Weather Station

1.0

Generated by Doxygen 1.8.14

Contents

1 Weather station **1**

1.1 Introduction . . . 1

1.2 Description . . . 1

1.3 Compilation . . . 2

1.4 Prototype Example . . . 2

2 File Index **3**

2.1 File List . . . 3

3 File Documentation **5**

3.1 bmp180_lib.c File Reference . . . 5

3.1.1 Function Documentation . . . 5

3.1.1.1 BMP180_Calibration() . . . 5

3.1.1.2 bmp180CalcAltitude() . . . 6

3.1.1.3 bmp180Convert() . . . 6

3.1.1.4 bmp180ReadLong() . . . 7

3.1.1.5 bmp180ReadPressure() . . . 7

3.1.1.6 bmp180ReadShort() . . . 7

3.1.1.7 bmp180ReadTemp() . . . 7

3.2 bmp180_lib.h File Reference . . . 7

3.2.1 Macro Definition Documentation . . . 8

3.2.1.1 BMP180_R . . . 8

3.2.1.2 BMP180_W . . . 8

3.2.1.3 F_CPU . . . 8

3.2.1.4 OSS . . . 8
3.2.2 Function Documentation . . . 8
3.2.2.1 BMP180_Calibration() . . . 8
3.2.2.2 bmp180CalcAltitude() . . . 9
3.2.2.3 bmp180Convert() . . . 9
3.2.2.4 bmp180ReadPressure() . . . 10
3.2.2.5 bmp180ReadShort() . . . 10
3.2.2.6 bmp180ReadTemp() . . . 10
3.3 define.h File Reference . . . 10
3.3.1 Macro Definition Documentation . . . 10
3.3.1.1 F_CPU . . . 10
3.4 dht11.c File Reference . . . 10
3.4.1 Function Documentation . . . 11
3.4.1.1 getdht11() . . . 11
3.4.1.2 Receive_data11() . . . 11
3.4.1.3 Request11() . . . 12
3.4.1.4 Response11() . . . 12
3.4.2 Variable Documentation . . . 12
3.4.2.1 c . . . 12
3.5 dht11.h File Reference . . . 12
3.5.1 Macro Definition Documentation . . . 13
3.5.1.1 DHT11_BIT . . . 13
3.5.1.2 DHT11_DDR . . . 13
3.5.1.3 DHT11_PIN . . . 13
3.5.1.4 DHT11_PORT . . . 13
3.5.1.5 F_CPU . . . 13
3.5.2 Function Documentation . . . 13
3.5.2.1 getdht11() . . . 13
3.6 dht22.c File Reference . . . 14
3.6.1 Function Documentation . . . 14

3.6.1.1 getdht22() . . . 14
3.6.1.2 Receive_data22() . . . 15
3.6.1.3 Request22() . . . 15
3.6.1.4 Response22() . . . 15
3.6.2 Variable Documentation . . . 15
3.6.2.1 c22 . . . 15
3.7 dht22.h File Reference . . . 15
3.7.1 Macro Definition Documentation . . . 16
3.7.1.1 DHT22_BIT . . . 16
3.7.1.2 DHT22_DDR . . . 16
3.7.1.3 DHT22_PIN . . . 16
3.7.1.4 DHT22_PORT . . . 17
3.7.1.5 F_CPU . . . 17
3.7.2 Function Documentation . . . 17
3.7.2.1 getdht22() . . . 17
3.8 lcd.c File Reference . . . 17
3.8.1 Function Documentation . . . 18
3.8.1.1 BCD_1() . . . 18
3.8.1.2 BCD_2() . . . 18
3.8.1.3 BCD_3() . . . 19
3.8.1.4 BCD_3Int() . . . 19
3.8.1.5 BCD_4Int() . . . 19
3.8.1.6 BCD_5Int() . . . 19
3.8.1.7 BCD_GetPointerBuf() . . . 19
3.8.1.8 BCD_Uchar() . . . 19
3.8.1.9 BCD_Uint() . . . 19
3.8.1.10 BCD_Ulong() . . . 20
3.8.1.11 LCDacl() . . . 20
3.8.1.12 LCDacr() . . . 20
3.8.1.13 LCDblank() . . . 20

3.8.1.14 LCDclear() . . . 20
3.8.1.15 LCDcursor_bl() . . . 20
3.8.1.16 LCDcursor_on() . . . 20
3.8.1.17 LCDcursor_vi() . . . 21
3.8.1.18 LCDcursorl() . . . 21
3.8.1.19 LCDcursorln() . . . 21
3.8.1.20 LCDcursorOFF() . . . 21
3.8.1.21 LCDcursorr() . . . 21
3.8.1.22 LCDcursorrn() . . . 21
3.8.1.23 LCDdata() . . . 21
3.8.1.24 LCDdataXY() . . . 22
3.8.1.25 LCDGotoXY() . . . 22
3.8.1.26 LCDinit() . . . 22
3.8.1.27 LCDnblank() . . . 22
3.8.1.28 LCDresshift() . . . 22
3.8.1.29 LCDscreenl() . . . 22
3.8.1.30 LCDscreenL() . . . 23
3.8.1.31 LCDscreenln() . . . 23
3.8.1.32 LCDscreenr() . . . 23
3.8.1.33 LCDscreenR() . . . 23
3.8.1.34 LCDscreenrn() . . . 23
3.8.1.35 LCDsendString() . . . 23
3.8.1.36 LCDstring_of_flashXY() . . . 23
3.8.1.37 LCDstring_of_sramXY() . . . 24
3.8.1.38 LCDstringXY() . . . 24
3.9 lcd.h File Reference . . . 24
3.9.1 Macro Definition Documentation . . . 25
3.9.1.1 BCD_SendData . . . 25
3.9.1.2 BCD_SYM . . . 26
3.9.1.3 BCD_USE_BUF . . . 26

3.9.1.4 CDDR . . . 26
3.9.1.5 CPORT . . . 26
3.9.1.6 DB0 . . . 26
3.9.1.7 DB1 . . . 26
3.9.1.8 DB2 . . . 26
3.9.1.9 DB3 . . . 26
3.9.1.10 DB4 . . . 27
3.9.1.11 DB5 . . . 27
3.9.1.12 DB6 . . . 27
3.9.1.13 DB7 . . . 27
3.9.1.14 DDDR . . . 27
3.9.1.15 DPIN . . . 27
3.9.1.16 DPORT . . . 27
3.9.1.17 E . . . 27
3.9.1.18 LINE0 . . . 28
3.9.1.19 LINE1 . . . 28
3.9.1.20 MIRROR_NULL . . . 28
3.9.1.21 RS . . . 28
3.9.1.22 RW . . . 28
3.9.2 Function Documentation . . . 28
3.9.2.1 BCD_1() . . . 28
3.9.2.2 BCD_2() . . . 28
3.9.2.3 BCD_3() . . . 29
3.9.2.4 BCD_3Int() . . . 29
3.9.2.5 BCD_4Int() . . . 29
3.9.2.6 BCD_5Int() . . . 29
3.9.2.7 BCD_GetPointerBuf() . . . 29
3.9.2.8 BCD_Uchar() . . . 29
3.9.2.9 BCD_Uint() . . . 29
3.9.2.10 BCD_Ulong() . . . 30

