
Chipsmall
深圳市津利帝科技有限公司
Geelead Technology Limited
Chipsmall
采购部

BME280

Combined humidity and pressure sensor

BME280 – Data sheet

Document revision	1.5
Document release date	May 2018
Document number	BST-BME280-DS002-13
Technical reference code	0 273 141 185
Notes	Data and descriptions in this document are subject to change without notice. Product photos and pictures are for illustration purposes only and may differ from the real product appearance

BME280
Digital Humidity, Pressure and Temperature Sensor

Key features

Package	2.5 mm x 2.5 mm x 0.93 mm metal lid LGA
Digital interface	I²C (up to 3.4 MHz) and SPI (3 and 4 wire, up to 10 MHz)
Supply voltage	V_{DD} main supply voltage range: 1.71 V to 3.6 V V_{DDIO} interface voltage range: 1.2 V to 3.6 V
Current consumption	1.8 µA @ 1 Hz humidity and temperature 2.8 µA @ 1 Hz pressure and temperature 3.6 µA @ 1 Hz humidity, pressure and temperature 0.1 µA in sleep mode
Operating range	-40…+85 °C, 0…100 % rel. humidity, 300…1100 hPa

- Humidity sensor and pressure sensor can be independently enabled / disabled
- Register and performance compatible to Bosch Sensortec BMP280 digital pressure sensor
- RoHS compliant, halogen-free, MSL1

Key parameters for humidity sensor

Response time $(\tau_{63\%})$	1 s
Accuracy tolerance	±3 % relative humidity
Hysteresis	±1% relative humidity

Key parameters for pressure sensor

RMS Noise	0.2 Pa, equiv. to 1.7 cm
Offset temperature coefficient	±1.5 Pa/K, equiv. to ±12.6 cm at 1 °C temperature change

Typical application

- Context awareness, e.g. skin detection, room change detection
- Fitness monitoring / well-being
  - Warning regarding dryness or high temperatures
  - Measurement of volume and air flow
- Home automation control
  - control heating, venting, air conditioning (HVAC)
- Internet of things
- GPS enhancement (e.g. time-to-first-fix improvement, dead reckoning, slope detection)
- Indoor navigation (change of floor detection, elevator detection)
- Outdoor navigation, leisure and sports applications
- Weather forecast
- Vertical velocity indication (rise/sink speed)

Target devices

- Handsets such as mobile phones, tablet PCs, GPS devices
- Navigation systems
- Gaming, e.g flying toys
- Camera (DSC, video)
- Home weather stations
- Flying toys
- Watches

General Description

The BME280 is as combined digital humidity, pressure and temperature sensor based on proven sensing principles. The sensor module is housed in an extremely compact metal-lid LGA package with a footprint of only 2.5 × 2.5 mm² with a height of 0.93 mm. Its small dimensions and its low power consumption allow the implementation in battery driven devices such as handsets, GPS modules or watches. The BME280 is register and performance compatible to the Bosch Sensortec BMP280 digital pressure sensor (see chapter 5.2 for details).

The BME280 achieves high performance in all applications requiring humidity and pressure measurement. These emerging applications of home automation control, in-door navigation, fitness as well as GPS refinement require a high accuracy and a low TCO at the same time.

The humidity sensor provides an extremely fast response time for fast context awareness applications and high overall accuracy over a wide temperature range.

The pressure sensor is an absolute barometric pressure sensor with extremely high accuracy and resolution and drastically lower noise than the Bosch Sensortec BMP180.

The integrated temperature sensor has been optimized for lowest noise and highest resolution. Its output is used for temperature compensation of the pressure and humidity sensors and can also be used for estimation of the ambient temperature.

The sensor provides both SPI and I²C interfaces and can be supplied using 1.71 to 3.6 V for the sensor supply V_{DD} and 1.2 to 3.6 V for the interface supply V_{DDIO}. Measurements can be triggered by the host or performed in regular intervals. When the sensor is disabled, current consumption drops to 0.1 µA.

BME280 can be operated in three power modes (see chapter 3.3):

- sleep mode
- normal mode
- forced mode

In order to tailor data rate, noise, response time and current consumption to the needs of the user, a variety of oversampling modes, filter modes and data rates can be selected.

Please contact your regional Bosch Sensortec partner for more information about software packages.

Index of Contents

1. Specification

If not stated otherwise,

- All values are valid over the full voltage range
- All minimum/maximum values are given for the full accuracy temperature range
- Minimum/maximum values of drifts, offsets and temperature coefficients are $\pm 3\sigma$ values over lifetime
- Typical values of currents and state machine timings are determined at 25 °C
- Minimum/maximum values of currents are determined using corner lots over complete temperature range
- Minimum/maximum values of state machine timings are determined using corner lots over 0…+65 °C temperature range

The specification tables are split into humidity, pressure, and temperature part of BME280.

1.1 General electrical specification

Table 1: Electrical parameter specification

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Supply Voltage Internal Domains	V_{DD}	ripple max. 50 mVpp	1.71	1.8	3.6	V
Supply Voltage I/O Domain	V_{DDIO}		1.2	1.8	3.6	V
Sleep current	I_{DDSL}			0.1	0.3	µA
Standby current (inactive period of normal mode)	I_{DDSB}			0.2	0.5	µA
Current during humidity measurement	I_{DDH}	Max value at 85 °C		340		µA
Current during pressure measurement	I_{DDP}	Max value at -40 °C		714		µA
Current during temperature measurement	I_{DDT}	Max value at 85 °C		350		µA
Start-up time	$t_{startup}$	Time to first communication after both $V_{DD} > 1.58$ V and $V_{DDIO} > 0.65$ V			2	ms
Power supply rejection ratio (DC)	PSRR	full V_{DD} range			±0.01 ±5	%RH/V Pa/V
Standby time accuracy	$\Delta t_{standby}$			±5	±25	%

