

Weather Sensors Reference Manual

Generated by Doxygen 1.8.14

Sun Oct 14 2018 20:12:43

Contents

| | | |
|----------|--|----------|
| 1 | WeatherSensors | 1 |
| 2 | Class Index | 3 |
| 2.1 | Class List | 3 |
| 3 | File Index | 5 |
| 3.1 | File List | 5 |
| 4 | Class Documentation | 7 |
| 4.1 | Sensor_BME280 Class Reference | 7 |
| 4.1.1 | Detailed Description | 8 |
| 4.1.2 | Constructor & Destructor Documentation | 8 |
| 4.1.2.1 | Sensor_BME280() | 8 |
| 4.1.3 | Member Function Documentation | 8 |
| 4.1.3.1 | absolutePressure() | 8 |
| 4.1.3.2 | altitude() [1/2] | 9 |
| 4.1.3.3 | altitude() [2/2] | 9 |
| 4.1.3.4 | begin() | 10 |
| 4.1.3.5 | get() | 10 |
| 4.1.3.6 | humidity() | 10 |
| 4.1.3.7 | pressure() | 11 |
| 4.1.3.8 | setPowerMode() | 11 |
| 4.1.3.9 | temperature() | 11 |
| 4.1.3.10 | WhoAmI() | 12 |
| 4.2 | Sensor_BMP280 Class Reference | 12 |

| | | |
|---------|--|----|
| 4.2.1 | Detailed Description | 13 |
| 4.2.2 | Constructor & Destructor Documentation | 13 |
| 4.2.2.1 | Sensor_BMP280() | 13 |
| 4.2.3 | Member Function Documentation | 13 |
| 4.2.3.1 | absolutePressure() | 13 |
| 4.2.3.2 | altitude() [1/2] | 14 |
| 4.2.3.3 | altitude() [2/2] | 14 |
| 4.2.3.4 | begin() | 15 |
| 4.2.3.5 | get() | 15 |
| 4.2.3.6 | pressure() | 15 |
| 4.2.3.7 | setPowerMode() | 15 |
| 4.2.3.8 | temperature() | 16 |
| 4.2.3.9 | WhoAmI() | 16 |
| 4.3 | Sensor_HDC1000 Class Reference | 16 |
| 4.3.1 | Detailed Description | 17 |
| 4.3.2 | Constructor & Destructor Documentation | 17 |
| 4.3.2.1 | Sensor_HDC1000() | 17 |
| 4.3.3 | Member Function Documentation | 17 |
| 4.3.3.1 | begin() | 18 |
| 4.3.3.2 | humidity() | 18 |
| 4.3.3.3 | setPowerMode() | 18 |
| 4.3.3.4 | temperature() | 18 |
| 4.4 | Sensor_OPT3001 Class Reference | 19 |
| 4.4.1 | Detailed Description | 19 |
| 4.4.2 | Member Function Documentation | 19 |
| 4.4.2.1 | begin() | 19 |
| 4.4.2.2 | light() | 20 |
| 4.4.2.3 | setPowerMode() | 20 |
| 4.4.2.4 | WhoAmI() | 20 |
| 4.5 | Sensor_TMP007 Class Reference | 20 |
| 4.5.1 | Detailed Description | 21 |
| 4.5.2 | Constructor & Destructor Documentation | 21 |
| 4.5.2.1 | Sensor_TMP007() | 21 |
| 4.5.3 | Member Function Documentation | 21 |
| 4.5.3.1 | begin() | 22 |
| 4.5.3.2 | external() | 22 |
| 4.5.3.3 | internal() | 22 |
| 4.5.3.4 | setPowerMode() | 22 |
| 4.5.3.5 | WhoAmI() | 23 |
| 4.6 | unit_conversion_s Struct Reference | 23 |
| 4.6.1 | Detailed Description | 23 |

| | |
|---|-----------|
| 5 File Documentation | 25 |
| 5.1 Sensor_BME280.h File Reference | 25 |
| 5.1.1 Detailed Description | 26 |
| 5.2 Sensor_BMP280.h File Reference | 27 |
| 5.2.1 Detailed Description | 27 |
| 5.3 Sensor_HDC1000.h File Reference | 28 |
| 5.3.1 Detailed Description | 29 |
| 5.4 Sensor_OPT3001.h File Reference | 30 |
| 5.4.1 Detailed Description | 31 |
| 5.5 Sensor_TMP007.h File Reference | 32 |
| 5.5.1 Detailed Description | 33 |
| 5.6 Sensor_Units.h File Reference | 34 |
| 5.6.1 Detailed Description | 35 |
| 5.6.2 Function Documentation | 36 |
| 5.6.2.1 conversion() | 36 |
| 5.6.2.2 symbolChar() | 36 |
| 5.6.2.3 symbolString() | 36 |
| 5.7 WeatherSensors.ino File Reference | 38 |
| 5.7.1 Detailed Description | 39 |
| 5.8 Wire_Utilities.h File Reference | 39 |
| 5.8.1 Detailed Description | 40 |
| 5.8.2 Function Documentation | 40 |
| 5.8.2.1 delayBusy() | 40 |
| 5.8.2.2 readRegister16() | 41 |
| 5.8.2.3 readRegister8() | 41 |
| 5.8.2.4 writeRegister16() | 42 |
| 5.8.2.5 writeRegister8() | 42 |
| Index | 43 |

Chapter 1

WeatherSensors

Library for the weather sensors of the Sensors BoosterPack and the CC1350 SensorTag

Developed with [embedXcode+](#)

