BME280_Driver

1.3

Generated by Doxygen 1.8.16

1 Data Structure Index 1

1.1 Data Structures . . . 1

2 File Index 3

2.1 File List . . . 3

3 Data Structure Documentation 5

3.1 BME280_ConfigRegisterUnion Union Reference . . . 5

3.1.1 Detailed Description . . . 5

3.1.2 Field Documentation . . . 5

3.1.2.1 __pad0__ . . . 5

3.1.2.2 Bits . . . 6

3.1.2.3 config . . . 6

3.1.2.4 filter_coeff . . . 6

3.1.2.5 spi3w_en . . . 6

3.1.2.6 t_sb . . . 6

3.2 BME280_ConfigType Struct Reference . . . 7

3.2.1 Detailed Description . . . 7

3.2.2 Field Documentation . . . 7

3.2.2.1 calib_data_1 . . . 7

3.2.2.2 calib_data_2 . . . 8

3.2.2.3 GPIOCallback_resetNSS . . . 8

3.2.2.4 GPIOCallback_SetNSS . . . 8

3.2.2.5 I2C_SlaveAddr . . . 8

3.2.2.6 ID . . . 8

3.2.2.7 Intf . . . 9

3.2.2.8 occupied . . . 9

3.3 BME280_CtrlHumRegisterUnion Union Reference . . . 9

3.3.1 Detailed Description . . . 9

3.3.2 Field Documentation . . . 9

3.3.2.1 __pad0__ . . . 10

3.3.2.2 Bits . . . 10

3.3.2.3 config . . . 10

3.3.2.4 osrs_h . . . 10

3.4 BME280_CtrlMeasRegisterUnion Union Reference . . . 10

3.4.1 Detailed Description . . . 11

3.4.2 Field Documentation . . . 11

3.4.2.1 Bits . . . 11

3.4.2.2 config . . . 11

3.4.2.3 mode . . . 11

3.4.2.4 osrs_p . . . 11

3.4.2.5 osrs_t . . . 11

3.5 BME280_HumidityReading Union Reference . . . 12

3.5.1 Detailed Description . . . 12
3.5.2 Field Documentation . . . 12
3.5.2.1 Data . . . 12
3.5.2.2 humidity . . . 12
3.5.2.3 lsb . . . 12
3.5.2.4 msb . . . 13
3.6 BME280_PressureReading Union Reference . . . 13
3.6.1 Detailed Description . . . 13
3.6.2 Field Documentation . . . 13
3.6.2.1 __pad0__ . . . 13
3.6.2.2 Data . . . 14
3.6.2.3 lsb . . . 14
3.6.2.4 msb . . . 14
3.6.2.5 pressure . . . 14
3.6.2.6 xlsb . . . 14
3.7 BME280_Settings Struct Reference . . . 14
3.7.1 Detailed Description . . . 15
3.7.2 Field Documentation . . . 15
3.7.2.1 filter . . . 15
3.7.2.2 Mode . . . 15
3.7.2.3 osrs_h . . . 16
3.7.2.4 osrs_p . . . 16
3.7.2.5 osrs_t . . . 16
3.7.2.6 t_stby . . . 16
3.8 BME280_StatusRegisterUnion Union Reference . . . 16
3.8.1 Detailed Description . . . 17
3.8.2 Field Documentation . . . 17
3.8.2.1 __pad0__ . . . 17
3.8.2.2 __pad1__ . . . 17
3.8.2.3 Bits . . . 17
3.8.2.4 config . . . 18
3.8.2.5 im_update . . . 18
3.8.2.6 measuring . . . 18
3.9 BME280_TemperatureReading Union Reference . . . 18
3.9.1 Detailed Description . . . 18
3.9.2 Field Documentation . . . 19
3.9.2.1 __pad0__ . . . 19
3.9.2.2 Data . . . 19
3.9.2.3 lsb . . . 19
3.9.2.4 msb . . . 19
3.9.2.5 temperature . . . 19
3.9.2.6 xlsb . . . 20

3.10 BME280_UncompensatedReadings Union Reference . . . 20
3.10.1 Detailed Description . . . 20
3.11 Calib1 Union Reference . . . 20
3.11.1 Detailed Description . . . 20
3.12 Calib2 Struct Reference . . . 20
3.12.1 Detailed Description . . . 20

4 File Documentation **21**

4.1 Core/BME280_Driver/bme280.c File Reference . . . 21
4.1.1 Function Documentation . . . 22
4.1.1.1 __attribute__() . . . 23
4.1.1.2 BME280_calculateMeasurementDelayMs() . . . 23
4.1.1.3 BME280_DeInit() . . . 23
4.1.1.4 BME280_getChipID() . . . 24
4.1.1.5 BME280_getFilterCoefficient() . . . 24
4.1.1.6 BME280_getHumidity_fixedPoint() . . . 24
4.1.1.7 BME280_getHumidity_floatingPoint() . . . 25
4.1.1.8 BME280_getHumidityOversampling() . . . 25
4.1.1.9 BME280_getInstance() . . . 26
4.1.1.10 BME280_getMeasuringStatus() . . . 26
4.1.1.11 BME280_getMode() . . . 27
4.1.1.12 BME280_getPressure_fixedPoint() . . . 27
4.1.1.13 BME280_getPressure_floatingPoint() . . . 27
4.1.1.14 BME280_getPressureOversampling() . . . 28
4.1.1.15 BME280_getSensorSettings() . . . 28
4.1.1.16 BME280_getStandbyTime() . . . 29
4.1.1.17 BME280_getTemperature_fixedPoint() . . . 29
4.1.1.18 BME280_getTemperature_floatingPoint() . . . 29
4.1.1.19 BME280_getTemperatureOversampling() . . . 30
4.1.1.20 BME280_getUpdateStatus() . . . 30
4.1.1.21 BME280_init() . . . 31
4.1.1.22 BME280_setAssertNSSCallback() . . . 31
4.1.1.23 BME280_setFilterCoefficient() . . . 31
4.1.1.24 BME280_setHumidityOversampling() . . . 32
4.1.1.25 BME280_setInterfaceType() . . . 32
4.1.1.26 BME280_setMode() . . . 33
4.1.1.27 BME280_setPressureOversampling() . . . 33
4.1.1.28 BME280_setReleaseNSSCallback() . . . 34
4.1.1.29 BME280_setStandbyTime() . . . 34
4.1.1.30 BME280_setTemperatureOversampling() . . . 34
4.1.1.31 BME280_softReset() . . . 35
4.2 Core/BME280_Driver/bme280.h File Reference . . . 36

4.2.1 Typedef Documentation . . . 38
4.2.1.1 BME280_boolean . . . 38
4.2.1.2 BME280_float64 . . . 39
4.2.1.3 BME280_Handle . . . 39
4.2.1.4 BME280_sint16 . . . 39
4.2.1.5 BME280_sint32 . . . 39
4.2.1.6 BME280_sint8 . . . 39
4.2.1.7 BME280_uint16 . . . 39
4.2.1.8 BME280_uint32 . . . 40
4.2.1.9 BME280_uint8 . . . 40
4.2.2 Enumeration Type Documentation . . . 40
4.2.2.1 BME280_Comm_Status . . . 40
4.2.2.2 BME280_FilterCoeff . . . 40
4.2.2.3 BME280_InterfaceType . . . 41
4.2.2.4 BME280_MeasuringStatus . . . 41
4.2.2.5 BME280_ModeType . . . 41
4.2.2.6 BME280_Oversampling_setting . . . 42
4.2.2.7 BME280_StandbyTime . . . 42
4.2.2.8 BME280_Status . . . 43
4.2.2.9 BME280_UpdateStatus . . . 43
4.2.3 Function Documentation . . . 44
4.2.3.1 BME280_calculateMeasurementDelayMs() . . . 44
4.2.3.2 BME280_DeInit() . . . 44
4.2.3.3 BME280_delayMs() . . . 45
4.2.3.4 BME280_getChipID() . . . 45
4.2.3.5 BME280_getFilterCoefficient() . . . 45
4.2.3.6 BME280_getHumidity_fixedPoint() . . . 46
4.2.3.7 BME280_getHumidity_floatingPoint() . . . 46
4.2.3.8 BME280_getHumidityOversampling() . . . 46
4.2.3.9 BME280_getInstance() . . . 47
4.2.3.10 BME280_getMeasuringStatus() . . . 47
4.2.3.11 BME280_getMode() . . . 48
4.2.3.12 BME280_getPressure_fixedPoint() . . . 48
4.2.3.13 BME280_getPressure_floatingPoint() . . . 49
4.2.3.14 BME280_getPressureOversampling() . . . 49
4.2.3.15 BME280_getSensorSettings() . . . 49
4.2.3.16 BME280_getStandbyTime() . . . 50
4.2.3.17 BME280_getTemperature_fixedPoint() . . . 50
4.2.3.18 BME280_getTemperature_floatingPoint() . . . 51
4.2.3.19 BME280_getTemperatureOversampling() . . . 51
4.2.3.20 BME280_getUpdateStatus() . . . 51
4.2.3.21 BME280_I2C_Master_Receive() . . . 52

4.2.3.22 BME280_I2C_Master_Transmit() . . . 52
4.2.3.23 BME280_init() . . . 53
4.2.3.24 BME280_setAssertNSSCallback() . . . 53
4.2.3.25 BME280_setFilterCoefficient() . . . 53
4.2.3.26 BME280_setHumidityOversampling() . . . 54
4.2.3.27 BME280_setInterfaceType() . . . 54
4.2.3.28 BME280_setMode() . . . 55
4.2.3.29 BME280_setPressureOversampling() . . . 55
4.2.3.30 BME280_setReleaseNSSCallback() . . . 56
4.2.3.31 BME280_setStandbyTime() . . . 56
4.2.3.32 BME280_setTemperatureOversampling() . . . 56
4.2.3.33 BME280_softReset() . . . 57
4.2.3.34 BME280_SPI_TransmitReceive() . . . 57
4.3 Core/BME280_Driver/bme280_private_defs.h File Reference . . . 58
4.3.1 Macro Definition Documentation . . . 59
4.3.1.1 BME280_CALIB_1_BLOCK_SIZE . . . 59
4.3.1.2 BME280_CALIB_2_BLOCK_SIZE . . . 59
4.3.1.3 BME280_CHIP_ID . . . 60
4.3.1.4 BME280_CONFIG_IM_UPDATE_MASK . . . 60
4.3.1.5 BME280_CONFIG_MEAS_MASK . . . 60
4.3.1.6 BME280_CONFIG_REGISTER . . . 60
4.3.1.7 BME280_CTRL_HUM_REGISTER . . . 60
4.3.1.8 BME280_CTRL_MEAS_REGISTER . . . 60
4.3.1.9 BME280_HUM_BLOCK_START_ADDRESS . . . 61
4.3.1.10 BME280_HUM_CALIB_BLOCK_SIZE . . . 61
4.3.1.11 BME280_HUM_REGISTER_LSB . . . 61
4.3.1.12 BME280_HUM_REGISTER_MSB . . . 61
4.3.1.13 BME280_I2C_READ_MASK . . . 61
4.3.1.14 BME280_I2C_TIMEOUT_MS . . . 61
4.3.1.15 BME280_I2C_WRITE_MASK . . . 62
4.3.1.16 BME280_ID_REGISTER . . . 62
4.3.1.17 BME280_IM_UPDATE_READY . . . 62
4.3.1.18 BME280_LSBYTE_MASK . . . 62
4.3.1.19 BME280_MAX_DISCOVERY_COUNT . . . 62
4.3.1.20 BME280_MAX_HUMIDITY_FIXED_POINT . . . 62
4.3.1.21 BME280_MAX_HUMIDITY_FLOATING_POINT . . . 63
4.3.1.22 BME280_MAX_PRESSURE_FIXED_POINT . . . 63
4.3.1.23 BME280_MAX_PRESSURE_FLOATING_POINT . . . 63
4.3.1.24 BME280_MAX_SENSOR_POOL_SIZE . . . 63
4.3.1.25 BME280_MAX_TEMPERATURE_FIXED_POINT . . . 63
4.3.1.26 BME280_MAX_TEMPERATURE_FLOATING_POINT . . . 63
4.3.1.27 BME280_MEASURING_DONE . . . 64

4.3.1.28 BME280_MIN_HUMIDITY_FIXED_POINT . . . 64
4.3.1.29 BME280_MIN_HUMIDITY_FLOATING_POINT . . . 64
4.3.1.30 BME280_MIN_PRESSURE_FIXED_POINT . . . 64
4.3.1.31 BME280_MIN_PRESSURE_FLOATING_POINT . . . 64
4.3.1.32 BME280_MIN_TEMPERATURE_FIXED_POINT . . . 64
4.3.1.33 BME280_MIN_TEMPERATURE_FLOATING_POINT . . . 65
4.3.1.34 BME280_PRESS_REGISTER_LSB . . . 65
4.3.1.35 BME280_PRESS_REGISTER_MSB . . . 65
4.3.1.36 BME280_PRESS_REGISTER_XLSB . . . 65
4.3.1.37 BME280_READINGS_BYTES_LENGTH . . . 65
4.3.1.38 BME280_RESET_REGISTER . . . 65
4.3.1.39 BME280_RESET_WORD . . . 66
4.3.1.40 BME280_SPI_READ_MASK . . . 66
4.3.1.41 BME280_SPI_TIMEOUT_MS . . . 66
4.3.1.42 BME280_SPI_WRITE_MASK . . . 66
4.3.1.43 BME280_START_READINGS_ADDRESS . . . 66
4.3.1.44 BME280_START_UP_TIME_MS . . . 66
4.3.1.45 BME280_STATUS_REGISTER . . . 67
4.3.1.46 BME280_TEMP_PRESS_BLOCK_START_ADDRESS . . . 67
4.3.1.47 BME280_TEMP_PRESS_CALIB_BLOCK_SIZE . . . 67
4.3.1.48 BME280_TEMP_REGISTER_LSB . . . 67
4.3.1.49 BME280_TEMP_REGISTER_MSB . . . 67
4.3.1.50 BME280_TEMP_REGISTER_XLSB . . . 67
4.4 Core/BME280_Driver/bme280_private_types.h File Reference . . . 68
4.4.1 Enumeration Type Documentation . . . 69
4.4.1.1 BME280_I2C_SensorSlaveAddress . . . 69
4.4.2 Function Documentation . . . 69
4.4.2.1 __attribute__() . . . 70
4.4.3 Variable Documentation . . . 70
4.4.3.1 BME280_UncompensatedReadings . . . 70
4.4.3.2 Calib1 . . . 70
4.4.3.3 Calib2 . . . 70
4.5 Core/BME280_Driver/bme280_unitTests.c File Reference . . . 70
4.5.1 Function Documentation . . . 71
4.5.1.1 BME280_UnitTest_FilterCoefficient() . . . 71
4.5.1.2 BME280_UnitTest_GeneralFuncs_I2C() . . . 72
4.5.1.3 BME280_UnitTest_GeneralFuncs_SPI() . . . 72
4.5.1.4 BME280_UnitTest_HumdityOversampling() . . . 73
4.5.1.5 BME280_UnitTest_PressureOversampling() . . . 74
4.5.1.6 BME280_UnitTest_ReadingsCheck_I2C() . . . 74
4.5.1.7 BME280_UnitTest_ReadingsCheck_SPI() . . . 75
4.5.1.8 BME280_UnitTest_StandbyTime() . . . 75

4.5.1.9 BME280_UnitTest_TemperatureOversampling() . . . 75
4.6 Core/BME280_Driver/bme280_unitTests.h File Reference . . . 76
4.6.1 Function Documentation . . . 77
4.6.1.1 BME280_GPIO_resetSlaveSelectPin() . . . 77
4.6.1.2 BME280_GPIO_setSlaveSelectPin() . . . 77
4.6.1.3 BME280_UnitTest_FilterCoefficient() . . . 77
4.6.1.4 BME280_UnitTest_GeneralFuncs_I2C() . . . 78
4.6.1.5 BME280_UnitTest_GeneralFuncs_SPI() . . . 78
4.6.1.6 BME280_UnitTest_HumdityOversampling() . . . 79
4.6.1.7 BME280_UnitTest_PressureOversampling() . . . 80
4.6.1.8 BME280_UnitTest_ReadingsCheck_I2C() . . . 80
4.6.1.9 BME280_UnitTest_ReadingsCheck_SPI() . . . 81
4.6.1.10 BME280_UnitTest_StandbyTime() . . . 81
4.6.1.11 BME280_UnitTest_TemperatureOversampling() . . . 81
4.7 Core/BME280_Driver/std_types.h File Reference . . . 82
4.7.1 Macro Definition Documentation . . . 83
4.7.1.1 FALSE . . . 83
4.7.1.2 False . . . 83
4.7.1.3 false . . . 83
4.7.1.4 LOGIC_HIGH . . . 83
4.7.1.5 LOGIC_LOW . . . 83
4.7.1.6 NULL_PTR . . . 84
4.7.1.7 TRUE . . . 84
4.7.1.8 True . . . 84
4.7.1.9 true . . . 84
4.7.2 Typedef Documentation . . . 84
4.7.2.1 boolean . . . 84
4.7.2.2 float32 . . . 84
4.7.2.3 float64 . . . 85
4.7.2.4 sint16 . . . 85
4.7.2.5 sint32 . . . 85
4.7.2.6 sint64 . . . 85
4.7.2.7 sint8 . . . 85
4.7.2.8 uint16 . . . 85
4.7.2.9 uint32 . . . 86
4.7.2.10 uint64 . . . 86
4.7.2.11 uint8 . . . 86

Index **87**

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

BME280_ConfigRegisterUnion
: Union used to create configuration register byte 5

BME280_ConfigType
: Configuration structure used for setup, configuration, and sensor functions 7

BME280_CtrlHumRegisterUnion
: Union used to create control humidity register byte 9

BME280_CtrlMeasRegisterUnion
: Union used to create control measurements register byte 10

BME280_HumidityReading
: Used to store readings from the sensor and access them directly from 'humidity' member . . . 12

BME280_PressureReading
: Used to store readings from the sensor and access them directly from 'pressure' member . . . 13

BME280_Settings
: Struct that contains sensor user settings 14

BME280_StatusRegisterUnion
: Union used to create status register byte 16

BME280_TemperatureReading
: Used to store readings from the sensor and access them directly from 'temperature' member . 18

BME280_UncompensatedReadings
: Union used to receive and access temp, press, & hum. data 20

Calib1
: Union used to receive and access calibration parameters 20

Calib2
: Struct used to receive and access calibration parameters 20

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Core/BME280_Driver/bme280.c 21
Core/BME280_Driver/bme280.h 36
Core/BME280_Driver/bme280_private_defs.h 58
Core/BME280_Driver/bme280_private_types.h 68
Core/BME280_Driver/bme280_unitTests.c 70
Core/BME280_Driver/bme280_unitTests.h 76
Core/BME280_Driver/std_types.h 82

Chapter 3

Data Structure Documentation

3.1 BME280_ConfigRegisterUnion Union Reference

Union used to create configuration register byte.