1.2 Humidity parameter specification

Table 2: Humidity parameter specification

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Operating range[1]	R_H	For temperatures < 0 °C and > 60 °C see Figure 1	-40	25	85	°C
			0		100	%RH
Supply current	$I_{DD,H}$	1 Hz forced mode, humidity and temperature		1.8	2.8	µA
Absolute accuracy tolerance	A_H	20…80 %RH, 25 °C, including hysteresis	-9	±3	+9	%RH
Hysteresis[2]	H_H	10→90→10 %RH, 25 °C	-3	±1	+3	%RH
Nonlinearity[3]	NL_H	10→90 %RH, 25 °C	0	1	3	%RH
Response time to complete 63% of step[4]	$\tau_{63\%}$	90→0 or 0→90 %RH, 25°C		1		s
Resolution	R_H			0.008		%RH
Noise in humidity (RMS)	N_H	Highest oversampling, see chapter 3.6		0.02		%RH
Long term stability	ΔH_{stab}	10…90 %RH, 25 °C		0.5		%RH/ year

---

[1] When exceeding the operating range (e.g. for soldering), humidity sensing performance is temporarily degraded and reconditioning is recommended as described in section 7.8. Operating range only for non-condensing environment.

[2] For hysteresis measurement the sequence 10→30→50→70→90→70→50→30→10 %RH is used. The hysteresis is defined as the difference between measurements of the humidity up / down branch and the averaged curve of both branches

[3] Non-linear contributions to the sensor data are corrected during the calculation of the relative humidity by the compensation formulas described in section 4.2.3.

[4] The air-flow in direction to the vent-hole of the device has to be dimensioned in a way that a sufficient air exchange inside to outside will be possible. To observe effects on the response time-scale of the device an air-flow velocity of approx. 1 m/s is needed.

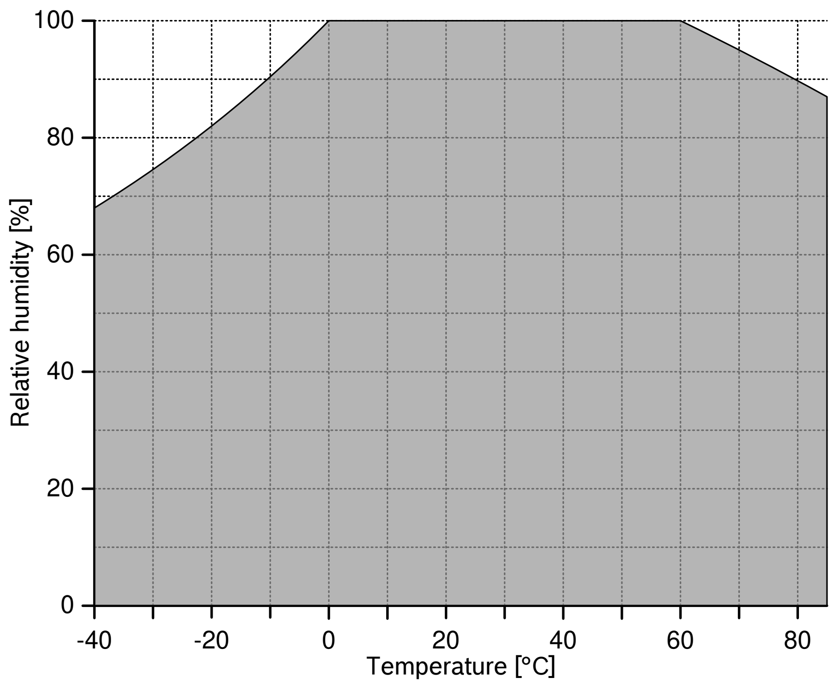

Figure 1: humidity sensor operating range

1.3 Pressure sensor specification

Table 3: Pressure parameter specification

Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Unit**
Operating temperature range	T_A	operational	-40	25	+85	°C
		full accuracy	0		+65	
Operating pressure range	P	full accuracy	300		1100	hPa
Supply current	$I_{DD,LP}$	1 Hz forced mode, pressure and temperature, lowest power		2.8	4.2	µA
Temperature coefficient of offset[5]	TCO_P	25…65 °C, 900 hPa		±1.5		Pa/K
				±12.6		cm/K
Absolute accuracy pressure						
	$A_{P,full}$	300 . . . 1100 hPa 0 . . . 65 °C	-3	±1.0	+3	hPa
Relative accuracy pressure V_{DD} = 3.3V	A_{rel}	700 ... 900hPa 25 . . . 40 °C	-0.4	±0.12	+0.4	hPa

---

[5] When changing temperature by e.g. 10 °C at constant pressure / altitude, the measured pressure / altitude will change by (10 × TCO_P).

Resolution of pressure output data	R_P	Highest oversampling		0.18		Pa
Noise in pressure	$N_{P,fullBW}$	Full bandwidth, highest oversampling See chapter 3.6	0.2	1.3	3.3	Pa
				11		cm
	$N_{P,filtered}$	Reduced bandwidth, highest oversampling See chapter 3.6	0.02	0.2	1.5	Pa
				1.7		cm
Solder drift		Minimum solder height 50µm	-0.5		+2.0	hPa
Long term stability[6]	ΔP_{stab}	per year	-1	±1.0	+1	hPa
Possible sampling rate	f_{sample_P}	Lowest oversampling, see chapter 9.2	157	182		Hz

1.4 Temperature sensor specification

Table 4: Temperature parameter specification

Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Unit**
Operating range	T	Operational	-40	25	85	°C
		Full accuracy	0		65	°C
Supply current	$I_{DD,T}$	1 Hz forced mode, temperature measurement only		1.0		µA
Absolute accuracy temperature[7]	$A_{T,25}$	25 °C	-1.5	±0.5	+1.5	°C
	$A_{T,full}$	0…65 °C	-3.0	±1.0	+3.0	°C
Output resolution	R_T	API output resolution	0.01			°C
RMS noise	N_T	Lowest oversampling		0.005		°C

---

[6] Long term stability is specified in the full accuracy operating pressure range 0 … 65 °C

[7] Temperature measured by the internal temperature sensor. This temperature value depends on the PCB temperature, sensor element self-heating and ambient temperature and is typically above ambient temperature.

