORTB
- #define **LCD_CE_DDR** DDRB
- #define **LCD_CE_PIN** 4
- #define **LCD_DC_PORT** PORTB
- #define **LCD_DC_DDR** DDRB
- #define **LCD_DC_PIN** 3
- #define **SDIN_PORT** PORTB
- #define **SDIN_DDR** DDRB
- #define **SDIN_PIN** 2
- #define **SCLK_PORT** PORTB
- #define **SCLK_DDR** DDRB
- #define **SCLK_PIN** 1
- #define **NUM_OF_CELLS** 504

Functions

- void LCD_clear (void)

  Clears LCD.
- void LCD_write_init (void)

  Displays init message defined init_msg_1.
- void LCD_init (void)

  Initializes LCD pins and LCD with default settings.
- void LCD_write_byte (unsigned char dat, unsigned char command)

  Write byte using software SPI to LCD.
- void LCD_write_english_string (unsigned char X, unsigned char Y, char ∗s)

  Write english string to display at specific location.
- void LCD_write_english_string_continue (char ∗s)

  Write string to next location.
- void LCD_write_english_string_continue_precise (char ∗s, uint16_t data_len)

  Write string to next location with precise length.
- void LCD_write_char (unsigned char c)

  Write single char to display (using english_font.h)
- void LCD_set_XY (unsigned char X, unsigned char Y)

  Set cursor on the screen to precise location.
- void LCD_write_whole_screen (unsigned char ∗cells, uint16_t cells_n, uint16_t start_x, uint16_t start_y)

  Display image consisting of bytes.
- void LCD_write_bytes_xy_defined_width (unsigned char ∗cells, uint16_t width, uint16_t size, uint16_t x, uint16_t y)

  Display image consisting of bytes with defined length after every n∗length bytes new line is sent.

Definitions LCD

- #define **LCD_RST_set** LCD_RST_PORT |= (1 << LCD_RST_PIN)

  Set RST pin to 1.
- #define **LCD_RST_clr** LCD_RST_PORT &= ~(1 << LCD_RST_PIN)

  Set RST pin to 0.
- #define **LCD_DC_set** LCD_DC_PORT |= (1 << LCD_DC_PIN)

  Set DC pin to 1.
- #define **LCD_DC_clr** LCD_DC_PORT &= ~(1 << LCD_DC_PIN)

  Set RST pin to 0.
- #define **SDIN_set** SDIN_PORT |= (1 << SDIN_PIN)

  Set SDIN (serial data in) pin to 1.
- #define **SDIN_clr** SDIN_PORT &= ~(1 << SDIN_PIN)

  Set SDIN (serial data in) pin to 0.
- #define **SCLK_set** SCLK_PORT |= (1 << SCLK_PIN)

  Set SCLK (serial clock) pin to 1.
- #define **SCLK_clr** SCLK_PORT &= ~(1 << SCLK_PIN)

  Set SCLK (serial clock) pin to 0.

3.5.1 Detailed Description

LCD Nokia 5110 library for AVR-GCC.

```
#include "nokia_5110_lcd.h"
```

Note

to modify this code to work with any pinout, modify pin definitions in nokia_5110_lcd.h

The library contains functions for controlling Nokia 5110 LCD screen via software SPI. With small adjustments it should work with any micrcontroller.

3.5.2 Function Documentation

3.5.2.1 LCD_clear()

```
void LCD_clear (
            void  )
```

Clears LCD.

Returns

none

3.5.2.2 LCD_init()

```
void LCD_init (
             void  )
```

Initializes LCD pins and LCD with default settings.

Returns

none

3.5.2.3 LCD_set_XY()

```
void LCD_set_XY (
             unsigned char X,
             unsigned char Y )
```

Set cursor on the screen to precise location.

Parameters

X	x coordinate on screen (0-84, columns)
Y	y coordinate on screen (0-6, rows)

Returns

none

3.5.2.4 LCD_write_byte()

```
void LCD_write_byte (
             unsigned char dat,
             unsigned char command )
```

Write byte using software SPI to LCD.

Parameters

dat	8bit value of byte
commad	1 bit value (1 if 8bits are command, 0 if 8bits are data)

Returns

none

3.5.2.5 LCD_write_bytes_xy_defined_width()

```
void LCD_write_bytes_xy_defined_width (
            unsigned char * cells,
            uint16_t width,
            uint16_t size,
            uint16_t x,
            uint16_t y )
```

Display image consisting of bytes with defined length after every n∗length bytes new line is sent.

Parameters

cells	∗char containing image bytes
width	width in bytes (pixels)
size	number of bytes
start_x	start x coordinate on screen (0-84, columns)
start_y	start y coordinate on screen (0-6, rows)

Returns

none

3.5.2.6 LCD_write_char()

```
void LCD_write_char (
            unsigned char c )
```

Write single char to display (using english_font.h)

Parameters

c	char to be displayed

Returns

none

3.5.2.7 LCD_write_english_string()

```
void LCD_write_english_string (
            unsigned char X,
            unsigned char Y,
            char * s )
```

Write english string to display at specific location.

Parameters

X	x coordinate on screen (0-84, columns)
Y	y coordinate on screen (0-6, rows)
∗s	string

Returns

none

3.5.2.8 LCD_write_english_string_continue()

```
void LCD_write_english_string_continue (
            char * s )
```

Write string to next location.

Parameters

∗s	string

Returns

none

3.5.2.9 LCD_write_english_string_continue_precise()

```
void LCD_write_english_string_continue_precise (
            char * s,
            uint16_t data_len )
```

Write string to next location with precise length.

Parameters

∗s	string
data_len	number of chars

Returns

none

3.5.2.10 LCD_write_init()

```
void LCD_write_init (
            void  )
```

Displays init message defined init_msg_1.