3.9.2.11 LCDacl() . . . 30
3.9.2.12 LCDacr() . . . 30
3.9.2.13 LCDblank() . . . 30
3.9.2.14 LCDclear() . . . 30
3.9.2.15 LCDcursor_bl() . . . 30
3.9.2.16 LCDcursor_on() . . . 30
3.9.2.17 LCDcursor_vi() . . . 31
3.9.2.18 LCDcursorl() . . . 31
3.9.2.19 LCDcursorln() . . . 31
3.9.2.20 LCDcursorOFF() . . . 31
3.9.2.21 LCDcursorr() . . . 31
3.9.2.22 LCDcursorrn() . . . 31
3.9.2.23 LCDdata() . . . 31
3.9.2.24 LCDdataXY() . . . 32
3.9.2.25 LCDGotoXY() . . . 32
3.9.2.26 LCDinit() . . . 32
3.9.2.27 LCDnblank() . . . 32
3.9.2.28 LCDresshift() . . . 32
3.9.2.29 LCDscreenl() . . . 32
3.9.2.30 LCDscreenL() . . . 33
3.9.2.31 LCDscreenln() . . . 33
3.9.2.32 LCDscreenr() . . . 33
3.9.2.33 LCDscreenR() . . . 33
3.9.2.34 LCDscreenrn() . . . 33
3.9.2.35 LCDsendString() . . . 33
3.9.2.36 LCDstring_of_flashXY() . . . 33
3.9.2.37 LCDstring_of_sramXY() . . . 34
3.9.2.38 LCDstringXY() . . . 34
3.10 main.c File Reference . . . 34
3.10.1 Macro Definition Documentation . . . 36

3.10.1.1 F_CPU 36
3.10.2 Enumeration Type Documentation 36
3.10.2.1 state 36
3.10.3 Function Documentation 36
3.10.3.1 external_interrupt_init() 36
3.10.3.2 get_sensors_data() 37
3.10.3.3 ISR() [1/2] 37
3.10.3.4 ISR() [2/2] 37
3.10.3.5 LCD_diaplay_in() 37
3.10.3.6 LCD_diasplay_pressure() 37
3.10.3.7 LCD_display_clock() 38
3.10.3.8 LCD_display_out() 38
3.10.3.9 main() 38
3.10.3.10 setting_btn_clock() 38
3.10.3.11 sleep_ms() 38
3.10.3.12 start_init() 38
3.10.3.13 switch_state() 39
3.10.3.14 timer1_init() 39
3.10.4 Variable Documentation 39
3.10.4.1 alt 39
3.10.4.2 BMP085_calibration_int16_t 39
3.10.4.3 BMP085_calibration_uint16_t 40
3.10.4.4 clr 40
3.10.4.5 error_code 40
3.10.4.6 hour 40
3.10.4.7 humidity11 40
3.10.4.8 humidity22 40
3.10.4.9 minute 41
3.10.4.10 pBuf 41
3.10.4.11 pressure 41

3.10.4.12 second . . . 41
3.10.4.13 STATE . . . 41
3.10.4.14 state . . . 41
3.10.4.15 temperature . . . 41
3.10.4.16 temperature11 . . . 42
3.10.4.17 temperature22 . . . 42
3.11 timeout.h File Reference . . . 42
3.11.1 Macro Definition Documentation . . . 42
3.11.1.1 F_CPU . . . 42
3.12 twi_lib.h File Reference . . . 42
3.12.1 Macro Definition Documentation . . . 43
3.12.1.1 _TWI_LIB_H_ . . . 43
3.12.2 Function Documentation . . . 43
3.12.2.1 checki2cReturnCode() . . . 43
3.12.2.2 i2cGetReceivedByte() . . . 43
3.12.2.3 i2cReceiveByteACK() . . . 44
3.12.2.4 i2cReceiveByteNACK() . . . 44
3.12.2.5 i2cSendByte() . . . 44
3.12.2.6 i2cSendStart() . . . 44
3.12.2.7 i2cSendStop() . . . 44
3.12.2.8 i2cSetBitrate() . . . 44
3.12.2.9 i2cWaitForComplete() . . . 44

Index **45**

Chapter 1

Weather station

Version

1.0

Author

Viacheslav Stadnichuk

Date

29.08.2018

Warning

Prototype weather station, data is not correct

1.1 Introduction

This code developed for education aims, not for commercial using. It's a graduation project GL C/Embedded Base↩ Camp

1.2 Description

Weather station consist of two temperature and humidity sensors (DHT11 indoor, DHT22 outdoor), one barometer (BMP180 I2C) LCD display (1602A) and micro controller Arduino Uno. Code writing without using Arduino libraries (BARE METAL AVR). Sensors interrogated every 30 seconds. States of display information changes every 30 seconds and with button help. Also weather station show current time (not accurate), time settings by button. A (very) simple weather station written in C for training purposes. The code is written for the purpose of acquaintance with timers, interrupts, buttons, display and different sensors. Seconds are counted in the interrupt timer. The time setting is done using two buttons. Thanks for helping .

1.3 Compilation

Project compiles with Makefile help (make all).

1.4 Prototype Example

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

bmp180_lib.c 5
bmp180_lib.h 7
define.h 10
dht11.c 10
dht11.h 12
dht22.c 14
dht22.h 15
lcd.c 17
lcd.h 24
main.c 34
timeout.h 42
twi_lib.h 42

Chapter 3

File Documentation

3.1 bmp180_lib.c File Reference

```
#include "bmp180_lib.h"
#include "twi_lib.h"
```

Functions

- uint16_t **bmp180ReadShort** (uint8_t address, uint8_t * **error_code**)
- uint32_t **bmp180ReadLong** (uint8_t address, uint8_t * **error_code**)
- int32_t **bmp180ReadTemp** (uint8_t * **error_code**)
- int32_t **bmp180ReadPressure** (uint8_t * **error_code**)
- void **bmp180Convert** (int16_t BMP180_calibration_int16_t[ ], int16_t BMP180_calibration_uint16_t[ ], int32_t * **temperature**, int32_t * **pressure**, uint8_t * **error_code**)

  Convert and displaying values from BMP180.
- void **BMP180_Calibration** (int16_t BMP180_calibration_int16_t[ ], int16_t BMP180_calibration_uint16_t[ ], uint8_t *errorcode)

  Calibration sensor.
- int32_t **bmp180CalcAltitude** (int32_t **pressure**)

  Calculating altitude (Height above sea level)

3.1.1 Function Documentation

3.1.1.1 BMP180_Calibration()

```
void BMP180_Calibration (
            int16_t BMP180_calibration_int16_t[],
            int16_t BMP180_calibration_uint16_t[],
            uint8_t * errorcode )
```

Calibration sensor.

Parameters

in	*BMP180_calibration_int16_t*	: Calibration coefficients for pressure
in	*BMP180_calibration_uint16_t*	: Calibration coefficients for temperature
in	∗*errorcode*	: Code of error transmitted by pointer

3.1.1.2 bmp180CalcAltitude()

```
int32_t bmp180CalcAltitude (
            int32_t pressure )
```

Calculating altitude (Height above sea level)

Parameters

in	∗*pressure*	: pressure transmitted by pointer
out	*altitude*	: calculating altitude

3.1.1.3 bmp180Convert()

```
void bmp180Convert (
            int16_t BMP180_calibration_int16_t[],
            int16_t BMP180_calibration_uint16_t[],
            int32_t * temperature,
            int32_t * pressure,
            uint8_t * error_code )
```

Convert and displaying values from BMP180.