2. Absolute maximum ratings

The absolute maximum ratings are determined over complete temperature range using corner lots. The values are provided in Table 5.

Table 5: Absolute maximum ratings

Parameter	Condition	Min	Max	Unit
Voltage at any supply pin	V_{DD} and V_{DDIO} pin	-0.3	4.25	V
Voltage at any interface pin		-0.3	V_{DDIO} + 0.3	V
Storage temperature	≤ 65% RH	-45	+85	°C
Pressure		0	20 000	hPa
ESD	HBM, at any pin		±2	kV
	CDM		±500	V
	Machine model		±200	V
Condensation	No power supplied	Allowed		

3. Functional description

3.1 Block diagram

Figure 2 shows a simplified block diagram of the BME280:

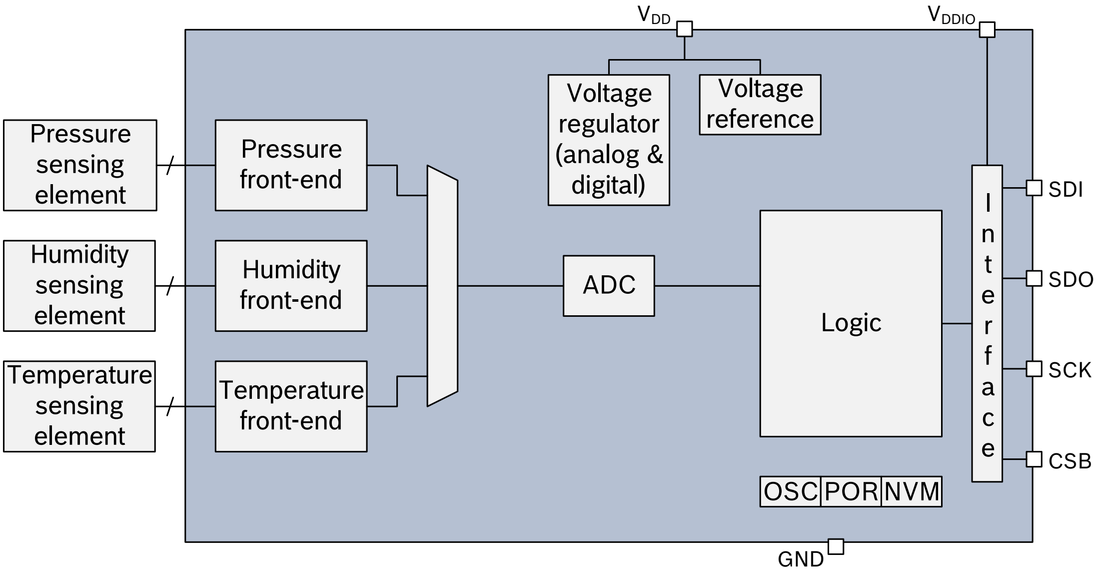

Figure 2: Block diagram of BME280

3.2 Power management

The BME280 has two distinct power supply pins

- V_{DD} is the main power supply for all internal analog and digital functional blocks
- V_{DDIO} is a separate power supply pin used for the supply of the digital interface

A power-on reset (POR) generator is built in; it resets the logic part and the register values after both V_{DD} and V_{DDIO} reach their minimum levels. There are no limitations on slope and sequence of raising the V_{DD} and V_{DDIO} levels. After powering up, the sensor settles in sleep mode (described in chapter 3.3.2).

It is prohibited to keep any interface pin (SDI, SDO, SCK or CSB) at a logical high level when V_{DDIO} is switched off. Such a configuration can permanently damage the device due an excessive current flow through the ESD protection diodes.

If V_{DDIO} is supplied, but V_{DD} is not, the interface pins are kept at a high-Z level. The bus can therefore already be used freely before the BME280 V_{DD} supply is established.

Resetting the sensor is possible by cycling V_{DD} level or by writing a soft reset command. Cycling the V_{DDIO} level will not cause a reset.

3.3 Sensor modes

The BME280 offers three sensor modes: sleep mode, forced mode and normal mode. These can be selected using the *mode*[1:0] setting (see chapter 5.4.5). The available modes are:

- Sleep mode: no operation, all registers accessible, lowest power, selected after startup
- Forced mode: perform one measurement, store results and return to sleep mode
- Normal mode: perpetual cycling of measurements and inactive periods.

The modes will be explained in detail in chapters 3.3.2 (sleep mode), 3.3.3 (forced mode) and 3.3.4 (normal mode).

3.3.1 Sensor mode transitions

The supported mode transitions are shown in Figure 3. If the device is currently performing a measurement, execution of mode switching commands is delayed until the end of the currently running measurement period. Further mode change commands or other write commands to the register *ctrl_hum* are ignored until the mode change command has been executed. Mode transitions other than the ones shown below are tested for stability but do not represent recommended use of the device.