Returns

none

3.5.2.11 LCD_write_whole_screen()

```
void LCD_write_whole_screen (
            unsigned char * cells,
            uint16_t cells_n,
            uint16_t start_x,
            uint16_t start_y )
```

Display image consisting of bytes.

Parameters

cells	∗char containing image bytes
cells_n	number of bytes
start_x	start x coordinate on screen (0-84, columns)
start_y	start y coordinate on screen (0-6, rows)

Returns

none

3.6 Timer Library <timer.h>

Timer library for AVR-GCC.

Definitions for 16-bit Timer/Counter1

Note

t_OVF = 1/F_CPU ∗ prescaler ∗ 2^n where n = 16, F_CPU = 16 MHz

- #define **TIM1_stop**() TCCR1B &= ∼((1<<CS12) | (1<<CS11) | (1<<CS10));
  Stop timer, prescaler 000 --> STOP.
- #define **TIM1_overflow_4ms**() TCCR1B &= ∼((1<<CS12) | (1<<CS11)); TCCR1B |= (1<<CS10);
  Set overflow 4ms, prescaler 001 --> 1.
- #define **TIM1_overflow_33ms**() TCCR1B &= ∼((1<<CS12) | (1<<CS10)); TCCR1B |= (1<<CS11);
  Set overflow 33ms, prescaler 010 --> 8.
- #define **TIM1_overflow_262ms**() TCCR1B &= ∼(1<<CS12); TCCR1B |= (1<<CS11) | (1<<CS10);
  Set overflow 262ms, prescaler 011 --> 64.
- #define **TIM1_overflow_1s**() TCCR1B &= ∼((1<<CS11) | (1<<CS10)); TCCR1B |= (1<<CS12);
  Set overflow 1s, prescaler 100 --> 256.
- #define **TIM1_overflow_4s**() TCCR1B &= ∼(1<<CS11); TCCR1B |= (1<<CS12) | (1<<CS10);
  Set overflow 4s, prescaler // 101 --> 1024.
- #define **TIM1_overflow_interrupt_enable**() TIMSK1 |= (1<<TOIE1);
  Enable overflow interrupt, 1 --> enable.
- #define **TIM1_overflow_interrupt_disable**() TIMSK1 &= ∼(1<<TOIE1);
  Disable overflow interrupt, 0 --> disable.

Definitions for 8-bit Timer/Counter0

Note

t_OVF = 1/F_CPU ∗ prescaler ∗ 2^n where n = 8, F_CPU = 16 MHz

- #define **TIM0_stop**() TCCR0B &= ∼((1<<CS02) | (1<<CS01) | (1<<CS00));
- #define **TIM0_overflow_16us**() TCCR0B &= ∼((1<<CS02) | (1<<CS01)); TCCR0B |= (1<<CS00);
  Set overflow 16us, prescaler 001 --> 1.
- #define **TIM0_overflow_128us**() TCCR0B &= ∼((1<<CS02) | (1<<CS00)); TCCR0B |= (1<<CS01);
  Set overflow 128us, prescaler 010 --> 8.
- #define **TIM0_overflow_1024us**() TCCR0B &= ∼(1<<CS02); TCCR0B |= (1<<CS01) | (1<<CS00);
  Set overflow 1024 us, prescaler 011 --> 64.
- #define **TIM0_overflow_4096us**() TCCR0B &= ∼((1<<CS01) | (1<<CS00)); TCCR0B |= (1<<CS02);
  Set overflow 4096us, prescaler 100 --> 256.
- #define **TIM0_overflow_16384us**() TCCR0B &= ∼(1<<CS01); TCCR0B |= (1<<CS02) | (1<<CS00);
  Set overflow 16384 us, prescaler // 101 --> 1024.
- #define **TIM0_overflow_interrupt_enable**() TIMSK0 |= (1<<TOIE0);
  Enable overflow interrupt, 1 --> enable.
- #define **TIM0_overflow_interrupt_disable**() TIMSK0 &= ∼(1<<TOIE0);
  Disable overflow interrupt, 0 --> disable.

Definitions for 8-bit Timer/Counter2

Note

t_OVF = 1/F_CPU ∗ prescaler ∗ 2^n where n = 8, F_CPU = 16 MHz

- #define **TIM2_stop**() TCCR2B &= ∼((1<<CS22) | (1<<CS21) | (1<<CS20));
- #define **TIM2_overflow_16us**() TCCR2B &= ∼((1<<CS22) | (1<<CS21)); TCCR2B |= (1<<CS20);

  Set overflow 16us, prescaler 001 --> 1.
- #define **TIM2_overflow_128us**() TCCR2B &= ∼((1<<CS22) | (1<<CS20)); TCCR2B |= (1<<CS21);

  Set overflow 128us, prescaler 010 --> 8.
- #define **TIM2_overflow_1024us**() TCCR2B &= ∼(1<<CS22); TCCR2B |= (1<<CS21) | (1<<CS20);

  Set overflow 1024 us, prescaler 011 --> 64.
- #define **TIM2_overflow_4096us**() TCCR2B &= ∼((1<<CS21) | (1<<CS20)); TCCR2B |= (1<<CS22);

  Set overflow 4096 us, prescaler 100 --> 256.
- #define **TIM2_overflow_16384us**() TCCR2B |= (1<<CS22) | (1<<CS21) | (1<<CS20);

  Set overflow 16384 us, prescaler // 101 --> 1024.
- #define **TIM2_overflow_interrupt_enable**() TIMSK2 |= (1<<TOIE2);

  Enable overflow interrupt, 1 --> enable.
- #define **TIM2_overflow_interrupt_disable**() TIMSK2 &= ∼(1<<TOIE2);

  Disable overflow interrupt, 0 --> disable.

3.6.1 Detailed Description

Timer library for AVR-GCC.

```
#include "timer.h"
```

The library contains macros for controlling the timer modules.