`#include <bme280_private_types.h>`

Data Fields

- BME280_uint8 config
- struct {
  BME280_uint8 spi3w_en:1
  BME280_uint8 __pad0__:1
  BME280_uint8 filter_coeff:3
  BME280_uint8 t_sb:3
  } Bits

3.1.1 Detailed Description

Union used to create configuration register byte.

Definition at line 29 of file bme280_private_types.h.

3.1.2 Field Documentation

3.1.2.1 __pad0__

`BME280_uint8 BME280_ConfigRegisterUnion::__pad0__`

Padding

Definition at line 33 of file bme280_private_types.h.

3.1.2.2 Bits

```
struct { ... } BME280_ConfigRegisterUnion::Bits
```

3.1.2.3 config

```
BME280_uint8 BME280_ConfigRegisterUnion::config
```

Definition at line 30 of file bme280_private_types.h.

3.1.2.4 filter_coeff

```
BME280_uint8 BME280_ConfigRegisterUnion::filter_coeff
```

Definition at line 34 of file bme280_private_types.h.

3.1.2.5 spi3w_en

```
BME280_uint8 BME280_ConfigRegisterUnion::spi3w_en
```

Definition at line 32 of file bme280_private_types.h.

3.1.2.6 t_sb

```
BME280_uint8 BME280_ConfigRegisterUnion::t_sb
```

Definition at line 35 of file bme280_private_types.h.

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

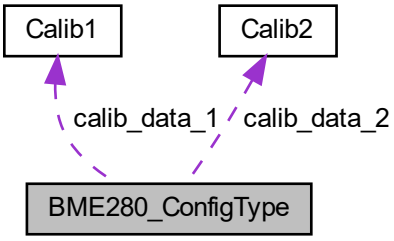
- Core/BME280_Driver/bme280_private_types.h

3.2 BME280_ConfigType Struct Reference

Configuration structure used for setup, configuration, and sensor functions.

`#include <bme280_private_types.h>`

Collaboration diagram for BME280_ConfigType:



Data Fields

- BME280_uint8 ID
- BME280_boolean occupied
- BME280_InterfaceType Intf
- BME280_I2C_SensorSlaveAddress I2C_SlaveAddr
- void(∗ GPIOCallback_SetNSS)(void)
- void(∗ GPIOCallback_resetNSS)(void)
- Calib1 calib_data_1
- Calib2 calib_data_2

3.2.1 Detailed Description

Configuration structure used for setup, configuration, and sensor functions.

Definition at line 219 of file bme280_private_types.h.

3.2.2 Field Documentation

3.2.2.1 calib_data_1

`Calib1 BME280_ConfigType::calib_data_1`

Contains calibration data obtained from sensor

Definition at line 226 of file bme280_private_types.h.

3.2.2.2 calib_data_2

```
Calib2 BME280_ConfigType::calib_data_2
```

Contains calibration data obtained from sensor

Definition at line 227 of file bme280_private_types.h.

3.2.2.3 GPIOCallback_resetNSS

```
void(* BME280_ConfigType::GPIOCallback_resetNSS) (void)
```

Callback function for resetting NSS pin

Definition at line 225 of file bme280_private_types.h.

3.2.2.4 GPIOCallback_SetNSS

```
void(* BME280_ConfigType::GPIOCallback_SetNSS) (void)
```

Callback function for setting NSS pin

Definition at line 224 of file bme280_private_types.h.

3.2.2.5 I2C_SlaveAddr

```
BME280_I2C_SensorSlaveAddress BME280_ConfigType::I2C_SlaveAddr
```

Definition at line 223 of file bme280_private_types.h.

3.2.2.6 ID

```
BME280_uint8 BME280_ConfigType::ID
```

Custom ID issued to the sensor by the user

Definition at line 220 of file bme280_private_types.h.

3.2.2.7 Intf

```
BME280_InterfaceType BME280_ConfigType::Intf
```

Interface type used with the sensor (I2C/SPI)

Definition at line 222 of file bme280_private_types.h.

3.2.2.8 occupied

```
BME280_boolean BME280_ConfigType::occupied
```

Flag used to determine if instance is occupied or empty

Definition at line 221 of file bme280_private_types.h.

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

- Core/BME280_Driver/bme280_private_types.h

3.3 BME280_CtrlHumRegisterUnion Union Reference

Union used to create control humidity register byte.

```
#include <bme280_private_types.h>
```

Data Fields

- BME280_uint8 config
- struct {
  BME280_uint8 osrs_h:3
  BME280_uint8 __pad0__:5
  } Bits

3.3.1 Detailed Description

Union used to create control humidity register byte.

Definition at line 183 of file bme280_private_types.h.

3.3.2 Field Documentation

3.3.2.1 __pad0__

```
BME280_uint8 BME280_CtrlHumRegisterUnion::__pad0__
```

Padding

Definition at line 187 of file bme280_private_types.h.

3.3.2.2 Bits

```
struct { ... } BME280_CtrlHumRegisterUnion::Bits
```

3.3.2.3 config

```
BME280_uint8 BME280_CtrlHumRegisterUnion::config
```

Definition at line 184 of file bme280_private_types.h.

3.3.2.4 osrs_h

```
BME280_uint8 BME280_CtrlHumRegisterUnion::osrs_h
```

Humidity over-sampling setting

Definition at line 186 of file bme280_private_types.h.

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

- Core/BME280_Driver/bme280_private_types.h

3.4 BME280_CtrlMeasRegisterUnion Union Reference

Union used to create control measurements register byte.

```
#include <bme280_private_types.h>
```

Data Fields

- BME280_uint8 config
- struct {
  BME280_uint8 mode:2
  BME280_uint8 osrs_p:3
  BME280_uint8 osrs_t:3
  } Bits

3.4.1 Detailed Description

Union used to create control measurements register byte.

Definition at line 170 of file bme280_private_types.h.

3.4.2 Field Documentation

3.4.2.1 Bits

```
struct { ... } BME280_CtrlMeasRegisterUnion::Bits
```

3.4.2.2 config

```
BME280_uint8 BME280_CtrlMeasRegisterUnion::config
```

Definition at line 171 of file bme280_private_types.h.

3.4.2.3 mode

```
BME280_uint8 BME280_CtrlMeasRegisterUnion::mode
```

Definition at line 173 of file bme280_private_types.h.

3.4.2.4 osrs_p

```
BME280_uint8 BME280_CtrlMeasRegisterUnion::osrs_p
```

Definition at line 174 of file bme280_private_types.h.

3.4.2.5 osrs_t

```
BME280_uint8 BME280_CtrlMeasRegisterUnion::osrs_t
```

Definition at line 175 of file bme280_private_types.h.

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

- Core/BME280_Driver/bme280_private_types.h

3.5 BME280_HumidityReading Union Reference

Used to store readings from the sensor and access them directly from 'humidity' member.

`#include <bme280_private_types.h>`

Data Fields

- BME280_sint16 humidity
- struct {
  BME280_sint16 lsb:8
  BME280_sint16 msb:8
  } Data

3.5.1 Detailed Description

Used to store readings from the sensor and access them directly from 'humidity' member.

Definition at line 158 of file bme280_private_types.h.

3.5.2 Field Documentation

3.5.2.1 Data

`struct { ... } BME280_HumidityReading::Data`

3.5.2.2 humidity

`BME280_sint16 BME280_HumidityReading::humidity`

Definition at line 159 of file bme280_private_types.h.

3.5.2.3 lsb

`BME280_sint16 BME280_HumidityReading::lsb`

Definition at line 161 of file bme280_private_types.h.

3.5.2.4 msb

```
BME280_sint16 BME280_HumidityReading::msb
```

Definition at line 162 of file bme280_private_types.h.

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

- Core/BME280_Driver/bme280_private_types.h

3.6 BME280_PressureReading Union Reference

Used to store readings from the sensor and access them directly from 'pressure' member.

```
#include <bme280_private_types.h>
```

Data Fields

- BME280_sint32 pressure
- struct {
  BME280_sint32 xlsb:4
  BME280_sint32 lsb:8
  BME280_sint32 msb:8
  BME280_sint32 __pad0__:12
  } Data

3.6.1 Detailed Description

Used to store readings from the sensor and access them directly from 'pressure' member.

Definition at line 141 of file bme280_private_types.h.

3.6.2 Field Documentation

3.6.2.1 __pad0__

```
BME280_sint32 BME280_PressureReading::__pad0__
```

Padding

Definition at line 147 of file bme280_private_types.h.

3.6.2.2 Data

```
struct { ... } BME280_PressureReading::Data
```

3.6.2.3 lsb

```
BME280_sint32 BME280_PressureReading::lsb
```

Definition at line 145 of file bme280_private_types.h.

3.6.2.4 msb

```
BME280_sint32 BME280_PressureReading::msb
```

Definition at line 146 of file bme280_private_types.h.

3.6.2.5 pressure

```
BME280_sint32 BME280_PressureReading::pressure
```

Definition at line 142 of file bme280_private_types.h.

3.6.2.6 xlsb

```
BME280_sint32 BME280_PressureReading::xlsb
```

Definition at line 144 of file bme280_private_types.h.

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

- Core/BME280_Driver/bme280_private_types.h

3.7 BME280_Settings Struct Reference

Struct that contains sensor user settings.

```
#include <bme280.h>
```

Data Fields

- BME280_ModeType Mode
- BME280_StandbyTime t_stby
- BME280_FilterCoeff filter
- BME280_Oversampling_setting osrs_t
- BME280_Oversampling_setting osrs_p
- BME280_Oversampling_setting osrs_h

3.7.1 Detailed Description

Struct that contains sensor user settings.

1. BME280_StandbyTime t_stby: Standby time between measurements
2. BME280_FilterCoeff filter: IIR filter coefficient
3. BME280_Oversampling_setting osrs_t: Over-sampling multiplier for temperature
4. BME280_Oversampling_setting osrs_p: Over-sampling multiplier for temperature
5. BME280_Oversampling_setting osrs_h: Over-sampling multiplier for humidity
6. BME280_ModeType Mode: Sensor mode (sleep, forced, normal)

Definition at line 205 of file bme280.h.

3.7.2 Field Documentation

3.7.2.1 filter

```
BME280_FilterCoeff BME280_Settings::filter
```

Filter coefficient

Definition at line 208 of file bme280.h.

3.7.2.2 Mode

```
BME280_ModeType BME280_Settings::Mode
```

Specifies the current mode for BME280

Definition at line 206 of file bme280.h.

3.7.2.3 osrs_h

```
BME280_Oversampling_setting BME280_Settings::osrs_h
```

Humidityover-sampling setting

Definition at line 211 of file bme280.h.

3.7.2.4 osrs_p

```
BME280_Oversampling_setting BME280_Settings::osrs_p
```

Pressure over-sampling setting

Definition at line 210 of file bme280.h.

3.7.2.5 osrs_t

```
BME280_Oversampling_setting BME280_Settings::osrs_t
```

Temperature over-sampling setting

Definition at line 209 of file bme280.h.

3.7.2.6 t_stby

```
BME280_StandbyTime BME280_Settings::t_stby
```

Standby time of sensor

Definition at line 207 of file bme280.h.

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

- Core/BME280_Driver/bme280.h

3.8 BME280_StatusRegisterUnion Union Reference

Union used to create status register byte.

```
#include <bme280_private_types.h>
```

Data Fields

- BME280_uint8 config
- struct {
  BME280_uint8 im_update:1
  BME280_uint8 __pad0__:2
  BME280_uint8 measuring:1
  BME280_uint8 __pad1__:4
  } Bits

3.8.1 Detailed Description

Union used to create status register byte.

Definition at line 195 of file bme280_private_types.h.

3.8.2 Field Documentation

3.8.2.1 __pad0__

```
BME280_uint8 BME280_StatusRegisterUnion::__pad0__
```

Padding

Definition at line 199 of file bme280_private_types.h.

3.8.2.2 __pad1__

```
BME280_uint8 BME280_StatusRegisterUnion::__pad1__
```

Padding

Definition at line 201 of file bme280_private_types.h.

3.8.2.3 Bits

```
struct { ... } BME280_StatusRegisterUnion::Bits
```

3.8.2.4 config

```
BME280_uint8 BME280_StatusRegisterUnion::config
```

Definition at line 196 of file bme280_private_types.h.

3.8.2.5 im_update

```
BME280_uint8 BME280_StatusRegisterUnion::im_update
```

Definition at line 198 of file bme280_private_types.h.

3.8.2.6 measuring

```
BME280_uint8 BME280_StatusRegisterUnion::measuring
```

Definition at line 200 of file bme280_private_types.h.

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

- Core/BME280_Driver/bme280_private_types.h

3.9 BME280_TemperatureReading Union Reference

Used to store readings from the sensor and access them directly from 'temperature' member.

```
#include <bme280_private_types.h>
```

Data Fields

- BME280_uint32 temperature
- struct {
  BME280_uint32 xlsb:4
  BME280_uint32 lsb:8
  BME280_uint32 msb:8
  BME280_uint32 __pad0__:12
  } Data

3.9.1 Detailed Description

Used to store readings from the sensor and access them directly from 'temperature' member.

Definition at line 125 of file bme280_private_types.h.

3.9.2 Field Documentation

3.9.2.1 __pad0__

```
BME280_uint32 BME280_TemperatureReading::__pad0__
```

Padding

Definition at line 131 of file bme280_private_types.h.

3.9.2.2 Data

```
struct { ... } BME280_TemperatureReading::Data
```

3.9.2.3 lsb

```
BME280_uint32 BME280_TemperatureReading::lsb
```

Definition at line 129 of file bme280_private_types.h.

3.9.2.4 msb

```
BME280_uint32 BME280_TemperatureReading::msb
```

Definition at line 130 of file bme280_private_types.h.

3.9.2.5 temperature

```
BME280_uint32 BME280_TemperatureReading::temperature
```

Definition at line 126 of file bme280_private_types.h.

3.9.2.6 xlsb

```
BME280_uint32 BME280_TemperatureReading::xlsb
```

Definition at line 128 of file bme280_private_types.h.

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

- Core/BME280_Driver/bme280_private_types.h

3.10 BME280_UncompensatedReadings Union Reference

Union used to receive and access temp, press, & hum. data.

```
#include <bme280_private_types.h>
```

3.10.1 Detailed Description

Union used to receive and access temp, press, & hum. data.

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

- Core/BME280_Driver/bme280_private_types.h

3.11 Calib1 Union Reference

Union used to receive and access calibration parameters.

```
#include <bme280_private_types.h>
```

3.11.1 Detailed Description

Union used to receive and access calibration parameters.

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

- Core/BME280_Driver/bme280_private_types.h

3.12 Calib2 Struct Reference

Struct used to receive and access calibration parameters.

```
#include <bme280_private_types.h>
```

3.12.1 Detailed Description

Struct used to receive and access calibration parameters.

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

- Core/BME280_Driver/bme280_private_types.h

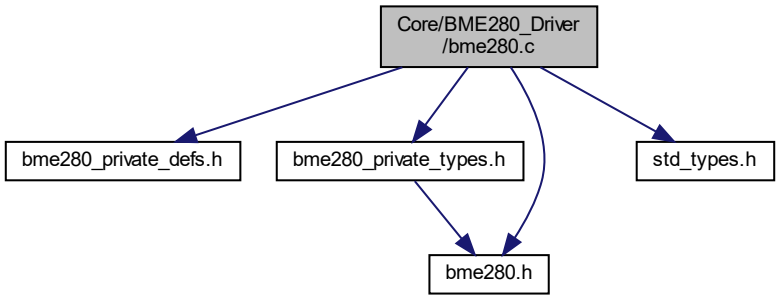
Chapter 4

File Documentation

4.1 Core/BME280_Driver/bme280.c File Reference

```
#include <bme280_private_defs.h>
#include <bme280_private_types.h>
#include "bme280.h"
#include <std_types.h>
```

Include dependency graph for bme280.c:



Functions

- BME280_Status BME280_getInstance (BME280_Handle *a_cfgPtr)

  Obtains an instance from the sensor pool if there is an available instance. Instance is uninitialized and must be initialized by calling BME280_init function to verify and initialize the sensor.
- BME280_Status BME280_setInterfaceType (BME280_Handle *a_cfgPtr, BME280_InterfaceType a_intf)

  API to set interface type (I2C/SPI) before initializing the sensor. Must be set according to the used interface before calling any sensor functions.
- BME280_Status BME280_getChipID (BME280_Handle *a_cfgPtr, BME280_uint8 *a_chipID)
- BME280_Status BME280_init (BME280_Handle *a_cfgPtr)

  Initialization function for the BME280 sensor.
- BME280_Status BME280_softReset (BME280_Handle *a_cfgPtr)

  Soft-resets the sensor by writing the reset byte into reset register.