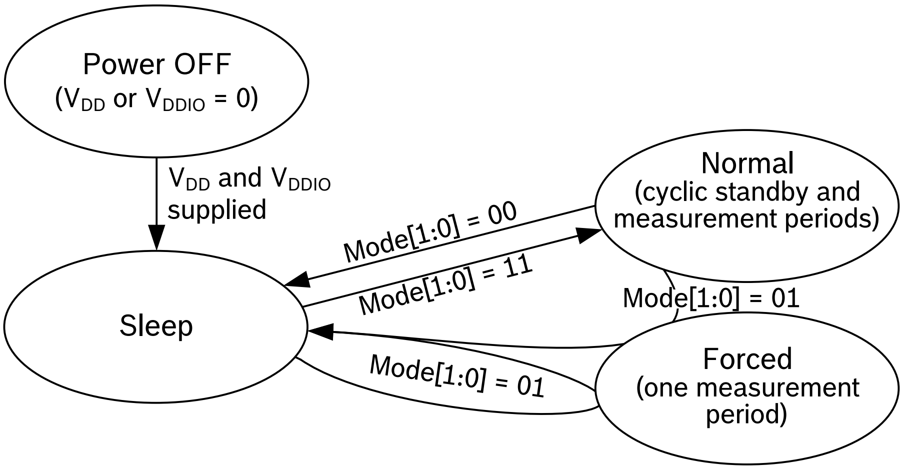

Figure 3: Sensor mode transition diagram

3.3.2 Sleep mode

Sleep mode is entered by default after power on reset. In sleep mode, no measurements are performed and power consumption (I_{DDSM}) is at a minimum. All registers are accessible; Chip-ID and compensation coefficients can be read. There are no special restrictions on interface timings.

3.3.3 Forced mode

In forced mode, a single measurement is performed in accordance to the selected measurement and filter options. When the measurement is finished, the sensor returns to sleep mode and the measurement results can be obtained from the data registers. For a next measurement, forced mode needs to be selected again. This is similar to BMP180 operation. Using forced mode is recommended for applications which require low sampling rate or host-based synchronization. The timing diagram is shown below.

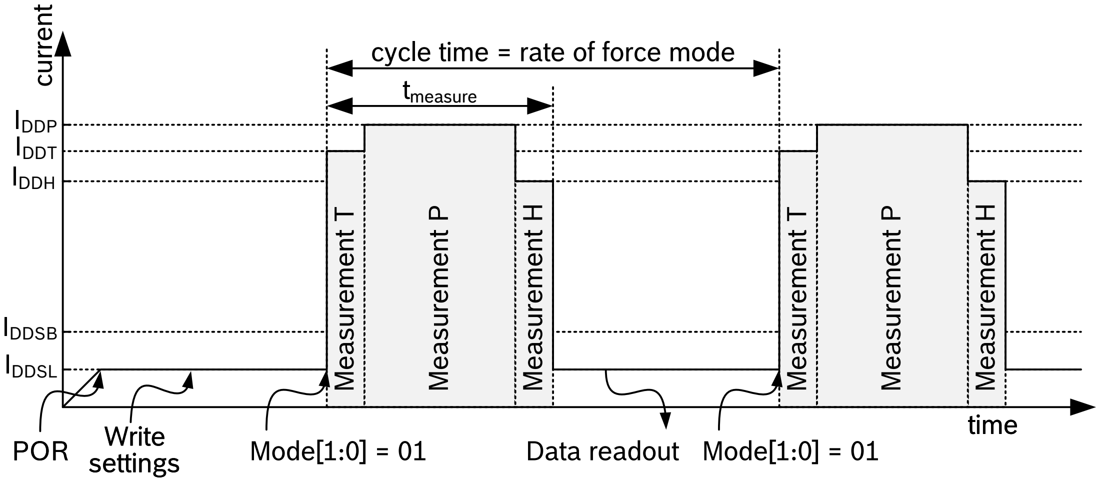

Figure 4: Forced mode timing diagram

3.3.4 Normal mode

Normal mode comprises an automated perpetual cycling between an (active) measurement period and an (inactive) standby period.

The measurements are performed in accordance to the selected measurement and filter options. The standby time is determined by the setting *t_sb*[2:0] and can be set to between 0.5 and 1000 ms according to Table 27.

The total cycle time depends on the sum of the active time (see chapter 9) and standby time $t_{standby}$.

The current in the standby period (I_{DDSB}) is slightly higher than in sleep mode. After setting the measurement and filter options and enabling normal mode, the last measurement results can always be obtained at the data registers without the need of further write accesses.

Using normal mode is recommended when using the IIR filter. This is useful for applications in which short-term disturbances (e.g. blowing into the sensor) should be filtered. The timing diagram is shown below:

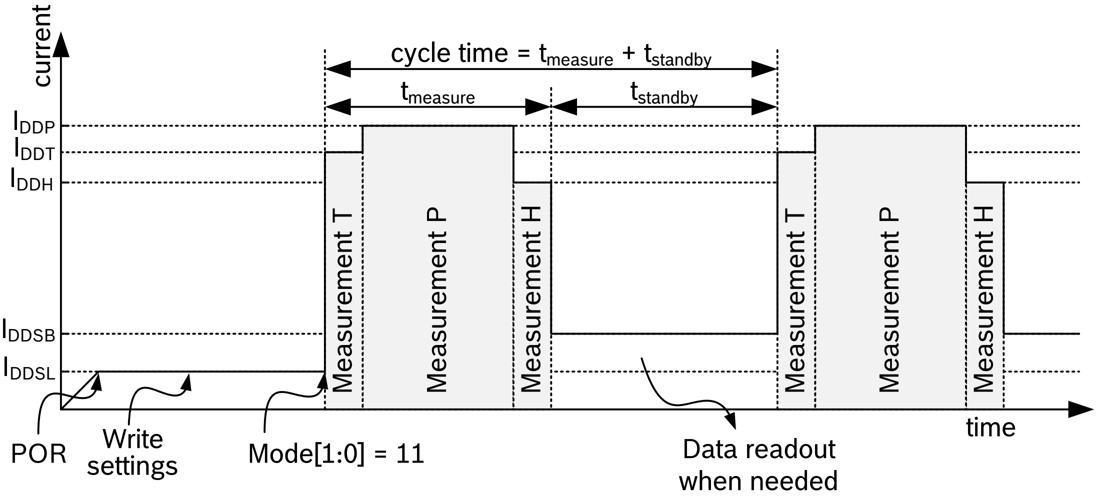

Figure 5: Normal mode timing diagram

3.4 Measurement flow

The BME280 measurement period consists of a temperature, pressure and humidity measurement with selectable oversampling. After the measurement period, the pressure and temperature data can be passed through an optional IIR filter, which removes short-term fluctuations in pressure (e.g. caused by slamming a door). For humidity, such a filter is not needed and has not been implemented. The flow is depicted in the diagram below.