Author

Rei Vilo

<http://embeddedcomputing.weebly.com>

Date

12 Nov 2016

Version

103

Copyright

(c) Rei Vilo, 2016-2018

CC = BY SA NC

See also

ReadMe.txt for references

Chapter 2

Class Index

2.1 Class List

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

| | | |
|------------------------------------|--|----|
| Sensor_BME280 | | |
| Class for sensor BME280 | | 7 |
| Sensor_BMP280 | | |
| Class for sensor BMP280 | | 12 |
| Sensor_HDC1000 | | |
| Class for sensor HDC1000 | | 16 |
| Sensor_OPT3001 | | |
| Class for sensor OPT3001 | | 19 |
| Sensor_TMP007 | | |
| Class for sensor TMP007 | | 20 |
| unit_conversion_s | | |
| Units | | 23 |

Chapter 3

File Index

3.1 File List

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

| | |
|------------------------------------|----|
| rtosTasks.h | ?? |
| Sensor_BME280.h | |
| Library header for BME280 sensor | 25 |
| Sensor_BMP280.h | |
| Library header for BMP280 sensor | 27 |
| Sensor_HDC1000.h | |
| Library header | 28 |
| Sensor_OPT3001.h | |
| Library header for OPT3001 sensor | 30 |
| Sensor_TMP007.h | |
| Library header for TMP007 sensor | 32 |
| Sensor_Units.h | |
| Library header | 34 |
| WeatherSensors.ino | |
| Main sketch | 38 |
| Wire_Uilities.h | |
| Library header | 39 |

Chapter 4

Class Documentation

4.1 Sensor_BME280 Class Reference

Class for sensor BME280.

```
#include <Sensor_BME280.h>
```

Public Member Functions

- [Sensor_BME280](#) (uint8_t address=0x77)
Constructor.
- void [begin](#) ()
Initialisation.
- String [WhoAml](#) ()
Who am I?
- uint8_t [get](#) ()
Acquire data.
- float [temperature](#) ()
Return temperature.
- float [humidity](#) ()
Return relative humidity.
- float [pressure](#) ()
Return pressure, relative to current altitude.
- float [absolutePressure](#) (float altitudeMeters=50.0)
Return absolute pressure, equivalent at sea level.
- float [altitude](#) (float seaLevelPressure=1013.250)
Return altitude based on pressure.
- float [altitude](#) (float referencePressure=1013.250, float referenceAltitude=0.0)
Return altitude based on reference pressure and altitude.
- void [setPowerMode](#) (uint8_t mode=LOW)
Set power mode.

4.1.1 Detailed Description

Class for sensor BME280.

Combined temperature, humidity and pressure sensor

See also

http://www.bosch-sensortec.com/de/homepage/products_3/environmental_sensors_1/bme280/bme280_1

4.1.2 Constructor & Destructor Documentation

4.1.2.1 Sensor_BME280()

```
Sensor_BME280::Sensor_BME280 (
    uint8_t address = 0x77 )
```

Constructor.

Parameters

| | |
|----------------|--------------------------------|
| <i>address</i> | default = BME280_SLAVE_ADDRESS |
|----------------|--------------------------------|

4.1.3 Member Function Documentation

4.1.3.1 absolutePressure()

```
float Sensor_BME280::absolutePressure (
    float altitudeMeters = 50.0 )
```

Return absolute pressure, equivalent at sea level.

Parameters

| | |
|-----------------------|----------------------------|
| <i>altitudeMeters</i> | current altitude, in meter |
|-----------------------|----------------------------|

Returns

absolute pressure at sea level, in hPa

Note

Use [conversion\(\)](#) for another unit

4.1.3.2 altitude() [1/2]

```
float Sensor_BME280::altitude (
    float seaLevelPressure = 1013.250 )
```

Return altitude based on pressure.

Parameters

| | |
|-------------------------|-------------------------------|
| <i>seaLevelPressure</i> | pressure at sea level, in hPa |
|-------------------------|-------------------------------|

Returns

altitude, in meter

Note

Use [conversion\(\)](#) for another unit

4.1.3.3 altitude() [2/2]

```
float Sensor_BME280::altitude (
    float referencePressure = 1013.250,
    float referenceAltitude = 0.0 )
```

Return altitude based on reference pressure and altitude.

Parameters

| | |
|--------------------------|------------------------------|
| <i>referencePressure</i> | reference pressure, in hPa |
| <i>referenceAltitude</i> | reference altitude, in meter |

Returns

altitude in meter

Note

The reference is a measure of the pressure at a known altitude.
Use [conversion\(\)](#) for another unit

4.1.3.4 begin()

```
void Sensor_BME280::begin ( )
```

Initialisation.

Parameters

| | |
|---------------|----------|
| <i>number</i> | of reads |
|---------------|----------|

Note

See Table # of the BME280 data-sheet

xxxxx.011 Default = 0x00 _____.001 Humidity oversampling x1

4.1.3.5 get()

```
uint8_t Sensor_BME280::get ( )
```

Acquire data.

Returns

0 if success, error code otherwise

```
do
{
    delay(100);
    result = myBME280.get();
    count++;
}
while ((result > 0) and (count < 8));
```

4.1.3.6 humidity()

```
float Sensor_BME280::humidity ( )
```

Return relative humidity.

Returns

relative humidity, in %

4.1.3.7 pressure()

```
float Sensor_BME280::pressure ( )
```

Return pressure, relative to current altitude.

Returns

pressure, in hPa

Note

Use [conversion\(\)](#) for another unit

4.1.3.8 setPowerMode()

```
void Sensor_BME280::setPowerMode (
    uint8_t mode = LOW )
```

Set power mode.

Parameters

| | |
|-------------|-----------------------------------|
| <i>mode</i> | default=LOW=sleep, HIGH=activated |
|-------------|-----------------------------------|

4.1.3.9 temperature()

```
float Sensor_BME280::temperature ( )
```

Return temperature.

Returns

temperature, in °K

Note

Use [conversion\(\)](#) for another unit

4.1.3.10 WhoAml()

```
String Sensor_BME280::WhoAmI ( )
```

Who am I?

Returns

Who am I? string

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

- [Sensor_BME280.h](#)
- [Sensor_BME280.cpp](#)

4.2 Sensor_BMP280 Class Reference

Class for sensor BMP280.

```
#include <Sensor_BMP280.h>
```

Public Member Functions

- [Sensor_BMP280](#) (uint8_t address=0x77)
Constructor.
- void [begin](#) ()
Initialisation.
- String [WhoAml](#) ()
Who am I?
- uint8_t [get](#) ()
Acquire data.
- float [temperature](#) ()
Return temperature.
- float [pressure](#) ()
Return pressure, relative to current altitude.
- float [absolutePressure](#) (float altitudeMeters=50.0)
Return absolute pressure, equivalent at sea level.
- float [altitude](#) (float seaLevelPressure=1013.250)
Return altitude based on pressure.
- float [altitude](#) (float referencePressure=1013.250, float referenceAltitude=0.0)
Return altitude based on reference pressure and altitude.
- void [setPowerMode](#) (uint8_t mode=LOW)
Set power mode.

4.2.1 Detailed Description

Class for sensor BMP280.

Combined humidity and pressure sensor

See also

http://www.bosch-sensortec.com/en/bst/products/all_products/bmp280

4.2.2 Constructor & Destructor Documentation

4.2.2.1 Sensor_BMP280()

```
Sensor_BMP280::Sensor_BMP280 (
    uint8_t address = 0x77 )
```

Constructor.

Parameters

| | |
|----------------|-------------------|
| <i>address</i> | I2C slave address |
|----------------|-------------------|

Note

Valid addresses are 0x76..0x78

4.2.3 Member Function Documentation

4.2.3.1 absolutePressure()

```
float Sensor_BMP280::absolutePressure (
    float altitudeMeters = 50.0 )
```

Return absolute pressure, equivalent at sea level.

Parameters

| | |
|-----------------------|----------------------------|
| <i>altitudeMeters</i> | current altitude, in meter |
|-----------------------|----------------------------|

Returns

absolute pressure at sea level, in hPa

Note

Use [conversion\(\)](#) for another unit

4.2.3.2 altitude() [1/2]

```
float Sensor_BMP280::altitude (
    float seaLevelPressure = 1013.250 )
```

Return altitude based on pressure.

Parameters

| | |
|-------------------------|-------------------------------|
| <i>seaLevelPressure</i> | pressure at sea level, in hPa |
|-------------------------|-------------------------------|

Returns

altitude, in meter

Note

Use [conversion\(\)](#) for another unit

4.2.3.3 altitude() [2/2]

```
float Sensor_BMP280::altitude (
    float referencePressure = 1013.250,
    float referenceAltitude = 0.0 )
```

Return altitude based on reference pressure and altitude.

Parameters

| | |
|--------------------------|------------------------------|
| <i>referencePressure</i> | reference pressure, in hPa |
| <i>referenceAltitude</i> | reference altitude, in meter |

Returns

altitude in meter

Note

The reference is a measure of the pressure at a known altitude.
Use [conversion\(\)](#) for another unit

4.2.3.4 begin()

```
void Sensor_BMP280::begin ( )
```

Initialisation.

Parameters

| | |
|---------------|----------|
| <i>number</i> | of reads |
|---------------|----------|

Note

See Table # of the BMP280 data-sheet

4.2.3.5 get()

```
uint8_t Sensor_BMP280::get ( )
```

Acquire data.

Returns

0 if success, error code otherwise

```
do
{
    delay(100);
    result = myBMP280.get();
    count++;
}
while ((result > 0) and (count < 8));
```

4.2.3.6 pressure()

```
float Sensor_BMP280::pressure ( )
```

Return pressure, relative to current altitude.

Returns

pressure, in hPa

Note

Use [conversion\(\)](#) for another unit

4.2.3.7 setPowerMode()

```
void Sensor_BMP280::setPowerMode (
    uint8_t mode = LOW )
```

Set power mode.

Parameters

| | |
|-------------|-----------------------------------|
| <i>mode</i> | default=LOW=sleep, HIGH=activated |
|-------------|-----------------------------------|

4.2.3.8 temperature()

```
float Sensor_BMP280::temperature ( )
```

Return temperature.

Returns

temperature, in °K

Note

Use [conversion\(\)](#) for another unit

4.2.3.9 WhoAmI()

```
String Sensor_BMP280::WhoAmI ( )
```

Who am I?

Returns

Who am I? string

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

- [Sensor_BMP280.h](#)
- [Sensor_BMP280.cpp](#)

4.3 Sensor_HDC1000 Class Reference

Class for sensor HDC1000.

```
#include <Sensor_HDC1000.h>
```

Public Member Functions

- [Sensor_HDC1000](#) (uint8_t address=0x43)
Constructor.
- void [begin](#) (uint8_t configuration=0b00010101)
Initialisation.
- void [get](#) ()
Acquisition.
- double [temperature](#) ()
Measure.
- double [humidity](#) ()
Measure.
- void [setPowerMode](#) (uint8_t mode=LOW)
Manage power.

4.3.1 Detailed Description

Class for sensor HDC1000.

Temperature and Humidity Sensor

See also

<http://www.ti.com/product/HDC1000>

4.3.2 Constructor & Destructor Documentation

4.3.2.1 Sensor_HDC1000()

```
Sensor_HDC1000::Sensor_HDC1000 (
    uint8_t address = 0x43 )
```

Constructor.

Parameters

| | |
|----------------|-------------------|
| <i>address</i> | I2C slave address |
|----------------|-------------------|

Note

Valid addresses are 0x40..0x43

4.3.3 Member Function Documentation

4.3.3.1 begin()

```
void Sensor_HDC1000::begin (
    uint8_t configuration = 0b00010101 )
```

Initialisation.

Parameters

| | |
|----------------------|--------------------------|
| <i>configuration</i> | default=HDC1000_SETTINGS |
|----------------------|--------------------------|

4.3.3.2 humidity()

```
double Sensor_HDC1000::humidity ( )
```

Measure.

Returns

Relative humidity in %

4.3.3.3 setPowerMode()

```
void Sensor_HDC1000::setPowerMode (
    uint8_t mode = LOW )
```

Manage power.

Parameters

| | |
|-------------|--------------------------|
| <i>mode</i> | LOW=default=off, HIGH=on |
|-------------|--------------------------|

4.3.3.4 temperature()

```
double Sensor_HDC1000::temperature ( )
```

Measure.

Returns

Temperature in °K

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

- [Sensor_HDC1000.h](#)
- [Sensor_HDC1000.cpp](#)

4.4 Sensor_OPT3001 Class Reference

Class for sensor OPT3001.

```
#include <Sensor_OPT3001.h>
```

Public Member Functions

- [Sensor_OPT3001](#) (uint8_t address=0x47)
Constructor.
- void [begin](#) (uint16_t configuration=0xc410, uint8_t interruptPin=11)
Initialisation.
- String [WhoAmI](#) ()
Who Am I?
- void [get](#) ()
Acquisition.
- float [light](#) ()
Measure.
- void [setPowerMode](#) (uint8_t mode=LOW)
Manage power.

4.4.1 Detailed Description

Class for sensor OPT3001.

Digital Ambient Light Sensor (ALS) with High Precision Human Eye Response

See also

<http://www.ti.com/product/OPT3001>

4.4.2 Member Function Documentation

4.4.2.1 begin()

```
void Sensor_OPT3001::begin (
    uint16_t configuration = 0xc410,
    uint8_t interruptPin = 11 )
```

Initialisation.

Parameters

| | |
|----------------------|--|
| <i>configuration</i> | default = 100 ms, OPT3001_100_MS or OPT3001_800_MS |
| <i>interruptPin</i> | default = 11 |

4.4.2.2 light()

```
float Sensor_OPT3001::light ( )
```

Measure.

Returns

light in lux

4.4.2.3 setPowerMode()

```
void Sensor_OPT3001::setPowerMode (
    uint8_t mode = LOW )
```

Manage power.

Parameters

| | |
|-------------|--------------------------|
| <i>mode</i> | LOW=default=off, HIGH=on |
|-------------|--------------------------|

4.4.2.4 WhoAmI()

```
String Sensor_OPT3001::WhoAmI ( )
```

Who Am I?

Returns

name of the sensor, string

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

- [Sensor_OPT3001.h](#)
- Sensor_OPT3001.cpp

4.5 Sensor_TMP007 Class Reference

Class for sensor TMP007.

```
#include <Sensor_TMP007.h>
```

Public Member Functions

- [Sensor_TMP007](#) (uint8_t address=0x40)
Constructor.
- void [begin](#) (uint16_t totalSamples=0x0400)
Initialisation.
- String [WhoAmI](#) ()
Who Am I?
- void [get](#) ()
Acquisition.
- float [internal](#) ()
Measure.
- float [external](#) ()
Measure.
- void [setPowerMode](#) (uint8_t mode=LOW)
Manage power.

4.5.1 Detailed Description

Class for sensor TMP007.

Infrared Thermopile Contactless Temperature Sensor with Integrated Math Engine

See also

<http://www.ti.com/product/TMP007>

4.5.2 Constructor & Destructor Documentation

4.5.2.1 Sensor_TMP007()

```
Sensor_TMP007::Sensor_TMP007 (
    uint8_t address = 0x40 )
```

Constructor.

Parameters

| | |
|----------------|----------------|
| <i>address</i> | default = 0x40 |
|----------------|----------------|

4.5.3 Member Function Documentation

4.5.3.1 begin()

```
void Sensor_TMP007::begin (
    uint16_t totalSamples = 0x0400 )
```

Initialisation.

Parameters

| | |
|---------------------|--|
| <i>totalSamples</i> | default = 4 samples, use pre-defined constants |
|---------------------|--|

4.5.3.2 external()

```
float Sensor_TMP007::external ( )
```

Measure.

Returns

External temperature in °K

4.5.3.3 internal()

```
float Sensor_TMP007::internal ( )
```

Measure.

Returns

Internal temperature in °K

4.5.3.4 setPowerMode()

```
void Sensor_TMP007::setPowerMode (
    uint8_t mode = LOW )
```

Manage power.

Parameters

| | |
|-------------|--------------------------|
| <i>mode</i> | LOW=default=off, HIGH=on |
|-------------|--------------------------|

4.5.3.5 WhoAmI()

```
String Sensor_TMP007::WhoAmI ( )
```

Who Am I?

Returns

name of the sensor, string

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

- [Sensor_TMP007.h](#)
- [Sensor_TMP007.cpp](#)

4.6 unit_conversion_s Struct Reference

Units.

```
#include <Sensor_Units.h>
```

Public Attributes

- float [gain](#)
gain
- float [base](#)
base
- char [symbol](#) [4]
symbol

4.6.1 Detailed Description

Units.

A unit contains gain and base for conversion based on the SI reference unit.

Note

For each set of units, all units are defined the SI reference unit

The documentation for this struct was generated from the following file:

- [Sensor_Units.h](#)

Chapter 5

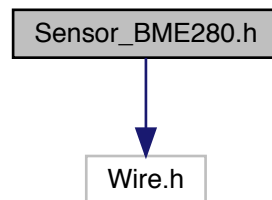
File Documentation

5.1 Sensor_BME280.h File Reference

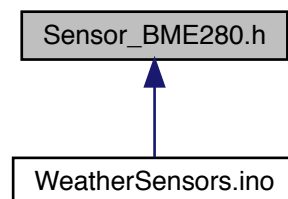
Library header for BME280 sensor.

```
#include "Wire.h"
```

Include dependency graph for Sensor_BME280.h:



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



Classes

- class [Sensor_BME280](#)
Class for sensor BME280.

Macros

- #define [Sensor_BME280_RELEASE](#) 103
Release.
- #define [BM280_SUCCESS](#) 0
success
- #define [BM280_ERROR](#) 1
error
- #define [BME280_SLAVE_ADDRESS](#) 0x77
Default BME280 I2C address.
- #define **BME280_FORCED_MODE** 0b01
- #define **BME280_SLEEP_MODE** 0b00
- #define **BME280_NORMAL_MODE** 0b11

5.1.1 Detailed Description

Library header for BME280 sensor.

BME280 Combined humidity and pressure sensor

Project SensorsBoosterPack
Developed with [embedXcode+](#)

Author

Rei Vilo
<http://embeddedcomputing.weebly.com>

Date

20 Aug 2017

Version

102

Copyright

(c) Rei Vilo, 2015-2018
CC = BY SA NC

See also

ReadMe.txt for references

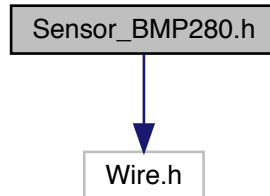
- Pressure Altimetry using the MPL3115A2
http://cache.freescale.com/files/sensors/doc/app_note/AN4528.pdf

5.2 Sensor_BMP280.h File Reference

Library header for BMP280 sensor.

```
#include "Wire.h"
```

Include dependency graph for Sensor_BMP280.h:



Classes

- class [Sensor_BMP280](#)
Class for sensor BMP280.

Macros

- #define [Sensor_BMP280_RELEASE](#) 102
Release.
- #define **BMP280_SLAVE_ADDRESS** 0x77
- #define [BM280_SUCCESS](#) 0
success
- #define [BM280_ERROR](#) 1
error
- #define **BMP280_FORCED_MODE** 0b01
- #define **BMP280_SLEEP_MODE** 0b00
- #define **BMP280_NORMAL_MODE** 0b11

5.2.1 Detailed Description

Library header for BMP280 sensor.

BMP280 Combined humidity and pressure sensor

Project SensorsBoosterPack
Developed with [embedXcode+](#)

Author

Rei Vilo

<http://embeddedcomputing.weebly.com>

Date

20 Aug 2015

Version

102

Copyright

(c) Rei Vilo, 2015-2018

CC = BY SA NC

See also

ReadMe.txt for references

- Pressure Altimetry using the MPL3115A2

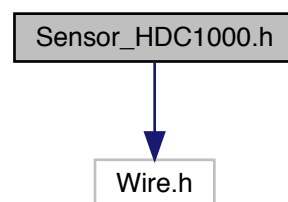
http://cache.freescale.com/files/sensors/doc/app_note/AN4528.pdf

5.3 Sensor_HDC1000.h File Reference

Library header.

```
#include "Wire.h"
```

Include dependency graph for Sensor_HDC1000.h:

**Classes**

- class [Sensor_HDC1000](#)

Class for sensor HDC1000.

Macros

- `#define Sensor_HDC1000_cpp`
- `#define HDC1000_I2C_ADDRESS 0x43`
- `#define HDC1000_RESET 0b10000000`
- `#define HDC1000_HEATER_DISABLED 0`
- `#define HDC1000_HEATER_ENABLED 0b00100000`
- `#define HDC1000_MODE_EITHER 0`
- `#define HDC1000_MODE_SEQUENCE 0b00010000`
- `#define HDC1000_TEMPERATURE_14_BITS 0`
- `#define HDC1000_TEMPERATURE_11_BITS 0b00000100`
- `#define HDC1000_HUMIDITY_14_BITS 0`
- `#define HDC1000_HUMIDITY_11_BITS 0b00000001`
- `#define HDC1000_HUMIDITY_8_BITS 0b00000010`
- `#define HDC1000_SETTINGS 0b00010101`

5.3.1 Detailed Description

Library header.

HDC1000 Temperature and Humidity Sensor

Project smartWatch

Developed with [embedXcode+](#)

Author

ReiVilo

ReiVilo

Date

12 Mar 2016

Version

101

Copyright

(c) Rei Vilo, 2016-2018

CC = BY SA NC

See also

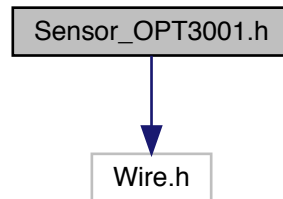
ReadMe.txt for references

5.4 Sensor_OPT3001.h File Reference

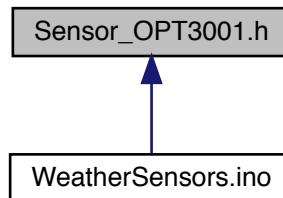
Library header for OPT3001 sensor.

```
#include "Wire.h"
```

Include dependency graph for Sensor_OPT3001.h:



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



Classes

- class [Sensor_OPT3001](#)
Class for sensor OPT3001.

Macros

- #define [Sensor_OPT3001_RELEASE](#) 102
Release.
- #define [OPT3001_SLAVE_ADDRESS](#) 0x47

- #define `OPT3001_100_MS_OFF` 0xc010
Conversion modes.
- #define `OPT3001_100_MS_ONCE` 0xc210
Conversion modes.
- #define `OPT3001_100_MS_CONTINUOUS` 0xc410
continuous
- #define `OPT3001_800_MS_ONCE` 0xc810
Conversion modes.
- #define `OPT3001_800_MS_OFF` 0xca10
Conversion modes.
- #define `OPT3001_800_MS_CONTINUOUS` 0xcc10
continuous
- #define `OPT3001_INTERRUPT_PIN` 11
Conversion modes.

5.4.1 Detailed Description

Library header for OPT3001 sensor.

OPT3001 Digital Ambient Light Sensor (ALS) with High Precision Human Eye Response

Project SensorsBoosterPack

Developed with `embedXcode+`

Author

a0273900 for initial C-library
Rei Vilo for Energia adapted C++-library
<http://embeddedcomputing.weebly.com>

Date

20 Aug 2015

Version

102

Copyright

(c) Rei Vilo, 2015-2018
CC = BY SA NC

See also

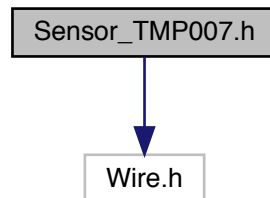
ReadMe.txt for references

5.5 Sensor_TMP007.h File Reference

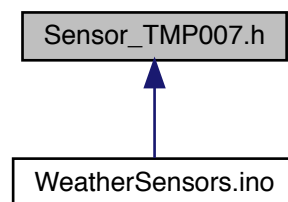
Library header for TMP007 sensor.

```
#include "Wire.h"
```

Include dependency graph for Sensor_TMP007.h:



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



Classes

- class [Sensor_TMP007](#)
Class for sensor TMP007.

Macros

- #define [Sensor_TMP007_cpp](#) 102
Release.
- #define [TMP007_SLAVE_ADDRESS](#) 0x40

- TMP007 constants.*
 - #define [TMP007_ONE_SAMPLE](#) 0x0000
- TMP007 constants.*
 - #define [TMP007_TWO_SAMPLES](#) 0x0200
- TMP007 constants.*
 - #define [TMP007_FOUR_SAMPLES](#) 0x0400
- TMP007 constants.*
 - #define [TMP007_EIGHT_SAMPLES](#) 0x0600
- TMP007 constants.*
 - #define [TMP007_SIXTEEN_SAMPLES](#) 0x0800
- TMP007 constants.*
 - #define [TMP007_ONE_SAMPLE_LOW_POWER](#) 0x0A00
- TMP007 constants.*
 - #define [TMP007_TWO_SAMPLES_LOW_POWER](#) 0x0C00
- TMP007 constants.*
 - #define [TMP007_FOUR_SAMPLES_LOW_POWER](#) 0x0E00
- TMP007 constants.*

5.5.1 Detailed Description

Library header for TMP007 sensor.

TMP007 Infrared Thermopile Contactless Temperature Sensor with Integrated Math Engine

Project SensorsBoosterPack

Developed with [embedXcode+](#)

Author

a0273900 Rei Vilo

<http://embeddedcomputing.weebly.com>

Date

20 Aug 2015

Version

102

Copyright

(c) Rei Vilo, 2015-2018

CC = BY SA NC

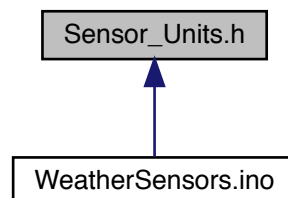
See also

ReadMe.txt for references

5.6 Sensor_Units.h File Reference

Library header.

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



Classes

- struct [unit_conversion_s](#)
Units.

Macros

- `#define` [Sensor_Units_RELEASE](#) 102
Release.

Functions

- `template<typename myType >`
`float` [conversion](#) (`float` value, `myType` unitFrom, `myType` unitTo)
Conversion utility.
- `template<typename myType >`
`String` [symbolString](#) (`myType` unit)
Unit symbol as String.
- `template<typename myType >`
`char *` [symbolChar](#) (`myType` unit)
Unit symbol as char.*
- `typedef` [unit_conversion_s](#) [temperature_unit_t](#)
Temperature units.
- `const` [temperature_unit_t](#) [KELVIN](#) = { 1, 0, "°K"}
°K degree kelvin, SI reference.
- `const` [temperature_unit_t](#) [CELSIUS](#) = { 1, -273.15, "°C"}
°C degree celsius.
- `const` [temperature_unit_t](#) [FAHRENHEIT](#) = { 1.8, -459.67, "°F"}

°F degree fahrenheit.

- typedef `unit_conversion_s pressure_unit_t`
Pressure units.
- const `pressure_unit_t PASCAL` = { 1, 0, "Pa"}
Pa pascal, SI reference.
- const `pressure_unit_t HECTOPASCAL` = { 1e-2, 0, "hPa"}
hPa hecto pascal, SI reference
- const `pressure_unit_t BAR` = { 1e-5, 0, "bar"}
bar
- const `pressure_unit_t ATMOSPHERE` = { 1.0 / 101325.0, 0, "atm"}
atmosphere
- const `pressure_unit_t PSI` = { 0.014503773801, 0, "atm"}
0.014503773801 pound force/square inch
- typedef `unit_conversion_s altitude_unit_t`
Altitude units.
- const `altitude_unit_t METRE` = { 1, 0, "m"}
m metre, SI reference
- const `altitude_unit_t FOOT` = { 0.3048, 0, "ft"}
ft foot
- typedef `unit_conversion_s light_unit_t`
Light units.
- const `light_unit_t LUX` = { 1, 0, "lx"}
lx, SI reference

5.6.1 Detailed Description

Library header.

Units conversion for sensors

Project SensorsBoosterPack

Developed with [embedXcode+](#)

Author

Rei Vilo

<http://embeddedcomputing.weebly.com>

Date

20 Aug 2017

Version

102

Copyright

(c) Rei Vilo, 2015-2018

CC = SA BY NC

See also

ReadMe.txt for references

5.6.2 Function Documentation

5.6.2.1 conversion()

```
template<typename myType >
float conversion (
    float value,
    myType unitFrom,
    myType unitTo )
```

Conversion utility.

Parameters

| | |
|-----------------|---|
| <i>value</i> | input value to be converted, float |
| <i>unitFrom</i> | unit of the input value to be converted |
| <i>unitTo</i> | unit for the output converted value |

Returns

output converted value, float

5.6.2.2 symbolChar()

```
template<typename myType >
char* symbolChar (
    myType unit )
```

Unit symbol as char*.

Parameters

| | |
|-------------|---------------|
| <i>unit</i> | unit constant |
|-------------|---------------|

Returns

symbol as char*

5.6.2.3 symbolString()

```
template<typename myType >
String symbolString (
    myType unit )
```

Unit symbol as String.

Parameters

| | |
|-------------|---------------|
| <i>unit</i> | unit constant |
|-------------|---------------|

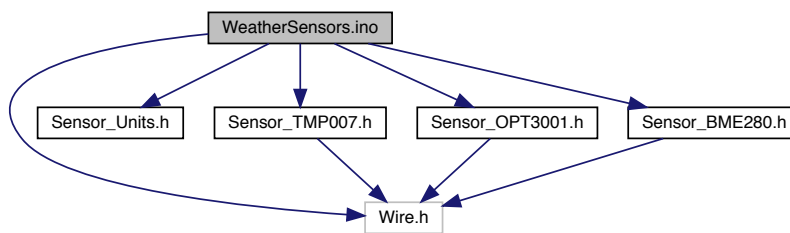
Returns

symbol as String

5.7 WeatherSensors.ino File Reference

Main sketch.

```
#include "Wire.h"
#include "Sensor_Units.h"
#include "Sensor_TMP007.h"
#include "Sensor_OPT3001.h"
#include "Sensor_BME280.h"
Include dependency graph for WeatherSensors.ino:
```



Macros

- `#define USE_TMP007 1`
- `#define USE_OPT3001 1`
- `#define USE_BME280 1`

Functions

- `void setup ()`
- `void loop ()`

Variables

- `Sensor_TMP007 myTMP007`
- `float TMP007_internal`
- `float TMP007_external`
- `Sensor_OPT3001 myOPT3001`
- `float OPT3001_light`
- `Sensor_BME280 myBME280`
- `float BME280_pressure`
- `float BME280_temperature`
- `float BME280_humidity`
- `const uint32_t period_ms = 10000`

5.7.1 Detailed Description

Main sketch.

Example for climate sensors

Developed with embedXcode+

Author

Rei Vilo

<http://embeddedcomputing.weebly.com>

Date

12 Nov 2016

Version

102

Copyright

(c) Rei Vilo, 2016-2018

CC = BY SA NC

See also

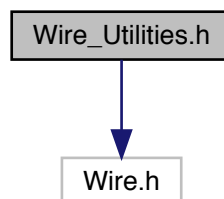
ReadMe.txt for references

5.8 Wire_Utilities.h File Reference

Library header.

```
#include "Wire.h"
```

Include dependency graph for Wire_Utilities.h:



Macros

- `#define Wire_Uilities_RELEASE 102`

Functions

- void `writeRegister8` (uint8_t device, uint8_t command, uint8_t data8)
Write 1 byte.
- void `writeRegister16` (uint8_t device, uint8_t command, uint16_t data16, uint8_t mode=MSBFIRST)
Write 2 bytes.
- uint8_t `readRegister8` (uint8_t device, uint8_t command)
Read 1 byte.
- uint16_t `readRegister16` (uint8_t device, uint8_t command, uint8_t mode=MSBFIRST)
Read 2 bytes.
- void `delayBusy` (uint32_t ms)
Delay without yield.

5.8.1 Detailed Description

Library header.

Utilities for 8- and 16-bit read and write operations

Project SensorsBoosterPack

Developed with [embedXcode+](#)

Author

Rei Vilo

<http://embeddedcomputing.weebly.com>

Date

20 Aug 2015

Version

102

Copyright

(c) Rei Vilo, 2015-2018
CC = BY SA NC

See also

ReadMe.txt for references

5.8.2 Function Documentation

5.8.2.1 `delayBusy()`

```
void delayBusy (
    uint32_t ms )
```

Delay without yield.

Parameters

| | |
|-----------|------------------------|
| <i>ms</i> | period to wait for, ms |
|-----------|------------------------|

5.8.2.2 readRegister16()

```
uint16_t readRegister16 (
    uint8_t device,
    uint8_t command,
    uint8_t mode = MSBFIRST )
```

Read 2 bytes.

Parameters

| | |
|----------------|---|
| <i>device</i> | I2C address, 7-bit coded |
| <i>command</i> | command or register, 8-bit |
| <i>mode</i> | default=MSBFIRST, other option=LSBFIRST |

Returns

data16 value, 16-bit

Note

- * with MSBFIRST, data16[15..8] read from command, data16[7..0] from command + 1
- * with LSBFIRST, data16[7..0] read from command, data16[15..8] from command + 1

5.8.2.3 readRegister8()

```
uint8_t readRegister8 (
    uint8_t device,
    uint8_t command )
```

Read 1 byte.

Parameters

| | |
|----------------|--------------------------|
| <i>device</i> | I2C address, 7-bit coded |
| <i>command</i> | command, 8-bit |

Returns

data8 value, 8-bit

5.8.2.4 writeRegister16()

```
void writeRegister16 (
    uint8_t device,
    uint8_t command,
    uint16_t data16,
    uint8_t mode = MSBFIRST )
```

Write 2 bytes.

Parameters

| | |
|----------------|---|
| <i>device</i> | I2C address, 7-bit coded |
| <i>command</i> | command or register, 8-bit |
| <i>data16</i> | value, 16-bit |
| <i>mode</i> | default=MSBFIRST, other option=LSBFIRST |

Note

- * with MSBFIRST, data16[15..8] written to command, data16[7..0] to command + 1
- * with LSBFIRST, data16[7..0] written to command, data16[15..8] to command + 1

5.8.2.5 writeRegister8()

```
void writeRegister8 (
    uint8_t device,
    uint8_t command,
    uint8_t data8 )
```

Write 1 byte.

Parameters

| | |
|----------------|----------------------------|
| <i>device</i> | I2C address, 7-bit coded |
| <i>command</i> | command or register, 8-bit |
| <i>data8</i> | value, 8-bit |

Index

- absolutePressure
 - Sensor_BME280, [8](#)
 - Sensor_BMP280, [13](#)
- altitude
 - Sensor_BME280, [9](#)
 - Sensor_BMP280, [14](#)
- begin
 - Sensor_BME280, [9](#)
 - Sensor_BMP280, [14](#)
 - Sensor_HDC1000, [17](#)
 - Sensor_OPT3001, [19](#)
 - Sensor_TMP007, [21](#)
- conversion
 - Sensor_Units.h, [36](#)
- delayBusy
 - Wire_Utilities.h, [40](#)
- external
 - Sensor_TMP007, [22](#)
- get
 - Sensor_BME280, [10](#)
 - Sensor_BMP280, [15](#)
- humidity
 - Sensor_BME280, [10](#)
 - Sensor_HDC1000, [18](#)
- internal
 - Sensor_TMP007, [22](#)
- light
 - Sensor_OPT3001, [20](#)
- pressure
 - Sensor_BME280, [10](#)
 - Sensor_BMP280, [15](#)
- readRegister16
 - Wire_Utilities.h, [41](#)
- readRegister8
 - Wire_Utilities.h, [41](#)
- Sensor_BME280, [7](#)
 - absolutePressure, [8](#)
 - altitude, [9](#)
 - begin, [9](#)
 - get, [10](#)
 - humidity, [10](#)
 - pressure, [10](#)
 - Sensor_BME280, [8](#)
 - setPowerMode, [11](#)
 - temperature, [11](#)
 - WhoAml, [11](#)
- Sensor_BME280.h, [25](#)
- Sensor_BMP280, [12](#)
 - absolutePressure, [13](#)
 - altitude, [14](#)
 - begin, [14](#)
 - get, [15](#)
 - pressure, [15](#)
 - Sensor_BMP280, [13](#)
 - setPowerMode, [15](#)
 - temperature, [16](#)
 - WhoAml, [16](#)
- Sensor_BMP280.h, [27](#)
- Sensor_HDC1000, [16](#)
 - begin, [17](#)
 - humidity, [18](#)
 - Sensor_HDC1000, [17](#)
 - setPowerMode, [18](#)
 - temperature, [18](#)
- Sensor_HDC1000.h, [28](#)
- Sensor_OPT3001, [19](#)
 - begin, [19](#)
 - light, [20](#)
 - setPowerMode, [20](#)
 - WhoAml, [20](#)
- Sensor_OPT3001.h, [30](#)
- Sensor_TMP007, [20](#)
 - begin, [21](#)
 - external, [22](#)
 - internal, [22](#)
 - Sensor_TMP007, [21](#)
 - setPowerMode, [22](#)
 - WhoAml, [23](#)
- Sensor_TMP007.h, [32](#)
- Sensor_Units.h, [34](#)
 - conversion, [36](#)
 - symbolChar, [36](#)
 - symbolString, [36](#)
- setPowerMode
 - Sensor_BME280, [11](#)
 - Sensor_BMP280, [15](#)
 - Sensor_HDC1000, [18](#)
 - Sensor_OPT3001, [20](#)
 - Sensor_TMP007, [22](#)
- symbolChar

- Sensor_Units.h, [36](#)
- symbolString
 - Sensor_Units.h, [36](#)
- temperature
 - Sensor_BME280, [11](#)
 - Sensor_BMP280, [16](#)
 - Sensor_HDC1000, [18](#)
- unit_conversion_s, [23](#)
- WeatherSensors.ino, [38](#)
- WhoAmI
 - Sensor_BME280, [11](#)
 - Sensor_BMP280, [16](#)
 - Sensor_OPT3001, [20](#)
 - Sensor_TMP007, [23](#)
- Wire_Utilities.h, [39](#)
 - delayBusy, [40](#)
 - readRegister16, [41](#)
 - readRegister8, [41](#)
 - writeRegister16, [42](#)
 - writeRegister8, [42](#)
- writeRegister16
 - Wire_Utilities.h, [42](#)
- writeRegister8
 - Wire_Utilities.h, [42](#)