

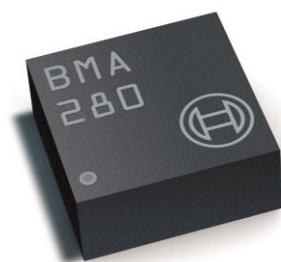
BMA280

Digital, triaxial acceleration sensor

Bosch Sensortec



BOSCH
Invented for life



BMA280: Data sheet

| | |
|-----------------------|--|
| Document revision | 1.11 |
| Document release date | October 2021 |
| Document number | BST-BMA280-DS000-14 |
| Sales Part Number(s) | 0 273 141 148 |
| Notes | Data 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's appearance. |

BMA280

14 BIT, DIGITAL, TRIAXIAL ACCELERATION SENSOR WITH INTELLIGENT ON-CHIP MOTION-TRIGGERED INTERRUPT CONTROLLER

Key features

- Ultra-Small package LGA package (12 pins), footprint 2mm x 2mm, height 0.95mm
- Digital interface SPI (4-wire, 3-wire), I²C, 2 interrupt pins
V_{DDIO} voltage range: 1.2V to 3.6V
- Programmable functionality Acceleration ranges $\pm 2g/\pm 4g/\pm 8g/\pm 16g$
Low-pass filter bandwidths 500Hz - <8Hz
up to an max. output data read out of 2kHz (unfiltered)
Integrated FIFO with a depth of 32 frames
- On-chip FIFO Motion-triggered interrupt-signal generation for
 - new data
 - any-motion (slope) detection
 - tap sensing (single tap / double tap)
 - orientation recognition
 - flat detection
 - low-g/high-g detection
 - no-motion / inactivity detection
- On-chip interrupt controller
- Ultra-low power Low current consumption, short wake-up time, advanced features for system power management
- Temperature sensor
- RoHS compliant, halogen-free

Typical applications

- Display profile switching
- Menu scrolling, tap / double tap sensing
- Gaming
- Pedometer / step counting
- Free-fall detection
- E-compass tilt compensation
- Drop detection for warranty logging
- Advanced system power management for mobile applications

General description

The BMA280 is a triaxial, low-g acceleration sensor with digital output for consumer applications. It allows measurements of acceleration in three perpendicular axes. An evaluation circuitry (ASIC) converts the output of a micromechanical acceleration-sensing structure (MEMS) that works according to the differential capacitance principle.

Package and interfaces of the BMA280 have been defined to match a multitude of hardware requirements. Since the sensor features an ultra-small footprint and a flat package it is ingeniously suited for mobile applications.



The BMA280 offers a variable V_{DDIO} voltage range from 1.2V to 3.6V and can be programmed to optimize functionality, performance and power consumption in customer specific applications. In addition it features an on-chip interrupt controller enabling motion-based applications without use of a microcontroller.

The BMA280 senses tilt, motion, inactivity and shock vibration in cell phones, handhelds, computer peripherals, man-machine interfaces, virtual reality features and game controllers.

Index of Contents

| | |
|---|-----------|
| 1. SPECIFICATION | 2 |
| 2. ABSOLUTE MAXIMUM RATINGS | 11 |
| 3. BLOCK DIAGRAM | 12 |
| 4. FUNCTIONAL DESCRIPTION | 13 |
| 4.1 SUPPLY VOLTAGE AND POWER MANAGEMENT | 13 |
| 4.2 POWER MODES..... | 14 |
| 4.3 SENSOR DATA | 18 |
| 4.3.1 ACCELERATION DATA | 18 |
| 4.3.2 TEMPERATURE SENSOR..... | 19 |
| 4.4 SELF-TEST | 20 |
| 4.5 OFFSET COMPENSATION | 21 |
| 4.5.1 SLOW COMPENSATION..... | 23 |
| 4.5.2 FAST COMPENSATION..... | 23 |
| 4.5.3 MANUAL COMPENSATION..... | 24 |
| 4.5.4 INLINE CALIBRATION | 24 |
| 4.6 NON-VOLATILE MEMORY..... | 25 |
| 4.7 INTERRUPT CONTROLLER..... | 26 |
| 4.7.1 GENERAL FEATURES | 26 |
| 4.7.2 MAPPING TO PHYSICAL INTERRUPT PINS (INTTYPE TO INT PIN#) | 27 |
| 4.7.3 ELECTRICAL BEHAVIOUR (INT PIN# TO OPEN-DRIVE OR PUSH-PULL)..... | 28 |
| 4.7.4 NEW DATA INTERRUPT..... | 28 |
| 4.7.5 SLOPE / ANY-MOTION DETECTION..... | 29 |
| 4.7.6 TAP SENSING | 31 |
| 4.7.7 ORIENTATION RECOGNITION | 34 |
| 4.7.8 FLAT DETECTION..... | 39 |
| 4.7.9 LOW-G INTERRUPT | 40 |
| 4.7.10 HIGH-G INTERRUPT | 41 |
| 4.7.11 NO-MOTION / SLOW MOTION DETECTION | 42 |
| 4.8 SOFTRESET..... | 44 |
| 5. FIFO OPERATION | 45 |
| 5.1 FIFO OPERATING MODES | 45 |
| 5.2 FIFO DATA READOUT | 46 |
| 5.3 FIFO FRAME COUNTER AND OVERRUN FLAG | 46 |
| 5.4 FIFO INTERRUPTS | 47 |

| | |
|--------------------------------------|-----------|
| 6. REGISTER DESCRIPTION | 48 |
| 6.1 GENERAL REMARKS | 48 |
| 6.2 REGISTER MAP | 49 |
| REGISTER 0x00 (BGW_CHIPID) | 50 |
| REGISTER 0x02 (ACCD_X_LSB) | 50 |
| REGISTER 0x03 (ACCD_X_MSB) | 51 |
| REGISTER 0x04 (ACCD_Y_LSB) | 52 |
| REGISTER 0x05 (ACCD_Y_MSB) | 53 |
| REGISTER 0x06 (ACCD_Z_LSB) | 54 |
| REGISTER 0x07 (ACCD_Z_MSB) | 55 |
| REGISTER 0x08 (ACCD_TEMP) | 56 |
| REGISTER 0x09 (INT_STATUS_0) | 57 |
| REGISTER 0x0A (INT_STATUS_1) | 58 |
| REGISTER 0x0B (INT_STATUS_2) | 59 |
| REGISTER 0x0C (INT_STATUS_3) | 60 |
| REGISTER 0x0E (FIFO_STATUS) | 61 |
| REGISTER 0x0F (PMU_RANGE) | 62 |
| REGISTER 0x10 (PMU_BW) | 62 |
| REGISTER 0x11 (PMU_LPW) | 63 |
| REGISTER 0x12 (PMU_LOW_NOISE) | 64 |
| REGISTER 0x13 (ACCD_HBW) | 65 |
| REGISTER 0x14 (BGW_SOFTRESET) | 66 |
| REGISTER 0x16 (INT_EN_0) | 66 |
| REGISTER 0x17 (INT_EN_1) | 67 |
| REGISTER 0x18 (INT_EN_2) | 68 |
| REGISTER 0x19 (INT_MAP_0) | 69 |
| REGISTER 0x1A (INT_MAP_1) | 70 |
| REGISTER 0x1B (INT_MAP_2) | 71 |
| REGISTER 0x1E (INT_SRC) | 72 |
| REGISTER 0x20 (INT_OUT_CTRL) | 73 |
| REGISTER 0x21 (INT_RST_LATCH) | 74 |
| REGISTER 0x22 (INT_0) | 74 |
| REGISTER 0x23 (INT_1) | 75 |
| REGISTER 0x24 (INT_2) | 75 |

| | |
|--|------------|
| REGISTER 0x25 (INT_3) | 76 |
| REGISTER 0x26 (INT_4) | 76 |
| REGISTER 0x27 (INT_5) | 77 |
| REGISTER 0x28 (INT_6) | 78 |
| REGISTER 0x29 (INT_7) | 78 |
| REGISTER 0x2A (INT_8) | 79 |
| REGISTER 0x2B (INT_9) | 80 |
| REGISTER 0x2C (INT_A) | 81 |
| REGISTER 0x2D (INT_B) | 82 |
| REGISTER 0x2E (INT_C) | 82 |
| REGISTER 0x2F (INT_D)..... | 83 |
| REGISTER 0x30 (FIFO_CONFIG_0) | 84 |
| REGISTER 0x32 (PMU_SELF_TEST) | 85 |
| REGISTER 0x33 (TRIM_NVM_CTRL) | 86 |
| REGISTER 0x34 (BGW_SPI3_WDT)..... | 87 |
| REGISTER 0x36 (OFC_CTRL) | 88 |
| REGISTER 0x37 (OFC_SETTING) | 89 |
| REGISTER 0x38 (OFC_OFFSET_X) | 90 |
| REGISTER 0x39 (OFC_OFFSET_Y) | 91 |
| REGISTER 0x3A (OFC_OFFSET_Z) | 92 |
| REGISTER 0x3B (TRIM_GP0) | 92 |
| REGISTER 0x3C (TRIM_GP1)..... | 93 |
| REGISTER 0x3E (FIFO_CONFIG_1)..... | 94 |
| REGISTER 0x3F (FIFO_DATA) | 95 |
| 7. DIGITAL INTERFACES | 96 |
| 7.1 SERIAL PERIPHERAL INTERFACE (SPI) | 97 |
| 7.2 INTER-INTEGRATED CIRCUIT (I ² C) | 101 |
| 7.2.1 SPI AND I ² C ACCESS RESTRICTIONS | 104 |
| 8. PIN-OUT AND CONNECTION DIAGRAM | 105 |
| 8.1 PIN-OUT | 105 |
| 8.2 CONNECTION DIAGRAM 4-WIRE SPI..... | 106 |
| 8.3 CONNECTION DIAGRAM 3-WIRE SPI..... | 107 |
| 8.4 CONNECTION DIAGRAM I ² C | 108 |

| | |
|--|------------|
| 9. PACKAGE | 109 |
| 9.1 OUTLINE DIMENSIONS | 109 |
| 9.2 SENSING AXES ORIENTATION..... | 110 |
| 9.3 LANDING PATTERN RECOMMENDATION..... | 111 |
| 9.4 MARKING..... | 112 |
| 9.4.1 MASS PRODUCTION SAMPLES | 112 |
| 9.4.2 ENGINEERING SAMPLES..... | 112 |
| 9.5 SOLDERING GUIDELINES..... | 113 |
| 9.6 HANDLING INSTRUCTIONS | 114 |
| 9.7 TAPE AND REEL SPECIFICATION..... | 115 |
| 9.7.1 ORIENTATION WITHIN THE REEL..... | 116 |
| 9.8 ENVIRONMENTAL SAFETY | 117 |
| 9.8.1 HALOGEN CONTENT | 117 |
| 9.8.2 INTERNAL PACKAGE STRUCTURE..... | 117 |
| 10. LEGAL DISCLAIMER..... | 118 |
| 10.1 ENGINEERING SAMPLES | 118 |
| 10.2 PRODUCT USE | 118 |
| 10.3 APPLICATION EXAMPLES AND HINTS..... | 118 |
| 11. DOCUMENT HISTORY AND MODIFICATION | 119 |

1. Specification

Unless stated otherwise, the given values are over lifetime, operating temperature and voltage ranges. Minimum/maximum values are $\pm 3\sigma$.

Table 1: Parameter specification

| OPERATING CONDITIONS | | | | | | |
|---|--------------|--|---------------|----------|---------------|---------|
| Parameter | Symbol | Condition | Min | Typ | Max | Units |
| Acceleration Range | g_{FS2g} | Selectable via serial digital interface | | ± 2 | | g |
| | g_{FS4g} | | | ± 4 | | g |
| | g_{FS8g} | | | ± 8 | | g |
| | g_{FS16g} | | | ± 16 | | g |
| Supply Voltage Internal Domains | V_{DD} | | 1.62 | 2.4 | 3.6 | V |
| Supply Voltage I/O Domain | V_{DDIO} | | 1.2 | 2.4 | 3.6 | V |
| Voltage Input Low Level | V_{IL} | SPI & I ² C | | | $0.3V_{DDIO}$ | - |
| Voltage Input High Level | V_{IH} | SPI & I ² C | $0.7V_{DDIO}$ | | | - |
| Voltage Output Low Level | V_{OL} | $I_{OL} = 3mA$, SPI & I ² C | | | $0.2V_{DDIO}$ | - |
| Voltage Output High Level | V_{OH} | $I_{OH} = 3mA$, SPI | $0.8V_{DDIO}$ | | | - |
| Total Supply Current in Normal Mode | I_{DD} | $T_A = 25^\circ C$, ODR_{max} . $V_{DD} = V_{DDIO} = 2.4V$ | | 130 | | μA |
| Total Supply Current in Suspend Mode | I_{DDsum} | $T_A = 25^\circ C$ $V_{DD} = V_{DDIO} = 2.4V$ | | 2.1 | | μA |
| Total Supply Current in Deep Suspend Mode | I_{DDdsum} | $T_A = 25^\circ C$ $V_{DD} = V_{DDIO} = 2.4V$ | | 1 | | μA |
| Total Supply Current in Low-power Mode 1 | I_{DDlp1} | $T_A = 25^\circ C$, unfiltered $V_{DD} = V_{DDIO} = 2.4V$ sleep duration = 25ms | | 6.5 | | μA |

| | | | | | | |
|--|-------------|---|-----|-----|-----|------------------|
| Total Supply Current in Low-power Mode 2 | I_{DDIp2} | $T_A=25^\circ\text{C}$, unfiltered $V_{DD} = V_{DDIO} = 2.4\text{V}$ sleep duration = 25ms | | 66 | | μA |
| Total Supply Current in Standby Mode | I_{DDsbm} | $T_A=25^\circ\text{C}$ $V_{DD} = V_{DDIO} = 2.4\text{V}$ | | 62 | | μA |
| Wake-Up Time 1 | $t_{w,up1}$ | from Low-power Mode 1 or Suspend Mode or Deep Suspend Mode, ODR_{max} . | | 1.3 | 1.8 | ms |
| Wake-Up Time 2 | $t_{w,up2}$ | from Low-power Mode 2 or Stand-by Mode, ODR_{max} . | | 1 | 1.2 | ms |
| Start-Up Time | $t_{s,up}$ | POR, ODR_{max} . | | | 3 | ms |
| Non-volatile memory (NVM) write-cycles | n_{NVM} | | | | 15 | cycles |
| Operating Temperature | T_A | | -40 | | +85 | $^\circ\text{C}$ |

OUTPUT SIGNAL

| Parameter | Symbol | Condition | Min | Typ | Max | Units |
|---------------------------------|-------------|---|-----|-----------|-----|-------|
| Sensitivity | S_{2g} | g_{FS2g} , $T_A=25^\circ\text{C}$ | | 4096 | | LSB/g |
| | S_{4g} | g_{FS4g} , $T_A=25^\circ\text{C}$ | | 2048 | | LSB/g |
| | S_{8g} | g_{FS8g} , $T_A=25^\circ\text{C}$ | | 1024 | | LSB/g |
| | S_{16g} | g_{FS16g} , $T_A=25^\circ\text{C}$ | | 512 | | LSB/g |
| Sensitivity Temperature Drift | TCS | g_{FS2g} , Nominal V_{DD} supplies | | 0.015 | | %/K |
| Zero-g Offset | Off | g_{FS2g} , $T_A=25^\circ\text{C}$, nominal V_{DD} supplies, over life-time | | ± 50 | | mg |
| Zero-g Offset Temperature Drift | TCO | g_{FS2g} , Nominal V_{DD} supplies | | ± 1.0 | | mg/K |
| Bandwidth | bw_8 | 2 nd order filter, bandwidth programmable | | 8 | | Hz |
| | bw_{16} | | | 16 | | Hz |
| | bw_{31} | | | 31 | | Hz |
| | bw_{63} | | | 63 | | Hz |
| | bw_{125} | | | 125 | | Hz |
| | bw_{250} | | | 250 | | Hz |
| | bw_{500} | | | 500 | | Hz |
| Output Data Rate | ODR_{max} | unfiltered | | 2000 | | Hz |
| Nonlinearity | NL | best fit straight line, g_{FS2g} | | ± 0.5 | | %FS |

| | | | | | | |
|--------------------------------------|-----------|--|-----|---------|----|-------------------|
| Output Noise Density | n_{rms} | $g_{FS2g}, T_A=25^\circ C$ Nominal V_{DD} supplies Normal mode | | 120 | | $\mu g/\sqrt{Hz}$ |
| Temperature Sensor Measurement Range | T_s | | -40 | | 85 | $^\circ C$ |
| Temperature Sensor Slope | dT_s | | | 0.5 | | K/LSB |
| Temperature Sensor Offset | OT_s | | | ± 2 | | K |

MECHANICAL CHARACTERISTICS

| Parameter | Symbol | Condition | Min | Typ | Max | Units |
|------------------------|--------|---|-----|-----------|-----|----------|
| Cross Axis Sensitivity | S | relative contribution between any two of the three axes | | 1 | | % |
| Alignment Error | E_A | relative to package outline | | ± 0.5 | | $^\circ$ |

2. Absolute maximum ratings

Table 2: Absolute maximum ratings

| Parameter | Condition | Min | Max | Units |
|--|---------------------------------|------|------------------------|-------|
| Voltage at Supply Pin | V _{DD} Pin | -0.3 | 4.25 | V |
| | V _{DDIO} Pin | -0.3 | 4.25 | V |
| Voltage at any Logic Pin | Non-Supply Pin | -0.3 | V _{DDIO} +0.3 | V |
| Passive Storage Temp. Range | ≤ 65% rel. H. | -50 | +150 | °C |
| None-volatile memory (NVM) Data Retention | T = 85°C, after 15 cycles | 10 | | y |
| Mechanical Shock | Duration ≤ 200μs | | 10,000 | g |
| | Duration ≤ 1.0ms | | 2,000 | g |
| | Free fall onto hard surfaces | | 1.8 | m |
| ESD | HBM, at any Pin | | 2 | kV |
| | CDM | | 500 | V |
| | MM | | 200 | V |

Note:

Stress above these limits may cause damage to the device. Exceeding the specified electrical limits may affect the device reliability or cause malfunction.

3. Block diagram

Figure 1 shows the basic building blocks of the BMA280:

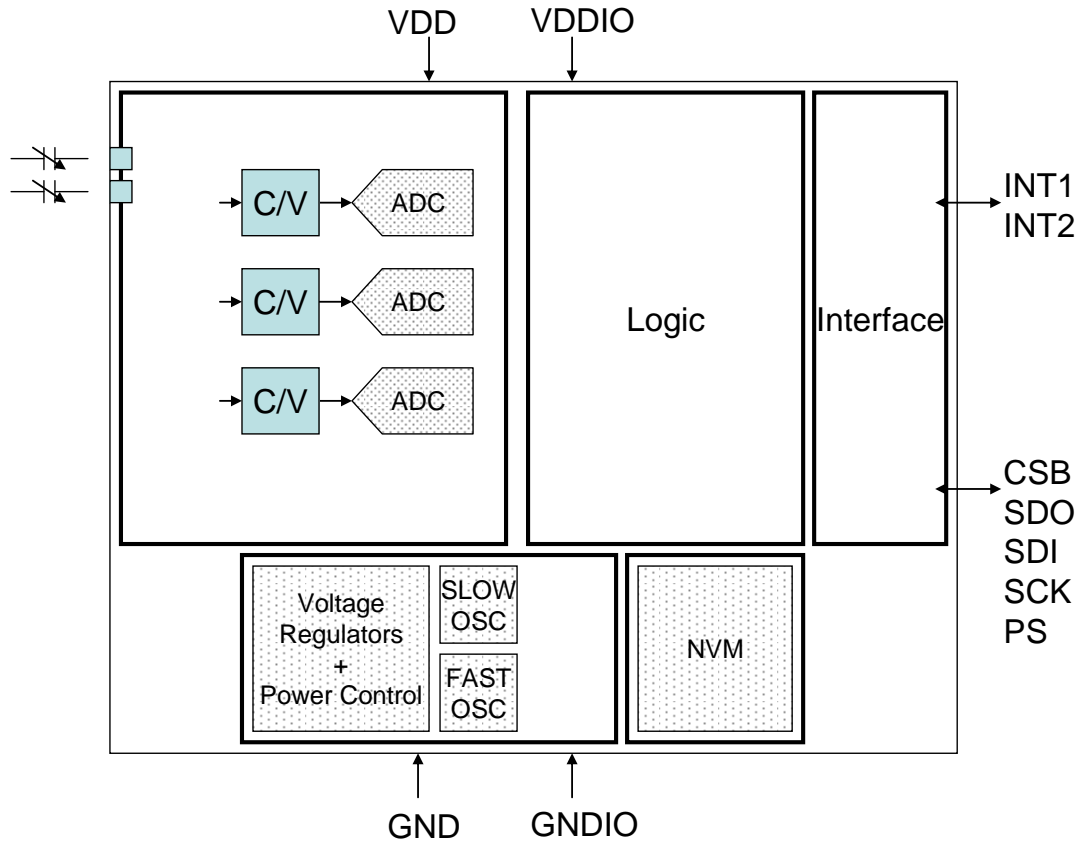


Figure 1: Block diagram of BMA280

4. Functional description

Note: Default values for registers can be found in chapter 6.

4.1 Supply voltage and power management

The BMA280 has two distinct power supply pins:

- V_{DD} is the main power supply for the internal blocks;
- V_{DDIO} is a separate power supply pin used for supplying power for the interface

There are no limitations on the voltage levels of both pins relative to each other, as long as each of them lies within its operating range. Furthermore, the device can be completely switched off ($V_{DD} = 0V$) while keeping the V_{DDIO} supply on ($V_{DDIO} > 0V$) or vice versa.

When the V_{DDIO} supply is switched off, all interface pins (CSB, SDI, SCK, PS) must be kept close to GND_{IO} potential.

The device contains a power-on reset (POR) generator. It resets the logic part and the register values after powering-on V_{DD} and V_{DDIO} . Please note, that all application specific settings which are not equal to the default settings (refer to 6.2 register map), must be re-set to its designated values after POR.

There are no constraints on the switching sequence of both supply voltages. In case the I²C interface shall be used, a direct electrical connection between V_{DDIO} supply and the PS pin is needed in order to ensure reliable protocol selection. For SPI interface mode the PS pin must be directly connected to GND_{IO} .

4.2 Power modes

The BMA280 has six different power modes. Besides normal mode, which represents the fully operational state of the device, there are five energy saving modes: deep-suspend mode, suspend mode, standby mode, low-power mode 1 and low-power mode 2.

The possible transitions between the power modes are illustrated in figure 2:

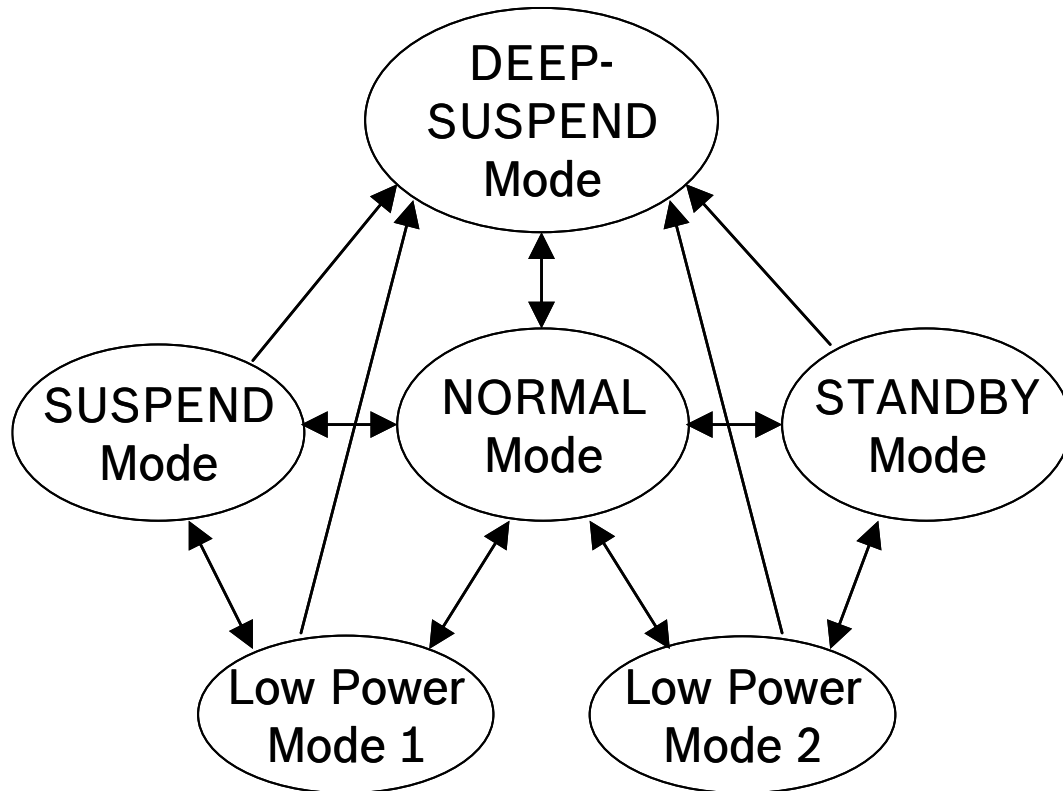


Figure 2: Power mode transition diagram

After **power-up** BMA280 is in normal mode so that all parts of the device are held powered-up and data acquisition is performed continuously.

In **deep-suspend** mode the device reaches the lowest possible power consumption. Only the interface section is kept alive. No data acquisition is performed and the content of the configuration registers is lost. Deep suspend mode is entered (left) by writing '1' ('0') to the (0x11) *deep_suspend* bit while (0x11) *suspend* bit is set to '0'. The I²C watchdog timer remains functional. The (0x11) *deep_suspend* bit, the (0x34) *spi3* bit, (0x34) *i2c_wdt_en* bit and the (0x34) *i2c_wdt_sel* bit are functional in deep-suspend mode. Equally the interrupt level and driver configuration registers (0x20) *int1_lvl*, (0x20) *int1_od*, (0x20) *int2_lvl*, and (0x20) *int2_od* are accessible. Still it is possible to enter normal mode by performing a softreset as described in chapter 4.8. Please note, that all application specific settings which are not equal to the default settings (refer to 6.2 register map), must be re-set to its designated values after leaving deep-suspend mode.

In **suspend mode** the whole analog part is powered down. No data acquisition is performed. While in suspend mode the latest acceleration data and the content of all configuration registers are kept. Writing to and reading from registers is supported except from the (0x3E) `fifo_config_1`, (0x30) `fifo_config_0` and (0x3F) `fifo_data` register. It is possible to enter normal mode by performing a softreset as described in chapter 4.8.

Suspend mode is entered (left) by writing '1' ('0') to the (0x11) suspend bit after bit (0x12) `lowpower_mode` has been set to '0'. Although write access to registers is supported at the full interface clock speed (SCL or SCK), a waiting period must be inserted between two consecutive write cycles (please refer also to section 7.2.1).