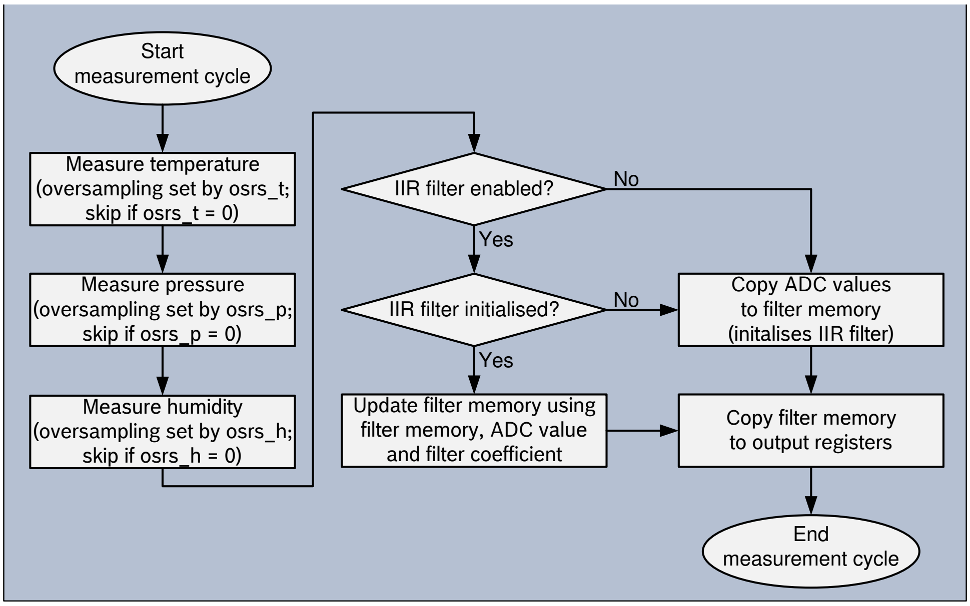

Figure 6: BME280 measurement cycle

The individual blocks of the diagram above will be detailed in the following subchapters.

3.4.1 Humidity measurement

The humidity measurement can be enabled or skipped. When enabled, several oversampling options exist. The humidity measurement is controlled by the *osrs_h*[2:0] setting, which is detailed in chapter 5.4.3. For the humidity measurement, oversampling is possible to reduce the noise. The resolution of the humidity measurement is fixed at 16 bit ADC output.

3.4.2 Pressure measurement

Pressure measurement can be enabled or skipped. When enabled, several oversampling options exist. The pressure measurement is controlled by the *osrs_p*[2:0] setting which is detailed in chapter 5.4.5. For the pressure measurement, oversampling is possible to reduce the noise. The resolution of the pressure data depends on the IIR filter (see chapter 3.4.4) and the oversampling setting (see chapter 5.4.5):

- When the IIR filter is enabled, the pressure resolution is 20 bit.
- When the IIR filter is disabled, the pressure resolution is 16 + (*osrs_p* – 1) bit, e.g. 18 bit when *osrs_p* is set to ‘3’.

3.4.3 Temperature measurement

Temperature measurement can be enabled or skipped. Skipping the measurement could be useful to measure pressure extremely rapidly. When enabled, several oversampling options exist. The temperature measurement is controlled by the *osrs_t*[2:0] setting which is detailed in chapter 5.4.5. For the temperature measurement, oversampling is possible to reduce the noise.
The resolution of the temperature data depends on the IIR filter (see chapter 3.4.4) and the oversampling setting (see chapter 5.4.5):

- When the IIR filter is enabled, the temperature resolution is 20 bit.
- When the IIR filter is disabled, the temperature resolution is 16 + (*osrs_t* – 1) bit, e.g. 18 bit when *osrs_t* is set to ‘3’.

3.4.4 IIR filter

The humidity value inside the sensor does not fluctuate rapidly and does not require low pass filtering. However, the environmental pressure is subject to many short-term changes, caused e.g. by slamming of a door or window, or wind blowing into the sensor. To suppress these disturbances in the output data without causing additional interface traffic and processor work load, the BME280 features an internal IIR filter. It effectively reduces the bandwidth of the temperature and pressure output signals[8] and increases the resolution of the pressure and temperature output data to 20 bit. The output of a next measurement step is filtered using the following formula:

$$\text{data\_filtered} = \frac{\text{data\_filtered\_old} \cdot (\text{filter\_coefficient} - 1) + \text{data\_ADC}}{\text{filter\_coefficient}}$$

Data_filtered_old is the data coming from the current filter memory, and data_ADC is the data coming from current ADC acquisition. Data_filtered is the new value of filter memory and the value that will be sent to the output registers.

The IIR filter can be configured to different filter coefficients, which slows down the response to the sensor inputs. Note that the response time with enabled IIR filter depends on the number of samples generated, which means that the data output rate must be known to calculate the actual response

---

[8] Since the BME280 does not sample continuously, filtering can suffer from signals with a frequency higher than the sampling rate of the sensor. E.g. environmental fluctuations caused by windows being opened and closed might have a frequency <5 Hz. Consequently, a sampling rate of ODR = 10 Hz is sufficient to obey the Nyquist theorem.

time. For register configuration, please refer to Table 28. A sample response time calculation is shown in chapter 9.4.

Table 6: *filter* settings

Filter coefficient	**Samples to reach ≥75 % of step response**
Filter off	1
2	2
4	5
8	11
16	22

In order to find a suitable setting for *filter*, please consult chapter 3.5.

When writing to the register *filter*, the filter is reset. The next ADC values will the pass through the filter unchanged and become the initial memory values for the filter. If temperature or pressure measurements are skipped, the corresponding filter memory will be kept unchanged even though the output registers are set to 0x80000. When the previously skipped measurement is re-enabled, the output will be filtered using the filter memory from the last time when the measurement was not skipped. If this is not desired, please write to the *filter* register in order to re-initialize the filter.