- BME280_uint16 BME280_calculateMeasurementDelayMs (BME280_Settings *a_settings)

  Calculates measurement time needed by the sensor to finish conversion based on the selected settings.
- BME280_Status BME280_getTemperature_fixedPoint (BME280_Handle *a_cfgPtr, BME280_sint32 *a_temperature)
- BME280_Status BME280_getPressure_fixedPoint (BME280_Handle *a_cfgPtr, BME280_uint32 *a_pressure)
- BME280_Status BME280_getHumidity_fixedPoint (BME280_Handle *a_cfgPtr, BME280_uint32 *a_humidity)
- BME280_Status BME280_getTemperature_floatingPoint (BME280_Handle *a_cfgPtr, BME280_float64 *a_temperature)
- BME280_Status BME280_getPressure_floatingPoint (BME280_Handle *a_cfgPtr, BME280_float64 *a_pressure)
- BME280_Status BME280_getHumidity_floatingPoint (BME280_Handle *a_cfgPtr, BME280_float64 *a_humidity)
- BME280_Status BME280_setPressureOversampling (BME280_Handle *a_cfgPtr, BME280_Oversampling_setting a_pressureOversampling)
- BME280_Status BME280_setTemperatureOversampling (BME280_Handle *a_cfgPtr, BME280_Oversampling_setting a_temperatureOversampling)
- BME280_Status BME280_setHumidityOversampling (BME280_Handle *a_cfgPtr, BME280_Oversampling_setting a_humidityOversampling)
- BME280_Status BME280_setStandbyTime (BME280_Handle *a_cfgPtr, BME280_StandbyTime a_standbyTime)
- BME280_Status BME280_setMode (BME280_Handle *a_cfgPtr, BME280_ModeType a_mode)
- BME280_Status BME280_setFilterCoefficient (BME280_Handle *a_cfgPtr, BME280_FilterCoeff a_filterCoeff)
- BME280_Status BME280_getUpdateStatus (BME280_Handle *a_cfgPtr, BME280_UpdateStatus *a_updateFlag)
- BME280_Status BME280_getMeasuringStatus (BME280_Handle *a_cfgPtr, BME280_MeasuringStatus *a_measureFlag)
- BME280_Status BME280_getMode (BME280_Handle *a_cfgPtr, BME280_ModeType *a_mode)
- BME280_Status BME280_getTemperatureOversampling (BME280_Handle *a_cfgPtr, BME280_Oversampling_setting *a_oversampling)
- BME280_Status BME280_getPressureOversampling (BME280_Handle *a_cfgPtr, BME280_Oversampling_setting *a_oversampling)
- BME280_Status BME280_getHumidityOversampling (BME280_Handle *a_cfgPtr, BME280_Oversampling_setting *a_oversampling)
- BME280_Status BME280_getFilterCoefficient (BME280_Handle *a_cfgPtr, BME280_FilterCoeff *a_filterCoeff)
- BME280_Status BME280_getStandbyTime (BME280_Handle *a_cfgPtr, BME280_StandbyTime *a_standbyTime)
- BME280_Status BME280_getSensorSettings (BME280_Handle *a_cfgPtr, BME280_Settings *a_settings)
- BME280_Status BME280_DeInit (BME280_Handle *a_cfgPtr)

  De-initializes sensor for the passed handle. De-init sequence is:
- BME280_Status BME280_setAssertNSSCallback (BME280_Handle *a_cfgPtr, void(*a_callback)(void))
- BME280_Status BME280_setReleaseNSSCallback (BME280_Handle *a_cfgPtr, void(*a_callback)(void))
- __attribute__ ((weak))

4.1.1 Function Documentation

4.1.1.1 __attribute__()

```
__attribute__ (
             (weak)  )
```

Definition at line 1884 of file bme280.c.

4.1.1.2 BME280_calculateMeasurementDelayMs()

```
BME280_uint16 BME280_calculateMeasurementDelayMs (
             BME280_Settings * a_settings )
```

Calculates measurement time needed by the sensor to finish conversion based on the selected settings.

Parameters

a_settings	Pointer to the settings struct which contains • Filter coefficient • Standby time • Sensor mode • Over-sampling settings for all parameters

Returns

Definition at line 1056 of file bme280.c.

4.1.1.3 BME280_DeInit()

```
BME280_Status BME280_DeInit (
             BME280_Handle * a_cfgPtr )
```

De-initializes sensor for the passed handle. De-init sequence is:

1. Set sensor mode to sleep mode
2. Soft reset the sensor to restore default register values
3. Lets handle point to NULL.

__Note: Does nothing if the handle is NULL or already de-initialized.

Parameters

a_cfgPtr	

Returns

a_cfgPtr is NULL and status flag

Definition at line 1813 of file bme280.c.

4.1.1.4 BME280_getChipID()

```
BME280_Status BME280_getChipID (
            BME280_Handle * a_cfgPtr,
            BME280_uint8 * a_chipID )
```

Definition at line 956 of file bme280.c.

4.1.1.5 BME280_getFilterCoefficient()

```
BME280_Status BME280_getFilterCoefficient (
            BME280_Handle * a_cfgPtr,
            BME280_FilterCoeff * a_filterCoeff )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_filterCoeff	Pointer to variable which will contain the filter coefficient setting

Returns

Definition at line 1756 of file bme280.c.

4.1.1.6 BME280_getHumidity_fixedPoint()

```
BME280_Status BME280_getHumidity_fixedPoint (
            BME280_Handle * a_cfgPtr,
            BME280_uint32 * a_humidity )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_humidity	Pointer to variable to store the humidity in as fixed point (unsigned 32 bit int)

Returns

Definition at line 1207 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.7 BME280_getHumidity_floatingPoint()

```
BME280_Status BME280_getHumidity_floatingPoint (
            BME280_Handle * a_cfgPtr,
            BME280_float64 * a_humidity )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_humidity	Pointer to variable to store the humidity in as floating point (double)

Returns

Definition at line 1354 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.8 BME280_getHumidityOversampling()

```
BME280_Status BME280_getHumidityOversampling (
            BME280_Handle * a_cfgPtr,
            BME280_Oversampling_setting * a_oversampling )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_oversampling	Pointer to variable which will contain the humidity oversampling setting

Returns

Definition at line 1746 of file bme280.c.

4.1.1.9 BME280_getInstance()

```
BME280_Status BME280_getInstance (
            BME280_Handle * a_cfgPtr )
```

Obtains an instance from the sensor pool if there is an available instance. Instance is uninitialized and must be initialized by calling BME280_init function to verify and initialize the sensor.

__Note: If handle is already an occupied instance, the handle is unchanged.

Parameters

a_cfgPtr	Pointer to sensor handle

Returns

Instance for an available sensor in the pool.

Definition at line 881 of file bme280.c.

References BME280_MAX_SENSOR_POOL_SIZE.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.10 BME280_getMeasuringStatus()

```
BME280_Status BME280_getMeasuringStatus (
            BME280_Handle * a_cfgPtr,
            BME280_MeasuringStatus * a_measureFlag )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_measureFlag	Pointer to update status variable which will contain the measuring status

Returns

Definition at line 1688 of file bme280.c.

4.1.1.11 BME280_getMode()

```
BME280_Status BME280_getMode (
            BME280_Handle * a_cfgPtr,
            BME280_ModeType * a_mode )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_mode	Pointer to measure status variable which will contain the sensor mode

Returns

Definition at line 1717 of file bme280.c.

4.1.1.12 BME280_getPressure_fixedPoint()

```
BME280_Status BME280_getPressure_fixedPoint (
            BME280_Handle * a_cfgPtr,
            BME280_uint32 * a_pressure )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_pressure	Pointer to variable to store the pressure in as fixed point (unsigned 32 bit int)

Returns

Definition at line 1155 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.13 BME280_getPressure_floatingPoint()

```
BME280_Status BME280_getPressure_floatingPoint (
            BME280_Handle * a_cfgPtr,
            BME280_float64 * a_pressure )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_pressure	Pointer to variable to store the pressure in as floating point (double)

Returns

Definition at line 1301 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.14 BME280_getPressureOversampling()

```
BME280_Status BME280_getPressureOversampling (
            BME280_Handle * a_cfgPtr,
            BME280_Oversampling_setting * a_oversampling )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_oversampling	Pointer to variable which will contain the pressure oversampling setting

Returns

Definition at line 1736 of file bme280.c.

4.1.1.15 BME280_getSensorSettings()

```
BME280_Status BME280_getSensorSettings (
            BME280_Handle * a_cfgPtr,
            BME280_Settings * a_settings )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_settings	Pointer to settings variable which will be filled with current sensor settings

Returns

Definition at line 1776 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.16 BME280_getStandbyTime()

```
BME280_Status BME280_getStandbyTime (
            BME280_Handle * a_cfgPtr,
            BME280_StandbyTime * a_standbyTime )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_standbyTime	Pointer to variable which will contain the sensor standby time

Returns

Definition at line 1766 of file bme280.c.

4.1.1.17 BME280_getTemperature_fixedPoint()

```
BME280_Status BME280_getTemperature_fixedPoint (
            BME280_Handle * a_cfgPtr,
            BME280_sint32 * a_temperature )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_temperature	Pointer to variable to store the temperature in as fixed point (signed 32 bit int)

Returns

Definition at line 1111 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.18 BME280_getTemperature_floatingPoint()

```
BME280_Status BME280_getTemperature_floatingPoint (
            BME280_Handle * a_cfgPtr,
            BME280_float64 * a_temperature )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_temperature	Pointer to variable to store the temperature in as floating point (double)

Returns

Definition at line 1257 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.19 BME280_getTemperatureOversampling()

```
BME280_Status BME280_getTemperatureOversampling (
            BME280_Handle * a_cfgPtr,
            BME280_Oversampling_setting * a_oversampling )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_oversampling	Pointer to variable which will contain the temperature oversampling setting

Returns

Definition at line 1726 of file bme280.c.

4.1.1.20 BME280_getUpdateStatus()

```
BME280_Status BME280_getUpdateStatus (
            BME280_Handle * a_cfgPtr,
            BME280_UpdateStatus * a_updateFlag )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_updateFlag	Pointer to update status variable which will contain the update status

Returns

Definition at line 1658 of file bme280.c.

4.1.1.21 BME280_init()

```
BME280_Status BME280_init (
            BME280_Handle * a_cfgPtr )
```

Initialization function for the BME280 sensor.

- Initializes sensor by reading its chip ID and soft-resetting it

Returns an empty handle for a new sensor (if pool limit is not reached)

Parameters

a_cfgPtr	Pointer to the sensor configuration struct

Returns

Definition at line 973 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.22 BME280_setAssertNSSCallback()

```
BME280_Status BME280_setAssertNSSCallback (
            BME280_Handle * a_cfgPtr,
            void(*)(void) a_callback )
```

Parameters

a_cfgPtr	
a_callback	

Returns

Definition at line 1845 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.23 BME280_setFilterCoefficient()

```
BME280_Status BME280_setFilterCoefficient (
            BME280_Handle * a_cfgPtr,
            BME280_FilterCoeff a_filterCoeff )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_filterCoeff	

Returns

Sets filter coefficient and status flag if operation succeeded

Definition at line 1617 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.24 BME280_setHumidityOversampling()

```
BME280_Status BME280_setHumidityOversampling (
            BME280_Handle * a_cfgPtr,
            BME280_Oversampling_setting a_humidityOversampling )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_humidityOversampling	

Returns

Sets humidity over-sampling setting and status flag if operation succeeded

Definition at line 1489 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.25 BME280_setInterfaceType()

```
BME280_Status BME280_setInterfaceType (
            BME280_Handle * a_cfgPtr,
            BME280_InterfaceType a_intf )
```

API to set interface type (I2C/SPI) before initializing the sensor. Must be set according to the used interface before calling any sensor functions.

Parameters

a_cfgPtr	Pointer to sensor handle.
a_intf	Interface type selected. See BME280_InterfaceType enum.

Returns

Definition at line 907 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.26 BME280_setMode()

```
BME280_Status BME280_setMode (
            BME280_Handle * a_cfgPtr,
            BME280_ModeType a_mode )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_mode	

Returns

Sets mode and status flag if operation succeeded

Definition at line 1577 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.27 BME280_setPressureOversampling()

```
BME280_Status BME280_setPressureOversampling (
            BME280_Handle * a_cfgPtr,
            BME280_Oversampling_setting a_pressureOversampling )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_pressureOversampling	

Returns

Sets pressure over-sampling setting and status flag if operation succeeded

Definition at line 1408 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.28 BME280_setReleaseNSSCallback()

```
BME280_Status BME280_setReleaseNSSCallback (
            BME280_Handle * a_cfgPtr,
            void(*)(void) a_callback )
```

Parameters

a_cfgPtr	
a_callback	

Returns

Definition at line 1863 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.29 BME280_setStandbyTime()

```
BME280_Status BME280_setStandbyTime (
            BME280_Handle * a_cfgPtr,
            BME280_StandbyTime a_standbyTime )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_standbyTime	

Returns

Sets standby time and status flag if operation succeeded

Definition at line 1537 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.30 BME280_setTemperatureOversampling()

```
BME280_Status BME280_setTemperatureOversampling (
            BME280_Handle * a_cfgPtr,
            BME280_Oversampling_setting a_temperatureOversampling )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_temperatureOversampling	

Returns

Sets temperature over-sampling setting and status flag if operation succeeded

Definition at line 1449 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.1.1.31 BME280_softReset()