In **standby mode** the analog part is powered down, while the digital part remains largely operational. No data acquisition is performed. Reading and writing registers is supported without any restrictions. The latest acceleration data and the content of all configuration registers are kept. Standby mode is entered (left) by writing '1' ('0') to the (0x11) suspend bit after bit (0x12) `lowpower_mode` has been set to '1'. It is also possible to enter normal mode by performing a softreset as described in chapter 4.8.

In **low-power mode 1**, the device is periodically switching between a sleep phase and a wake-up phase. The wake-up phase essentially corresponds to operation in normal mode with complete power-up of the circuitry. The sleep phase essentially corresponds to operation in suspend mode. Low-power mode is entered (left) by writing '1' ('0') to the (0x11) `lowpower_en` bit with bit (0x12) `lowpower_mode` set to '0'. Read access to registers is possible except from the (0x3F) `fifo_data` register. However, unless the register access is synchronised with the wake-up phase, the restrictions of the suspend mode apply.

Low-power mode 2 is very similar to low-power mode 1, but register access is possible at any time without restrictions. It consumes more power than low-power mode 1. In low-power mode 2 the device is periodically switching between a sleep phase and a wake-up phase. The wake-up phase essentially corresponds to operation in normal mode with complete power-up of the circuitry. The sleep phase essentially corresponds to operation in standby mode. Low-power mode is entered (left) by writing '1' ('0') to the (0x11) `lowpower_en` bit with bit (0x12) `lowpower_mode` set to '1'.

The **timing behaviour** of the low-power modes 1 and 2 depends on the setting of the (0x12) `sleeptimer_mode` bit. When (0x12) `sleeptimer_mode` is set to '0', the event-driven time-base mode (EDT) is selected. In EDT the duration of the wake-up phase depends on the number of samples required by the enabled interrupt engines. If an interrupt is detected, the device stays in the wake-up phase as long as the interrupt condition endures (non-latched interrupt), or until the latch time expires (temporary interrupt), or until the interrupt is reset (latched interrupt). If no interrupt is detected, the device enters the sleep phase immediately after the required number of acceleration samples have been taken and an active interface access cycle has ended. The EDT mode is recommended for power-critical applications which do not use the FIFO. Also, EDT mode is compatible with legacy BST sensors. Figure 3 shows the timing diagram for low-power modes 1 and 2 when EDT is selected.

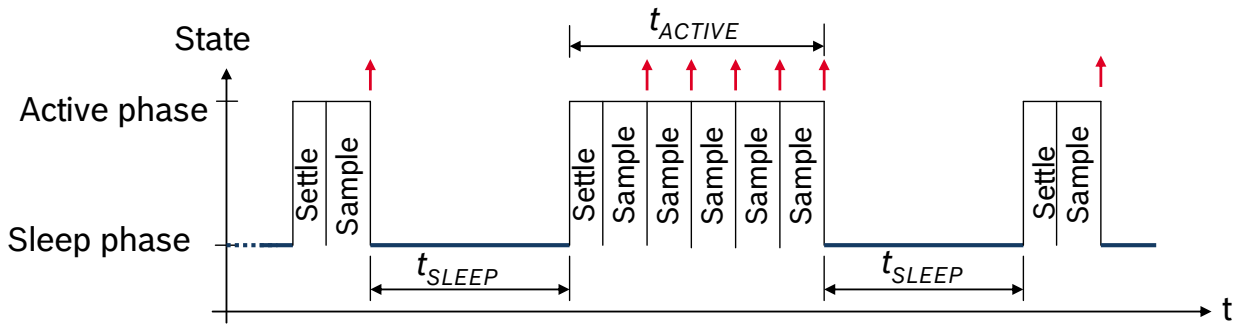


Figure 3: Timing Diagram for low-power mode 1/2, EDT

When $(0x12)$ *sleeptimer_mode* is set to '1', the equidistant-sampling mode (EST) is selected. The use of the EST mode is recommended when the FIFO is used since it ensures that equidistant samples are sampled into the FIFO regardless of whether the active phase is extended by active interrupt engines or interface activity. In EST mode the sleep time t_{SLEEP} is defined as shown in Figure 4. The FIFO sampling time t_{SAMPLE} is the sum of the sleep time t_{SLEEP} and the sensor data sampling time t_{SSMP} . Since interrupt engines can extend the active phase to exceed the sleep time t_{SLEEP} , equidistant sampling is only guaranteed if the bandwidth has been chosen such that $1/(2 * bw) = n * t_{SLEEP}$ where n is an integer. If this condition is infringed, equidistant sampling is not possible. Once the sleep time has elapsed the device will store the next available sample in the FIFO. This set-up condition is not recommended as it may result in timing jitter.

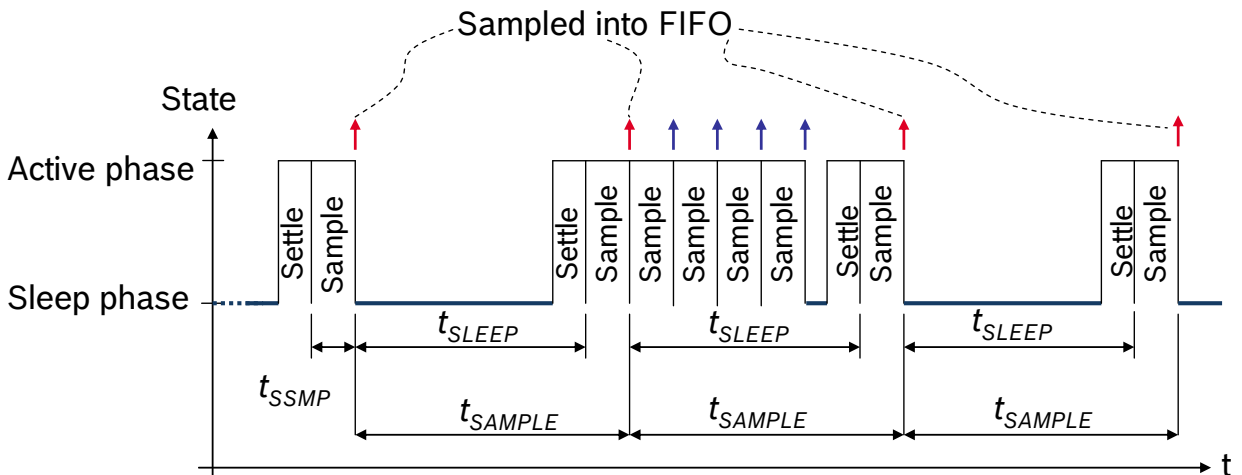


Figure 4: Timing Diagram for low-power mode 1/2, EST

The sleep time for lower-power mode 1 and 2 is set by the (0x11) *sleep_dur* bits as shown in the following table:

Table 3: Sleep phase duration settings

| (0x11) <i>sleep_dur</i> | Sleep Phase Duration t_{sleep} |
|----------------------------|--|
| 0000b | 0.5ms |
| 0001b | 0.5ms |
| 0010b | 0.5ms |
| 0011b | 0.5ms |
| 0100b | 0.5ms |
| 0101b | 0.5ms |
| 0110b | 1ms |
| 0111b | 2ms |
| 1000b | 4ms |
| 1001b | 6ms |
| 1010b | 10ms |
| 1011b | 25ms |
| 1100b | 50ms |
| 1101b | 100ms |
| 1110b | 500ms |
| 1111b | 1s |

The current consumption of the BMA280 in low-power mode 1 (I_{DDlp1}) and low-power mode 2 (I_{DDlp2}) can be estimated with the following formulae:

$$I_{DDlp1} \approx \frac{t_{sleep} \cdot I_{DDsum} + t_{active} \cdot I_{DD}}{t_{sleep} + t_{active}}$$

$$I_{DDlp2} \approx \frac{t_{sleep} \cdot I_{DDsbm} + t_{active} \cdot I_{DD}}{t_{sleep} + t_{active}}$$

When estimating the length of the wake-up phase t_{active} , the corresponding typical wake-up time, $t_{w,up1}$ or $t_{w,up2}$ and t_{ut} (given in Table 4) have to be considered:

If bandwidth is ≥ 31.25 Hz:

$$t_{active} = t_{ut} + t_{w,up1} - 0.9 \text{ ms (or } t_{active} = t_{ut} + t_{w,up2} - 0.9 \text{ ms)}$$

else:

$$t_{active} = 4 t_{ut} + t_{w,up1} - 0.9 \text{ ms (or } t_{active} = 4 t_{ut} + t_{w,up2} - 0.9 \text{ ms)}$$

During the wake-up phase all analog modules are held powered-up, while during the sleep phase most analog modules are powered down. Consequently, a wake-up time of at least $t_{w,up1}$ ($t_{w,up2}$) is needed to settle the analog modules so that reliable acceleration data are generated.

4.3 Sensor data

4.3.1 Acceleration data

The width of acceleration data is 14 bits given in two's complement representation. The 14 bits for each axis are split into an MSB upper part (one byte containing bits 13 to 6) and an LSB lower part (one byte containing bits 5 to 0 of acceleration and a (0x02, 0x04, 0x06) *new_data* flag). Reading the acceleration data registers shall always start with the LSB part. In order to ensure the integrity of the acceleration data, the content of an MSB register is locked by reading the corresponding LSB register (shadowing procedure). When shadowing is enabled, the MSB must always be read in order to remove the data lock. The shadowing procedure can be disabled (enabled) by writing '1' ('0') to the bit *shadow_dis*. With shadowing disabled, the content of both MSB and LSB registers is updated by a new value immediately. Unused bits of the LSB registers may have any value and should be ignored. The (0x02, 0x04, 0x06) *new_data* flag of each LSB register is set if the data registers have been updated. The flag is reset if either the corresponding MSB or LSB part is read.

Two different streams of acceleration data are available, unfiltered and filtered. The unfiltered data is sampled with 2kHz. The sampling rate of the filtered data depends on the selected filter bandwidth and is always twice the selected bandwidth ($BW = ODR/2$). Which kind of data is stored in the acceleration data registers depends on bit (0x13) *data_high_bw*. If (0x13) *data_high_bw* is '0' ('1'), then filtered (unfiltered) data is stored in the registers. Both data streams are offset-compensated.

The bandwidth of filtered acceleration data is determined by setting the (0x10) *bw* bit as followed:

Table 4: Bandwidth configuration

| bw | Bandwidth | Update Time t_{ut} |
|-----------|------------------|--|
| 00xxx | *) | - |
| 01000 | 7.81Hz | 64ms |
| 01001 | 15.63Hz | 32ms |
| 01010 | 31.25Hz | 16ms |
| 01011 | 62.5Hz | 8ms |
| 01100 | 125Hz | 4ms |
| 01101 | 250Hz | 2ms |
| 01110 | 500Hz | 1ms |
| 01111 | unfiltered | 0.5ms |
| 1xxxx | *) | - |

*) Note: Settings 00xxx result in a bandwidth of 7.81 Hz; settings 1xxxx result in ODR_{max} (unfiltered signal). It is recommended to actively set an application specific and an appropriate bandwidth and to use the range from '01000b' to '01111b' only in order to be compatible with future products.

The BMA280 supports four different acceleration measurement ranges. A measurement range is selected by setting the *(0x0F)* range bits as follows:

Table 5: Range selection

| Range | Acceleration measurement range | Resolution |
|--------|--------------------------------|-------------|
| 0011 | ±2g | 0.244mg/LSB |
| 0101 | ±4g | 0.488mg/LSB |
| 1000 | ±8g | 0.977mg/LSB |
| 1100 | ±16g | 1.953mg/LSB |
| others | reserved | - |

4.3.2 Temperature sensor

The width of temperature data is 8 bits given in two's complement representation. Temperature values are available in the *(0x08) temp* register.

The slope of the temperature sensor is 0.5K/LSB, its center temperature is 23°C [*(0x08) temp* = 0x00].

4.4 Self-test

This feature permits to check the sensor functionality by applying electrostatic forces to the sensor core instead of external accelerations. By actually deflecting the seismic mass, the entire signal path of the sensor can be tested. Activating the self-test results in a static offset of the acceleration data; any external acceleration or gravitational force applied to the sensor during active self-test will be observed in the output as a superposition of both acceleration and self-test signal.

Before the self-test is enabled the g-range should be set to 4 g. The self-test is activated individually for each axis by writing the proper value to the (0x32) *self_test_axis* bits ('01b' for x-axis, '10b' for y-axis, '11b' for z-axis, '00b' to deactivate self-test). It is possible to control the direction of the deflection through bit (0x32) *self_test_sign*. The excitation occurs in negative (positive) direction if (0x32) *self_test_sign* = '0b' ('1b'). After the self-test is enabled, the user should wait 50ms before interpreting the acceleration data.

In order to ensure a proper interpretation of the self-test signal it is recommended to perform the self-test for both (positive and negative) directions and then to calculate the difference of the resulting acceleration values. Table 6 shows the minimum differences for each axis. The actually measured signal differences can be significantly larger.

Table 6: Self-test difference values

| | x-axis signal | y-axis signal | z-axis signal |
|-------------------------------------|----------------------|----------------------|----------------------|
| resulting minimum difference signal | 800 mg | 800 mg | 400 mg |

It is recommended to perform a reset of the device after a self-test has been performed. If the reset cannot be performed, the following sequence must be kept to prevent unwanted interrupt generation: disable interrupts, change parameters of interrupts, wait for at least 50ms, enable desired interrupts.

4.5 Offset compensation

Offsets in measured signals can have several causes but they are always unwanted and disturbing in many cases. Therefore, the BMA280 offers an advanced set of four digital offset compensation methods which are closely matched to each other. These are slow, fast, and manual compensation as well as inline calibration.

The compensation is performed with unfiltered data, and is then applied to both, unfiltered and filtered data. If necessary the result of this computation is saturated to prevent any overflow errors (the smallest or biggest possible value is set, depending on the sign). However, the registers used to read and write compensation values have a width of 8 bits.

An overview of the offset compensation principle is given in figure 5:

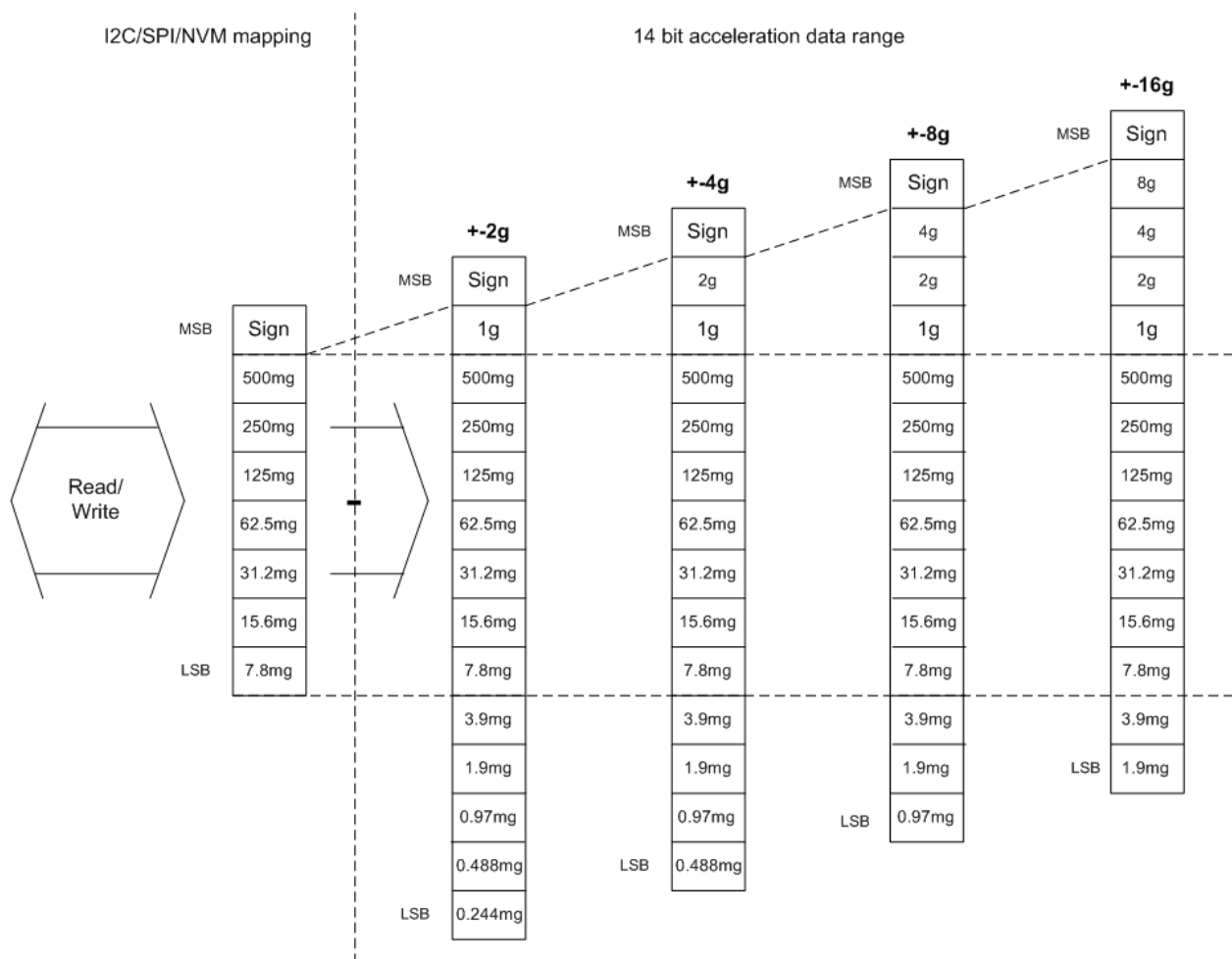


Figure 5: Principle of offset compensation

The public offset compensation registers (*0x38*) *offset_x*, (*0x39*) *offset_y*, (*0x3A*) *offset_z* are images of the corresponding registers in the NVM. With each image update (see section 4.6 Non-volatile memory for details) the contents of the NVM registers are written to the public registers. The public registers can be over-written by the user at any time. After changing the contents of the public registers by either an image update or manually, all 8bit values are extended to 14bit values for internal computation. In the opposite direction, if an internally computed value changes it is converted to an 8bit value and stored in the public register.

Depending on the selected g-range the conversion from 14bit to 8bit values can result in a loss of accuracy of one to several LSB. This is shown in figure 5.

In case an internally computed compensation value is too small or too large to fit into the corresponding register, it is saturated in order to prevent an overflow error.

By writing '1' to the (*0x36*) *offset_reset* bit, all offset compensation registers are reset to zero.

4.5.1 Slow compensation

Slow compensation is based on a 1st order high-pass filter, which continuously drives the average value of the output data stream of each axis to zero. The bandwidth of the high-pass filter is configured with bit (0x37) *cut_off* according to Table 7.

Table 7: Compensation period settings

| (0x37) <i>cut_off</i> | high-pass filter bandwidth | Example <i>bw</i> = 500 Hz |
|--------------------------|--|--|
| 0b | $\frac{1\text{Hz} \times bw}{1000\text{ Hz}}$ | $\frac{1\text{Hz} \times 500\text{ Hz}}{1000\text{ Hz}} = 0.5\text{ Hz}$ |
| 1b | $\frac{10\text{Hz} \times bw}{1000\text{ Hz}}$ | $\frac{10\text{Hz} \times 500\text{ Hz}}{1000\text{ Hz}} = 5\text{ Hz}$ |

**bw*: please insert selected decimal data bandwidth value [Hz] from table 4

The slow compensation can be enabled (disabled) for each axis independently by setting the bits (0x36) *hp_x_en*, *hp_y_en*, *hp_z_en* to '1' ('0'), respectively.

Slow compensation should not be used in combination with low-power mode. In low-power mode the conditions (availability of necessary data) for proper function of slow compensation are not fulfilled.

4.5.2 Fast compensation

Fast compensation is a one-shot process by which the compensation value is set in such a way that when added to the raw acceleration, the resulting acceleration value of each axis approaches the target value. This is best suited for “end-of-line trimming” with the customer’s device positioned in a well-defined orientation. For fast compensation the g-range has to be switched to 2g.

The algorithm in detail: An average of 16 consecutive acceleration values is computed and the difference between target value and computed value is written to (0x38, 0x39, 0x3A) *offset_filt_x/y/z*. The public registers (0x38, 0x39, 0x3A) *offset_filt_x/y/z* are updated with the contents of the internal registers (using saturation if necessary) and can be read by the user.

Fast compensation is triggered for each axis individually by setting the (0x36) *cal_trigger* bits as shown in Table 8:

Table 8: Fast compensation axis selection

| (0x36) <i>cal_trigger</i> | Selected Axis |
|------------------------------|---------------|
| 00b | none |
| 01b | x |
| 10b | y |
| 11b | z |

Register (0x36) *cal_trigger* is a write-only register. Once triggered, the status of the fast correction process is reflected in the status bit (0x36) *cal_rdy*. Bit (0x36) *cal_rdy* is '0' while the correction is in progress. Otherwise it is '1'. Bit (0x36) *cal_rdy* is '0' when (0x36) *cal_trigger* is not '00'.

For the fast offset compensation, the compensation target can be chosen by setting the bits (0x37) *offset_target_x*, (0x37) *offset_target_y*, and (0x37) *offset_target_z* according to Table 9:

Table 9: Offset target settings

| (0x37) <i>offset_target_x/y/z</i> | Target value |
|--------------------------------------|--------------|
| 00b | 0g |
| 01b | +1g |
| 10b | -1g |
| 11b | 0g |

Fast compensation should not be used in combination with any of the low-power modes. In low-power mode the conditions (availability of necessary data) for proper function of fast compensation are not fulfilled.

4.5.3 Manual compensation

The contents of the public compensation registers (0x38, 0x39, 0x3A) *offset_filt_x/y/z* can be set manually via the digital interface. It is recommended to write into these registers directly after a new data interrupt has occurred in order not to disturb running offset computations.

Writing to the offset compensation registers is not allowed while the fast compensation procedure is running.

4.5.4 Inline calibration

For certain applications, it is often desirable to calibrate the offset once and to store the compensation values permanently. This can be achieved by using one of the aforementioned offset compensation methods to determine the proper compensation values and then storing these values permanently in the NVM. See section 4.6 Non-volatile memory for details of the storing procedure.

Each time the device is reset, the compensation values are loaded from the non-volatile memory into the image registers and used for offset compensation until they are possibly overwritten using one of the other compensation methods.

4.6 Non-volatile memory

The entire memory of the BMA280 consists of three different kinds of registers: hard-wired, volatile, and non-volatile. Part of it can be both read and written by the user. Access to non-volatile memory is only possible through (volatile) image registers.

Altogether, there are eight registers (octets) with NVM backup which are accessible by the user. The addresses of the image registers range from 0x38 to 0x3C. While the addresses up to 0x3A are used for offset compensation (see 4.4 Offset Compensation), addresses 0x3B and 0x3C are general purpose registers not linked to any sensor-specific functionality.

The content of the NVM is loaded to the image registers after a reset (either POR or softreset) or after a user request which is performed by writing '1' to the write-only bit (0x33) *nvm_load*. As long as the image update is in progress, bit (0x33) *nvm_rdy* is '0', otherwise it is '1'.

The image registers can be read and written like any other register.

Writing to the NVM is a three-step procedure:

1. Write the new contents to the image registers.
2. Write '1' to bit (0x33) *nvm_prog_mode* in order to unlock the NVM.
3. Write '1' to bit (0x33) *nvm_prog_trig* and keep '1' in bit (0x33) *nvm_prog_mode* in order to trigger the write process.

Writing to the NVM always renews the entire NVM contents. It is possible to check the write status by reading bit (0x33) *nvm_rdy*. While (0x33) *nvm_rdy* = '0', the write process is still in progress; if (0x33) *nvm_rdy* = '1', then writing is completed. As long as the write process is ongoing, no change of power mode and image registers is allowed. Also, the NVM write cycle must not be initiated while image registers are updated, in low-power mode, and in suspend mode.

Please note that the number of permitted NVM write-cycles is limited as specified in Table 1. The number of remaining write-cycles can be obtained by reading bits (0x33) *nvm_remain*.

4.7 Interrupt controller

The BMA280 is equipped with eight programmable interrupt engines. Each interrupt can be independently enabled and configured. If the trigger condition of an enabled interrupt is fulfilled, the corresponding status bit is set to '1' and the selected interrupt pin is activated. The BMA280 provides two interrupt pins, INT1 and INT2; interrupts can be freely mapped to any of these pins. The state of a specific interrupt pin is derived from a logic 'or' combination of all interrupts mapped to it.

The interrupt status registers are updated when a new data word is written into the acceleration data registers. If an interrupt is disabled, all active status bits associated with it are immediately reset.

4.7.1 General features

An interrupt is cleared depending on the selected interrupt mode, which is common to all interrupts. There are three different interrupt modes: non-latched, latched, and temporary. The mode is selected by the (0x21) *latch_int* bits according to Table 10.

Table 10: Interrupt mode selection

| (0x21) <i>latch_int</i> | Interrupt mode |
|----------------------------|-------------------|
| 0000b | non-latched |
| 0001b | temporary, 250ms |
| 0010b | temporary, 500ms |
| 0011b | temporary, 1s |
| 0100b | temporary, 2s |
| 0101b | temporary, 4s |
| 0110b | temporary, 8s |
| 0111b | latched |
| 1000b | non-latched |
| 1001b | temporary, 250µs |
| 1010b | temporary, 500µs |
| 1011b | temporary, 1ms |
| 1100b | temporary, 12.5ms |
| 1101b | temporary, 25ms |
| 1110b | temporary, 50ms |
| 1111b | latched |

An interrupt is generated if its activation condition is met. It can not be cleared as long as the activation condition is fulfilled. In the non-latched mode the interrupt status bit and the selected pin (the contribution to the 'or' condition for INT1 and/or INT2) are cleared as soon as the activation condition is no more valid. Exceptions to this behavior are the new data, orientation, and flat interrupts, which are automatically reset after a fixed time.

In latched mode an asserted interrupt status and the selected pin are cleared by writing '1' to bit (0x21) *reset_int*. If the activation condition still holds when it is cleared, the interrupt status is asserted again with the next change of the acceleration registers.

In the temporary mode an asserted interrupt and selected pin are cleared after a defined period of time. The behaviour of the different interrupt modes is shown graphically in figure 6. The timings in this mode are subject to the same tolerances as the bandwidths (see Table 1).

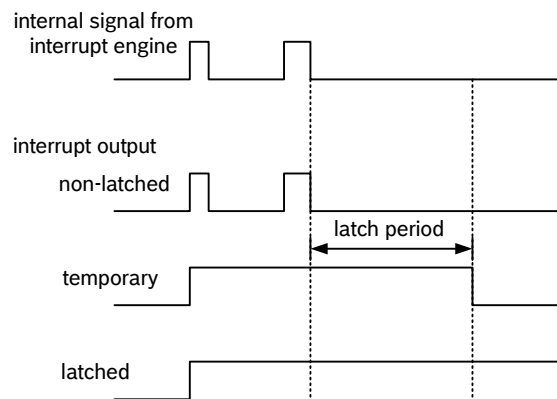


Figure 6: Interrupt modes

Several interrupt engines can use either unfiltered or filtered acceleration data as their input. For these interrupts, the source can be selected with the bits in register (0x1E). These are (0x1E) *int_src_data*, (0x1E) *int_src_tap*, (0x1E) *int_src_slo_no_mot*, (0x1E) *int_src_slope*, (0x1E) *int_src_high*, and (0x1E) *int_src_low*. Setting the respective bits to '0' ('1') selects filtered (unfiltered) data as input. The orientation recognition and flat detection interrupt always use filtered input data.

It is strongly recommended to set interrupt parameters prior to enabling the interrupt. Changing parameters of an already enabled interrupt may cause unwanted interrupt generation and generation of a false interrupt history. A safe way to change parameters of an enabled interrupt is to keep the following sequence: disable the desired interrupt, change parameters, wait for at least 10ms, and then re-enable the desired interrupt.

4.7.2 Mapping to physical interrupt pins (inttype to INT Pin#)

Registers (0x19) to (0x1B) are dedicated to mapping of interrupts to the interrupt pins "INT1" or "INT2". Setting (0x19) *int1_*"inttype" to '1' ('0') maps (unmaps) "inttype" to pin "INT1". Correspondingly setting (0x1B) *int2_*"inttype" to '1' ('0') maps (unmaps) "inttype" to pin "INT2".

Note: "inttype" to be replaced with the precise notation, given in the memory map in chapter 6.

Example: For flat interrupt (*int1_flat*): Setting (0x19) *int1_flat* to '1' maps *int1_flat* to pin "INT1".

4.7.3 Electrical behaviour (INT pin# to open-drive or push-pull)

Both interrupt pins can be configured to show the desired electrical behaviour. The 'active' level of each interrupt pin is determined by the $(0x20)$ *int1_lvl* and $(0x20)$ *int2_lvl* bits.

If $(0x20)$ *int1_lvl* = '1' ('0') / $(0x20)$ *int2_lvl* = '1' ('0'), then pin "INT1" / pin "INT2" is active '1' ('0'). The characteristic of the output driver of the interrupt pins may be configured with bits $(0x20)$ *int1_od* and $(0x20)$ *int2_od*. By setting bits $(0x20)$ *int1_od* / $(0x20)$ *int2_od* to '1', the output driver shows open-drive characteristic, by setting the configuration bits to '0', the output driver shows push-pull characteristic. When open-drive characteristic is selected in the design, external pull-up or pull-down resistor should be applied according the *int_lvl* configuration.

4.7.4 New data interrupt

This interrupt serves for synchronous reading of acceleration data. It is generated after storing a new value of z-axis acceleration data in the data register. The interrupt is cleared automatically when the next data acquisition cycle starts. The interrupt status is '0' for at least 50µs.

The interrupt mode of the new data interrupt is fixed to non-latched.

It is enabled (disabled) by writing '1' ('0') to bit $(0x17)$ *data_en*. The interrupt status is stored in bit $(0x0A)$ *data_int*.

Due to the settling time of the filter, the first interrupt after wake-up from suspend or standby mode will take longer than the update time.

4.7.5 Slope / any-motion detection

Slope / any-motion detection uses the slope between successive acceleration signals to detect changes in motion. An interrupt is generated when the slope (absolute value of acceleration difference) exceeds a preset threshold. It is cleared as soon as the slope falls below the threshold. The principle is made clear in figure 7.

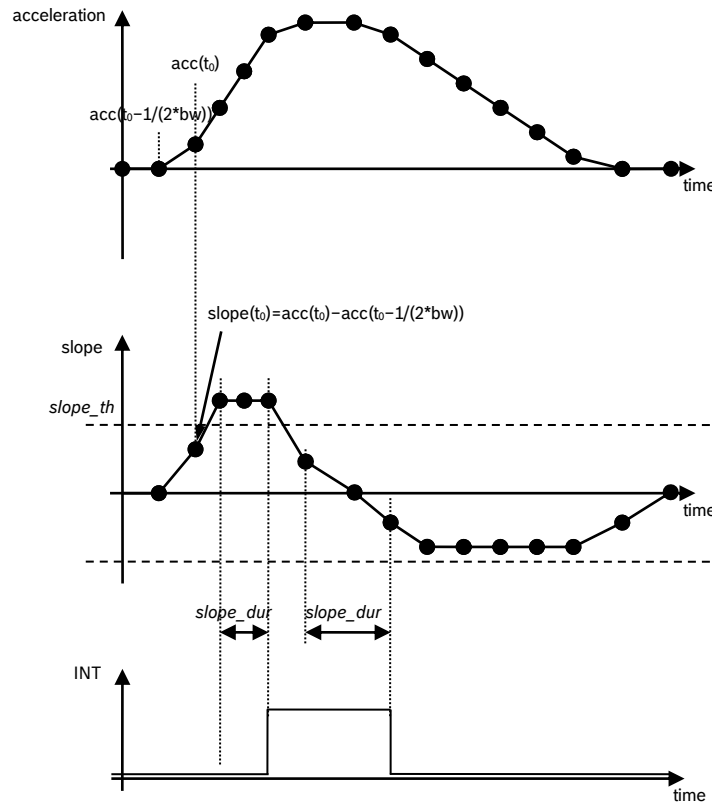


Figure 7: Principle of any-motion detection

The threshold is defined through register (0x28) *slope_th*. In terms of scaling 1 LSB of (0x28) *slope_th* corresponds to 3.91 mg in 2g-range (7.81 mg in 4g-range, 15.6 mg in 8g-range and 31.3 mg in 16g-range). Therefore the maximum value is 996 mg in 2g-range (1.99g in 4g-range, 3.98g in 8g-range and 7.97g in 16g-range).

The time difference between the successive acceleration signals depends on the selected bandwidth and equates to $1/(2 \cdot \text{bandwidth})$ ($t = 1/(2 \cdot bw)$). In order to suppress false triggers, the interrupt is only generated (cleared) if a certain number N of consecutive slope data points is larger (smaller) than the slope threshold given by (0x28) *slope_th*. This number is set by the (0x27) *slope_dur* bits. It is $N = (0x27) \text{ slope_dur} + 1$ for (0x27).

Example: (0x27) *slope_dur* = 00b, ..., 11b = 1decimal, ..., 4decimal.

4.7.5.1 Enabling (disabling) for each axis

Any-motion detection can be enabled (disabled) for each axis separately by writing '1' ('0') to bits (0x16) *slope_en_x*, (0x16) *slope_en_y*, (0x16) *slope_en_z*. The criteria for any-motion detection are fulfilled and the slope interrupt is generated if the slope of any of the enabled axes exceeds the threshold (0x28) *slope_th* for [(0x27) *slope_dur* +1] consecutive times. As soon as the slopes of all enabled axes fall or stay below this threshold for [(0x27) *slope_dur* +1] consecutive times the interrupt is cleared unless interrupt signal is latched.

4.7.5.2 Axis and sign information of slope / any motion interrupt

The interrupt status is stored in bit (0x09) *slope_int*. The any-motion interrupt supplies additional information about the detected slope. The axis which triggered the interrupt is given by that one of bits (0x0B) *slope_first_x*, (0x0B) *slope_first_y*, (0x0B) *slope_first_z* that contains a value of '1'. The sign of the triggering slope is held in bit (0x0B) *slope_sign* until the interrupt is retriggered. If (0x0B) *slope_sign* = '0' ('1'), the sign is positive (negative).

4.7.6 Tap sensing

Tap sensing has a functional similarity with a common laptop touch-pad or clicking keys of a computer mouse. A tap event is detected if a pre-defined slope of the acceleration of at least one axis is exceeded. Two different tap events are distinguished: A 'single tap' is a single event within a certain time, followed by a certain quiet time. A 'double tap' consists of a first such event followed by a second event within a defined time frame.

Single tap interrupt is enabled (disabled) by writing '1' ('0') to bit (0x16) *s_tap_en*. Double tap interrupt is enabled (disabled) by writing '1' ('0') to bit (0x16) *d_tap_en*.

While temporary latching is used do not simultaneously enable single tap interrupt and double tap interrupt.

The status of the single tap interrupt is stored in bit (0x09) *s_tap_int*, the status of the double tap interrupt is stored in bit (0x09) *d_tap_int*.

The slope threshold for detecting a tap event is set by bits (0x2B) *tap_th*. The meaning of (0x2B) *tap_th* depends on the range setting. 1 LSB of (0x2B) *tap_th* corresponds to a slope of 62.5mg in 2g-range, 125mg in 4g-range, 250mg in 8g-range, and 500mg in 16g-range.

In figure 8 the meaning of the different timing parameters is visualized:

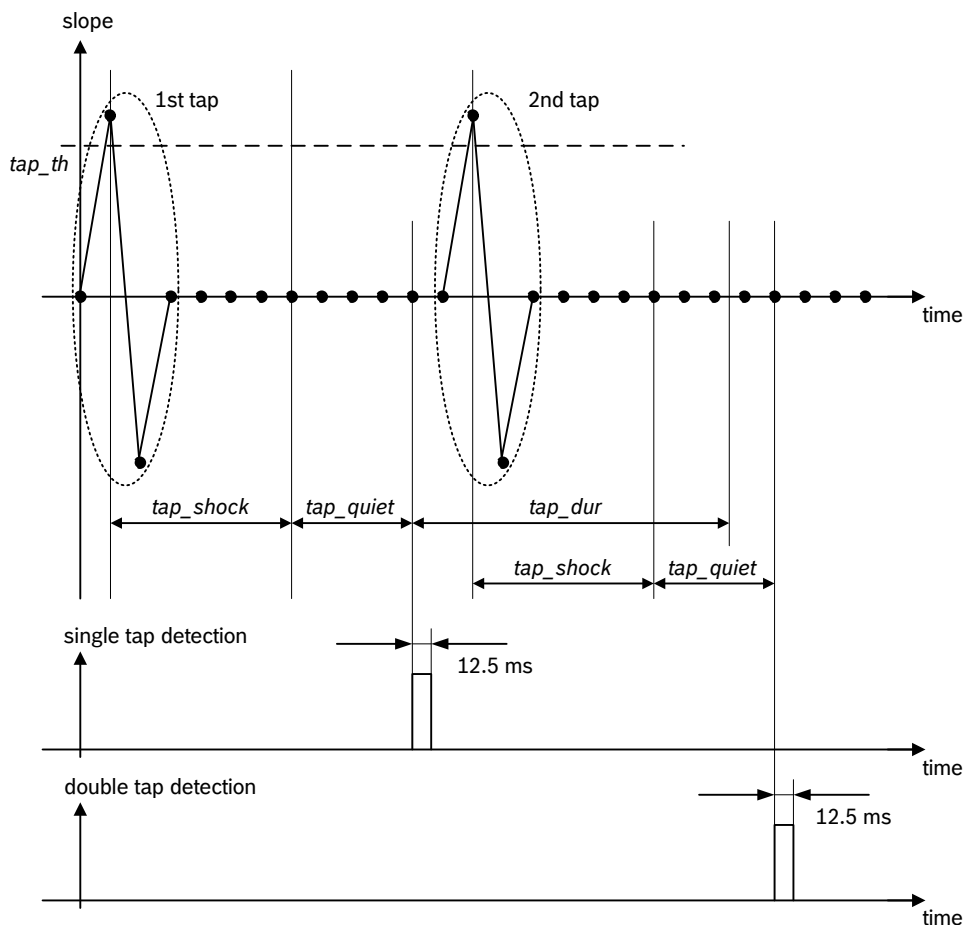


Figure 8: Timing of tap detection

The parameters $(0x2A)$ *tap_shock* and $(0x2A)$ *tap_quiet* apply to both single tap and double tap detection, while $(0x2A)$ *tap_dur* applies to double tap detection only. Within the duration of $(0x2A)$ *tap_shock* any slope exceeding $(0x2B)$ *tap_th* after the first event is ignored. Contrary to this, within the duration of $(0x2A)$ *tap_quiet* no slope exceeding $(0x2B)$ *tap_th* must occur, otherwise the first event will be cancelled.

4.7.6.1 Single tap detection

A single tap is detected and the single tap interrupt is generated after the combined durations of $(0x2A)$ *tap_shock* and $(0x2A)$ *tap_quiet*, if the corresponding slope conditions are fulfilled. The interrupt is cleared after a delay of 12.5 ms.

Do not map single-tap to any INT pin if you do not want to use it.

4.7.6.2 Double tap detection

A double tap interrupt is generated if an event fulfilling the conditions for a single tap occurs within the set duration in $(0x2A)$ *tap_dur* after the completion of the first tap event. The interrupt is automatically cleared after a delay of 12.5 ms.

4.7.6.3 Selecting the timing of tap detection

For each of parameters $(0x2A)$ *tap_shock* and $(0x2A)$ *tap_quiet* two values are selectable. By writing '0' ('1') to bit $(0x2A)$ *tap_shock* the duration of $(0x2A)$ *tap_shock* is set to 50 ms (75 ms). By writing '0' ('1') to bit $(0x2A)$ *tap_quiet* the duration of $(0x2A)$ *tap_quiet* is set to 30 ms (20 ms).

The length of $(0x2A)$ *tap_dur* can be selected by setting the $(0x2A)$ *tap_dur* bits according to Table 11:

Table 11: Selection of *tap_dur*

| $(0x2A)$ <i>tap_dur</i> | length of <i>tap_dur</i> |
|---|---------------------------------|
| 000b | 50 ms |
| 001b | 100 ms |
| 010b | 150 ms |
| 011b | 200 ms |
| 100b | 250 ms |
| 101b | 375 ms |
| 110b | 500 ms |
| 111b | 700 ms |

4.7.6.4 Axis and sign information of tap sensing

The sign of the slope of the first tap which triggered the interrupt is stored in bit $(0x0B)$ *tap_sign* ('0' means positive sign, '1' means negative sign). The value of this bit persists after clearing the interrupt.

The axis which triggered the interrupt is indicated by bits $(0x0B)$ *tap_first_x*, $(0x0B)$ *tap_first_y*, and $(0x0B)$ *tap_first_z*.

The bit corresponding to the triggering axis contains a '1' while the other bits hold a '0'. These bits are cleared together with clearing the interrupt status.

4.7.6.5 Tap sensing in low power mode

In low-power mode, a limited number of samples is processed after wake-up to decide whether an interrupt condition is fulfilled. The number of samples is selected by bits $(0x2B)$ *tap_samp* according to Table 12.

Table 12: Meaning of $(0x2B)$ *tap_samp*

| $(0x2B)$ <i>tap_samp</i> | Number of Samples |
|-----------------------------|-------------------|
| 00b | 2 |
| 01b | 4 |
| 10b | 8 |
| 11b | 16 |

4.7.7 Orientation recognition

The orientation recognition feature informs on an orientation change of the sensor with respect to the gravitational field vector 'g'. The measured acceleration vector components with respect to the gravitational field are defined as shown in figure 9.

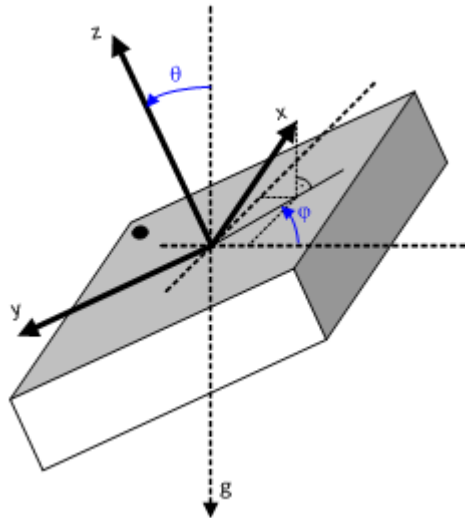


Figure 9: Definition of vector components

Therefore, the magnitudes of the acceleration vectors are calculated as follows:

$$\begin{aligned} \text{acc_x} &= 1g \times \sin\theta \times \cos\varphi \\ \text{acc_y} &= -1g \times \sin\theta \times \sin\varphi \\ \text{acc_z} &= 1g \times \cos\theta \\ \text{acc_y}/\text{acc_x} &= -\tan\varphi \end{aligned}$$

Depending on the magnitudes of the acceleration vectors the orientation of the device in the space is determined and stored in the three (0x0C) *orient* bits. These bits may not be reset in the sleep phase of low-power mode. There are three orientation calculation modes with different thresholds for switching between different orientations: symmetrical, high-asymmetrical, and low-asymmetrical. The mode is selected by setting the (0x2C) *orient_mode* bits as given in Table 13.

Table 13: Orientation mode settings

| (0x2C) <i>orient_mode</i> | Orientation Mode |
|------------------------------|-------------------|
| 00b | symmetrical |
| 01b | high-asymmetrical |
| 10b | low-asymmetrical |
| 11b | symmetrical |

For each orientation mode the $(0x0C)$ *orient* bits have a different meaning as shown in Table 14 to Table 16:

Table 14: Meaning of the $(0x0C)$ *orient* bits in symmetrical mode

| $(0x0C)$ <i>orient</i> | Name | Angle | Condition |
|---------------------------|----------------------|-----------------------------------|--|
| x00 | portrait upright | $315^\circ < \varphi < 45^\circ$ | $ \text{acc}_y < \text{acc}_x - \text{'hyst'}$ and $\text{acc}_x - \text{'hyst'} \geq 0$ |
| x01 | portrait upside down | $135^\circ < \varphi < 225^\circ$ | $ \text{acc}_y < \text{acc}_x - \text{'hyst'}$ and $\text{acc}_x + \text{'hyst'} < 0$ |
| x10 | landscape left | $45^\circ < \varphi < 135^\circ$ | $ \text{acc}_y \geq \text{acc}_x + \text{'hyst'}$ and $\text{acc}_y < 0$ |
| x11 | landscape right | $225^\circ < \varphi < 315^\circ$ | $ \text{acc}_y \geq \text{acc}_x + \text{'hyst'}$ and $\text{acc}_y \geq 0$ |

Table 15: Meaning of the $(0x0C)$ *orient* bits in high-asymmetrical mode

| $(0x0C)$ <i>orient</i> | Name | Angle | Condition |
|---------------------------|----------------------|-----------------------------------|--|
| x00 | portrait upright | $297^\circ < \varphi < 63^\circ$ | $ \text{acc}_y < 2 \cdot \text{acc}_x - \text{'hyst'}$ and $\text{acc}_x - \text{'hyst'} \geq 0$ |
| x01 | portrait upside down | $117^\circ < \varphi < 243^\circ$ | $ \text{acc}_y < 2 \cdot \text{acc}_x - \text{'hyst'}$ and $\text{acc}_x + \text{'hyst'} < 0$ |
| x10 | landscape left | $63^\circ < \varphi < 117^\circ$ | $ \text{acc}_y \geq 2 \cdot \text{acc}_x + \text{'hyst'}$ and $\text{acc}_y < 0$ |
| x11 | landscape right | $243^\circ < \varphi < 297^\circ$ | $ \text{acc}_y \geq 2 \cdot \text{acc}_x + \text{'hyst'}$ and $\text{acc}_y \geq 0$ |

Table 16: Meaning of the $(0x0C)$ *orient* bits in low-asymmetrical mode

| $(0x0C)$ <i>orient</i> | Name | Angle | Condition |
|---------------------------|----------------------|-----------------------------------|--|
| x00 | portrait upright | $333^\circ < \varphi < 27^\circ$ | $ \text{acc}_y < 0.5 \cdot \text{acc}_x - \text{'hyst'}$ and $\text{acc}_x - \text{'hyst'} \geq 0$ |
| x01 | portrait upside down | $153^\circ < \varphi < 207^\circ$ | $ \text{acc}_y < 0.5 \cdot \text{acc}_x - \text{'hyst'}$ and $\text{acc}_x + \text{'hyst'} < 0$ |
| x10 | landscape left | $27^\circ < \varphi < 153^\circ$ | $ \text{acc}_y \geq 0.5 \cdot \text{acc}_x + \text{'hyst'}$ and $\text{acc}_y < 0$ |
| x11 | landscape right | $207^\circ < \varphi < 333^\circ$ | $ \text{acc}_y \geq 0.5 \cdot \text{acc}_x + \text{'hyst'}$ and $\text{acc}_y \geq 0$ |

In the preceding tables, the parameter 'hyst' stands for a hysteresis, which can be selected by setting the $(0x2C)$ *orient_hyst* bits. 1 LSB of $(0x2C)$ *orient_hyst* always corresponds to 62.5 mg, in any g-range (i.e. increment is independent from g-range setting). It is important to note that by using a hysteresis $\neq 0$ the actual switching angles become different from the angles given in the tables since there is an overlap between the different orientations.

The most significant bit of the (0x0C) *orient* bits (which is displayed as an 'x' in the above given tables) contains information about the direction of the z-axis. It is set to '0' ('1') if $acc_z \geq 0$ ($acc_z < 0$).

Figure 10 shows the typical switching conditions between the four different orientations for the symmetrical mode i.e. without hysteresis:

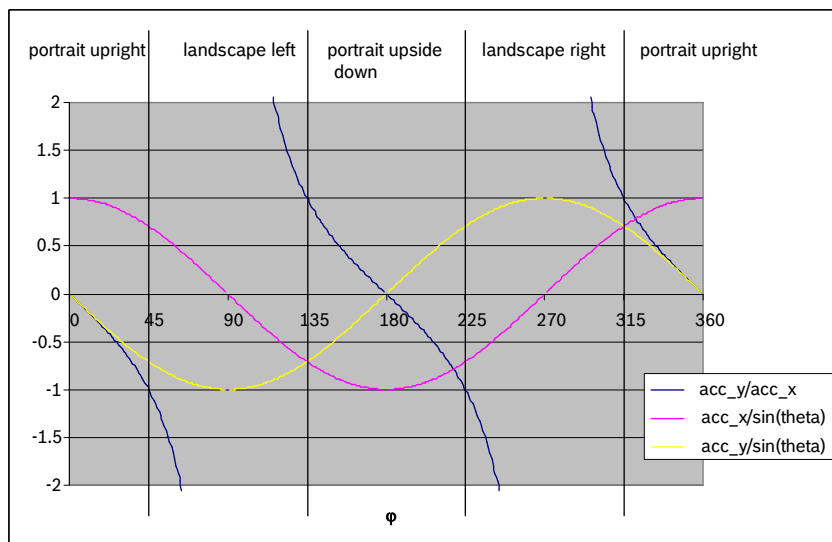


Figure 10: Typical orientation switching conditions w/o hysteresis

The orientation interrupt is enabled (disabled) by writing '1' ('0') to bit (0x16) *orient_en*. The interrupt is generated if the value of (0x0C) *orient* has changed. It is automatically cleared after one stable period of the (0x0C) *orient* value. The interrupt status is stored in the (0x09) *orient_int* bit. The register (0x0C) *orient* always reflects the current orientation of the device, irrespective of which interrupt mode has been selected. Bit (0x0C) *orient*<2> reflects the device orientation with respect to the z-axis. The bits (0x0C) *orient*<1:0> reflect the device orientation in the x-y-plane. The conventions associated with register (0x0C) *orient* are detailed in chapter 6.

4.7.7.1 Orientation blocking

The change of the (0x0C) *orient* value and – as a consequence – the generation of the interrupt can be blocked according to conditions selected by setting the value of the (0x2C) *orient_blocking* bits as described by Table 17.

Table 17: Blocking conditions for orientation recognition

| (0x2C) <i>orient_blocking</i> | Conditions |
|----------------------------------|---|
| 00b | no blocking |
| 01b | theta blocking or acceleration in any axis > 1.5g |
| 10b | theta blocking or acceleration slope in any axis > 0.2 g or acceleration in any axis > 1.5g |
| 11b | theta blocking or acceleration slope in any axis > 0.4 g or acceleration in any axis > 1.5g and value of orient is not stable for at least 100 ms |

The theta blocking is defined by the following inequality:

$$|\tan \theta| < \frac{\sqrt{blocking_theta}}{8}$$

The parameter *blocking_theta* of the above given equation stands for the contents of the (0x2D) *orient_theta* bits. It is possible to define a blocking angle between 0° and 44.8°. The internal blocking algorithm saturates the acceleration values before further processing. As a consequence, the blocking angles are strictly valid only for a device at rest; they can be different if the device is moved.

Example:

To get a maximum blocking angle of 19° the parameter *blocking_theta* is determined in the following way: $(8 * \tan(19^\circ))^2 = 7.588$, therefore, *blocking_value* = 8dec = 001000b has to be chosen.

In order to avoid unwanted generation of the orientation interrupt in a nearly flat position ($z \sim 0$, sign change due to small movements or noise), a hysteresis of 0.2 g is implemented for the z-axis, i. e. a after a sign change the interrupt is only generated after $|z| > 0.2$ g.

4.7.7.2 Up-Down Interrupt Suppression Flag

Per default an orientation interrupt is triggered when any of the bits in register (0x0C) *orient*



changes state. The BMA280 can be configured to trigger orientation interrupts only when the device position changes in the x-y-plane while orientation changes with respect to the z-axis are ignored. A change of the orientation of the z-axis, and hence a state change of bit (0x0C) *orient<2>* is ignored (considered) when bit (0x2D) *orient_ud_en* is set to '0' ('1').

4.7.8 Flat detection

The flat detection feature gives information about the orientation of the devices' z-axis relative to the g-vector, i. e. it recognizes whether the device is in a flat position or not.

The flat angle Θ is adjustable by $(0x2E)$ *flat_theta* from 0° to 44.8°. The flat angle can be set according to following formula:

$$\Theta = \text{atan}\left(\frac{1}{8}\sqrt{\text{flat_theta}}\right)$$

A hysteresis of the flat detection can be enabled by $(0x2F)$ *flat_hy* bits. In this case the flat position is set if the angle drops below following threshold:

$$\Theta_{\text{hyst,ll}} = \text{atan}\left(\frac{1}{8}\sqrt{\text{flat_theta} \cdot \left(1 - \frac{\text{flat_hy}}{1024}\right) - \frac{\text{flat_hy}}{16}}\right)$$

The flat position is reset if the angle exceeds the following threshold:

$$\Theta_{\text{hyst,ul}} = \text{atan}\left(\frac{1}{8}\sqrt{\text{flat_theta} \cdot \left(1 + \frac{\text{flat_hy}}{1024}\right) + \frac{\text{flat_hy}}{16}}\right)$$

The flat interrupt is enabled (disabled) by writing '1' ('0') to bit $(0x16)$ *flat_en*. The flat value is stored in the $(0x0C)$ *flat* bit if the interrupt is enabled. This value is '1' if the device is in the flat position, it is '0' otherwise. The flat interrupt is generated if the flat value has changed and the new value is stable for at least the time given by the $(0x2F)$ *flat_hold_time* bits. A flat interrupt may be also generated if the flat interrupt is enabled. The actual status of the interrupt is stored in the $(0x09)$ *flat_int* bit. The flat orientation of the sensor can always be determined from reading the $(0x0C)$ *flat* bit after interrupt generation. If unlatched interrupt mode is used, the $(0x09)$ *flat_int* value and hence the interrupt is automatically cleared after one sample period. If temporary or latched interrupt mode is used, the $(0x09)$ *flat_int* value is kept fixed until the latch time expires or the interrupt is reset.

The meaning of the $(0x2F)$ *flat_hold_time* bits can be seen from Table 18.

Table 18: Meaning of *flat_hold_time*

| $(0x2F)$ <i>flat_hold_time</i> | Time |
|--|-------------|
| 00b | 0 |
| 01b | 512 ms |
| 10b | 1024 ms |
| 11b | 2048 ms |

4.7.9 Low-g interrupt

This interrupt is based on the comparison of acceleration data against a low-g threshold, which is most useful for free-fall detection.

The interrupt is enabled (disabled) by writing '1' ('0') to the (0x17) *low_en* bit. There are two modes available, 'single' mode and 'sum' mode. In 'single' mode, the acceleration of each axis is compared with the threshold; in 'sum' mode, the sum of absolute values of all accelerations $|acc_x| + |acc_y| + |acc_z|$ is compared with the threshold. The mode is selected by the contents of the (0x24) *low_mode* bit: '0' means 'single' mode, '1' means 'sum' mode.

The low-g threshold is set through the (0x23) *low_th* register. 1 LSB of (0x23) *low_th* always corresponds to an acceleration of 7.81 mg (i.e. increment is independent from g-range setting).

A hysteresis can be selected by setting the (0x24) *low_hy* bits. 1 LSB of (0x24) *low_hy* always corresponds to an acceleration difference of 125 mg in any g-range (as well, increment is independent from g-range setting).

The low-g interrupt is generated if the absolute values of the acceleration of all axes ('and' relation, in case of single mode) or their sum (in case of sum mode) are lower than the threshold for at least the time defined by the (0x22) *low_dur* register. The interrupt is reset if the absolute value of the acceleration of at least one axis ('or' relation, in case of single mode) or the sum of absolute values (in case of sum mode) is higher than the threshold plus the hysteresis for at least one data acquisition. In bit (0x09) *low_int* the interrupt status is stored.

The relation between the content of (0x22) *low_dur* and the actual delay of the interrupt generation is: $delay [ms] = [(0x22) low_dur + 1] \cdot 2 \text{ ms}$. Therefore, possible delay times range from 2 ms to 512 ms.

4.7.10 High-g interrupt

This interrupt is based on the comparison of acceleration data against a high-g threshold for the detection of shock or other high-acceleration events.

The high-g interrupt is enabled (disabled) per axis by writing '1' ('0') to bits (0x17) *high_en_x*, (0x17) *high_en_y*, and (0x17) *high_en_z*, respectively. The high-g threshold is set through the (0x26) *high_th* register. The meaning of an LSB of (0x26) *high_th* depends on the selected g-range: it corresponds to 7.81 mg in 2g-range, 15.63 mg in 4g-range, 31.25 mg in 8g-range, and 62.5 mg in 16g-range (i.e. increment depends from g-range setting).

A hysteresis can be selected by setting the (0x24) *high_hy* bits. Analogously to (0x26) *high_th*, the meaning of an LSB of (0x24) *high_hy* is g-range dependent: It corresponds to an acceleration difference of 125 mg in 2g-range, 250 mg in 4g-range, 500 mg in 8g-range, and 1000mg in 16g-range (as well, increment depends from g-range setting).

The high-g interrupt is generated if the absolute value of the acceleration of at least one of the enabled axes ('or' relation) is higher than the threshold for at least the time defined by the (0x25) *high_dur* register. The interrupt is reset if the absolute value of the acceleration of all enabled axes ('and' relation) is lower than the threshold minus the hysteresis for at least the time defined by the (0x25) *high_dur* register. In bit (0x09) *high_int* the interrupt status is stored. The relation between the content of (0x25) *high_dur* and the actual delay of the interrupt generation is $\text{delay [ms]} = [(0x22) \text{ low_dur} + 1] \cdot 2 \text{ ms}$. Therefore, possible delay times range from 2 ms to 512 ms. The interrupt will be cleared immediately once acceleration is lower than threshold.

4.7.10.1 Axis and sign information of high-g interrupt

The axis which triggered the interrupt is indicated by bits (0x0C) *high_first_x*, (0x0C) *high_first_y*, and (0x0C) *high_first_z*. The bit corresponding to the triggering axis contains a '1' while the other bits hold a '0'. These bits are cleared together with clearing the interrupt status. The sign of the triggering acceleration is stored in bit (0x0C) *high_sign*. If (0x0C) *high_sign* = '0' ('1'), the sign is positive (negative).

4.7.11 No-motion / slow motion detection

The slow-motion/no-motion interrupt engine can be configured in two modes.

In slow-motion mode an interrupt is triggered when the measured slope of at least one enabled axis exceeds the programmable slope threshold for a programmable number of samples. Hence the engine behaves similar to the any-motion interrupt, but with a different set of parameters. In order to suppress false triggers, the interrupt is only generated (cleared) if a certain number N of consecutive slope data points is larger (smaller) than the slope threshold given by $(0x27) slo_no_mot_dur<1:0>$. The number is $N = (0x27) slo_no_mot_dur<1:0> + 1$.

In no-motion mode an interrupt is generated if the slope on all selected axes remains smaller than a programmable threshold for a programmable delay time. Figure 11 shows the timing diagram for the no-motion interrupt. The scaling of the threshold value is identical to that of the slow-motion interrupt. However, in no-motion mode register $(0x27) slo_no_mot_dur$ defines the delay time before the no-motion interrupt is triggered. Table 19 lists the delay times adjustable with register $(0x27) slo_no_mot_dur$. The timer tick period is 1 second. Hence using short delay times can result in considerable timing uncertainty.

If bit $(0x18) slo_no_mot_sel$ is set to '1' ('0') the no-motion/slow-motion interrupt engine is configured in the no-motion (slow-motion) mode. Common to both modes, the engine monitors the slopes of the axes that have been enabled with bits $(0x18) slo_no_mot_en_x$, $(0x18) slo_no_mot_en_y$, and $(0x18) slo_no_mot_en_z$ for the x-axis, y-axis and z-axis, respectively. The measured slope values are continuously compared against the threshold value defined in register $(0x29) slo_no_mot_th$. The scaling is such that 1 LSB of $(0x29) slo_no_mot_th$ corresponds to 3.91 mg in 2g-range (7.81 mg in 4g-range, 15.6 mg in 8g-range and 31.3 mg in 16g-range). Therefore the maximum value is 996 mg in 2g-range (1.99g in 4g-range, 3.98g in 8g-range and 7.97g in 16g-range). The time difference between the successive acceleration samples depends on the selected bandwidth and equates to $1/(2 * bw)$.

Table 19: No-motion time-out periods

| $(0x27) slo_no_mot_dur$ | Delay time | $(0x27) slo_no_mot_dur$ | Delay time | $(0x27) slo_no_mot_dur$ | Delay Time |
|----------------------------|------------|----------------------------|------------|----------------------------|------------|
| 0 | 1 s | 16 | 40 s | 32 | 88 s |
| 1 | 2 s | 17 | 48 s | 33 | 96 s |
| 2 | 3 s | 18 | 56 s | 34 | 104 s |
| ... | ... | 19 | 64 s | ... | ... |
| 14 | 15 s | 20 | 72 s | 62 | 328 s |
| 15 | 16 s | 21 | 80 s | 63 | 336 s |

Note: $slo_no_mot_dur$ values 22 to 31 are not specified

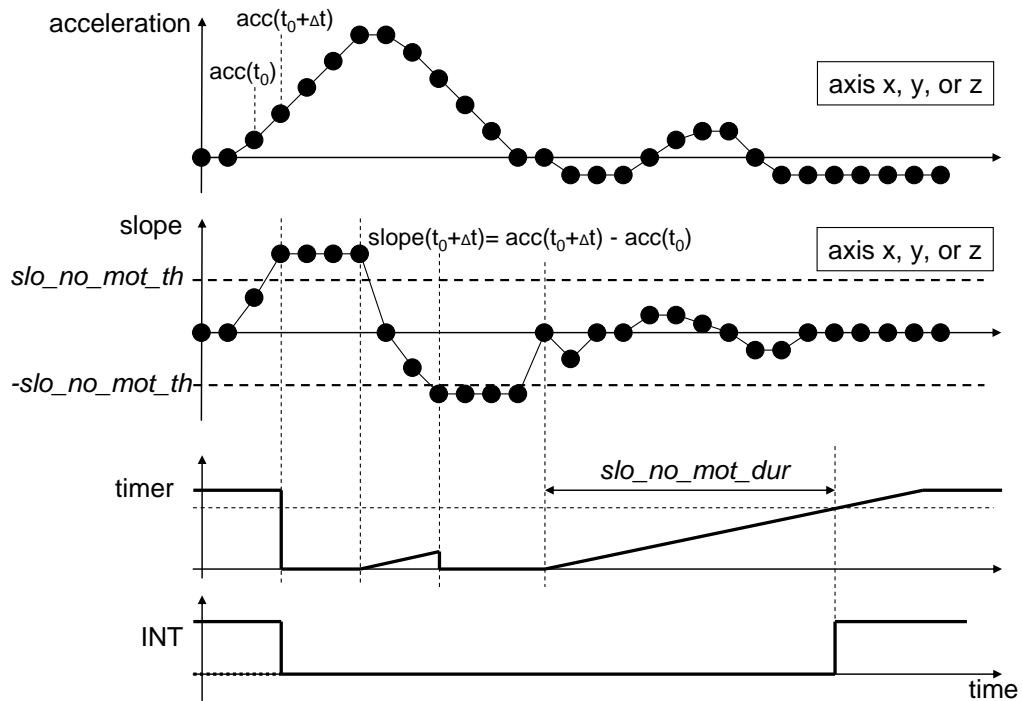


Figure 11: Timing of No-motion interrupt

4.8 Softreset

A softreset causes all user configuration settings to be overwritten with their default value and the sensor to enter normal mode.

A softreset is initiated by means of writing value 0xB6 to register (0x14) *softreset*. Subsequently a waiting time of $t_{w,up1}$ (max.) is required prior to accessing any configuration registers.

5. FIFO Operation

5.1 FIFO Operating Modes

The BMA280 features an integrated FIFO memory capable of storing up to 32 frames. Conceptually each frame consists of three 16 bit words corresponding to the x, y and z- axis, which are sampled at the same point in time. At the core of the FIFO is a buffer memory, which can be configured to operate in the following modes:

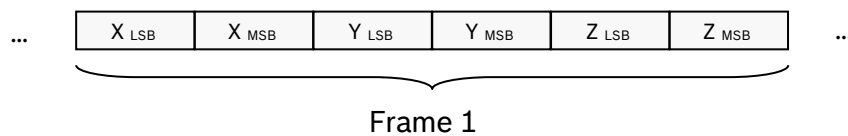
- **FIFO Mode:** In FIFO mode the acceleration data of the selected axes are stored in the buffer memory. If enabled, a watermark interrupt is triggered when the buffer has filled up to a configurable level. The buffer will be continuously filled until the fill level reaches 32 frames. When it is full the data collection is stopped, and all additional samples are ignored. Once the buffer is full, a FIFO-full interrupt is generated if it has been enabled.
- **STREAM Mode:** In STREAM mode the acceleration data of the selected axes are stored in the buffer until it is full. The buffer has a depth of 31 frames. When the buffer is full the data collection continues and oldest entry is discarded. If enabled, a watermark interrupt is triggered when the buffer is filled to a configurable level. Once the buffer is full, a FIFO-full interrupt is generated if it has been enabled.
- **BYPASS Mode:** In bypass mode, only the current sensor data can be read out from the FIFO address. Essentially, the FIFO behaves like the STREAM mode with a depth of 1. Compared to reading the data from the normal data registers, the advantage to the user is that the packages X, Y, Z are from the same timestamp, while the data registers are updated sequentially and hence mixing of data from different axes can occur.

The primary FIFO operating mode is selected with register (*0x3E*) *fifo_mode* according to '00b' for BYPASS mode, '01b' for FIFO mode, and '10b' for STREAM mode. Writing to register (*0x3E*) clears the buffer content and resets the FIFO-full and watermark interrupts. When reading register (*0x3E*) *fifo_mode* always contains the current operating mode.

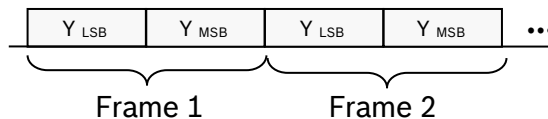
5.2 FIFO Data Readout

The FIFO stores the data that are also available at the acceleration read-out registers (0x02) to (0x07). Thus, all configuration settings apply to the FIFO data as well as the acceleration data readout registers. The FIFO read out is possible through register (0x3F). The readout can be performed using burst mode since the read address counter is no longer incremented, when it has reached address (0x3F). This implies that the trapping also occurs when the burst read access starts below address (0x3F). A single burst can read out one or more frames at a time. Register (0x3E) *fifo_data_select* controls the acceleration data of which axes are stored in the FIFO. Possible settings for register (0x3E) *fifo_data_select* are '00b' for x, y- and z-axis, '01b' for x-axis only, '10b' for y-axis, '11b' for z-axis only. The depth of the FIFO is independent of whether all or a single axis have been selected. Writing to register (0x3E) clears the buffer content and resets the FIFO-full and watermark interrupts.

If all axes are enabled, the format of the data read-out from register (0x3F) is as follows:



If only one axis is enabled, the format of the data read-out from register (0x3F) is as follows (example shown: y-axis only, other axes are equivalent).



If a frame is not completely read due to an incomplete read operation, the remaining part of the frame is discarded. In this case the FIFO aligns to the next frame during the next read operation. In order for the discarding mechanism to operate correctly, there must be a delay of at least 1.5 us between the last data bit of the partially read frame and the first address bit of the next FIFO read access. Otherwise frames must not be read out partially.

If the FIFO is read beyond the FIFO fill level zeroes (0) will be read. If the FIFO is read beyond the FIFO fill level the read or burst read access time must not exceed the sampling time t_{SAMPLE} . Otherwise frames may be lost.

5.3 FIFO Frame Counter and Overrun Flag

Register (0x0E) *fifo_frame_counter* reflects the current fill level of the buffer. If additional frames are written to the buffer although the FIFO is full, the (0x0E) *fifo_overrun* bit is set to '1'. The FIFO buffer is cleared, the FIFO fill level indicated in register (0x0E) *fifo_frame_counter* and the (0x0E) *fifo_overrun* bit are both set to '0' each time one a write access to one of the FIFO configuration registers (0x3E) or (0x30) occurs. The (0x0E) *fifo_overrun* bit is not reset when the FIFO fill level (0x0E) *fifo_frame_counter* has decremented to '0' due to reading from register (0x3F).



5.4 FIFO Interrupts

The FIFO controller can generate two different interrupt events, a FIFO-full and a watermark event. The FIFO-full and watermark interrupts are functional in all FIFO operating modes. The watermark interrupt is asserted when the fill level in the buffer has reached the frame count defined by register (0x30) *fifo_water_mark_trigger_retain*. In order to enable (disable) the watermark interrupt, the (0x17) *int_fwm_en* bit must be set to '1' ('0'). To map the watermark interrupt signal to INT1 pin (INT2 pin), (0x1A) *int1_fwm* ((0x1A) *int2_fwm*) bit must be set to '1'. The status of the watermark interrupt may be read back through the (0x0A) *fifo_wm_int* bit. Writing to register (0x30) *fifo_water_mark_trigger_retain* clears the FIFO buffer.

The FIFO-full interrupt is triggered when the buffer has been completely filled. In FIFO mode this occurs 32, in STREAM mode 31 samples, and in BYPASS mode 1 sample after the buffer has been cleared. In order to enable the FIFO-full interrupt, bit (0x17) *int_fful_en* as well as one or both of bits (0x1A) *int1_fful* or (0x1A) *int2_fful* must also be set to '1'. The status of the FIFO-full interrupt may be read back through bit (0x0A) *fifo_full_int*.

6. Register description

6.1 General remarks

The entire communication with the device is performed by reading from and writing to registers. Registers have a width of 8 bits; they are mapped to a common space of 64 addresses from $(0x00)$ up to $(0x3F)$. Within the used range there are several registers which are either completely or partially marked as 'reserved'. Any reserved bit is ignored when it is written and no specific value is guaranteed when read. It is recommended not to use registers at all which are completely marked as 'reserved'. Furthermore it is recommended to mask out (logical *and* with zero) reserved bits of registers which are partially marked as reserved.

Registers with addresses from $(0x00)$ up to $(0x0E)$ are read-only. Any attempt to write to these registers is ignored. There are bits within some registers that trigger internal sequences. These bits are configured for write-only access, e. g. $(0x21)$ *reset_int* or the entire $(0x14)$ *softreset* register, and read as value '0'.

6.2 Register map

| Register Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Access | Default | |
|------------------|---|-------------------------|-----------------|----------------------|-----------------------|----------------------|---------------|---------------|--------|---------|------|
| 0x3F | fifo_data_output_register<7:0> | | | | | | | | ro | 0x00 | |
| 0x3E | fifo_mode<1:0> | | | | fifo_data_select<1:0> | | | | w/r | 0x00 | |
| 0x3D | | | | | | | | | w/r | 0xFF | |
| 0x3C | GP1<7:0> | | | | | | | | w/r | 0x00 | |
| 0x3B | GP0<7:0> | | | | | | | | w/r | 0x00 | |
| 0x3A | offset_z<7:0> | | | | | | | | w/r | 0x00 | |
| 0x39 | offset_y<7:0> | | | | | | | | w/r | 0x00 | |
| 0x38 | offset_x<7:0> | | | | | | | | w/r | 0x00 | |
| 0x37 | offset_target_z<1:0> | | | offset_target_y<1:0> | | offset_target_x<1:0> | | cut_off | | w/r | 0x00 |
| 0x36 | offset_reset | cal_trigger<1:0> | | cal_rdy | hp_z_en | | hp_y_en | hp_x_en | w/r | 0x10 | |
| 0x35 | | | | | | | | | w/r | 0x00 | |
| 0x34 | | | | | | | | | w/r | 0x00 | |
| 0x33 | nvm_remain<3:0> | | | nvm_load | | i2c_wdt_en | i2c_wdt_sel | spi3 | w/r | 0x00 | |
| 0x32 | | | | | | | | | w/r | 0xF0 | |
| 0x31 | | | | | | | | | w/r | 0x00 | |
| 0x30 | fifo_water_mark_level_trigger_retain<5:0> | | | | | | | | w/r | 0xFF | |
| 0x2F | flat_hold_time<1:0> | | | | flat_theta<5:0> | | flat_hy<2:0> | | w/r | 0x11 | |
| 0x2E | | | | | | | | | w/r | 0x08 | |
| 0x2D | orient_ud_en | | | | | | | | w/r | 0x48 | |
| 0x2C | orient_hyst<2:0> | | | orient_blocking<1:0> | | orient_mode<1:0> | | | | w/r | 0x18 |
| 0x2B | tap_samp<1:0> | | tap_th<4:0> | | | | tap_dur<2:0> | | w/r | 0x0A | |
| 0x2A | tap_quiet | tap_shock | | | | | | | w/r | 0x04 | |
| 0x29 | slo_no_mot_th<7:0> | | | | | | | | w/r | 0x14 | |
| 0x28 | slo_th<7:0> | | | | | | | | w/r | 0x14 | |
| 0x27 | slo_no_mot_dur<5:0> | | | | slo_dur<1:0> | | | | w/r | 0x00 | |
| 0x26 | high_th<7:0> | | | | | | | | w/r | 0xC0 | |
| 0x25 | high_dur<7:0> | | | | | | | | w/r | 0x0F | |
| 0x24 | high_hy<1:0> | | low_mode | | low_hy<1:0> | | | | w/r | 0x81 | |
| 0x23 | low_th<7:0> | | | | | | | | w/r | 0x30 | |
| 0x22 | low_dur<7:0> | | | | | | | | w/r | 0x09 | |
| 0x21 | reset_int | | | | | latch_int<3:0> | | | | w/r | 0x00 |
| 0x20 | | | | | int2_od | int2_lm | int1_od | int1_lm | w/r | 0x05 | |
| 0x1F | | | | | | | | | w/r | 0xFF | |
| 0x1E | int_src_data | | int_src_tap | int_src_slo_no_mot | int_src_slope | int_src_high | int_src_low | | | w/r | 0x00 |
| 0x1D | | | | | | | | | w/r | 0xFF | |
| 0x1C | | | | | | | | | w/r | 0xFF | |
| 0x1B | int2_flat | int2_orient | int2_s_tap | int2_d_tap | int2_slo_no_mot | int2_slope | int2_high | int2_low | w/r | 0x00 | |
| 0x1A | int2_data | int2_fwm | int2_full | | | int1_full | int1_fwm | int1_data | w/r | 0x00 | |
| 0x19 | int1_flat | int1_orient | int1_s_tap | int1_d_tap | int1_slo_no_mot | int1_slope | int1_high | int1_low | w/r | 0x00 | |
| 0x18 | slo_no_mot_sel | | | | | | | | w/r | 0x00 | |
| 0x17 | int_fwm_en | | int_full_en | data_en | low_en | high_en_z | high_en_y | high_en_x | w/r | 0x00 | |
| 0x16 | flat_en | orient_en | s_tap_en | d_tap_en | slope_en_z | | slope_en_y | slope_en_x | w/r | 0x00 | |
| 0x15 | | | | | | | | | w/r | 0xFF | |
| 0x14 | softreset | | | | | | | | wo | 0x00 | |
| 0x13 | data_high_bw | shadow_dis | | | | | | | w/r | 0x00 | |
| 0x12 | lowpower_mode | | sleeptimer_mode | | | | | | | w/r | 0x00 |
| 0x11 | suspend | lowpower_en | deep_suspend | sleep_dur<3:0> | | | | | | w/r | 0x00 |
| 0x10 | bw<4:0> | | | | | | | | w/r | 0x0F | |
| 0x0F | range<3:0> | | | | | | | | w/r | 0x03 | |
| 0x0E | fifo_overrun | fifo_frame_counter<6:0> | | | | | | | | ro | 0x00 |
| 0x0D | | | | | | | | | w/r | 0xFF | |
| 0x0C | flat | orient<2:0> | | | high_sign | high_first_z | high_first_y | high_first_x | ro | 0x00 | |
| 0x0B | tap_sign | tap_first_z | tap_first_y | tap_first_x | slope_sign | slope_first_z | slope_first_y | slope_first_x | ro | 0x00 | |
| 0x0A | data_int | fifo_wm_int | fifo_full_int | | | | | | | ro | 0x00 |
| 0x09 | flat_int | orient_int | s_tap_int | d_tap_int | slo_no_mot_int | slope_int | high_int | low_int | ro | 0x00 | |
| 0x08 | temp<7:0> | | | | | | | | ro | 0x00 | |
| 0x07 | acc_z_msb<13:6> | | | | | | | | ro | 0x00 | |
| 0x06 | acc_z_lsb<5:0> | | | | new_data_z | | | | ro | 0x00 | |
| 0x05 | acc_y_msb<13:6> | | | | new_data_y | | | | ro | 0x00 | |
| 0x04 | acc_y_lsb<5:0> | | | | new_data_x | | | | ro | 0x00 | |
| 0x03 | acc_x_msb<13:6> | | | | | | | | ro | 0x00 | |
| 0x02 | acc_x_lsb<5:0> | | | | | | | | ro | 0x00 | |
| 0x01 | | | | | | | | | ro | - | |
| 0x00 | chip_id<7:0> | | | | | | | | ro | 0xFB | |

common w/r registers: Application specific settings which are not equal to the default settings, must be re-set to its designated values after POR, soft-reset and wake up from deep suspend.

user w/r registers: Initial default content = 0x00. Freely programmable by the user.

Remains unchanged after POR, soft-reset and wake up from deep suspend.

Figure 12: Register map

Register 0x00 (BGW_CHIPID)

The register contains the chip identification code.

| Name | 0x00 | BGW_CHIPID | | |
|-------------|--------------|------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | chip_id<7:4> | | | |

| | | | | |
|-------------|--------------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | chip_id<3:0> | | | |

chip_id<7:0>: Fixed value b'1111'1011

Register 0x02 (ACCD_X_LSB)

The register contains the least-significant bits of the X-channel acceleration readout value. When reading out X-channel acceleration values, data consistency is guaranteed if the ACCD_X_LSB is read out before the ACCD_X_MSB and shadow_dis='0'. In this case, after the ACCD_X_LSB has been read, the value in the ACCD_X_MSB register is locked until the ACCD_X_MSB has been read. This condition is inherently fulfilled if a burst-mode read access is performed. Acceleration data may be read from register ACCD_X_LSB at any time except during power-up and in DEEP_SUSPEND mode.

| Name | 0x02 | ACCD_X_LSB | | |
|-------------|----------------|------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | acc_x_lsb<5:2> | | | |

| | | | | |
|-------------|----------------|-----|-----------|------------|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | acc_x_lsb<1:0> | | undefined | new_data_x |

acc_x_lsb<5:2>: Upper 4 bits of least significant 6 bits of acceleration read-back value; (two's-complement format)

acc_x_lsb<1:0>: Lower 2 bits of least significant 6 bits of acceleration read-back value; (two's-complement format)

undefined: random data; to be ignored.

new_data_x: ,0': acceleration value has not been updated since it has been read out last
 ,1': acceleration value has been updated since it has been read out last

**Register 0x03 (ACCD_X_MSB)**

The register contains the most-significant bits of the X-channel acceleration readout value. When reading out X-channel acceleration values, data consistency is guaranteed if the ACCD_X_LSB is read out before the ACCD_X_MSB and shadow_dis='0'. In this case, after the ACCD_X_LSB has been read, the value in the ACCD_X_MSB register is locked until the ACCD_X_MSB has been read. This condition is inherently fulfilled if a burst-mode read access is performed. Acceleration data may be read from register ACCD_X_MSB at any time except during power-up and in DEEP_SUSPEND mode.

| Name | 0x02 | ACCD_X_MSB | | |
|-------------|------------------|------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | acc_x_msb<13:10> | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | acc_x_msb<9:6> | | | |

acc_x_msb<13:6>: Most significant 8 bits of acceleration read-back value (two's-complement format)

Register 0x04 (ACCD_Y_LSB)

The register contains the least-significant bits of the Y-channel acceleration readout value. When reading out Y-channel acceleration values, data consistency is guaranteed if the ACCD_Y_LSB is read out before the ACCD_Y_MSB and shadow_dis='0'. In this case, after the ACCD_Y_LSB has been read, the value in the ACCD_Y_MSB register is locked until the ACCD_Y_MSB has been read. This condition is inherently fulfilled if a burst-mode read access is performed. Acceleration data may be read from register ACCD_Y_LSB at any time except during power-up and in DEEP_SUSPEND mode.

| Name | 0x04 | ACCD_Y_LSB | | |
|-------------|----------------|------------|-----------|------------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | acc_y_lsb<5:2> | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | acc_y_lsb<1:0> | | undefined | new_data_y |

acc_y_lsb<5:2>: Upper 4 bits of least significant 6 bits of acceleration read-back value; (two's-complement format)

acc_y_lsb<1:0>: Lower 2 bits of least significant 6 bits of acceleration read-back value; (two's-complement format)

undefined: random data; to be ignored

new_data_y: ,0': acceleration value has not been updated since it has been read out last
 ,1': acceleration value has been updated since it has been read out last

Register 0x05 (ACCD_Y_MSB)

The register contains the most-significant bits of the Y-channel acceleration readout value. When reading out Y-channel acceleration values, data consistency is guaranteed if the ACCD_Y_LSB is read out before the ACCD_Y_MSB and shadow_dis='0'. In this case, after the ACCD_Y_LSB has been read, the value in the ACCD_Y_MSB register is locked until the ACCD_Y_MSB has been read. This condition is inherently fulfilled if a burst-mode read access is performed. Acceleration data may be read from register ACCD_Y_MSB at any time except during power-up and in DEEP_SUSPEND mode.

| Name | 0x05 | ACCD_Y_MSB | | |
|-------------|------------------|------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | acc_y_msb<13:10> | | | |

| | | | | |
|-------------|----------------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | acc_y_msb<9:6> | | | |

acc_y_msb<13:6>: Most significant 8 bits of acceleration read-back value (two's-complement format)

Register 0x06 (ACCD_Z_LSB)

The register contains the least-significant bits of the Z-channel acceleration readout value. When reading out Z-channel acceleration values, data consistency is guaranteed if the ACCD_Z_LSB is read out before the ACCD_Z_MSB and shadow_dis='0'. In this case, after the ACCD_Z_LSB has been read, the value in the ACCD_Z_MSB register is locked until the ACCD_Z_MSB has been read. This condition is inherently fulfilled if a burst-mode read access is performed. Acceleration data may be read from register ACCD_Z_LSB at any time except during power-up and in DEEP_SUSPEND mode.

| Name | 0x06 | ACCD_Z_LSB | | |
|-------------|----------------|------------|-----------|------------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | acc_z_lsb<5:2> | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | acc_z_lsb<1:0> | | undefined | new_data_z |

Acc_z_lsb<5:2>: Upper 4 bits of least significant 6 bits of acceleration readback value; (two's-complement format)

Acc_z_lsb<1:0>: Lower 2 bits of least significant 6 bits of acceleration read-back value; (two's-complement format)

undefined: random data; to be ignored

new_data_z: ,0': acceleration value has not been updated since it has been read out last
 ,1': acceleration value has been updated since it has been read out last

Register 0x07 (ACCD_Z_MSB)

The register contains the most-significant bits of the Z-channel acceleration readout value. When reading out Z-channel acceleration values, data consistency is guaranteed if the ACCD_Z_LSB is read out before the ACCD_Z_MSB and shadow_dis='0'. In this case, after the ACCD_Z_LSB has been read, the value in the ACCD_Z_MSB register is locked until the ACCD_Z_MSB has been read. This condition is inherently fulfilled if a burst-mode read access is performed. Acceleration data may be read from register ACCD_Z_MSB at any time except during power-up and in DEEP_SUSPEND mode.

| Name | 0x07 | ACCD_Z_MSB | | |
|-------------|------------------|------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | acc_z_msb<13:10> | | | |

| | | | | |
|-------------|----------------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | acc_z_msb<9:6> | | | |

acc_z_msb<13:6>: Most significant 8 bits of acceleration read-back value (two's-complement format)

Register 0x08 (ACCD_TEMP)

The register contains the current chip temperature represented in two's complement format. A readout value of temp<7:0>=0x00 corresponds to a temperature of 23°C.

| Name | 0x08 | ACCD_TEMP | | |
|-------------|-----------|-----------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | temp<7:4> | | | |

| | | | | |
|-------------|-----------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | temp<3:0> | | | |

temp<7:0>: Temperature value (two's complement format)

Register 0x09 (INT_STATUS_0)

The register contains interrupt status flags. Each flag is associated with a specific interrupt function. It is set when the associated interrupt triggers. The setting of latch_int<3:0> controls if the interrupt signal and hence the respective interrupt flag will be permanently latched, temporarily latched or not latched. The interrupt function associated with a specific status flag must be enabled.

| Name | 0x09 | INT_STATUS_0 | | |
|-------------|----------|--------------|-----------|-----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | flat_int | orient_int | s_tap_int | d_tap_int |

| | | | | |
|-------------|----------------|-----------|----------|---------|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | slo_no_mot_int | slope_int | high_int | low_int |

flat_int: flat interrupt status: '0'→inactive, '1' →active
 orient_int: orientation interrupt status: '0'→inactive, '1' →active
 s_tap_int: single tap interrupt status: '0'→inactive, '1' →active
 d_tap_int: double tap interrupt status: '0'→inactive, '1' →active
 slo_not_mot_int: slow/no-motion interrupt status: '0'→inactive, '1' →active
 slope_int: slope interrupt status: '0'→inactive, '1' →active
 high_int: high-g interrupt status: '0'→inactive, '1' →active
 low_int: low-g interrupt status: '0'→inactive, '1' →active

**Register 0x0A (INT_STATUS_1)**

The register contains interrupt status flags. Each flag is associated with a specific interrupt function. It is set when the associated interrupt engine triggers. The setting of latch_int<3:0> controls if the interrupt signal and hence the respective interrupt flag will be permanently latched, temporarily latched or not latched. The interrupt function associated with a specific status flag must be enabled.

| Name | 0x0A | INT_STATUS_1 | | |
|-------------|----------|--------------|---------------|----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | data_int | fifo_wm_int | fifo_full_int | reserved |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | Reserved | | | |

data_int: data ready interrupt status: '0'→inactive, '1' →active
 fifo_wm_int: FIFO watermark interrupt status: '0'→inactive, '1' →active
 fifo_full_int: FIFO full interrupt status: '0'→inactive, '1' →active
 reserved: reserved, write to '0'

Register 0x0B (INT_STATUS_2)

The register contains interrupt status flags. Each flag is associated with a specific interrupt engine. It is set when the associated interrupt engine triggers. The setting of latch_int<3:0> controls if the interrupt signal and hence the respective interrupt flag will be permanently latched, temporarily latched or not latched. The interrupt function associated with a specific status flag must be enabled.

| Name | 0x0B | INT_STATUS_2 | | |
|-------------|----------|--------------|-------------|-------------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | tap_sign | tap_first_z | tap_first_y | tap_first_x |

| Bit | 3 | 2 | 1 | 0 |
|-------------|------------|---------------|---------------|---------------|
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | slope_sign | slope_first_z | slope_first_y | slope_first_x |

| | |
|----------------|---|
| tap_sign: | sign of single/double tap triggering signal was '0' → positive, or '1' → negative |
| tap_first_z: | single/double tap interrupt: '1' → triggered by, or '0' → not triggered by z-axis |
| tap_first_y: | single/double tap interrupt: '1' → triggered by, or '0' → not triggered by y-axis |
| tap_first_x: | single/double tap interrupt: '1' → triggered by, or '0' → not triggered by x-axis |
| slope_sign: | slope sign of slope tap triggering signal was '0' → positive, or '1' → negative |
| slope_first_z: | slope interrupt: '1' → triggered by, or '0' → not triggered by z-axis |
| slope_first_y: | slope interrupt: '1' → triggered by, or '0' → not triggered by y-axis |
| slope_first_x: | slope interrupt: '1' → triggered by, or '0' → not triggered by x-axis |

Register 0x0C (INT_STATUS_3)

The register contains interrupt status flags. Each flag is associated with a specific interrupt engine. It is set when the associated interrupt engine triggers. With the exception of orient<3:0> the setting of latch_int<3:0> controls if the interrupt signal and hence the respective interrupt flag will be permanently latched, temporarily latched or not latched. The interrupt function associated with a specific status flag must be enabled.

| Name | 0x0C | INT_STATUS_3 | | |
|-------------|------|--------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | flat | orient<2:0> | | |

| Bit | 3 | 2 | 1 | 0 |
|-------------|-----------|--------------|--------------|--------------|
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | high_sign | high_first_z | high_first_y | high_first_x |

- flat: device is in '1' → flat, or '0' → non flat position;
only valid if (0x16) flat_en = '1'
- orient<2>: Orientation value of z-axis: '0' → upward looking, or '1' → downward looking. The flag always reflect the current orientation status, independent of the setting of latch_int<3:0>. The flag is not updated as long as an orientation blocking condition is active.
- orient<1:0>: orientation value of x-y-plane:
'00' → portrait upright; '01' → portrait upside down;
'10' → landscape left; '11' → landscape right;
The flags always reflect the current orientation status, independent of the setting of latch_int<3:0>. The flag is not updated as long as an orientation blocking condition is active.
- high_sign: sign of acceleration signal that triggered high-g interrupt was '0' → positive, '1' → negative
- high_first_z: high-g interrupt: '1' → triggered by, or '0' → not triggered by z-axis
- high_first_y: high-g interrupt: '1' → triggered by, or '0' → not triggered by y-axis
- high_first_x: high-g interrupt: '1' → triggered by, or '0' → not triggered by x-axis

**Register 0x0E (FIFO_STATUS)**

The register contains FIFO status flags.

| Name | 0x0E | FIFO_STATUS | | |
|-------------|---------------|-------------------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | fifo_ overrun | fifo_frame_counter<6:4> | | |

| | | | | |
|-------------|-------------------------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | fifo_frame_counter<3:0> | | | |

fifo_ overrun: FIFO overrun condition has '1' → occurred, or '0' → not occurred; flag can be cleared by writing to the FIFO configuration register FIFO_CONFIG_1 only

fifo_frame_counter<6:4>: Current fill level of FIFO buffer. An empty FIFO corresponds to 0x00. The frame counter can be cleared by reading out all frames from the FIFO buffer or writing to the FIFO configuration register FIFO_CONFIG_1.

Register 0x0F (PMU_RANGE)

The register allows the selection of the accelerometer g-range.

| Name | 0x0F | PMU_RANGE | | |
|-------------|----------|-----------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | Reserved | | | |

0

| | | | | |
|-------------|------------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 1 | 1 |
| Content | range<3:0> | | | |

range<3:0>: Selection of accelerometer g-range:
 '0011b' → ±2g range; '0101b' → ±4g range; '1000b' → ±8g range;
 '1100b' → ±16g range; all other settings → reserved (do not use)

reserved: write '0'

Register 0x10 (PMU_BW)

The register allows the selection of the acceleration data filter bandwidth.

| Name | 0x10 | PMU_BW | | |
|-------------|----------|--------|-----|-------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | Reserved | | | bw<4> |

0

| | | | | |
|-------------|---------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 1 | 1 | 1 | 1 |
| Content | bw<3:0> | | | |

bw<4:0>: Selection of data filter bandwidth:
 '00xxb' → 7.81 Hz, '0100b' → 7.81 Hz, '01001b' → 15.63 Hz,
 '01010b' → 31.25 Hz, '01011b' → 62.5 Hz, '01100b' → 125 Hz,
 '01101b' → 250 Hz, '01110b' → 500 Hz, '01111b' → ODR_{max.},
 '1xxxxb' → ODR_{max.}

reserved: write '0'

Register 0x11 (PMU_LPW)

Selection of the main power modes and the low power sleep period.

| Name | 0x11 | PMU_LPW | | |
|-------------|---------|-------------|--------------|--------------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | suspend | lowpower_en | deep_suspend | sleep_dur<3> |

| Bit | 3 | 2 | 1 | 0 |
|-------------|----------------|-----|-----|----------|
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | sleep_dur<2:0> | | | reserved |

suspend, low_power_en, deep_suspend:

Main power mode configuration setting {suspend; lowpower_en; deep_suspend}:

```

{0; 0; 0} → NORMAL mode;
{0; 0; 1} → DEEP_SUSPEND mode;
{0; 1; 0} → LOW_POWER mode;
{1; 0; 0} → SUSPEND mode;
{all other} → illegal
  
```

Please note that only certain power mode transitions are permitted.

sleep_dur<3:0>: Configures the sleep phase duration in LOW_POWER mode:

```

'0000b' to '0101b' → 0.5 ms, '0110b' → 1 ms,
'0111b' → 2 ms, '1000b' → 4 ms,
'1001b' → 6 ms, '1010b' → 10 ms,
'1011b' → 25 ms, '1100b' → 50 ms,
'1101b' → 100 ms, '1110b' → 500 ms,
'1111b' → 1 s
  
```

Please note, that all application specific settings which are not equal to the default settings (refer to 6.2 register map), must be re-set to its designated values after DEEP_SUSPEND.

Register 0x12 (PMU_LOW_POWER)

Configuration settings for low power mode.

| Name | 0x12 | PMU_LOW_POWER | | |
|-------------|----------|---------------|-----------------|----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | lowpower_mode | sleeptimer_mode | reserved |

| | | | | |
|-------------|----------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | Reserved | | | |

lowpower_mode: select '0' → LPM1, or '1' → LPM2 configuration for SUSPEND and LOW_POWER mode. In the LPM1 configuration the power consumption in LOW_POWER mode and SUSPEND mode is significantly reduced when compared to LPM2 configuration, but the FIFO is not accessible and writing to registers must be slowed down. In the LPM2 configuration the power consumption in LOW_POWER mode is reduced compared to NORMAL mode, but the FIFO is fully accessible and registers can be written to at full speed.

sleeptimer_mode: when in LOW_POWER mode '0' → use event-driven time-base mode (compatible with BMA250), or '1' → use equidistant sampling time-base mode. Equidistant sampling of data into the FIFO is maintained in equidistant time-base mode only.

reserved: write '0'

**Register 0x13 (ACCD_HBW)**

Acceleration data acquisition and data output format.

| Name | 0x13 | ACCD_HBW | | |
|-------------|--------------|---------------------|----------|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 (1 in 8-bit mode) | 0 | 0 |
| Content | data_high_bw | shadow_dis | reserved | |

| | | | | |
|-------------|----------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | Reserved | | | |

data_high_bw: select whether '1' → unfiltered, or '0' → filtered data may be read from the acceleration data registers.

shadow_dis: '1' → disable, or '0' → the shadowing mechanism for the acceleration data output registers. When shadowing is enabled, the content of the acceleration data component in the MSB register is locked, when the component in the LSB is read, thereby ensuring the integrity of the acceleration data during read-out. The lock is removed when the MSB is read.

reserved: write '0'

Register 0x14 (BGW_SOFTRESET)

Controls user triggered reset of the sensor.

| Name | 0x14 | BGW_SOFTRESET | | |
|-------------|-----------|---------------|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | W | W | W | W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | Softreset | | | |

| | | | | |
|-------------|-----------|---|---|---|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | W | W | W | W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | Softreset | | | |

softreset: 0xB6 → triggers a reset. Other values are ignored. Following a delay, all user configuration settings are overwritten with their default state or the setting stored in the NVM, wherever applicable. This register is functional in all operation modes. Please note that all application specific settings which are not equal to the default settings (refer to 6.2 register map), must be reconfigured to their designated values.

Register 0x16 (INT_EN_0)

Controls which interrupt engines in group 0 are enabled.

| Name | 0x16 | INT_EN_0 | | |
|-------------|---------|-----------|----------|----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | flat_en | orient_en | s_tap_en | d_tap_en |

| | | | | |
|-------------|----------|------------|------------|------------|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | slope_en_z | slope_en_y | slope_en_x |

flat_en: flat interrupt: '0' → disabled, or '1' → enabled
 orient_en: orientation interrupt: '0' → disabled, or '1' → enabled
 s_tap_en: single tap interrupt: '0' → disabled, or '1' → enabled
 d_tap_en: double tap interrupt: '0' → disabled, or '1' → enabled
 reserved: write '0'
 slope_en_z: slope interrupt, z-axis component: '0' → disabled, or '1' → enabled
 slope_en_y: slope interrupt, y-axis component: '0' → disabled, or '1' → enabled
 slope_en_x: slope interrupt, x-axis component: '0' → disabled, or '1' → enabled

Register 0x17 (INT_EN_1)

Controls which interrupt engines in group 1 are enabled.

| Name | 0x17 | INT_EN_1 | | |
|--------------------|----------|------------|-------------|---------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | int_fwm_en | int_full_en | data_en |

| Bit | 3 | 2 | 1 | 0 |
|--------------------|--------|-----------|-----------|-----------|
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | low_en | high_en_z | high_en_y | high_en_x |

reserved: write '0'
 int_fwm_en: FIFO watermark interrupt: '0'→disabled, or '1' →enabled
 int_full_en: FIFO full interrupt: '0'→disabled, or '1' →enabled
 data_en: data ready interrupt: '0'→disabled, or '1' →enabled
 low_en: low-g interrupt: '0'→disabled, or '1' →enabled
 high_en_z: high-g interrupt, z-axis component: '0'→disabled, or '1' →enabled
 high_en_y: high-g interrupt, y-axis component: '0'→disabled, or '1' →enabled
 high_en_x: high-g interrupt, x-axis component: '0'→disabled, or '1' →enabled

**Register 0x18 (INT_EN_2)**

Controls which interrupt engines in group 2 are enabled.

| Name | 0x18 | INT_EN_2 | | |
|-------------|----------|----------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | Reserved | | | |

| | | | | |
|-------------|----------------|-----------------|-----------------|-----------------|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | slo_no_mot_sel | slo_no_mot_en_z | slo_no_mot_en_y | slo_no_mot_en_x |

reserved: write '0'

slo_no_mot_sel: select '0'→slow-motion, '1' →no-motion interrupt function

slo_no_mot_en_z: slow/n-motion interrupt, z-axis component: '0'→disabled, or '1' →enabled

slo_no_mot_en_y: slow/n-motion interrupt, y-axis component: '0'→disabled, or '1' →enabled

slo_no_mot_en_x: slow/n-motion interrupt, x-axis component: '0'→disabled, or '1' →enabled

Register 0x19 (INT_MAP_0)

Controls which interrupt signals are mapped to the INT1 pin.

| Name | 0x19 | INT_MAP_0 | | |
|-------------|-----------|-------------|------------|------------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | int1_flat | int1_orient | int1_s_tap | int1_d_tap |

| | | | | |
|-------------|-----------------|------------|-----------|----------|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | int1_slo_no_mot | int1_slope | int1_high | int1_low |

| | |
|------------------|---|
| int1_flat: | map flat interrupt to INT1 pin: '0'→disabled, or '1' →enabled |
| int1_orient: | map orientation interrupt to INT1 pin: '0'→disabled, or '1' →enabled |
| int1_s_tap: | map single tap interrupt to INT1 pin: '0'→disabled, or '1' →enabled |
| int1_d_tap: | map double tap interrupt to INT1 pin: '0'→disabled, or '1' →enabled |
| int1_slo_no_mot: | map slow/no-motion interrupt to INT1 pin: '0'→disabled, or '1' →enabled |
| int1_slope: | map slope interrupt to INT1 pin: '0'→disabled, or '1' →enabled |
| int1_high: | map high-g to INT1 pin: '0'→disabled, or '1' →enabled |
| int1_low: | map low-g to INT1 pin: '0'→disabled, or '1' →enabled |

Register 0x1A (INT_MAP_1)

Controls which interrupt signals are mapped to the INT1 and INT2 pins.

| Name | 0x1A | INT_MAP_1 | | |
|--------------------|-----------|-----------|------------|----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | int2_data | int2_fwm | int2_ffull | reserved |

| | | | | |
|--------------------|----------|------------|----------|-----------|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | int1_ffull | int1_fwm | int1_data |

int2_data: map data ready interrupt to INT2 pin: '0'→disabled, or '1' →enabled
 int2_fwm: map FIFO watermark interrupt to INT2 pin: '0'→disabled, or '1' →enabled
 int2_ffull: map FIFO full interrupt to INT2 pin: '0'→disabled, or '1' →enabled
 reserved: write '0'
 int1_ffull: map FIFO full interrupt to INT1 pin: '0'→disabled, or '1' →enabled
 int1_fwm: map FIFO watermark interrupt to INT1 pin: '0'→disabled, or '1' →enabled
 int1_data: map data ready interrupt to INT1 pin: '0'→disabled, or '1' →enabled

Register 0x1B (INT_MAP_2)

Controls which interrupt signals are mapped to the INT2 pin.

| Name | 0x1B | INT_MAP_2 | | |
|--------------------|-----------|-------------|------------|------------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | int2_flat | int2_orient | int2_s_tap | int2_d_tap |

| | | | | |
|--------------------|-----------------|------------|-----------|----------|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | int2_slo_no_mot | int2_slope | int2_high | int2_low |

| | |
|------------------|---|
| int2_flat: | map flat interrupt to INT2 pin: '0'→disabled, or '1' →enabled |
| int2_orient: | map orientation interrupt to INT2 pin: '0'→disabled, or '1' →enabled |
| int2_s_tap: | map single tap interrupt to INT2 pin: '0'→disabled, or '1' →enabled |
| int2_d_tap: | map double tap interrupt to INT2 pin: '0'→disabled, or '1' →enabled |
| int2_slo_no_mot: | map slow/no-motion interrupt to INT2 pin: '0'→disabled, or '1' →enabled |
| int2_slope: | map slope interrupt to INT2 pin: '0'→disabled, or '1' →enabled |
| int2_high: | map high-g to INT2 pin: '0'→disabled, or '1' →enabled |
| int2_low: | map low-g to INT2 pin: '0'→disabled, or '1' →enabled |

Register 0x1E (INT_SRC)

Contains the data source definition for interrupts with selectable data source.

| Name | 0x1E | INT_SRC | | |
|--------------------|----------|---------|--------------|-------------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | Reserved | | int_src_data | int_src_tap |

| | | | | |
|--------------------|--------------------|---------------|--------------|-------------|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | int_src_slo_no_mot | int_src_slope | int_src_high | int_src_low |

reserved: write '0'

int_src_data: select '0'→filtered, or '1' →unfiltered data for new data interrupt

int_src_tap: select '0'→filtered, or '1' →unfiltered data for single-/double tap interrupt

int_src_slo_no_mot: select '0'→filtered, or '1' →unfiltered data for slow/no-motion interrupt

int_src_slope: select '0'→filtered, or '1' →unfiltered data for slope interrupt

int_src_high: select '0'→filtered, or '1' →unfiltered data for high-g interrupt

int_src_low: select '0'→filtered, or '1' →unfiltered data for low-g interrupt

Register 0x20 (INT_OUT_CTRL)

Contains the behavioural configuration (electrical behaviour) of the interrupt pins.

| Name | 0x20 | INT_OUT_CTRL | | |
|-------------|----------|--------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | Reserved | | | |

| | | | | |
|-------------|---------|----------|---------|----------|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 1 | 0 | 1 |
| Content | int2_od | int2_lvl | int1_od | int1_lvl |

reserved: write '0'
 int2_od: select '0' → push-pull, or '1' → open drain behavior for INT2 pin
 int2_lvl: select '0' → active low, or '1' → active high level for INT2 pin
 int1_od: select '0' → push-pull, or '1' → open drain behavior for INT1 pin
 int1_lvl: select '0' → active low, or '1' → active high level for INT1 pin

Register 0x21 (INT_RST_LATCH)

Contains the interrupt reset bit and the interrupt mode selection.

| Name | 0x21 | INT_RST_LATCH | | |
|-------------|-----------|---------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reset_int | Reserved | | |

| | | | | |
|-------------|----------------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | latch_int<3:0> | | | |

reset_int: write '1' → clear any latched interrupts, or '0' → keep latched interrupts active

reserved: write '0'

latch_int<3:0>: '0000b' → non-latched, '0001b' → temporary, 250 ms,
 '0010b' → temporary, 500 ms, '0011b' → temporary, 1 s,
 '0100b' → temporary, 2 s, '0101b' → temporary, 4 s,
 '0110b' → temporary, 8 s, '0111b' → latched,
 '1000b' → non-latched, '1001b' → temporary, 250 μs,
 '1010b' → temporary, 500 μs, '1011b' → temporary, 1 ms,
 '1100b' → temporary, 12.5 ms, '1101b' → temporary, 25 ms,
 '1110b' → temporary, 50 ms, '1111b' → latched

Register 0x22 (INT_0)

Contains the delay time definition for the low-g interrupt.

| Name | 0x22 | INT_0 | | |
|-------------|--------------|-------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | low_dur<7:4> | | | |

| | | | | |
|-------------|--------------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 1 | 0 | 0 | 1 |
| Content | low_dur<3:0> | | | |

low_dur<7:0>: low-g interrupt trigger delay according to $[low_dur<7:0> + 1] \cdot 2$ ms in a range from 2 ms to 512 ms; the default corresponds to a delay of 20 ms.

Register 0x23 (INT_1)

Contains the threshold definition for the low-g interrupt.

| Name | 0x23 | INT_1 | | |
|-------------|-------------|-------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 1 | 1 |
| Content | low_th<7:4> | | | |

| | | | | |
|-------------|-------------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | low_th<3:0> | | | |

low_th<7:0>: low-g interrupt trigger threshold according to low_th<7:0> · 7.81 mg in a range from 0 g to 1.992 g; the default value corresponds to an acceleration of 375 mg

Register 0x24 (INT_2)

Contains the low-g interrupt mode selection, the low-g interrupt hysteresis setting, and the high-g interrupt hysteresis setting.

| Name | 0x24 | INT_2 | | |
|-------------|--------------|-------|----------|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 1 | 0 | 0 | 0 |
| Content | high_hy<1:0> | | reserved | |

| | | | | |
|-------------|----------|----------|-------------|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 1 |
| Content | reserved | low_mode | low_hy<1:0> | |

high_hy<1:0>: hysteresis of high-g interrupt according to high_hy<1:0> · 125 mg (2-g range), high_hy<1:0> · 250 mg (4-g range), high_hy<1:0> · 500 mg (8-g range), or high_hy<1:0> · 1000 mg (16-g range)

low_mode: select low-g interrupt '0' single-axis mode, or '1' axis-summing mode

low_hy<1:0>: hysteresis of low-g interrupt according to low_hy<1:0> · 125 mg independent of the selected accelerometer g-range

Register 0x25 (INT_3)

Contains the delay time definition for the high-g interrupt.

| Name | 0x25 | INT_3 | | |
|-------------|---------------|-------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | high_dur<7:4> | | | |

| | | | | |
|-------------|---------------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 1 | 1 | 1 | 1 |
| Content | high_dur<3:0> | | | |

high_dur<7:0>: high-g interrupt trigger delay according to $[high_dur<7:0> + 1] \cdot 2$ ms in a range from 2 ms to 512 ms; the default corresponds to a delay of 32 ms.

Register 0x26 (INT_4)

Contains the threshold definition for the high-g interrupt.

| Name | 0x26 | INT_4 | | |
|-------------|--------------|-------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 1 | 1 | 0 | 0 |
| Content | high_th<7:4> | | | |

| | | | | |
|-------------|--------------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | high_th<3:0> | | | |

high_th<7:0>: threshold of high-g interrupt according to $high_th<7:0> \cdot 7.81$ mg (2-g range), $high_th<7:0> \cdot 15.63$ mg (4-g range), $high_th<7:0> \cdot 31.25$ mg (8-g range), or $high_th<7:0> \cdot 62.5$ mg (16-g range)

Register 0x27 (INT_5)

Contains the definition of the number of samples to be evaluated for the slope interrupt (any-motion detection) and the slow/no-motion interrupt trigger delay.

| Name | 0x27 | INT_5 | | |
|-------------|---------------------|-------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | slo_no_mot_dur<5:2> | | | |

| | | | | |
|-------------|---------------------|-----|----------------|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | slo_no_mot_dur<1:0> | | slope_dur<1:0> | |

slo_no_mot_dur<5:0>: Function depends on whether the slow-motion or no-motion interrupt function has been selected. If the slow-motion interrupt function has been enabled (slo_no_mot_sel = '0') then [slo_no_mot_dur<1:0>+1] consecutive slope data points must be above the slow/no-motion threshold (slo_no_mot_th) for the slow-/no-motion interrupt to trigger. If the no-motion interrupt function has been enabled (slo_no_mot_sel = '1') then slo_no_motion_dur<5:0> defines the time for which no slope data points must exceed the slow/no-motion threshold (slo_no_mot_th) for the slow/no-motion interrupt to trigger. The delay time in seconds may be calculated according with the following equation:

$$\text{slo_no_mot_dur}\langle 5:4 \rangle = 'b00' \rightarrow [\text{slo_no_mot_dur}\langle 3:0 \rangle + 1]$$

$$\text{slo_no_mot_dur}\langle 5:4 \rangle = 'b01' \rightarrow [\text{slo_no_mot_dur}\langle 3:0 \rangle \cdot 4 + 20]$$

$$\text{slo_no_mot_dur}\langle 5 \rangle = '1' \rightarrow [\text{slo_no_mot_dur}\langle 4:0 \rangle \cdot 8 + 88]$$

slope_dur<1:0>: slope interrupt triggers if [slope_dur<1:0>+1] consecutive slope data points are above the slope interrupt threshold slope_th<7:0>

Register 0x28 (INT_6)

Contains the threshold definition for the any-motion interrupt.

| Name | 0x28 | INT_6 | | |
|-------------|---------------|-------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 1 |
| Content | slope_th<7:4> | | | |

| | | | | |
|-------------|---------------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 1 | 0 | 0 |
| Content | slope_th<3:0> | | | |

slope_th<7:0>: Threshold of the any-motion interrupt. It is range-dependent and defined as a sample-to-sample difference according to

- slope_th<7:0> · 3.91 mg (2-g range) /
- slope_th<7:0> · 7.81 mg (4-g range) /
- slope_th<7:0> · 15.63 mg (8-g range) /
- slope_th<7:0> · 31.25 mg (16-g range)

Register 0x29 (INT_7)

Contains the threshold definition for the slow/no-motion interrupt.

| Name | 0x29 | INT_7 | | |
|-------------|--------------------|-------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 1 |
| Content | slo_no_mot_th<7:4> | | | |

| | | | | |
|-------------|--------------------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 1 | 0 | 0 |
| Content | slo_no_mot_th<3:0> | | | |

slo_no_mot_th<7:0>: Threshold of slow/no-motion interrupt. It is range-dependent and defined as a sample-to-sample difference according to

- slo_no_mot_th<7:0> · 3.91 mg (2-g range),
- slo_no_mot_th<7:0> · 7.81 mg (4-g range),
- slo_no_mot_th<7:0> · 15.63 mg (8-g range),
- slo_no_mot_th<7:0> · 31.25 mg (16-g range)

**Register 0x2A (INT_8)**

Contains the timing definitions for the single tap and double tap interrupts.

| Name | 0x2A | INT_8 | | |
|--------------------|-----------|--------------|----------|----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | tap_quiet | tap_shock | reserved | reserved |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 1 | 0 | 0 |
| Content | reserved | tap_dur<2:0> | | |

tap_quiet: selects a tap quiet duration of '0' → 30 ms, '1' → 20 ms

tap_shock: selects a tap shock duration of '0' → 50 ms, '1' → 75 ms

reserved: write '0'

tap_dur<2:0>: selects the length of the time window for the second shock event for double tap detection according to '000b' → 50 ms, '001b' → 100 ms, '010b' → 150 ms, '011b' → 200 ms, '100b' → 250 ms, '101b' → 375 ms, '110b' → 500 ms, '111b' → 700 ms.

**Register 0x2B (INT_9)**

Contains the definition of the number of samples processed by the single / double-tap interrupt engine after wake-up in low-power mode. It also defines the threshold definition for the single and double tap interrupts.

| Name | 0x2B | INT_9 | | |
|-------------|---------------|----------|-----|-----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | tap_samp<1:0> | reserved | | tap_th<4> |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 1 | 0 | 1 | 0 |
| Content | tap_th<3:0> | | | |

tap_samp<1:0>: selects the number of samples that are processed after wake-up in the low-power mode according to '00b' → 2 samples, '01b' → 4 samples, '10b' → 8 samples, and '11b' → 16 samples

reserved: write '0'

tap_th<4:0>: threshold of the single/double-tap interrupt corresponding to an acceleration difference of tap_th<4:0> · 62.5mg (2g-range), tap_th<4:0> · 125mg (4g-range), tap_th<4:0> · 250mg (8g-range), and tap_th<4:0> · 500mg (16g-range).

**Register 0x2C (INT_A)**

Contains the definition of hysteresis, blocking, and mode for the orientation interrupt

| Name | 0x2C | INT_A | | |
|-------------|----------|------------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 1 |
| Content | reserved | orient_hyst<2:0> | | |

| Bit | 3 | 2 | 1 | 0 |
|-------------|----------------------|-----|------------------|-----|
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 1 | 0 | 0 | 0 |
| Content | orient_blocking<1:0> | | orient_mode<1:0> | |

reserved: write '0'

orient_hyst<2:0>: sets the hysteresis of the orientation interrupt; 1 LSB corresponds to 62.5 mg irrespective of the selected g-range

orient_blocking<1:0>: selects the blocking mode that is used for the generation of the orientation interrupt. The following blocking modes are available:

- '00b' → no blocking,
- '01b' → theta blocking or acceleration in any axis > 1.5g,
- '10b' → ,theta blocking or acceleration slope in any axis > 0.2 g or acceleration in any axis > 1.5g
- '11b' → theta blocking or acceleration slope in any axis > 0.4 g or acceleration in any axis > 1.5g and value of orient is not stable for at least 100ms

orient_mode<1:0>: sets the thresholds for switching between the different orientations. The settings: '00b' → symmetrical, '01b' → high-asymmetrical, '10b' → low-asymmetrical, '11b' → symmetrical.

Register 0x2D (INT_B)

Contains the definition of the axis orientation, up/down masking, and the theta blocking angle for the orientation interrupt.

| Name | 0x2D | INT_B | | |
|-------------|-------------------|--------------|-------------------|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | n/a | 1 | 0 | 0 |
| Content | reserved | orient_ud_en | orient_theta<5:4> | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 1 | 0 | 0 | 0 |
| Content | orient_theta<3:0> | | | |

orient_ud_en: change of up/down-bit '1' → generates an orientation interrupt, '0' → is ignored and will not generate an orientation interrupt

orient_theta<5:0>: defines a blocking angle between 0° and 44.8°

Register 0x2E (INT_C)

Contains the definition of the flat threshold angle for the flat interrupt.

| Name | 0x2E | INT_C | | |
|-------------|-----------------|-----------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | n/a | n/a | 0 | 0 |
| Content | Reserved | flat_theta<5:4> | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 1 | 0 | 0 | 0 |
| Content | flat_theta<3:0> | | | |

reserved: write '0'

flat_theta<5:0>: defines threshold for detection of flat position in range from 0° to 44.8°.

**Register 0x2F (INT_D)**

Contains the definition of the flat interrupt hold time and flat interrupt hysteresis.

| Name | 0x2F | INT_D | | |
|-------------|----------|-------|---------------------|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 1 |
| Content | Reserved | | flat_hold_time<1:0> | |

| Bit | 3 | 2 | 1 | 0 |
|-------------|----------|--------------|-----|-----|
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 1 |
| Content | reserved | flat_hy<2:0> | | |

reserved: write '0'

flat_hold_time<1:0>: delay time for which the flat value must remain stable for the flat interrupt to be generated: '00b' → 0 ms, '01b' → 512 ms, '10b' → 1024 ms, '11b' → 2048 ms

flat_hy<2:0>: defines flat interrupt hysteresis; flat value must change by more than twice the value of flat interrupt hysteresis to detect a state change. For details see chapter 4.7.8.

'000b' → hysteresis of the flat detection disabled

**Register 0x30 (FIFO_CONFIG_0)**

Contains the FIFO watermark level.

| Name | 0x30 | FIFO_CONFIG_0 | | |
|-------------|----------|---------------|---|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | n/a | n/a | 0 | 0 |
| Content | Reserved | | fifo_water_mark_level_trigger_retain<5:4> | |

| | | | | |
|-------------|---|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | fifo_water_mark_level_trigger_retain<3:0> | | | |

reserved: write '0'

fifo_water_mark_level_trigger_retain<5:0>:

fifo_water_mark_level_trigger_retain<5:0> defines the FIFO watermark level.
An interrupt will be generated, when the number of entries in the FIFO is equal to fifo_water_mark_level_trigger_retain<5:0>;

**Register 0x32 (PMU_SELF_TEST)**

Contains the settings for the sensor self-test configuration and trigger.

| Name | 0x32 | PMU_SELF_TEST | | |
|-------------|----------|---------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | |

| | | | | |
|-------------|------------|----------------|---------------------|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved_0 | self_test_sign | self_test-axis<1:0> | |

reserved: write '0x0'

self_test_sign: select sign of self-test excitation as '1' → positive, or '0' → negative

self_test_axis: select axis to be self-tested: '00b' → self-test disabled, '01b' → x-axis, '10b' → y-axis, or '11b' → z-axis; when a self-test is performed, only the acceleration data readout value of the selected axis is valid; after the self-test has been enabled a delay of a least 50 ms is necessary for the read-out value to settle

Register 0x33 (TRIM_NVM_CTRL)

Contains the control settings for the few-time programmable non-volatile memory (NVM).

| Name | 0x33 | TRIM_NVM_CTRL | | |
|-------------|-----------------|---------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | nvm_remain<3:0> | | | |

| | | | | |
|-------------|----------|---------|---------------|---------------|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R | W | R/W |
| Reset Value | 0 | n/a | 0 | 0 |
| Content | nvm_load | nvm_rdy | nvm_prog_trig | nvm_prog_mode |

nvm_remain<3:0>: number of remaining write cycles permitted for NVM; the number is decremented each time a write to the NVM is triggered

nvm_load: '1' → trigger, or '0' → do not trigger an update of all configuration registers from NVM; the nvm_rdy flag must be '1' prior to triggering the update

nvm_rdy: status of NVM controller: '0' → NVM write / NVM update operation is in progress, '1' → NVM is ready to accept a new write or update trigger

nvm_prog_trig: '1' → trigger, or '0' → do not trigger an NVM write operation; the trigger is only accepted if the NVM was unlocked before and nvm_remain<3:0> is greater than '0'; flag nvm_rdy must be '1' prior to triggering the write cycle

nvm_prog_mode: '1' → unlock, or '0' → lock NVM write operation

**Register 0x34 (BGW_SPI3_WDT)**

Contains settings for the digital interfaces.

| Name | 0x34 | BGW_SPI3_WDT | | |
|-------------|----------|--------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | Reserved | | | |

| | | | | |
|-------------|----------|------------|-------------|------|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | i2c_wdt_en | i2c_wdt_sel | spi3 |

reserved: write '0'

i2c_wdt_en: if I²C interface mode is selected then '1' → enable, or '0' → disables the watchdog at the SDI pin (= SDA for I²C)i2c_wdt_sel: select an I²C watchdog timer period of '0' → 1 ms, or '1' → 50 ms

spi3: select '0' → 4-wire SPI, or '1' → 3-wire SPI mode

Register 0x36 (OFC_CTRL)

Contains control signals and configuration settings for the fast and the slow offset compensation.

| Name | 0x36 | OFC_CTRL | | |
|-------------|--------------|------------------|---------|---------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | W | W | W | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | offset_reset | cal_trigger<1:0> | | cal_rdy |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | hp_z_en | hp_y_en | hp_x_en |

offset_reset: '1' → set all offset compensation registers (0x38 to 0x3A) to zero, or '0' → keep their values

offset_trigger<1:0>: trigger fast compensation for '01b' → x-axis, '10b' → y-axis, or '11b' → z-axis; '00b' → do not trigger offset compensation; offset compensation must not be triggered when cal_rdy is '0'

cal_rdy: indicates the state of the fast compensation: '0' → offset compensation is in progress, or '1' → offset compensation is ready to be retriggered

reserved: write '0'

hp_z_en: '1' → enable, or '0' → disable slow offset compensation for the z-axis

hp_y_en: '1' → enable, or '0' → disable slow offset compensation for the y-axis

hp_x_en: '1' → enable, or '0' → disable slow offset compensation for the x-axis

Register 0x37 (OFC_SETTING)

Contains configuration settings for the fast and the slow offset compensation.

| Name | 0x37 | OFC_SETTING | | |
|-------------|--------------------|----------------------|-----|--------------------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | offset_target_z<1:0> | | offset_target_y<1> |
| | | | | > |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | offset_target_y<0> | offset_target_x<1:0> | | cut_off |
| | > | | | |

reserved: write '0'

offset_target_z<1:0>: offset compensation target value for z-axis is '00b' → 0 g, '01b' → +1 g, '10b' → -1 g, or '11b' → 0 g

offset_target_y<1:0>: offset compensation target value for y-axis is '00b' → 0 g, '01b' → +1 g, '10b' → -1 g, or '11b' → 0 g

offset_target_x<1:0>: offset compensation target value for x-axis is '00b' → 0 g, '01b' → +1 g, '10b' → -1 g, or '11b' → 0 g

cut_off:

| (0x37) cut_off | high-pass filter bandwidth | Example bw = 500 Hz |
|-------------------|--|--|
| 0b | $\frac{1\text{Hz} \times bw}{1000\text{ Hz}}$ | $\frac{1\text{Hz} \times 500\text{ Hz}}{1000\text{ Hz}} = 0.5\text{ Hz}$ |
| 1b | $\frac{10\text{Hz} \times bw}{1000\text{ Hz}}$ | $\frac{10\text{Hz} \times 500\text{ Hz}}{1000\text{ Hz}} = 5\text{ Hz}$ |

*bw: please insert selected decimal data bandwidth value [Hz] from table 4

Register 0x38 (OFC_OFFSET_X)

Contains the offset compensation value for x-axis acceleration readout data.

| Name | 0x38 | OFC_OFFSET_X | | |
|-------------|---------------|--------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | offset_x<7:4> | | | |

| | | | | |
|-------------|---------------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | offset_x<3:0> | | | |

offset_x<7:0>: offset value, which is subtracted from the internal filtered and unfiltered x-axis acceleration data; the offset value is represented with two's complement notation, with a mapping of +127 → +0.992g, 0 → 0 g, and -128 → -1 g; the scaling is independent of the selected g-range; the content of the offset_x<7:0> may be written to the NVM; it is automatically restored from the NVM after each power-on or softreset; offset_x<7:0> may be written directly by the user; it is generated automatically after triggering the fast offset compensation procedure for the x-axis

**Register 0x39 (OFC_OFFSET_Y)**

Contains the offset compensation value for y-axis acceleration readout data.

| Name | 0x39 | OFC_OFFSET_Y | | |
|-------------|---------------|--------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | offset_y<7:4> | | | |

| | | | | |
|-------------|---------------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | offset_y<3:0> | | | |

offset_y<7:0>: offset value, which is subtracted from the internal filtered and unfiltered y-axis acceleration data; the offset value is represented with two's complement notation, with a mapping of +127 → +0.992g, 0 → 0 g, and -128 → -1 g; the scaling is independent of the selected g-range; the content of the offset_y<7:0> may be written to the NVM; it is automatically restored from the NVM after each power-on or softreset; offset_y<7:0> may be written directly by the user; it is generated automatically after triggering the fast offset compensation procedure for the y-axis

Register 0x3A (OFC_OFFSET_Z)

Contains the offset compensation value for z-axis acceleration readout data.

| Name | 0x3A | OFC_OFFSET_Z | | |
|-------------|---------------|--------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | offset_z<7:4> | | | |

| | | | | |
|-------------|---------------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | offset_z<3:0> | | | |

offset_z<7:0>: offset value, which is subtracted from the internal filtered and unfiltered z-axis acceleration data; the offset value is represented with two's complement notation, with a mapping of +127 → +0.992g, 0 → 0 g, and -128 → -1 g; the scaling is independent of the selected g-range; the content of the offset_z<7:0> may be written to the NVM; it is automatically restored from the NVM after each power-on or softreset; offset_z<7:0> may be written directly by the user; it is generated automatically after triggering the fast offset compensation procedure for the z-axis

Register 0x3B (TRIM_GP0)

Contains general purpose data register with NVM back-up.

| Name | 0x3B | TRIM_GP0 | | |
|-------------|----------|----------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | GP0<7:4> | | | |

| | | | | |
|-------------|----------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | GP0<3:0> | | | |

GP0<7:0>: general purpose NVM image register not linked to any sensor-specific functionality; register may be written to NVM and is restored after each power-up or softreset

**Register 0x3C (TRIM_GP1)**

Contains general purpose data register with NVM back-up.

| Name | 0x3C | TRIM_GP1 | | |
|-------------|----------|----------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | GP1<7:4> | | | |

| | | | | |
|-------------|----------|-----|-----|-----|
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | GP1<3:0> | | | |

GP1<7:0>: general purpose NVM image register not linked to any sensor-specific functionality; register may be written to NVM and is restored after each power-up or softreset

**Register 0x3E (FIFO_CONFIG_1)**

Contains FIFO configuration settings. The FIFO buffer memory is cleared and the fifo-full flag is cleared when writing to FIFO_CONFIG_1 register.

| Name | 0x3E | FIFO_CONFIG_1 | | |
|-------------|----------------|---------------|-----------------------|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | fifo_mode<1:0> | | Reserved | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R/W | R/W | R/W | R/W |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | Reserved | | fifo_data_select<1:0> | |

fifo_mode<1:0>: selects the FIFO operating mode:
 '00b' → BYPASS (buffer depth of 1 frame; old data is discarded),
 '01b' → FIFO (data collection stops when buffer is filled with 32 frames),
 '10b' → STREAM (sampling continues when buffer is full; old is discarded),
 '11b' → reserved, do not use

fifo_data_select<1:0>: selects whether '00b' → X+Y+Z, '01b' → X only, '10b' → Y only,
 '11b' → Z only acceleration data are stored in the FIFO

Register 0x3F (FIFO_DATA)

FIFO data readout register. The format of the LSB and MSB components corresponds to that of the acceleration data readout registers. The new data flag is preserved. Read burst access may be used since the address counter will not increment when the read burst is started at the address of FIFO_DATA. The entire frame is discarded when a frame is only partially read out.

| Name | 0x3F | FIFO_DATA | | |
|-------------|--------------------------------|-----------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | fifo_data_output_register<7:4> | | | |

| Bit | 3 | 2 | 1 | 0 |
|-------------|--------------------------------|-----|-----|-----|
| Read/Write | R | R | R | R |
| Reset Value | n/a | n/a | n/a | n/a |
| Content | fifo_data_output_register<3:0> | | | |

fifo_data_output_register<7:0>: FIFO data readout; data format depends on the setting of register fifo_data_select<1:0>:

if X+Y+Z data are selected, the data of frame n is reading out in the order of X-lsb(n), X-msb(n), Y-lsb(n), Y-msb(n), Z-lsb(n), Z-msb(n);
 if X-only is selected, the data of frame n and n+1 are reading out in the order of X-lsb(n), X-msb(n), X-lsb(n+1), X-msb(n+1); the Y-only and Z-only modes behave analogously

7. Digital interfaces

The BMA280 supports two serial digital interface protocols for communication as a slave with a host device (when operating in general mode): SPI and I²C. The active interface is selected by the state of the Pin#11 (PS) 'protocol select' pin: '0' ('1') selects SPI (I²C). For details please refer to section 8).

By default, SPI operates in the standard 4-wire configuration. It can be re-configured by software to work in 3-wire mode instead of standard 4-wire mode.

Both interfaces share the same pins. The mapping for each interface is given in the following table:

Table 20: Mapping of the interface pins

| Pin# | Name | use w/ SPI | use w/ I ² C | Description |
|------|------|---------------|----------------------------|--|
| 1 | SDO | SDO | address | SPI: Data Output (4-wire mode) I ² C: Used to set LSB of I ² C address |
| 2 | SDx | SDI | SDA | SPI: Data Input (4-wire mode) Data Input / Output (3-wire mode) I ² C: Serial Data |
| 10 | CSB | CSB | unused | Chip Select (enable) |
| 12 | SCx | SCK | SCL | SPI: Serial Clock I ² C: Serial Clock |

The following table shows the electrical specifications of the interface pins:

Table 21: Electrical specification of the interface pins

| Parameter | Symbol | Condition | Min | Typ | Max | Units |
|---|-----------------------|--------------------------------------|-----|-----|-----|-------|
| Pull-up Resistance, CSB pin | R _{up} | Internal Pull-up Resistance to VDDIO | 75 | 100 | 125 | kΩ |
| Input Capacitance | C _{in} | | | 5 | 10 | pF |
| I ² C Bus Load Capacitance (max. drive capability) | C _{I2C_Load} | | | | 400 | pF |

7.1 Serial peripheral interface (SPI)

The timing specification for SPI of the BMA280 is given in the following table:

Table 22: SPI timing

| Parameter | Symbol | Condition | Min | Max | Units |
|---|------------------------------|---|-----|-----|---------------|
| Clock Frequency | f_{SPI} | Max. Load on SDI or SDO = 25pF, $V_{\text{DDIO}} \geq 1.62\text{V}$ | | 10 | MHz |
| | | $V_{\text{DDIO}} < 1.62\text{V}$ | | 7.5 | MHz |
| SCK Low Pulse | t_{SCKL} | | 20 | | ns |
| SCK High Pulse | t_{SCKH} | | 20 | | ns |
| SDI Setup Time | $t_{\text{SDI_setup}}$ | | 20 | | ns |
| SDI Hold Time | $t_{\text{SDI_hold}}$ | | 20 | | ns |
| SDO Output Delay | $t_{\text{SDO_OD}}$ | Load = 25pF, $V_{\text{DDIO}} \geq 1.62\text{V}$ | | 30 | ns |
| | | Load = 25pF, $V_{\text{DDIO}} < 1.62\text{V}$ | | 50 | ns |
| | | Load = 250pF, $V_{\text{DDIO}} > 2.4\text{V}$ | | 40 | ns |
| CSB Setup Time | $t_{\text{CSB_setup}}$ | | 20 | | ns |
| CSB Hold Time | $t_{\text{CSB_hold}}$ | | 40 | | ns |
| Idle time between write accesses, normal mode, standby mode, low-power mode 2 | $t_{\text{IDLE_wacc_nm}}$ | | 2 | | μs |
| Idle time between write accesses, suspend mode, low-power mode 1 | $t_{\text{IDLE_wacc_sum}}$ | | 450 | | μs |

The following figure shows the definition of the SPI timings given in the following figure:

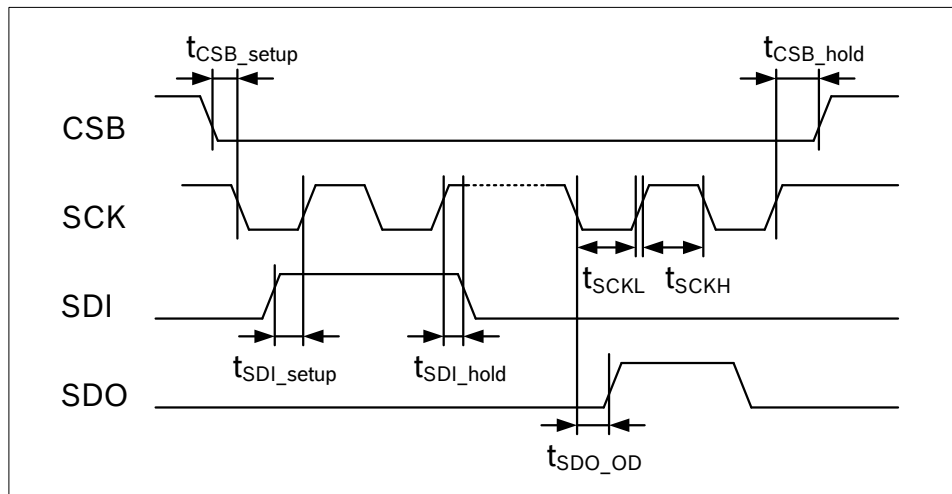


Figure 13: SPI timing diagram

The SPI interface of the BMA280 is compatible with two modes, '00' and '11'. The automatic selection between [CPOL = '0' and CPHA = '0'] and [CPOL = '1' and CPHA = '1'] is controlled based on the value of SCK after a falling edge of CSB.

Two configurations of the SPI interface are supported by the BMA280: 4-wire and 3-wire. The same protocol is used by both configurations. The device operates in 4-wire configuration by default. It can be switched to 3-wire configuration by writing '1' to (0x34) spi3. Pin SDI is used as the common data pin in 3-wire configuration.

For single byte read as well as write operations, 16-bit protocols are used. The BMA280 also supports multiple-byte read operations.

In SPI 4-wire configuration CSB (chip select low active), SCK (serial clock), SDI (serial data input), and SDO (serial data output) pins are used. The communication starts when the CSB is pulled low by the SPI master and stops when CSB is pulled high. SCK is also controlled by SPI master. SDI and SDO are driven at the falling edge of SCK and should be captured at the rising edge of SCK.

The basic write operation waveform for 4-wire configuration is depicted in figure 14. During the entire write cycle SDO remains in high-impedance state.

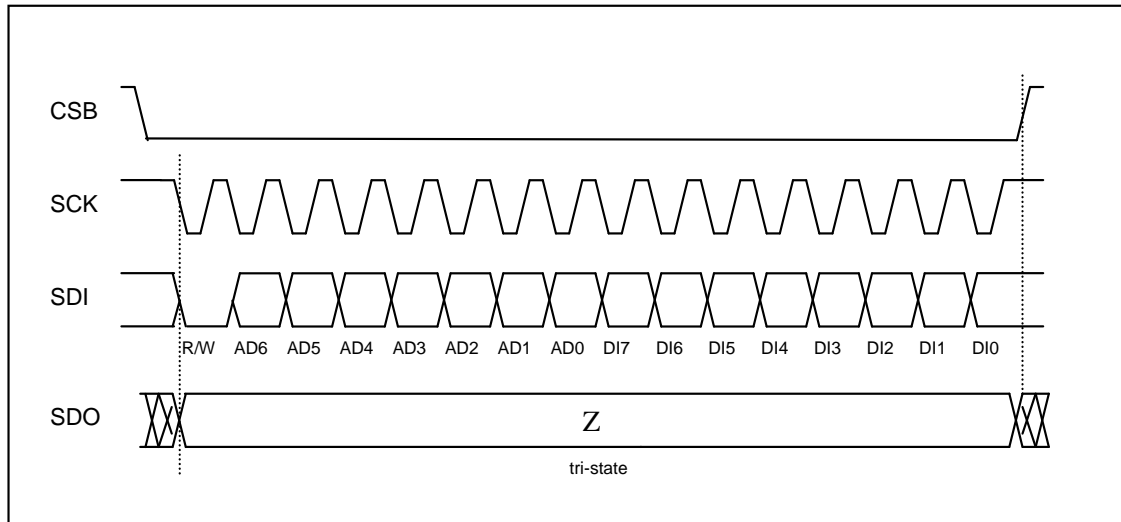


Figure 14: 4-wire basic SPI write sequence (mode '11')

The basic read operation waveform for 4-wire configuration is depicted in figure 15:

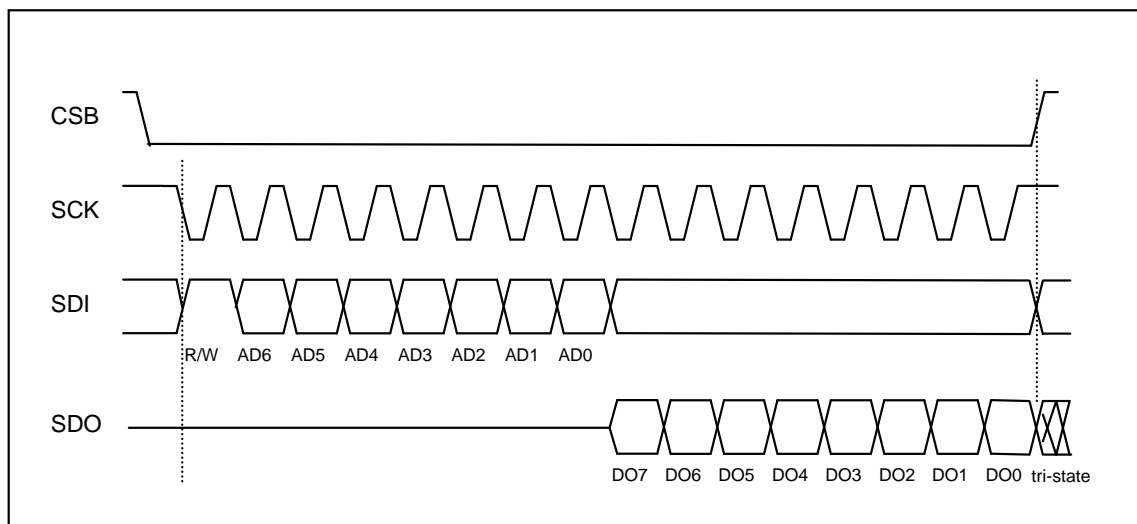


Figure 15: 4-wire basic SPI read sequence (mode '11')

The data bits are used as follows:

Bit0: Read/Write bit. When 0, the data SDI is written into the chip. When 1, the data SDO from the chip is read.

Bit1-7: Address AD(6:0).

Bit8-15: when in write mode, these are the data SDI, which will be written into the address. When in read mode, these are the data SDO, which are read from the address.

Multiple read operations are possible by keeping CSB low and continuing the data transfer. Only the first register address has to be written. Addresses are automatically incremented after each read access as long as CSB stays active low.

The principle of multiple read is shown in figure 16:

| Start | | Control byte | | | | | | Data byte | | | | | | Data byte | | | | | | Data byte | | | | | | Stop | | | | | |
|---------|---|------------------------|---|---|---|---|---|----------------------------|---|---|---|---|---|----------------------------|---|---|---|---|---|----------------------------|---|---|---|---|---|------|---|---|---|---|---------|
| RW | | Register address (02h) | | | | | | Data register - adress 02h | | | | | | Data register - adress 03h | | | | | | Data register - adress 04h | | | | | | | | | | | |
| CSB = 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | CSB = 1 |

Figure 16: SPI multiple read

In SPI 3-wire configuration CSB (chip select low active), SCK (serial clock), and SDI (serial data input and output) pins are used. The communication starts when the CSB is pulled low by the SPI master and stops when CSB is pulled high. SCK is also controlled by SPI master. SDI is driven (when used as input of the device) at the falling edge of SCK and should be captured (when used as the output of the device) at the rising edge of SCK.

The protocol as such is the same in 3-wire configuration as it is in 4-wire configuration. The basic operation waveform (read or write access) for 3-wire configuration is depicted in figure 17:

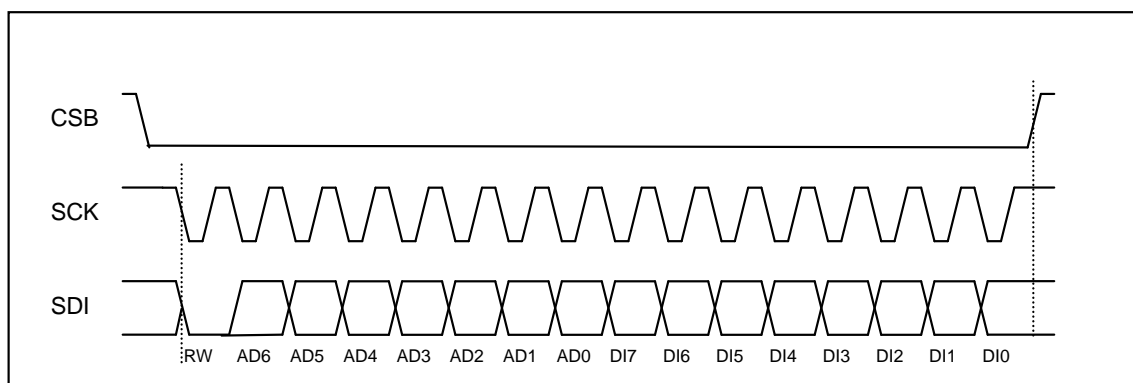


Figure 17: 3-wire basic SPI read or write sequence (mode '11')

7.2 Inter-Integrated Circuit (I²C)

The I²C bus uses SCL (= SCx pin, serial clock) and SDA (= SDx pin, serial data input and output) signal lines. Both lines are connected to V_{DDIO} externally via pull-up resistors so that they are pulled high when the bus is free.

The I²C interface of the BMA280 is compatible with the I²C Specification UM10204 Rev. 03 (19 June 2007), available at <http://www.nxp.com>. The BMA280 supports I²C standard mode and fast mode, only 7-bit address mode is supported.

The default I²C address of the device is 0011000b (0x18). It is used if the SDO pin is pulled to 'GND'. The alternative address 0011001b (0x19) is selected by pulling the SDO pin to 'V_{DDIO}'.

The timing specification for I²C of the BMA280 is given in Table 23:

Table 23: I²C timings

| Parameter | Symbol | Condition | Min | Max | Units |
|---|---------------------------|-----------|-----|-----|-------|
| Clock Frequency | f _{SCL} | | | 400 | kHz |
| SCL Low Period | t _{LOW} | | 1.3 | | μs |
| SCL High Period | t _{HIGH} | | 0.6 | | |
| SDA Setup Time | t _{SUDAT} | | 0.1 | | |
| SDA Hold Time | t _{HDDAT} | | 0.0 | | |
| Setup Time for a repeated Start Condition | t _{SUSTA} | | 0.6 | | |
| Hold Time for a Start Condition | t _{HDSTA} | | 0.6 | | |
| Setup Time for a Stop Condition | t _{SUSTO} | | 0.6 | | |
| Time before a new Transmission can start | t _{BUF} | | 1.3 | | |
| Idle time between write accesses, normal mode, standby mode, low-power mode 2 | t _{IDLE_wacc_nm} | | 2 | | μs |
| Idle time between write accesses, suspend mode, low-power mode 1 | t _{IDLE_wacc_s} | | 450 | | μs |

Figure 18 shows the definition of the I²C timings given in Table 23:

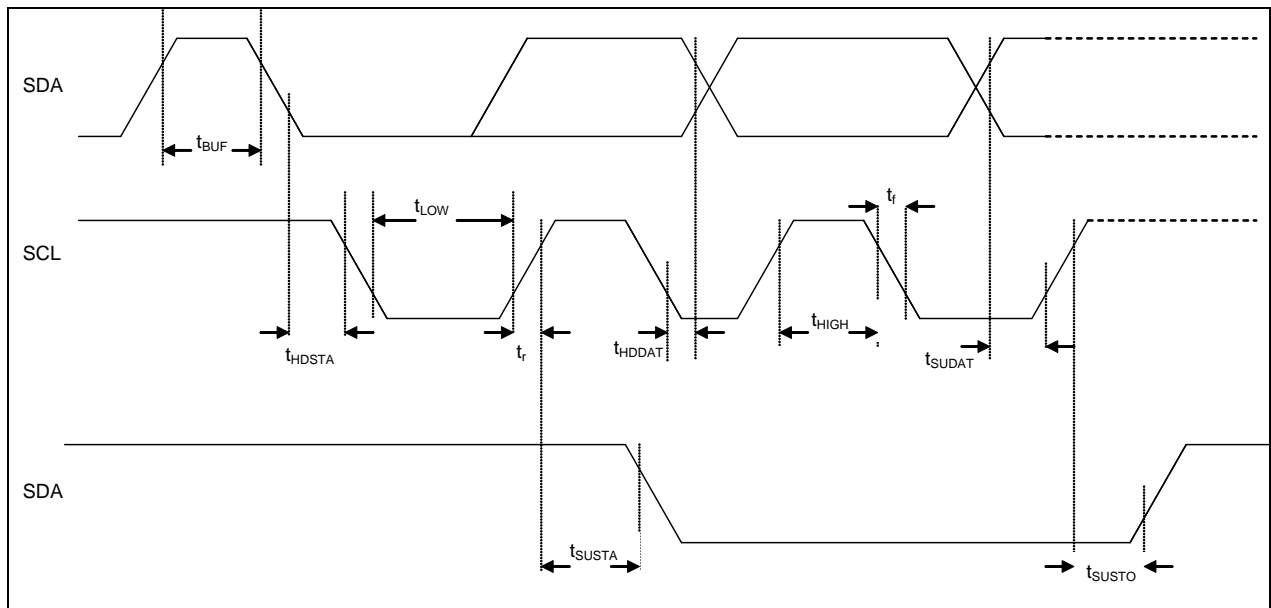


Figure 18: I²C timing diagram

The I²C protocol works as follows:

START: Data transmission on the bus begins with a high to low transition on the SDA line while SCL is held high (start condition (S) indicated by I²C bus master). Once the START signal is transferred by the master, the bus is considered busy.

STOP: Each data transfer should be terminated by a Stop signal (P) generated by master. The STOP condition is a low to HIGH transition on SDA line while SCL is held high.

ACK: Each byte of data transferred must be acknowledged. It is indicated by an acknowledge bit sent by the receiver. The transmitter must release the SDA line (no pull down) during the acknowledge pulse while the receiver must then pull the SDA line low so that it remains stable low during the high period of the acknowledge clock cycle.

In the following diagrams these abbreviations are used:

| | |
|-------|---------------------------|
| S | Start |
| P | Stop |
| ACKS | Acknowledge by slave |
| ACKM | Acknowledge by master |
| NACKM | Not acknowledge by master |
| RW | Read / Write |

A START immediately followed by a STOP (without SCK toggling from logic “1” to logic “0”) is not supported. If such a combination occurs, the STOP is not recognized by the device.

I²C write access:

I²C write access can be used to write a data byte in one sequence.

The sequence begins with start condition generated by the master, followed by 7 bits slave address and a write bit (RW = 0). The slave sends an acknowledge bit (ACK = 0) and releases the bus. Then the master sends the one byte register address. The slave again acknowledges the transmission and waits for the 8 bits of data which shall be written to the specified register address. After the slave acknowledges the data byte, the master generates a stop signal and terminates the writing protocol.

Example of an I²C write access:

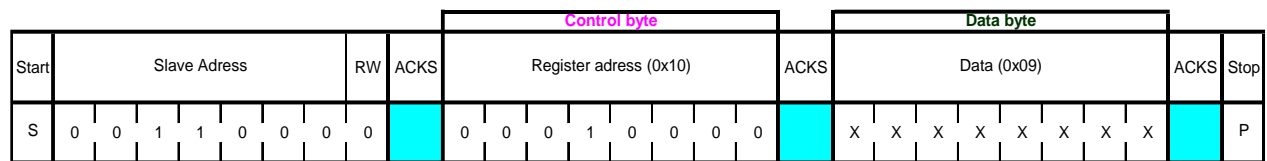


Figure 19: I²C write

I²C read access:

I²C read access also can be used to read one or multiple data bytes in one sequence.

A read sequence consists of a one-byte I²C write phase followed by the I²C read phase. The two parts of the transmission must be separated by a repeated start condition (Sr). The I²C write phase addresses the slave and sends the register address to be read. After slave acknowledges the transmission, the master generates again a start condition and sends the slave address together with a read bit (RW = 1). Then the master releases the bus and waits for the data bytes to be read out from slave. After each data byte the master has to generate an acknowledge bit (ACK = 0) to enable further data transfer. A NACKM (ACK = 1) from the master stops the data being transferred from the slave. The slave releases the bus so that the master can generate a STOP condition and terminate the transmission.

The register address is automatically incremented and, therefore, more than one byte can be sequentially read out. Once a new data read transmission starts, the start address will be set to the register address specified in the latest I²C write command. By default the start address is set at 0x00. In this way repetitive multi-bytes reads from the same starting address are possible.

In order to prevent the I²C slave of the device to lock-up the I²C bus, a watchdog timer (WDT) is implemented. The WDT observes internal I²C signals and resets the I²C interface if the bus is locked-up by the BMA280. The activity and the timer period of the WDT can be configured through the bits (0x34) *i2c_wdt_en* and (0x34) *i2c_wdt_sel*.

Writing '1' ('0') to (0x34) *i2c_wdt_en* activates (de-activates) the WDT. Writing '0' ('1') to (0x34) *i2c_wdt_se* selects a timer period of 1 ms (50 ms).

Example of an I²C read access:

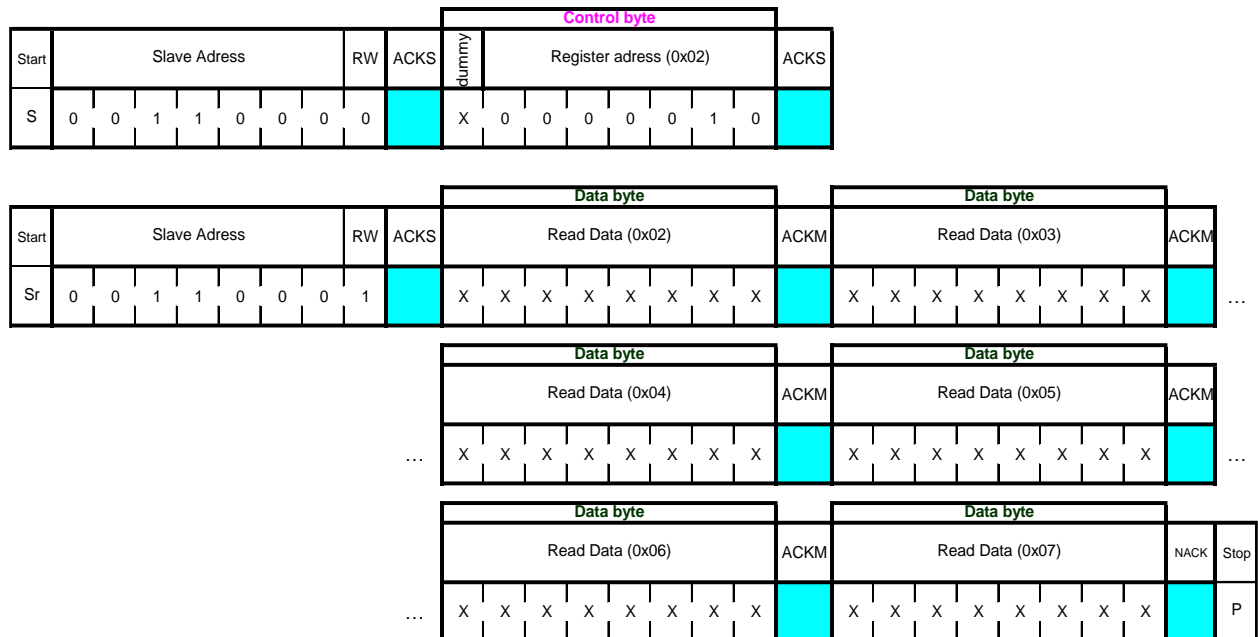


Figure 20: I²C multiple read

7.2.1 SPI and I²C Access Restrictions

In order to allow for the correct internal synchronisation of data written to the BMA280, certain access restrictions apply for consecutive write accesses or a write/read sequence through the SPI as well as I²C interface. The required waiting period depends on whether the device is operating in normal mode (or standby mode, or low-power mode 2) or suspend mode (or low-power mode 1).

As illustrated in figure 21, an interface idle time of at least 2 μ s is required following a write operation when the device operates in normal mode (or standby mode, or low-power mode 2). In suspend mode (or low-power mode 1) an interface idle time of at least 450 μ s is required.

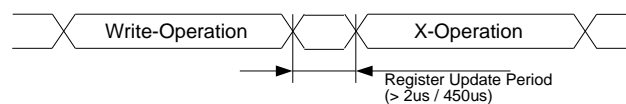


Figure 21: Post-Write Access Timing Constraints

8. Pin-out and connection diagram

8.1 Pin-out

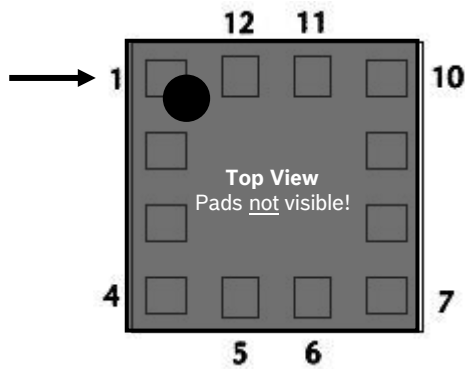


Figure 22: Pin-out top view

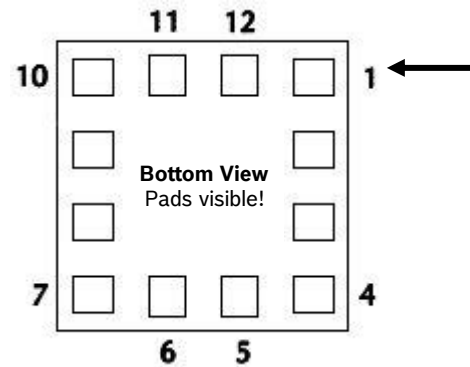


Figure 23: Pin-out bottom view

Table 24: Pin description

| Pin# | Name | I/O Type | Description | Connect to | | |
|------|-------|-------------|---|-------------------|-------------------|--------------------------|
| | | | | in SPI 4W | In SPI 3W | in I ² C |
| 1 | SDO | Digital out | Serial data output in SPI Address select in I ² C mode see chapter 7.2 | SDO | DNC (float) | GND for default addr. |
| 2 | SDx | Digital I/O | SDA serial data I/O in I ² C SDI serial data input in SPI 4W SDA serial data I/O in SPI 3W | SDI | SDA | SDA |
| 3 | VDDIO | Supply | Digital I/O supply voltage (1.2V ... 3.6V) | V _{DDIO} | V _{DDIO} | V _{DDIO} |
| 4 | NC | -- | | GND | GND | GND |
| 5 | INT1 | Digital out | Interrupt output 1 * | INT1 | INT1 | INT1 |
| 6 | INT2 | Digital out | Interrupt output 2 * | INT2 | INT2 | INT2 |
| 7 | VDD | Supply | Power supply for analog & digital domain (1.62V ... 3.6V) | V _{DD} | V _{DD} | V _{DD} |
| 8 | GNDIO | Ground | Ground for I/O | GND | GND | GND |
| 9 | GND | Ground | Ground for digital & analog | GND | GND | GND |
| 10 | CSB | Digital in | Chip select for SPI mode | CSB | CSB | DNC (float) |
| 11 | PS | Digital in | Protocol select (GND = SPI, V _{DDIO} = I ² C) | GND | GND | V _{DDIO} |
| 12 | SCx | Digital in | SCK for SPI serial clock SCL for I ² C serial clock | SCK | SCK | SCL |

* If INT1 and/or INT2 are not used, please do not connect them (DNC).

8.2 Connection diagram 4-wire SPI

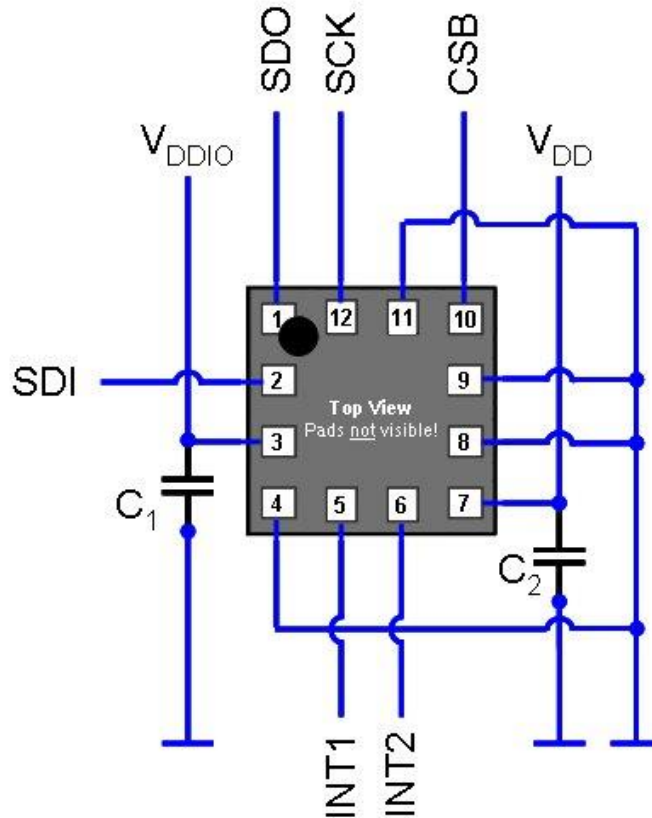


Figure 24: 4-wire SPI connection

Note: the recommended value for C_1 , C_2 is 100 nF.

8.3 Connection diagram 3-wire SPI

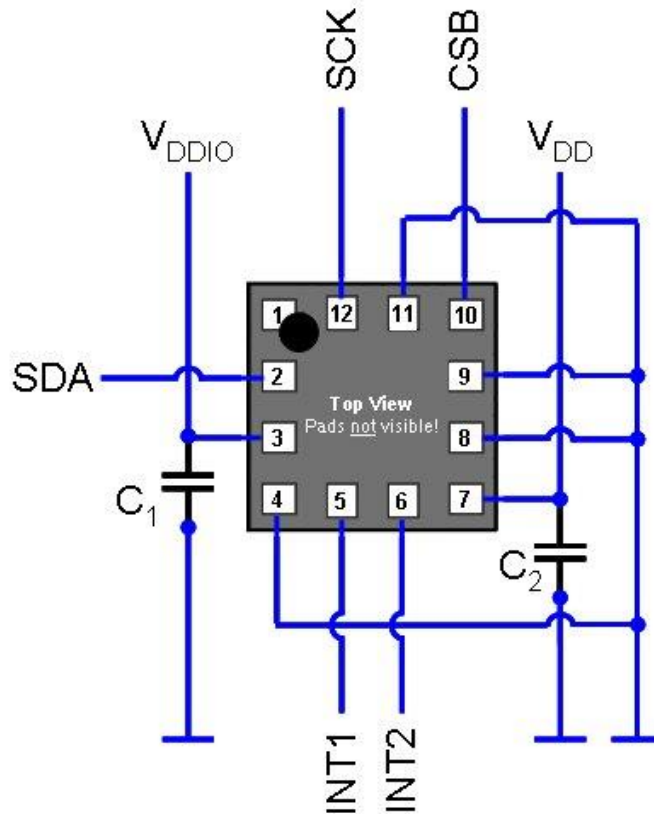


Figure 25: 3-wire SPI connection

Note: the recommended value for C_1 , C_2 is 100 nF.

8.4 Connection diagram I²C

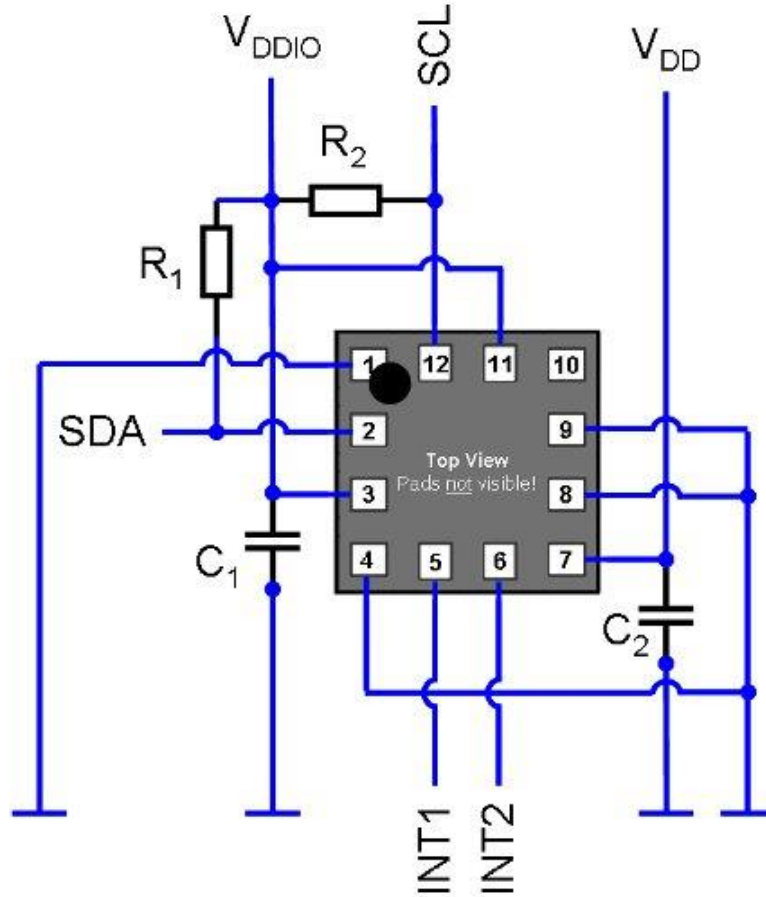


Figure 26: I²C connection

Note: the recommended value for C_1 , C_2 is 100 nF.

9. Package

9.1 Outline dimensions

The sensor housing is a standard LGA package. Its dimensions are the following.

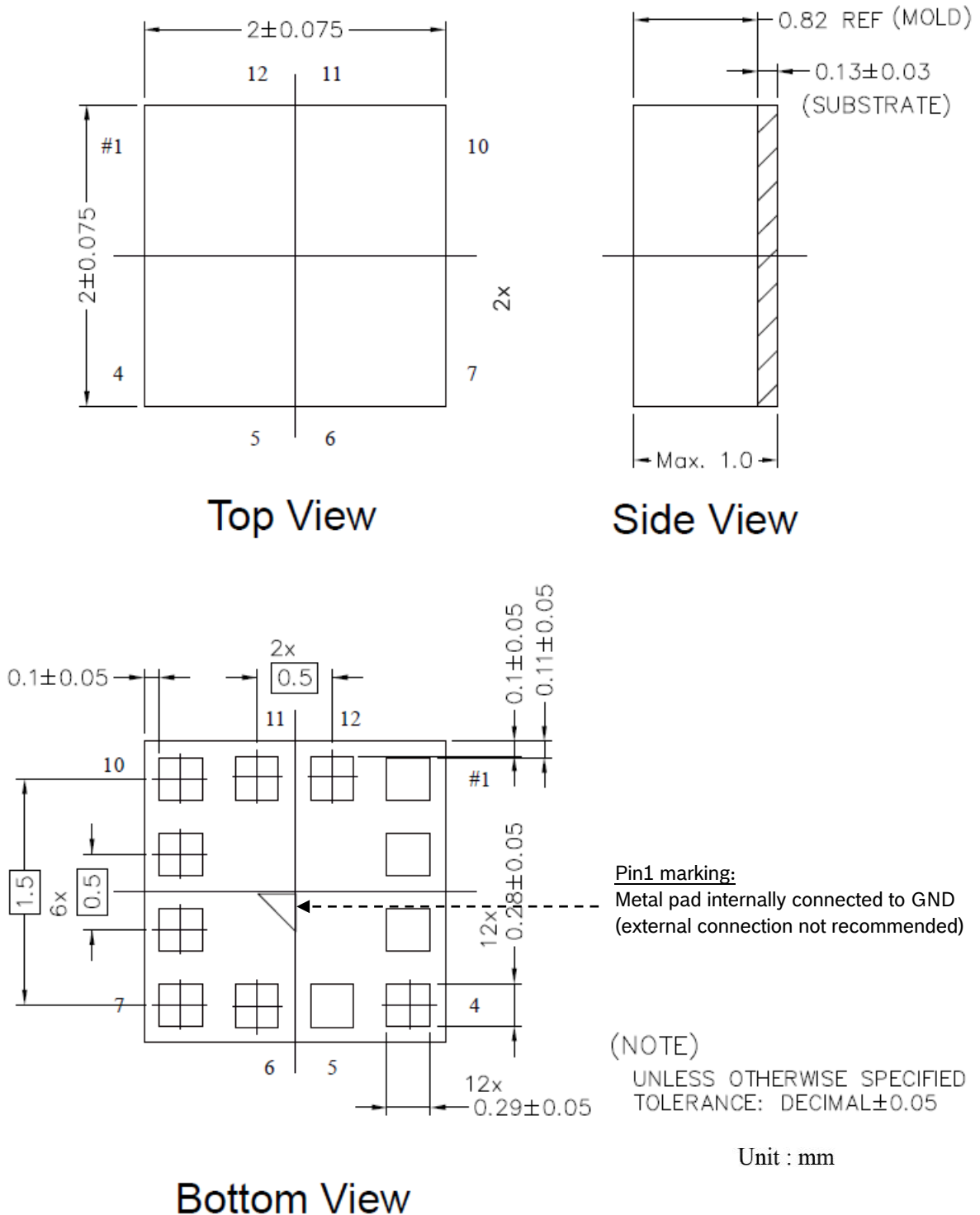


Figure 27: Package outline dimensions

9.2 Sensing axes orientation

If the sensor is accelerated in the indicated directions, the corresponding channel will deliver a positive acceleration signal (dynamic acceleration). If the sensor is at rest and the force of gravity is acting along the indicated directions, the output of the corresponding channel will be negative (static acceleration).

Example: If the sensor is at rest or at uniform motion in a gravity field according to the figure given below, the output signals are:

- $\pm 0g$ for the X channel
- $\pm 0g$ for the Y channel
- $+ 1g$ for the Z channel

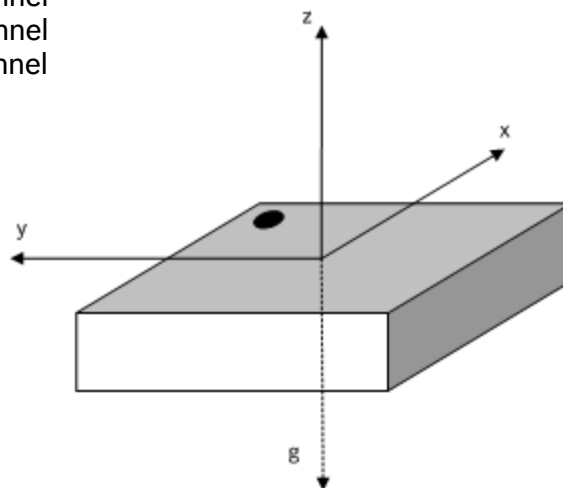
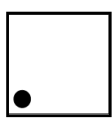
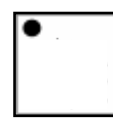
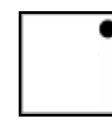
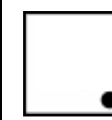


Figure 28: Orientation of sensing axis

The following table lists all corresponding output signals on X, Y, and Z while the sensor is at rest or at uniform motion in a gravity field under assumption of a $\pm 2g$ range setting and a top down gravity vector as shown above.

Table 25: Output signals depending on sensor orientation

| Sensor Orientation (gravity vector ↓) |  |  |  |  | upright | inverted |
|--|---|---|---|---|------------|----------------|
| Output Signal X | 0g / 0LSB | 1g/4096LSB | 0g / 0LSB | -1g / -4096LSB | 0g / 0LSB | 0g / 0LSB |
| Output Signal Y | -1g / -4096LSB | 0g / 0LSB | 1g / 4096LSB | 0g / 0LSB | 0g / 0LSB | 0g / 0LSB |
| Output Signal Z | 0g / 0LSB | 0g / 0LSB | 0g / 0LSB | 0g / 0LSB | 1g/4096LSB | -1g / -4096LSB |

9.3 Landing Pattern Recommendation

For the design of the landing patterns, we recommend the following dimensioning:

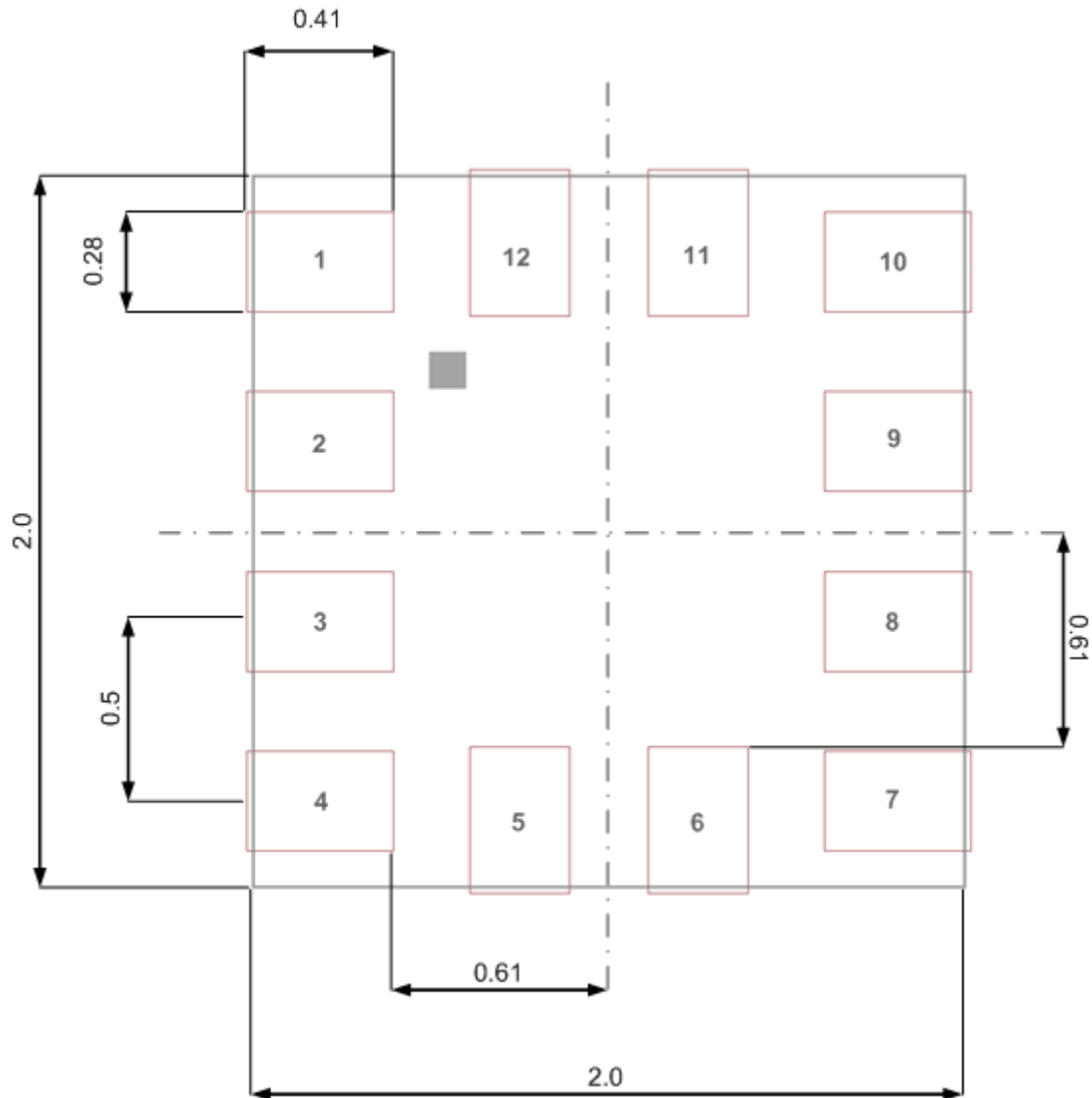


Figure 29: Landing patterns; dimensions are in mm

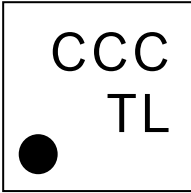
Same tolerances as given for the outline dimensions (Chapter 9.1, Figure 27) should be assumed.

A wiring no-go area in the top layer of the PCB below the sensor is strongly recommended (e.g. no vias, wires or other metal structures).

9.4 Marking

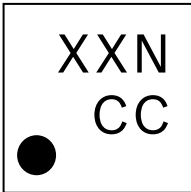
9.4.1 Mass production samples

Table 26: Marking of mass production samples

| Labeling | Name | Symbol | Remark |
|---|------------------|--------|--|
|  | Lot counter | CCC | 3 alphanumeric digits, variable to generate mass production trace-code |
| | Product number | T | 1 alphanumeric digit, fixed to identify product type, T = "D" |
| | Sub-con ID | L | 1 alphanumeric digit, variable to identify sub-con |
| | Pin 1 identifier | ● | -- |

9.4.2 Engineering samples

Table 27: Marking of engineering samples

| Labeling | Name | Symbol | Remark |
|---|------------------|--------|---|
|  | Eng. sample ID | N | 1 alphanumeric digit, fixed to identify engineering sample, N = "*" or "e" or "E" |
| | Sample ID | XX | 2 alphanumeric digits, variable to generate trace-code. |
| | Counter ID | CC | 2 alphanumeric digits, variable to generate trace-code |
| | Pin 1 identifier | ● | -- |

9.5 Soldering guidelines

The moisture sensitivity level of the BMA280 sensors corresponds to JEDEC Level 1, see also

- IPC/JEDEC J-STD-020C "Joint Industry Standard: Moisture/Reflow Sensitivity Classification for non-hermetic Solid State Surface Mount Devices"
- IPC/JEDEC J-STD-033A "Joint Industry Standard: Handling, Packing, Shipping and Use of Moisture/Reflow Sensitive Surface Mount Devices"

The sensor fulfils the lead-free soldering requirements of the above-mentioned IPC/JEDEC standard, i.e. reflow soldering with a peak temperature up to 260°C.

| Profile Feature | Pb-Free Assembly |
|---|------------------|
| Average Ramp-Up Rate (T _{Smax} to T _p) | 3° C/second max. |
| Preheat | |
| - Temperature Min (T _{Smin}) | 150 °C |
| - Temperature Max (T _{Smax}) | 200 °C |
| - Time (t _s to t _{Smax}) | 60-180 seconds |
| Time maintained above: | |
| - Temperature (T _L) | 217 °C |
| - Time (t _L) | 60-150 seconds |
| Peak/Classification Temperature (T _p) | 260 °C |
| Time within 5 °C of actual Peak Temperature (t _p) | 20-40 seconds |
| Ramp-Down Rate | 6 °C/second max. |
| Time 25 °C to Peak Temperature | 8 minutes max. |

Note 1: All temperatures refer to topside of the package, measured on the package body surface.

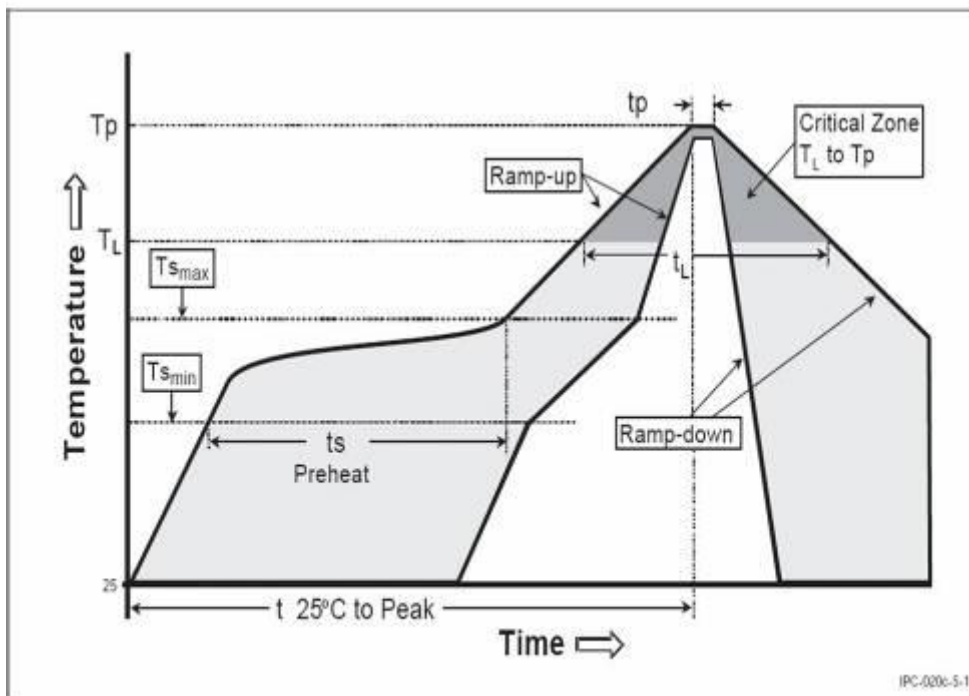


Figure 30: Soldering profile

9.6 Handling instructions

Micromechanical sensors are designed to sense acceleration with high accuracy even at low amplitudes and contain highly sensitive structures inside the sensor element. The MEMS sensor can tolerate mechanical shocks up to several thousand g's. However, these limits might be exceeded in conditions with extreme shock loads such as e.g. hammer blow on or next to the sensor, dropping of the sensor onto hard surfaces etc.

We recommend to avoid g-forces beyond the specified limits during transport, handling and mounting of the sensors in a defined and qualified installation process.

This device has built-in protections against high electrostatic discharges or electric fields (e.g. 2kV HBM); however, anti-static precautions should be taken as for any other CMOS component. Unless otherwise specified, proper operation can only occur when all terminal voltages are kept within the supply voltage range. Unused inputs must always be tied to a defined logic voltage level.



9.7 Tape and reel specification

The BMA280 is shipped in a standard cardboard box.
The box dimension for 1 reel is: L x W x H = 35cm x 35cm x 6cm.
BMA280 quantity: 10,000pcs per reel, please handle with care.

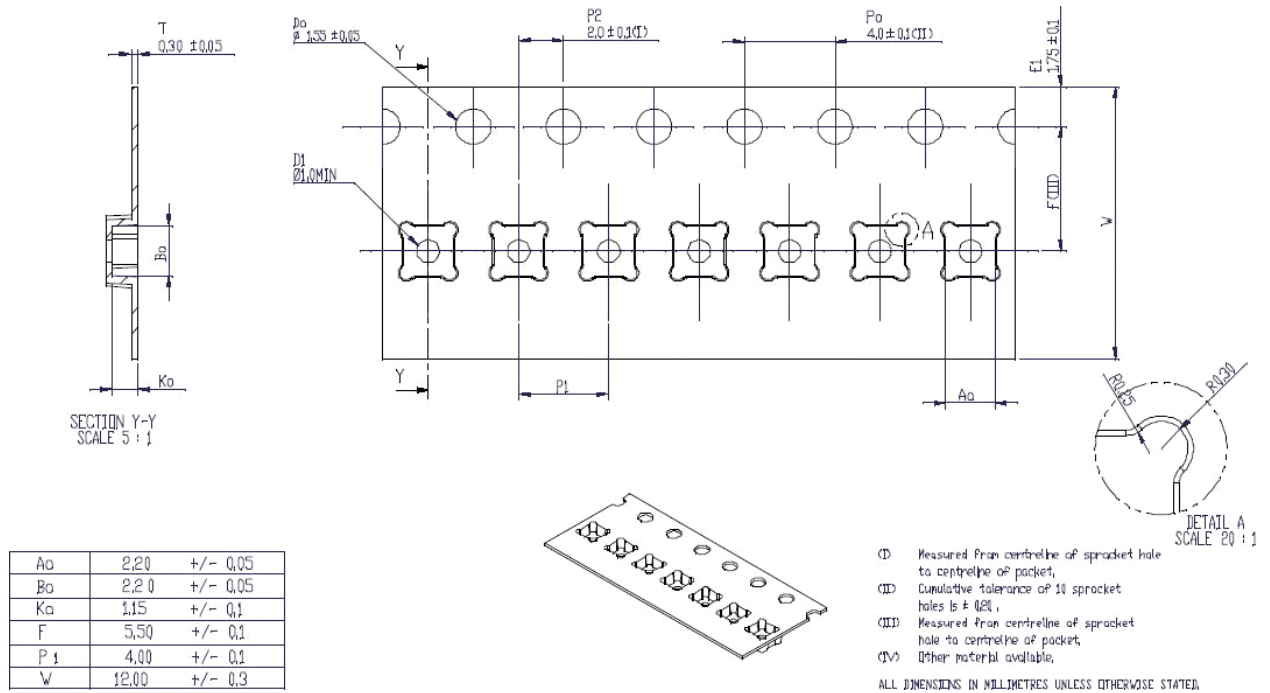


Figure 31: Tape and reel dimensions in mm

9.7.1 Orientation within the reel

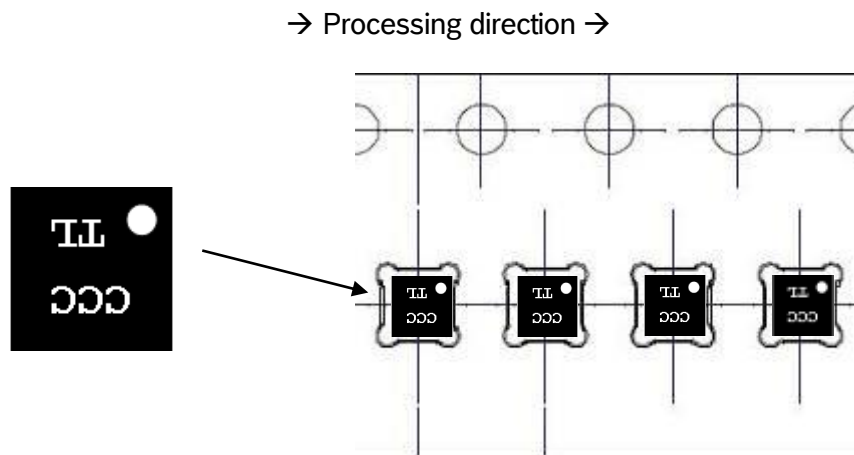


Figure 32: Orientation of the BMA280 devices relative to the tape



9.8 Environmental safety

The BMA280 sensor meets the requirements of the EC restriction of hazardous substances (RoHS) directive, see also:

RoHS–Directive 2011/65/EU and its amendments, including the amendment 2015/863/EU on the restriction of the use of certain hazardous substances in electrical and electronic equipment.

9.8.1 Halogen content

The BMA280 is halogen-free. For more details on the analysis results please contact your Bosch Sensortec representative.

9.8.2 Internal package structure

Within the scope of Bosch Sensortec's ambition to improve its products and secure the mass product supply, Bosch Sensortec qualifies additional sources (e.g. 2nd source) for the LGA package of the BMA280.

While Bosch Sensortec took care that all of the technical packages parameters are described above are 100% identical for all sources, there can be differences in the chemical content and the internal structural between the different package sources.

However, as secured by the extensive product qualification process of Bosch Sensortec, this has no impact to the usage or to the quality of the BMA280 product.

10. Legal disclaimer

10.1 Engineering samples

Engineering Samples are marked with an asterisk (*), (E) or (e). Samples may vary from the valid technical specifications of the product series contained in this data sheet. They are therefore not intended or fit for resale to third parties or for use in end products. Their sole purpose is internal client testing. The testing of an engineering sample may in no way replace the testing of a product series. Bosch Sensortec assumes no liability for the use of engineering samples. The Purchaser shall indemnify Bosch Sensortec from all claims arising from the use of engineering samples.

10.2 Product use

Bosch Sensortec products are developed for the consumer goods industry. They may only be used within the parameters of this product data sheet. They are not fit for use in life-sustaining or safety-critical systems. Safety-critical systems are those for which a malfunction is expected to lead to bodily harm, death or severe property damage. In addition, they shall not be used directly or indirectly for military purposes (including but not limited to nuclear, chemical or biological proliferation of weapons or development of missile technology), nuclear power, deep sea or space applications (including but not limited to satellite technology).

Bosch Sensortec products are released on the basis on the legal and normative requirements relevant to the Bosch Sensortec product for use in the following geographical target market: BE, BG, DK, DE, EE, FI, FR, GR, IE, IT, HR, LV, LT, LU, MT, NL, AT, PL, PT, RO, SE, SK, SI, ES, CZ, HU, CY, US, CN, JP, KR, TW. If you need further information or have further requirements, please contact your local sales contact.

The resale and/or use of Bosch Sensortec products are at the purchaser's own risk and his own responsibility. The examination of fitness for the intended use is the sole responsibility of the purchaser.

The purchaser shall indemnify Bosch Sensortec from all third party claims arising from any product use not covered by the parameters of this product data sheet or not approved by Bosch Sensortec and reimburse Bosch Sensortec for all costs in connection with such claims.

The purchaser accepts the responsibility to monitor the market for the purchased products, particularly with regard to product safety, and to inform Bosch Sensortec without delay of all safety-critical incidents.

10.3 Application examples and hints

With respect to any examples or hints given herein, any typical values stated herein and/or any information regarding the application of the device, Bosch Sensortec hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights or copyrights of any third party. The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics. They are provided for illustrative purposes only and no evaluation regarding infringement of intellectual property rights or copyrights or regarding functionality, performance or error has been made.

11. Document history and modification

| Rev. No | Chapter | Description of modification/changes | Date |
|---------|--|---|----------------|
| 0,1 | All | Initial, advance prelim. data sheet | Jan 2011 |
| 0.2 | All | Internal update, not for release | - |
| 0.3 | All | Complete review | 23 Sept 2011 |
| 0.4 | 1 | Update table 1 | 24 Nov 2011 |
| | 4.2 | Update | |
| | 4.8, 6.2 | New chapter on softreset | |
| | 5.2, 5.4 | Update | |
| 1.0 | 4.8, 6.2 | Update | 09 Jan 2012 |
| | 4.2, 4.4, 4.7.4, 4.7.6, 6.2 | Update | |
| | 1 | Update | |
| 1.1 | 4.1, 4.7.8, 6.2, 9 | Update | 14 Mar 2012 |
| 1.2 | 2, 4.1, 6.2, 9.1 | Update | 27 Apr 2012 |
| 1.3 | 1, 2, 4, 5.4, 6.2, 7, 8, 9 | Update | 01 August 2012 |
| 1.4 | 1, 4.1, 4.2, 6.2, 7.1, 9.3 | Internal Update | 24 Sep 2012 |
| 1.5 | 4.5.2, 4.7.8, 6.2, 9.1, 9.3 | Update | 22 Oct 2012 |
| 1.6 | 1, 4.2, 4.7.3, 4.7.7, 6.2, 9.3, 9.4.1, 9.8 | Update | 21 May 2013 |
| 1.7 | 4.7 | Update Single tap; update high-g interrupt | 12 Dec 13 |
| | 6.2 | Update 0x0F; update 0x2B; update 0x32 | |
| 1.8 | 4.2 | sleeptimer_en->sleeptimer_mode, linguistic improvement: "... a wake-up time of at least ... | 01 August 2014 |
| | 4.5.1 | Slow compensation update (High-pass filter cut off frequency) | |
| | 4.7.6 | Tap sensing update (temporary latched interrupt) | |
| | 6.2 | 0x37 cut_of update | |
| | 7.2 | I ² C description update | |
| 1.9 | 10. | Disclaimer update | August 2019 |
| 1.10 | 10. | Disclaimer update | November 2020 |
| 1.11 | 9.8 | ROHS directive update | October 2021 |

Bosch Sensortec GmbH
 Gerhard-Kindler-Strasse 9
 72770 Reutlingen / Germany

contact@bosch-sensortec.com
 www.bosch-sensortec.com

Modifications reserved | Printed in Germany
 Specifications subject to change without notice
 Document number: BST-BMA280-DS000-14
 Revision_1.11_October_2021

Bosch Sensortec