The step response (e.g. response to in sudden change in height) of the different filter settings is displayed in Figure 7.

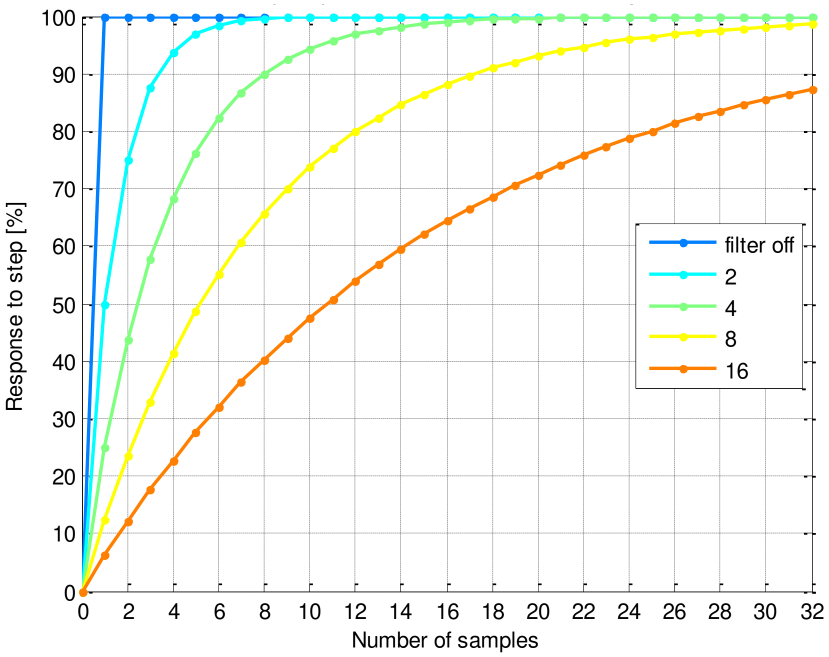

Figure 7: Step response at different IIR filter settings

3.5 Recommended modes of operation

The different oversampling options, filter settings and sensor modes result in a large number of possible settings. In this chapter, a number of settings recommended for various scenarios are presented.

3.5.1 Weather monitoring

Description: Only a very low data rate is needed. Power consumption is minimal. Noise of pressure values is of no concern. Humidity, pressure and temperature are monitored.

Table 7: Settings and performance for weather monitoring

Suggested settings for weather monitoring	
Sensor mode	forced mode, 1 sample / minute
Oversampling settings	pressure ×1, temperature ×1, humidity ×1
IIR filter settings	filter off
Performance for suggested settings	
Current consumption	0.16 µA
RMS Noise	3.3 Pa / 30 cm, 0.07 %RH
Data output rate	1/60 Hz

3.5.2 Humidity sensing

Description: A low data rate is needed. Power consumption is minimal. Forced mode is used to minimize power consumption and to synchronize readout, but using normal mode would also be possible.

Table 8: Settings and performance for humidity sensing

Suggested settings for weather monitoring	
Sensor mode	forced mode, 1 sample / second
Oversampling settings	pressure ×0, temperature ×1, humidity ×1
IIR filter settings	filter off
Performance for suggested settings	
Current consumption	2.9 µA
RMS Noise	0.07 %RH
Data output rate	1 Hz

3.5.3 Indoor navigation

Lowest possible altitude noise is needed. A very low bandwidth is preferred. Increased power consumption is tolerated. Humidity is measured to help detect room changes. This setting is suggested for the Android settings ‘SENSOR_DELAY_NORMAL’ and ‘SENSOR_DELAY_UI’.

Table 9: Settings and performance for indoor navigation

Suggested settings for indoor navigation	
Sensor mode	normal mode, $t_{standby}$ = 0.5 ms
Oversampling settings	pressure ×16, temperature ×2, humidity ×1
IIR filter settings	filter coefficient 16
Performance for suggested settings	
Current consumption	633 µA
RMS Noise	0.2 Pa / 1.7 cm
Data output rate	25Hz
Filter bandwidth	0.53 Hz
Response time (75%)	0.9 s

3.5.4 Gaming

Low altitude noise is needed. The required bandwidth is ~2 Hz in order to respond quickly to altitude changes (e.g. be able to dodge a flying monster in a game). Increased power consumption is tolerated. Humidity sensor is disabled. This setting is suggested for the Android settings 'SENSOR_DELAY_GAMING' and 'SENSOR_DELAY_FASTEST'.

Table 10: Settings and performance for gaming

Suggested settings for gaming	
Sensor mode	normal mode, $t_{standby}$ = 0.5 ms
Oversampling settings	pressure ×4, temperature ×1, humidity ×0
IIR filter settings	filter coefficient 16
Performance for suggested settings	
Current consumption	581 µA
RMS Noise	0.3 Pa / 2.5 cm
Data output rate	83 Hz
Filter bandwidth	1.75 Hz
Response time (75%)	0.3 s

3.6 Noise

The noise depends on the oversampling and, for pressure and temperature, on the filter setting used. The stated values were determined in a controlled environment and are based on the average standard deviation of 32 consecutive measurement points taken at highest sampling speed. This is needed in order to exclude long term drifts from the noise measurement. The noise depends both on humidity/pressure oversampling and temperature oversampling, since the temperature value is used for humidity/pressure temperature compensation. The oversampling combinations use below results in an optimal power to noise ratio.