```
BME280_Status BME280_softReset (
            BME280_Handle * a_cfgPtr )
```

Soft-resets the sensor by writing the reset byte into reset register.

Parameters

a_cfgPtr	Pointer to the sensor configuration struct

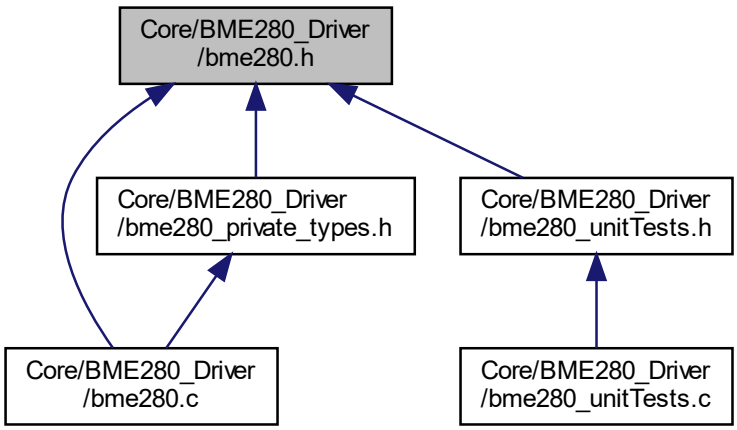
Returns

BME280_Status

Definition at line 1014 of file bme280.c.

4.2 Core/BME280_Driver/bme280.h File Reference

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



Data Structures

- struct BME280_Settings

  Struct that contains sensor user settings.

Typedefs

- typedef struct BME280_ConfigType * BME280_Handle
- typedef unsigned char BME280_boolean
- typedef unsigned char BME280_uint8
- typedef unsigned char BME280_sint8
- typedef unsigned short BME280_uint16
- typedef signed short BME280_sint16
- typedef unsigned long BME280_uint32
- typedef signed long BME280_sint32
- typedef double BME280_float64

Enumerations

- enum BME280_StandbyTime {
  BME280_T_STDBY_0_5_MS = (0x00), BME280_T_STDBY_62_5_MS = (0x01), BME280_T_STDBY_125_MS = (0x02), BME280_T_STDBY_250_MS = (0x03),
  BME280_T_STDBY_500_MS = (0x04), BME280_T_STDBY_1000_MS = (0x05), BME280_T_STDBY_10_MS = (0x06), BME280_T_STDBY_20_MS = (0x07),
  BME280_T_STDBY_NOT_SPECIFIED = (0xFF) }

  Enum for available standby times used in configuration.

- enum BME280_FilterCoeff { BME280_FILTER_COEFF_OFF = (0x00), BME280_FILTER_COEFF_2 = (0x02), BME280_FILTER_COEFF_4 = (0x04), BME280_FILTER_COEFF_8 = (0x08), BME280_FILTER_COEFF_16 = (0x10), BME280_FILTER_NOT_SPECIFIED = (0xFF) }

  Enum for available IIR filter coefficients used in configuration.
- enum BME280_Oversampling_setting { BME280_OVERSAMPLING_OFF = 0x00, BME280_OVERSAMPLING_x1 = 0x01, BME280_OVERSAMPLING_x2 = 0x02, BME280_OVERSAMPLING_x4 = 0x03, BME280_OVERSAMPLING_x8 = 0x04, BME280_OVERSAMPLING_x16 = 0x05, BME280_OVERSAMPLING_NOT_SPECIFIE = (0xFF) }
- enum BME280_Status { BME280_OK = 0, BME280_IS_INSTANCE = 0xF1, BME280_FOUND_EMPTY_INSTANCE = 0x50, BME280_NULL_ERROR = 0xFF, BME280_COMM_ERROR = 0x0A, BME280_NOT_YET_OBTAINED = 0x3F, BME280_NOT_IMPLEMENTED = 0xAC, BME280_POOL_FULL = 0x94, BME280_NOT_INSTANCE = 0x43, BME280_NO_INTERFACE_SPECIFIED = 0x58, BME280_SETTING_FAILED = 0x62, BME280_CALLBACK_NOT_SET = 0xBD }

  Enum for API return status states.
- enum BME280_Comm_Status { BME280_Comm_Error, BME280_Comm_OK }

  Used by the user to return status if communication failed or succeeded based on target implementation. Should be returned through weak functions to be implemented by the user.
- enum BME280_ModeType { BME280_Mode_Sleep = 0b00, BME280_Mode_Forced = 0b10, BME280_Mode_Normal = 0b11, BME280_Mode_Not_Specified = (0xFF) }

  Enum for sensor mode (sleep, normal, forced) used in configuration.
- enum BME280_MeasuringStatus { BME280_Measuring_Finished = 0x00, BME280_Measuring_Running = 0x01 }

  Indicates if the sensor is measuring or has finished measuring.
- enum BME280_UpdateStatus { BME280_Update_Finished = 0x00, BME280_Update_Copying = 0x01 }

  Indicates if the sensor is copying data to registers or has finished copying.
- enum BME280_InterfaceType { BME280_Interface_SPI = 0, BME280_Interface_I2C = 1, BME280_Interface_Not_Specified = (0xFF) }

  Enum for interface used by the sensor.

Functions

- BME280_Status BME280_getInstance (BME280_Handle *a_cfgPtr)

  Obtains an instance from the sensor pool if there is an available instance. Instance is uninitialized and must be initialized by calling BME280_init function to verify and initialize the sensor.
- BME280_Status BME280_setInterfaceType (BME280_Handle *a_cfgPtr, BME280_InterfaceType a_intf)

  API to set interface type (I2C/SPI) before initializing the sensor. Must be set according to the used interface before calling any sensor functions.
- BME280_Status BME280_init (BME280_Handle *a_cfgPtr)

  Initialization function for the BME280 sensor.
- BME280_Status BME280_softReset (BME280_Handle *a_cfgPtr)

  Soft-resets the sensor by writing the reset byte into reset register.
- BME280_uint16 BME280_calculateMeasurementDelayMs (BME280_Settings *a_settings)

  Calculates measurement time needed by the sensor to finish conversion based on the selected settings.
- BME280_uint8 BME280_getChipID (BME280_Handle *a_cfgPtr, BME280_uint8 *a_chipID)
- BME280_Status BME280_getTemperature_floatingPoint (BME280_Handle *a_cfgPtr, BME280_float64 *a_temperature)
- BME280_Status BME280_getPressure_floatingPoint (BME280_Handle *a_cfgPtr, BME280_float64 *a_pressure)
- BME280_Status BME280_getHumidity_floatingPoint (BME280_Handle *a_cfgPtr, BME280_float64 *a_humidity)

- BME280_Status BME280_getTemperature_fixedPoint (BME280_Handle *a_cfgPtr, BME280_sint32 *a_temperature)
- BME280_Status BME280_getPressure_fixedPoint (BME280_Handle *a_cfgPtr, BME280_uint32 *a_pressure)
- BME280_Status BME280_getHumidity_fixedPoint (BME280_Handle *a_cfgPtr, BME280_uint32 *a_humidity)
- BME280_Status BME280_getUpdateStatus (BME280_Handle *a_cfgPtr, BME280_UpdateStatus *a_updateFlag)
- BME280_Status BME280_getMeasuringStatus (BME280_Handle *a_cfgPtr, BME280_MeasuringStatus *a_measureFlag)
- BME280_Status BME280_getMode (BME280_Handle *a_cfgPtr, BME280_ModeType *a_mode)
- BME280_Status BME280_getSensorSettings (BME280_Handle *a_cfgPtr, BME280_Settings *a_settings)
- BME280_Status BME280_getTemperatureOversampling (BME280_Handle *a_cfgPtr, BME280_Oversampling_setting *a_oversampling)
- BME280_Status BME280_getPressureOversampling (BME280_Handle *a_cfgPtr, BME280_Oversampling_setting *a_oversampling)
- BME280_Status BME280_getHumidityOversampling (BME280_Handle *a_cfgPtr, BME280_Oversampling_setting *a_oversampling)
- BME280_Status BME280_getFilterCoefficient (BME280_Handle *a_cfgPtr, BME280_FilterCoeff *a_filterCoeff)
- BME280_Status BME280_getStandbyTime (BME280_Handle *a_cfgPtr, BME280_StandbyTime *a_standbyTime)
- BME280_Status BME280_setPressureOversampling (BME280_Handle *a_cfgPtr, BME280_Oversampling_setting a_pressureOversampling)
- BME280_Status BME280_setTemperatureOversampling (BME280_Handle *a_cfgPtr, BME280_Oversampling_setting a_temperatureOversampling)
- BME280_Status BME280_setHumidityOversampling (BME280_Handle *a_cfgPtr, BME280_Oversampling_setting a_humidityOversampling)
- BME280_Status BME280_setStandbyTime (BME280_Handle *a_cfgPtr, BME280_StandbyTime a_standbyTime)
- BME280_Status BME280_setMode (BME280_Handle *a_cfgPtr, BME280_ModeType a_mode)
- BME280_Status BME280_setFilterCoefficient (BME280_Handle *a_cfgPtr, BME280_FilterCoeff a_filterCoeff)
- BME280_Status BME280_DeInit (BME280_Handle *a_cfgPtr)

  De-initializes sensor for the passed handle. De-init sequence is:
- BME280_Status BME280_setAssertNSSCallback (BME280_Handle *a_cfgPtr, void(*a_callback)(void))
- BME280_Status BME280_setReleaseNSSCallback (BME280_Handle *a_cfgPtr, void(*a_callback)(void))
- BME280_Comm_Status BME280_SPI_TransmitReceive (BME280_uint8 *txData, BME280_uint8 *rxData, BME280_uint16 size, BME280_uint32 timeout)
- BME280_Status BME280_delayMs (BME280_uint32 a_milliseconds)
- BME280_Status BME280_I2C_Master_Transmit (BME280_uint8 sensorAddr, BME280_uint8 *txData, BME280_uint16 size, BME280_uint32 timeout)
- BME280_Status BME280_I2C_Master_Receive (BME280_uint8 sensorAddr, BME280_uint8 *rxData, BME280_uint16 size, BME280_uint32 timeout)

4.2.1 Typedef Documentation

4.2.1.1 BME280_boolean

```
typedef unsigned char BME280_boolean
```

Definition at line 30 of file bme280.h.

4.2.1.2 BME280_float64

```
typedef double BME280_float64
```

Definition at line 72 of file bme280.h.

4.2.1.3 BME280_Handle

```
typedef struct BME280_ConfigType* BME280_Handle
```

Definition at line 24 of file bme280.h.

4.2.1.4 BME280_sint16

```
typedef signed short BME280_sint16
```

Definition at line 54 of file bme280.h.

4.2.1.5 BME280_sint32

```
typedef signed long BME280_sint32
```

Definition at line 66 of file bme280.h.

4.2.1.6 BME280_sint8

```
typedef unsigned char BME280_sint8
```

Definition at line 42 of file bme280.h.

4.2.1.7 BME280_uint16

```
typedef unsigned short BME280_uint16
```

Definition at line 48 of file bme280.h.

4.2.1.8 BME280_uint32

```
typedef unsigned long BME280_uint32
```

Definition at line 60 of file bme280.h.

4.2.1.9 BME280_uint8

```
typedef unsigned char BME280_uint8
```

Definition at line 36 of file bme280.h.

4.2.2 Enumeration Type Documentation

4.2.2.1 BME280_Comm_Status

```
enum BME280_Comm_Status
```

Used by the user to return status if communication failed or succeeded based on target implementation. Should be returned through weak functions to be implemented by the user.

Enumerator

BME280_Comm_Error	Communication occurred during transmission
BME280_Comm_OK	Communication success

Definition at line 148 of file bme280.h.

4.2.2.2 BME280_FilterCoeff

```
enum BME280_FilterCoeff
```

Enum for available IIR filter coefficients used in configuration.

Enumerator

BME280_FILTER_COEFF_OFF	Filter off
BME280_FILTER_COEFF_2	2
BME280_FILTER_COEFF_4	4

Enumerator

BME280_FILTER_COEFF_8	8
BME280_FILTER_COEFF_16	16
BME280_FILTER_NOT_SPECIFIED	

Definition at line 95 of file bme280.h.

4.2.2.3 BME280_InterfaceType

`enum BME280_InterfaceType`

Enum for interface used by the sensor.

Enumerator

BME280_Interface_SPI	Selected SPI interface
BME280_Interface_I2C	Selected I2C interface
BME280_Interface_Not_Specified	

Definition at line 187 of file bme280.h.

4.2.2.4 BME280_MeasuringStatus

`enum BME280_MeasuringStatus`

Indicates if the sensor is measuring or has finished measuring.

Enumerator

BME280_Measuring_Finished	
BME280_Measuring_Running	

Definition at line 169 of file bme280.h.

4.2.2.5 BME280_ModeType

`enum BME280_ModeType`

Enum for sensor mode (sleep, normal, forced) used in configuration.

Enumerator

BME280_Mode_Sleep	
BME280_Mode_Forced	
BME280_Mode_Normal	
BME280_Mode_Not_Specified	

Definition at line 157 of file bme280.h.

4.2.2.6 BME280_Oversampling_setting

`enum BME280_Oversampling_setting`

Enumerator

BME280_OVERSAMPLING_OFF	Skipped measurement
BME280_OVERSAMPLING_x1	x1
BME280_OVERSAMPLING_x2	x2
BME280_OVERSAMPLING_x4	x4
BME280_OVERSAMPLING_x8	x8
BME280_OVERSAMPLING_x16	x16
BME280_OVERSAMPLING_NOT_SPECIFIED	

Definition at line 107 of file bme280.h.

4.2.2.7 BME280_StandbyTime

`enum BME280_StandbyTime`

Enum for available standby times used in configuration.

Enumerator

BME280_T_STDBY_0_5_MS	0.5ms
BME280_T_STDBY_62_5_MS	62.5ms
BME280_T_STDBY_125_MS	125ms
BME280_T_STDBY_250_MS	250ms
BME280_T_STDBY_500_MS	500ms
BME280_T_STDBY_1000_MS	1000ms
BME280_T_STDBY_10_MS	10ms
BME280_T_STDBY_20_MS	20ms
BME280_T_STDBY_NOT_SPECIFIED	

Definition at line 79 of file bme280.h.

4.2.2.8 BME280_Status

enum BME280_Status

Enum for API return status states.

Enumerator

BME280_OK	General code for operation success
BME280_IS_INSTANCE	Code indicating that the handle is already an instance in the sensor pool
BME280_FOUND_EMPTY_INSTANCE	Code indicating that there is an empty instance found in the sensor pool
BME280_NULL_ERROR	General code for null arguments to functions
BME280_COMM_ERROR	General code for communication failure, not to be confused with users' BME280_Comm_Error in BME280_Comm_Status which the user uses to implement weak functions
BME280_NOT_YET_OBTAINED	General code used for status flag when the result has not been obtained yet through API functions
BME280_NOT_IMPLEMENTED	General error code for when weak functions are not implemented by the user
BME280_POOL_FULL	Error code indicating that the sensor pool is full
BME280_NOT_INSTANCE	Error code indicating that the handle is not an existing instance in the pool
BME280_NO_INTERFACE_SPECIFIED	Error code indicating that there was no communication interface specified, see BME280_setInterfaceType function
BME280_SETTING_FAILED	General error code returned if a sensor setting was not set and validated
BME280_CALLBACK_NOT_SET	Error code when callbacks for GPIO functions are not set by the user

Definition at line 121 of file bme280.h.

4.2.2.9 BME280_UpdateStatus

enum BME280_UpdateStatus

Indicates if the sensor is copying data to registers or has finished copying.

Enumerator

BME280_Update_Finished	
BME280_Update_Copying	

Definition at line 179 of file bme280.h.

4.2.3 Function Documentation

4.2.3.1 BME280_calculateMeasurementDelayMs()

```
BME280_uint16 BME280_calculateMeasurementDelayMs (
            BME280_Settings * a_settings )
```

Calculates measurement time needed by the sensor to finish conversion based on the selected settings.

Parameters

a_settings	Pointer to the settings struct which contains • Filter coefficient • Standby time • Sensor mode • Over-sampling settings for all parameters

Returns

Definition at line 1056 of file bme280.c.

4.2.3.2 BME280_DeInit()

```
BME280_Status BME280_DeInit (
            BME280_Handle * a_cfgPtr )
```

De-initializes sensor for the passed handle. De-init sequence is:

1. Set sensor mode to sleep mode
2. Soft reset the sensor to restore default register values
3. Lets handle point to NULL.

__Note: Does nothing if the handle is NULL or already de-initialized.

Parameters

a_cfgPtr	

Returns

a_cfgPtr is NULL and status flag

Definition at line 1813 of file bme280.c.

4.2.3.3 BME280_delayMs()

```
BME280_Status BME280_delayMs (
            BME280_uint32 a_milliseconds )
```

4.2.3.4 BME280_getChipID()

```
BME280_uint8 BME280_getChipID (
            BME280_Handle * a_cfgPtr,
            BME280_uint8 * a_chipID )
```

Definition at line 956 of file bme280.c.

4.2.3.5 BME280_getFilterCoefficient()

```
BME280_Status BME280_getFilterCoefficient (
            BME280_Handle * a_cfgPtr,
            BME280_FilterCoeff * a_filterCoeff )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_filterCoeff	Pointer to variable which will contain the filter coefficient setting

Returns

Definition at line 1756 of file bme280.c.

4.2.3.6 BME280_getHumidity_fixedPoint()

```
BME280_Status BME280_getHumidity_fixedPoint (
            BME280_Handle * a_cfgPtr,
            BME280_uint32 * a_humidity )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_humidity	Pointer to variable to store the humidity in as fixed point (unsigned 32 bit int)

Returns

Definition at line 1207 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.7 BME280_getHumidity_floatingPoint()

```
BME280_Status BME280_getHumidity_floatingPoint (
            BME280_Handle * a_cfgPtr,
            BME280_float64 * a_humidity )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_humidity	Pointer to variable to store the humidity in as floating point (double)

Returns

Definition at line 1354 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.8 BME280_getHumidityOversampling()

```
BME280_Status BME280_getHumidityOversampling (
            BME280_Handle * a_cfgPtr,
            BME280_Oversampling_setting * a_oversampling )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_oversampling	Pointer to variable which will contain the humidity oversampling setting

Returns

Definition at line 1746 of file bme280.c.

4.2.3.9 BME280_getInstance()

```
BME280_Status BME280_getInstance (
            BME280_Handle * a_cfgPtr )
```

Obtains an instance from the sensor pool if there is an available instance. Instance is uninitialized and must be initialized by calling BME280_init function to verify and initialize the sensor.

__Note: If handle is already an occupied instance, the handle is unchanged.

Parameters

a_cfgPtr	Pointer to sensor handle

Returns

Instance for an available sensor in the pool.

Definition at line 881 of file bme280.c.

References BME280_MAX_SENSOR_POOL_SIZE.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.10 BME280_getMeasuringStatus()

```
BME280_Status BME280_getMeasuringStatus (
            BME280_Handle * a_cfgPtr,
            BME280_MeasuringStatus * a_measureFlag )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_measureFlag	Pointer to update status variable which will contain the measuring status

Returns

Definition at line 1688 of file bme280.c.

4.2.3.11 BME280_getMode()

```
BME280_Status BME280_getMode (
            BME280_Handle * a_cfgPtr,
            BME280_ModeType * a_mode )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_mode	Pointer to measure status variable which will contain the sensor mode

Returns

Definition at line 1717 of file bme280.c.

4.2.3.12 BME280_getPressure_fixedPoint()

```
BME280_Status BME280_getPressure_fixedPoint (
            BME280_Handle * a_cfgPtr,
            BME280_uint32 * a_pressure )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_pressure	Pointer to variable to store the pressure in as fixed point (unsigned 32 bit int)

Returns

Definition at line 1155 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.13 BME280_getPressure_floatingPoint()

```
BME280_Status BME280_getPressure_floatingPoint (
            BME280_Handle * a_cfgPtr,
            BME280_float64 * a_pressure )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_pressure	Pointer to variable to store the pressure in as floating point (double)

Returns

Definition at line 1301 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.14 BME280_getPressureOversampling()

```
BME280_Status BME280_getPressureOversampling (
            BME280_Handle * a_cfgPtr,
            BME280_Oversampling_setting * a_oversampling )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_oversampling	Pointer to variable which will contain the pressure oversampling setting

Returns

Definition at line 1736 of file bme280.c.

4.2.3.15 BME280_getSensorSettings()

```
BME280_Status BME280_getSensorSettings (
            BME280_Handle * a_cfgPtr,
            BME280_Settings * a_settings )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_settings	Pointer to settings variable which will be filled with current sensor settings

Returns

Definition at line 1776 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.16 BME280_getStandbyTime()

```
BME280_Status BME280_getStandbyTime (
            BME280_Handle * a_cfgPtr,
            BME280_StandbyTime * a_standbyTime )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_standbyTime	Pointer to variable which will contain the sensor standby time

Returns

Definition at line 1766 of file bme280.c.

4.2.3.17 BME280_getTemperature_fixedPoint()

```
BME280_Status BME280_getTemperature_fixedPoint (
            BME280_Handle * a_cfgPtr,
            BME280_sint32 * a_temperature )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_temperature	Pointer to variable to store the temperature in as fixed point (signed 32 bit int)

Returns

Definition at line 1111 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.18 BME280_getTemperature_floatingPoint()

```
BME280_Status BME280_getTemperature_floatingPoint (
            BME280_Handle * a_cfgPtr,
            BME280_float64 * a_temperature )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_temperature	Pointer to variable to store the temperature in as floating point (double)

Returns

Definition at line 1257 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.19 BME280_getTemperatureOversampling()

```
BME280_Status BME280_getTemperatureOversampling (
            BME280_Handle * a_cfgPtr,
            BME280_Oversampling_setting * a_oversampling )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_oversampling	Pointer to variable which will contain the temperature oversampling setting

Returns

Definition at line 1726 of file bme280.c.

4.2.3.20 BME280_getUpdateStatus()

```
BME280_Status BME280_getUpdateStatus (
            BME280_Handle * a_cfgPtr,
            BME280_UpdateStatus * a_updateFlag )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_updateFlag	Pointer to update status variable which will contain the update status

Returns

Definition at line 1658 of file bme280.c.

4.2.3.21 BME280_I2C_Master_Receive()

```
BME280_Status BME280_I2C_Master_Receive (
            BME280_uint8 sensorAddr,
            BME280_uint8 * rxData,
            BME280_uint16 size,
            BME280_uint32 timeout )
```

Parameters

sensorAddr	Sensor address, assumed to be masked with the R/W bit from the internal API functions
rxData	Pointer to buffer to receive data in
size	Number of bytes to receive in rxData
timeout	Timeout in milliseconds

Returns

4.2.3.22 BME280_I2C_Master_Transmit()

```
BME280_Status BME280_I2C_Master_Transmit (
            BME280_uint8 sensorAddr,
            BME280_uint8 * txData,
            BME280_uint16 size,
            BME280_uint32 timeout )
```

Parameters

sensorAddr	Sensor address, assumed to be masked with the R/W bit from the internal API functions
txData	Pointer to data to be transmitted
size	Number of bytes to transmit
timeout	Timeout in milliseconds

Returns

4.2.3.23 BME280_init()

```
BME280_Status BME280_init (
            BME280_Handle * a_cfgPtr )
```

Initialization function for the BME280 sensor.