Note

Based on Microchip Atmel ATmega328P manual and no source file is needed for the library.

Author

Tomas Fryza, Dept. of Radio Electronics, Brno University of Technology, Czechia

Copyright

3.7 Water symbols static <water_symbols.h>

Application symbols stored as array for Nokia 5110 LCD, AVR-GCC compatible.

Variables

- const char **init_msg_1** [340]
- const char **water_level_default** [70]
- const char **water_level_error** [70]
- const char **water_level_10** [70]
- const char **water_level_20** [70]
- const char **water_level_30** [70]
- const char **water_level_40** [70]
- const char **water_level_50** [70]
- const char **water_level_60** [70]
- const char **water_level_70** [70]
- const char **water_level_80** [70]
- const char **water_level_90** [70]
- const char **water_level_100** [70]

3.7.1 Detailed Description

Application symbols stored as array for Nokia 5110 LCD, AVR-GCC compatible.

```
#include "water_symbols.h"
```

Chapter 4

File Documentation

4.1 C:/Users/gkaretka/Documents/GitHub/DE2_Project_2021-22/code/water_height_meter/english_font.h File Reference

Variables

- const unsigned char **font6x8** [ ][6]

4.2 C:/Users/gkaretka/Documents/GitHub/DE2_Project_2021-22/code/water_height_meter/english_font.h

Go to the documentation of this file.

```
1 /*
2  * english_font.c
3  *
4  * Created: 27-Jul-14 15:27:46
5  *  Author: 4a4ik
6  */
7
20 // 6 x 8 font
21 // 1 pixel space at left and bottom
22 // index = ASCII - 32
23
24 #ifndef ENGLISH_FONT_H_
25 #define ENGLISH_FONT_H_
26
27 const unsigned char font6x8[][6] =
28 {
29     { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // sp
30     { 0x00, 0x00, 0x00, 0x2f, 0x00, 0x00 },   // !
31     { 0x00, 0x00, 0x07, 0x00, 0x07, 0x00 },   // "
32     { 0x00, 0x14, 0x7f, 0x14, 0x7f, 0x14 },   // #
33     { 0x00, 0x24, 0x2a, 0x7f, 0x2a, 0x12 },   // $
34     { 0x00, 0x62, 0x64, 0x08, 0x13, 0x23 },   // %
35     { 0x00, 0x36, 0x49, 0x55, 0x22, 0x50 },   // &
36     { 0x00, 0x00, 0x05, 0x03, 0x00, 0x00 },   // '
37     { 0x00, 0x00, 0x1c, 0x22, 0x41, 0x00 },   // (
38     { 0x00, 0x00, 0x41, 0x22, 0x1c, 0x00 },   // )
39     { 0x00, 0x14, 0x08, 0x3E, 0x08, 0x14 },   // *
40     { 0x00, 0x08, 0x08, 0x3E, 0x08, 0x08 },   // +
41     { 0x00, 0x00, 0x00, 0xA0, 0x60, 0x00 },   // ,
42     { 0x00, 0x08, 0x08, 0x08, 0x08, 0x08 },   // -
43     { 0x00, 0x00, 0x60, 0x60, 0x00, 0x00 },   // .
44     { 0x00, 0x20, 0x10, 0x08, 0x04, 0x02 },   // /
45     { 0x00, 0x3E, 0x51, 0x49, 0x45, 0x3E },   // 0
46     { 0x00, 0x00, 0x42, 0x7F, 0x40, 0x00 },   // 1
47     { 0x00, 0x42, 0x61, 0x51, 0x49, 0x46 },   // 2
48     { 0x00, 0x21, 0x41, 0x45, 0x4B, 0x31 },   // 3
49     { 0x00, 0x18, 0x14, 0x12, 0x7F, 0x10 },   // 4
```

```
50     { 0x00, 0x27, 0x45, 0x45, 0x45, 0x39 },   // 5
51     { 0x00, 0x3C, 0x4A, 0x49, 0x49, 0x30 },   // 6
52     { 0x00, 0x01, 0x71, 0x09, 0x05, 0x03 },   // 7
53     { 0x00, 0x36, 0x49, 0x49, 0x49, 0x36 },   // 8
54     { 0x00, 0x06, 0x49, 0x49, 0x29, 0x1E },   // 9
55     { 0x00, 0x00, 0x36, 0x36, 0x00, 0x00 },   // :
56     { 0x00, 0x00, 0x56, 0x36, 0x00, 0x00 },   // ;
57     { 0x00, 0x08, 0x14, 0x22, 0x41, 0x00 },   // <
58     { 0x00, 0x14, 0x14, 0x14, 0x14, 0x14 },   // =
59     { 0x00, 0x00, 0x41, 0x22, 0x14, 0x08 },   // >
60     { 0x00, 0x02, 0x01, 0x51, 0x09, 0x06 },   // ?
61     { 0x00, 0x32, 0x49, 0x59, 0x51, 0x3E },   // @
62     { 0x00, 0x7C, 0x12, 0x11, 0x12, 0x7C },   // A
63     { 0x00, 0x7F, 0x49, 0x49, 0x49, 0x36 },   // B
64     { 0x00, 0x3E, 0x41, 0x41, 0x41, 0x22 },   // C
65     { 0x00, 0x7F, 0x41, 0x41, 0x22, 0x1C },   // D
66     { 0x00, 0x7F, 0x49, 0x49, 0x49, 0x41 },   // E
67     { 0x00, 0x7F, 0x09, 0x09, 0x09, 0x01 },   // F
68     { 0x00, 0x3E, 0x41, 0x49, 0x49, 0x7A },   // G
69     { 0x00, 0x7F, 0x08, 0x08, 0x08, 0x7F },   // H
70     { 0x00, 0x00, 0x41, 0x7F, 0x41, 0x00 },   // I
71     { 0x00, 0x20, 0x40, 0x41, 0x3F, 0x01 },   // J
72     { 0x00, 0x7F, 0x08, 0x14, 0x22, 0x41 },   // K
73     { 0x00, 0x7F, 0x40, 0x40, 0x40, 0x40 },   // L
74     { 0x00, 0x7F, 0x02, 0x0C, 0x02, 0x7F },   // M
75     { 0x00, 0x7F, 0x04, 0x08, 0x10, 0x7F },   // N
76     { 0x00, 0x3E, 0x41, 0x41, 0x41, 0x3E },   // O
77     { 0x00, 0x7F, 0x09, 0x09, 0x09, 0x06 },   // P
78     { 0x00, 0x3E, 0x41, 0x51, 0x21, 0x5E },   // Q
79     { 0x00, 0x7F, 0x09, 0x19, 0x29, 0x46 },   // R
80     { 0x00, 0x46, 0x49, 0x49, 0x49, 0x31 },   // S
81     { 0x00, 0x01, 0x01, 0x7F, 0x01, 0x01 },   // T
82     { 0x00, 0x3F, 0x40, 0x40, 0x40, 0x3F },   // U
83     { 0x00, 0x1F, 0x20, 0x40, 0x20, 0x1F },   // V
84     { 0x00, 0x3F, 0x40, 0x38, 0x40, 0x3F },   // W
85     { 0x00, 0x63, 0x14, 0x08, 0x14, 0x63 },   // X
86     { 0x00, 0x07, 0x08, 0x70, 0x08, 0x07 },   // Y
87     { 0x00, 0x61, 0x51, 0x49, 0x45, 0x43 },   // Z
88     { 0x00, 0x00, 0x7F, 0x41, 0x41, 0x00 },   // [
89     { 0x00, 0x55, 0x2A, 0x55, 0x2A, 0x55 },   // 55
90     { 0x00, 0x00, 0x41, 0x41, 0x7F, 0x00 },   // ]
91     { 0x00, 0x04, 0x02, 0x01, 0x02, 0x04 },   // ^
92     { 0x00, 0x40, 0x40, 0x40, 0x40, 0x40 },   // _
93     { 0x00, 0x00, 0x01, 0x02, 0x04, 0x00 },   // '
94     { 0x00, 0x20, 0x54, 0x54, 0x54, 0x78 },   // a
95     { 0x00, 0x7F, 0x48, 0x44, 0x44, 0x38 },   // b
96     { 0x00, 0x38, 0x44, 0x44, 0x44, 0x20 },   // c
97     { 0x00, 0x38, 0x44, 0x44, 0x48, 0x7F },   // d
98     { 0x00, 0x38, 0x54, 0x54, 0x54, 0x18 },   // e
99     { 0x00, 0x08, 0x7E, 0x09, 0x01, 0x02 },   // f
100     { 0x00, 0x18, 0xA4, 0xA4, 0xA4, 0x7C },   // g
101     { 0x00, 0x7F, 0x08, 0x04, 0x04, 0x78 },   // h
102     { 0x00, 0x00, 0x44, 0x7D, 0x40, 0x00 },   // i
103     { 0x00, 0x40, 0x80, 0x84, 0x7D, 0x00 },   // j
104     { 0x00, 0x7F, 0x10, 0x28, 0x44, 0x00 },   // k
105     { 0x00, 0x00, 0x41, 0x7F, 0x40, 0x00 },   // l
106     { 0x00, 0x7C, 0x04, 0x18, 0x04, 0x78 },   // m
107     { 0x00, 0x7C, 0x08, 0x04, 0x04, 0x78 },   // n
108     { 0x00, 0x38, 0x44, 0x44, 0x44, 0x38 },   // o
109     { 0x00, 0xFC, 0x24, 0x24, 0x24, 0x18 },   // p
110     { 0x00, 0x18, 0x24, 0x24, 0x18, 0xFC },   // q
111     { 0x00, 0x7C, 0x08, 0x04, 0x04, 0x08 },   // r
112     { 0x00, 0x48, 0x54, 0x54, 0x54, 0x20 },   // s
113     { 0x00, 0x04, 0x3F, 0x44, 0x40, 0x20 },   // t
114     { 0x00, 0x3C, 0x40, 0x40, 0x20, 0x7C },   // u
115     { 0x00, 0x1C, 0x20, 0x40, 0x20, 0x1C },   // v
116     { 0x00, 0x3C, 0x40, 0x30, 0x40, 0x3C },   // w
117     { 0x00, 0x44, 0x28, 0x10, 0x28, 0x44 },   // x
118     { 0x00, 0x1C, 0xA0, 0xA0, 0xA0, 0x7C },   // y
119     { 0x00, 0x44, 0x64, 0x54, 0x4C, 0x44 },   // z
120     { 0x14, 0x14, 0x14, 0x14, 0x14, 0x14 }    // horiz lines
121 };
122
123 #endif /* ENGLISH_FONT_H_ */
```

4.3 C:/Users/gkaretka/Documents/GitHub/DE2_Project_2021-22/code/water_height_meter/gpio.h File Reference

```
#include <avr/io.h>
```

Functions

Functions

- void GPIO_config_output (volatile uint8_t ∗reg_name, uint8_t pin_num)
  Configure one output pin in Data Direction Register.
- void GPIO_config_input_pullup (volatile uint8_t ∗reg_name, uint8_t pin_num)
  Configure one input pin and enable pull-up.
- void GPIO_write_low (volatile uint8_t ∗reg_name, uint8_t pin_num)
  Write one pin to a low value.
- void GPIO_write_high (volatile uint8_t ∗reg_name, uint8_t pin_num)
  Write one pin to a low value.
- void GPIO_toggle (volatile uint8_t ∗reg_name, uint8_t pin_num)
  Write one pin to a low value.
- void GPIO_config_input_nopull (volatile uint8_t ∗reg_name, uint8_t pin_num)
  Write one pin to a low value.
- uint8_t GPIO_read (volatile uint8_t ∗reg_name, uint8_t pin_num)
  Read a value from input pin.

4.4 C:/Users/gkaretka/Documents/GitHub/DE2_Project_2021-22/code/water_height_meter/gpio.h

Go to the documentation of this file.

```
1 #ifndef GPIO_H
2 #define GPIO_H
3
4 /************************************************************************
5  *
6  * GPIO library for AVR-GCC.
7  * ATmega328P (Arduino Uno), 16 MHz, AVR 8-bit Toolchain 3.6.2
8  *
9  * Copyright (c) 2019-Present Tomas Fryza
10  * Dept. of Radio Electronics, Brno University of Technology, Czechia
11  * This work is licensed under the terms of the MIT license.
12  *
13  ************************************************************************/
14
34 /* Includes ----------------------------------------------------------*/
35 #include <avr/io.h>
36
37
38 /* Function prototypes -----------------------------------------------*/
49 void GPIO_config_output(volatile uint8_t *reg_name, uint8_t pin_num);
50
51
52 /* GPIO_config_input_nopull */
53
54
61 void GPIO_config_input_pullup(volatile uint8_t *reg_name, uint8_t pin_num);
62
63
70 void GPIO_write_low(volatile uint8_t *reg_name, uint8_t pin_num);
71
72
79 void GPIO_write_high(volatile uint8_t *reg_name, uint8_t pin_num);
80
81
88 void GPIO_toggle(volatile uint8_t *reg_name, uint8_t pin_num);
89
90
97 void GPIO_config_input_nopull(volatile uint8_t *reg_name, uint8_t pin_num);
98
105 uint8_t GPIO_read(volatile uint8_t *reg_name, uint8_t pin_num);
106
107
108 #endif
```

4.5 C:/Users/gkaretka/Documents/GitHub/DE2_Project_2021-22/code/water_height_meter/HC-SR04.h File Reference

Functions

Functions

- void init_ultrasonic_sensor (void)
  Init ultrasound sensor. Setup pins.
- uint32_t get_dist ()
  Return last measured distance in mm.
- uint32_t get_dist_avg ()
  Return average of 10 most recent measured distances in mm.
- uint32_t get_cnt ()
  Return counts of timer of input pulse.

4.6 C:/Users/gkaretka/Documents/GitHub/DE2_Project_2021-22/code/water_height_meter/HC-SR04.h

Go to the documentation of this file.

```
1
11 #ifndef HC_SR04_H_
12 #define HC_SR04_H_
13
14 /* Function prototypes -----------------------------------------------*/
15
24 void init_ultrasonic_sensor(void);
25
30 uint32_t get_dist();
31
36 uint32_t get_dist_avg();
37
42 uint32_t get_cnt();
43
44 #endif /* HC-SR04_H_ */
```

4.7 C:/Users/gkaretka/Documents/GitHub/DE2_Project_2021-22/code/water_height_meter/main.h File Reference

```
#include <avr/io.h>
#include <util/delay.h>
#include <stdlib.h>
#include <avr/interrupt.h>
#include "water_symbols.h"
#include "nokia_5110_lcd.h"
#include "HC-SR04.h"
#include "gpio.h"
#include "timer.h"
#include <string.h>
```

Macros

- #define **F_CPU** 16000000L

4.8 C:/Users/gkaretka/Documents/GitHub/DE2_Project_2021-22/code/water_height_meter/main.h

Go to the documentation of this file.

```
1 /*
2  * main.h
3  *
4  * Created: 24. 11. 2021 11:43:41
5  *  Author: gkaretka
6  */
7
18 #ifndef MAIN_H_
19 #define MAIN_H_
20
21 #ifndef F_CPU
22 #define F_CPU 16000000L
23 #endif
24
25 #include <avr/io.h>
26 #include <util/delay.h>
27 #include <stdlib.h>
28 #include <avr/io.h>
29 #include <avr/interrupt.h>
30
31 #include "water_symbols.h"
32 #include "nokia_5110_lcd.h"
33 #include "HC-SR04.h"
34 #include "gpio.h"
35 #include "timer.h"
36 #include <string.h>
37
38 #endif /* MAIN_H_ */
```

4.9 C:/Users/gkaretka/Documents/GitHub/DE2_Project_2021-22/code/water_height_meter/nokia_5110_lcd.h File Reference

```
#include <avr/io.h>
#include <util/delay.h>
#include "water_symbols.h"
```

Macros

- #define **F_CPU** 16000000L
- #define **ARDUINO_UNO**
- #define **LCD_RST_PORT** PORTB
- #define **LCD_RST_DDR** DDRB
- #define **LCD_RST_PIN** 5
- #define **LCD_CE_PORT** PORTB
- #define **LCD_CE_DDR** DDRB
- #define **LCD_CE_PIN** 4
- #define **LCD_DC_PORT** PORTB
- #define **LCD_DC_DDR** DDRB
- #define **LCD_DC_PIN** 3
- #define **SDIN_PORT** PORTB
- #define **SDIN_DDR** DDRB
- #define **SDIN_PIN** 2
- #define **SCLK_PORT** PORTB
- #define **SCLK_DDR** DDRB
- #define **SCLK_PIN** 1
- #define **NUM_OF_CELLS** 504

Definitions LCD

- #define **LCD_RST_set** LCD_RST_PORT |= (1 << LCD_RST_PIN)
  Set RST pin to 1.
- #define **LCD_RST_clr** LCD_RST_PORT &= ~(1 << LCD_RST_PIN)
  Set RST pin to 0.
- #define **LCD_DC_set** LCD_DC_PORT |= (1 << LCD_DC_PIN)
  Set DC pin to 1.
- #define **LCD_DC_clr** LCD_DC_PORT &= ~(1 << LCD_DC_PIN)
  Set RST pin to 0.
- #define **SDIN_set** SDIN_PORT |= (1 << SDIN_PIN)
  Set SDIN (serial data in) pin to 1.
- #define **SDIN_clr** SDIN_PORT &= ~(1 << SDIN_PIN)
  Set SDIN (serial data in) pin to 0.
- #define **SCLK_set** SCLK_PORT |= (1 << SCLK_PIN)
  Set SCLK (serial clock) pin to 1.
- #define **SCLK_clr** SCLK_PORT &= ~(1 << SCLK_PIN)
  Set SCLK (serial clock) pin to 0.

Functions

Functions

- void LCD_clear (void)
  Clears LCD.
- void LCD_write_init (void)
  Displays init message defined init_msg_1.
- void LCD_init (void)
  Initializes LCD pins and LCD with default settings.
- void LCD_write_byte (unsigned char dat, unsigned char command)
  Write byte using software SPI to LCD.
- void LCD_write_english_string (unsigned char X, unsigned char Y, char *s)
  Write english string to display at specific location.
- void LCD_write_english_string_continue (char *s)
  Write string to next location.
- void LCD_write_english_string_continue_precise (char *s, uint16_t data_len)
  Write string to next location with precise length.
- void LCD_write_char (unsigned char c)
  Write single char to display (using english_font.h)
- void LCD_set_XY (unsigned char X, unsigned char Y)
  Set cursor on the screen to precise location.
- void LCD_write_whole_screen (unsigned char *cells, uint16_t cells_n, uint16_t start_x, uint16_t start_y)
  Display image consisting of bytes.
- void LCD_write_bytes_xy_defined_width (unsigned char *cells, uint16_t width, uint16_t size, uint16_t x, uint16_t y)
  *Display image consisting of bytes with defined length after every n*length bytes new line is sent.*

4.10 C:/Users/gkaretka/Documents/GitHub/DE2_Project_2021-22/code/water_height_meter/nokia_5110_lcd.h

Go to the documentation of this file.

```
1
15 #ifndef NOKIA_5110_LCD_H_
16 #define NOKIA_5110_LCD_H_
17
18 #ifndef F_CPU
```

```
19      #define F_CPU 16000000L
20  #endif
21
22  #include <avr/io.h>
23  #include <util/delay.h>
24  #include "water_symbols.h"
25
26  //#define ARDUINO_MEGA
27  #define ARDUINO_UNO
28
29  /* Arduino Mega
30   * E5 - RST
31   * G5 - CE - can be asserted low
32   * E3 - DC - data/command
33   * H3 - DIN - data in
34   * H4 - CLK - SPI clk
35   */
36
37  /* Arduino Uno
38   * B5 - RST
39   * B4 - CE - can be asserted low
40   * B3 - DC - data/command
41   * B2 - DIN - data in
42   * B1 - CLK - SPI clk
43  */
44  #ifdef ARDUINO_MEGA
45  #define LCD_RST_PORT    PORTE
46  #define LCD_RST_DDR     DDRE
47  #define LCD_RST_PIN     5
48
49  #define LCD_CE_PORT     PORTG
50  #define LCD_CE_DDR      DDRG
51  #define LCD_CE_PIN      5
52
53  #define LCD_DC_PORT     PORTE
54  #define LCD_DC_DDR      DDRE
55  #define LCD_DC_PIN      3
56
57  #define SDIN_PORT       PORTH
58  #define SDIN_DDR        DDRH
59  #define SDIN_PIN        3
60
61  #define SCLK_PORT       PORTH
62  #define SCLK_DDR        DDRH
63  #define SCLK_PIN        4
64
65  #else
66
67  #define LCD_RST_PORT    PORTB
68  #define LCD_RST_DDR     DDRB
69  #define LCD_RST_PIN     5
70
71  #define LCD_CE_PORT     PORTB
72  #define LCD_CE_DDR      DDRB
73  #define LCD_CE_PIN      4
74
75  #define LCD_DC_PORT     PORTB
76  #define LCD_DC_DDR      DDRB
77  #define LCD_DC_PIN      3
78
79  #define SDIN_PORT       PORTB
80  #define SDIN_DDR        DDRB
81  #define SDIN_PIN        2
82
83  #define SCLK_PORT       PORTB
84  #define SCLK_DDR        DDRB
85  #define SCLK_PIN        1
86
87  #endif
88
89  #define NUM_OF_CELLS    504
90
91  /*
92   *
93   *  DO NOT EDIT ANYTHING BELOW
94   *
95   */
96
97  /* Macros to ease the work */
98
104 #define LCD_RST_set   LCD_RST_PORT    |=  (1 << LCD_RST_PIN)
106 #define LCD_RST_clr   LCD_RST_PORT    &= ~(1 << LCD_RST_PIN)
107
109 #define LCD_DC_set    LCD_DC_PORT     |=  (1 << LCD_DC_PIN)
111 #define LCD_DC_clr    LCD_DC_PORT     &= ~(1 << LCD_DC_PIN)
112
114 #define SDIN_set      SDIN_PORT       |=  (1 << SDIN_PIN)
```

```
116 #define SDIN_clr     SDIN_PORT      &= ~(1 « SDIN_PIN)
117
119 #define SCLK_set     SCLK_PORT      |=  (1 « SCLK_PIN)
121 #define SCLK_clr     SCLK_PORT      &= ~(1 « SCLK_PIN)
122
123 /* Function prototypes ------------------------------------------------*/
124
133 void LCD_clear(void);
134
139 void LCD_write_init(void);
140
145 void LCD_init(void);
146
153 void LCD_write_byte(unsigned char dat, unsigned char command);
154
162 void LCD_write_english_string(unsigned char X, unsigned char Y, char *s);
163
169 void LCD_write_english_string_continue(char *s);
170
177 void LCD_write_english_string_continue_precise(char *s, uint16_t data_len);
178
184 void LCD_write_char(unsigned char c);
185
192 void LCD_set_XY(unsigned char X, unsigned char Y);
193
202 void LCD_write_whole_screen(unsigned char *cells, uint16_t cells_n, uint16_t start_x, uint16_t start_y);
203
214 void LCD_write_bytes_xy_defined_width(unsigned char *cells, uint16_t width, uint16_t size, uint16_t x,
       uint16_t y);
215
216
217 #endif /* NOKIA_5110_LCD_H_ */
```

4.11 C:/Users/gkaretka/Documents/GitHub/DE2_Project_2021-22/code/water_height_meter/timer.h File Reference

```
#include <avr/io.h>
```

Macros

Definitions for 16-bit Timer/Counter1

Note

t_OVF = 1/F_CPU ∗ prescaler ∗ 2^n where n = 16, F_CPU = 16 MHz

- #define **TIM1_stop**() TCCR1B &= ~((1<<CS12) | (1<<CS11) | (1<<CS10));
  Stop timer, prescaler 000 --> STOP.
- #define **TIM1_overflow_4ms**() TCCR1B &= ~((1<<CS12) | (1<<CS11)); TCCR1B |= (1<<CS10);
  Set overflow 4ms, prescaler 001 --> 1.
- #define **TIM1_overflow_33ms**() TCCR1B &= ~((1<<CS12) | (1<<CS10)); TCCR1B |= (1<<CS11);
  Set overflow 33ms, prescaler 010 --> 8.
- #define **TIM1_overflow_262ms**() TCCR1B &= ~(1<<CS12); TCCR1B |= (1<<CS11) | (1<<CS10);
  Set overflow 262ms, prescaler 011 --> 64.
- #define **TIM1_overflow_1s**() TCCR1B &= ~((1<<CS11) | (1<<CS10)); TCCR1B |= (1<<CS12);
  Set overflow 1s, prescaler 100 --> 256.
- #define **TIM1_overflow_4s**() TCCR1B &= ~(1<<CS11); TCCR1B |= (1<<CS12) | (1<<CS10);
  Set overflow 4s, prescaler // 101 --> 1024.
- #define **TIM1_overflow_interrupt_enable**() TIMSK1 |= (1<<TOIE1);
  Enable overflow interrupt, 1 --> enable.
- #define **TIM1_overflow_interrupt_disable**() TIMSK1 &= ~(1<<TOIE1);
  Disable overflow interrupt, 0 --> disable.

Definitions for 8-bit Timer/Counter0

Note

t_OVF = 1/F_CPU ∗ prescaler ∗ 2^n where n = 8, F_CPU = 16 MHz

- #define **TIM0_stop**() TCCR0B &= ∼((1<<CS02) | (1<<CS01) | (1<<CS00));
- #define **TIM0_overflow_16us**() TCCR0B &= ∼((1<<CS02) | (1<<CS01)); TCCR0B |= (1<<CS00);
  Set overflow 16us, prescaler 001 --> 1.
- #define **TIM0_overflow_128us**() TCCR0B &= ∼((1<<CS02) | (1<<CS00)); TCCR0B |= (1<<CS01);
  Set overflow 128us, prescaler 010 --> 8.
- #define **TIM0_overflow_1024us**() TCCR0B &= ∼(1<<CS02); TCCR0B |= (1<<CS01) | (1<<CS00);
  Set overflow 1024 us, prescaler 011 --> 64.
- #define **TIM0_overflow_4096us**() TCCR0B &= ∼((1<<CS01) | (1<<CS00)); TCCR0B |= (1<<CS02);
  Set overflow 4096us, prescaler 100 --> 256.
- #define **TIM0_overflow_16384us**() TCCR0B &= ∼(1<<CS01); TCCR0B |= (1<<CS02) | (1<<CS00);
  Set overflow 16384 us, prescaler // 101 --> 1024.
- #define **TIM0_overflow_interrupt_enable**() TIMSK0 |= (1<<TOIE0);
  Enable overflow interrupt, 1 --> enable.
- #define **TIM0_overflow_interrupt_disable**() TIMSK0 &= ∼(1<<TOIE0);
  Disable overflow interrupt, 0 --> disable.

Definitions for 8-bit Timer/Counter2

Note

t_OVF = 1/F_CPU ∗ prescaler ∗ 2^n where n = 8, F_CPU = 16 MHz

- #define **TIM2_stop**() TCCR2B &= ∼((1<<CS22) | (1<<CS21) | (1<<CS20));
- #define **TIM2_overflow_16us**() TCCR2B &= ∼((1<<CS22) | (1<<CS21)); TCCR2B |= (1<<CS20);
  Set overflow 16us, prescaler 001 --> 1.
- #define **TIM2_overflow_128us**() TCCR2B &= ∼((1<<CS22) | (1<<CS20)); TCCR2B |= (1<<CS21);
  Set overflow 128us, prescaler 010 --> 8.
- #define **TIM2_overflow_1024us**() TCCR2B &= ∼(1<<CS22); TCCR2B |= (1<<CS21) | (1<<CS20);
  Set overflow 1024 us, prescaler 011 --> 64.
- #define **TIM2_overflow_4096us**() TCCR2B &= ∼((1<<CS21) | (1<<CS20)); TCCR2B |= (1<<CS22);
  Set overflow 4096 us, prescaler 100 --> 256.
- #define **TIM2_overflow_16384us**() TCCR2B |= (1<<CS22) | (1<<CS21) | (1<<CS20);
  Set overflow 16384 us, prescaler // 101 --> 1024.
- #define **TIM2_overflow_interrupt_enable**() TIMSK2 |= (1<<TOIE2);
  Enable overflow interrupt, 1 --> enable.
- #define **TIM2_overflow_interrupt_disable**() TIMSK2 &= ∼(1<<TOIE2);
  Disable overflow interrupt, 0 --> disable.

4.12 C:/Users/gkaretka/Documents/GitHub/DE2_Project_2021-22/code/water_height_meter/timer.h

Go to the documentation of this file.

```
1 #ifndef TIMER_H
2 # define TIMER_H
3
4 /************************************************************************
5  *
6  * Timer library for AVR-GCC.
7  * ATmega328P (Arduino Uno), 16 MHz, AVR 8-bit Toolchain 3.6.2
8  *
9  * Copyright (c) 2019-Present Tomas Fryza
10  * Dept. of Radio Electronics, Brno University of Technology, Czechia
11  * This work is licensed under the terms of the MIT license.
12  *
13  ************************************************************************/
14
34 /* Includes ----------------------------------------------------------*/
35 #include <avr/io.h>
36
```

```
37
38 /* Defines ------------------------------------------------------------*/
44 #define TIM1_stop()           TCCR1B &= ~((1«CS12) | (1«CS11) | (1«CS10));
46 #define TIM1_overflow_4ms()   TCCR1B &= ~((1«CS12) | (1«CS11)); TCCR1B |= (1«CS10);
48 #define TIM1_overflow_33ms()  TCCR1B &= ~((1«CS12) | (1«CS10)); TCCR1B |= (1«CS11);
50 #define TIM1_overflow_262ms() TCCR1B &= ~(1«CS12); TCCR1B |= (1«CS11) | (1«CS10);
52 #define TIM1_overflow_1s()    TCCR1B &= ~((1«CS11) | (1«CS10)); TCCR1B |= (1«CS12);
54 #define TIM1_overflow_4s()    TCCR1B &= ~(1«CS11); TCCR1B |= (1«CS12) | (1«CS10);
56 #define TIM1_overflow_interrupt_enable()  TIMSK1 |= (1«TOIE1);
58 #define TIM1_overflow_interrupt_disable() TIMSK1 &= ~(1«TOIE1);
59
64 #define TIM0_stop()                       TCCR0B &= ~((1«CS02) | (1«CS01) | (1«CS00));
66 #define TIM0_overflow_16us()              TCCR0B &= ~((1«CS02) | (1«CS01)); TCCR0B |= (1«CS00);
68 #define TIM0_overflow_128us()             TCCR0B &= ~((1«CS02) | (1«CS00)); TCCR0B |= (1«CS01);
70 #define TIM0_overflow_1024us()            TCCR0B &= ~(1«CS02); TCCR0B |= (1«CS01) | (1«CS00);
72 #define TIM0_overflow_4096us()            TCCR0B &= ~((1«CS01) | (1«CS00)); TCCR0B |= (1«CS02);
74 #define TIM0_overflow_16384us()           TCCR0B &= ~(1«CS01); TCCR0B |= (1«CS02) | (1«CS00);
76 #define TIM0_overflow_interrupt_enable()  TIMSK0 |= (1«TOIE0);
78 #define TIM0_overflow_interrupt_disable() TIMSK0 &= ~(1«TOIE0);
79
84 #define TIM2_stop()           TCCR2B &= ~((1«CS22) | (1«CS21) | (1«CS20));
86 #define TIM2_overflow_16us()   TCCR2B &= ~((1«CS22) | (1«CS21)); TCCR2B |= (1«CS20);
88 #define TIM2_overflow_128us()  TCCR2B &= ~((1«CS22) | (1«CS20)); TCCR2B |= (1«CS21);
90 #define TIM2_overflow_1024us() TCCR2B &= ~(1«CS22); TCCR2B |= (1«CS21) | (1«CS20);
92 #define TIM2_overflow_4096us()    TCCR2B &= ~((1«CS21) | (1«CS20)); TCCR2B |= (1«CS22);
94 #define TIM2_overflow_16384us()    TCCR2B |= (1«CS22) | (1«CS21) | (1«CS20);
96 #define TIM2_overflow_interrupt_enable()  TIMSK2 |= (1«TOIE2);
98 #define TIM2_overflow_interrupt_disable() TIMSK2 &= ~(1«TOIE2);
99
102 #endif
```

4.13 C:/Users/gkaretka/Documents/GitHub/DE2_Project_2021-22/code/water_height_meter/water_symbols.h File Reference

Variables

- const char **init_msg_1** [340]
- const char **water_level_default** [70]
- const char **water_level_error** [70]
- const char **water_level_10** [70]
- const char **water_level_20** [70]
- const char **water_level_30** [70]
- const char **water_level_40** [70]
- const char **water_level_50** [70]
- const char **water_level_60** [70]
- const char **water_level_70** [70]
- const char **water_level_80** [70]
- const char **water_level_90** [70]
- const char **water_level_100** [70]

4.14 C:/Users/gkaretka/Documents/GitHub/DE2_Project_2021-22/code/water_height_meter/water_symbols.h

Go to the documentation of this file.

```
1
10 #ifndef WATER_SYMBOLS_H
11 #define WATER_SYMBOLS_H
12
13 const char init_msg_1[340];
14
15 const char water_level_default[70];
16 const char water_level_error[70];
17 const char water_level_10[70];
18 const char water_level_20[70];
```

```
19 const char water_level_30[70];
20 const char water_level_40[70];
21 const char water_level_50[70];
22 const char water_level_60[70];
23 const char water_level_70[70];
24 const char water_level_80[70];
25 const char water_level_90[70];
26 const char water_level_100[70];
27
28 #endif /* WATER_SYMBOLS_H_ */
```

Index