Table 11: Noise and current for humidity

Humidity / temperature oversampling setting	**Typical RMS noise in humidity [%RH] at 25 °C**	**Typ. current [µA] at 1 Hz forced mode, 25 °C, humidity and temperature measurement, incl. I_{DDSM}**
×1 / ×1	0.07	1.8
×2 / ×1	0.05	2.5
×4 / ×1	0.04	3.8
×8 / ×1	0.03	6.5
×16 / ×1	0.02	11.7

Table 12: Noise and current for pressure

Typical RMS noise in pressure [Pa] at 25 °C						**Typ. current [µA] at 1 Hz forced mode, 25 °C, pressure and temperature measurement, incl. I_{DDSM}**
Pressure / temperature oversampling setting	**IIR filter coefficient**					
	off	**2**	**4**	**8**	**16**	
×1 / ×1	3.3	1.9	1.2	0.9	0.4	2.8
×2 / ×1	2.6	1.5	1.0	0.6	0.4	4.2
×4 / ×1	2.1	1.2	0.8	0.5	0.3	7.1
×8 / ×1	1.6	1.0	0.6	0.4	0.2	12.8
×16 / ×2	1.3	0.8	0.5	0.4	0.2	24.9

Table 13: Temperature dependence of pressure noise

RMS noise at different temperatures	
Temperature	Typical change in noise compared to 25 °C
-10 °C	+25 %
25 °C	±0 %
75 °C	-5 %

Table 14: Noise in temperature

Temperature oversampling setting	**Typical RMS noise in temperature [°C] at 25 °C**
×1	0.005
×2	0.004
×4	0.003
×8	0.003
×16	0.002

4. Data readout

To read out data after a conversion, it is strongly recommended to use a burst read and not address every register individually. This will prevent a possible mix-up of bytes belonging to different measurements and reduce interface traffic. Note that in I²C mode, even when pressure was not measured, reading the unused registers is faster than reading temperature and humidity data separately.
Data readout is done by starting a burst read from 0xF7 to 0xFC (temperature and pressure) or from 0xF7 to 0xFE (temperature, pressure and humidity). The data are read out in an unsigned 20-bit format both for pressure and for temperature and in an unsigned 16-bit format for humidity. It is strongly recommended to use the BME280 API, available from Bosch Sensortec, for readout and compensation. For details on memory map and interfaces, please consult chapters 5 and 6 respectively.

After the uncompensated values for pressure, temperature and humidity 'ut', 'up' and 'uh' have been read, the actual humidity, pressure and temperature needs to be calculated using the compensation parameters stored in the device. The procedure is elaborated in chapter 4.2.

4.1 Data register shadowing

In normal mode, the timing of measurements is not necessarily synchronized to the readout by the user. This means that new measurement results may become available while the user is reading the results from the previous measurement. In this case, shadowing is performed in order to guarantee data consistency. Shadowing will only work if all data registers are read in a single burst read. Therefore, the user must use burst reads if he does not synchronize data readout with the measurement cycle. Using several independent read commands may result in inconsistent data.
If a new measurement is finished and the data registers are still being read, the new measurement results are transferred into shadow data registers. The content of shadow registers is transferred into data registers as soon as the user ends the burst read, even if not all data registers were read.
The end of the burst read is marked by the rising edge of CSB pin in SPI case or by the recognition of a stop condition in I²C case. After the end of the burst read, all user data registers are updated at once.

4.2 Output compensation

The BME280 output consists of the ADC output values. However, each sensing element behaves differently. Therefore, the actual pressure and temperature must be calculated using a set of calibration parameters. In this chapter, the method to read out the trimming values will be given. The recommended calculation uses fixed point arithmetic and is given in chapter 4.2.3.
In high-level languages like Matlab™ or LabVIEW™, fixed-point code may not be well supported. In this case the floating-point code in appendix 8.1 can be used as an alternative.
For 8-bit micro controllers, the variable size may be limited. In this case a simplified 32 bit integer code with reduced accuracy is given in appendix 8.2.

4.2.1 Computational requirements

In the table below an overview is given for the number of clock cycles needed for compensation on a 32 bit Cortex-M3 micro controller with GCC optimization level -O2. This controller does not feature a floating point unit, thus all floating-point calculations are emulated. Floating point is only recommended for PC application, where an FPU is present and these calculations are performed drastically faster.

Table 15: Computational requirements for compensation formulas

Compensation of	**Number of clocks (ARM Cortex-M3)**		
	32 bit integer	64 bit integer	Double precision
Humidity	~83	–	~2900 [9]
Temperature	~46	–	~2400 [9]
Pressure	~112 [10]	~1400	~5400 [9]

4.2.2 Trimming parameter readout

The trimming parameters are programmed into the devices' non-volatile memory (NVM) during production and cannot be altered by the customer. Each compensation word is a 16-bit signed or unsigned integer value stored in two's complement. As the memory is organized into 8-bit words, two words must always be combined in order to represent the compensation word. The 8-bit registers are named calib00…calib41 and are stored at memory addresses 0x88…0xA1 and 0xE1…0xE7. The corresponding compensation words are named dig_T# for temperature compensation related values, dig_P# for pressure related values and dig_H# for humidity related values. The mapping is seen in Table 16.

Table 16: Compensation parameter storage, naming and data type

Register Address	**Register content**	**Data type**
0x88 / 0x89	dig_T1 [7:0] / [15:8]	unsigned short
0x8A / 0x8B	dig_T2 [7:0] / [15:8]	signed short
0x8C / 0x8D	dig_T3 [7:0] / [15:8]	signed short
0x8E / 0x8F	dig_P1 [7:0] / [15:8]	unsigned short
0x90 / 0x91	dig_P2 [7:0] / [15:8]	signed short
0x92 / 0x93	dig_P3 [7:0] / [15:8]	signed short
0x94 / 0x95	dig_P4 [7:0] / [15:8]	signed short
0x96 / 0x97	dig_P5 [7:0] / [15:8]	signed short
0x98 / 0x99	dig_P6 [7:0] / [15:8]	signed short
0x9A / 0x9B	dig_P7 [7:0] / [15:8]	signed short
0x9C / 0x9D	dig_P8 [7:0] / [15:8]	signed short
0x9E / 0x9F	dig_P9 [7:0] / [15:8]	signed short
0xA1	dig_H1 [7:0]	unsigned char
0xE1 / 0xE2	dig_H2 [7:0] / [15:8]	signed short
0xE3	dig_H3 [7:0]	unsigned char
0xE4 / 0xE5[3:0]	dig_H4 [11:4] / [3:0]	signed short
0xE5[7:4] / 0xE6	dig_H5 [3:0] / [11:4]	signed short