- Initializes sensor by reading its chip ID and soft-resetting it

Returns an empty handle for a new sensor (if pool limit is not reached)

Parameters

a_cfgPtr	Pointer to the sensor configuration struct

Returns

Definition at line 973 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.24 BME280_setAssertNSSCallback()

```
BME280_Status BME280_setAssertNSSCallback (
            BME280_Handle * a_cfgPtr,
            void(*)(void) a_callback )
```

Parameters

a_cfgPtr	
a_callback	

Returns

Definition at line 1845 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.25 BME280_setFilterCoefficient()

```
BME280_Status BME280_setFilterCoefficient (
            BME280_Handle * a_cfgPtr,
            BME280_FilterCoeff a_filterCoeff )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_filterCoeff	

Returns

Sets filter coefficient and status flag if operation succeeded

Definition at line 1617 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.26 BME280_setHumidityOversampling()

```
BME280_Status BME280_setHumidityOversampling (
            BME280_Handle * a_cfgPtr,
            BME280_Oversampling_setting a_humidityOversampling )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_humidityOversampling	

Returns

Sets humidity over-sampling setting and status flag if operation succeeded

Definition at line 1489 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.27 BME280_setInterfaceType()

```
BME280_Status BME280_setInterfaceType (
            BME280_Handle * a_cfgPtr,
            BME280_InterfaceType a_intf )
```

API to set interface type (I2C/SPI) before initializing the sensor. Must be set according to the used interface before calling any sensor functions.

Parameters

a_cfgPtr	Pointer to sensor handle.
a_intf	Interface type selected. See BME280_InterfaceType enum.

Returns

Definition at line 907 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.28 BME280_setMode()

```
BME280_Status BME280_setMode (
            BME280_Handle * a_cfgPtr,
            BME280_ModeType a_mode )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_mode	

Returns

Sets mode and status flag if operation succeeded

Definition at line 1577 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.29 BME280_setPressureOversampling()

```
BME280_Status BME280_setPressureOversampling (
            BME280_Handle * a_cfgPtr,
            BME280_Oversampling_setting a_pressureOversampling )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_pressureOversampling	

Returns

Sets pressure over-sampling setting and status flag if operation succeeded

Definition at line 1408 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.30 BME280_setReleaseNSSCallback()

```
BME280_Status BME280_setReleaseNSSCallback (
            BME280_Handle * a_cfgPtr,
            void(*)(void) a_callback )
```

Parameters

a_cfgPtr	
a_callback	

Returns

Definition at line 1863 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.31 BME280_setStandbyTime()

```
BME280_Status BME280_setStandbyTime (
            BME280_Handle * a_cfgPtr,
            BME280_StandbyTime a_standbyTime )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_standbyTime	

Returns

Sets standby time and status flag if operation succeeded

Definition at line 1537 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.32 BME280_setTemperatureOversampling()

```
BME280_Status BME280_setTemperatureOversampling (
            BME280_Handle * a_cfgPtr,
            BME280_Oversampling_setting a_temperatureOversampling )
```

Parameters

a_cfgPtr	Pointer to sensor handle
a_temperatureOversampling	

Returns

Sets temperature over-sampling setting and status flag if operation succeeded

Definition at line 1449 of file bme280.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.2.3.33 BME280_softReset()

```
BME280_Status BME280_softReset (
            BME280_Handle * a_cfgPtr )
```

Soft-resets the sensor by writing the reset byte into reset register.

Parameters

a_cfgPtr	Pointer to the sensor configuration struct

Returns

BME280_Status

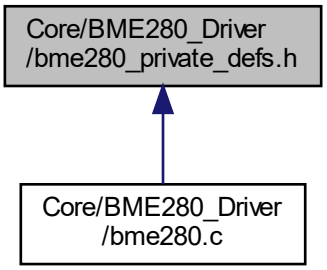
Definition at line 1014 of file bme280.c.

4.2.3.34 BME280_SPI_TransmitReceive()

```
BME280_Comm_Status BME280_SPI_TransmitReceive (
            BME280_uint8 * txData,
            BME280_uint8 * rxData,
            BME280_uint16 size,
            BME280_uint32 timeout )
```

4.3 Core/BME280_Driver/bme280_private_defs.h File Reference

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



Macros

- #define BME280_MAX_PRESSURE_FIXED_POINT (110000)
- #define BME280_MIN_PRESSURE_FIXED_POINT (30000)
- #define BME280_MAX_HUMIDITY_FIXED_POINT (102400)
- #define BME280_MIN_HUMIDITY_FIXED_POINT (0)
- #define BME280_MAX_TEMPERATURE_FIXED_POINT (8500)
- #define BME280_MIN_TEMPERATURE_FIXED_POINT (-4000)
- #define BME280_MAX_PRESSURE_FLOATING_POINT (110000.0)
- #define BME280_MIN_PRESSURE_FLOATING_POINT (30000.0)
- #define BME280_MAX_HUMIDITY_FLOATING_POINT (100.0)
- #define BME280_MIN_HUMIDITY_FLOATING_POINT (0.0)
- #define BME280_MAX_TEMPERATURE_FLOATING_POINT (85)
- #define BME280_MIN_TEMPERATURE_FLOATING_POINT (-40)
- #define BME280_MAX_DISCOVERY_COUNT ((uint8)0x05) /* Max discovery count of BME280 device on interface*/
- #define BME280_IM_UPDATE_READY ((uint8)0x00) /* IM UPDATE flag ready state */
- #define BME280_MEASURING_DONE ((uint8)0x00) /* Measuring flag ready state */
- #define BME280_SPI_READ_MASK(reg) (reg|0x80) /* Mask read register address in case of SPI*/
- #define BME280_SPI_WRITE_MASK(reg) (reg&0x7F) /* Mask write register address in case of SPI*/
- #define BME280_SPI_TIMEOUT_MS (10) /* SPI timeout in milliseconds, used in BME280_SPI_TransmitReceive function */
- #define BME280_I2C_READ_MASK(reg) ((reg<<1)&0xFF) /* Mask read register address in case of I2C*/
- #define BME280_I2C_WRITE_MASK(reg) ((reg<<1)&0xFE) /* Mask write register address in case of I2C*/
- #define BME280_I2C_TIMEOUT_MS (100) /* I2C timeout in milliseconds, used in BME280_I2C_Master_Transmit/Receive function */
- #define BME280_LSBYTE_MASK (0x0F)
- #define BME280_RESET_WORD (0xB6) /* Reset word which will reset BME280 when written into reset register*/
- #define BME280_READINGS_BYTES_LENGTH (8) /* Temperature, humidity, and pressure readings length in bytes */
- #define BME280_START_UP_TIME_MS ((uint8)0x02) /* Sensor boot time according to data-sheet in milliseconds*/

- #define BME280_CHIP_ID ((uint8)(0x60)) /* Sensor chip ID, always the same for BME280, non-editable*/
- #define BME280_TEMP_PRESS_CALIB_BLOCK_SIZE (26) /* Temperature and pressure calibration block size in bytes mapped to sensor */
- #define BME280_HUM_CALIB_BLOCK_SIZE (7) /* Humidity calibration block size in bytes mapped to sensor*/
- #define BME280_CALIB_1_BLOCK_SIZE (13) /* Calib 1 block size mapped to structures */
- #define BME280_CALIB_2_BLOCK_SIZE (19) /* Calib 2 block size mapped to structures */
- #define BME280_MAX_SENSOR_POOL_SIZE ((4)) /* Maximum pool size of available sensor instances */
- #define BME280_HUM_REGISTER_LSB (0xFE)
- #define BME280_HUM_REGISTER_MSB (0xFD)
- #define BME280_TEMP_REGISTER_XLSB (0xFC)
- #define BME280_TEMP_REGISTER_LSB (0xFB)
- #define BME280_TEMP_REGISTER_MSB (0xFA)
- #define BME280_PRESS_REGISTER_XLSB (0xF9)
- #define BME280_PRESS_REGISTER_LSB (0xF8)
- #define BME280_PRESS_REGISTER_MSB (0xF7)
- #define BME280_START_READINGS_ADDRESS (0xF7)
- #define BME280_TEMP_PRESS_BLOCK_START_ADDRESS (0x88)
- #define BME280_HUM_BLOCK_START_ADDRESS (0xE1)
- #define BME280_CONFIG_REGISTER (0xF5)
- #define BME280_CTRL_MEAS_REGISTER (0xF4)
- #define BME280_STATUS_REGISTER (0xF3)
- #define BME280_CTRL_HUM_REGISTER (0xF2)
- #define BME280_RESET_REGISTER (0xE0)
- #define BME280_ID_REGISTER (0xD0)
- #define BME280_CONFIG_IM_UPDATE_MASK(config_reg) (config_reg & (uint8)0x01)
- #define BME280_CONFIG_MEAS_MASK(config_reg) ((config_reg>>3) & (uint8)0x01)

4.3.1 Macro Definition Documentation

4.3.1.1 BME280_CALIB_1_BLOCK_SIZE

```
#define BME280_CALIB_1_BLOCK_SIZE (13) /* Calib 1 block size mapped to structures */
```

Definition at line 63 of file bme280_private_defs.h.

4.3.1.2 BME280_CALIB_2_BLOCK_SIZE

```
#define BME280_CALIB_2_BLOCK_SIZE (19) /* Calib 2 block size mapped to structures */
```

Definition at line 64 of file bme280_private_defs.h.

4.3.1.3 BME280_CHIP_ID

```
#define BME280_CHIP_ID ((uint8)(0x60)) /* Sensor chip ID, always the same for BME280, non-editable*/
```

Definition at line 59 of file bme280_private_defs.h.

4.3.1.4 BME280_CONFIG_IM_UPDATE_MASK

```
#define BME280_CONFIG_IM_UPDATE_MASK(
              config_reg ) (config_reg & (uint8)0x01)
```

Definition at line 93 of file bme280_private_defs.h.

4.3.1.5 BME280_CONFIG_MEAS_MASK

```
#define BME280_CONFIG_MEAS_MASK(
              config_reg ) ((config_reg>>3) & (uint8)0x01)
```

Definition at line 94 of file bme280_private_defs.h.

4.3.1.6 BME280_CONFIG_REGISTER

```
#define BME280_CONFIG_REGISTER (0xF5)
```

Definition at line 82 of file bme280_private_defs.h.

4.3.1.7 BME280_CTRL_HUM_REGISTER

```
#define BME280_CTRL_HUM_REGISTER (0xF2)
```

Definition at line 86 of file bme280_private_defs.h.

4.3.1.8 BME280_CTRL_MEAS_REGISTER

```
#define BME280_CTRL_MEAS_REGISTER (0xF4)
```

Definition at line 83 of file bme280_private_defs.h.

4.3.1.9 BME280_HUM_BLOCK_START_ADDRESS

```
#define BME280_HUM_BLOCK_START_ADDRESS (0xE1)
```

Definition at line 80 of file bme280_private_defs.h.

4.3.1.10 BME280_HUM_CALIB_BLOCK_SIZE

```
#define BME280_HUM_CALIB_BLOCK_SIZE (7) /* Humidity calibration block size in bytes mapped to
sensor*/
```

Definition at line 62 of file bme280_private_defs.h.

4.3.1.11 BME280_HUM_REGISTER_LSB

```
#define BME280_HUM_REGISTER_LSB (0xFE)
```

Definition at line 67 of file bme280_private_defs.h.

4.3.1.12 BME280_HUM_REGISTER_MSB

```
#define BME280_HUM_REGISTER_MSB (0xFD)
```

Definition at line 68 of file bme280_private_defs.h.

4.3.1.13 BME280_I2C_READ_MASK

```
#define BME280_I2C_READ_MASK(
              reg ) ((reg<<1)&0xFF) /* Mask read register address in case of I2C*/
```

Definition at line 49 of file bme280_private_defs.h.

4.3.1.14 BME280_I2C_TIMEOUT_MS

```
#define BME280_I2C_TIMEOUT_MS (100) /* I2C timeout in milliseconds, used in BME280_I2C_Master_Transmit/Receive
function */
```

Definition at line 51 of file bme280_private_defs.h.

4.3.1.15 BME280_I2C_WRITE_MASK

```
#define BME280_I2C_WRITE_MASK(
              reg ) ((reg<<1)&0xFE) /* Mask write register address in case of I2C*/
```

Definition at line 50 of file bme280_private_defs.h.

4.3.1.16 BME280_ID_REGISTER

```
#define BME280_ID_REGISTER (0xD0)
```

Definition at line 88 of file bme280_private_defs.h.

4.3.1.17 BME280_IM_UPDATE_READY

```
#define BME280_IM_UPDATE_READY ((uint8)0x00) /* IM UPDATE flag ready state */
```

Definition at line 43 of file bme280_private_defs.h.

4.3.1.18 BME280_LSBYTE_MASK

```
#define BME280_LSBYTE_MASK (0x0F)
```

Definition at line 52 of file bme280_private_defs.h.

4.3.1.19 BME280_MAX_DISCOVERY_COUNT

```
#define BME280_MAX_DISCOVERY_COUNT ((uint8)0x05) /* Max discovery count of BME280 device on
interface*/
```

Definition at line 42 of file bme280_private_defs.h.

4.3.1.20 BME280_MAX_HUMIDITY_FIXED_POINT

```
#define BME280_MAX_HUMIDITY_FIXED_POINT (102400)
```

Definition at line 25 of file bme280_private_defs.h.

4.3.1.21 BME280_MAX_HUMIDITY_FLOATING_POINT

```
#define BME280_MAX_HUMIDITY_FLOATING_POINT (100.0)
```

Definition at line 36 of file bme280_private_defs.h.

4.3.1.22 BME280_MAX_PRESSURE_FIXED_POINT

```
#define BME280_MAX_PRESSURE_FIXED_POINT (110000)
```

Definition at line 22 of file bme280_private_defs.h.

4.3.1.23 BME280_MAX_PRESSURE_FLOATING_POINT

```
#define BME280_MAX_PRESSURE_FLOATING_POINT (110000.0)
```

Definition at line 33 of file bme280_private_defs.h.

4.3.1.24 BME280_MAX_SENSOR_POOL_SIZE

```
#define BME280_MAX_SENSOR_POOL_SIZE ((4)) /* Maximum pool size of available sensor instances */
```

Definition at line 65 of file bme280_private_defs.h.

4.3.1.25 BME280_MAX_TEMPERATURE_FIXED_POINT

```
#define BME280_MAX_TEMPERATURE_FIXED_POINT (8500)
```

Definition at line 28 of file bme280_private_defs.h.

4.3.1.26 BME280_MAX_TEMPERATURE_FLOATING_POINT

```
#define BME280_MAX_TEMPERATURE_FLOATING_POINT (85)
```

Definition at line 39 of file bme280_private_defs.h.

4.3.1.27 BME280_MEASURING_DONE

```
#define BME280_MEASURING_DONE ((uint8)0x00) /* Measuring flag ready state */
```

Definition at line 44 of file bme280_private_defs.h.

4.3.1.28 BME280_MIN_HUMIDITY_FIXED_POINT

```
#define BME280_MIN_HUMIDITY_FIXED_POINT (0)
```

Definition at line 26 of file bme280_private_defs.h.

4.3.1.29 BME280_MIN_HUMIDITY_FLOATING_POINT

```
#define BME280_MIN_HUMIDITY_FLOATING_POINT (0.0)
```

Definition at line 37 of file bme280_private_defs.h.

4.3.1.30 BME280_MIN_PRESSURE_FIXED_POINT

```
#define BME280_MIN_PRESSURE_FIXED_POINT (30000)
```

Definition at line 23 of file bme280_private_defs.h.

4.3.1.31 BME280_MIN_PRESSURE_FLOATING_POINT

```
#define BME280_MIN_PRESSURE_FLOATING_POINT (30000.0)
```

Definition at line 34 of file bme280_private_defs.h.

4.3.1.32 BME280_MIN_TEMPERATURE_FIXED_POINT

```
#define BME280_MIN_TEMPERATURE_FIXED_POINT (-4000)
```

Definition at line 29 of file bme280_private_defs.h.

4.3.1.33 BME280_MIN_TEMPERATURE_FLOATING_POINT

```
#define BME280_MIN_TEMPERATURE_FLOATING_POINT (-40)
```

Definition at line 40 of file bme280_private_defs.h.

4.3.1.34 BME280_PRESS_REGISTER_LSB

```
#define BME280_PRESS_REGISTER_LSB (0xF8)
```

Definition at line 75 of file bme280_private_defs.h.

4.3.1.35 BME280_PRESS_REGISTER_MSB

```
#define BME280_PRESS_REGISTER_MSB (0xF7)
```

Definition at line 76 of file bme280_private_defs.h.

4.3.1.36 BME280_PRESS_REGISTER_XLSB

```
#define BME280_PRESS_REGISTER_XLSB (0xF9)
```

Definition at line 74 of file bme280_private_defs.h.

4.3.1.37 BME280_READINGS_BYTES_LENGTH

```
#define BME280_READINGS_BYTES_LENGTH (8) /* Temperature, humidity, and pressure readings length
in bytes */
```

Definition at line 57 of file bme280_private_defs.h.

4.3.1.38 BME280_RESET_REGISTER

```
#define BME280_RESET_REGISTER (0xE0)
```

Definition at line 87 of file bme280_private_defs.h.

4.3.1.39 BME280_RESET_WORD

```
#define BME280_RESET_WORD (0xB6) /* Reset word which will reset BME280 when written into reset
register*/
```

Definition at line 56 of file bme280_private_defs.h.

4.3.1.40 BME280_SPI_READ_MASK

```
#define BME280_SPI_READ_MASK(
              reg ) (reg|0x80) /* Mask read register address in case of SPI*/
```

Definition at line 45 of file bme280_private_defs.h.

4.3.1.41 BME280_SPI_TIMEOUT_MS

```
#define BME280_SPI_TIMEOUT_MS (10) /* SPI timeout in milliseconds, used in BME280_SPI_TransmitReceive
function */
```

Definition at line 47 of file bme280_private_defs.h.

4.3.1.42 BME280_SPI_WRITE_MASK

```
#define BME280_SPI_WRITE_MASK(
              reg ) (reg&0x7F) /* Mask write register address in case of SPI*/
```

Definition at line 46 of file bme280_private_defs.h.

4.3.1.43 BME280_START_READINGS_ADDRESS

```
#define BME280_START_READINGS_ADDRESS (0xF7)
```

Definition at line 78 of file bme280_private_defs.h.

4.3.1.44 BME280_START_UP_TIME_MS

```
#define BME280_START_UP_TIME_MS ((uint8)0x02) /* Sensor boot time according to data-sheet in
milliseconds*/
```

Definition at line 58 of file bme280_private_defs.h.

4.3.1.45 BME280_STATUS_REGISTER

```
#define BME280_STATUS_REGISTER (0xF3)
```

Definition at line 84 of file bme280_private_defs.h.

4.3.1.46 BME280_TEMP_PRESS_BLOCK_START_ADDRESS

```
#define BME280_TEMP_PRESS_BLOCK_START_ADDRESS (0x88)
```

Definition at line 79 of file bme280_private_defs.h.

4.3.1.47 BME280_TEMP_PRESS_CALIB_BLOCK_SIZE

```
#define BME280_TEMP_PRESS_CALIB_BLOCK_SIZE (26) /* Temperature and pressure calibration block
size in bytes mapped to sensor */
```

Definition at line 61 of file bme280_private_defs.h.

4.3.1.48 BME280_TEMP_REGISTER_LSB

```
#define BME280_TEMP_REGISTER_LSB (0xFB)
```

Definition at line 71 of file bme280_private_defs.h.

4.3.1.49 BME280_TEMP_REGISTER_MSB

```
#define BME280_TEMP_REGISTER_MSB (0xFA)
```

Definition at line 72 of file bme280_private_defs.h.

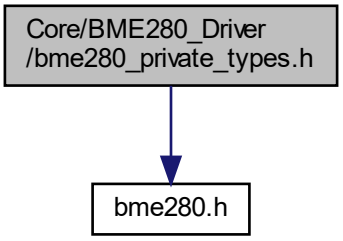
4.3.1.50 BME280_TEMP_REGISTER_XLSB

```
#define BME280_TEMP_REGISTER_XLSB (0xFC)
```

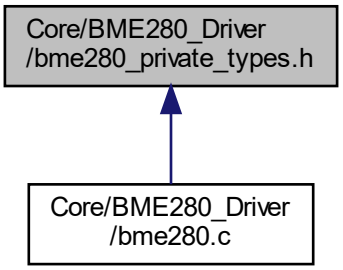
Definition at line 70 of file bme280_private_defs.h.

4.4 Core/BME280_Driver/bme280_private_types.h File Reference

`#include <bme280.h>`
Include dependency graph for bme280_private_types.h:



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



Data Structures

- union BME280_ConfigRegisterUnion

  Union used to create configuration register byte.
- union BME280_TemperatureReading

  Used to store readings from the sensor and access them directly from 'temperature' member.
- union BME280_PressureReading

  Used to store readings from the sensor and access them directly from 'pressure' member.
- union BME280_HumidityReading

  Used to store readings from the sensor and access them directly from 'humidity' member.
- union BME280_CtrlMeasRegisterUnion

  Union used to create control measurements register byte.
- union BME280_CtrlHumRegisterUnion

Union used to create control humidity register byte.

- union BME280_StatusRegisterUnion

  Union used to create status register byte.

- struct BME280_ConfigType

  Configuration structure used for setup, configuration, and sensor functions.

Enumerations

- enum BME280_I2C_SensorSlaveAddress { BME280_I2C_Addr = (0x76) }

  *Used to identify BME280 slave address in I2C interface mode. Must be set if the device's SDO is connected to **GND** datasheet.*

Functions

- union __attribute__ ((packed, aligned(1)))

Variables

- Calib1
- Calib2
- BME280_UncompensatedReadings

4.4.1 Enumeration Type Documentation

4.4.1.1 BME280_I2C_SensorSlaveAddress

`enum BME280_I2C_SensorSlaveAddress`

Used to identify BME280 slave address in I2C interface mode. Must be set if the device's SDO is connected to **GND** datasheet.

Enumerator

BME280_I2C_Addr	

Definition at line 210 of file bme280_private_types.h.

4.4.2 Function Documentation

4.4.2.1 __attribute__()

```
union __attribute__ (
            (packed, aligned(1))  )
```

< Padding

Definition at line 43 of file bme280_private_types.h.

4.4.3 Variable Documentation

4.4.3.1 BME280_UncompensatedReadings

`BME280_UncompensatedReadings`

Definition at line 117 of file bme280_private_types.h.

4.4.3.2 Calib1

`Calib1`

Definition at line 64 of file bme280_private_types.h.

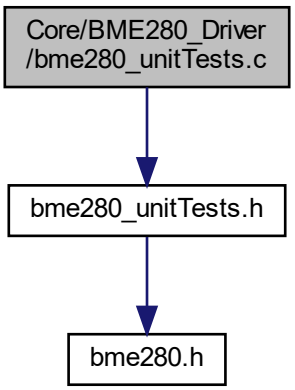
4.4.3.3 Calib2

`Calib2`

Definition at line 93 of file bme280_private_types.h.

4.5 Core/BME280_Driver/bme280_unitTests.c File Reference

`#include <bme280_unitTests.h>`
Include dependency graph for bme280_unitTests.c:



Functions

- BME280_Status BME280_UnitTest_PressureOversampling (BME280_Handle *a_cfgPtr)

  Tests setting all oversampling settings. Already assumes initialized handle.
- BME280_Status BME280_UnitTest_TemperatureOversampling (BME280_Handle *a_cfgPtr)

  Tests setting all oversampling settings. Already assumes initialized handle.
- BME280_Status BME280_UnitTest_HumdityOversampling (BME280_Handle *a_cfgPtr)

  Tests setting all oversampling settings. Already assumes initialized handle.
- BME280_Status BME280_UnitTest_FilterCoefficient (BME280_Handle *a_cfgPtr)

  Tests setting all filter coefficient settings. Already assumes initialized handle.
- BME280_Status BME280_UnitTest_StandbyTime (BME280_Handle *a_cfgPtr)

  Tests setting all standby time settings. Already assumes initialized handle.
- BME280_Status BME280_UnitTest_GeneralFuncs_SPI ()

  Tests oversampling functions and gets readings from sensor and checks them. Used for SPI mode.
- BME280_Status BME280_UnitTest_ReadingsCheck_SPI ()

  Sets specific options to the sensor and gets all readings. Used for SPI mode.
- BME280_Status BME280_UnitTest_GeneralFuncs_I2C ()

  Tests oversampling functions and gets readings from sensor and checks them. Used for I2C mode.
- BME280_Status BME280_UnitTest_ReadingsCheck_I2C ()

  Sets specific options to the sensor and gets all readings. Used for I2C mode.

4.5.1 Function Documentation

4.5.1.1 BME280_UnitTest_FilterCoefficient()

```
BME280_Status BME280_UnitTest_FilterCoefficient (
            BME280_Handle * a_cfgPtr )
```

Tests setting all filter coefficient settings. Already assumes initialized handle.

Parameters

a_cfgPtr	Pointer to sensor handle

Returns

Definition at line 183 of file bme280_unitTests.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.5.1.2 BME280_UnitTest_GeneralFuncs_I2C()

```
BME280_Status BME280_UnitTest_GeneralFuncs_I2C ( )
```

Tests oversampling functions and gets readings from sensor and checks them. Used for I2C mode.

Returns

Definition at line 470 of file bme280_unitTests.c.

4.5.1.3 BME280_UnitTest_GeneralFuncs_SPI()

```
BME280_Status BME280_UnitTest_GeneralFuncs_SPI ( )
```

Tests oversampling functions and gets readings from sensor and checks them. Used for SPI mode.

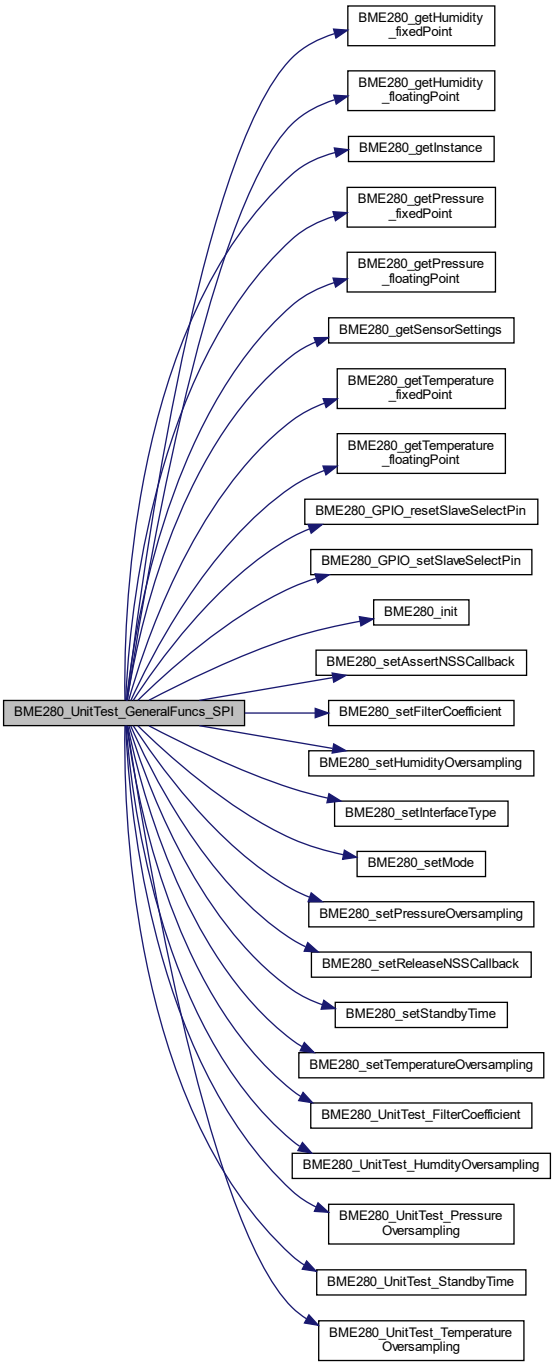
Complete unit testing functions for various settings of the sensor. Checks all settings and obtains readings.

Returns

Definition at line 295 of file bme280_unitTests.c.

References BME280_FILTER_COEFF_2, BME280_FOUND_EMPTY_INSTANCE, BME280_getHumidity_fixed↩ Point(), BME280_getHumidity_floatingPoint(), BME280_getInstance(), BME280_getPressure_fixedPoint(), BM↩ E280_getPressure_floatingPoint(), BME280_getSensorSettings(), BME280_getTemperature_fixedPoint(), BM↩ E280_getTemperature_floatingPoint(), BME280_GPIO_resetSlaveSelectPin(), BME280_GPIO_setSlaveSelect↩ Pin(), BME280_init(), BME280_Interface_SPI, BME280_Mode_Normal, BME280_OK, BME280_OVERSAMP↩ LING_x16, BME280_OVERSAMPLING_x4, BME280_OVERSAMPLING_x8, BME280_setAssertNSSCallback(), BME280_setFilterCoefficient(), BME280_setHumidityOversampling(), BME280_setInterfaceType(), BME280_set↩ Mode(), BME280_setPressureOversampling(), BME280_setReleaseNSSCallback(), BME280_setStandbyTime(), BME280_setTemperatureOversampling(), BME280_T_STDBY_125_MS, BME280_UnitTest_FilterCoefficient(), BME280_UnitTest_HumdityOversampling(), BME280_UnitTest_PressureOversampling(), BME280_UnitTest_↩ StandbyTime(), and BME280_UnitTest_TemperatureOversampling().

Here is the call graph for this function:



4.5.1.4 BME280_UnitTest_HumdityOversampling()

```
BME280_Status BME280_UnitTest_HumdityOversampling (
            BME280_Handle * a_cfgPtr )
```

Tests setting all oversampling settings. Already assumes initialized handle.

Parameters

a_cfgPtr	Pointer to sensor handle

Returns

Definition at line 131 of file bme280_unitTests.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.5.1.5 BME280_UnitTest_PressureOversampling()

```
BME280_Status BME280_UnitTest_PressureOversampling (
            BME280_Handle * a_cfgPtr )
```

Tests setting all oversampling settings. Already assumes initialized handle.

Unit testing functions for all options. Assumes initialized handle.

Parameters

a_cfgPtr	Pointer to sensor handle

Returns

Definition at line 20 of file bme280_unitTests.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.5.1.6 BME280_UnitTest_ReadingsCheck_I2C()

```
BME280_Status BME280_UnitTest_ReadingsCheck_I2C ( )
```

Sets specific options to the sensor and gets all readings. Used for I2C mode.

Returns

Definition at line 560 of file bme280_unitTests.c.

4.5.1.7 BME280_UnitTest_ReadingsCheck_SPI()

```
BME280_Status BME280_UnitTest_ReadingsCheck_SPI ( )
```

Sets specific options to the sensor and gets all readings. Used for SPI mode.

Returns

Definition at line 394 of file bme280_unitTests.c.

4.5.1.8 BME280_UnitTest_StandbyTime()

```
BME280_Status BME280_UnitTest_StandbyTime (
            BME280_Handle * a_cfgPtr )
```

Tests setting all standby time settings. Already assumes initialized handle.

Parameters

a_cfgPtr	Pointer to sensor handle

Returns

Definition at line 226 of file bme280_unitTests.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.5.1.9 BME280_UnitTest_TemperatureOversampling()

```
BME280_Status BME280_UnitTest_TemperatureOversampling (
            BME280_Handle * a_cfgPtr )
```

Tests setting all oversampling settings. Already assumes initialized handle.

Parameters

a_cfgPtr	Pointer to sensor handle

Returns

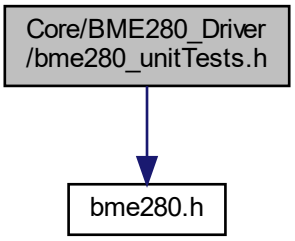
Definition at line 73 of file bme280_unitTests.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

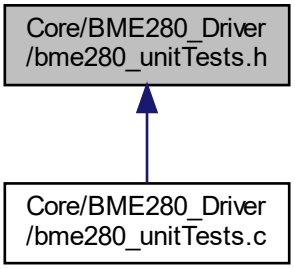
4.6 Core/BME280_Driver/bme280_unitTests.h File Reference

```
#include "bme280.h"
```

Include dependency graph for bme280_unitTests.h:



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



Functions

- BME280_Status BME280_UnitTest_PressureOversampling (BME280_Handle *a_cfgPtr)

  Tests setting all oversampling settings. Already assumes initialized handle.
- BME280_Status BME280_UnitTest_TemperatureOversampling (BME280_Handle *a_cfgPtr)

  Tests setting all oversampling settings. Already assumes initialized handle.
- BME280_Status BME280_UnitTest_HumdityOversampling (BME280_Handle *a_cfgPtr)

  Tests setting all oversampling settings. Already assumes initialized handle.
- BME280_Status BME280_UnitTest_FilterCoefficient (BME280_Handle *a_cfgPtr)

Tests setting all filter coefficient settings. Already assumes initialized handle.

- BME280_Status BME280_UnitTest_StandbyTime (BME280_Handle *a_cfgPtr)

  Tests setting all standby time settings. Already assumes initialized handle.

- BME280_Status BME280_UnitTest_GeneralFuncs_SPI ()

  Tests oversampling functions and gets readings from sensor and checks them. Used for SPI mode.

- BME280_Status BME280_UnitTest_ReadingsCheck_SPI ()

  Sets specific options to the sensor and gets all readings. Used for SPI mode.

- BME280_Status BME280_UnitTest_GeneralFuncs_I2C ()

  Tests oversampling functions and gets readings from sensor and checks them. Used for I2C mode.

- BME280_Status BME280_UnitTest_ReadingsCheck_I2C ()

  Sets specific options to the sensor and gets all readings. Used for I2C mode.

- void BME280_GPIO_setSlaveSelectPin (void)
- void BME280_GPIO_resetSlaveSelectPin (void)

4.6.1 Function Documentation

4.6.1.1 BME280_GPIO_resetSlaveSelectPin()

```
void BME280_GPIO_resetSlaveSelectPin (
            void  )
```

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.6.1.2 BME280_GPIO_setSlaveSelectPin()

```
void BME280_GPIO_setSlaveSelectPin (
            void  )
```

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.6.1.3 BME280_UnitTest_FilterCoefficient()

```
BME280_Status BME280_UnitTest_FilterCoefficient (
            BME280_Handle * a_cfgPtr )
```

Tests setting all filter coefficient settings. Already assumes initialized handle.

Parameters

a_cfgPtr	Pointer to sensor handle

Returns

Definition at line 183 of file bme280_unitTests.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.6.1.4 BME280_UnitTest_GeneralFuncs_I2C()

```
BME280_Status BME280_UnitTest_GeneralFuncs_I2C ( )
```

Tests oversampling functions and gets readings from sensor and checks them. Used for I2C mode.

Returns

Definition at line 470 of file bme280_unitTests.c.

4.6.1.5 BME280_UnitTest_GeneralFuncs_SPI()

```
BME280_Status BME280_UnitTest_GeneralFuncs_SPI ( )
```

Tests oversampling functions and gets readings from sensor and checks them. Used for SPI mode.

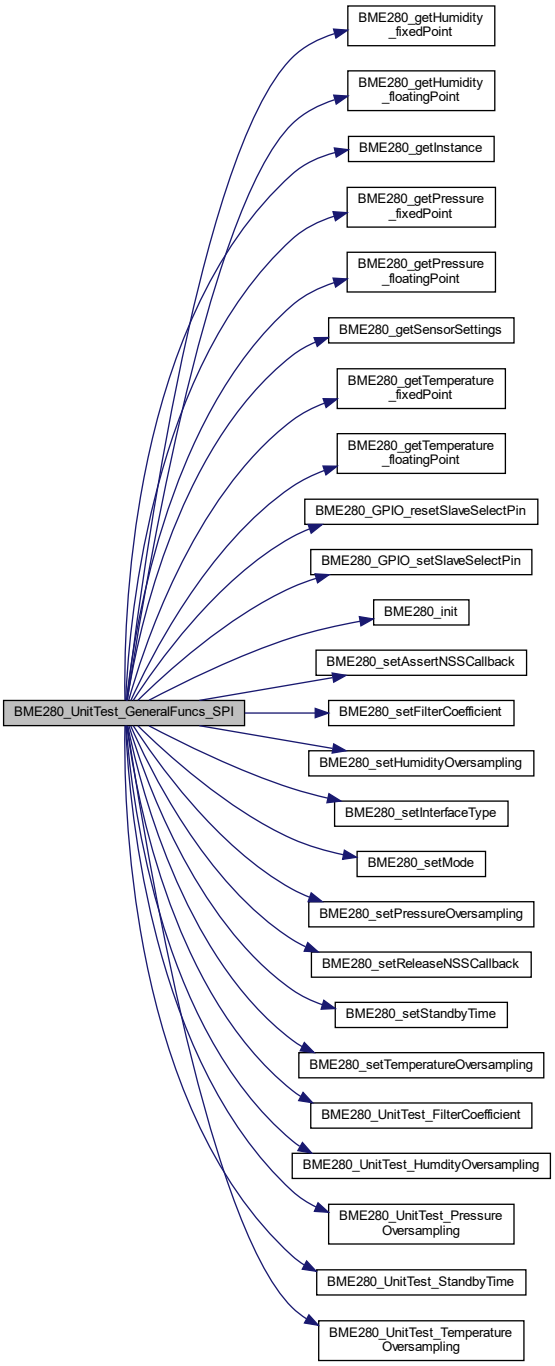
Complete unit testing functions for various settings of the sensor. Checks all settings and obtains readings.

Returns

Definition at line 295 of file bme280_unitTests.c.

References BME280_FILTER_COEFF_2, BME280_FOUND_EMPTY_INSTANCE, BME280_getHumidity_fixed↩Point(), BME280_getHumidity_floatingPoint(), BME280_getInstance(), BME280_getPressure_fixedPoint(), BM↩E280_getPressure_floatingPoint(), BME280_getSensorSettings(), BME280_getTemperature_fixedPoint(), BM↩E280_getTemperature_floatingPoint(), BME280_GPIO_resetSlaveSelectPin(), BME280_GPIO_setSlaveSelect↩Pin(), BME280_init(), BME280_Interface_SPI, BME280_Mode_Normal, BME280_OK, BME280_OVERSAMP↩LING_x16, BME280_OVERSAMPLING_x4, BME280_OVERSAMPLING_x8, BME280_setAssertNSSCallback(), BME280_setFilterCoefficient(), BME280_setHumidityOversampling(), BME280_setInterfaceType(), BME280_set↩Mode(), BME280_setPressureOversampling(), BME280_setReleaseNSSCallback(), BME280_setStandbyTime(), BME280_setTemperatureOversampling(), BME280_T_STDBY_125_MS, BME280_UnitTest_FilterCoefficient(), BME280_UnitTest_HumdityOversampling(), BME280_UnitTest_PressureOversampling(), BME280_UnitTest_↩StandbyTime(), and BME280_UnitTest_TemperatureOversampling().

Here is the call graph for this function:



4.6.1.6 BME280_UnitTest_HumdityOversampling()

```
BME280_Status BME280_UnitTest_HumdityOversampling (
            BME280_Handle * a_cfgPtr )
```

Tests setting all oversampling settings. Already assumes initialized handle.

Parameters

a_cfgPtr	Pointer to sensor handle

Returns

Definition at line 131 of file bme280_unitTests.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.6.1.7 BME280_UnitTest_PressureOversampling()

```
BME280_Status BME280_UnitTest_PressureOversampling (
            BME280_Handle * a_cfgPtr )
```

Tests setting all oversampling settings. Already assumes initialized handle.

Unit testing functions for all options. Assumes initialized handle.

Parameters

a_cfgPtr	Pointer to sensor handle

Returns

Definition at line 20 of file bme280_unitTests.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.6.1.8 BME280_UnitTest_ReadingsCheck_I2C()

```
BME280_Status BME280_UnitTest_ReadingsCheck_I2C ( )
```

Sets specific options to the sensor and gets all readings. Used for I2C mode.

Returns

Definition at line 560 of file bme280_unitTests.c.

4.6.1.9 BME280_UnitTest_ReadingsCheck_SPI()

```
BME280_Status BME280_UnitTest_ReadingsCheck_SPI ( )
```

Sets specific options to the sensor and gets all readings. Used for SPI mode.

Returns

Definition at line 394 of file bme280_unitTests.c.

4.6.1.10 BME280_UnitTest_StandbyTime()

```
BME280_Status BME280_UnitTest_StandbyTime (
            BME280_Handle * a_cfgPtr )
```

Tests setting all standby time settings. Already assumes initialized handle.

Parameters

a_cfgPtr	Pointer to sensor handle

Returns

Definition at line 226 of file bme280_unitTests.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.6.1.11 BME280_UnitTest_TemperatureOversampling()

```
BME280_Status BME280_UnitTest_TemperatureOversampling (
            BME280_Handle * a_cfgPtr )
```

Tests setting all oversampling settings. Already assumes initialized handle.

Parameters

a_cfgPtr	Pointer to sensor handle

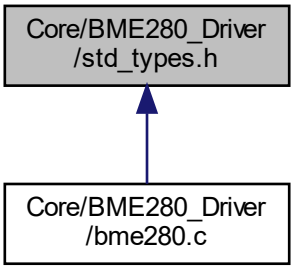
Returns

Definition at line 73 of file bme280_unitTests.c.

Referenced by BME280_UnitTest_GeneralFuncs_SPI().

4.7 Core/BME280_Driver/std_types.h File Reference

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



Macros

- #define TRUE (1U)
- #define True (1U)
- #define true (1U)
- #define FALSE (0U)
- #define False (0U)
- #define false (0U)
- #define LOGIC_HIGH (1U)
- #define LOGIC_LOW (0U)
- #define NULL_PTR ((void∗)0)

Typedefs

- typedef unsigned char uint8
- typedef unsigned short uint16
- typedef unsigned long uint32
- typedef unsigned long long uint64
- typedef unsigned char boolean
- typedef signed char sint8
- typedef signed short sint16
- typedef signed long sint32
- typedef signed long long sint64
- typedef float float32
- typedef double float64

4.7.1 Macro Definition Documentation

4.7.1.1 FALSE

```
#define FALSE (0U)
```

Definition at line 22 of file std_types.h.

4.7.1.2 False

```
#define False (0U)
```

Definition at line 26 of file std_types.h.

4.7.1.3 false

```
#define false (0U)
```

Definition at line 30 of file std_types.h.

4.7.1.4 LOGIC_HIGH

```
#define LOGIC_HIGH (1U)
```

Definition at line 34 of file std_types.h.

4.7.1.5 LOGIC_LOW

```
#define LOGIC_LOW (0U)
```

Definition at line 38 of file std_types.h.

4.7.1.6 NULL_PTR

```
#define NULL_PTR ((void*)0)
```

Definition at line 42 of file std_types.h.

4.7.1.7 TRUE

```
#define TRUE (1U)
```

Definition at line 12 of file std_types.h.

4.7.1.8 True

```
#define True (1U)
```

Definition at line 15 of file std_types.h.

4.7.1.9 true

```
#define true (1U)
```

Definition at line 18 of file std_types.h.

4.7.2 Typedef Documentation

4.7.2.1 boolean

```
typedef unsigned char boolean
```

Definition at line 48 of file std_types.h.

4.7.2.2 float32

```
typedef float float32
```

Definition at line 55 of file std_types.h.

4.7.2.3 float64

```
typedef double float64
```

Definition at line 56 of file std_types.h.

4.7.2.4 sint16

```
typedef signed short sint16
```

Definition at line 51 of file std_types.h.

4.7.2.5 sint32

```
typedef signed long sint32
```

Definition at line 52 of file std_types.h.

4.7.2.6 sint64

```
typedef signed long long sint64
```

Definition at line 53 of file std_types.h.

4.7.2.7 sint8

```
typedef signed char sint8
```

Definition at line 50 of file std_types.h.

4.7.2.8 uint16

```
typedef unsigned short uint16
```

Definition at line 45 of file std_types.h.

4.7.2.9 uint32

```
typedef unsigned long uint32
```

Definition at line 46 of file std_types.h.

4.7.2.10 uint64

```
typedef unsigned long long uint64
```

Definition at line 47 of file std_types.h.

4.7.2.11 uint8

```
typedef unsigned char uint8
```

Definition at line 44 of file std_types.h.

Index

__attribute__
    bme280.c, 22
    bme280_private_types.h, 69
__pad0__
    BME280_ConfigRegisterUnion, 5
    BME280_CtrlHumRegisterUnion, 9
    BME280_PressureReading, 13
    BME280_StatusRegisterUnion, 17
    BME280_TemperatureReading, 19
__pad1__
    BME280_StatusRegisterUnion, 17

Bits
    BME280_ConfigRegisterUnion, 5
    BME280_CtrlHumRegisterUnion, 10
    BME280_CtrlMeasRegisterUnion, 11
    BME280_StatusRegisterUnion, 17
bme280.c
    __attribute__, 22
    BME280_calculateMeasurementDelayMs, 23
    BME280_DeInit, 23
    BME280_getChipID, 24
    BME280_getFilterCoefficient, 24
    BME280_getHumidity_fixedPoint, 24
    BME280_getHumidity_floatingPoint, 25
    BME280_getHumidityOversampling, 25
    BME280_getInstance, 26
    BME280_getMeasuringStatus, 26
    BME280_getMode, 26
    BME280_getPressure_fixedPoint, 27
    BME280_getPressure_floatingPoint, 27
    BME280_getPressureOversampling, 28
    BME280_getSensorSettings, 28
    BME280_getStandbyTime, 28
    BME280_getTemperature_fixedPoint, 29
    BME280_getTemperature_floatingPoint, 29
    BME280_getTemperatureOversampling, 30
    BME280_getUpdateStatus, 30
    BME280_init, 30
    BME280_setAssertNSSCallback, 31
    BME280_setFilterCoefficient, 31
    BME280_setHumidityOversampling, 32
    BME280_setInterfaceType, 32
    BME280_setMode, 33
    BME280_setPressureOversampling, 33
    BME280_setReleaseNSSCallback, 33
    BME280_setStandbyTime, 34
    BME280_setTemperatureOversampling, 34
    BME280_softReset, 35
bme280.h
    BME280_boolean, 38
    BME280_calculateMeasurementDelayMs, 44
    BME280_CALLBACK_NOT_SET, 43
    BME280_COMM_ERROR, 43
    BME280_Comm_Error, 40
    BME280_Comm_OK, 40
    BME280_Comm_Status, 40
    BME280_DeInit, 44
    BME280_delayMs, 45
    BME280_FILTER_COEFF_16, 41
    BME280_FILTER_COEFF_2, 40
    BME280_FILTER_COEFF_4, 40
    BME280_FILTER_COEFF_8, 41
    BME280_FILTER_COEFF_OFF, 40
    BME280_FILTER_NOT_SPECIFIED, 41
    BME280_FilterCoeff, 40
    BME280_float64, 38
    BME280_FOUND_EMPTY_INSTANCE, 43
    BME280_getChipID, 45
    BME280_getFilterCoefficient, 45
    BME280_getHumidity_fixedPoint, 45
    BME280_getHumidity_floatingPoint, 46
    BME280_getHumidityOversampling, 46
    BME280_getInstance, 47
    BME280_getMeasuringStatus, 47
    BME280_getMode, 48
    BME280_getPressure_fixedPoint, 48
    BME280_getPressure_floatingPoint, 48
    BME280_getPressureOversampling, 49
    BME280_getSensorSettings, 49
    BME280_getStandbyTime, 50
    BME280_getTemperature_fixedPoint, 50
    BME280_getTemperature_floatingPoint, 50
    BME280_getTemperatureOversampling, 51
    BME280_getUpdateStatus, 51
    BME280_Handle, 39
    BME280_I2C_Master_Receive, 52
    BME280_I2C_Master_Transmit, 52
    BME280_init, 52
    BME280_Interface_I2C, 41
    BME280_Interface_Not_Specified, 41
    BME280_Interface_SPI, 41
    BME280_InterfaceType, 41
    BME280_IS_INSTANCE, 43
    BME280_Measuring_Finished, 41
    BME280_Measuring_Running, 41
    BME280_MeasuringStatus, 41
    BME280_Mode_Forced, 42
    BME280_Mode_Normal, 42

BME280_Mode_Not_Specified, 42
BME280_Mode_Sleep, 42
BME280_ModeType, 41
BME280_NO_INTERFACE_SPECIFIED, 43
BME280_NOT_IMPLEMENTED, 43
BME280_NOT_INSTANCE, 43
BME280_NOT_YET_OBTAINED, 43
BME280_NULL_ERROR, 43
BME280_OK, 43
BME280_OVERSAMPLING_NOT_SPECIFIED, 42
BME280_OVERSAMPLING_OFF, 42
BME280_Oversampling_setting, 42
BME280_OVERSAMPLING_x1, 42
BME280_OVERSAMPLING_x16, 42
BME280_OVERSAMPLING_x2, 42
BME280_OVERSAMPLING_x4, 42
BME280_OVERSAMPLING_x8, 42
BME280_POOL_FULL, 43
BME280_setAssertNSSCallback, 53
BME280_setFilterCoefficient, 53
BME280_setHumidityOversampling, 54
BME280_setInterfaceType, 54
BME280_setMode, 55
BME280_setPressureOversampling, 55
BME280_setReleaseNSSCallback, 55
BME280_setStandbyTime, 56
BME280_setTemperatureOversampling, 56
BME280_SETTING_FAILED, 43
BME280_sint16, 39
BME280_sint32, 39
BME280_sint8, 39
BME280_softReset, 57
BME280_SPI_TransmitReceive, 57
BME280_StandbyTime, 42
BME280_Status, 43
BME280_T_STDBY_0_5_MS, 42
BME280_T_STDBY_1000_MS, 42
BME280_T_STDBY_10_MS, 42
BME280_T_STDBY_125_MS, 42
BME280_T_STDBY_20_MS, 42
BME280_T_STDBY_250_MS, 42
BME280_T_STDBY_500_MS, 42
BME280_T_STDBY_62_5_MS, 42
BME280_T_STDBY_NOT_SPECIFIED, 42
BME280_uint16, 39
BME280_uint32, 39
BME280_uint8, 40
BME280_Update_Copying, 43
BME280_Update_Finished, 43
BME280_UpdateStatus, 43
BME280_boolean
bme280.h, 38
BME280_calculateMeasurementDelayMs
bme280.c, 23
bme280.h, 44
BME280_CALIB_1_BLOCK_SIZE
bme280_private_defs.h, 59
BME280_CALIB_2_BLOCK_SIZE
bme280_private_defs.h, 59
BME280_CALLBACK_NOT_SET
bme280.h, 43
BME280_CHIP_ID
bme280_private_defs.h, 59
BME280_COMM_ERROR
bme280.h, 43
BME280_Comm_Error
bme280.h, 40
BME280_Comm_OK
bme280.h, 40
BME280_Comm_Status
bme280.h, 40
BME280_CONFIG_IM_UPDATE_MASK
bme280_private_defs.h, 60
BME280_CONFIG_MEAS_MASK
bme280_private_defs.h, 60
BME280_CONFIG_REGISTER
bme280_private_defs.h, 60
BME280_ConfigRegisterUnion, 5
__pad0__, 5
Bits, 5
config, 6
filter_coeff, 6
spi3w_en, 6
t_sb, 6
BME280_ConfigType, 7
calib_data_1, 7
calib_data_2, 7
GPIOCallback_resetNSS, 8
GPIOCallback_SetNSS, 8
I2C_SlaveAddr, 8
ID, 8
Intf, 8
occupied, 9
BME280_CTRL_HUM_REGISTER
bme280_private_defs.h, 60
BME280_CTRL_MEAS_REGISTER
bme280_private_defs.h, 60
BME280_CtrlHumRegisterUnion, 9
__pad0__, 9
Bits, 10
config, 10
osrs_h, 10
BME280_CtrlMeasRegisterUnion, 10
Bits, 11
config, 11
mode, 11
osrs_p, 11
osrs_t, 11
BME280_DeInit
bme280.c, 23
bme280.h, 44
BME280_delayMs
bme280.h, 45
BME280_FILTER_COEFF_16
bme280.h, 41
BME280_FILTER_COEFF_2

bme280.h, 40
BME280_FILTER_COEFF_4
bme280.h, 40
BME280_FILTER_COEFF_8
bme280.h, 41
BME280_FILTER_COEFF_OFF
bme280.h, 40
BME280_FILTER_NOT_SPECIFIED
bme280.h, 41
BME280_FilterCoeff
bme280.h, 40
BME280_float64
bme280.h, 38
BME280_FOUND_EMPTY_INSTANCE
bme280.h, 43
BME280_getChipID
bme280.c, 24
bme280.h, 45
BME280_getFilterCoefficient
bme280.c, 24
bme280.h, 45
BME280_getHumidity_fixedPoint
bme280.c, 24
bme280.h, 45
BME280_getHumidity_floatingPoint
bme280.c, 25
bme280.h, 46
BME280_getHumidityOversampling
bme280.c, 25
bme280.h, 46
BME280_getInstance
bme280.c, 26
bme280.h, 47
BME280_getMeasuringStatus
bme280.c, 26
bme280.h, 47
BME280_getMode
bme280.c, 26
bme280.h, 48
BME280_getPressure_fixedPoint
bme280.c, 27
bme280.h, 48
BME280_getPressure_floatingPoint
bme280.c, 27
bme280.h, 48
BME280_getPressureOversampling
bme280.c, 28
bme280.h, 49
BME280_getSensorSettings
bme280.c, 28
bme280.h, 49
BME280_getStandbyTime
bme280.c, 28
bme280.h, 50
BME280_getTemperature_fixedPoint
bme280.c, 29
bme280.h, 50
BME280_getTemperature_floatingPoint
bme280.c, 29
bme280.h, 50
BME280_getTemperatureOversampling
bme280.c, 30
bme280.h, 51
BME280_getUpdateStatus
bme280.c, 30
bme280.h, 51
BME280_GPIO_resetSlaveSelectPin
bme280_unitTests.h, 77
BME280_GPIO_setSlaveSelectPin
bme280_unitTests.h, 77
BME280_Handle
bme280.h, 39
BME280_HUM_BLOCK_START_ADDRESS
bme280_private_defs.h, 60
BME280_HUM_CALIB_BLOCK_SIZE
bme280_private_defs.h, 61
BME280_HUM_REGISTER_LSB
bme280_private_defs.h, 61
BME280_HUM_REGISTER_MSB
bme280_private_defs.h, 61
BME280_HumidityReading, 12
Data, 12
humidity, 12
lsb, 12
msb, 12
BME280_I2C_Addr
bme280_private_types.h, 69
BME280_I2C_Master_Receive
bme280.h, 52
BME280_I2C_Master_Transmit
bme280.h, 52
BME280_I2C_READ_MASK
bme280_private_defs.h, 61
BME280_I2C_SensorSlaveAddress
bme280_private_types.h, 69
BME280_I2C_TIMEOUT_MS
bme280_private_defs.h, 61
BME280_I2C_WRITE_MASK
bme280_private_defs.h, 61
BME280_ID_REGISTER
bme280_private_defs.h, 62
BME280_IM_UPDATE_READY
bme280_private_defs.h, 62
BME280_init
bme280.c, 30
bme280.h, 52
BME280_Interface_I2C
bme280.h, 41
BME280_Interface_Not_Specified
bme280.h, 41
BME280_Interface_SPI
bme280.h, 41
BME280_InterfaceType
bme280.h, 41
BME280_IS_INSTANCE
bme280.h, 43

BME280_LSBYTE_MASK
bme280_private_defs.h, 62
BME280_MAX_DISCOVERY_COUNT
bme280_private_defs.h, 62
BME280_MAX_HUMIDITY_FIXED_POINT
bme280_private_defs.h, 62
BME280_MAX_HUMIDITY_FLOATING_POINT
bme280_private_defs.h, 62
BME280_MAX_PRESSURE_FIXED_POINT
bme280_private_defs.h, 63
BME280_MAX_PRESSURE_FLOATING_POINT
bme280_private_defs.h, 63
BME280_MAX_SENSOR_POOL_SIZE
bme280_private_defs.h, 63
BME280_MAX_TEMPERATURE_FIXED_POINT
bme280_private_defs.h, 63
BME280_MAX_TEMPERATURE_FLOATING_POINT
bme280_private_defs.h, 63
BME280_MEASURING_DONE
bme280_private_defs.h, 63
BME280_Measuring_Finished
bme280.h, 41
BME280_Measuring_Running
bme280.h, 41
BME280_MeasuringStatus
bme280.h, 41
BME280_MIN_HUMIDITY_FIXED_POINT
bme280_private_defs.h, 64
BME280_MIN_HUMIDITY_FLOATING_POINT
bme280_private_defs.h, 64
BME280_MIN_PRESSURE_FIXED_POINT
bme280_private_defs.h, 64
BME280_MIN_PRESSURE_FLOATING_POINT
bme280_private_defs.h, 64
BME280_MIN_TEMPERATURE_FIXED_POINT
bme280_private_defs.h, 64
BME280_MIN_TEMPERATURE_FLOATING_POINT
bme280_private_defs.h, 64
BME280_Mode_Forced
bme280.h, 42
BME280_Mode_Normal
bme280.h, 42
BME280_Mode_Not_Specified
bme280.h, 42
BME280_Mode_Sleep
bme280.h, 42
BME280_ModeType
bme280.h, 41
BME280_NO_INTERFACE_SPECIFIED
bme280.h, 43
BME280_NOT_IMPLEMENTED
bme280.h, 43
BME280_NOT_INSTANCE
bme280.h, 43
BME280_NOT_YET_OBTAINED
bme280.h, 43
BME280_NULL_ERROR
bme280.h, 43
BME280_OK
bme280.h, 43
BME280_OVERSAMPLING_NOT_SPECIFIED
bme280.h, 42
BME280_OVERSAMPLING_OFF
bme280.h, 42
BME280_Oversampling_setting
bme280.h, 42
BME280_OVERSAMPLING_x1
bme280.h, 42
BME280_OVERSAMPLING_x16
bme280.h, 42
BME280_OVERSAMPLING_x2
bme280.h, 42
BME280_OVERSAMPLING_x4
bme280.h, 42
BME280_OVERSAMPLING_x8
bme280.h, 42
BME280_POOL_FULL
bme280.h, 43
BME280_PRESS_REGISTER_LSB
bme280_private_defs.h, 65
BME280_PRESS_REGISTER_MSB
bme280_private_defs.h, 65
BME280_PRESS_REGISTER_XLSB
bme280_private_defs.h, 65
BME280_PressureReading, 13
__pad0__, 13
Data, 13
lsb, 14
msb, 14
pressure, 14
xlsb, 14
bme280_private_defs.h
BME280_CALIB_1_BLOCK_SIZE, 59
BME280_CALIB_2_BLOCK_SIZE, 59
BME280_CHIP_ID, 59
BME280_CONFIG_IM_UPDATE_MASK, 60
BME280_CONFIG_MEAS_MASK, 60
BME280_CONFIG_REGISTER, 60
BME280_CTRL_HUM_REGISTER, 60
BME280_CTRL_MEAS_REGISTER, 60
BME280_HUM_BLOCK_START_ADDRESS, 60
BME280_HUM_CALIB_BLOCK_SIZE, 61
BME280_HUM_REGISTER_LSB, 61
BME280_HUM_REGISTER_MSB, 61
BME280_I2C_READ_MASK, 61
BME280_I2C_TIMEOUT_MS, 61
BME280_I2C_WRITE_MASK, 61
BME280_ID_REGISTER, 62
BME280_IM_UPDATE_READY, 62
BME280_LSBYTE_MASK, 62
BME280_MAX_DISCOVERY_COUNT, 62
BME280_MAX_HUMIDITY_FIXED_POINT, 62
BME280_MAX_HUMIDITY_FLOATING_POINT, 62
BME280_MAX_PRESSURE_FIXED_POINT, 63

BME280_MAX_PRESSURE_FLOATING_POINT, 63
BME280_MAX_SENSOR_POOL_SIZE, 63
BME280_MAX_TEMPERATURE_FIXED_POINT, 63
BME280_MAX_TEMPERATURE_FLOATING_POINT, 63
BME280_MEASURING_DONE, 63
BME280_MIN_HUMIDITY_FIXED_POINT, 64
BME280_MIN_HUMIDITY_FLOATING_POINT, 64
BME280_MIN_PRESSURE_FIXED_POINT, 64
BME280_MIN_PRESSURE_FLOATING_POINT, 64
BME280_MIN_TEMPERATURE_FIXED_POINT, 64
BME280_MIN_TEMPERATURE_FLOATING_POINT, 64
BME280_PRESS_REGISTER_LSB, 65
BME280_PRESS_REGISTER_MSB, 65
BME280_PRESS_REGISTER_XLSB, 65
BME280_READINGS_BYTES_LENGTH, 65
BME280_RESET_REGISTER, 65
BME280_RESET_WORD, 65
BME280_SPI_READ_MASK, 66
BME280_SPI_TIMEOUT_MS, 66
BME280_SPI_WRITE_MASK, 66
BME280_START_READINGS_ADDRESS, 66
BME280_START_UP_TIME_MS, 66
BME280_STATUS_REGISTER, 66
BME280_TEMP_PRESS_BLOCK_START_ADDRESS, 67
BME280_TEMP_PRESS_CALIB_BLOCK_SIZE, 67
BME280_TEMP_REGISTER_LSB, 67
BME280_TEMP_REGISTER_MSB, 67
BME280_TEMP_REGISTER_XLSB, 67
bme280_private_types.h
__attribute__, 69
BME280_I2C_Addr, 69
BME280_I2C_SensorSlaveAddress, 69
BME280_UncompensatedReadings, 70
Calib1, 70
Calib2, 70
BME280_READINGS_BYTES_LENGTH
bme280_private_defs.h, 65
BME280_RESET_REGISTER
bme280_private_defs.h, 65
BME280_RESET_WORD
bme280_private_defs.h, 65
BME280_setAssertNSSCallback
bme280.c, 31
bme280.h, 53
BME280_setFilterCoefficient
bme280.c, 31
bme280.h, 53
BME280_setHumidityOversampling
bme280.c, 32
bme280.h, 54
BME280_setInterfaceType
bme280.c, 32
bme280.h, 54
BME280_setMode
bme280.c, 33
bme280.h, 55
BME280_setPressureOversampling
bme280.c, 33
bme280.h, 55
BME280_setReleaseNSSCallback
bme280.c, 33
bme280.h, 55
BME280_setStandbyTime
bme280.c, 34
bme280.h, 56
BME280_setTemperatureOversampling
bme280.c, 34
bme280.h, 56
BME280_SETTING_FAILED
bme280.h, 43
BME280_Settings, 14
filter, 15
Mode, 15
osrs_h, 15
osrs_p, 16
osrs_t, 16
t_stby, 16
BME280_sint16
bme280.h, 39
BME280_sint32
bme280.h, 39
BME280_sint8
bme280.h, 39
BME280_softReset
bme280.c, 35
bme280.h, 57
BME280_SPI_READ_MASK
bme280_private_defs.h, 66
BME280_SPI_TIMEOUT_MS
bme280_private_defs.h, 66
BME280_SPI_TransmitReceive
bme280.h, 57
BME280_SPI_WRITE_MASK
bme280_private_defs.h, 66
BME280_StandbyTime
bme280.h, 42
BME280_START_READINGS_ADDRESS
bme280_private_defs.h, 66
BME280_START_UP_TIME_MS
bme280_private_defs.h, 66
BME280_Status
bme280.h, 43
BME280_STATUS_REGISTER
bme280_private_defs.h, 66
BME280_StatusRegisterUnion, 16
__pad0__, 17
__pad1__, 17
Bits, 17

config, 17
im_update, 18
measuring, 18
BME280_T_STDBY_0_5_MS
bme280.h, 42
BME280_T_STDBY_1000_MS
bme280.h, 42
BME280_T_STDBY_10_MS
bme280.h, 42
BME280_T_STDBY_125_MS
bme280.h, 42
BME280_T_STDBY_20_MS
bme280.h, 42
BME280_T_STDBY_250_MS
bme280.h, 42
BME280_T_STDBY_500_MS
bme280.h, 42
BME280_T_STDBY_62_5_MS
bme280.h, 42
BME280_T_STDBY_NOT_SPECIFIED
bme280.h, 42
BME280_TEMP_PRESS_BLOCK_START_ADDRESS
bme280_private_defs.h, 67
BME280_TEMP_PRESS_CALIB_BLOCK_SIZE
bme280_private_defs.h, 67
BME280_TEMP_REGISTER_LSB
bme280_private_defs.h, 67
BME280_TEMP_REGISTER_MSB
bme280_private_defs.h, 67
BME280_TEMP_REGISTER_XLSB
bme280_private_defs.h, 67
BME280_TemperatureReading, 18
__pad0__, 19
Data, 19
lsb, 19
msb, 19
temperature, 19
xlsb, 19
BME280_uint16
bme280.h, 39
BME280_uint32
bme280.h, 39
BME280_uint8
bme280.h, 40
BME280_UncompensatedReadings, 20
bme280_private_types.h, 70
BME280_UnitTest_FilterCoefficient
bme280_unitTests.c, 71
bme280_unitTests.h, 77
BME280_UnitTest_GeneralFuncs_I2C
bme280_unitTests.c, 71
bme280_unitTests.h, 78
BME280_UnitTest_GeneralFuncs_SPI
bme280_unitTests.c, 72
bme280_unitTests.h, 78
BME280_UnitTest_HumdityOversampling
bme280_unitTests.c, 73
bme280_unitTests.h, 79
BME280_UnitTest_PressureOversampling
bme280_unitTests.c, 74
bme280_unitTests.h, 80
BME280_UnitTest_ReadingsCheck_I2C
bme280_unitTests.c, 74
bme280_unitTests.h, 80
BME280_UnitTest_ReadingsCheck_SPI
bme280_unitTests.c, 74
bme280_unitTests.h, 80
BME280_UnitTest_StandbyTime
bme280_unitTests.c, 75
bme280_unitTests.h, 81
BME280_UnitTest_TemperatureOversampling
bme280_unitTests.c, 75
bme280_unitTests.h, 81
bme280_unitTests.c
BME280_UnitTest_FilterCoefficient, 71
BME280_UnitTest_GeneralFuncs_I2C, 71
BME280_UnitTest_GeneralFuncs_SPI, 72
BME280_UnitTest_HumdityOversampling, 73
BME280_UnitTest_PressureOversampling, 74
BME280_UnitTest_ReadingsCheck_I2C, 74
BME280_UnitTest_ReadingsCheck_SPI, 74
BME280_UnitTest_StandbyTime, 75
BME280_UnitTest_TemperatureOversampling, 75
bme280_unitTests.h
BME280_GPIO_resetSlaveSelectPin, 77
BME280_GPIO_setSlaveSelectPin, 77
BME280_UnitTest_FilterCoefficient, 77
BME280_UnitTest_GeneralFuncs_I2C, 78
BME280_UnitTest_GeneralFuncs_SPI, 78
BME280_UnitTest_HumdityOversampling, 79
BME280_UnitTest_PressureOversampling, 80
BME280_UnitTest_ReadingsCheck_I2C, 80
BME280_UnitTest_ReadingsCheck_SPI, 80
BME280_UnitTest_StandbyTime, 81
BME280_UnitTest_TemperatureOversampling, 81
BME280_Update_Copying
bme280.h, 43
BME280_Update_Finished
bme280.h, 43
BME280_UpdateStatus
bme280.h, 43
boolean
std_types.h, 84

Calib1, 20
bme280_private_types.h, 70
Calib2, 20
bme280_private_types.h, 70
calib_data_1
BME280_ConfigType, 7
calib_data_2
BME280_ConfigType, 7
config
BME280_ConfigRegisterUnion, 6
BME280_CtrlHumRegisterUnion, 10
BME280_CtrlMeasRegisterUnion, 11
BME280_StatusRegisterUnion, 17

Core/BME280_Driver/bme280.c, 21
Core/BME280_Driver/bme280.h, 36
Core/BME280_Driver/bme280_private_defs.h, 58
Core/BME280_Driver/bme280_private_types.h, 68
Core/BME280_Driver/bme280_unitTests.c, 70
Core/BME280_Driver/bme280_unitTests.h, 76
Core/BME280_Driver/std_types.h, 82

Data
    BME280_HumidityReading, 12
    BME280_PressureReading, 13
    BME280_TemperatureReading, 19

FALSE
    std_types.h, 83
False
    std_types.h, 83
false
    std_types.h, 83
filter
    BME280_Settings, 15
filter_coeff
    BME280_ConfigRegisterUnion, 6
float32
    std_types.h, 84
float64
    std_types.h, 84

GPIOCallback_resetNSS
    BME280_ConfigType, 8
GPIOCallback_SetNSS
    BME280_ConfigType, 8

humidity
    BME280_HumidityReading, 12

I2C_SlaveAddr
    BME280_ConfigType, 8
ID
    BME280_ConfigType, 8
im_update
    BME280_StatusRegisterUnion, 18
Intf
    BME280_ConfigType, 8

LOGIC_HIGH
    std_types.h, 83
LOGIC_LOW
    std_types.h, 83
lsb
    BME280_HumidityReading, 12
    BME280_PressureReading, 14
    BME280_TemperatureReading, 19

measuring
    BME280_StatusRegisterUnion, 18
Mode
    BME280_Settings, 15
mode
    BME280_CtrlMeasRegisterUnion, 11
msb
    BME280_HumidityReading, 12
    BME280_PressureReading, 14
    BME280_TemperatureReading, 19

NULL_PTR
    std_types.h, 83

occupied
    BME280_ConfigType, 9
osrs_h
    BME280_CtrlHumRegisterUnion, 10
    BME280_Settings, 15
osrs_p
    BME280_CtrlMeasRegisterUnion, 11
    BME280_Settings, 16
osrs_t
    BME280_CtrlMeasRegisterUnion, 11
    BME280_Settings, 16

pressure
    BME280_PressureReading, 14

sint16
    std_types.h, 85
sint32
    std_types.h, 85
sint64
    std_types.h, 85
sint8
    std_types.h, 85
spi3w_en
    BME280_ConfigRegisterUnion, 6
std_types.h
    boolean, 84
    FALSE, 83
    False, 83
    false, 83
    float32, 84
    float64, 84
    LOGIC_HIGH, 83
    LOGIC_LOW, 83
    NULL_PTR, 83
    sint16, 85
    sint32, 85
    sint64, 85
    sint8, 85
    TRUE, 84
    True, 84
    true, 84
    uint16, 85
    uint32, 85
    uint64, 86
    uint8, 86

t_sb
    BME280_ConfigRegisterUnion, 6
t_stby
    BME280_Settings, 16

temperature
BME280_TemperatureReading, 19
TRUE
std_types.h, 84
True
std_types.h, 84
true
std_types.h, 84

uint16
std_types.h, 85
uint32
std_types.h, 85
uint64
std_types.h, 86
uint8
std_types.h, 86

xlsb
BME280_PressureReading, 14
BME280_TemperatureReading, 19