Parameters

in	*BMP180_calibration_int16_t*	: Calibration coefficients for pressure
in	*BMP180_calibration_uint16_t*	: Calibration coefficients for temperature
in	∗*errorcode*	: Code of error transmitted by pointer
in	∗*temperature*	: temperature from DHT180 transmitted by pounter
in	∗*pressure*	: pressure transmitted by pointer

3.1.1.4 bmp180ReadLong()

```
uint32_t bmp180ReadLong (
            uint8_t address,
            uint8_t * error_code )
```

3.1.1.5 bmp180ReadPressure()

```
int32_t bmp180ReadPressure (
            uint8_t * error_code )
```

3.1.1.6 bmp180ReadShort()

```
uint16_t bmp180ReadShort (
            uint8_t address,
            uint8_t * error_code )
```

3.1.1.7 bmp180ReadTemp()

```
int32_t bmp180ReadTemp (
            uint8_t * error_code )
```

3.2 bmp180_lib.h File Reference

```
#include <avr/io.h>
#include <util/delay.h>
#include <stdio.h>
#include <util/twi.h>
#include <math.h>
```

Macros

- #define **F_CPU** 16000000UL
- #define **OSS** 3

  Accuracy mode.
- #define **BMP180_R** 0xEF

  Read from sensor.
- #define **BMP180_W** 0xEE

  Write to sensor.

Functions

- void **BMP180_Calibration** (int16_t BMP180_calibration_int16_t[ ], int16_t BMP180_calibration_uint16_t[ ], uint8_t ∗errorcode)

  Calibration sensor.
- uint16_t **bmp180ReadShort** (uint8_t address, uint8_t ∗errorcode)
- int32_t **bmp180ReadTemp** (uint8_t ∗ **error_code**)
- int32_t **bmp180ReadPressure** (uint8_t ∗errorcode)
- void **bmp180Convert** (int16_t BMP180_calibration_int16_t[ ], int16_t BMP180_calibration_uint16_t[ ], int32_t ∗ **temperature**, int32_t ∗ **pressure**, uint8_t ∗ **error_code**)

  Convert and displaying values from BMP180.
- int32_t **bmp180CalcAltitude** (int32_t **pressure**)

  Calculating altitude (Height above sea level)

3.2.1 Macro Definition Documentation

3.2.1.1 BMP180_R

```
#define BMP180_R 0xEF
```

Read from sensor.

3.2.1.2 BMP180_W

```
#define BMP180_W 0xEE
```

Write to sensor.

3.2.1.3 F_CPU

```
#define F_CPU 16000000UL
```

3.2.1.4 OSS

```
#define OSS 3
```

Accuracy mode.

3.2.2 Function Documentation

3.2.2.1 BMP180_Calibration()

```
void BMP180_Calibration (
            int16_t BMP180_calibration_int16_t[],
            int16_t BMP180_calibration_uint16_t[],
            uint8_t * errorcode )
```

Calibration sensor.

Parameters

in	*BMP180_calibration_int16_t*	: Calibration coefficients for pressure
in	*BMP180_calibration_uint16_t*	: Calibration coefficients for temperature
in	∗*errorcode*	: Code of error transmitted by pointer

3.2.2.2 bmp180CalcAltitude()

```
int32_t bmp180CalcAltitude (
            int32_t pressure )
```

Calculating altitude (Height above sea level)

Parameters

in	∗*pressure*	: pressure transmitted by pointer
out	*altitude*	: calculating altitude

3.2.2.3 bmp180Convert()

```
void bmp180Convert (
            int16_t BMP180_calibration_int16_t[],
            int16_t BMP180_calibration_uint16_t[],
            int32_t * temperature,
            int32_t * pressure,
            uint8_t * error_code )
```

Convert and displaying values from BMP180.

Parameters

in	*BMP180_calibration_int16_t*	: Calibration coefficients for pressure
in	*BMP180_calibration_uint16_t*	: Calibration coefficients for temperature
in	∗*errorcode*	: Code of error transmitted by pointer
in	∗*temperature*	: temperature from DHT180 transmitted by pounter
in	∗*pressure*	: pressure transmitted by pointer

3.2.2.4 bmp180ReadPressure()

```
int32_t bmp180ReadPressure (
            uint8_t * errorcode )
```

3.2.2.5 bmp180ReadShort()

```
uint16_t bmp180ReadShort (
            uint8_t address,
            uint8_t * errorcode )
```

3.2.2.6 bmp180ReadTemp()

```
int32_t bmp180ReadTemp (
            uint8_t * error_code )
```

3.3 define.h File Reference

Macros

- #define **F_CPU** 16000000UL /* Quartz Frequency of the MCU */

3.3.1 Macro Definition Documentation

3.3.1.1 F_CPU

```
#define F_CPU 16000000UL /* Quartz Frequency of the MCU */
```

3.4 dht11.c File Reference

```
#include "dht11.h"
#include <avr/interrupt.h>
```

Functions

- void **Request11** ()
  Micro controller send start pulse/request
- void **Response11** ()
  Receive response from DHT11
- uint8_t **Receive_data11** ()
  Receive data from sensor.
- void **getdht11** (uint16_t * **temperature**, uint16_t *humidity)
  Receiving and Calculating temperature and humidity.

Variables

- uint8_t **c** =0
  Temp.

3.4.1 Function Documentation

3.4.1.1 getdht11()

```
void getdht11 (
            uint16_t * temperature,
            uint16_t * humidity )
```

Receiving and Calculating temperature and humidity.

Parameters

in	**temperature*	: temperature value transmitted by pointer
in	**humidity*	: humidity value transmitted by pointer

3.4.1.2 Receive_data11()

```
uint8_t Receive_data11 ( )
```

Receive data from sensor.

3.4.1.3 Request11()

```
void Request11 ( )
```

Micro controller send start pulse/request

3.4.1.4 Response11()

```
void Response11 ( )
```

Receive response from DHT11

3.4.2 Variable Documentation

3.4.2.1 c

```
uint8_t c =0
```

Temp.

3.5 dht11.h File Reference

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

Macros

- #define **F_CPU** 16000000UL
- #define **DHT11_BIT** 4

  Number of pin DHT11.
- #define **DHT11_PORT** PORTB

  PORT DHT11.
- #define **DHT11_DDR** DDRB

  PORT DDR DHT11.
- #define **DHT11_PIN** PINB

  Pin DHT11.

Functions

- void **getdht11** (uint16_t * **temperature11**, uint16_t * **humidity11**)
  Receiving and Calculating temperature and humidity.

3.5.1 Macro Definition Documentation

3.5.1.1 DHT11_BIT

```
#define DHT11_BIT 4
```

Number of pin DHT11.

3.5.1.2 DHT11_DDR

```
#define DHT11_DDR DDRB
```

PORT DDR DHT11.

3.5.1.3 DHT11_PIN

```
#define DHT11_PIN PINB
```

Pin DHT11.

3.5.1.4 DHT11_PORT

```
#define DHT11_PORT PORTB
```

PORT DHT11.

3.5.1.5 F_CPU

```
#define F_CPU 16000000UL
```

3.5.2 Function Documentation

3.5.2.1 getdht11()

```
void getdht11 (
            uint16_t * temperature,
            uint16_t * humidity )
```

Receiving and Calculating temperature and humidity.

Parameters

in	∗*temperature*	: temperature value transmitted by pointer
in	∗*humidity*	: humidity value transmitted by pointer

3.6 dht22.c File Reference

```
#include "dht22.h"
#include <avr/interrupt.h>
```

Functions

- void **Request22** ()

  Micro controller send start pulse/request.
- void **Response22** ()

  Receive response from DHT11.
- uint8_t **Receive_data22** ()

  Receive data from sensor.
- void **getdht22** (uint16_t ∗ **temperature22**, uint16_t ∗ **humidity22**)

  Receiving and Calculating temperature and humidity.

Variables

- uint8_t **c22** =0

  temp

3.6.1 Function Documentation

3.6.1.1 getdht22()

```
void getdht22 (
            uint16_t * temperature22,
            uint16_t * humidity22 )
```

Receiving and Calculating temperature and humidity.

Parameters

in	∗*temperature*	: temperature value transmitted by pointer
in	∗*humidity*	: humidity value transmitted by pointer

3.6.1.2 Receive_data22()

```
uint8_t Receive_data22 ( )
```

Receive data from sensor.

3.6.1.3 Request22()

```
void Request22 ( )
```

Micro controller send start pulse/request.

3.6.1.4 Response22()

```
void Response22 ( )
```

Receive response from DHT11.

3.6.2 Variable Documentation

3.6.2.1 c22

```
uint8_t c22 =0
```

temp

3.7 dht22.h File Reference

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

Macros

- #define **F_CPU** 16000000UL
- #define **DHT22_BIT** 3

  Number of pin DHT22.
- #define **DHT22_PORT** PORTB

  PORT DHT22.
- #define **DHT22_DDR** DDRB

  DDR DHT22.
- #define **DHT22_PIN** PINB

  Pin DHT22.

Functions

- void **getdht22** (uint16_t * **temperature22**, uint16_t * **humidity22**)

  Receiving and Calculating temperature and humidity.

3.7.1 Macro Definition Documentation

3.7.1.1 DHT22_BIT

```
#define DHT22_BIT 3
```

Number of pin DHT22.

3.7.1.2 DHT22_DDR

```
#define DHT22_DDR DDRB
```

DDR DHT22.

3.7.1.3 DHT22_PIN

```
#define DHT22_PIN PINB
```

Pin DHT22.

3.7.1.4 DHT22_PORT

```
#define DHT22_PORT PORTB
```

PORT DHT22.

3.7.1.5 F_CPU

```
#define F_CPU 16000000UL
```

3.7.2 Function Documentation

3.7.2.1 getdht22()

```
void getdht22 (
            uint16_t * temperature22,
            uint16_t * humidity22 )
```

Receiving and Calculating temperature and humidity.

Parameters

in	∗*temperature*	: temperature value transmitted by pointer
in	∗*humidity*	: humidity value transmitted by pointer

3.8 lcd.c File Reference

```
#include "define.h"
#include "lcd.h"
```

Functions

- void **LCDdata** (uint8_t i)
- void **LCDdataXY** (uint8_t a, uint8_t b, uint8_t **c**)
- void **LCDGotoXY** (uint8_t x, uint8_t y)
- void **LCDstringXY** (char ∗i, uint8_t x, uint8_t y)
- void **LCDsendString** (char ∗s)
- void **LCDstring_of_sramXY** (uint8_t ∗data, uint8_t x, uint8_t y)
- void **LCDstring_of_flashXY** (const uint8_t ∗FlashLoc, uint8_t x, uint8_t y)
- void **LCDinit** (void)

- void **LCDblank** (void)
- void **LCDnblank** (void)
- void **LCDclear** (void)
- void **LCDcursor_bl** (void)
- void **LCDcursor_on** (void)
- void **LCDcursor_vi** (void)
- void **LCDcursorOFF** (void)
- void **LCDacr** (void)
- void **LCDacl** (void)
- void **LCDcursorl** (void)
- void **LCDcursorr** (void)
- void **LCDcursorln** (uint8_t n)
- void **LCDcursorrn** (uint8_t n)
- void **LCDscreenl** (void)
- void **LCDscreenr** (void)
- void **LCDscreenln** (uint8_t n)
- void **LCDscreenrn** (uint8_t n)
- void **LCDscreenL** (void)
- void **LCDscreenR** (void)
- void **LCDresshift** (void)
- char * **BCD_GetPointerBuf** (void)
- void **BCD_1** (uint8_t value)
- void **BCD_2** (uint8_t value)
- void **BCD_3** (uint8_t value)
- void **BCD_3Int** (uint16_t value)
- void **BCD_4Int** (uint16_t value)
- void **BCD_5Int** (uint16_t value)
- void **BCD_Uchar** (uint8_t value)
- void **BCD_Uint** (uint16_t value)
- void **BCD_Ulong** (uint32_t value)

3.8.1 Function Documentation

3.8.1.1 BCD_1()

```
void BCD_1 (
            uint8_t value )
```

3.8.1.2 BCD_2()

```
void BCD_2 (
            uint8_t value )
```

3.8.1.3 BCD_3()

```
void BCD_3 (
            uint8_t value )
```

3.8.1.4 BCD_3Int()

```
void BCD_3Int (
            uint16_t value )
```

3.8.1.5 BCD_4Int()

```
void BCD_4Int (
            uint16_t value )
```

3.8.1.6 BCD_5Int()

```
void BCD_5Int (
            uint16_t value )
```

3.8.1.7 BCD_GetPointerBuf()

```
char* BCD_GetPointerBuf (
            void  )
```

3.8.1.8 BCD_Uchar()

```
void BCD_Uchar (
            uint8_t value )
```

3.8.1.9 BCD_Uint()

```
void BCD_Uint (
            uint16_t value )
```

3.8.1.10 BCD_Ulong()

```
void BCD_Ulong (
            uint32_t value )
```

3.8.1.11 LCDacl()

```
void LCDacl (
            void  )
```

3.8.1.12 LCDacr()

```
void LCDacr (
            void  )
```

3.8.1.13 LCDblank()

```
void LCDblank (
            void  )
```

3.8.1.14 LCDclear()

```
void LCDclear (
            void  )
```

3.8.1.15 LCDcursor_bl()

```
void LCDcursor_bl (
            void  )
```

3.8.1.16 LCDcursor_on()

```
void LCDcursor_on (
            void  )
```

3.8.1.17 LCDcursor_vi()

```
void LCDcursor_vi (
            void  )
```

3.8.1.18 LCDcursorl()

```
void LCDcursorl (
            void  )
```

3.8.1.19 LCDcursorln()

```
void LCDcursorln (
            uint8_t n )
```

3.8.1.20 LCDcursorOFF()

```
void LCDcursorOFF (
            void  )
```

3.8.1.21 LCDcursorr()

```
void LCDcursorr (
            void  )
```

3.8.1.22 LCDcursorrn()

```
void LCDcursorrn (
            uint8_t n )
```

3.8.1.23 LCDdata()

```
void LCDdata (
            uint8_t i )
```

3.8.1.24 LCDdataXY()

```
void LCDdataXY (
             uint8_t a,
             uint8_t b,
             uint8_t c )
```

3.8.1.25 LCDGotoXY()

```
void LCDGotoXY (
             uint8_t x,
             uint8_t y )
```

3.8.1.26 LCDinit()

```
void LCDinit (
             void  )
```

3.8.1.27 LCDnblank()

```
void LCDnblank (
             void  )
```

3.8.1.28 LCDresshift()

```
void LCDresshift (
             void  )
```

3.8.1.29 LCDscreenl()

```
void LCDscreenl (
             void  )
```

3.8.1.30 LCDscreenL()

```
void LCDscreenL (
            void  )
```

3.8.1.31 LCDscreenln()

```
void LCDscreenln (
            uint8_t n )
```

3.8.1.32 LCDscreenr()

```
void LCDscreenr (
            void  )
```

3.8.1.33 LCDscreenR()

```
void LCDscreenR (
            void  )
```

3.8.1.34 LCDscreenrn()

```
void LCDscreenrn (
            uint8_t n )
```

3.8.1.35 LCDsendString()

```
void LCDsendString (
            char * s )
```

3.8.1.36 LCDstring_of_flashXY()

```
void LCDstring_of_flashXY (
            const uint8_t * FlashLoc,
            uint8_t x,
            uint8_t y )
```

3.8.1.37 LCDstring_of_sramXY()

```
void LCDstring_of_sramXY (
            uint8_t * data,
            uint8_t x,
            uint8_t y )
```

3.8.1.38 LCDstringXY()

```
void LCDstringXY (
            char * i,
            uint8_t x,
            uint8_t y )
```

3.9 lcd.h File Reference

```
#include <inttypes.h>
```

Macros

- #define **DPIN** PIND
- #define **DDDR** DDRD
- #define **DPORT** PORTD
- #define **DB0** 0
- #define **DB1** 1
- #define **DB2** 2
- #define **DB3** 3
- #define **DB4** 7
- #define **DB5** 6
- #define **DB6** 5
- #define **DB7** 4
- #define **CDDR** DDRB
- #define **CPORT** PORTB
- #define **E** 0
- #define **RW** 2 /∗ R/W R / W = 1 is read from the LCD, R / W = 0 is written in the LCD ∗/
- #define **RS** 1 /∗ RS RS = 0 send the command to the LCD, RS = 1 send the data to the LCD ∗/
- #define **LINE0** 0x00
- #define **LINE1** 0x40
- #define **BCD_SendData**(data) /∗ LCD_WriteData(data) ∗/
- #define **MIRROR_NULL**
- #define **BCD_USE_BUF**
- #define **BCD_SYM**

Functions

- void **LCDGotoXY** (uint8_t, uint8_t)
- void **LCDdata** (uint8_t)
- void **LCDdataXY** (uint8_t, uint8_t, uint8_t)
- void **LCDsendString** (char *)
- void **LCDstringXY** (char *, uint8_t, uint8_t)
- void **LCDstring_of_sramXY** (uint8_t *, uint8_t, uint8_t)
- void **LCDstring_of_flashXY** (const uint8_t *, uint8_t, uint8_t)
- void **LCDinit** (void)
- void **LCDblank** (void)
- void **LCDnblank** (void)
- void **LCDclear** (void)
- void **LCDcursor_bl** (void)
- void **LCDcursor_on** (void)
- void **LCDcursor_vi** (void)
- void **LCDcursorOFF** (void)
- void **LCDacr** (void)
- void **LCDacl** (void)
- void **LCDcursorl** (void)
- void **LCDcursorr** (void)
- void **LCDcursorln** (uint8_t)
- void **LCDcursorrn** (uint8_t)
- void **LCDscreenl** (void)
- void **LCDscreenr** (void)
- void **LCDscreenln** (uint8_t)
- void **LCDscreenrn** (uint8_t)
- void **LCDscreenL** (void)
- void **LCDscreenR** (void)
- void **LCDresshift** (void)
- char * **BCD_GetPointerBuf** (void)
- void **BCD_1** (uint8_t value)
- void **BCD_2** (uint8_t value)
- void **BCD_3** (uint8_t value)
- void **BCD_3Int** (uint16_t value)
- void **BCD_4Int** (uint16_t value)
- void **BCD_5Int** (uint16_t value)
- void **BCD_Uchar** (uint8_t value)
- void **BCD_Uint** (uint16_t value)
- void **BCD_Ulong** (uint32_t value)

3.9.1 Macro Definition Documentation

3.9.1.1 BCD_SendData

```
#define BCD_SendData(
              data ) /* LCD_WriteData(data) */
```

3.9.1.2 BCD_SYM

```
#define BCD_SYM
```

3.9.1.3 BCD_USE_BUF

```
#define BCD_USE_BUF
```

3.9.1.4 CDDR

```
#define CDDR DDRB
```

3.9.1.5 CPORT

```
#define CPORT PORTB
```

3.9.1.6 DB0

```
#define DB0 0
```

3.9.1.7 DB1

```
#define DB1 1
```

3.9.1.8 DB2

```
#define DB2 2
```

3.9.1.9 DB3

```
#define DB3 3
```

3.9.1.10 DB4

```
#define DB4 7
```

3.9.1.11 DB5

```
#define DB5 6
```

3.9.1.12 DB6

```
#define DB6 5
```

3.9.1.13 DB7

```
#define DB7 4
```

3.9.1.14 DDDR

```
#define DDDR DDRD
```

3.9.1.15 DPIN

```
#define DPIN PIND
```

3.9.1.16 DPORT

```
#define DPORT PORTD
```

3.9.1.17 E

```
#define E 0
```

3.9.1.18 LINE0

```
#define LINE0 0x00
```

3.9.1.19 LINE1

```
#define LINE1 0x40
```

3.9.1.20 MIRROR_NULL

```
#define MIRROR_NULL
```

3.9.1.21 RS

```
#define RS 1 /* RS RS = 0 send the command to the LCD, RS = 1 send the data to the LCD */
```

3.9.1.22 RW

```
#define RW 2 /* R/W R / W = 1 is read from the LCD, R / W = 0 is written in the LCD */
```

3.9.2 Function Documentation

3.9.2.1 BCD_1()

```
void BCD_1 (
            uint8_t value )
```

3.9.2.2 BCD_2()

```
void BCD_2 (
            uint8_t value )
```

3.9.2.3 BCD_3()

```
void BCD_3 (
            uint8_t value )
```

3.9.2.4 BCD_3Int()

```
void BCD_3Int (
            uint16_t value )
```

3.9.2.5 BCD_4Int()

```
void BCD_4Int (
            uint16_t value )
```

3.9.2.6 BCD_5Int()

```
void BCD_5Int (
            uint16_t value )
```

3.9.2.7 BCD_GetPointerBuf()

```
char* BCD_GetPointerBuf (
            void  )
```

3.9.2.8 BCD_Uchar()

```
void BCD_Uchar (
            uint8_t value )
```

3.9.2.9 BCD_Uint()

```
void BCD_Uint (
            uint16_t value )
```

3.9.2.10 BCD_Ulong()

```
void BCD_Ulong (
            uint32_t value )
```

3.9.2.11 LCDacl()

```
void LCDacl (
            void  )
```

3.9.2.12 LCDacr()

```
void LCDacr (
            void  )
```

3.9.2.13 LCDblank()

```
void LCDblank (
            void  )
```

3.9.2.14 LCDclear()

```
void LCDclear (
            void  )
```

3.9.2.15 LCDcursor_bl()

```
void LCDcursor_bl (
            void  )
```

3.9.2.16 LCDcursor_on()

```
void LCDcursor_on (
            void  )
```

3.9.2.17 LCDcursor_vi()

```
void LCDcursor_vi (
            void  )
```

3.9.2.18 LCDcursorl()

```
void LCDcursorl (
            void  )
```

3.9.2.19 LCDcursorln()

```
void LCDcursorln (
            uint8_t  )
```

3.9.2.20 LCDcursorOFF()

```
void LCDcursorOFF (
            void  )
```

3.9.2.21 LCDcursorr()

```
void LCDcursorr (
            void  )
```

3.9.2.22 LCDcursorrn()

```
void LCDcursorrn (
            uint8_t  )
```

3.9.2.23 LCDdata()

```
void LCDdata (
            uint8_t  )
```

3.9.2.24 LCDdataXY()

```
void LCDdataXY (
             uint8_t ,
             uint8_t ,
             uint8_t  )
```

3.9.2.25 LCDGotoXY()

```
void LCDGotoXY (
             uint8_t ,
             uint8_t  )
```

3.9.2.26 LCDinit()

```
void LCDinit (
             void  )
```

3.9.2.27 LCDnblank()

```
void LCDnblank (
             void  )
```

3.9.2.28 LCDresshift()

```
void LCDresshift (
             void  )
```

3.9.2.29 LCDscreenl()

```
void LCDscreenl (
             void  )
```

3.9.2.30 LCDscreenL()

```
void LCDscreenL (
            void  )
```

3.9.2.31 LCDscreenln()

```
void LCDscreenln (
            uint8_t  )
```

3.9.2.32 LCDscreenr()

```
void LCDscreenr (
            void  )
```

3.9.2.33 LCDscreenR()

```
void LCDscreenR (
            void  )
```

3.9.2.34 LCDscreenrn()

```
void LCDscreenrn (
            uint8_t  )
```

3.9.2.35 LCDsendString()

```
void LCDsendString (
            char *  )
```

3.9.2.36 LCDstring_of_flashXY()

```
void LCDstring_of_flashXY (
            const uint8_t * ,
            uint8_t ,
            uint8_t  )
```

3.9.2.37 LCDstring_of_sramXY()

```
void LCDstring_of_sramXY (
             uint8_t * ,
             uint8_t ,
             uint8_t  )
```

3.9.2.38 LCDstringXY()

```
void LCDstringXY (
             char * ,
             uint8_t ,
             uint8_t  )
```

3.10 main.c File Reference

```
#include <avr/io.h>
#include <avr/sleep.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <stdint.h>
#include <stdbool.h>
#include <stdlib.h>
#include <avr/pgmspace.h>
#include "lcd.h"
#include "dht11.h"
#include "dht22.h"
#include "bmp180_lib.h"
```

Macros

- #define **F_CPU** 16000000UL

  Frequency CPU.

Enumerations

- enum **state** { **IN_MODE** = 0, **OUT_MODE**, **PRESSURE_MODE** }

Functions

- void **switch_state** (enum **state** new_state)

  Change state information on the display.
- **ISR** (TIMER1_OVF_vect)

  Interrupt for count seconds.
- **ISR** (INT0_vect)

  External interrupt for change mode.
- void **timer1_init** ()

  Counting seconds initialize timer.
- void **external_interrupt_init** (void)

  Initializing external interrupts INT0 on a falling edge for change mode.
- void **sleep_ms** (uint16_t ms_val)

  Enables sleep mode to reduce power consumption.
- void **start_init** ()

  Start initializing and setting for all devices (sensors, buttons, LCD)
- void **LCD_display_clock** ()

  Display clock on the screen.
- void **setting_btn_clock** ()

  Setting time by buttons.
- void **get_sensors_data** ()

  Get data from all sensors every 30 seconds.
- void **LCD_diaplay_in** ()

  Display time, temperature and humidity in door.
- void **LCD_display_out** ()

  Display time, temperature and humidity out door.
- void **LCD_diasplay_pressure** ()

  Display time, atmosphere pressure and altitude (height above sea level) in door.
- int **main** (void)

Variables

- volatile uint8_t **second**
- volatile uint8_t **minute**
- volatile uint8_t **hour**
- char * **pBuf**

  Buffer for LCD.
- uint16_t **temperature11** = 0

  Temperature from DHT11.
- uint16_t **humidity11** = 0

  Humidity from DHT22.
- uint16_t **temperature22** = 0

  Temperature from DHT22.
- uint16_t **humidity22** = 0

  Humidity from DHT22.
- int32_t **temperature** = 0

  Temperature from BMP180.
- long **pressure** = 0

  Pressure from BMP180.
- int16_t **BMP085_calibration_int16_t** [8]

Calibration BMP180.

- int16_t **BMP085_calibration_uint16_t** [3]

  Calibration BMP180.
- uint8_t **error_code** =0

  Error of BMP180.
- long **alt**

  Altitude.
- uint8_t **clr** = 0

  Counter for clear display.
- enum **state** **state**
- enum **state** **STATE** = **IN_MODE**

  Global state flag.

3.10.1 Macro Definition Documentation

3.10.1.1 F_CPU

```
#define F_CPU 16000000UL
```

Frequency CPU.

3.10.2 Enumeration Type Documentation

3.10.2.1 state

```
enum state
```

Enumerator

IN_MODE	In room data.
OUT_MODE	Out room data.
PRESSURE_MODE	Pressure and altitude.

3.10.3 Function Documentation

3.10.3.1 external_interrupt_init()

```
void external_interrupt_init (
            void  )
```

Initializing external interrupts INT0 on a falling edge for change mode.

3.10.3.2 get_sensors_data()

```
void get_sensors_data ( )
```

Get data from all sensors every 30 seconds.

3.10.3.3 ISR() [1/2]

```
ISR (
             TIMER1_OVF_vect  )
```

Interrupt for count seconds.

3.10.3.4 ISR() [2/2]

```
ISR (
             INT0_vect  )
```

External interrupt for change mode.

3.10.3.5 LCD_diaplay_in()

```
void LCD_diaplay_in ( )
```

Display time, temperature and humidity in door.

3.10.3.6 LCD_diasplay_pressure()

```
void LCD_diasplay_pressure ( )
```

Display time, atmosphere pressure and altitude (height above sea level) in door.

3.10.3.7 LCD_display_clock()

```
void LCD_display_clock ( )
```

Display clock on the screen.

3.10.3.8 LCD_display_out()

```
void LCD_display_out ( )
```

Display time, temperature and humidity out door.

3.10.3.9 main()

```
int main (
             void  )
```

3.10.3.10 setting_btn_clock()

```
void setting_btn_clock ( )
```

Setting time by buttons.

3.10.3.11 sleep_ms()

```
void sleep_ms (
             uint16_t ms_val )
```

Enables sleep mode to reduce power consumption.

Parameters

in	*ms_val*	: sleep time in milliseconds

3.10.3.12 start_init()

```
void start_init ( )
```

Start initializing and setting for all devices (sensors, buttons, LCD)

3.10.3.13 switch_state()

```
void switch_state (
            enum  state new_state )
```

Change state information on the display.

Parameters

in	*new_state*	: current state of information

3.10.3.14 timer1_init()

```
void timer1_init ( )
```

Counting seconds initialize timer.

3.10.4 Variable Documentation

3.10.4.1 alt

```
long alt
```

Altitude.

3.10.4.2 BMP085_calibration_int16_t

```
int16_t BMP085_calibration_int16_t[8]
```

Calibration BMP180.

3.10.4.3 BMP085_calibration_uint16_t

```
int16_t BMP085_calibration_uint16_t[3]
```

Calibration BMP180.

3.10.4.4 clr

```
uint8_t clr = 0
```

Counter for clear display.

3.10.4.5 error_code

```
uint8_t error_code =0
```

Error of BMP180.

3.10.4.6 hour

```
volatile uint8_t hour
```

3.10.4.7 humidity11

```
uint16_t humidity11 = 0
```

Humidity from DHT22.

3.10.4.8 humidity22

```
uint16_t humidity22 = 0
```

Humidity from DHT22.

3.10.4.9 minute

```
volatile uint8_t minute
```

3.10.4.10 pBuf

```
char* pBuf
```

Buffer for LCD.

3.10.4.11 pressure

```
long pressure = 0
```

Pressure from BMP180.

3.10.4.12 second

```
volatile uint8_t second
```

3.10.4.13 STATE

```
enum state STATE = IN_MODE
```

Global state flag.

3.10.4.14 state

```
enum state state
```

3.10.4.15 temperature

```
int32_t temperature = 0
```

Temperature from BMP180.

3.10.4.16 temperature11

```
uint16_t temperature11 = 0
```

Temperature from DHT11.

3.10.4.17 temperature22

```
uint16_t temperature22 = 0
```

Temperature from DHT22.

3.11 timeout.h File Reference

```
#include <util/delay.h>
```

Macros

- #define **F_CPU** 16000000UL

3.11.1 Macro Definition Documentation

3.11.1.1 F_CPU

```
#define F_CPU 16000000UL
```

3.12 twi_lib.h File Reference

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

Macros

- #define **_TWI_LIB_H_** 1

Functions

- void **i2cSetBitrate** (uint16_t bitratekHz)

  Set the I2C transaction bitrate (in KHz)
- void **i2cSendStart** (void)

  Send an I2C start condition in Master mode or repeated start condition.
- uint8_t **i2cSendStop** (void)

  Send an I2C stop condition in Master mode.
- void **i2cSendByte** (unsigned char data)

  Send an (address|R/W) combination or a data byte over I2C.
- void **i2cReceiveByteACK** (void)

  Receive a data byte over I2C.
- void **i2cReceiveByteNACK** (void)

  Receive a data byte over I2C.
- uint8_t **i2cGetReceivedByte** (void)

  get received byte back
- uint8_t **i2cWaitForComplete** (void)

  Wait for current I2C operation to complete.
- uint8_t **checki2cReturnCode** (uint8_t expected_return_code)

  Check for expected error code.

3.12.1 Macro Definition Documentation

3.12.1.1 _TWI_LIB_H_

```
#define _TWI_LIB_H_ 1
```

3.12.2 Function Documentation

3.12.2.1 checki2cReturnCode()

```
uint8_t checki2cReturnCode (
            uint8_t expected_return_code )
```

Check for expected error code.

3.12.2.2 i2cGetReceivedByte()

```
uint8_t i2cGetReceivedByte (
            void  )
```

get received byte back

3.12.2.3 i2cReceiveByteACK()

```
void i2cReceiveByteACK (
            void  )
```

Receive a data byte over I2C.

3.12.2.4 i2cReceiveByteNACK()

```
void i2cReceiveByteNACK (
            void  )
```

Receive a data byte over I2C.

3.12.2.5 i2cSendByte()

```
void i2cSendByte (
            unsigned char data )
```

Send an (address|R/W) combination or a data byte over I2C.

3.12.2.6 i2cSendStart()

```
void i2cSendStart (
            void  )
```

Send an I2C start condition in Master mode or repeated start condition.

3.12.2.7 i2cSendStop()

```
uint8_t i2cSendStop (
            void  )
```

Send an I2C stop condition in Master mode.

3.12.2.8 i2cSetBitrate()

```
void i2cSetBitrate (
            uint16_t bitratekHz )
```

Set the I2C transaction bitrate (in KHz)

3.12.2.9 i2cWaitForComplete()

```
uint8_t i2cWaitForComplete (
            void  )
```

Wait for current I2C operation to complete.

Index

_TWI_LIB_H_
twi_lib.h, 43

alt
main.c, 39

BCD_1
lcd.c, 18
lcd.h, 28
BCD_2
lcd.c, 18
lcd.h, 28
BCD_3
lcd.c, 18
lcd.h, 28
BCD_3Int
lcd.c, 19
lcd.h, 29
BCD_4Int
lcd.c, 19
lcd.h, 29
BCD_5Int
lcd.c, 19
lcd.h, 29
BCD_GetPointerBuf
lcd.c, 19
lcd.h, 29
BCD_SYM
lcd.h, 25
BCD_SendData
lcd.h, 25
BCD_USE_BUF
lcd.h, 26
BCD_Uchar
lcd.c, 19
lcd.h, 29
BCD_Uint
lcd.c, 19
lcd.h, 29
BCD_Ulong
lcd.c, 19
lcd.h, 29
BMP085_calibration_int16_t
main.c, 39
BMP085_calibration_uint16_t
main.c, 39
BMP180_Calibration
bmp180_lib.c, 5
bmp180_lib.h, 8
BMP180_R
bmp180_lib.h, 8
BMP180_W
bmp180_lib.h, 8
bmp180_lib.c, 5
BMP180_Calibration, 5
bmp180CalcAltitude, 6
bmp180Convert, 6
bmp180ReadLong, 6
bmp180ReadPressure, 7
bmp180ReadShort, 7
bmp180ReadTemp, 7
bmp180_lib.h, 7
BMP180_Calibration, 8
BMP180_R, 8
BMP180_W, 8
bmp180CalcAltitude, 9
bmp180Convert, 9
bmp180ReadPressure, 9
bmp180ReadShort, 10
bmp180ReadTemp, 10
F_CPU, 8
OSS, 8
bmp180CalcAltitude
bmp180_lib.c, 6
bmp180_lib.h, 9
bmp180Convert
bmp180_lib.c, 6
bmp180_lib.h, 9
bmp180ReadLong
bmp180_lib.c, 6
bmp180ReadPressure
bmp180_lib.c, 7
bmp180_lib.h, 9
bmp180ReadShort
bmp180_lib.c, 7
bmp180_lib.h, 10
bmp180ReadTemp
bmp180_lib.c, 7
bmp180_lib.h, 10

c
dht11.c, 12
c22
dht22.c, 15
CDDR
lcd.h, 26
CPORT
lcd.h, 26
checki2cReturnCode
twi_lib.h, 43

clr
main.c, 40

DB0
lcd.h, 26
DB1
lcd.h, 26
DB2
lcd.h, 26
DB3
lcd.h, 26
DB4
lcd.h, 26
DB5
lcd.h, 27
DB6
lcd.h, 27
DB7
lcd.h, 27
DDDR
lcd.h, 27
DHT11_BIT
dht11.h, 13
DHT11_DDR
dht11.h, 13
DHT11_PIN
dht11.h, 13
DHT11_PORT
dht11.h, 13
DHT22_BIT
dht22.h, 16
DHT22_DDR
dht22.h, 16
DHT22_PIN
dht22.h, 16
DHT22_PORT
dht22.h, 16
DPIN
lcd.h, 27
DPORT
lcd.h, 27
define.h, 10
F_CPU, 10
dht11.c, 10
c, 12
getdht11, 11
Receive_data11, 11
Request11, 11
Response11, 12
dht11.h, 12
DHT11_BIT, 13
DHT11_DDR, 13
DHT11_PIN, 13
DHT11_PORT, 13
F_CPU, 13
getdht11, 13
dht22.c, 14
c22, 15
getdht22, 14
Receive_data22, 15
Request22, 15
Response22, 15
dht22.h, 15
DHT22_BIT, 16
DHT22_DDR, 16
DHT22_PIN, 16
DHT22_PORT, 16
F_CPU, 17
getdht22, 17

E
lcd.h, 27
error_code
main.c, 40
external_interrupt_init
main.c, 36

F_CPU
bmp180_lib.h, 8
define.h, 10
dht11.h, 13
dht22.h, 17
main.c, 36
timeout.h, 42

get_sensors_data
main.c, 37
getdht11
dht11.c, 11
dht11.h, 13
getdht22
dht22.c, 14
dht22.h, 17

hour
main.c, 40
humidity11
main.c, 40
humidity22
main.c, 40

i2cGetReceivedByte
twi_lib.h, 43
i2cReceiveByteACK
twi_lib.h, 43
i2cReceiveByteNACK
twi_lib.h, 44
i2cSendByte
twi_lib.h, 44
i2cSendStart
twi_lib.h, 44
i2cSendStop
twi_lib.h, 44
i2cSetBitrate
twi_lib.h, 44
i2cWaitForComplete
twi_lib.h, 44
ISR

main.c, 37

LCD_diaplay_in
main.c, 37
LCD_diasplay_pressure
main.c, 37
LCD_display_clock
main.c, 37
LCD_display_out
main.c, 38
LCDGotoXY
lcd.c, 22
lcd.h, 32
LCDacl
lcd.c, 20
lcd.h, 30
LCDacr
lcd.c, 20
lcd.h, 30
LCDblank
lcd.c, 20
lcd.h, 30
LCDclear
lcd.c, 20
lcd.h, 30
LCDcursor_bl
lcd.c, 20
lcd.h, 30
LCDcursor_on
lcd.c, 20
lcd.h, 30
LCDcursor_vi
lcd.c, 20
lcd.h, 30
LCDcursorOFF
lcd.c, 21
lcd.h, 31
LCDcursorl
lcd.c, 21
lcd.h, 31
LCDcursorln
lcd.c, 21
lcd.h, 31
LCDcursorr
lcd.c, 21
lcd.h, 31
LCDcursorrn
lcd.c, 21
lcd.h, 31
LCDdata
lcd.c, 21
lcd.h, 31
LCDdataXY
lcd.c, 21
lcd.h, 31
LCDinit
lcd.c, 22
lcd.h, 32
LCDnblank
lcd.c, 22
lcd.h, 32
LCDresshift
lcd.c, 22
lcd.h, 32
LCDscreenL
lcd.c, 22
lcd.h, 32
LCDscreenl
lcd.c, 22
lcd.h, 32
LCDscreenln
lcd.c, 23
lcd.h, 33
LCDscreenR
lcd.c, 23
lcd.h, 33
LCDscreenr
lcd.c, 23
lcd.h, 33
LCDscreenrn
lcd.c, 23
lcd.h, 33
LCDsendString
lcd.c, 23
lcd.h, 33
LCDstring_of_flashXY
lcd.c, 23
lcd.h, 33
LCDstring_of_sramXY
lcd.c, 23
lcd.h, 33
LCDstringXY
lcd.c, 24
lcd.h, 34
LINE0
lcd.h, 27
LINE1
lcd.h, 28
lcd.c, 17
BCD_1, 18
BCD_2, 18
BCD_3, 18
BCD_3Int, 19
BCD_4Int, 19
BCD_5Int, 19
BCD_GetPointerBuf, 19
BCD_Uchar, 19
BCD_Uint, 19
BCD_Ulong, 19
LCDGotoXY, 22
LCDacl, 20
LCDacr, 20
LCDblank, 20
LCDclear, 20
LCDcursor_bl, 20
LCDcursor_on, 20
LCDcursor_vi, 20

LCDcursorOFF, 21
LCDcursorl, 21
LCDcursorln, 21
LCDcursorr, 21
LCDcursorrn, 21
LCDdata, 21
LCDdataXY, 21
LCDinit, 22
LCDnblank, 22
LCDresshift, 22
LCDscreenL, 22
LCDscreenl, 22
LCDscreenln, 23
LCDscreenR, 23
LCDscreenr, 23
LCDscreenrn, 23
LCDsendString, 23
LCDstring_of_flashXY, 23
LCDstring_of_sramXY, 23
LCDstringXY, 24
lcd.h, 24
BCD_1, 28
BCD_2, 28
BCD_3, 28
BCD_3Int, 29
BCD_4Int, 29
BCD_5Int, 29
BCD_GetPointerBuf, 29
BCD_SYM, 25
BCD_SendData, 25
BCD_USE_BUF, 26
BCD_Uchar, 29
BCD_Uint, 29
BCD_Ulong, 29
CDDR, 26
CPORT, 26
DB0, 26
DB1, 26
DB2, 26
DB3, 26
DB4, 26
DB5, 27
DB6, 27
DB7, 27
DDDR, 27
DPIN, 27
DPORT, 27
E, 27
LCDGotoXY, 32
LCDacl, 30
LCDacr, 30
LCDblank, 30
LCDclear, 30
LCDcursor_bl, 30
LCDcursor_on, 30
LCDcursor_vi, 30
LCDcursorOFF, 31
LCDcursorl, 31
LCDcursorln, 31
LCDcursorr, 31
LCDcursorrn, 31
LCDdata, 31
LCDdataXY, 31
LCDinit, 32
LCDnblank, 32
LCDresshift, 32
LCDscreenL, 32
LCDscreenl, 32
LCDscreenln, 33
LCDscreenR, 33
LCDscreenr, 33
LCDscreenrn, 33
LCDsendString, 33
LCDstring_of_flashXY, 33
LCDstring_of_sramXY, 33
LCDstringXY, 34
LINE0, 27
LINE1, 28
MIRROR_NULL, 28
RS, 28
RW, 28

MIRROR_NULL
lcd.h, 28
main
main.c, 38
main.c, 34
alt, 39
BMP085_calibration_int16_t, 39
BMP085_calibration_uint16_t, 39
clr, 40
error_code, 40
external_interrupt_init, 36
F_CPU, 36
get_sensors_data, 37
hour, 40
humidity11, 40
humidity22, 40
ISR, 37
LCD_diaplay_in, 37
LCD_diasplay_pressure, 37
LCD_display_clock, 37
LCD_display_out, 38
main, 38
minute, 40
pBuf, 41
pressure, 41
STATE, 41
second, 41
setting_btn_clock, 38
sleep_ms, 38
start_init, 38
state, 36, 41
switch_state, 39
temperature, 41
temperature11, 41
temperature22, 42

timer1_init, 39
minute
main.c, 40

OSS
bmp180_lib.h, 8

pBuf
main.c, 41
pressure
main.c, 41

Receive_data11
dht11.c, 11
Receive_data22
dht22.c, 15
Request11
dht11.c, 11
Request22
dht22.c, 15
Response11
dht11.c, 12
Response22
dht22.c, 15
RS
lcd.h, 28
RW
lcd.h, 28

STATE
main.c, 41
second
main.c, 41
setting_btn_clock
main.c, 38
sleep_ms
main.c, 38
start_init
main.c, 38
state
main.c, 36, 41
switch_state
main.c, 39

temperature
main.c, 41
temperature11
main.c, 41
temperature22
main.c, 42
timeout.h, 42
F_CPU, 42
timer1_init
main.c, 39
twi_lib.h, 42
_TWI_LIB_H_, 43
checki2cReturnCode, 43
i2cGetReceivedByte, 43
i2cReceiveByteACK, 43
i2cReceiveByteNACK, 44
i2cSendByte, 44
i2cSendStart, 44
i2cSendStop, 44
i2cSetBitrate, 44
i2cWaitForComplete, 44