---

[9] Use only recommended for high-level programming languages like Matlab™ or LabVIEW™

[10] Use only recommended for 8-bit micro controllers

0xE7	dig_H6	signed char

4.2.3 Compensation formulas

Please note that it is strongly advised to use the API available from Bosch Sensortec to perform readout and compensation. If this is not wanted, the code below can be applied at the user’s risk. Both pressure and temperature values are expected to be received in 20 bit format, positive, stored in a 32 bit signed integer. Humidity is expected to be received in 16 bit format, positive, stored in a 32 bit signed integer.

The variable t_fine (signed 32 bit) carries a fine resolution temperature value over to the pressure and humidity compensation formula and could be implemented as a global variable.

The data type “BME280_S32_t” should define a 32 bit signed integer variable type and can usually be defined as “long signed int”.

The data type “BME280_U32_t” should define a 32 bit unsigned integer variable type and can usually be defined as “long unsigned int”.

For best possible calculation accuracy in pressure, 64 bit integer support is needed. If this is not possible on your platform, please see appendix 8.2 for a 32 bit alternative.

The data type “BME280_S64_t” should define a 64 bit signed integer variable type, which on most supporting platforms can be defined as “long long signed int”. The revision of the code is rev.1.1.

```
// Returns temperature in DegC, resolution is 0.01 DegC. Output value of “5123” equals 51.23 DegC.
// t_fine carries fine temperature as global value
BME280_S32_t t_fine;
BME280_S32_t BME280_compensate_T_int32(BME280_S32_t adc_T)
{
   BME280_S32_t var1, var2, T;
   var1  = ((((adc_T>>3) – ((BME280_S32_t)dig_T1<<1))) * ((BME280_S32_t)dig_T2)) >> 11;
   var2  = (((((adc_T>>4) – ((BME280_S32_t)dig_T1)) * ((adc_T>>4) – ((BME280_S32_t)dig_T1))) >> 12) *
      ((BME280_S32_t)dig_T3)) >> 14;
   t_fine = var1 + var2;
   T  = (t_fine * 5 + 128) >> 8;
   return T;
}

// Returns pressure in Pa as unsigned 32 bit integer in Q24.8 format (24 integer bits and 8 fractional
bits).
// Output value of “24674867” represents 24674867/256 = 96386.2 Pa = 963.862 hPa
BME280_U32_t BME280_compensate_P_int64(BME280_S32_t adc_P)
{
   BME280_S64_t var1, var2, p;
   var1 = ((BME280_S64_t)t_fine) – 128000;
   var2 = var1 * var1 * (BME280_S64_t)dig_P6;
   var2 = var2 + ((var1*(BME280_S64_t)dig_P5)<<17);
   var2 = var2 + (((BME280_S64_t)dig_P4)<<35);
   var1 = ((var1 * var1 * (BME280_S64_t)dig_P3)>>8) + ((var1 * (BME280_S64_t)dig_P2)<<12);
   var1 = (((((BME280_S64_t)1)<<47)+var1))*((BME280_S64_t)dig_P1)>>33;
   if (var1 == 0)
   {
      return 0; // avoid exception caused by division by zero
   }
   p = 1048576-adc_P;
   p = (((p<<31)-var2)*3125)/var1;
```

```
    var1 = (((BME280_S64_t)dig_P9) * (p>>13) * (p>>13)) >> 25;
    var2 = (((BME280_S64_t)dig_P8) * p) >> 19;
    p = ((p + var1 + var2) >> 8) + (((BME280_S64_t)dig_P7)<<4);
    return (BME280_U32_t)p;
}

// Returns humidity in %RH as unsigned 32 bit integer in Q22.10 format (22 integer and 10 fractional bits).
// Output value of “47445” represents 47445/1024 = 46.333 %RH
BME280_U32_t bme280_compensate_H_int32(BME280_S32_t adc_H)
{
    BME280_S32_t v_x1_u32r;

    v_x1_u32r = (t_fine - ((BME280_S32_t)76800));
    v_x1_u32r = (((((adc_H << 14) - (((BME280_S32_t)dig_H4) << 20) - (((BME280_S32_t)dig_H5) * v_x1_u32r)) +
        ((BME280_S32_t)16384)) >> 15) * (((((((v_x1_u32r * ((BME280_S32_t)dig_H6)) >> 10) * (((v_x1_u32r *
        ((BME280_S32_t)dig_H3)) >> 11) + ((BME280_S32_t)32768))) >> 10) + ((BME280_S32_t)2097152)) *
        ((BME280_S32_t)dig_H2) + 8192) >> 14));
    v_x1_u32r = (v_x1_u32r - (((((v_x1_u32r >> 15) * (v_x1_u32r >> 15)) >> 7) * ((BME280_S32_t)dig_H1)) >>
        4));
    v_x1_u32r = (v_x1_u32r < 0 ? 0 : v_x1_u32r);
    v_x1_u32r = (v_x1_u32r > 419430400 ? 419430400 : v_x1_u32r);
    return (BME280_U32_t)(v_x1_u32r>>12);
}
```

5. Global memory map and register description

5.1 General remarks

The entire communication with the device is performed by reading from and writing to registers. Registers have a width of 8 bits. There are several registers which are reserved; they should not be written to and no specific value is guaranteed when they are read. For details on the interface, consult chapter 6.

5.2 Register compatibility to BMP280

The BME280 is downward register compatible to the BMP280, which means that the pressure and temperature control and readout is identical to BMP280. However, the following exceptions have to be considered: