

BMA456

Digital, triaxial acceleration sensor

Bosch Sensortec



BOSCH

Invented for life



Data Sheet BMA456

Part number(s) 0 273 141 282

Document revision 1.2

Release date April 2019

Document number BST-BMA456-DS000-01

Notes Subject to change without notice.
Product photos and pictures are for illustration purposes only and may differ from the real product's appearance.

Confidential and under NDA



BMA456

16 bit, digital, triaxial acceleration sensor with intelligent on-chip motion-triggered interrupt features optimized for wearable applications.

Key features

- Small package size LGA package (12 pins), footprint 2mm x 2mm, height 0.65 mm
- 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 684Hz - <8Hz
up to a max. output data read out of 1.6 kHz
Integrated FIFO on sensor with 1 kb
Step Counter optimized for wearable devices
Activity Recognition: Running, Walking, Still
Tilt-On-Wrist detection
Tap/Double tap interrupt
Any/No-Motion interrupt
- Ultra-low power Low current consumption of data acquisition and all integrated features
- (Secondary) Auxiliary Interface Hub for ext. Magnetometer and data synchronization
- RoHS compliant, halogen-free

Typical applications

- Applications with height constraints
- Plug 'n' Play Step-Counter solution with watermark functionality
- Fitness applications / Activity Tracking
- Power management for wearable applications
- Display on/off and profile switching
- User interface without hardware buttons
- E-compass tilt compensation and data synchronization
- High performance angle measurements



Table of contents

| | |
|--|-----------|
| 1. SPECIFICATION..... | 7 |
| 2. ABSOLUTE MAXIMUM RATINGS..... | 10 |
| 3. QUICK START GUIDE | 11 |
| Note about using the BMA456: | 11 |
| First application setup examples algorithms:..... | 11 |
| 4. FUNCTIONAL DESCRIPTION..... | 15 |
| 4.1. SUPPLY VOLTAGE AND POWER MANAGEMENT | 16 |
| 4.2. DEVICE INITIALIZATION | 17 |
| 4.3. POWER MODES | 18 |
| 4.4. SENSOR DATA..... | 20 |
| Acceleration Data | 20 |
| Filter Settings..... | 20 |
| Accelerometer data processing for low power mode | 21 |
| Data Ready Interrupt..... | 21 |
| Temperature Sensor..... | 21 |
| Sensor Time | 22 |
| Configuration Changes..... | 22 |
| 4.5. FIFO..... | 24 |
| Frames | 24 |
| Conditions and Details..... | 27 |
| FIFO data synchronization..... | 29 |
| FIFO synchronization with external interrupts..... | 30 |
| FIFO Interrupts..... | 30 |
| FIFO Reset..... | 30 |
| 4.6. INTERRUPT FEATURES | 31 |
| Global Configuration..... | 31 |
| Step Detector / Step Counter..... | 33 |
| Walking activity recognition..... | 36 |
| Tilt on Wrist..... | 37 |
| Double tap / Tap detection | 39 |
| Any Motion / No motion detection | 40 |
| 4.7. GENERAL INTERRUPT PIN CONFIGURATION | 42 |
| Electrical Interrupt Pin Behavior | 42 |
| Interrupt Pin Mapping..... | 42 |
| 4.8. AUXILIARY SENSOR INTERFACE..... | 43 |
| Structure and Concept..... | 43 |
| Interface Configuration | 43 |
| Setup mode (AUX_IF_CONF.aux_manual_en =0b1) | 44 |
| Data mode (AUX_IF_CONF.aux_manual_en=0)..... | 47 |
| Delay (Time Offset) | 47 |



| | | |
|-----------|---------------------------------------|-----------|
| 4.9. | SENSOR SELF-TEST | 48 |
| 4.10. | OFFSET COMPENSATION | 49 |
| | Manual Offset Compensation | 49 |
| | Inline Calibration..... | 49 |
| 4.11. | NON-VOLATILE MEMORY | 50 |
| 4.12. | SOFT-RESET | 50 |
| 5. | REGISTER DESCRIPTION..... | 51 |
| 5.1. | GENERAL REMARKS | 51 |
| 5.2. | REGISTER MAP | 51 |
| | Register (0x00) CHIP_ID..... | 61 |
| | Register (0x02) ERR_REG..... | 61 |
| | Register (0x03) STATUS..... | 62 |
| | Register (0x0A) DATA_0..... | 62 |
| | Register (0x0B) DATA_1..... | 63 |
| | Register (0x0C) DATA_2..... | 63 |
| | Register (0x0D) DATA_3..... | 64 |
| | Register (0x0E) DATA_4..... | 64 |
| | Register (0x0F) DATA_5..... | 65 |
| | Register (0x10) DATA_6 | 65 |
| | Register (0x11) DATA_7..... | 66 |
| | Register (0x12) DATA_8 | 66 |
| | Register (0x13) DATA_9 | 67 |
| | Register (0x14) DATA_10..... | 67 |
| | Register (0x15) DATA_11 | 68 |
| | Register (0x16) DATA_12 | 68 |
| | Register (0x17) DATA_13..... | 69 |
| | Register (0x18) SENSORTIME_0..... | 69 |
| | Register (0x19) SENSORTIME_1..... | 70 |
| | Register (0x1A) SENSORTIME_2 | 70 |
| | Register (0x1B) EVENT | 71 |
| | Register (0x1C) INT_STATUS_0..... | 71 |
| | Register (0x1D) INT_STATUS_1..... | 72 |
| | Register (0x1E) STEP_COUNTER_0 | 72 |
| | Register (0x1F) STEP_COUNTER_1..... | 73 |
| | Register (0x20) STEP_COUNTER_2..... | 73 |
| | Register (0x21) STEP_COUNTER_3..... | 74 |
| | Register (0x22) TEMPERATURE | 74 |
| | Register (0x24) FIFO_LENGTH_0 | 75 |
| | Register (0x25) FIFO_LENGTH_1 | 75 |
| | Register (0x26) FIFO_DATA..... | 76 |
| | Register (0x27) ACTIVITY_TYPE | 76 |
| | Register (0x2A) INTERNAL_STATUS | 77 |
| | Register (0x40) ACC_CONF | 78 |



| | |
|--|------------|
| Register (0x41) ACC_RANGE..... | 79 |
| Register (0x44) AUX_CONF | 80 |
| Register (0x45) FIFO_DOWNS..... | 80 |
| Register (0x46) FIFO_WTM_0..... | 81 |
| Register (0x47) FIFO_WTM_1..... | 82 |
| Register (0x48) FIFO_CONFIG_0..... | 82 |
| Register (0x49) FIFO_CONFIG_1..... | 83 |
| Register (0x4B) AUX_DEV_ID..... | 84 |
| Register (0x4C) AUX_IF_CONF..... | 84 |
| Register (0x4D) AUX_RD_ADDR..... | 85 |
| Register (0x4E) AUX_WR_ADDR..... | 85 |
| Register (0x4F) AUX_WR_DATA..... | 86 |
| Register (0x53) INT1_IO_CTRL..... | 86 |
| Register (0x54) INT2_IO_CTRL..... | 87 |
| Register (0x55) INT_LATCH | 88 |
| Register (0x56) INT1_MAP..... | 89 |
| Register (0x57) INT2_MAP..... | 89 |
| Register (0x58) INT_MAP_DATA..... | 90 |
| Register (0x59) INIT_CTRL..... | 90 |
| Register (0x5E) FEATURES_IN..... | 91 |
| Register (0x5F) INTERNAL_ERROR..... | 95 |
| Register (0x6A) NVM_CONF | 95 |
| Register (0x6B) IF_CONF | 96 |
| Register (0x6D) ACC_SELF_TEST..... | 96 |
| Register (0x70) NV_CONF..... | 97 |
| Register (0x71) OFFSET_0..... | 98 |
| Register (0x72) OFFSET_1..... | 98 |
| Register (0x73) OFFSET_2..... | 99 |
| Register (0x7C) PWR_CONF..... | 99 |
| Register (0x7D) PWR_CTRL..... | 100 |
| Register (0x7E) CMD..... | 100 |
| 6. DIGITAL INTERFACES..... | 102 |
| 6.1. INTERFACES..... | 102 |
| 6.2. PRIMARY INTERFACE..... | 103 |
| 6.3. PRIMARY INTERFACE I2C/SPI PROTOCOL SELECTION | 104 |
| 6.4. SPI INTERFACE AND PROTOCOL | 104 |
| 6.5. PRIMARY I2C INTERFACE | 109 |
| 6.6. SPI AND I ² C ACCESS RESTRICTIONS..... | 113 |
| 6.7. AUXILIARY INTERFACE | 113 |
| 7. PIN-OUT AND CONNECTION DIAGRAMS..... | 114 |
| 7.1. PIN-OUT | 114 |
| 7.2. CONNECTION DIAGRAMS WITHOUT AUXILIARY INTERFACE | 115 |



| | |
|---|------------|
| SPI | 115 |
| I2C..... | 116 |
| 7.3. CONNECTION DIAGRAMS WITH AUXILIARY INTERFACE | 116 |
| SPI | 116 |
| I2C..... | 117 |
| 8. PACKAGE | 118 |
| 8.1. PACKAGE OUTLINE DIMENSIONS | 118 |
| 8.2. SENSING AXIS ORIENTATION | 119 |
| 8.3. LANDING PATTERN RECOMMENDATION | 121 |
| 8.4. MARKING | 122 |
| Mass production..... | 122 |
| Engineering samples..... | 122 |
| 8.5. SOLDERING GUIDELINES | 123 |
| 8.6. HANDLING INSTRUCTIONS | 124 |
| 8.7. TAPE AND REEL SPECIFICATION | 125 |
| 8.8. ENVIRONMENTAL SAFETY | 126 |
| Halogen content..... | 126 |
| Internal package structure..... | 126 |
| 9. LEGAL DISCLAIMER..... | 127 |
| 9.1. ENGINEERING SAMPLES..... | 127 |
| 9.2. PRODUCT USE..... | 127 |
| 9.3. APPLICATION EXAMPLES AND HINTS..... | 127 |
| 10. DOCUMENT HISTORY AND MODIFICATION..... | 128 |



1. Specification

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

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 | 1.8 | 3.6 | V |
| Supply Voltage I/O Domain | V_{DDIO} | | 1.2 | 1.8 | 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} | $V_{DDIO} \geq 1.62V$, $I_{OL} \leq 2mA$, SPI | | | $0.2V_{DDIO}$ | - |
| | | $V_{DDIO} < 1.62V$, $I_{OL} \leq 1.5mA$, SPI | | | $0.2V_{DDIO}$ | - |
| Voltage Output High Level | V_{OH} | $V_{DDIO} \geq 1.62V$, $I_{OH} \leq 2mA$, SPI | $0.8V_{DDIO}$ | | | - |
| | | $V_{DDIO} \leq 1.62V$, $I_{OH} \leq 1.5mA$, SPI | $0.8V_{DDIO}$ | | | - |
| Total Supply Current in Performance mode | I_{DD} | Nominal V_{DD} and V_{DDIO} , 25°C, g_{FS4g} | | 150 | | μA |
| Total Supply Current in Suspend Mode | I_{DDsum} | Nominal V_{DD} and V_{DDIO} , 25°C | | 3.5 | | μA |
| Total Supply Current in Low-power Mode | I_{DDlp1} | Nominal V_{DD} and V_{DDIO} , 25°C 50 Hz ODR | | 14 | | μA |
| Power-Up Time | ts_{up} | | | | 1 | ms |
| Non-volatile memory (NVM) write-cycles | n_{NVM} | | | | 15 | cycles |

| Operating Temperature | T_A | | -40 | | +85 | °C |
|---|--------------|--|------|----------------------|------|-------------------|
| OUTPUT SIGNAL | | | | | | |
| Parameter | Symbol | Condition | Min | Typ | Max | Units |
| Sensitivity | S_{2g} | $g_{FS2g}, T_A=25^\circ C$ | | 16384 | | LSB/g |
| | S_{4g} | $g_{FS4g}, T_A=25^\circ C$ | | 8192 | | LSB/g |
| | S_{8g} | $g_{FS8g}, T_A=25^\circ C$ | | 4096 | | LSB/g |
| | S_{16g} | $g_{FS16g}, T_A=25^\circ C$ | | 2048 | | LSB/g |
| Sensitivity Temperature Drift | TCS | | | 0.005 | | %/K |
| Zero-g Offset | Off | Nominal V_{DD} and V_{DDIO} , $25^\circ C$, g_{FS4g} | | 20 | | mg |
| Zero-g Offset Temperature Drift | TCO | X/Y - Axes | | 0.2 | | mg/K |
| | | Z - Axis | | 0.35 | | mg/K |
| Output Data Rate | ODR_{PERF} | Performance mode | 12.5 | | 1600 | Hz |
| Output data rate and BW in Performance mode | $ODR_{12.5}$ | 3dB cutoff frequency of the accelerometer according to ODR with normal filter mode | | 5.06 | | Hz |
| | ODR_{25} | | | 10.12 | | Hz |
| | ODR_{50} | | | 20.25 | | Hz |
| | ODR_{100} | | | 40.5 | | Hz |
| | ODR_{200} | | | 80 | | Hz |
| | ODR_{400} | | | 162 (155 for Z axis) | | Hz |
| | ODR_{800} | | | 324 (262 for Z axis) | | Hz |
| | ODR_{1600} | | | 684 (353 for Z axis) | | Hz |
| Output Data Rate | ODR_{LPM} | Low-power mode | 0.78 | | 400 | Hz |
| Nonlinearity | NL | Nominal V_{DD} and V_{DDIO} , $25^\circ C$, g_{FS4g} | | 0.5 | | %FS |
| Output Noise Density | n_{dens} | Nominal V_{DD} and V_{DDIO} , $25^\circ C$, g_{FS4g} | | 120 | | $\mu g/\sqrt{Hz}$ |
| Power Supply Rejection Ratio | PSRR | | | 1 | | mg/50mV |

**MECHANICAL CHARACTERISTICS**

| Parameter | Symbol | Condition | Min | Typ | Max | Units |
|------------------------|----------------|---|------------|------------|------------|--------------|
| Cross Axis Sensitivity | S | relative contribution between any two of the three axes | | 0.5 | | % |
| Alignment Error | E _A | relative to package outline | | 0.5 | | ° |



2. Absolute maximum ratings

Absolute maximum ratings

| Parameter | Condition | Min | Max | Units |
|---|------------------------------|------|----------------------------|-------|
| Voltage at Supply Pin | V _{DD} Pin | -0.3 | 4 | V |
| | V _{DDIO} Pin | -0.3 | 4 | V |
| Voltage at any Logic Pin | Non-Supply Pin | -0.3 | V _{DDIO} +0.3, <4 | 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, at any pin | HBM | | 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. Quick Start Guide

The purpose of this chapter is to help developers who want to start working with the BMA456 by giving you some very basic hands-on application examples to get started.

Note about using the BMA456:

The communication between application processor and BMA456 will happen either over i2c or spi interface. For more information about the interfaces, read the related chapter 6.

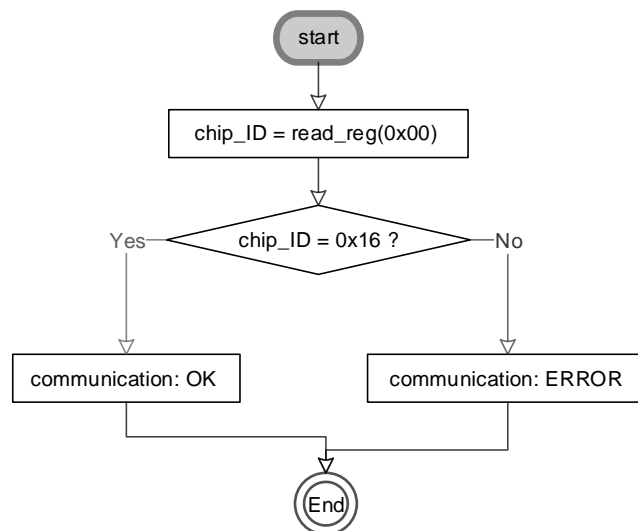
- Before starting the test, the device has to be properly connected to the master (AP) and powered up. For more information about it, read the related chapter 7. Pin-out and Connection Diagrams.

First application setup examples algorithms:

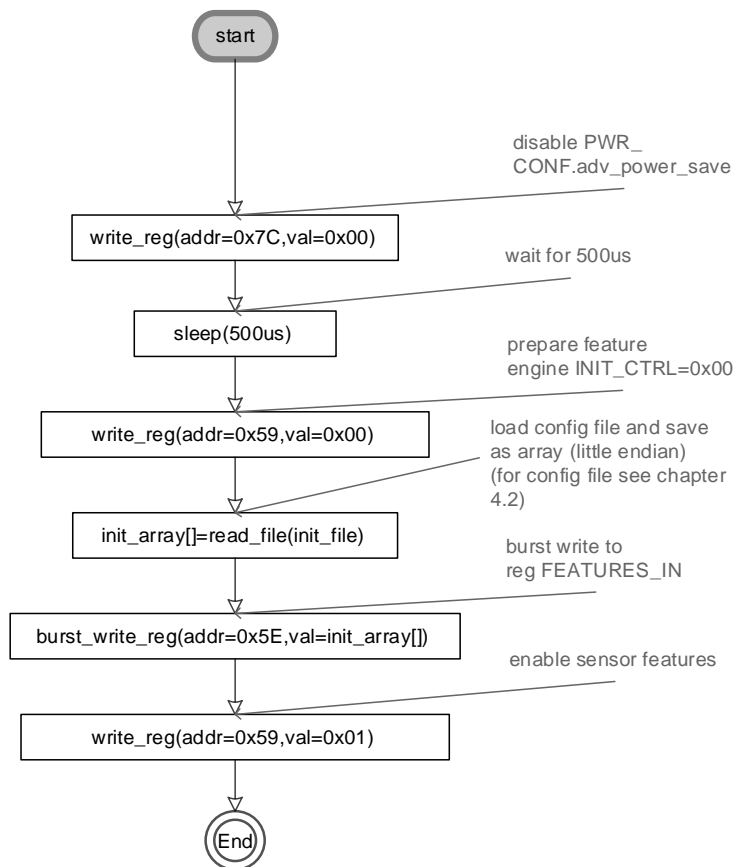
After correct power up by setting the correct voltage to the appropriate external pins, the BMA456 enters automatically into the Power On Reset (POR) sequence. In order to properly make use of the BMA456, certain steps from host processor side are needed. The most typical operations will be explained in the following application examples in form of flow-diagrams.

Example 1: Testing communication with the BMA456 and initializing feature engine

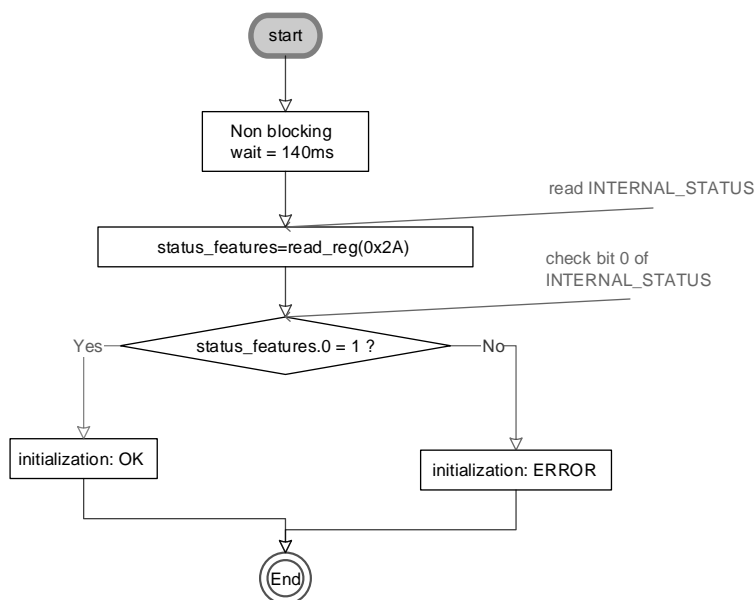
- a. -reading chip id (checking correct communication)



- b. -performing initialization sequence (interrupt feature engine)

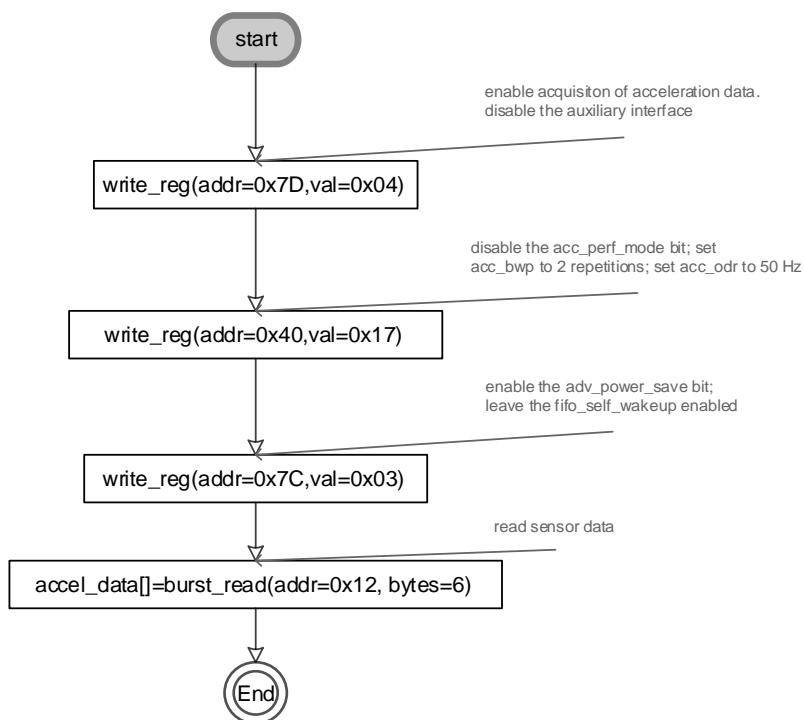


c. -checking the correct status of the interrupt feature engine



**Example 2: Reading acceleration data from BMA456 (example: low power mode)**

-setting data processing parameters (power, bandwidth, range) and reading sensor data



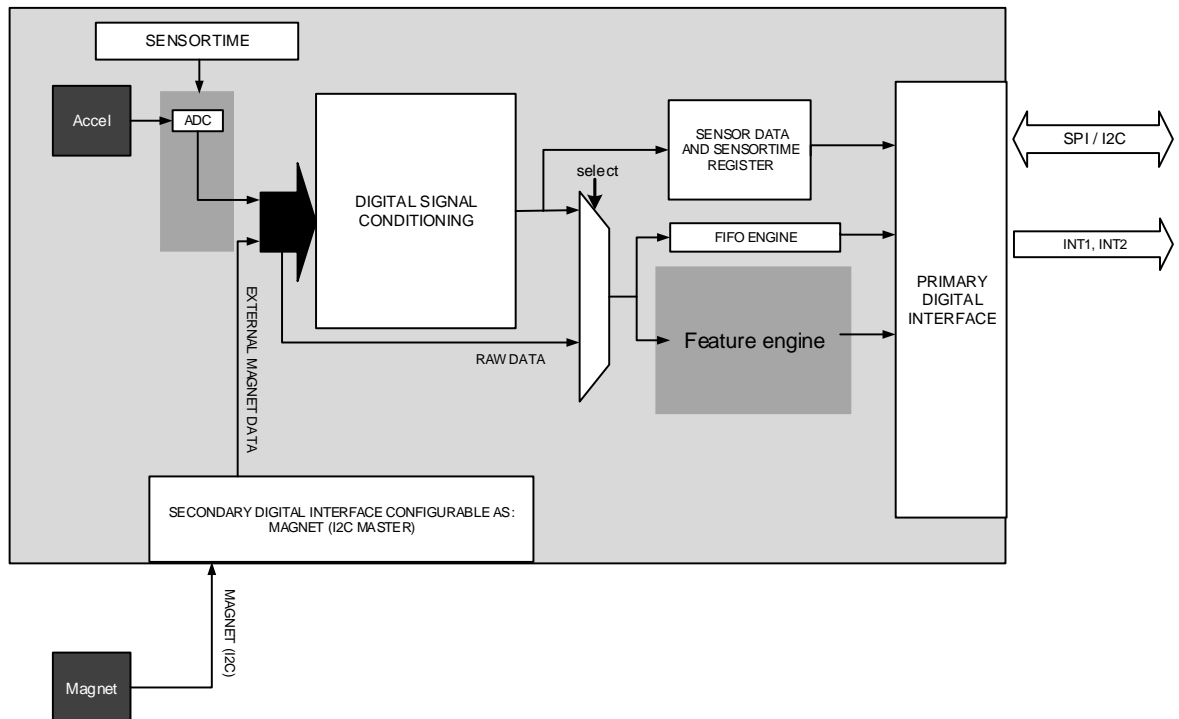
Further steps:

The BMA456 has many more capabilities that are described in this document and include FIFO, power saving modes, synchronization capabilities with host processor, data synchronization and integration with third party sensors, many interrupts generation and more features like step counter, etc.



4. Functional Description

Block Diagram





4.1. Supply Voltage and Power Management

BMA456 has two distinct power supply pins:

- VDD is the main power supply.
- VDDIO is a separate power supply pin used for supplying power for the interface including the auxiliary interface.

There are no limitations with respect to the voltage level applied to the VDD and VDDIO pins, as long as it lies within the respective operating range. Furthermore, the device can be completely switched off ($VDD = 0V$) while keeping the VDDIO supply within operating range or vice versa. However if the VDDIO supply is switched off, all interface pins (CSB, SDX, SCX) must be kept close to GNDIO potential. The device is reset when the supply voltage applied to at least one supply pin VDD or VDDIO falls below the specified minimum values. No constraints exist for the minimum slew-rate of the voltage applied to the VDD and VDDIO pins.

4.2. Device Initialization

After power up sequence the accelerometer is in suspend mode, device must be initialized through the following procedure. Initialization has to be performed as well after every POR or soft reset.

- Disable advanced power save mode: [PWR_CONF.adv_power_save](#) = 0b0
- Wait for 450 µs. The register [SENSORTIME_0](#) increments every 39.25 µsec and may be used for accurate timing.
- Write [INIT_CTRLinit_ctrl](#) = 0x00
- Load configuration file
 - Burst write initialization data to Register [FEATURES_IN](#). The configuration file is included in the driver available on the Bosch Sensortec website (www.bosch-sensortec.com) or from your regional support team. Optionally the configuration file can be written to the Register [FEATURES_IN](#) in several consecutive burst write access. Every burst write must contain an even number of bytes.
 - Optionally:
Burst read configuration file from Register [FEATURES_IN](#) and check correctness
- Enable sensor features – write 0x01 into register [INIT_CTRLinit_ctrl](#). This operation must not be performed more than once after POR or soft reset.
- Wait until Register [INTERNAL_STATUS.message](#) contains the value 0b1. This will happen after at most 140-150 msec.

After initialization sequence has been completed, the device is in configuration mode (power mode). Now it is possible to switch to the required power mode and all features are ready to use as described in chapter 4.

4.3. Power Modes

The power state of the BMA456 is controlled through the registers [PWR_CONF](#) and [PWR_CTRL](#). The Register [PWR_CTRL](#) enables and disables the accelerometer and the auxiliary sensor. The Register [PWR_CONF](#) controls which power state the sensors enter if they are enabled or disabled in the Register [PWR_CTRL](#). The power state impacts the behavior of the sensor with respect to start-up time, available functions, etc. but not the sensor data quality. The sensor data quality is controlled in the Registers [ACC_CONF](#).

In all global power configurations both register contents and FIFO contents are retained.

Low Power Mode: This power configuration aggressively reduces power of the device as much as possible. The low power mode configuration is activated through enabling [PWR_CONF.adv power save](#)=0b1 and disabling [ACC_CONF.acc perf mode](#)=0b0. In this configuration these externally user visible features may not be available:

- Register writes need an inter-write-delay of at least **1000 us**.
- The sensors log data into the FIFO in performance and low power mode. When the FIFO watermark interrupt is active, the FIFO is accessible for reading in low power mode until a burst read operation on Register [FIFO_DATA](#) completes when [PWR_CONF.fifo self wakeup](#)=0b1. When [PWR_CONF.fifo self wakeup](#)=0b0, the user needs to disable advanced power save mode ([PWR_CONF.adv power save](#)=0b0) before reading the FIFO and wait for 250 µs.
- To read out FIFO data w/o a FIFO watermark interrupt, the advanced power save configuration needs to be disabled ([PWR_CONF.adv power save](#)=0b0)

The table below shows a few examples with the optimal power configurations

| Usecase | ACC_CONF.acc perf mod | PWR_CONF.adv power sa | PWR_CTRL.acc en | Power consumption |
|-----------------------------|---------------------------------------|---------------------------------------|---------------------------------|-------------------------------------|
| Configuration mode | x | 0 | x | tbd |
| Suspend (lowest power mode) | x | 1 | 0 | suspend power |
| Performance mode accel | 1 | x | 1 | accel power |
| Low power mode | 0 | 1 | 1 | Depends on ACC_CONF |

The [PWR_CTRL](#) register is used to enable and disable sensors. Per default, all sensors are disabled. Acceleration sensor must be enabled by setting [PWR_CTRL.acc en](#)=0b1.



The auxiliary sensor functionality is supported only when the auxiliary interface is connected for the auxiliary sensor operation. If the auxiliary interface is not used for auxiliary sensor operation, then the auxiliary sensor interface must remain disabled by setting [PWR_CTRLaux_en](#)=0b0 (default).

To change the power mode of the auxiliary sensor, both the power mode of the auxiliary interface and the auxiliary sensor part needs to be changed, e.g. to set the auxiliary sensor to suspend mode:

- Set the auxiliary sensor interface to suspend in Register [PWR_CTRLaux_en](#)=0b0. Changing the auxiliary sensor interface power mode to suspend does not imply any mode change in the auxiliary sensor.
- The auxiliary sensor part itself must be put into suspend mode by writing the respective configuration bits of the auxiliary sensor part. The power mode of the auxiliary sensor part is controlled by setting the BMA456 auxiliary sensor interface into manual mode by [AUX_IF_CONFaux_manual_en](#)=0b1 and then communicating with the auxiliary sensor part through the BMA456 registers [AUX_RD_ADDR](#), [AUX_WR_ADDR](#), and [AUX_WR_DATA](#). For details see Chapter 4.8.

4.4. Sensor Data

Acceleration Data

The width of acceleration data is 16 bits given in two's complement representation in the registers [DATA 8](#) to [DATA 13](#). The 16 bits for each axis are split into an MSB upper part and an LSB lower part. 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).

Filter Settings

The accelerometer digital filter can be configured through the Register [ACC_CONF](#).

Note:

Illegal settings in configuration registers will result in an error code in Register [ERR_REG](#). The content of the data register is undefined, and if the FIFO is used, it may contain no value.

Accelerometer data processing for performance mode

Performance mode is enabled with [ACC_CONF.acc_perf_mode](#)=0b1. In this power mode, the accelerometer data is sampled at equidistant points in the time, defined by the accelerometer output data rate parameter [ACC_CONF.acc_odr](#). The output data rate can be configured in one of eight different valid ODR configurations going from 12.5 Hz up to 1600Hz.

The filter bandwidth shows a 3db cutoff frequency shown in the following table:

Table 12: 3dB cutoff frequency of the accelerometer according to ODR with normal filter mode

| Accelerometer ODR [Hz] | 12.5 | 25 | 50 | 100 | 200 | 400 | 800 | 1600 |
|---------------------------|------|-------|-------|------|-----|----------------------------|----------------------------|----------------------------|
| 3dB Cutoff frequency [Hz] | 5.06 | 10.12 | 20.25 | 40.5 | 80 | 162 (155 for Z axis) | 324 (262 for Z axis) | 684 (353 for Z axis) |

Accelerometer data processing for low power mode

Low power mode can be enabled by [PWR_CONF.adv_power_save=0b1](#) and [ACC_CONF.acc_perf_mode=0b0](#). In this power mode, the accelerometer regularly changes between a suspend power mode phase where no measurement is performed and a performance power mode phase, where data is acquired. The period of the duty cycle for changing between suspend and performance mode will be determined by the output data rate ([ACC_CONF.acc_odr](#)). The output data rate can be configured in one of 10 different valid ODR configurations going from 0.78Hz up to 400Hz. The samples acquired during the normal mode phase will be averaged and the result will be the output data. The number of averaged samples can be determined by the parameter [ACC_CONF.acc_bwp](#) through the following formula:

$$\begin{aligned}\text{averaged samples} &= 2^{(\text{Val}(\text{acc_bwp}))} \\ \text{skipped samples} &= (1600/\text{ODR}) - \text{averaged samples}\end{aligned}$$

A higher number of averaged samples will result in a lower noise level of the signal, but since the performance power mode phase is increased, the power consumption will also rise.

Data Ready Interrupt

This interrupt fires whenever a new data sample set from accelerometer, and auxiliary sensor is complete. This allows a low latency data readout. In non-latched mode, the interrupt and the flag in Register [INT_STATUS_1](#) are cleared automatically after 1/(3200Hz). If this automatic clearance is unwanted, latched-mode can be used (see chapter 4.7).

In order to enable/use the data ready interrupt map it on the desired interrupt pin via [INT_MAP_DATA](#).

Temperature Sensor

The temperature sensor has 8 bits. The temperature value is defined in Register [TEMPERATURE](#) and updated every 1.28 s.

It is always on, when a sensor is active.

| Value | Temperature |
|-------|-------------|
| 0x7F | 150 °C |
| ... | ... |
| 0x00 | 23 °C |
| ... | ... |
| 0x81 | -104 °C |
| 0x80 | Invalid |

When there is no valid temperature information available (i.e. last measurement before the time defined above), the temperature indicates an invalid value: 0x80.

Sensor Time

The BMA456 supports the concept of sensortime. Its core element is a free running counter with a width of 24 bits. It increments with a resolution of 39.0625us. The user can access the current state of the counter by reading registers [SENSORTIME_0](#) to [SENSORTIME_2](#).

All sensor events e.g. updates of data registers are synchronous to this sensor time register as defined in the table below. With every update of the data register or the FIFO, a bit m in the registers [SENSORTIME_0](#) to [SENSORTIME_2](#) toggles where m depends on the output data rate for the data register and the output data rate and the FIFO downsampling rate for the FIFO. The table below shows which bit toggles for which update rate of data register and FIFO

| Bit m in sensor_time | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|------------------------|--------|--------|-------|-------|-------|-------|------|------|
| Resolution [s] | 327.68 | 163.84 | 81.92 | 40.96 | 20.48 | 10.24 | 5.12 | 2.56 |
| Update rate [Hz] | 0.0031 | 0.0061 | 0.012 | 0.024 | 0.049 | 0.10 | 0.20 | 0.39 |

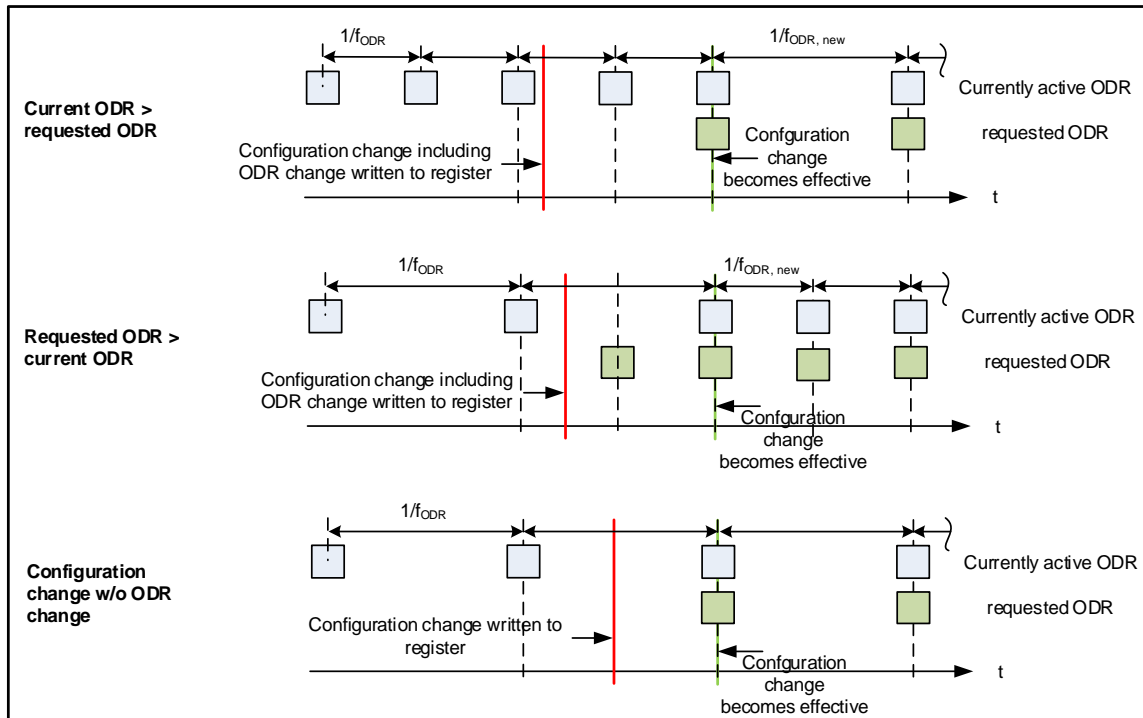
| Bit m in sensor_time | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------------------|------|------|-------|------|------|----|----|-----|-----|-----|-------|-------|--------|-------|-------|-------|
| Resolution [ms] | 1280 | 640 | 320 | 160 | 80 | 40 | 20 | 10 | 5 | 2.5 | 1.250 | 0.625 | 0.3125 | 0.156 | 0.078 | 0.039 |
| Update rate [Hz] | 0.78 | 1.56 | 3.125 | 6.25 | 12.5 | 25 | 50 | 100 | 200 | 400 | 800 | 1600 | 3200 | | | |

The sensortime is synchronized with the data capturing in the data register and the FIFO. Between the data sampling and the data capturing there is a delay which depends on the settings in the Register [ACC_CONF](#). The sensortime supports multiple seconds of sample counting and a sub-microsecond resolution, see Register [SENSORTIME_0](#) for details.

Burst reads on the registers [SENSORTIME_0](#) to [SENSORTIME_2](#) deliver always consistent values, i.e. the value of the register does not change during the burst read.

Configuration Changes

If accelerometer configuration settings in registers [ACC_CONF](#), [ACC_RANGE](#), or [AUX_CONF](#) are changed while the accelerometer ([PWR_CTRL.acc_en](#) = 0b1) or auxiliary sensor ([PWR_CTRL.aux_en](#) = 0b1) is enabled, the configuration changes are not immediately applied. The configuration changes become effective if a sampling event for the currently active ODR coincides with a sampling event for the newly requested ODR on the sensortime sampling grid. In the case where the currently active ODR equals the newly requested ODR, the configuration changes become effective at the next sampling event. See also following figure.





4.5. FIFO

The device supports the following FIFO operating modes:

- Streaming mode: overwrites oldest data on FIFO full condition
- FIFO mode: discards newest data on FIFO full condition

The FIFO depth is 1024 byte and supports the following interrupts:

- FIFO full interrupt
- FIFO watermark interrupt

FIFO is enabled with [FIFO_CONFIG_1.fifo_acc_en=0b1](#) (0b0= disabled). To enable the FIFO for the auxiliary interface (magnetometer) set [FIFO_CONFIG_1.fifo_aux_en=0b1](#) (0b0=disabled).

Frames

The FIFO captures data in frames, which consist of a header and a payload. The FIFO can be configured to skip the header (headerless mode) in which case only payload is stored.

- In header mode (standard configuration) each regular frame consists of a one byte header describing properties of the frame, (which sensors are included in this frame) and the data itself. Beside the regular frames, there are control frames.
- In headerless mode the FIFO contains sampled data only

Header mode

The header has a length of 8 bit and the following format:

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---------|--------------|---|--------------|---|---|---|-------------|---|
| Content | fh_mode<1:0> | | fh_parm<3:0> | | | | fh_ext<1:0> | |

These *fh_mode* and *fh_parm* and *fh_ext* fields are defined below

| fh_mode<1:0> | Definition | fh_parm <3:0> | fh_ext<1:0> |
|---------------|------------|-----------------|----------------------|
| 0b10 | Regular | Enabled sensors | Tag of INT2 and INT1 |
| 0b01 | Control | Control opcode | |
| 0b00 and 0b11 | Reserved | Na | |

f_parm=0b0000 is invalid for regular mode, a header of 0x80 indicates an uninitialized frame.

In a regular frame, fh_parm frame defines which sensors are included in the data part of the frame. The format is

| Name | fh_parm<3:0> | | | |
|---------|--------------|---------------|----------|---------------|
| Bit | 3 | 2 | 1 | 0 |
| Content | Reserved | FIFO_aux_data | Reserved | FIFO_acc_data |

When FIFO_<sensor x>_data is 0b1 (0b0) data for sensor x is included (not included) in the data part of the frame.

The fh_ext<1:0> field are used for external tagging.

The data format for data frames is identical to the format defined for the [Register \(0x0A\) DATA_0](#) to [Register \(0x17\) DATA_13](#) register. Only frames which contain data of at least one sensor will be written into the FIFO. E.g. fh_parm=0b0101 the data in the frame are shown below. If the read burst length is less than 8 byte, the number of auxiliary sensor data in the frame is reduced to the burst length.

| DATA[X] | Acronym | |
|---------|-------------------|--|
| X=0 | AUX_0 | copy of register Val(AUX_RD_ADDR) in auxiliary sensor register map |
| X=1 | AUX_1 | copy of register Val(AUX_RD_ADDR)+1 in auxiliary sensor register map |
| X=2 | AUX_2 | copy of register Val(AUX_RD_ADDR)+2 in auxiliary sensor register map |
| X=3 | AUX_3 | copy of register Val(AUX_RD_ADDR)+3 in auxiliary sensor register map |
| X=4 | AUX_4 | copy of register Val(AUX_RD_ADDR)+4 in auxiliary sensor register map |
| X=5 | AUX_5 | copy of register Val(AUX_RD_ADDR)+5 in auxiliary sensor register map |
| X=6 | AUX_6 | copy of register Val(AUX_RD_ADDR)+6 in auxiliary sensor register map |
| X=7 | AUX_7 | copy of register Val(AUX_RD_ADDR)+7 in auxiliary sensor register map |
| X=8 | ACC_X<7:0> (LSB) | |
| X=9 | ACC_X<15:8> (MSB) | |
| X=10 | ACC_Y<7:0> (LSB) | |
| X=11 | ACC_Y<15:8> (MSB) | |
| X=12 | ACC_Z<7:0> (LSB) | |
| X=13 | ACC_Z<15:8> (MSB) | |

Headerless mode

When the data rates of all enabled sensor elements are identical, the FIFO header may be disabled in [FIFO CONFIG 1.fifo header en](#).

The headerless mode supports only regular frames. To be able to distinguish frames from each other, all frames must have the same size. For this reason, any change in configuration that have an impact to frame size or order of data within a frame will cause an instant flush of FIFO, restarting capturing of data with the new settings.



If the auxiliary sensor interface is enabled, the number of auxiliary sensor bytes in a FIFO frame is always [AUX_IF_CONF.aux_rd_burst](#) bytes (see chapter 4.8). If the burst length is less than 8, BMA456 will pad the values read from the auxiliary sensor. E.g. if [AUX_IF_CONF.aux_rd_burst](#)=0b01 (2 Bytes), a frame with auxiliary sensor and accelerometer data will look like

| DATA[X] | Acronym | |
|---------|-------------------|--|
| X=0 | AUX_0 | copy of register Val(AUX_RD_ADDR.read_addr) in auxiliary sensor register map |
| X=1 | AUX_1 | copy of register Val(AUX_RD_ADDR.read_addr +1) in auxiliary sensor register map |
| X=2 | Padding byte | Undefined value |
| X=3 | Padding byte | Undefined value |
| X=4 | Padding byte | Undefined value |
| X=5 | Padding byte | Undefined value |
| X=6 | Padding byte | Undefined value |
| X=7 | Padding byte | Undefined value |
| X=8 | ACC_X<7:0> (LSB) | |
| X=9 | ACC_X<15:8> (MSB) | |
| X=10 | ACC_Y<7:0> (LSB) | |
| X=11 | ACC_Y<15:8> (MSB) | |
| X=12 | ACC_Z<7:0> (LSB) | |
| X=13 | ACC_Z<15:8> (MSB) | |

Conditions and Details

Frame rates

The frame sampling rate of the FIFO is defined by the maximum output data rate of the sensors enabled for FIFO sampling. The FIFO sampling configuration is set in register [FIFO_CONFIG 0](#) to [FIFO_CONFIG 1](#). It is possible to select filtered or pre-filtered data as an input to the FIFO. If unfiltered data are selected in register [FIFO_DOWNS.acc_fifo_filt_data](#) for the accelerometer, the sample rate is 1600 Hz. The input data rate to the FIFO can be reduced by selecting a down-sampling factor 2^k in register [FIFO_DOWNS.acc_fifo_downs](#), where $k=[0,1..7]$.

FIFO Overflow

In the case of overflow the FIFO can either stop recording data or overwrite the oldest data. The behavior is controlled by Register [FIFO_CONFIG 0.fifo_stop_on_full](#). When [FIFO_CONFIG 0.fifo_stop_on_full](#) = 0b0, the FIFO logic may delete the oldest frames. If header mode is enabled, the skip frame is prepended at the next FIFO readout, when the free FIFO space falls below the maximum size frame.

If [FIFO_CONFIG 0.fifo_stop_on_full](#) = 0b1, the newest frame may be discarded, if the free FIFO space falls below the maximum size frame. If header mode is enabled, a skip frame is prepended at the next FIFO readout (which is **not** the position where the frame(s) have been discarded).

During a FIFO read operation of the host, no data at the FIFO tail may be dropped. If the host reads the FIFO with a slower rate than it is filled, it may happen that the sensor needs to drop new data, even when [FIFO_CONFIG 0.fifo_stop_on_full](#) = 0b0. These events are recorded in the Register [ERR_REG.fifo_err](#).

Control frames

Control frames are only supported in header mode. There are a number of control frames defined through the *fh_parm* field. These are shown in below.

A skip frame indicates the number of skipped frames after a FIFO overrun occurred, a sensortime frame contains the sensortime when the last sampled frame stored in the FIFO is read, a FIFO input config frames indicates a change in sensor configuration which affects the sensor data.

The FIFO fill level is contained in registers [FIFO_LENGTH 1.fifo_byte_counter 13 8](#) and [FIFO_LENGTH 0.fifo_byte_counter 7 0](#) and includes the control frames, with the exception of the sensortime frame.

| fh_mode<3:0> | Definition |
|---------------------------|-------------------------|
| 0x0 | Skip Frame |
| 0x1 | Sensortime Frame |
| 0x2 | Fifo_Input_Config Frame |
| 0x3 | Reserved |
| 0x4 | Sample Drop Frame |
| 0x5 – 0x7 | Reserved |

Skip Frame (fh_parm=0x0):

In the case of FIFO overflows, a skip_frame is prepended to the FIFO content, when read out next time. The data for the frame consists of one byte and contains the number of skipped frames. When more than 0xFF frames have been skipped, 0xFF is returned. A skip frame is expected always as first frame in a FIFO read burst. A skip frame does not consume memory in the FIFO.

Sensortime Frame (fh_parm=0x1):

The data for the sensortime frame consists content of the Register [SENSORTIME_0](#) to [SENSORTIME_2](#) when the last byte of the last sample frame was read. A sensortime frame is always expected as last frame in the FIFO. A sensortime frame is only sent if the FIFO becomes empty during the burst read. A sensortime frame does not consume memory in the FIFO. Sensortime frames are enabled (disabled) by setting [FIFO_CONFIG_0.fifo_time_en](#) to 0b1 (0b0).

Fifo_Input_Config Frame (fh_parm=0x2):

Whenever the filter configuration of the FIFO input data sources changes, a FIFO input config frame is inserted into the FIFO, before the configuration change becomes active. E.g. when the bandwidth for the accelerometer filter is changed in Register [ACC_CONF](#), a FIFO input config frame is inserted before the first frame with accelerometer data with the new bandwidth configuration. The FIFO input config frame contains one byte of data with the format

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---------|----------|---|---------------|-----------------|----------|----------|------------------|-----------------|
| Content | reserved | | aux_ if_ch | aux_ conf_ch | reserved | reserved | acc_ range_ch | acc_ conf_ch |

aux_if_ch: A write to Register [AUX_IF_CONF](#), [AUX_RD_ADDR](#), or [AUX_WR_ADDR](#) becomes active.

aux_conf_ch: A write to Register [AUX_CONF](#) becomes active.

acc_range_ch: A write to Register [ACC_RANGE](#) becomes active.

acc_conf_ch: A write to Register [ACC_CONF](#) or acc_FIFO_filt_data or acc_FIFO_downsampling in Register [FIFO_DOWNS](#) becomes active.

Sample Drop Frame

A sample drop frame has always one byte payload, defined through

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---------|----------|---|---|---|---|----------|----------|--------------|
| Content | reserved | | | | | aux_drop | reserved | acc_ drop |

Sample drop frame will be inserted after a Fifo_Input_Config frame at the ODR tick at which the sample was dropped and only if no other sensor provides a valid sample at this ODR tick. If another sensor provides valid data, the data of this sensor is just not included and the appropriate header bit of the data frame is not set.

Sample drop frames will be inserted only for transition phases after configuration changes, not for samples dropped between sensor enable and first valid sample. For a detailed description of configuration changes see Section 4.4, Subsection “Configuration Changes”.

FIFO Partial frame reads

When a frame is only partially read through the Register [FIFO_DATA](#) it will be repeated completely with the next access both in headerless and in header mode. In headermode, this includes the header. In the case of a FIFO overflow between the first partial read and the second read attempt, the frame may be deleted.

FIFO overreads

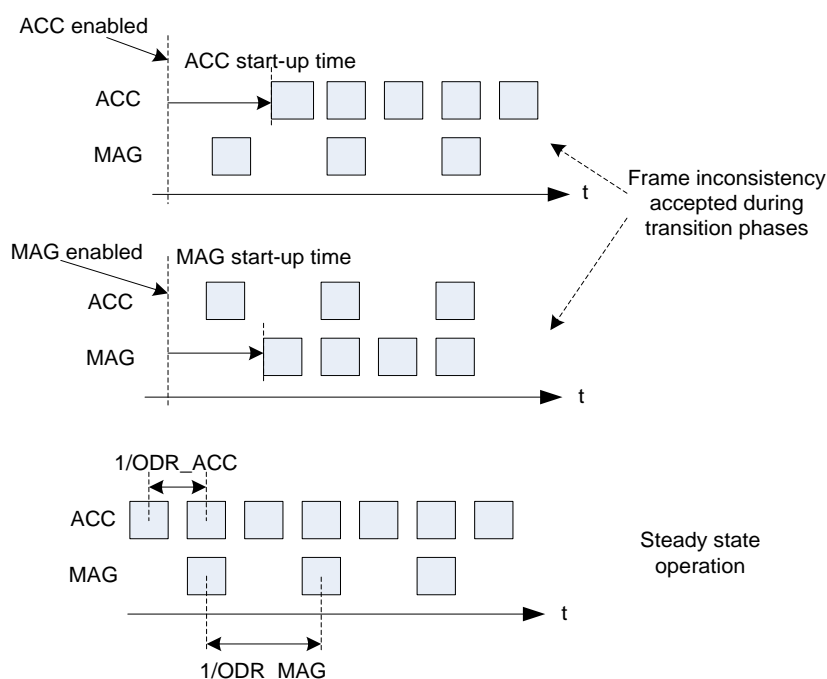
When more data are read from the FIFO than it contains valid data, 0x8000 is returned in headerless mode. In header mode 0x80 indicates an invalid frame.

FIFO data synchronization

All sensor data are sampled with respect to a common ODR time grid. Even if a different ODR is selected for the acceleration and the magnetic sensor the data remains synchronized:

If a frame contains a sample from a sensor element with ODR x , then it must contain also samples of all sensor elements with an ODR $y \geq x$. This applies for steady state operation. In transition phases, it is more important not to lose data, therefore exceptions are possible if the sensor elements with ODR $y \geq x$ do not have data, e.g. due to a sensor configuration change.

FIFO Data Synchronization Scheme in the following figure illustrates the steady state and transient operating conditions.



FIFO synchronization with external interrupts

External interrupts may be synchronized into the FIFO data. For this operation mode the [FIFO_CONFIG_1.fifo_tag_int1_en](#) and [FIFO_CONFIG_1.fifo_tag_int2_en](#) need to be enabled, as well as [INT1_IO_CTRLinput_en](#) and [INT2_IO_CTRLinput_en](#). The fh_ext field in FIFO header will then be set according to the signal at the INT1/INT2 inputs.

FIFO Interrupts

The FIFO supports two interrupts, a FIFO full interrupt and a watermark interrupt:

- The FIFO full interrupt is issued when the FIFO fill level is above the full threshold. The full threshold is reached just before the last two frames are stored in the FIFO.
- The FIFO watermark is issued when the FIFO fill level is equal or above a watermark defined in Register [FIFO_WTM_0](#) and [FIFO_WTM_1](#).

In order to enable/use the FIFO full or watermark interrupts map them on the desired interrupt pin via [INT_MAP_DATA](#).

Both interrupts are suppressed when a read operation on the Register [FIFO_DATA](#) is ongoing. Latched FIFO interrupts will only get cleared, if the status register gets read and the fill level is below the corresponding FIFO interrupt (full or watermark).

FIFO Reset

The user can trigger a FIFO reset by writing the command fifo_flash (0xB0) in [CMD](#). Automatic resets are only performed in the following cases:

- A sensor is enabled or disabled in headerless mode
- A transition between headerless and headermode or vice versa has occurred.
- Size of auxiliary sensor data in a frame changed in header or headerless mode

4.6. Interrupt Features

Global Configuration

The configuration of the interrupt feature engine is described in the Register [FEATURES_IN](#). In order to reconfigure the features, the user must perform a burst read of the whole content from the Register [FEATURES_IN](#), followed by a modification of the content, and finally a burst write of the modified content to the Register [FEATURES_IN](#). The content of the successive bytes read or written in burst mode correspond to the single bytes 0x00 to 0x3F described in [FEATURES_IN](#).

Make sure sensor is initialized properly before the feature configuration is performed (see description in chapter 4.2.)

The output of the interrupt features can be read out of the status registers listed below.

Interrupt feature status registers

| Feature | Output status |
|------------------------|--|
| Step Detector/Counter | INT STATUS 0.step_counter_out |
| Activity Recognition | INT STATUS 0.activity_type_out |
| Tilt on wrist | INT STATUS 0.wrist_tilt_out |
| Double Tap / Tap | INT STATUS 0.wakeup_out |
| Any Motion / No Motion | INT STATUS 0.any_no_motion_out |
| Error Interrupt | INT STATUS 0.error_int_out |

Additionally, the Step counter numeric value is stored in the registers: [STEP_COUNTER_0](#), [STEP_COUNTER_1](#), [STEP_COUNTER_2](#) and [STEP_COUNTER_3](#).

The Error Interrupt signals that the sensor stopped after a fatal error. The Device reinitialization must be done.

The Features (algorithms) have as input data the acceleration samples, which are acquired at 50Hz.

Minimum Bandwidth Settings

If Performance mode is enabled ([ACC_CONF.acc_perf_mode](#) is 0b1, so continuous mode is used), then the features are functioning correctly, regardless to the ODR and the Bandwidth that the Host would set.

If Performance Mode is disabled ([ACC_CONF.acc_perf_mode](#) is 0b0) (device in low power mode), then the minimum ODR setting must comply with the following restrictions:

1. The ODR must be set to minimum 50 Hz for the most features except Double Tap/Tap
2. The ODR must be set to minimum 200 Hz for the use of Double Tap/ Tap feature

If the minimum requirements are not met, the corresponding flag from the register [INTERNAL_STATUS](#) is set.

Axes remapping for interrupt features

If the sensor orientation is different than described in chapter 8.2 the sensor axis must be remapped to use the integrated features.

Axes remapping register allows the host to freely map individual axis to the coordinate system of the used platform. Individual axis can be mapped to any other defined axis. The sign value of the axis can be also configured. For example x axis can be mapped to $-x$ axis, $+y$ axis, $-y$ axis, $+z$ axis or $-z$ axis. Similarly other axis also have its own combination.

Note:

The axis remapping does apply only to the data fetched into the Features. The [DATA 0](#) to [DATA 13](#) registers and FIFO are not affected and should be accordingly remapped on the driver level.

Configuration settings:

1. [FEATURES_IN.general_settings.axes_remapping.map_x_axis](#) – describes which axis shall be mapped to x axis.
2. [FEATURES_IN.general_settings.axes_remapping.map_x_axis_sign](#) – describes whether the mapped axis shall be inverted or not to be inverted.
3. [FEATURES_IN.general_settings.axes_remapping.map_y_axis](#) – describes which axis shall be mapped to y axis.
4. [FEATURES_IN.general_settings.axes_remapping.map_y_axis_sign](#) – describes whether the mapped axis shall be inverted or not to be inverted.
5. [FEATURES_IN.general_settings.axes_remapping.map_z_axis](#) – describes which axis shall be mapped to z axis.
6. [FEATURES_IN.general_settings.axes_remapping.map_z_axis_sign](#) – describes whether the mapped axis shall be inverted or not to be inverted.

Step Detector / Step Counter

The Step Counter algorithm is optimized on high accuracy, while Step Detector is optimized on low latency. Both are running in parallel, once enabled, but the Step Detector interrupt output is mutually exclusive with the Step Counter watermark interrupt. The Step Counter is optimized for wearable devices and wrist use case.

In addition the Step Counter implements the function required for step counting in Android 4.4 and higher as well: https://source.android.com/devices/sensors/sensor-types.html#step_counter.

The Step Detector implements the function required for step counting in Android 4.4 and higher: https://source.android.com/devices/sensors/sensor-types.html#step_detector.

Step Counter:

Step Counter can be enabled in [FEATURES_IN.step_counter.settings_26.en_counter](#).

The step counter accumulates the steps detected by the step detector interrupt, and makes available the 32 bit current step counter value in the following 4 registers, each holding 8bit: [STEP_COUNTER_0](#), [STEP_COUNTER_1](#), [STEP_COUNTER_2](#) and corresponding [STEP_COUNTER_3](#). There are situations when the step counting value is different than the sum of steps detected by the step detector.

By enabling the [FEATURES_IN.step_counter.settings_26.reset_counter](#) flag, the accumulated step number value is reset. Afterwards, the value of this flag is automatically reset and counting is restarted. The step counter watermark option can be useful if host needs to receive an interrupt every time a certain number of steps were counted. If [FEATURES_IN.step_counter.settings_26.watermark_level](#) is set to 10 (holding an implicit factor of 20x), every 200 steps are counted an interrupt will be raised on [INT_STATUS_0.step_counter_out](#). As the steps are buffered internally, the output may be triggered between 200-210 steps. The exact number of steps recorded is available in the registers [STEP_COUNTER_0](#), [STEP_COUNTER_1](#), [STEP_COUNTER_2](#) and [STEP_COUNTER_3](#). When the watermark level was reached, the corresponding interrupt bit is asserted, [INT_STATUS_0.step_counter_out](#).

Step Detector:

The Step Detector feature is optimized for low latency. If [FEATURES_IN.step_counter.settings_26.en_detector](#) is set, an interrupt is triggered for every step detected. So, every time a new step is detected, it asserts the corresponding interrupt output [INT_STATUS_0.step_counter_out](#).

Step Counter Presets (phone/wrist use case):

The integrated step counter can be configured to work with either phone platform or wrist platform. The step counter parameters [FEATURE_IN.step_counter.settings_1.param_1](#) to [FEATURE_IN.step_counter.settings_25.param_25](#) must be configured from the host side to either of the configurations in the following table.

| Parameter Name | Phone Configuration | Wrist Configuration |
|----------------|---------------------|---------------------|
| PARAM_1 | 306 | 301 |
| PARAM_2 | 30950 | 31700 |
| PARAM_3 | 132 | 315 |
| PARAM_4 | 27804 | 31451 |
| PARAM_5 | 7 | 4 |
| PARAM_6 | 30052 | 31551 |
| PARAM_7 | 32426 | 27853 |
| PARAM_8 | 1375 | 1219 |
| PARAM_9 | 2750 | 2437 |
| PARAM_10 | 1375 | 1219 |
| PARAM_11 | -5994 | -6420 |
| PARAM_12 | 16879 | 17932 |
| PARAM_13 | 1 | 1 |
| PARAM_14 | 12 | 39 |
| PARAM_15 | 12 | 25 |
| PARAM_16 | 74 | 150 |
| PARAM_17 | 160 | 160 |
| PARAM_18 | 0 | 1 |
| PARAM_19 | 12 | 12 |
| PARAM_20 | 15600 | 15600 |
| PARAM_21 | 256 | 256 |
| PARAM_22 | 0 | 1 |
| PARAM_23 | 0 | 3 |
| PARAM_24 | 0 | 1 |
| PARAM_25 | 0 | 14 |

By default, the wrist configuration is available for use. If the platform is a wrist operated device, then there is no need to overwrite the step counter parameter values. If the configuration needs to be modified then the following steps must be followed:

1. Disable step counter, step detector, and activity detection
2. Modify the 25 parameters of step counter
3. Enable step counter, step detector, and activity detection

After re-enabling the features, the new parameters value will be used.

Customized Step Counter Sensitivity Configuration (overwrites Step Counter Presets)

The Step Counter and Detector sensitivity can be modified manually by setting the step counter parameters in the register map with the support of the corresponding field application engineer. The default parameters are set by the step counter preset (phone/wrist) which is used.

Configuration settings:

1. [FEATURES_IN.step_counter.settings_26.watermark_level](#) - Watermark level; the Step-counter will trigger output every time this number of steps are counted. Holds implicitly a 20x factor, so the range is 0 to 20460, with resolution of 20 steps. If 0, the Step Counter watermark is disabled and Step Detector enabled.
2. [FEATURES_IN.step_counter.settings_26.reset_counter](#) – flag to reset the counted steps. This is only interpreted if the step counter is enabled.
3. [FEATURES_IN.step_counter.settings_26.en_counter](#) – indicates if the Step Counter feature is enabled or not.
4. [FEATURES_IN.step_counter.settings_26.en_detector](#) – indicates if the Step Detector feature is enabled or not.

[FEATURES_IN.step_counter.settings_1.param_1](#) – there are 25 parameters, which configure the use case (wrist or phone).



Walking activity recognition

This feature can detect the actual walking activity status of a user. It can distinguish between walking, running and still.

The change of the status can be read in [INT STATUS 0.activity type out](#); the status itself can be read in [ACTIVITY TYPE.activity type out](#). The walking activity recognition can be enabled by [FEATURES IN.step counter.settings 26.en activity](#). Step counter has to be enabled as well by [FEATURES IN.step counter settings 26.en counter](#).

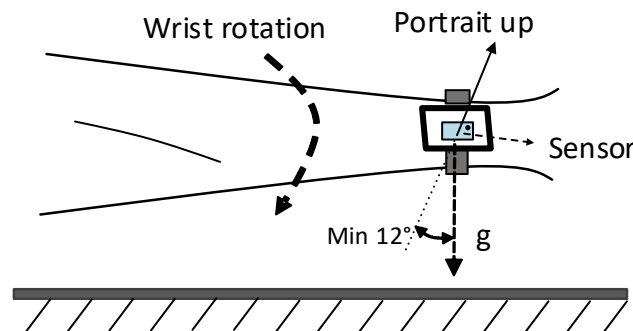
Configuration settings:

[FEATURES IN.step counter.settings 26.en activity](#) – enables walking activity recognition.

**Tilt on Wrist**

How to perform the “Tilt on Wrist” gesture ideally:

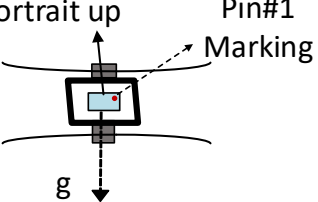
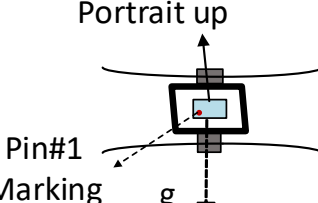
1. Wrap the device on your wrist.
2. Lift the hand wearing the device up to a comfortable position and tilt it to minimum 12° to gravity (see the image below for the valid position)
3. Wait for a second.



Valid position for Tilt on Wrist

The Tilt on Wrist Interrupt is detected with one accelerometer axis. Therefore, you should make sure that the active axis is mapped to the portrait up direction of the device by using the axis remapping functionality of BMA456. See the examples in the table below.

The table below shows the required remapping configurations for the Tilt on Wrist function based on the sensor's different orientations inside the device. More information about how to remap the axis direction can be found in the general interrupt description in Chapter 4.6.

| No. | Sensor direction | Mount layer | Register configuration | Remarks |
|-----|--|-------------|---|---|
| 1 | Portrait up  | Top layer | map_x_axis = 0 map_x_axis_sign = 0 map_y_axis = 1 map_y_axis_sign = 0 map_z_axis = 2 map_z_axis_sign = 0 | X = +x axis Y = +y axis Z = +z axis Same as the original mapping |
| 2 | Portrait up  | Top layer | map_x_axis = 0 map_x_axis_sign = 1 map_y_axis = 1 map_y_axis_sign = 1 map_z_axis = 2 map_z_axis_sign = 0 | X = -x axis Y = -y axis Z = +z axis x, y inverted |



| | | | | |
|---|--|--------------|--|---|
| 3 | | Top layer | $\text{map_x_axis} = 1$ $\text{map_x_axis_sign} = 1$ $\text{map_y_axis} = 0$ $\text{map_y_axis_sign} = 0$ $\text{map_z_axis} = 2$ $\text{map_z_axis_sign} = 0$ | $X = -y$ axis $Y = +x$ axis $Z = +z$ axis x, y swapped, and y inverted |
| 4 | | Top layer | $\text{map_x_axis} = 1$ $\text{map_x_axis_sign} = 0$ $\text{map_y_axis} = 0$ $\text{map_y_axis_sign} = 1$ $\text{map_z_axis} = 2$ $\text{map_z_axis_sign} = 0$ | $X = +y$ axis $Y = -x$ axis $Z = +z$ axis x, y swapped, and x inverted |
| 5 | | Bottom layer | $\text{map_x_axis} = 1$ $\text{map_x_axis_sign} = 0$ $\text{map_y_axis} = 0$ $\text{map_y_axis_sign} = 0$ $\text{map_z_axis} = 2$ $\text{map_z_axis_sign} = 1$ | $X = y$ axis $Y = x$ axis $Z = -z$ axis x, y swapped, and z inverted |
| 6 | | Bottom layer | $\text{map_x_axis} = 1$ $\text{map_x_axis_sign} = 1$ $\text{map_y_axis} = 0$ $\text{map_y_axis_sign} = 1$ $\text{map_z_axis} = 2$ $\text{map_z_axis_sign} = 1$ | $X = -y$ axis $Y = -x$ axis $Z = -z$ axis x, y swapped, and x, y, z inverted |
| 7 | | Bottom layer | $\text{map_x_axis} = 0$ $\text{map_x_axis_sign} = 1$ $\text{map_y_axis} = 1$ $\text{map_y_axis_sign} = 0$ $\text{map_z_axis} = 2$ $\text{map_z_axis_sign} = 1$ | $X = -x$ axis $Y = y$ axis $Z = -z$ axis x, z inverted |
| 8 | | Bottom layer | $\text{map_x_axis} = 0$ $\text{map_x_axis_sign} = 0$ $\text{map_y_axis} = 1$ $\text{map_y_axis_sign} = 1$ $\text{map_z_axis} = 2$ $\text{map_z_axis_sign} = 1$ | $X = x$ axis $Y = -y$ axis $Z = -z$ axis y, z inverted |

Configuration settings

[FEATURES_IN.wrist_tilt.settings.enable](#) – enables/disables the Tilt on Wrist feature.



Double tap / Tap detection

The gesture “Tap / Double tap” can be enabled by [FEATURES_IN.tap_doubletap.settings.enable](#). The detection is done by measuring the acceleration on z-direction.

By default double tap detection is enabled. To enable single tap detection (and disable double tap), you can configure the parameter: [FEATURES_IN.tap_doubletap.settings.single_tab_en](#) – 1 bit – flag.

The detection sensitivity can be customized using the register [FEATURES_IN.tap_doubletap.settings.sensitivity](#). The sensitivity range is from 0 (high) to 7 (low).

How to perform the gesture ideally:

1. Use your index finger to tap on the device slightly harder than how you touch a button on a touchscreen.
2. Tap again directly, if “Double tap” detection is enabled.
3. Wait for half a second.

Configuration settings

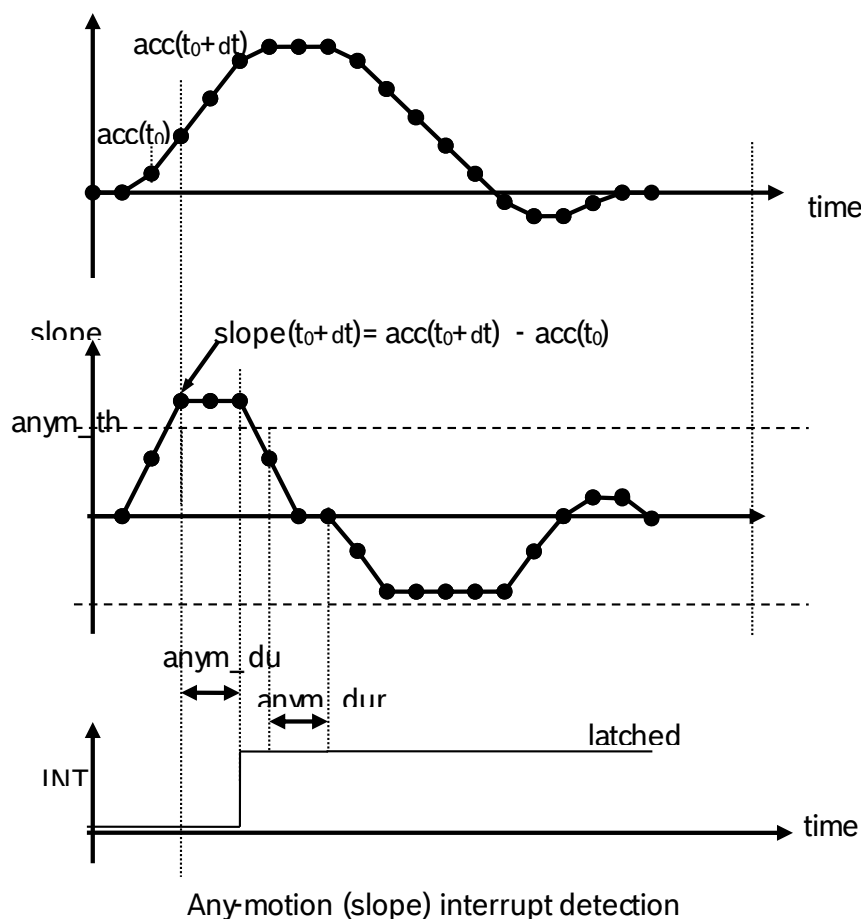
1. [FEATURES_IN.tap_doubletap.settings.enable](#) – enables/disables the Wakeup feature.
2. [FEATURES_IN.tap_doubletap.settings.sensitivity](#) – configures the detection sensitivity which ranges from 0 (high) to 7 (low).
3. [FEATURES_IN.tap_doubletap.settings.single_tab_en](#) – enables single tap detection. By default, double tap is enabled.

Any Motion / No motion detection

Any Motion Detection:

The any-motion detection uses the slope between two acceleration signals to detect changes in motion. The interrupt is configured by setting at least one of the following flags: [FEATURES_IN.any_motion.settings_2.en_x](#), [FEATURES_IN.any_motion.settings_2.en_y](#) and [FEATURES_IN.any_motion.settings_2.en_z](#), respectively for each axis.

It generates an interrupt when the absolute value of the slope (the difference between two accelerations) exceeds the preset [FEATURES_IN.any_motion.settings_1.threshold](#) for a certain number, [FEATURES_IN.any_motion.settings_2.duration](#), of consecutive data points.



The slope (difference) is being computed between the current acceleration sample and the reference sample. The reference sample is updated while the Anymotion is detected; basically this means the reference is the last state when sensor detected Anymotion.

If the same number of data points falls below the [FEATURES_IN.any_motion.settings_1.threshold](#), the interrupt is reset.

No Motion Detection:

The interrupt engine can also be configured as a No-motion interrupt, when register [FEATURES_IN.any_motion.settings_1.nomotion_sel](#) is set.

No-motion is generated when the slope on all selected axis remains smaller than a programmable [FEATURES_IN.any_motion.settings_1.threshold](#) for a programmable time. The signals and timings relevant to the no-motion interrupt functionality are depicted in the figure below.

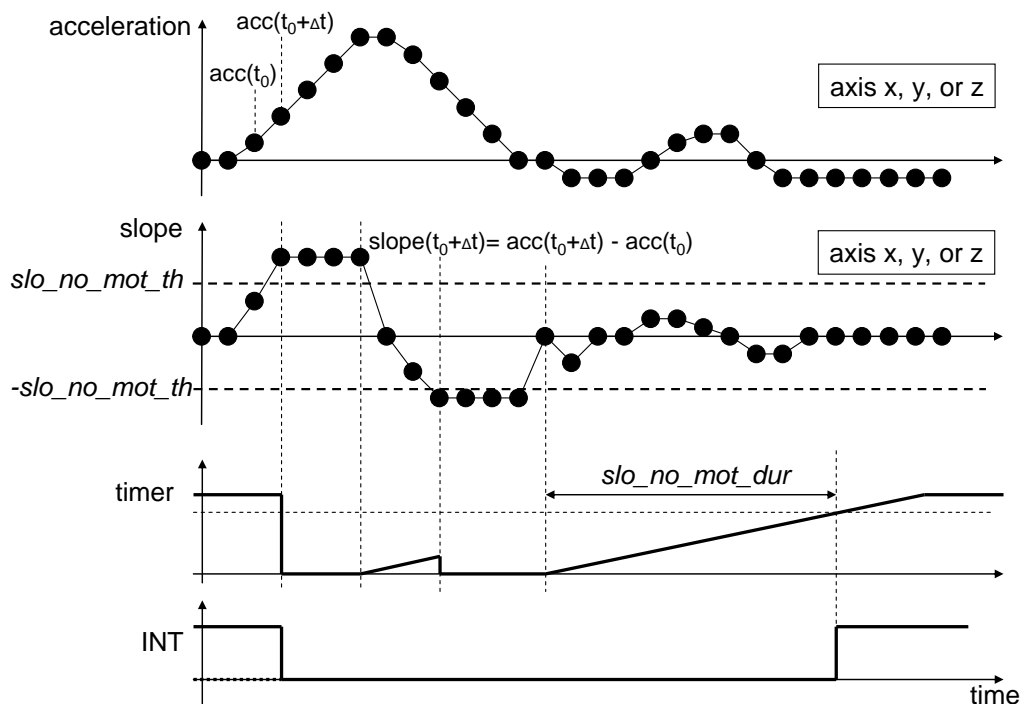


Figure 1 Signal timings No-motion interrupt

Register [FEATURES_IN.any_motion.settings_2.duration](#) defines the number of consecutive slope data points of the selected axis which must exceed the threshold for an interrupt to be asserted.

Configuration settings:

1. [FEATURES_IN.any_motion.settings_2.duration](#) – the number of consecutive data points for which the threshold condition must be respected, for interrupt assertion.
2. [FEATURES_IN.any_motion.settings_1.threshold](#) – the slope threshold.
3. [FEATURES_IN.any_motion.settings_2.en_x](#) – indicates if this feature is enabled for x axis
4. [FEATURES_IN.any_motion.settings_2.en_y](#) – indicates if this feature is enabled for y axis
5. [FEATURES_IN.any_motion.settings_2.en_z](#) – indicates if this feature is enabled for z axis
6. [FEATURES_IN.any_motion.settings_1.nomotion_sel](#) – indicates if No-motion (1) or Any-motion (0) is selected.

4.7. General Interrupt Pin configuration

Electrical Interrupt Pin Behavior

Both interrupt pins INT1 and INT2 can be configured to show the desired electrical behavior. Interrupt pins can be enabled in [INT1_IO_CTRL.output_en](#) respectively [INT2_IO_CTRL.output_en](#). The characteristic of the output driver of the interrupt pins may be configured with bits [INT1_IO_CTRL.od](#) and [INT2_IO_CTRL.od](#). By setting these bits to 0b1, the output driver shows open-drive characteristic, by setting the configuration bits to 0b0, the output driver shows push-pull characteristic.

The electrical behavior of the Interrupt pins, whenever an interrupt is triggered, can be configured as either “active-high” or “active-low” via [INT1_IO_CTRL.lv](#) respectively [INT2_IO_CTRL.lv](#).

Both interrupt pins can be configured as input pins via [INT1_IO_CTRL.input_en](#) respectively [INT2_IO_CTRL.input_en](#). This is necessary when FIFO tag feature is used (see chapter 0). If both are enabled, the input (e.g. marking FIFO) is driven by the interrupt output. BMA456 supports edge and level triggered interrupt inputs, this can be configured through [INT1_IO_CTRL.edge_ctrl](#) respectively [INT2_IO_CTRL.edge_ctrl](#).

BMA456 supports non-latched and latched interrupts modes for data-ready, FIFO full and FIFO watermark. The mode is selected by [INT_LATCH.int_latch](#). The feature interrupts described in chapter 4.6 support only latched mode described below.

In latched mode an asserted interrupt status in [INT_STATUS_0](#) or [INT_STATUS_1](#) and the selected pin are cleared if the corresponding status register is read. If more than one interrupt pin is used in latched mode, all interrupts in [INT_STATUS_0](#) should be mapped to one pin and all interrupts in [INT_STATUS_1](#) should be mapped to the other pin. If just one interrupt pin is used all interrupts may be mapped to this pin. If the activation condition still holds when it is cleared, the interrupt status is asserted again when the interrupt condition holds again.

In the non-latched mode (only for data-ready, FIFO full and FIFO watermark) the interrupt status bit and the selected pin are reset as soon as the activation condition is not valid anymore.

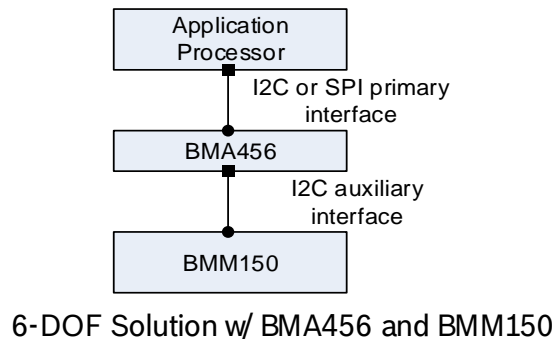
Interrupt Pin Mapping

In order, for the Host to react to the features output, they can be mapped to the external PIN1 or PIN2, by setting the corresponding bits from the registers [INT1_MAP](#), respectively [INT2_MAP](#). To disconnect the features outputs to the external pins, the same corresponding bits must be reset, from the registers, [INT1_MAP](#), respectively [INT2_MAP](#).

Once a feature triggered the output pin, the Host can read out the corresponding bit from the register, [INT_STATUS_0](#) (Feature Interrupts) or [INT_STATUS_1](#) (FIFO and data ready).

4.8. Auxiliary Sensor Interface

The auxiliary interface allows to attach one auxiliary sensor (e.g. magnetometer) on dedicated auxiliary sensor interface as shown in Figure...



Structure and Concept

The BMA456 controls the data acquisition of the auxiliary sensor and presents the data to the application processor through the primary I2C or SPI interface. No other I2C master or slave devices must be attached to the auxiliary sensor interface.

The BMA456 autonomously reads the sensor data from a compatible auxiliary sensor without intervention of the application processor and stores the data in its data registers and FIFO. The initial setup of the auxiliary sensor after power-on is done through indirect addressing.

The main benefits of the auxiliary sensor interface are

- Synchronization of sensor data of auxiliary sensor and accelerometer. This results in an improved sensor data fusion quality.
- Usage of the BMA456 FIFO for auxiliary sensor data (BMM150 does not have a FIFO). This is important for monitoring applications.

Interface Configuration

The configuration registers that control the auxiliary sensor interface operation, are only affecting the interface to the auxiliary sensor, not the configuration of the sensor itself (this must be done in setup mode).

There are three basis configurations of the auxiliary sensor interface:

- No auxiliary sensor access
- Setup mode: Auxiliary sensor access in manual mode
- Data mode: Auxiliary sensor access through hardware readout loop.

The setup of the auxiliary sensor itself must be done through the primary interface using indirect addressing in setup mode. When collecting sensor data, the BMA456 autonomously triggers the



measurement of the auxiliary sensor using the auxiliary sensor forced mode and the data readout from the auxiliary sensor (data mode).

In setup mode, the auxiliary sensor may be configured and trim data may be read out from the auxiliary sensor. In the data mode the auxiliary sensor data are continuously copied into BMA456 registers and may be read out from BMA456 directly over the primary interface. For a BMM150 magnetometer, these are the auxiliary sensor data itself and Hall resistance, temperature is not required. The table below shows how to configure these three modes using the registers [PWR_CONF](#), [PWR_CTRL](#), and [AUX_IF_CONF.aux_manual_en](#).

| Mode | AUX_IF_CONF.aux_manual_en | PWR_CONF.adv_power_save | PWR_CTRL.aux_en |
|----------------------------|---|---|---------------------------------|
| No auxiliary sensor access | 1 | 1 | 0 |
| Setup mode | 1 | 0 | 0 |
| Data mode | 0 | x | 1 |

The auxiliary sensor interface mode may be enabled by setting bit [IF_CONF.if_mode](#) according to the following table.

| IF_CONF.if_mode | Result |
|---------------------------------|---------------------------------|
| 0 | Secondary IF disabled (default) |
| 1 | AuxIF enabled |

The auxiliary sensor interface operates at 400 kHz. This results in an I2C readout delay of about 250 us for 10 bytes of data.

Setup mode ([AUX_IF_CONF.aux_manual_en](#) = 0b1)

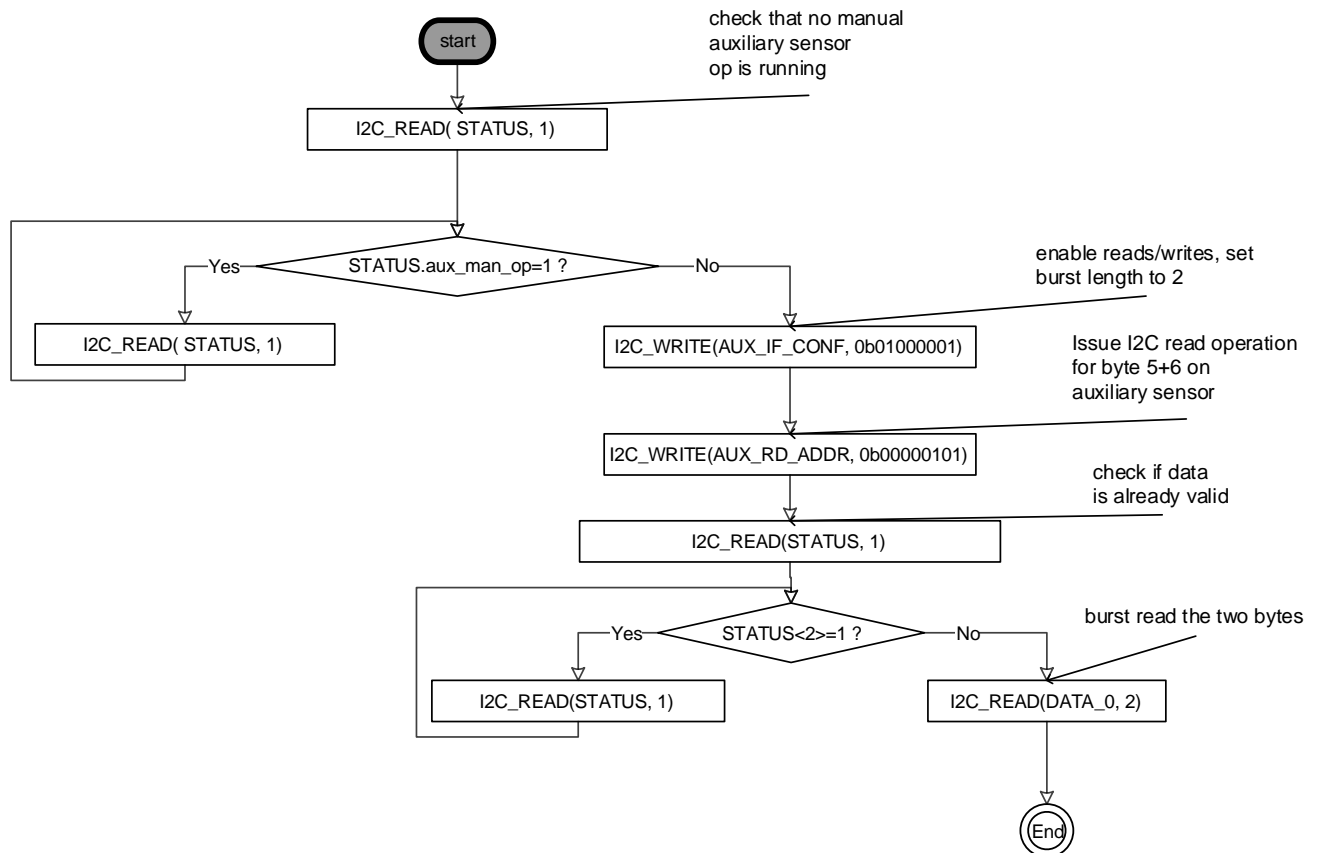
Through the primary interface the auxiliary sensor may be accessed using indirect addressing through the [AUX_*](#) registers. [AUX_RD_ADDR](#) and [AUX_WR_ADDR](#) define the address of the register to read/write in the auxiliary sensor register map and triggers the operation itself, when the auxiliary sensor interface is enabled through [PWR_CTRL.aux_en](#).

For reads, the number of data bytes defined in [AUX_IF_CONF.aux_rd_burst](#) are read from the auxiliary sensor and written into the BMA456 Register [DATA_0](#) to [DATA_7](#). For writes only single bytes are written, independent of the settings in [AUX_IF_CONF.aux_rd_burst](#). The data for the I2C write to auxiliary sensor must be stored in [AUX_WR_DATA](#) before the auxiliary sensor register address is written into [AUX_WR_ADDR](#).

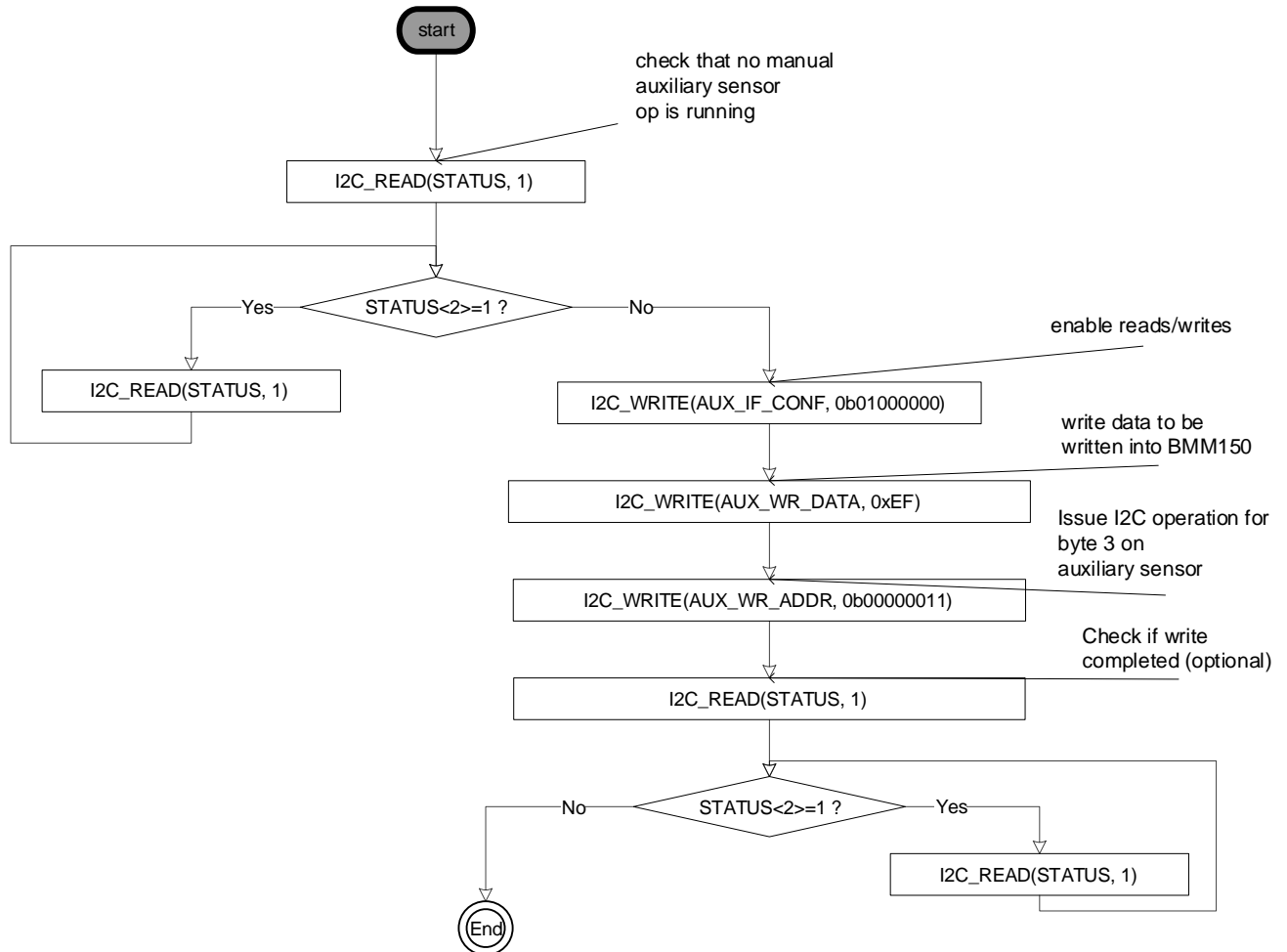
When a read or write operation is triggered by writing to [AUX_RD_ADDR](#) and [AUX_WR_ADDR](#), [STATUS.aux_man_op](#) is set and it is reset when the operation is completed. For reads the [DATA_0](#) to [DATA_7](#) contains the read data, for writes [AUX_WR_DATA](#) may be overwritten again.

Configuration phase of the auxiliary sensor.

Example: Read bytes 5 and 6 of auxiliary sensor



Example: Write 0xEF into byte 3 of auxiliary sensor



Data mode ([AUX_IF_CONF.aux_manual_en=0](#))

[AUX_RD_ADDR.read_addr](#) defines the address of the data register from which to read the number of data bytes configured in [AUX_IF_CONF.aux_rd_burst](#) from AUX_0... AUX_7 data of the auxiliary sensor. These data are stored in the [DATA_0](#) up to [DATA_7](#) register. The data ready status is set in [STATUS.drdy_aux](#), it is typically cleared through reading one of the [DATA_0](#) to [DATA_7](#) registers.

[AUX_WR_ADDR.write_addr](#) defines the register address of auxiliary sensor to start a measurement in forced mode in the auxiliary sensor register map. The delay (time offset) between triggering an auxiliary sensor measurement and reading the measurement data is specified in [AUX_CONF.aux_offset](#). Reading of the data is done in a single I2C read operation with a burst length specified in [AUX_IF_CONF.aux_rd_burst](#). For BMM150 [AUX_IF_CONF.aux_rd_burst](#) should be set to 0b11, i.e. 8 bytes. If [AUX_IF_CONF.aux_rd_burst](#) is set to a value lower than 8 bytes, the remaining auxiliary sensor data in the Register [DATA_0](#) to [DATA_7](#) and the FIFO are undefined.

It is recommended to disable the auxiliary sensor interface ([IF_CONF.if_mode](#)=0b0) before setting up [AUX_RD_ADDR.read_addr](#) and [AUX_WR_ADDR.write_addr](#) for the data mode. This does not put the auxiliary sensor itself into suspend mode but avoids gathering unwanted data during this phase. Afterwards the auxiliary sensor interface can be enabled ([IF_CONF.if_mode](#)=0b1) again.

Delay (Time Offset)

BMA456 supports starting the measurement of the sensor at the auxiliary sensor interface between 2.5 and 37.5 ms before the Register DATA are updated. This offset is defined in [AUX_CONF.aux_offset](#). If set to 0b0, the measurement is done right after the last Register DATA update, therefore this measurement will be included in the next register DATA update.

4.9. Sensor Self-Test

The BMA456 has a comprehensive self test function for the MEMS element 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 8g. The self-test is activated for all axes by writing [ACC_SELF_TEST.acc self test en](#) = 1b1. The self-test is disabled by writing [ACC_SELF_TEST.acc self test en](#) = 1b0. It is possible to control the direction of the deflection through bit [ACC_SELF_TEST.acc self test sign](#). The excitation occurs in positive (negative) direction if [ACC_SELF_TEST.acc self test sign](#) = 1b1 (b0). The amplitude of the deflection has to be set low by writing [ACC_SELF_TEST.acc self test amp](#) = 1b0. 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. The table below shows the minimum differences for each axis in order for the self test to pass. The actually measured signal differences can be significantly larger.

Self-test: Resulting minimum difference signal for BMA456.

| | x-axis signal | y-axis signal | z-axis signal |
|--------|---------------|---------------|---------------|
| BMA456 | 1800 mg | 1800 mg | 1800 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, and enable desired interrupts.

The recommended self test procedure is as follows:

1. Enable accelerometer with register [PWR_CTRL.acc en](#)=1b1.
2. Set ±8g range in register [ACC_RANGE.acc range](#)
3. Set self test amplitude to low by setting [ACC_SELF_TEST.acc self test amp](#) = 1b0
4. Set [ACC_CONF.acc odr=1600Hz](#), Continuous sampling mode, [ACC_CONF.acc bwp](#)=norm_avg4, [ACC_CONF.acc perf mode](#)=1b1.
5. Wait for > 2 ms
6. Enable self-test and set positive self-test polarity ([ACC_SELF_TEST.acc self test sign](#)= 1b1)
7. Wait for > 50ms
8. Read and store positive acceleration value of each axis from registers [DATA_8](#) to [DATA_13](#)
9. Enable self-test and set negative self-test polarity [ACC_SELF_TEST.acc self test sign](#)= 1b0)
10. Wait for > 50ms
11. Read and store negative acceleration value of each axis from registers [DATA_8](#) to [DATA_13](#)
12. Calculate difference of positive and negative acceleration values and compare against threshold values

4.10. Offset Compensation

BMA456 offers manual compensation as well as inline calibration.

Offset compensation is performed with pre-filtered data, and the offset is then applied to both, pre-filtered 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).

The public offset compensation Registers [OFFSET_0](#) to [OFFSET_2](#) are images of the corresponding registers in the NVM. With each image update the contents of the NVM registers are written to the public registers. The public registers can be overwritten by the user at any time.

The offset compensation registers have a width of 8 bit using two's complement notation. The offset resolution (LSB) is 3.9 mg and the offset range is ± 0.5 g. Both are independent of the range setting. Offset compensation needs to be enabled through [NV_CONF.acc_off_en](#) = 0b1

Manual Offset Compensation

The contents of the public compensation Register [OFFSET_0](#) to [OFFSET_2](#) may be set manually via the digital interface. After modifying the Register [OFFSET_0](#) to [OFFSET_2](#) the next data sample is not valid.

Offset compensation needs to be enabled through [NV_CONF.acc_off_en](#).

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 manual offset compensation to determine the proper compensation values and then storing these values permanently in the NVM.

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.

4.11. Non-Volatile Memory

The registers [NV_CONF](#) and [OFFSET_0](#) to [OFFSET_2](#) have an NVM backup which are accessible by the user.

The content of the NVM is loaded to the image registers after a reset (either POR or softreset). As long as the image update is in progress, [STATUS.cmd_rdy](#) is 0b0, otherwise it is 0b1.

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

Writing to the NVM is a 4-step procedure:

1. Set [PWR_CONF.adv_power_save](#) = 0b0
2. Write the new contents to the image registers.
3. Write 0b1 to bit [NVM_CONF.nvm_prog_en](#) in order to unlock the NVM.
4. Write *prog_nvm* to the [CMD](#) register 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 [STATUS.cmd_rdy](#). While [STATUS.cmd_rdy](#) = 0b0, the write process is still in progress; when [STATUS.cmd_rdy](#) = 0b1, writing is completed. An NVM write cycle can only be initiated, if [PWR_CONF.adv_power_save](#) = 0b0.

Until boot phase is finished (after POR or softreset), the serial interface is not operational. The NVM shadow registers must not be accessed during an ongoing NVM command (initiated through the Register [CMD](#)). In all other cases, register can be read or written.

As long as an NVM read (during sensor boot and soft reset) or an NVM write is ongoing, writes to sensor registers are discarded, reads return the Register [STATUS](#) independent of the read address.

4.12. Soft-Reset

A softreset can be initiated at any time by writing the command *softreset* (0xB6) to register [CMD](#). The softreset performs a fundamental reset to the device which is largely equivalent to a power cycle. Following a delay, all user configuration settings are overwritten with their default state (setting stored in the NVM) wherever applicable. This command is functional in all operation modes but must not be performed while NVM writing operation is in progress.



5. Register Description

5.1. General Remarks

Registers can be read and written in all power configurations with the exception of [FEATURES_IN](#) and [FIFO_DATA](#) which need [PWR_CONF.adv power save](#) set to 0b0.

5.2. Register Map

| | | | |
|------------|-----------|------------|----------|
| read/write | read only | write only | reserved |
|------------|-----------|------------|----------|

| ID: | | | | | | | | | | |
|------------------|------------------------------|---------------|-----------|---|---|---------|-------------------|--------------------|------------------|------------------|
| Register Address | Register Name | Default Value | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x7E | CMD | 0x00 | cmd | | | | | | | |
| 0x7D | PWR_CTL | 0x00 | reserved | | | | | acc_en | reserved | aux_en |
| 0x7C | PWR_CONF | 0x03 | reserved | | | | | | fifo_self_wakeup | adv_power_save |
| 0x7B | - | - | reserved | | | | | | | |
| ... | - | - | reserved | | | | | | | |
| 0x74 | - | - | reserved | | | | | | | |
| 0x73 | OFFSET_2 | 0x00 | off_acc_z | | | | | | | |
| 0x72 | OFFSET_1 | 0x00 | off_acc_y | | | | | | | |
| 0x71 | OFFSET_0 | 0x00 | off_acc_x | | | | | | | |
| 0x70 | NV_CONF | 0x00 | reserved | | | | acc_off_en | i2c_wdt_en | i2c_wdt_sel | spi_en |
| 0x6F | - | - | reserved | | | | | | | |
| 0x6E | - | - | reserved | | | | | | | |
| 0x6D | ACCSelfTestI | 0x00 | reserved | | | | acc_self_test_amp | acc_self_test_sign | reserved | acc_self_test_en |
| 0x6C | - | - | reserved | | | | | | | |
| 0x6B | IF_CONF | 0x00 | reserved | | | if_mode | reserved | | | spi3 |
| 0x6A | NVM_CONF | 0x00 | reserved | | | | | | nvm_prog_en | reserved |
| 0x69 | - | - | reserved | | | | | | | |
| ... | - | - | reserved | | | | | | | |
| 0x60 | - | - | reserved | | | | | | | |



| | | | | | | | | | | |
|------|--------------------------------|------|-----------------|----------------------|-------------|----------------|------------------|-------------------|------------------|-------------------|
| 0x5F | INTERNAL ERROR | 0x00 | reserved | | | | | int_err_2 | int_err_1 | reserved |
| 0x5E | FEATURES_IN | 0x00 | features_in | | | | | | | |
| 0x5D | - | - | reserved | | | | | | | |
| ... | - | - | reserved | | | | | | | |
| 0x5A | - | - | reserved | | | | | | | |
| 0x59 | INIT_CTRL | 0x90 | init_ctrl | | | | | | | |
| 0x58 | INT_MAP_DATA | 0x00 | reserved | int2_drdy | int2_fwm | int2_ffull | reserved | int1_drdy | int1_fwm | int1_ffull |
| 0x57 | INT2_MAP | 0x00 | error_int_out | any_no_motion_output | wakeup_out | reserved | wrist_tilt_out | activity_type_out | step_counter_out | reserved |
| 0x56 | INT1_MAP | 0x00 | error_int_out | any_no_motion_output | wakeup_out | reserved | wrist_tilt_out | activity_type_out | step_counter_out | reserved |
| 0x55 | INT_LATCH | 0x00 | reserved | | | | | | | int_latch |
| 0x54 | INT2_IO_CTRL | 0x00 | reserved | | | input_en | output_en | od | lvl | edge_ctrl |
| 0x53 | INT1_IO_CTRL | 0x00 | reserved | | | input_en | output_en | od | lvl | edge_ctrl |
| 0x52 | - | - | reserved | | | | | | | |
| ... | - | - | reserved | | | | | | | |
| 0x50 | - | - | reserved | | | | | | | |
| 0x4F | AUX_WRITE_DATA | 0x02 | write_data | | | | | | | |
| 0x4E | AUX_WRITE_ADDR | 0x4C | write_addr | | | | | | | |
| 0x4D | AUX_READ_ADDR | 0x42 | read_addr | | | | | | | |
| 0x4C | AUX_IF_CONF | 0x83 | aux_manual_en | reserved | | | | | aux_rd_burst | |
| 0x4B | AUX_DEVICE_ID | 0x20 | i2c_device_addr | | | | | | | reserved |
| 0x4A | - | - | reserved | | | | | | | |
| 0x49 | FIFO_CONFIG_1 | 0x10 | reserved | fifo_acc_en | fifo_aux_en | fifo_header_en | fifo_tag_int1_en | fifo_tag_int2_en | reserved | |
| 0x48 | FIFO_CONFIG_0 | 0x02 | reserved | | | | | | fifo_time_en | fifo_stop_on_full |



| | | | | | | | | |
|------|---|------|------------------------|--------------------|--------------------------|----------------------|----------|-------------------|
| 0x47 | FIFO_W TM_1 | 0x02 | reserved | | | fifo_water_mark_12_8 | | |
| 0x46 | FIFO_W TM_0 | 0x00 | fifo_water_mark_7_0 | | | | | |
| 0x45 | FIFO_D OWNS | 0x80 | acc_fifo_ filt_data | acc_fifo_downs | | | reserved | |
| 0x44 | AUX_CO NF | 0x46 | aux_offset | | | aux_odr | | |
| 0x43 | - | - | reserved | | | | | |
| 0x42 | - | - | reserved | | | | | |
| 0x41 | ACC_RA NGE | 0x01 | reserved | | | | | acc_range |
| 0x40 | ACC_CO NF | 0xA8 | acc_perf _mode | acc_bwp | | | acc_odr | |
| 0x3F | - | - | reserved | | | | | |
| ... | - | - | reserved | | | | | |
| 0x2B | - | - | reserved | | | | | |
| 0x2A | INTERN AL_STAT US | 0x00 | odr_high _error | odr_50hz _error | axes_re map_err or | message | | |
| 0x29 | - | - | reserved | | | | | |
| 0x28 | - | - | reserved | | | | | |
| 0x27 | ACTIVIT Y_TYPE | 0x00 | reserved | | | | | activity_type_out |
| 0x26 | FIFO_DA TA | 0x00 | fifo_data | | | | | |
| 0x25 | FIFO_LE NGTH_1 | 0x00 | reserved | | fifo_byte_counter_13_8 | | | |
| 0x24 | FIFO_LE NGTH_0 | 0x00 | fifo_byte_counter_7_0 | | | | | |
| 0x23 | - | - | reserved | | | | | |
| 0x22 | TEMPER ATURE | 0x00 | temperature | | | | | |
| 0x21 | STEP_C OUNTER _3 | 0x00 | step_counter_out_3 | | | | | |
| 0x20 | STEP_C OUNTER _2 | 0x00 | step_counter_out_2 | | | | | |
| 0x1F | STEP_C OUNTER _1 | 0x00 | step_counter_out_1 | | | | | |



| | | | | | | | | | | |
|------|--------------------------------|------|--------------------|----------------------|--------------|------------|----------------|-------------------|------------------|--------------|
| 0x1E | STEP COUNTER_0 | 0x00 | step_counter_out_0 | | | | | | | |
| 0x1D | INT STATUS_1 | 0x00 | acc_drdy_int | reserved | aux_drdy_int | reserved | | | fwm_int | ffull_int |
| 0x1C | INT STATUS_0 | 0x00 | error_int_out | any_no_motion_output | wakeup_out | reserved | wrist_tilt_out | activity_type_out | step_counter_out | reserved |
| 0x1B | EVENT | 0x01 | reserved | | | | | | | por_detected |
| 0x1A | SENSOR_TIME_2 | 0x00 | sensor_time_23_16 | | | | | | | |
| 0x19 | SENSOR_TIME_1 | 0x00 | sensor_time_15_8 | | | | | | | |
| 0x18 | SENSOR_TIME_0 | 0x00 | sensor_time_7_0 | | | | | | | |
| 0x17 | DATA_13 | 0x00 | acc_z_15_8 | | | | | | | |
| 0x16 | DATA_12 | 0x00 | acc_z_7_0 | | | | | | | |
| 0x15 | DATA_11 | 0x00 | acc_y_15_8 | | | | | | | |
| 0x14 | DATA_10 | 0x00 | acc_y_7_0 | | | | | | | |
| 0x13 | DATA_9 | 0x00 | acc_x_15_8 | | | | | | | |
| 0x12 | DATA_8 | 0x00 | acc_x_7_0 | | | | | | | |
| 0x11 | DATA_7 | 0x00 | aux_r_15_8 | | | | | | | |
| 0x10 | DATA_6 | 0x00 | aux_r_7_0 | | | | | | | |
| 0x0F | DATA_5 | 0x00 | aux_z_15_8 | | | | | | | |
| 0x0E | DATA_4 | 0x00 | aux_z_7_0 | | | | | | | |
| 0x0D | DATA_3 | 0x00 | aux_y_15_8 | | | | | | | |
| 0x0C | DATA_2 | 0x00 | aux_y_7_0 | | | | | | | |
| 0x0B | DATA_1 | 0x00 | aux_x_15_8 | | | | | | | |
| 0x0A | DATA_0 | 0x00 | aux_x_7_0 | | | | | | | |
| 0x09 | - | - | reserved | | | | | | | |
| ... | - | - | reserved | | | | | | | |
| 0x04 | - | - | reserved | | | | | | | |
| 0x03 | STATUS | 0x10 | drdy_acc | reserved | drdy_aux | cmd_rdy | reserved | aux_man_op | reserved | |
| 0x02 | ERR_REGISTER | 0x00 | aux_err | fifo_err | reserved | error_code | | | cmd_err | fatal_err |
| 0x01 | - | - | reserved | | | | | | | |
| 0x00 | CHIP_ID | 0x16 | chip_id | | | | | | | |

**FEATURES IN**

| Register Address | Register Name | Default Value | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|------------------|--|---------------|-----------------|---|-----------------|---------------|-------------|-----------------|-----------------|--------|-----------------|
| 0x5E: 0x3F | general settings axes remapping[1] | 0x00 | reserved | | | | | | | | map_z_axis_sign |
| 0x5E: 0x3E | general settings axes remapping[0] | 0x88 | map_z_axis | | map_y_axis_sign | map_y_axis | | map_x_axis_sign | map_x_axis | | |
| 0x5E: 0x3D | general settings config id[1] | 0x00 | identification | | | | | | | | |
| 0x5E: 0x3C | general settings config id[0] | 0x00 | identification | | | | | | | | |
| 0x5E: 0x3B | wrist tilt settings[1] | 0x00 | reserved | | | | | | | | |
| 0x5E: 0x3A | wrist tilt settings[0] | 0x00 | reserved | | | | | | | | enable |
| 0x5E: 0x39 | tap double tap settings[1] | 0x00 | reserved | | | | | | | | |
| 0x5E: 0x38 | tap double tap settings[0] | 0x06 | reserved | | | single_tap_en | sensitivity | | | enable | |
| 0x5E: 0x37 | step counter settings 26[1] | 0x00 | reserved | | en_activity | en_counter | en_detector | reset_counter | watermark_level | | |
| 0x5E: 0x36 | step counter settings 26[0] | 0x00 | watermark_level | | | | | | | | |
| 0x5E: 0x35 | step counter settings 25[1] | 0x00 | param_25 | | | | | | | | |



| | | | |
|---------------|---|------|----------|
| 0x5E: 0x34 | step counter settings 25 1 | 0x0E | param_25 |
| 0x5E: 0x33 | step counter settings 24 1 | 0x00 | param_24 |
| 0x5E: 0x32 | step counter settings 24 1 | 0x01 | param_24 |
| 0x5E: 0x31 | step counter settings 23 1 | 0x00 | param_23 |
| 0x5E: 0x30 | step counter settings 23 1 | 0x03 | param_23 |
| 0x5E: 0x2F | step counter settings 22 1 | 0x00 | param_22 |
| 0x5E: 0x2E | step counter settings 22 1 | 0x01 | param_22 |
| 0x5E: 0x2D | step counter settings 21 1 | 0x01 | param_21 |
| 0x5E: 0x2C | step counter settings 21 1 | 0x00 | param_21 |
| 0x5E: 0x2B | step counter settings 20 1 | 0x3C | param_20 |
| 0x5E: 0x2A | step counter settings 20 1 | 0xF0 | param_20 |



| | | | |
|---------------|---|------|----------|
| 0x5E: 0x29 | step counter settings 19 1 | 0x00 | param_19 |
| 0x5E: 0x28 | step counter settings 19 0 | 0x0C | param_19 |
| 0x5E: 0x27 | step counter settings 18 1 | 0x00 | param_18 |
| 0x5E: 0x26 | step counter settings 18 0 | 0x01 | param_18 |
| 0x5E: 0x25 | step counter settings 17 1 | 0x00 | param_17 |
| 0x5E: 0x24 | step counter settings 17 0 | 0xA0 | param_17 |
| 0x5E: 0x23 | step counter settings 16 1 | 0x00 | param_16 |
| 0x5E: 0x22 | step counter settings 16 0 | 0x96 | param_16 |
| 0x5E: 0x21 | step counter settings 15 1 | 0x00 | param_15 |
| 0x5E: 0x20 | step counter settings 15 0 | 0x19 | param_15 |
| 0x5E: 0x1F | step counter settings 14 1 | 0x00 | param_14 |



| | | | |
|---------------|--|------|----------|
| 0x5E: 0x1E | step counter settings 14[0] ↓ | 0x27 | param_14 |
| 0x5E: 0x1D | step counter settings 13[1] ↓ | 0x00 | param_13 |
| 0x5E: 0x1C | step counter settings 13[0] ↓ | 0x01 | param_13 |
| 0x5E: 0x1B | step counter settings 12[1] ↓ | 0x46 | param_12 |
| 0x5E: 0x1A | step counter settings 12[0] ↓ | 0x0C | param_12 |
| 0x5E: 0x19 | step counter settings 11[1] ↓ | 0xE6 | param_11 |
| 0x5E: 0x18 | step counter settings 11[0] ↓ | 0xEC | param_11 |
| 0x5E: 0x17 | step counter settings 10[1] ↓ | 0x04 | param_10 |
| 0x5E: 0x16 | step counter settings 10[0] ↓ | 0xC3 | param_10 |
| 0x5E: 0x15 | step counter settings 9[1] ↓ | 0x09 | param_9 |
| 0x5E: 0x14 | step counter settings 9[0] ↓ | 0x85 | param_9 |
| 0x5E: 0x13 | step counter settings 8[1] ↓ | 0x04 | param_8 |



| | | | |
|---------------|--|------|---------|
| 0x5E: 0x12 | step_cou nter.setti ngs_8[0] | 0xC3 | param_8 |
| 0x5E: 0x11 | step_cou nter.setti ngs_7[1] | 0x6C | param_7 |
| 0x5E: 0x10 | step_cou nter.setti ngs_7[0] | 0xCD | param_7 |
| 0x5E: 0x0F | step_cou nter.setti ngs_6[1] | 0x7B | param_6 |
| 0x5E: 0x0E | step_cou nter.setti ngs_6[0] | 0x3F | param_6 |
| 0x5E: 0x0D | step_cou nter.setti ngs_5[1] | 0x00 | param_5 |
| 0x5E: 0x0C | step_cou nter.setti ngs_5[0] | 0x04 | param_5 |
| 0x5E: 0x0B | step_cou nter.setti ngs_4[1] | 0x7A | param_4 |
| 0x5E: 0x0A | step_cou nter.setti ngs_4[0] | 0xDB | param_4 |
| 0x5E: 0x09 | step_cou nter.setti ngs_3[1] | 0x01 | param_3 |
| 0x5E: 0x08 | step_cou nter.setti ngs_3[0] | 0x3B | param_3 |
| 0x5E: 0x07 | step_cou nter.setti ngs_2[1] | 0x7B | param_2 |
| 0x5E: 0x06 | step_cou nter.setti ngs_2[0] | 0xD4 | param_2 |
| 0x5E: 0x05 | step_cou nter.setti ngs_1[1] | 0x01 | param_1 |
| 0x5E: 0x04 | step_cou nter.setti ngs_1[0] | 0x2D | param_1 |

| | | | | | | | |
|---------------|---|------|-----------|------|------|------------------|-----------|
| 0x5E: 0x03 | any motion on setting bits 2[1] | 0x00 | z_en | y_en | x_en | duration | |
| 0x5E: 0x02 | any motion on setting bits 2[0] | 0x05 | duration | | | | |
| 0x5E: 0x01 | any motion on setting bits 1[1] | 0x00 | reserved | | | nomotion _sel | threshold |
| 0x5E: 0x00 | any motion on setting bits 1[0] | 0xAA | threshold | | | | |

**Register (0x00) CHIP_ID**

DESCRIPTION: Chip identification code

RESET: 0x16

DEFINITION (Go to [register map](#)):

| Name | Register (0x00) CHIP_ID | | | |
|-------------|-------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 1 |
| Content | chip_id | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 1 | 1 | 0 |
| Content | chip_id | | | |

chip_id: Chip identification code for BMA456.

Register (0x02) ERR_REG

DESCRIPTION: Reports sensor error conditions

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x02) ERR_REG | | | |
|-------------|-------------------------|----------|----------|------------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | n/a | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_err | fifo_err | reserved | error_code |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | error_code | | cmd_err | fatal_err |

fatal_err: Fatal Error, chip is not in operational state (Boot-, power-system). This flag will be reset only by power-on-reset or softreset.

cmd_err: Command execution failed.

error_code: Error codes for persistent errors

| error_code | | |
|------------|----------|----------------------------|
| 0x00 | no_error | no error is reported |
| 0x01 | acc_err | error in Register ACC_CONF |

fifo_err: Error in FIFO detected: Input data was discarded in stream mode. This flag will be reset when read.



aux_err: Error in I2C-Master detected. This flag will be reset when read.

Register (0x03) STATUS

DESCRIPTION: Sensor status flags

RESET: 0x10

DEFINITION (Go to [register map](#)):

| Name | Register (0x03) STATUS | | | |
|-------------|------------------------|------------|----------|---------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | n/a | R | R |
| Reset Value | 0 | 0 | 0 | 1 |
| Content | drdy_acc | reserved | drdy_aux | cmd_rdy |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | n/a | R | n/a | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | aux_man_op | reserved | |

aux_man_op: '1'('0') indicate a (no) manual auxiliary interface operation is ongoing.

cmd_rdy: CMD decoder status. '0' -> Command in progress '1' -> Command decoder is ready to accept a new command

drdy_aux: Data ready for auxiliary sensor. It gets reset when one auxiliary DATA register is read out

drdy_acc: Data ready for accelerometer. It gets reset when one accelerometer DATA register is read out

Register (0x0A) DATA_0

DESCRIPTION: AUX_X(LSB)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x0A) DATA_0 | | | |
|-------------|------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_x_7_0 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_x_7_0 | | | |

**Register (0x0B) DATA_1**

DESCRIPTION: AUX_X(MSB)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x0B) DATA_1 | | | |
|-------------|------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_x_15_8 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_x_15_8 | | | |

Register (0x0C) DATA_2

DESCRIPTION: AUX_Y(LSB)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x0C) DATA_2 | | | |
|-------------|------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_y_7_0 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_y_7_0 | | | |

**Register (0x0D) DATA_3**

DESCRIPTION: AUX_Y(MSB)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x0D) DATA_3 | | | |
|-------------|------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_y_15_8 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_y_15_8 | | | |

Register (0x0E) DATA_4

DESCRIPTION: AUX_Z(LSB)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x0E) DATA_4 | | | |
|-------------|------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_z_7_0 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_z_7_0 | | | |

**Register (0x0F) DATA_5**

DESCRIPTION: AUX_Z(MSB)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x0F) DATA_5 | | | |
|-------------|------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_z_15_8 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_z_15_8 | | | |

Register (0x10) DATA_6

DESCRIPTION: AUX_R(LSB)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x10) DATA_6 | | | |
|-------------|------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_r_7_0 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_r_7_0 | | | |

**Register (0x11) DATA_7**

DESCRIPTION: AUX_R(MSB)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x11) DATA_7 | | | |
|-------------|------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_r_15_8 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | aux_r_15_8 | | | |

Register (0x12) DATA_8

DESCRIPTION: ACC_X(LSB)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x12) DATA_8 | | | |
|-------------|------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | acc_x_7_0 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | acc_x_7_0 | | | |

**Register (0x13) DATA_9**

DESCRIPTION: ACC_X(MSB)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x13) DATA_9 | | | |
|-------------|------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | acc_x_15_8 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | acc_x_15_8 | | | |

Register (0x14) DATA_10

DESCRIPTION: ACC_Y(LSB)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x14) DATA_10 | | | |
|-------------|-------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | acc_y_7_0 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | acc_y_7_0 | | | |

**Register (0x15) DATA_11**

DESCRIPTION: ACC_Y(MSB)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x15) DATA_11 | | | |
|-------------|-------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | acc_y_15_8 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | acc_y_15_8 | | | |

Register (0x16) DATA_12

DESCRIPTION: ACC_Z(LSB)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x16) DATA_12 | | | |
|-------------|-------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | acc_z_7_0 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | acc_z_7_0 | | | |

**Register (0x17) DATA_13**

DESCRIPTION: ACC_Z(MSB)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x17) DATA_13 | | | |
|-------------|-------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | acc_z_15_8 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | acc_z_15_8 | | | |

Register (0x18) SENSORTIME_0

DESCRIPTION: Sensor time <7:0>

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x18) SENSORTIME_0 | | | |
|-------------|------------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | sensor_time_7_0 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | sensor_time_7_0 | | | |

sensor_time_7_0: Sensor time <7:0> in units of 39.0625 us.

**Register (0x19) SENSORTIME_1**

DESCRIPTION: Sensor time <15:8>

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x19) SENSORTIME_1 | | | |
|-------------|------------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | sensor_time_15_8 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | sensor_time_15_8 | | | |

sensor_time_15_8: Sensor time <15:8> in units of 10 ms.

Register (0x1A) SENSORTIME_2

DESCRIPTION: Sensor time <23:16>

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x1A) SENSORTIME_2 | | | |
|-------------|------------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | sensor_time_23_16 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | sensor_time_23_16 | | | |

sensor_time_23_16: Sensor time <23:16> in units of 2.56 s.

Register (0x1B) EVENT

DESCRIPTION: Sensor status flags

RESET: 0x01

DEFINITION (Go to [register map](#)):

| Name | Register (0x1B) EVENT | | | |
|-------------|-----------------------|-----|-----|--------------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | n/a | n/a | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | n/a | n/a | n/a | R |
| Reset Value | 0 | 0 | 0 | 1 |
| Content | reserved | | | por_detected |

por_detected: '1' after device power up or softreset. Clear-on-read

Register (0x1C) INT_STATUS_0

DESCRIPTION: Interrupt/Feature Status. Will be cleared on read.

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x1C) INT_STATUS_0 | | | |
|-------------|------------------------------|-------------------|------------------|----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | error_int_out | any_no_motion_out | wakeup_out | reserved |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | wrist_tilt_out | activity_type_out | step_counter_out | reserved |

step_counter_out: Step-counter watermark or Step-detector output.

activity_type_out: Step counter activity output(Running, Walking, Still)

wrist_tilt_out: Wrist tilt output

wakeup_out: Wakeup output

any_no_motion_out: Any-motion/No-motion detection output

error_int_out: Error interrupt output

Register (0x1D) INT_STATUS_1

DESCRIPTION: Interrupt Status. Will be cleared on read.

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x1D) INT_STATUS_1 | | | |
|-------------|------------------------------|----------|--------------|-----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | n/a | R | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | acc_drdy_int | reserved | aux_drdy_int | reserved |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | n/a | n/a | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | fwm_int | ffull_int |

ffull_int: FIFO Full Interrupt

fwm_int: FIFO Watermark Interrupt

aux_drdy_int: Auxiliary sensor data ready interrupt

acc_drdy_int: Accelerometer data ready interrupt

Register (0x1E) STEP_COUNTER_0

DESCRIPTION: Step counting value byte-0

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x1E) STEP_COUNTER_0 | | | |
|-------------|--------------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | step_counter_out_0 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | step_counter_out_0 | | | |

step_counter_out_0: Step counting value byte-0 (least significant byte)

**Register (0x1F) STEP_COUNTER_1**

DESCRIPTION: Step counting value byte-1

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x1F) STEP_COUNTER_1 | | | |
|-------------|--------------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | step_counter_out_1 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | step_counter_out_1 | | | |

step_counter_out_1: Step counting value byte-1

Register (0x20) STEP_COUNTER_2

DESCRIPTION: Step counting value byte-2

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x20) STEP_COUNTER_2 | | | |
|-------------|--------------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | step_counter_out_2 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | step_counter_out_2 | | | |

step_counter_out_2: Step counting value byte-2

**Register (0x21) STEP_COUNTER_3**

DESCRIPTION: Step counting value byte-3

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x21) STEP_COUNTER_3 | | | |
|-------------|--------------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | step_counter_out_3 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | step_counter_out_3 | | | |

step_counter_out_3: Step counting value byte-3 (most significant byte)

Register (0x22) TEMPERATURE

DESCRIPTION: Contains the temperature value of the sensor

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x22) TEMPERATURE | | | |
|-------------|-----------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | temperature | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | temperature | | | |

temperature: Temperature value in two's complement representation in units of 1 Kelvin: 0x00 corresponds to 23 degree Celsius.

**Register (0x24) FIFO_LENGTH_0**

DESCRIPTION: FIFO byte count register (LSB)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x24) FIFO_LENGTH_0 | | | |
|-------------|-------------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | fifo_byte_counter_7_0 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | fifo_byte_counter_7_0 | | | |

fifo_byte_counter_7_0: Current fill level of FIFO buffer.

Register (0x25) FIFO_LENGTH_1

DESCRIPTION: FIFO byte count register (MSB)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x25) FIFO_LENGTH_1 | | | |
|-------------|-------------------------------|-----|------------------------|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | n/a | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | fifo_byte_counter_13_8 | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | fifo_byte_counter_13_8 | | | |

fifo_byte_counter_13_8: FIFO byte counter bits 13..8

**Register (0x26) FIFO_DATA**

DESCRIPTION: FIFO data output register

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x26) FIFO_DATA | | | |
|-------------|---------------------------|---|---|---|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | fifo_data | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | fifo_data | | | |

fifo_data: FIFO read data.

Register (0x27) ACTIVITY_TYPE

DESCRIPTION: Step counter activity output(Running, Walking, Still)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x27) ACTIVITY_TYPE | | | |
|-------------|-------------------------------|-----|-------------------|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | n/a | n/a | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | n/a | n/a | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | activity_type_out | |

activity_type_out: Step counter activity output(Running, Walking, Still)

| activity_type_out | | |
|-------------------|---------|-----------------|
| 0x00 | still | user not moving |
| 0x01 | walking | user walking |
| 0x02 | running | user running |
| 0x03 | unknown | unknown state |

**Register (0x2A) INTERNAL_STATUS**

DESCRIPTION: Error bits and message indicating internal status

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x2A) INTERNAL_STATUS | | | |
|-------------|---------------------------------|----------------|------------------|---------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | odr_high_error | odr_50hz_error | axes_remap_error | message |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | R | R | R | R |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | message | | | |

message: Internal Status Message

| message | | |
|---------|----------|-------------------------|
| 0x00 | not_init | ASIC is not initialized |
| 0x01 | init_ok | ASIC initialized |
| 0x02 | init_err | Initialization error |
| 0x03 | drv_err | Invalid driver |
| 0x04 | sns_stop | Sensor stopped |

axes_remap_error: Axes remapped wrongly because a source axis is not assigned to more than one target axis.

odr_50hz_error: The minimum bandwidth conditions are not respected for the features which require 50 Hz data.

odr_high_error: The minimum bandwidth conditions are not respected for the Wakeup Detection.

**Register (0x40) ACC_CONF**

DESCRIPTION: Sets the output data rate, the bandwidth, and the read mode of the acceleration sensor

RESET: 0xA8

DEFINITION (Go to [register map](#)):

| Name | Register (0x40) ACC_CONF | | | |
|-------------|--------------------------|----|----|----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 1 | 0 | 1 | 0 |
| Content | acc_perf_mode | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 1 | 0 | 0 | 0 |
| Content | acc_odr | | | |

acc_odr: ODR in Hz. The output data rate is independent of the power mode setting for the sensor, but not all settings are supported in all power modes.

| acc_odr | | |
|---------|----------|----------|
| 0x00 | reserved | Reserved |
| 0x01 | odr_0p78 | 25/32 |
| 0x02 | odr_1p5 | 25/16 |
| 0x03 | odr_3p1 | 25/8 |
| 0x04 | odr_6p25 | 25/4 |
| 0x05 | odr_12p5 | 25/2 |
| 0x06 | odr_25 | 25 |
| 0x07 | odr_50 | 50 |
| 0x08 | odr_100 | 100 |
| 0x09 | odr_200 | 200 |
| 0x0a | odr_400 | 400 |
| 0x0b | odr_800 | 800 |
| 0x0c | odr_1k6 | 1600 |
| 0x0d | odr_3k2 | Reserved |
| 0x0e | odr_6k4 | Reserved |
| 0x0f | odr_12k8 | Reserved |

acc_bwp: Bandwidth parameter, determines filter configuration (acc_perf_mode=1) and averaging for undersampling mode (acc_perf_mode=0)

| acc_bwp | | |
|---------|-----------|--|
| 0x00 | osr4_avg1 | acc_perf_mode = 1 -> OSR4 mode; acc_perf_mode = 0 -> no averaging |
| 0x01 | osr2_avg2 | acc_perf_mode = 1 -> OSR2 mode; acc_perf_mode = 0 -> average 2 samples |



| | | |
|------|------------|--|
| 0x02 | norm_avg4 | acc_perf_mode = 1 -> normal mode; acc_perf_mode = 0 -> average 4 samples |
| 0x03 | cic_avg8 | acc_perf_mode = 1 -> Reserved; acc_perf_mode = 0 -> average 8 samples |
| 0x04 | res_avg16 | acc_perf_mode = 1 -> Reserved; acc_perf_mode = 0 -> average 16 samples |
| 0x05 | res_avg32 | acc_perf_mode = 1 -> Reserved; acc_perf_mode = 0 -> average 32 samples |
| 0x06 | res_avg64 | acc_perf_mode = 1 -> Reserved; acc_perf_mode = 0 -> average 64 samples |
| 0x07 | res_avg128 | acc_perf_mode = 1 -> Reserved; acc_perf_mode = 0 -> average 128 samples |

acc_perf_mode: Select accelerometer filter performance mode:

| acc_perf_mode | | |
|---------------|---------|-----------------------------|
| 0x00 | cic_avg | averaging mode. |
| 0x01 | cont | continuous filter function. |

Register (0x41) ACC_RANGE

DESCRIPTION: Selection of the Accelerometer g-range

RESET: 0x01

DEFINITION (Go to [register map](#)):

| Name | Register (0x41) ACC_RANGE | | | |
|-------------|---------------------------|-----|-----------|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | n/a | n/a | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | n/a | n/a | RW | RW |
| Reset Value | 0 | 0 | 0 | 1 |
| Content | reserved | | acc_range | |

acc_range: Accelerometer g-range

| acc_range | | |
|-----------|-----------|--------|
| 0x00 | range_2g | +/-2g |
| 0x01 | range_4g | +/-4g |
| 0x02 | range_8g | +/-8g |
| 0x03 | range_16g | +/-16g |

**Register (0x44) AUX_CONF**

DESCRIPTION: Sets the output data rate of the Auxiliary interface

RESET: 0x46

DEFINITION (Go to [register map](#)):

| Name | Register (0x44) AUX_CONF | | | |
|-------------|--------------------------|----|----|----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 1 | 0 | 0 |
| Content | aux_offset | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 1 | 1 | 0 |
| Content | aux_odr | | | |

aux_odr: Select the poll rate for the sensor attached to the Auxiliary interface.

| aux_odr | | |
|---------|----------|----------|
| 0x00 | reserved | Reserved |
| 0x01 | odr_0p78 | 25/32 |
| 0x02 | odr_1p5 | 25/16 |
| 0x03 | odr_3p1 | 25/8 |
| 0x04 | odr_6p25 | 25/4 |
| 0x05 | odr_12p5 | 25/2 |
| 0x06 | odr_25 | 25 |
| 0x07 | odr_50 | 50 |
| 0x08 | odr_100 | 100 |
| 0x09 | odr_200 | 200 |
| 0x0a | odr_400 | 400 |
| 0x0b | odr_800 | 800 |
| 0x0c | odr_1k6 | Reserved |
| 0x0d | odr_3k2 | Reserved |
| 0x0e | odr_6k4 | Reserved |
| 0x0f | odr_12k8 | Reserved |

aux_offset: trigger-readout offset in units of 2.5 ms. If set to zero, the offset is maximum, i.e. after readout a trigger is issued immediately.

Register (0x45) FIFO_DOWNS

DESCRIPTION: Configure Accelerometer downsampling rates for FIFO

RESET: 0x80

DEFINITION (Go to [register map](#)):



| Name | Register (0x45) FIFO_DOWNS | | | |
|-------------|----------------------------|----------------|-----|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 1 | 0 | 0 | 0 |
| Content | acc_fifo_filt_data | acc_fifo_downs | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | n/a | n/a | n/a | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | |

acc_fifo_downs: Downsampling for accelerometer data ($2^{\text{acc_fifo_downs}}$)

acc_fifo_filt_data: selects filtered or unfiltered Accelerometer data for fifo

| acc_fifo_filt_data | | |
|--------------------|------------|-----------------|
| 0x00 | unfiltered | Unfiltered data |
| 0x01 | filtered | Filtered data |

Register (0x46) FIFO_WTM_0

DESCRIPTION: FIFO Watermark level LSB

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x46) FIFO_WTM_0 | | | |
|-------------|----------------------------|----|----|----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | fifo_water_mark_7_0 | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | fifo_water_mark_7_0 | | | |

**Register (0x47) FIFO_WTM_1**

DESCRIPTION: FIFO Watermark level MSB

RESET: 0x02

DEFINITION (Go to [register map](#)):

| Name | Register (0x47) FIFO_WTM_1 | | | |
|-------------|----------------------------|-----|-----|----------------------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | n/a | n/a | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | fifo_water_mark_12_8 |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 1 | 0 |
| Content | fifo_water_mark_12_8 | | | |

Register (0x48) FIFO_CONFIG_0

DESCRIPTION: FIFO frame content configuration

RESET: 0x02

DEFINITION (Go to [register map](#)):

| Name | Register (0x48) FIFO_CONFIG_0 | | | |
|-------------|-------------------------------|-----|--------------|-------------------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | n/a | n/a | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | n/a | n/a | RW | RW |
| Reset Value | 0 | 0 | 1 | 0 |
| Content | reserved | | fifo_time_en | fifo_stop_on_full |

fifo_stop_on_full: Stop writing samples into FIFO when FIFO is full.

| fifo_stop_on_full | | |
|-------------------|---------|---------------------------------------|
| 0x00 | disable | do not stop writing to FIFO when full |
| 0x01 | enable | Stop writing into FIFO when full. |

fifo_time_en: Return sensortime frame after the last valid data frame.

| fifo_time_en | | |
|--------------|---------|--------------------------------|
| 0x00 | disable | do not return sensortime frame |
| 0x01 | enable | return sensortime frame |

**Register (0x49) FIFO_CONFIG_1**

DESCRIPTION: FIFO frame content configuration

RESET: 0x10

DEFINITION (Go to [register map](#)):

| Name | Register (0x49) FIFO_CONFIG_1 | | | |
|-------------|-------------------------------|------------------|-------------|----------------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 1 |
| Content | reserved | fifo_acc_en | fifo_aux_en | fifo_header_en |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | n/a | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | fifo_tag_int1_en | fifo_tag_int2_en | reserved | |

fifo_tag_int2_en: FIFO interrupt 2 tag enable

| fifo_tag_int2_en | | |
|-------------------------|---------|-------------|
| 0x00 | disable | disable tag |
| 0x01 | enable | enable tag |

fifo_tag_int1_en: FIFO interrupt 1 tag enable

| fifo_tag_int1_en | | |
|-------------------------|---------|-------------|
| 0x00 | disable | disable tag |
| 0x01 | enable | enable tag |

fifo_header_en: FIFO frame header enable

| fifo_header_en | | |
|-----------------------|---------|--|
| 0x00 | disable | no header is stored (output data rate of all enabled sensors need to be identical) |
| 0x01 | enable | header is stored |

fifo_aux_en: Store Auxiliary data in FIFO (all 3 axes)

| fifo_aux_en | | |
|--------------------|---------|-----------------------------|
| 0x00 | disable | no Auxiliary data is stored |
| 0x01 | enable | Auxiliary data is stored |

fifo_acc_en: Store Accelerometer data in FIFO (all 3 axes)

| fifo_acc_en | | |
|--------------------|---------|---------------------------------|
| 0x00 | disable | no Accelerometer data is stored |
| 0x01 | enable | Accelerometer data is stored |

**Register (0x4B) AUX_DEV_ID**

DESCRIPTION: Auxiliary interface slave device id

RESET: 0x20

DEFINITION (Go to [register map](#)):

| Name | Register (0x4B) AUX_DEV_ID | | | |
|-------------|----------------------------|----|----|----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 1 | 0 |
| Content | i2c_device_addr | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | i2c_device_addr | | | reserved |

i2c_device_addr: I2C device address of Auxiliary slave

Register (0x4C) AUX_IF_CONF

DESCRIPTION: Auxiliary interface configuration

RESET: 0x83

DEFINITION (Go to [register map](#)):

| Name | Register (0x4C) AUX_IF_CONF | | | |
|-------------|-----------------------------|----------|--------------|-----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | n/a | n/a | n/a |
| Reset Value | 1 | 0 | 0 | 0 |
| Content | aux_manual_en | reserved | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | n/a | n/a | RW | RW |
| Reset Value | 0 | 0 | 1 | 1 |
| Content | reserved | | aux_rd_burst | |

aux_rd_burst: Burst data length (1,2,6,8 byte)

| aux_rd_burst | | |
|--------------|-----|----------------|
| 0x00 | BL1 | Burst length 1 |
| 0x01 | BL2 | Burst length 2 |
| 0x02 | BL6 | Burst length 6 |
| 0x03 | BL8 | Burst length 8 |

aux_manual_en: Enable auxiliary interface manual mode.

| aux_manual_en | | |
|---------------|---------|------------|
| 0x00 | disable | Data mode |
| 0x01 | enable | Setup mode |

**Register (0x4D) AUX_RD_ADDR**

DESCRIPTION: Auxiliary interface read register address

RESET: 0x42

DEFINITION (Go to [register map](#)):

| Name | Register (0x4D) AUX_RD_ADDR | | | |
|-------------|-----------------------------|----|----|----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 1 | 0 | 0 |
| Content | read_addr | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 1 | 0 |
| Content | read_addr | | | |

read_addr: Address to read

Register (0x4E) AUX_WR_ADDR

DESCRIPTION: Auxiliary interface write register address

RESET: 0x4C

DEFINITION (Go to [register map](#)):

| Name | Register (0x4E) AUX_WR_ADDR | | | |
|-------------|-----------------------------|----|----|----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 1 | 0 | 0 |
| Content | write_addr | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 1 | 1 | 0 | 0 |
| Content | write_addr | | | |

write_addr: Address to write

**Register (0x4F) AUX_WR_DATA**

DESCRIPTION: Auxiliary interface write data

RESET: 0x02

DEFINITION (Go to [register map](#)):

| Name | Register (0x4F) AUX_WR_DATA | | | |
|-------------|-----------------------------|----|----|----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | write_data | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 1 | 0 |
| Content | write_data | | | |

write_data: Data to write

Register (0x53) INT1_IO_CTRL

DESCRIPTION: Configure the electrical behaviour of the interrupt pins

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x53) INT1_IO_CTRL | | | |
|-------------|------------------------------|-----|-----|-----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | n/a | n/a | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | input_en |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | output_en | od | lv | edge_ctrl |

edge_ctrl: Configure trigger condition of INT1 pin (input)

| edge_ctrl | | |
|-----------|----------|-------|
| 0x00 | level_tr | Level |
| 0x01 | edge_tr | Edge |

lv: Configure level of INT1 pin

| lv | | |
|------|-------------|-------------|
| 0x00 | active_low | active low |
| 0x01 | active_high | active high |



od: Configure behaviour of INT1 pin to open drain.

| od | | |
|------|------------|------------|
| 0x00 | push_pull | push-pull |
| 0x01 | open_drain | open drain |

output_en: Output enable for INT1 pin

| output_en | | |
|-----------|-----|-----------------|
| 0x00 | off | Output disabled |
| 0x01 | on | Output enabled |

input_en: Input enable for INT1 pin

| input_en | | |
|----------|-----|----------------|
| 0x00 | off | Input disabled |
| 0x01 | on | Input enabled |

Register (0x54) INT2_IO_CTRL

DESCRIPTION: Configure the electrical behaviour of the interrupt pins

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x54) INT2_IO_CTRL | | | |
|-------------|------------------------------|-----|-----|-----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | n/a | n/a | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | input_en |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | output_en | od | lvl | edge_ctrl |

edge_ctrl: Configure trigger condition of INT2 pin (input)

| edge_ctrl | | |
|-----------|----------|-------|
| 0x00 | level_tr | Level |
| 0x01 | edge_tr | Edge |

lvl: Configure level of INT2 pin

| lvl | | |
|------|-------------|-------------|
| 0x00 | active_low | active low |
| 0x01 | active_high | active high |



od: Configure behaviour of INT2 pin to open drain.

| od | | |
|------|------------|------------|
| 0x00 | push_pull | push-pull |
| 0x01 | open_drain | open drain |

output_en: Output enable for INT2 pin

| output_en | | |
|-----------|-----|-----------------|
| 0x00 | off | Output disabled |
| 0x01 | on | Output enabled |

input_en: Input enable for INT2 pin

| input_en | | |
|----------|-----|----------------|
| 0x00 | off | Input disabled |
| 0x01 | on | Input enabled |

Register (0x55) INT_LATCH

DESCRIPTION: Configure interrupt modes

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x55) INT_LATCH | | | |
|-------------|---------------------------|-----|-----|-----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | n/a | n/a | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | n/a | n/a | n/a | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | int_latch |

int_latch: Latched/non-latched/temporary interrupt modes

| int_latch | | |
|-----------|-----------|-------------|
| 0x00 | none | non latched |
| 0x01 | permanent | latched |

Register (0x56) INT1_MAP

DESCRIPTION: Interrupt/Feature mapping on INT1

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x56) INT1_MAP | | | |
|-------------|--------------------------|-------------------|------------------|----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | RW | RW | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | error_int_out | any_no_motion_out | wakeup_out | reserved |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | wrist_tilt_out | activity_type_out | step_counter_out | reserved |

step_counter_out: Step-counter watermark or Step-detector output.

activity_type_out: Step counter activity output(Running, Walking, Still)

wrist_tilt_out: Wrist tilt output

wakeup_out: Wakeup output

any_no_motion_out: Any-motion/No-motion detection output

error_int_out: Error interrupt output

Register (0x57) INT2_MAP

DESCRIPTION: Interrupt/Feature mapping on INT2

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x57) INT2_MAP | | | |
|-------------|--------------------------|-------------------|------------------|----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | RW | RW | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | error_int_out | any_no_motion_out | wakeup_out | reserved |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | wrist_tilt_out | activity_type_out | step_counter_out | reserved |

step_counter_out: Step-counter watermark or Step-detector output.

activity_type_out: Step counter activity output(Running, Walking, Still)

wrist_tilt_out: Wrist tilt output

wakeup_out: Wakeup output



any_no_motion_out: Any-motion/No-motion detection output

error_int_out: Error interrupt output

Register (0x58) INT_MAP_DATA

DESCRIPTION: Interrupt mapping hardware interrupts

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x58) INT_MAP_DATA | | | |
|-------------|------------------------------|-----------|----------|------------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | int2_drdy | int2_fwm | int2_ffull |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | n/a | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | int1_drdy | int1_fwm | int1_ffull |

int1_ffull: FIFO Full interrupt mapped to INT1

int1_fwm: FIFO Watermark interrupt mapped to INT1

int1_drdy: Data Ready interrupt mapped to INT1

int2_ffull: FIFO Full interrupt mapped to INT2

int2_fwm: FIFO Watermark interrupt mapped to INT2

int2_drdy: Data Ready interrupt mapped to INT2

Register (0x59) INIT_CTRL

DESCRIPTION: Start initialization

RESET: 0x90

DEFINITION (Go to [register map](#)):

| Name | Register (0x59) INIT_CTRL | | | |
|-------------|---------------------------|----|----|----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 1 | 0 | 0 | 1 |
| Content | init_ctrl | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | init_ctrl | | | |

init_ctrl: Start initialization

**Register (0x5E) FEATURES_IN**

DESCRIPTION: Feature configuration read/write port

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x5E) FEATURES_IN | | | |
|-------------|-----------------------------|----|----|----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | features_in | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | features_in | | | |

features_in: Feature configuration read/write data

| Address | Bit | Name | Description | Reset | Access |
|---------------|--------|--------------|---|--------|--------|
| any_motion | | | | | |
| 0x5E: 0x00 | | settings_1 | Any-motion / No-motion detection general configuration flags - part 1 | 0x00AA | |
| | 10...0 | threshold | Slope threshold value for Any-motion / No-motion detection in 5.11g format. Range is 0 to 1g. Default value is 0xAA = 83mg. | 0xAA | RW |
| | 11 | nomotion_sel | Indicates if Nomotion (1) or Any-motion (0) is selected; default value is 0 – Any-motion. | 0x0 | RW |
| | | | | | |
| 0x5E: 0x02 | | settings_2 | Any-motion / No-motion detection general configuration flags - part 2 | 0x0005 | |
| | 12...0 | duration | Defines the number of consecutive data points for which the threshold condition must be respected, for interrupt assertion. It is expressed in 50 Hz samples (20 ms). Range is 0 to 163sec. Default value is 5=100ms. | 0x5 | RW |
| | 13 | x_en | Enables the feature on a per-axis basis | 0x0 | RW |
| | 14 | y_en | Enables the feature on a per-axis basis | 0x0 | RW |
| | 15 | z_en | Enables the feature on a per-axis basis | 0x0 | RW |
| | | | | | |
| step_counter | | | | | |
| | | settings_1 | Step Counter setting | 0x012D | |



| | | | | | |
|---------------|--------|-------------|-----------------------|--------|----|
| 0x5E: 0x04 | 15...0 | param_1 | Step Counter param 1 | 0x12D | RW |
| 0x5E: 0x06 | | settings_2 | Step Counter setting | 0x7BD4 | |
| | 15...0 | param_2 | Step Counter param 2 | 0x7BD4 | RW |
| 0x5E: 0x08 | | settings_3 | Step Counter setting | 0x013B | |
| | 15...0 | param_3 | Step Counter param 3 | 0x13B | RW |
| 0x5E: 0x0A | | settings_4 | Step Counter setting | 0x7ADB | |
| | 15...0 | param_4 | Step Counter param 4 | 0x7ADB | RW |
| 0x5E: 0x0C | | settings_5 | Step Counter setting | 0x0004 | |
| | 15...0 | param_5 | Step Counter param 5 | 0x4 | RW |
| 0x5E: 0x0E | | settings_6 | Step Counter setting | 0x7B3F | |
| | 15...0 | param_6 | Step Counter param 6 | 0x7B3F | RW |
| 0x5E: 0x10 | | settings_7 | Step Counter setting | 0x6CCD | |
| | 15...0 | param_7 | Step Counter param 7 | 0x6CCD | RW |
| 0x5E: 0x12 | | settings_8 | Step Counter setting | 0x04C3 | |
| | 15...0 | param_8 | Step Counter param 8 | 0x4C3 | RW |
| 0x5E: 0x14 | | settings_9 | Step Counter setting | 0x0985 | |
| | 15...0 | param_9 | Step Counter param 9 | 0x985 | RW |
| 0x5E: 0x16 | | settings_10 | Step Counter setting | 0x04C3 | |
| | 15...0 | param_10 | Step Counter param 10 | 0x4C3 | RW |
| 0x5E: 0x18 | | settings_11 | Step Counter setting | 0xE6EC | |
| | 15...0 | param_11 | Step Counter param 11 | 0xE6EC | RW |
| 0x5E: 0x1A | | settings_12 | Step Counter setting | 0x460C | |
| | 15...0 | param_12 | Step Counter param 12 | 0x460C | RW |
| 0x5E: 0x1C | | settings_13 | Step Counter setting | 0x0001 | |
| | 15...0 | param_13 | Step Counter param 13 | 0x1 | RW |
| 0x5E: 0x1E | | settings_14 | Step Counter setting | 0x0027 | |
| | 15...0 | param_14 | Step Counter param 14 | 0x27 | RW |
| 0x5E: 0x20 | | settings_15 | Step Counter setting | 0x0019 | |
| | 15...0 | param_15 | Step Counter param 15 | 0x19 | RW |
| 0x5E: 0x22 | | settings_16 | Step Counter setting | 0x0096 | |
| | 15...0 | param_16 | Step Counter param 16 | 0x96 | RW |
| 0x5E: 0x24 | | settings_17 | Step Counter setting | 0x00A0 | |
| | 15...0 | param_17 | Step Counter param 17 | 0xA0 | RW |
| 0x5E: 0x26 | | settings_18 | Step Counter setting | 0x0001 | |
| | 15...0 | param_18 | Step Counter param 18 | 0x1 | RW |
| 0x5E: 0x28 | | settings_19 | Step Counter setting | 0x000C | |
| | 15...0 | param_19 | Step Counter param 19 | 0xC | RW |
| 0x5E: 0x2A | | settings_20 | Step Counter setting | 0x3CF0 | |
| | 15...0 | param_20 | Step Counter param 20 | 0x3CF0 | RW |
| 0x5E: 0x2C | | settings_21 | Step Counter setting | 0x0100 | |
| | 15...0 | param_21 | Step Counter param 21 | 0x100 | RW |

| | | | | | |
|------------------|--------|-----------------|--|--------|----|
| 0x5E: | | settings_22 | Step Counter setting | 0x0001 | |
| 0x2E | 15...0 | param_22 | Step Counter param 22 | 0x1 | RW |
| 0x5E: | | settings_23 | Step Counter setting | 0x0003 | |
| 0x30 | 15...0 | param_23 | Step Counter param 23 | 0x3 | RW |
| 0x5E: | | settings_24 | Step Counter setting | 0x0001 | |
| 0x32 | 15...0 | param_24 | Step Counter param 24 | 0x1 | RW |
| 0x5E: | | settings_25 | Step Counter setting | 0x000E | |
| 0x34 | 15...0 | param_25 | Step Counter param 25 | 0xE | RW |
| 0x5E: | | settings_26 | Step Counter and Step Detector Settings | 0x0000 | |
| 0x36 | 9...0 | watermark_level | Watermark level; the Step-counter will trigger output every time this number of steps are counted. Holds implicitly a 20x factor, so the range is 0 to 20460, with resolution of 20 steps. If 0, the output is disabled. | 0x0 | RW |
| | 10 | reset_counter | Flag to reset the counted steps. This is only interpreted if the step counter is enabled. | 0x0 | RW |
| | 11 | en_detector | Enables the Step Detector. | 0x0 | RW |
| | 12 | en_counter | Enables the Step Counter. | 0x0 | RW |
| | 13 | en_activity | Enables the activity detection(Running, Walking, Still) | 0x0 | RW |
| | | | | | |
| tap_doubletap | | | | | |
| 0x5E: | | settings | Tap general configuration flags | 0x0006 | |
| 0x38 | 0 | enable | Enables the feature | 0x0 | RW |
| | 3...1 | sensitivity | Configures Tap sensitivity, the range goes from 0 (high sensitive) to 7 (low sensitive). | 0x3 | RW |
| | 4 | single_tap_en | Flag for enabling single tap detection (and disabling double tap). By default double tap detection is being enabled. | 0x0 | RW |
| | | | | | |
| wrist_tilt | | | | | |
| 0x5E: | | settings | Wrist tilt configuration flags | 0x0000 | |
| 0x3A | 0 | enable | Enables the feature | 0x0 | RW |
| | | | | | |
| general_settings | | | | | |
| 0x5E: | | config_id | Describes configuration identification code | 0x0000 | |
| 0x3C | 15...0 | identification | Describes configuration identification code | 0x0 | R |



| 0x5E: 0x3E | | axes_remapping | Describes axes remapping | | | 0x0088 | | | | | | | | | | | | | | | | |
|---------------|-----------------|--|---|--|-------|--------|-------------|-------------|--------------|---|---|----------|-----------------------------------|-----------------------------------|--------|---------------|---------------|----------|----------|----------|-----|----|
| | 1...0 | map_x_axis | Map the x axis to desired axis. <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0x00</td><td>x axis</td><td>Map to x-axis</td></tr><tr><td>0x01</td><td>y axis</td><td>Map to y-axis</td></tr><tr><td>0x02</td><td>z axis</td><td>Map to z-axis</td></tr><tr><td>0x03</td><td>reserved</td><td>reserved</td></tr></table> | | | Value | Name | Description | 0x00 | x axis | Map to x-axis | 0x01 | y axis | Map to y-axis | 0x02 | z axis | Map to z-axis | 0x03 | reserved | reserved | 0x0 | RW |
| | Value | Name | Description | | | | | | | | | | | | | | | | | | | |
| | 0x00 | x axis | Map to x-axis | | | | | | | | | | | | | | | | | | | |
| | 0x01 | y axis | Map to y-axis | | | | | | | | | | | | | | | | | | | |
| | 0x02 | z axis | Map to z-axis | | | | | | | | | | | | | | | | | | | |
| | 0x03 | reserved | reserved | | | | | | | | | | | | | | | | | | | |
| | 2 | map_x_axis_sign | Map the x axis sign to the desired one. <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0x00</td><td>non-inverted</td><td>Clear this bit to non invert the x axis</td></tr><tr><td>0x01</td><td>inverted</td><td>Set this bit to invert the x axis</td></tr></table> | | | Value | Name | Description | 0x00 | non-inverted | Clear this bit to non invert the x axis | 0x01 | inverted | Set this bit to invert the x axis | 0x0 | RW | | | | | | |
| Value | Name | Description | | | | | | | | | | | | | | | | | | | | |
| 0x00 | non-inverted | Clear this bit to non invert the x axis | | | | | | | | | | | | | | | | | | | | |
| 0x01 | inverted | Set this bit to invert the x axis | | | | | | | | | | | | | | | | | | | | |
| 4...3 | map_y_axis | Map the y axis to desired axis <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0x00</td><td>x axis</td><td>Map to x-axis</td></tr><tr><td>0x01</td><td>y axis</td><td>Map to y-axis</td></tr><tr><td>0x02</td><td>z axis</td><td>Map to z-axis</td></tr><tr><td>0x03</td><td>reserved</td><td>reserved</td></tr></table> | | | Value | Name | Description | 0x00 | x axis | Map to x-axis | 0x01 | y axis | Map to y-axis | 0x02 | z axis | Map to z-axis | 0x03 | reserved | reserved | 0x1 | RW | |
| Value | Name | Description | | | | | | | | | | | | | | | | | | | | |
| 0x00 | x axis | Map to x-axis | | | | | | | | | | | | | | | | | | | | |
| 0x01 | y axis | Map to y-axis | | | | | | | | | | | | | | | | | | | | |
| 0x02 | z axis | Map to z-axis | | | | | | | | | | | | | | | | | | | | |
| 0x03 | reserved | reserved | | | | | | | | | | | | | | | | | | | | |
| 5 | map_y_axis_sign | Map the y axis sign to the desired one <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0x00</td><td>non-inverted</td><td>Clear this bit to non invert the y axis</td></tr><tr><td>0x01</td><td>inverted</td><td>Set this bit to invert the y axis</td></tr></table> | | | Value | Name | Description | 0x00 | non-inverted | Clear this bit to non invert the y axis | 0x01 | inverted | Set this bit to invert the y axis | 0x0 | RW | | | | | | | |
| Value | Name | Description | | | | | | | | | | | | | | | | | | | | |
| 0x00 | non-inverted | Clear this bit to non invert the y axis | | | | | | | | | | | | | | | | | | | | |
| 0x01 | inverted | Set this bit to invert the y axis | | | | | | | | | | | | | | | | | | | | |
| 7...6 | map_z_axis | Map the z axis to desired axis <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0x00</td><td>x axis</td><td>Map to x-axis</td></tr><tr><td>0x01</td><td>y axis</td><td>Map to y-axis</td></tr><tr><td>0x02</td><td>z axis</td><td>Map to z-axis</td></tr><tr><td>0x03</td><td>reserved</td><td>reserved</td></tr></table> | | | Value | Name | Description | 0x00 | x axis | Map to x-axis | 0x01 | y axis | Map to y-axis | 0x02 | z axis | Map to z-axis | 0x03 | reserved | reserved | 0x2 | RW | |
| Value | Name | Description | | | | | | | | | | | | | | | | | | | | |
| 0x00 | x axis | Map to x-axis | | | | | | | | | | | | | | | | | | | | |
| 0x01 | y axis | Map to y-axis | | | | | | | | | | | | | | | | | | | | |
| 0x02 | z axis | Map to z-axis | | | | | | | | | | | | | | | | | | | | |
| 0x03 | reserved | reserved | | | | | | | | | | | | | | | | | | | | |
| 8 | map_z_axis_sign | Map the z axis sign to the desired one <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0x00</td><td>non-inverted</td><td>Clear this bit to non invert the z axis</td></tr><tr><td>0x01</td><td>inverted</td><td>Set this bit to invert the z axis</td></tr></table> | | | Value | Name | Description | 0x00 | non-inverted | Clear this bit to non invert the z axis | 0x01 | inverted | Set this bit to invert the z axis | 0x0 | RW | | | | | | | |
| Value | Name | Description | | | | | | | | | | | | | | | | | | | | |
| 0x00 | non-inverted | Clear this bit to non invert the z axis | | | | | | | | | | | | | | | | | | | | |
| 0x01 | inverted | Set this bit to invert the z axis | | | | | | | | | | | | | | | | | | | | |

**Register (0x5F) INTERNAL_ERROR**

DESCRIPTION: Internal error flags

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x5F) INTERNAL_ERROR | | | |
|-------------|--------------------------------|-----------|-----------|----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | n/a | n/a | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | n/a | R | R | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | int_err_2 | int_err_1 | reserved |

int_err_1: Internal error flag - long processing time, processing halted

int_err_2: Internal error flag - fatal error, processing halted

Register (0x6A) NVM_CONF

DESCRIPTION: NVM controller mode (Prog/Erase or Read only)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x6A) NVM_CONF | | | |
|-------------|--------------------------|-----|-------------|----------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | n/a | n/a | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | n/a | n/a | RW | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | nvm_prog_en | reserved |

nvm_prog_en: Enable NVM programming

| nvm_prog_en | | |
|-------------|---------|---------|
| 0x00 | disable | disable |
| 0x01 | enable | enable |

Register (0x6B) IF_CONF

DESCRIPTION: Serial interface settings

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x6B) IF_CONF | | | |
|-------------|-------------------------|-----|-----|---------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | n/a | n/a | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | if_mode |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | n/a | n/a | n/a | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | spi3 |

spi3: Configure SPI Interface Mode for primary interface

| | | |
|-------------|------|-----------------|
| spi3 | | |
| 0x00 | spi4 | SPI 4-wire mode |
| 0x01 | spi3 | SPI 3-wire mode |

if_mode: Auxiliary interface configuration

| | | |
|----------------|--------------|----------------------------------|
| if_mode | | |
| 0x00 | p_auto_s_off | Auxiliary interface:off |
| 0x01 | p_auto_s_mag | Auxiliary interface:Magnetometer |

Register (0x6D) ACC_SELF_TEST

DESCRIPTION: Settings for the sensor self-test configuration and trigger

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x6D) ACC_SELF_TEST | | | |
|-------------|-------------------------------|------------------------|----------|------------------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | n/a | n/a | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | n/a | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | acc_self_test_am p | acc_self_test_sig n | reserved | acc_self_test_en |



acc_self_test_en: Enable accelerometer self-test

| acc_self_test_en | | |
|------------------|----------|----------|
| 0x00 | disabled | disabled |
| 0x01 | enabled | enabled |

acc_self_test_sign: select sign of self-test excitation as

| acc_self_test_sign | | |
|--------------------|----------|----------|
| 0x00 | negative | negative |
| 0x01 | positive | positive |

acc_self_test_amp: select amplitude of the selftest deflection:

| acc_self_test_amp | | |
|-------------------|------|------|
| 0x00 | low | low |
| 0x01 | high | high |

Register (0x70) NV_CONF

DESCRIPTION: NVM backed configuration bits.

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x70) NV_CONF | | | |
|-------------|-------------------------|------------|-------------|--------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | n/a | n/a | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | acc_off_en | i2c_wdt_en | i2c_wdt_sel | spi_en |

spi_en: disable the I2C and enable SPI for the primary interface, when it is in autoconfig mode

| spi_en | | |
|--------|----------|--------------|
| 0x00 | disabled | I2C enabled |
| 0x01 | enabled | I2C disabled |

i2c_wdt_sel: Select timer period for I2C Watchdog

| i2c_wdt_sel | | |
|-------------|-----------|------------------------------------|
| 0x00 | wdt_short | I2C watchdog timeout after 1.25 ms |
| 0x01 | wdt_long | I2C watchdog timeout after 40 ms |



i2c_wdt_en: I2C Watchdog at the SDI pin in I2C interface mode

| i2c_wdt_en | | |
|------------|---------|----------------------|
| 0x00 | Disable | Disable I2C watchdog |
| 0x01 | Enable | Enable I2C watchdog |

acc_off_en: Add the offset defined in the off_acc_[xyz] OFFSET register to filtered and unfiltered Accelerometer data

| acc_off_en | | |
|------------|----------|----------|
| 0x00 | disabled | Disabled |
| 0x01 | enabled | Enabled |

Register (0x71) OFFSET_0

DESCRIPTION: Offset compensation for Accelerometer X-axis (NVM backed)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x71) OFFSET_0 | | | |
|-------------|--------------------------|----|----|----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | off_acc_x | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | off_acc_x | | | |

off_acc_x: Accelerometer offset compensation (X-axis).

Register (0x72) OFFSET_1

DESCRIPTION: Offset compensation for Accelerometer Y-axis (NVM backed)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x72) OFFSET_1 | | | |
|-------------|--------------------------|----|----|----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | off_acc_y | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | off_acc_y | | | |

off_acc_y: Accelerometer offset compensation (Y-axis).

Register (0x73) OFFSET_2

DESCRIPTION: Offset compensation for Accelerometer Z-axis (NVM backed)

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x73) OFFSET_2 | | | |
|-------------|--------------------------|----|----|----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | off_acc_z | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | off_acc_z | | | |

off_acc_z: Accelerometer offset compensation (Z-axis).

Register (0x7C) PWR_CONF

DESCRIPTION: Power mode configuration register

RESET: 0x03

DEFINITION (Go to [register map](#)):

| Name | Register (0x7C) PWR_CONF | | | |
|-------------|--------------------------|-----|------------------|----------------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | n/a | n/a | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | n/a | n/a | RW | RW |
| Reset Value | 0 | 0 | 1 | 1 |
| Content | reserved | | fifo_self_wakeup | adv_power_save |

| adv_power_save | | |
|----------------|---------|--|
| 0x00 | aps_off | advanced power save disabled (fast clk always enabled). |
| 0x01 | aps_on | advanced power mode enabled (slow clk is active when no measurement is ongoing.) |

| fifo_self_wakeup | | |
|------------------|---------|--|
| 0x00 | fsw_off | FIFO read disabled in advanced power saving mode. |
| 0x01 | fsw_on | FIFO read enabled after interrupt in advanced power saving mode. |

**Register (0x7D) PWR_CTRL**

DESCRIPTION: Sensor enable register

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x7D) PWR_CTRL | | | |
|-------------|--------------------------|--------|----------|--------|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | n/a | n/a | n/a | n/a |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | n/a | RW | n/a | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | reserved | acc_en | reserved | aux_en |

| | | |
|---------------|---------|--------------------------------|
| aux_en | | |
| 0x00 | mag_off | Disables the auxiliary sensor. |
| 0x01 | mag_on | Enables the auxiliary sensor. |

| | | |
|---------------|---------|-----------------------------|
| acc_en | | |
| 0x00 | acc_off | Disables the Accelerometer. |
| 0x01 | acc_on | Enables the Accelerometer. |

Register (0x7E) CMD

DESCRIPTION: Command Register

RESET: 0x00

DEFINITION (Go to [register map](#)):

| Name | Register (0x7E) CMD | | | |
|-------------|---------------------|----|----|----|
| Bit | 7 | 6 | 5 | 4 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | cmd | | | |
| Bit | 3 | 2 | 1 | 0 |
| Read/Write | RW | RW | RW | RW |
| Reset Value | 0 | 0 | 0 | 0 |
| Content | cmd | | | |

cmd: Available commands (Note: Register will always read as 0x00):

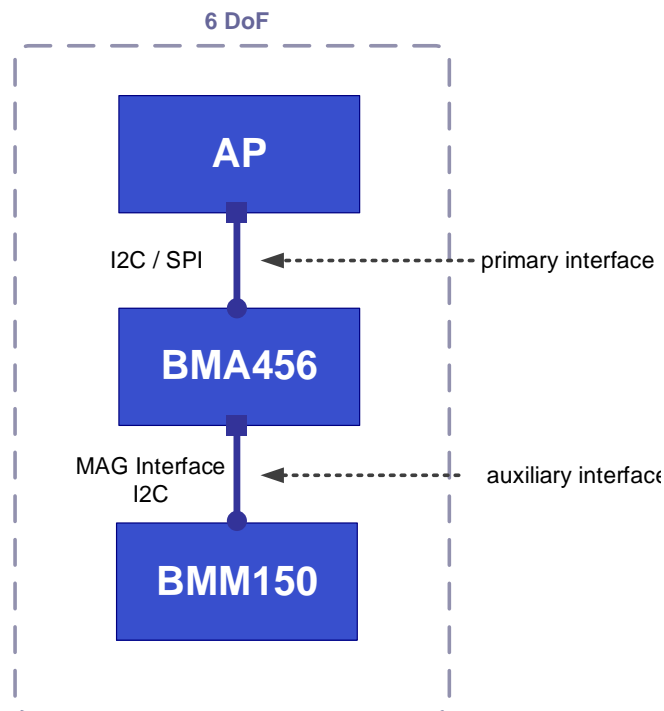
| cmd | | |
|------|------------|--|
| 0xa0 | nvm_prog | Writes the NVM backed registers into NVM |
| 0xb0 | fifo_flush | Clears all data in the FIFO, does not change FIFO_CONFIG and FIFO_DOWNS registers |
| 0xb6 | softreset | Triggers a reset, all user configuration settings are overwritten with their default state |

6. Digital Interfaces

6.1. Interfaces

Beside the standard primary interface (I2C and SPI configurable), where sensor acts as a slave to the application processor, BMA456 supports an auxiliary interface. See picture below.

If the auxiliary interface is enabled, the BMA456 can be connected to an external sensor (e.g. a magnetometer) in order to build a 6-DoF solution. Then the BMA456 will act as a master to the external sensor, reading the sensor data automatically and providing it to the application processor via the primary interface.





6.2. Primary Interface

By default, the BMA456 operates in I2C mode. The BMA456 interface can also be configured to operate in a SPI 4-wire configuration. It can also be re-configured by software to work in 3-wire mode instead of 4-wire mode.

All 3 possible digital interfaces share partly the same pins. The mapping for the primary interface of the BMA456 is given in the following table:

| Pin# | Name | I/O Type | Description | Connect to (Primary IF) | | |
|------|------|-------------|---|-------------------------|---------------------|---------------------------------|
| | | | | in SPI4W | in SPI3W | in I2C |
| 1 | SDO | Digital I/O | Serial data output in SPI Address select in I ² C mode see chapter 7.2 | SDO | DNC (float) | GND for default I2C 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 |
| 5 | INT1 | Digital I/O | Interrupt output 1 (default) (Input for external FIFO sync) * | INT1 (FIFO sync) | INT1 (FIFO sync) | INT1 (FIFO sync) |
| 6 | INT2 | Digital I/O | Interrupt output 2 (default) (Input for external FIFO sync) * | INT2 (FIFO sync) | INT2 (FIFO sync) | INT2 (FIFO sync) |
| 10 | CSB | Digital in | Chip select for SPI mode | CSB | CSB | V _{DDIO} |
| 12 | SCX | Digital in | SCK for SPI serial dock SCL for I ² C serial dock | SCK | SCK | SCL |

* INT1 and/or INT2 can also be configured as an input in case the external data synchronization in FIFO is used. See chapter 0. If INT1 and/or INT2 are not used, please do not connect them (DNC).

The following table shows the electrical specifications 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 | pF |
| I ² C Bus Load Capacitance (max drive capability) | C _{I2C_Load} | | | | 400 | pF |

6.3. Primary Interface I2C/SPI Protocol Selection

The protocol is automatically selected based on the chip select CSB pin behavior after power-up.

At reset/power-up, BMA456 is in I2C mode. If CSB is connected to VDDIO during power-up and not changed the sensor interface works in I2C mode. For using I2C, it is recommended to hard-wire the CSB line to VDDIO. Since power-on-reset is only executed when, both VDD and VDDIO are established, there is no risk of incorrect protocol detection due to power-up sequence.

If CSB sees a rising edge after power-up, the BMA456 interface switches to SPI until a reset or the next power-up occurs. Therefore, a CSB rising edge is needed before starting the SPI communication. Hence, it is recommended to perform a SPI single read of register [CHIP_ID](#) (the obtained value will be invalid) before the actual communication start, in order to use the SPI interface.

If toggling of the CSB bit is not possible without data communication, there is in addition the spi_en bit in Register [NV_CONF](#), which can be used to permanently set the primary interface to SPI without the need to toggle the CSB pin at every power-up or reset.

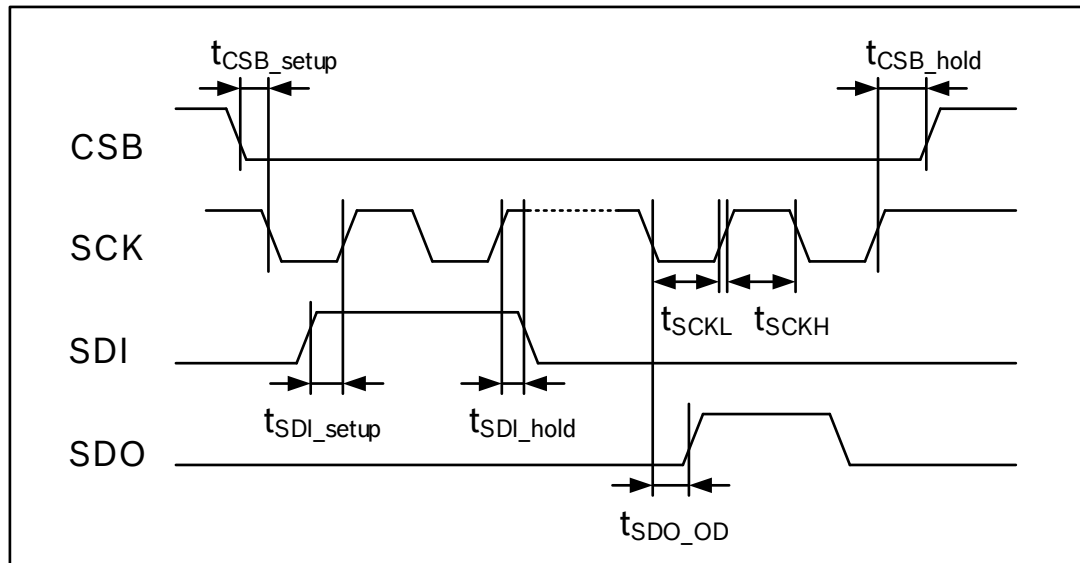
6.4. SPI interface and protocol

The timing specification for SPI of the BMA456 is given in the following table:

SPI timing, valid at $V_{DDIO} \geq 1.71V$

| Parameter | Symbol | Condition | Min | Max | Units |
|--|-------------------------|---|------|-----|---------|
| Clock Frequency | f_{SPI} | Max. Load on SDI or SDO = 30pF, $V_{DDIO} \geq 1.62V$ | | 10 | MHz |
| | | $V_{DDIO} < 1.62V$ | | 7 | MHz |
| SCK Low Pulse | t_{SCKL} | $V_{DDIO} \geq 1.62V$ | 45 | | ns |
| SCK High Pulse | t_{SCKH} | $V_{DDIO} \geq 1.62V$ | 45 | | ns |
| SCK Low Pulse | t_{SCKL} | $V_{DDIO} < 1.62V$ | | 66 | ns |
| SCK High Pulse | t_{SCKH} | $V_{DDIO} < 1.62V$ | | 66 | ns |
| SDI Setup Time | t_{SDI_setup} | | 20 | | ns |
| SDI Hold Time | t_{SDI_hold} | | 20 | | ns |
| SDO Output Delay | t_{SDO_OD} | Load = 30pF, $V_{DDIO} \geq 1.62V$ | | 30 | ns |
| CSB Setup Time | t_{CSB_setup} | | 40 | | ns |
| CSB Hold Time | t_{CSB_hold} | | 40 | | ns |
| Idle time between write accesses, suspend mode, low-power mode 1 | $t_{IDLE_wa_cc_sum}$ | | 1000 | | μs |
| Idle time after write and read access, active state | $t_{IDLE_wr_act}$ | | 2 | | μs |

The following figure shows the definition of the SPI timings:



SPI timing diagram

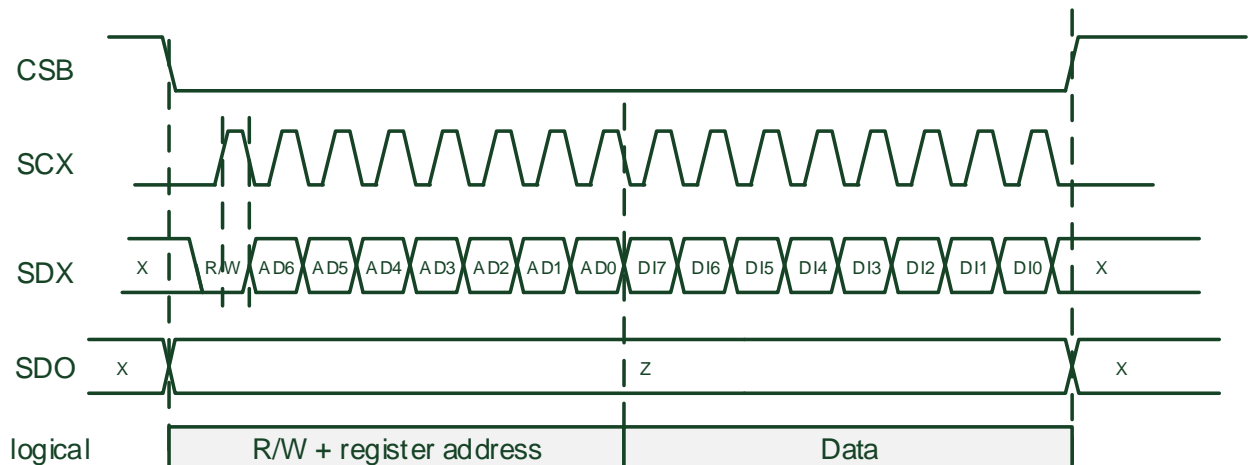
The SPI interface of the BMA456 is compatible with two modes, '00' [CPOL = '0' and CPHA = '0'] and '11' [CPOL = '1' and CPHA = '1']. The automatic selection between '00' and '11' is controlled based on the value of SCK after a falling edge of CSB.

Two configurations of the SPI interface are supported by the BMA456: 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 [IF_CONF.spi3](#) = 0b1. Pin SDI is used as the common data pin in 3-wire configuration.

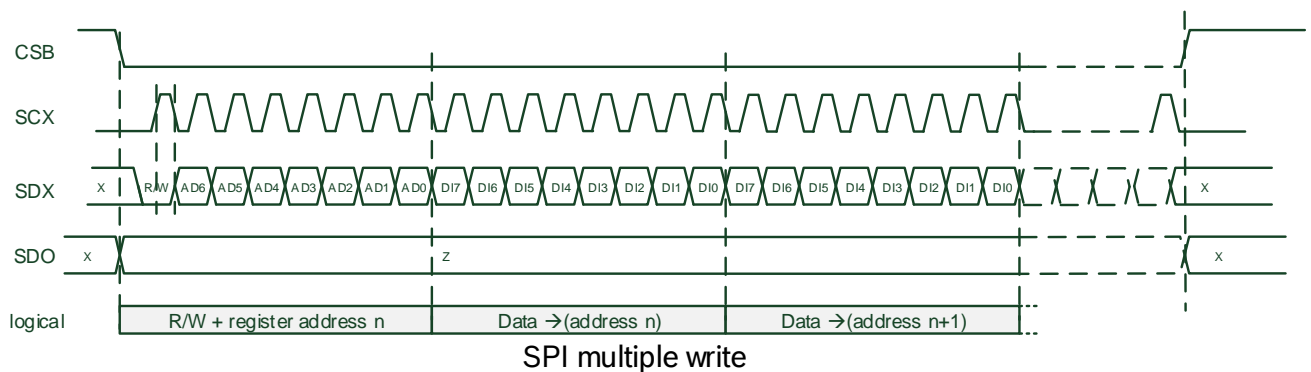
For single byte read as well as write operations, 16-bit protocols are used. The BMA456 also supports multiple-byte read and write operations.

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.

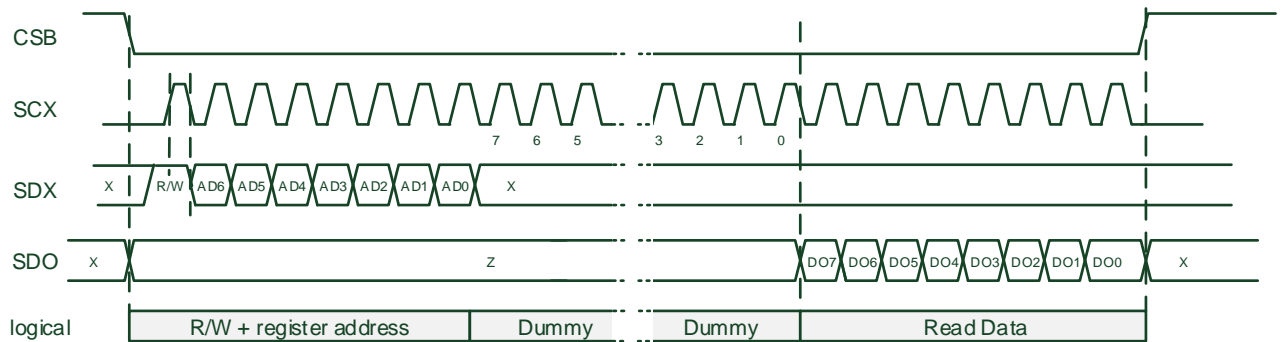
The basic write operation waveform for 4-wire configuration is depicted in the following figure. During the entire write cycle SDO remains in high-impedance state.

**4-wire basic SPI write sequence (mode '00')**

Multiple write 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 write access as long as CSB stays active low. The principle of multiple write is shown in figure below:



The basic read operation waveform for 4-wire configuration is depicted in the figure below. Please note that the first byte received from the BMA456 via the SDO line correspond to a dummy byte and the 2nd byte correspond to the value read out of the specified register address. That means, for a basic read operation two bytes have to be read and the first has to be dropped and the second byte must be interpreted.



4-wire basic SPI read sequence (mode '00')

The data bits are used as follows:

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

AD6-AD0: Address

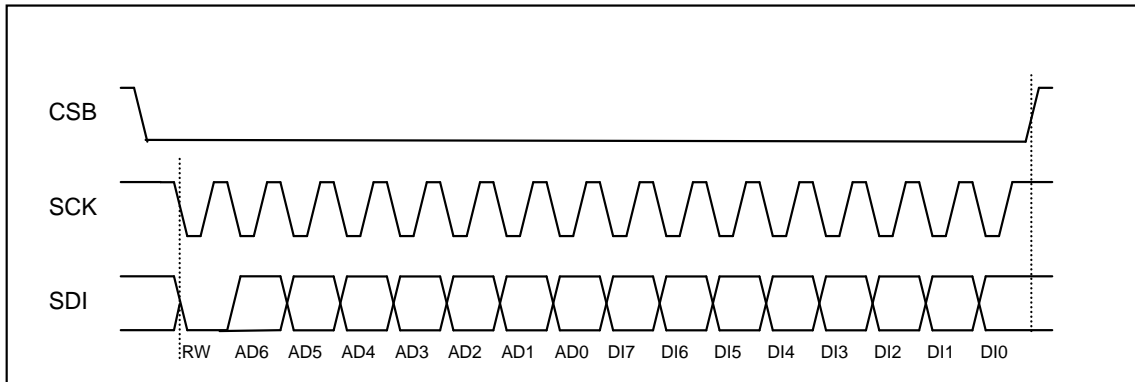
DI7-DI0: When in write mode, these are the data SDI, which will be written into the address.

DO7-DO0: 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. Please note that the first byte received from the BMA456 via the SDO line correspond to a dummy byte and the 2nd byte correspond to the value read out of the specified register address. The successive bytes read out correspond to values of incremented register addresses. That means, for a multiple read operation of n bytes, n+1 bytes have to be read, the first has to be dropped and the successive bytes must be interpreted.

In SPI 3-wire configuration CSB (chip select low active), SCK (serial clock), and SDI (serial data input and output) pins are used. While SCK is high, 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 wave-form (read or write access) for 3-wire configuration is depicted in the figure below:



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

6.5. Primary I2C Interface

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 default I²C address of the device is 0b00011000 (0x18). It is used if the SDO pin is pulled to 'GND'. The alternative address 0b00011001 (0x19) is selected by pulling the SDO pin to 'VDDIO'.

The I²C interface of the BMA456 is compatible with the I²C Specification UM10204 Rev. 03 (19 June 2007), available at <http://www.nxp.com>. The BMA456 supports **I²C standard mode and fast mode**, only 7-bit address mode is supported. For V_{DDIO} = 1.2V to 1.62 V the guaranteed voltage output levels are slightly relaxed as described in Table 1 of the electrical specification section.

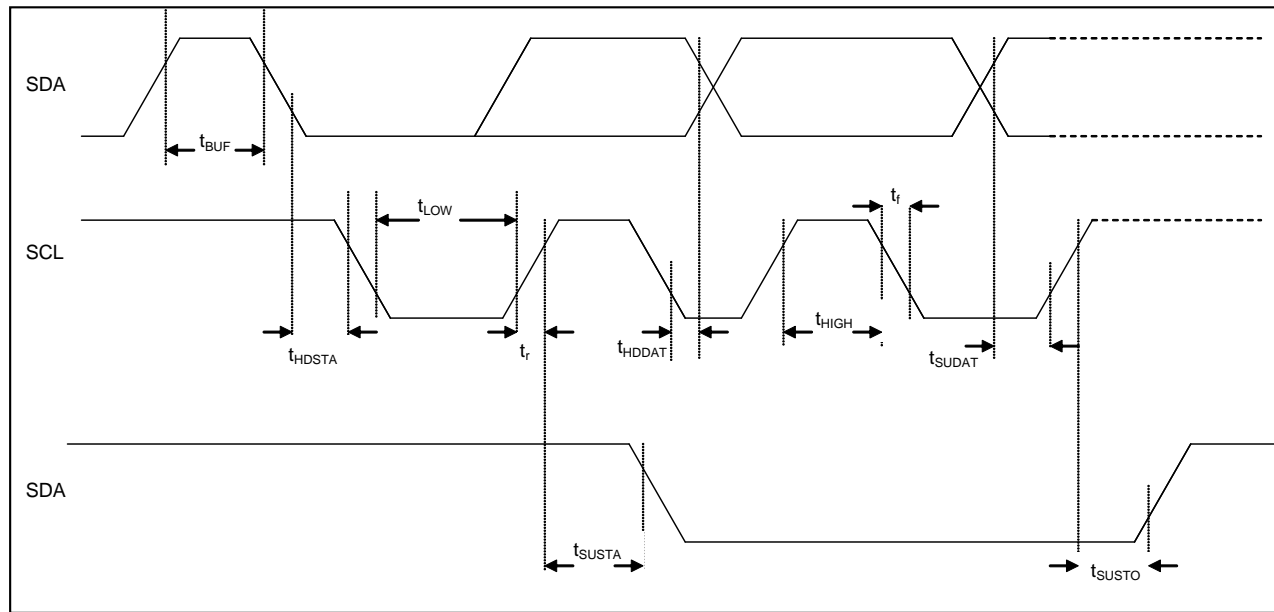
BMA456 also supports an **extended I²C mode** that allows using clock frequencies up to 1 MHz. In this mode all timings of the fast mode apply and it additionally supports clock frequencies up to 1MHz.

The timing specification for I²C of the BMA456 is given in the following table:

| Parameter | Symbol | Condition | Min | Max | Units |
|---|---------------------------|----------------|------|------|-------|
| Clock Frequency | f _{SCL} | | | 1000 | 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} | low power mode | 400 | | |
| | | normal mode | 1.3 | | |
| Idle time between write accesses, normal mode, standby mode, low-power mode | t _{IDLE_wacc_nm} | low power mode | 1000 | | |
| | | normal mode | 1.3 | | |
| Idle time between write accesses, suspend mode, low-power mode | t _{IDLE_wacc_s} | | 1000 | | |



The figure below shows the definition of the I²C timings given in Table 28:



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.

ACKS: 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 SCL toggling from 'VDDIO' to 'GND') 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 (ACKS = 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:

| Start | Slave Address | | | | | | | R/W | ACK | | Register address (0x41) | | | | | | | ACK | Register data (0x01) | | | | | | | ACK | Stop |
|-------|---------------|---|---|---|---|---|---|-----|-----|---|-------------------------|---|---|---|---|---|---|-----|----------------------|---|---|---|---|---|---|-----|------|
| S | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | X | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | P |

| | |
|---|-----------------|
|  | Master -> Slave |
|  | Slave -> Master |

I²C write

Multi-byte writes are supported without restriction on normal registers with auto-increment, on special registers with address trap.

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 (S). 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 (ACKS = 0) to enable further data transfer. A NACKM (ACKS = 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 since 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.



| Start | Slave I2C ID | | | | | | | | R/W | ACK | Register address (0x12) | | | | | | | | ACK |
|-------|--------------|---|---|---|---|---|---|---|-----|-----|-------------------------|---|---|---|---|---|---|---|-----|
| S | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | X | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------------|--------------|---|---|---|---|---|---|---|-----|-----|------------------------------|---|---|---|---|---|---|---|-----|------------------------------|---|---|---|---|---|---|---|-----|
| Repeat Start | Slave I2C ID | | | | | | | | R/W | ACK | Data byte | | | | | | | | ACK | Data byte | | | | | | | | ACK |
| | | | | | | | | | | | Register data - address 0x12 | | | | | | | | | Register data - address 0x13 | | | | | | | | |
| Sr | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | X | X | X | X | X | X | X | X | 0 | X | X | X | X | X | X | X | X | 0 | ... |

Master -> Slave

Slave -> Master

| | | | | | | | | | | | | | | | | | | | |
|-----|------------------------------|---|---|---|---|---|---|---|-----|------------------------------|---|---|---|---|---|---|---|-----|-----|
| ... | Data byte | | | | | | | | ACK | Data byte | | | | | | | | ACK | |
| | Register data - address 0x14 | | | | | | | | | Register data - address 0x15 | | | | | | | | | |
| ... | X | X | X | X | X | X | X | X | 0 | X | X | X | X | X | X | X | X | 0 | ... |

| | | | | | | | | | | | | | | | | | | | |
|-----|------------------------------|---|---|---|---|---|---|---|-----|------------------------------|---|---|---|---|---|---|---|------|------|
| ... | Data byte | | | | | | | | ACK | Data byte | | | | | | | | NACK | Stop |
| | Register data - address 0x16 | | | | | | | | | Register data - address 0x17 | | | | | | | | | |
| ... | X | X | X | X | X | X | X | X | 0 | X | X | X | X | X | X | X | X | 1 | P |

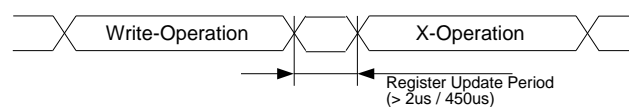
 Master -> Slave
 Slave -> Master

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 BMA456. The activity and the timer period of the WDT can be configured through the bits [NV_CONF.i2c_wdt_en](#) and [NV_CONF.i2c_wdt_sel](#).

6.6. SPI and I²C Access Restrictions

In order to allow for the correct internal synchronization of data written to the BMA456, 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 other modes.

As illustrated in the figure below, an interface idle time of at least 2 μ s is required following a write operation when the device operates in normal mode. In suspend mode an interface idle time of least 1000 μ s is required.



Post-Write Access Timing Constraints

6.7. Auxiliary Interface

The BMA456 allows attaching an external sensor (MAG-sensor) to a BMA456 via the auxiliary interface. The connection diagrams for the auxiliary interface are depicted in the chapter 7.3. The timings of the secondary I²C interface are the same as for the primary I²C interface, see chapter 6.5.

BM456 acts as a master of the secondary interface, controls the data acquisition of the MAG-sensor (slave of the secondary interface) and presents the data to the application processor (AP) in the user registers of the BMA456 through the primary interface. No external pull-up resistors need to be connected, since an internal pull-up can be configured in the BMA456 (default value: internal pull-up is off, please contact your regional sales representative if you want to use this functionality). No additional I²C master or slave devices must be attached to the magnetometer interfaces.

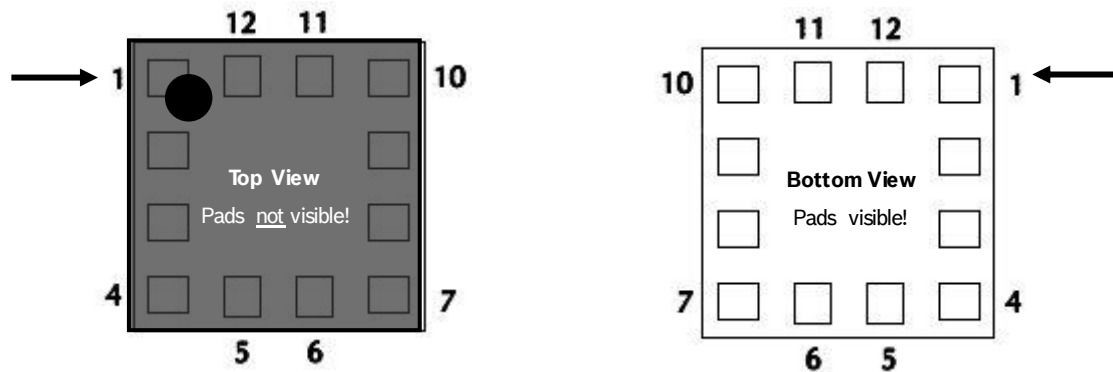
The BMA456 autonomously reads out the sensor data from BMM150 without intervention of the AP and stores the data in its data registers (per default) and FIFO (see [Register FIFO CONFIG 1.fifo aux en](#)). The initial setup of the BMM150 after power-on is done through indirect addressing in the BMA456. From a system perspective the initialization for BMM150 when attached to BMA456 should be possible within 100ms.

More information about the usage of Auxiliary Interface can be found in chapter 4.8.



7. Pin-out and Connection Diagrams

7.1. Pin-out



Pin description

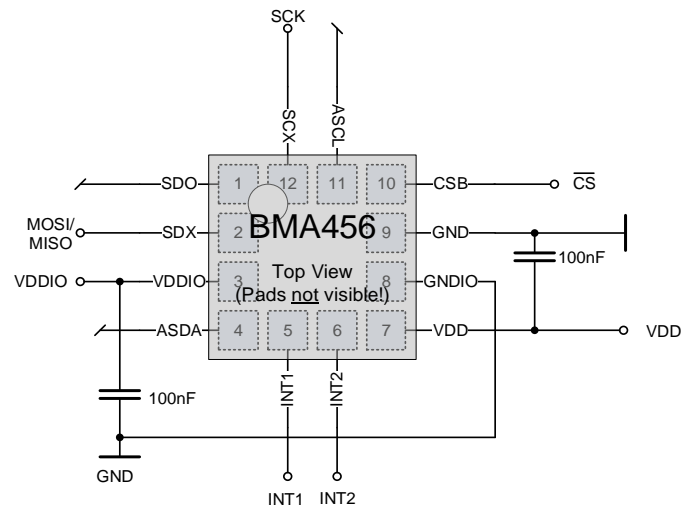
| Pin# | Name | I/O Type | Description | Connect to | | |
|------|-------|-------------|---|---|--|---|
| | | | | in SPI 4W | In SPI 3W | in I ² C |
| 1 | SDO | Digital I/O | Serial data output in SPI Address select in I ² C mode see chapter 7.2 | SDO | DNC (float) | GND for default I ² C addr. |
| 2 | 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 | ASDA | Digital I/O | Serial data I/O – Secondary Interface (I ² C Master for Magnetometer) | VDDIO/ GNDIO/NC or (ASDA - Secondary interface) | VDDIO/ GNDIO/NC or (ASDA - Secondary interface) | VDDIO/ GNDIO/NC or (ASDA - Secondary interface) |
| 5 | INT1 | Digital I/O | Interrupt output 1 (default) (Input for external FIFO sync) * | INT1 (FIFO sync) | INT1 (FIFO sync) | INT1 (FIFO sync) |
| 6 | INT2 | Digital I/O | Interrupt output 2 (default) (Input for external FIFO sync) * | INT2 (FIFO sync) | INT2 (FIFO sync) | INT2 (FIFO sync) |
| 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 | V _{DDIO} |
| 11 | ASCL | Digital out | Digital dock (out) – Secondary Interface (I ² C Master for Magnetometer) | VDDIO/ GNDIO/NC or (ASCL - Secondary interface) | VDDIO/ GNDIO/ NC or (ASCL - Secondary interface) | VDDIO/ GNDIO/ NC or (ASCL - Secondary interface) |
| 12 | SCX | Digital in | SCK for SPI serial dock SCL for I ² C serial dock | SCK | SCK | SCL |

* INT1 and/or INT2 can also be configured as an input in case the external data synchronization in FIFO is used. See chapter 4.5. If INT1 and/or INT2 are not used, please do not connect them (DNC).

7.2. Connection Diagrams without Auxiliary Interface

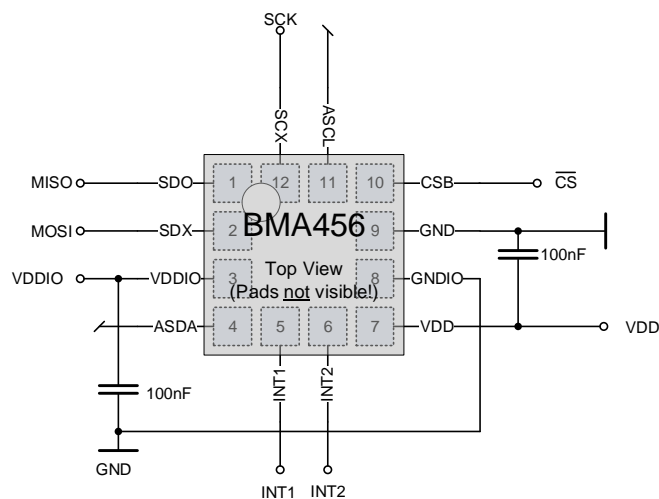
SPI

3-wire



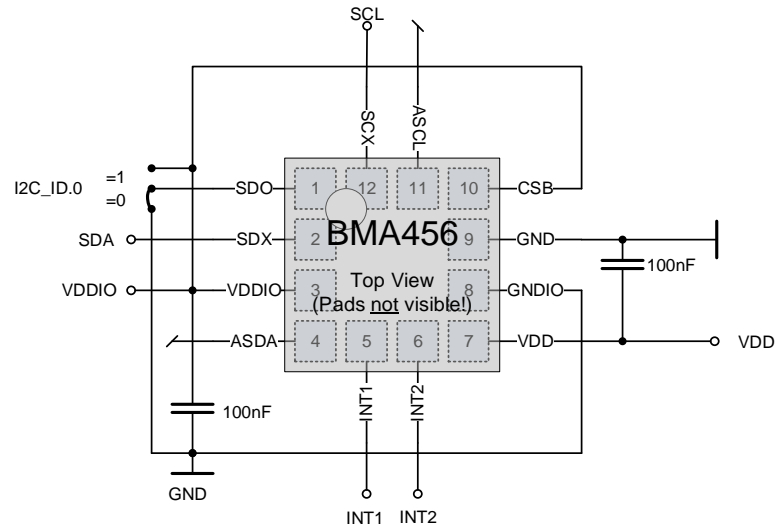
It is recommended to use 100nF decoupling capacitors at pin 3 (VDDIO) and pin 7 (VDD).

4-wire



It is recommended to use 100nF decoupling capacitors at pin 3 (VDDIO) and pin 7 (VDD).

I2C

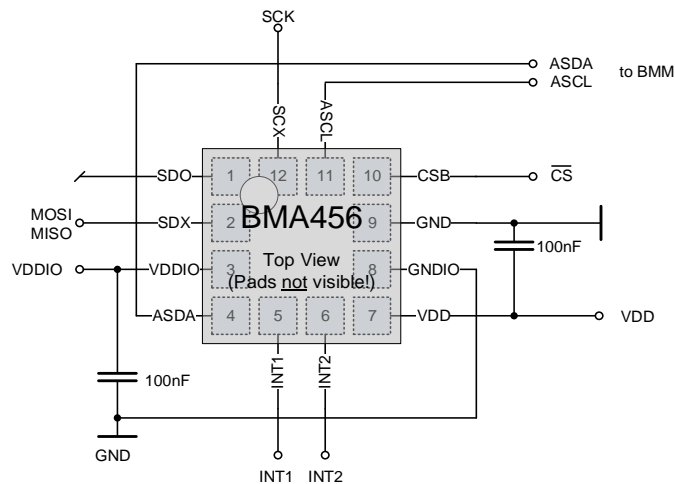


It is recommended to use 100nF decoupling capacitors at pin 3 (VDDIO) and pin 7 (VDD).

7.3. Connection Diagrams with Auxiliary Interface

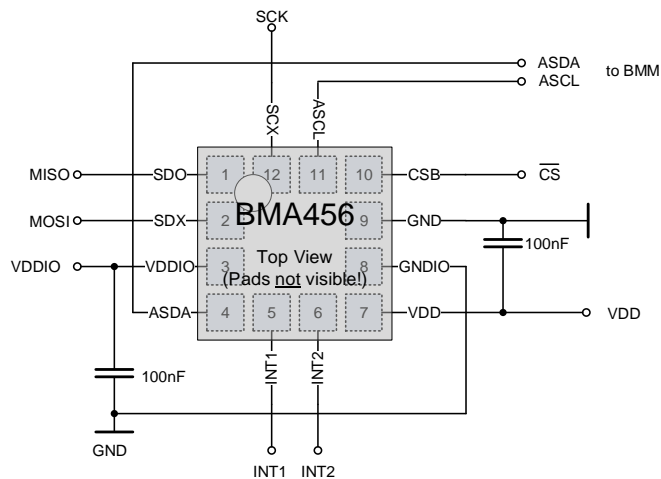
SPI

3-wire



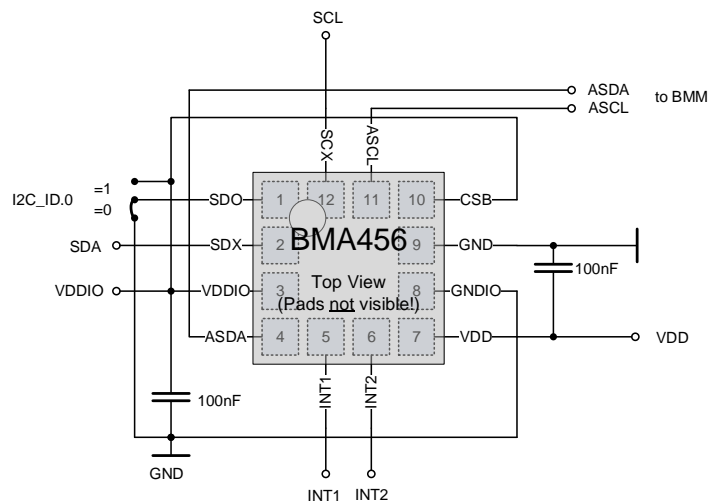
It is recommended to use 100nF decoupling capacitors at pin 3 (VDDIO) and pin 7 (VDD).

4-wire



It is recommended to use 100nF decoupling capacitors at pin 3 (VDDIO) and pin 7 (VDD).

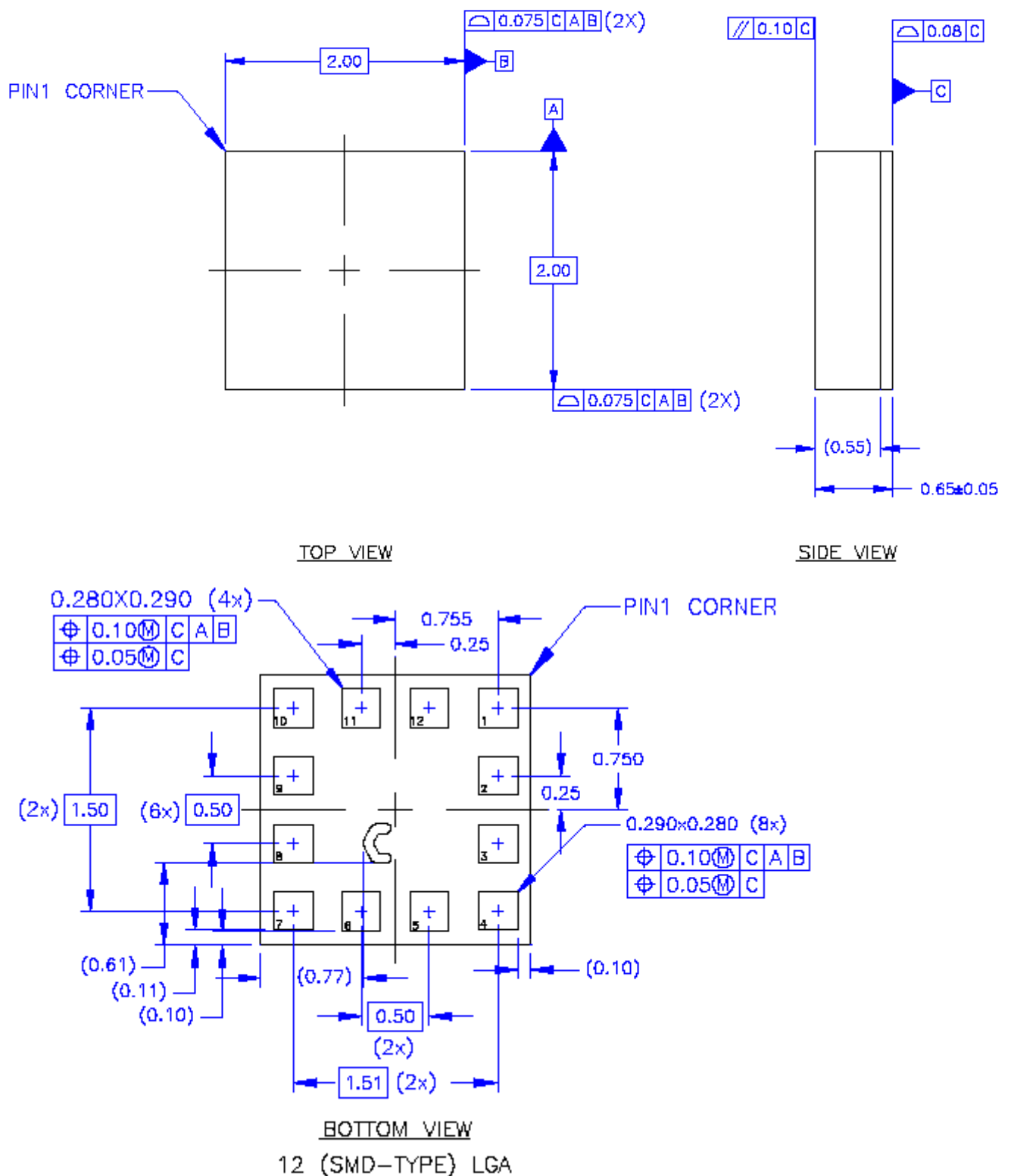
I2C



It is recommended to use 100nF decoupling capacitors at pin 3 (VDDIO) and pin 7 (VDD).

8. Package

8.1. Package outline dimensions

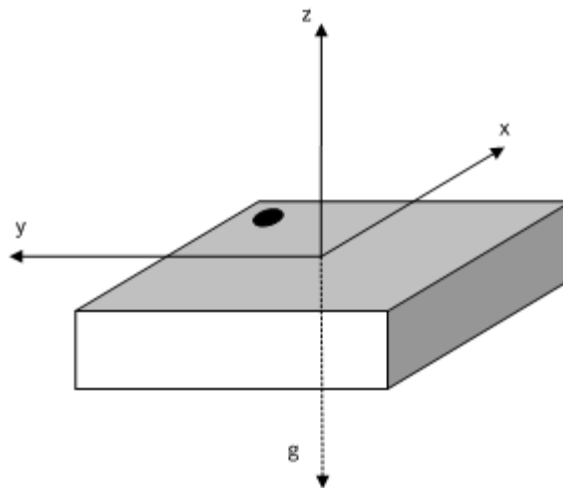


8.2. Sensing axis orientation

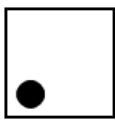
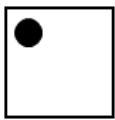
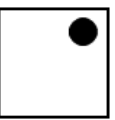
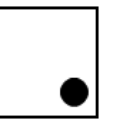
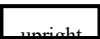
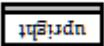
If the sensor is accelerated in the indicated directions, the corresponding channel will deliver a positive acceleration signal (dynamic acceleration). If the sensor is at rest and the force of gravity is acting along the indicated directions, the output of the corresponding channel will be negative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

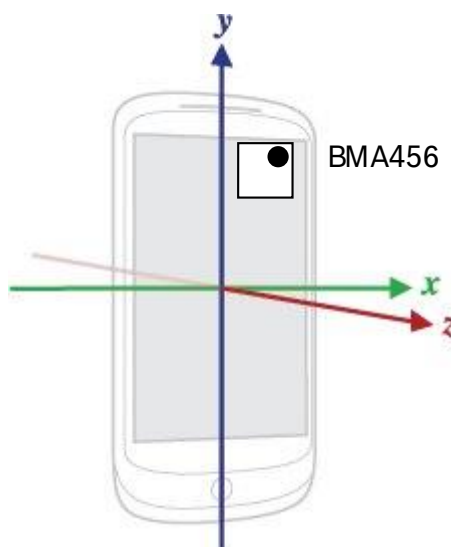
- $\pm 0g$ for the X channel
- $\pm 0g$ for the Y channel
- $+ 1g$ for the Z channel



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 $\pm 4g$ range setting, a 16 bit resolution, and a top down gravity vector as shown above.

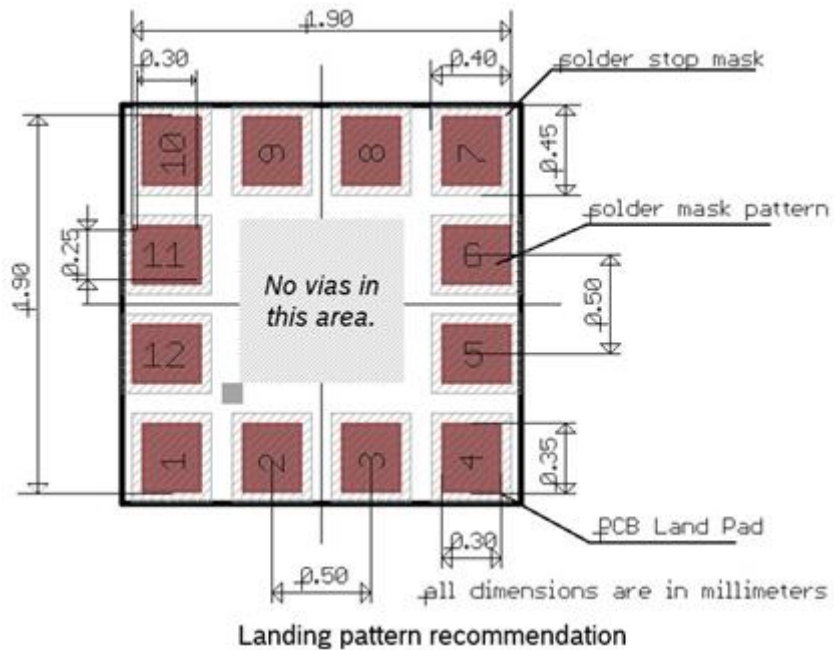
| Sensor Orientation (gravity vector ↓) |  |  |  |  |  |  |
|--|---|---|---|---|---|---|
| Output Signal X | 0g / 0 LSB | 1g / 8192 LSB | 0g / 0 LSB | -1g / -8192 LSB | 0g / 0 LSB | 0g / 0 LSB |
| Output Signal Y | -1g / -8192 LSB | 0g / 0 LSB | 1g / 8192 LSB | 0g / 0 LSB | 0g / 0 LSB | 0g / 0 LSB |
| Output Signal Z | 0g / 0 LSB | 0g / 0 LSB | 0g / 0 LSB | 0g / 0 LSB | 1g / 8192 LSB | -1g / -8192 LSB |

For reference the figure below shows the Android device orientation with an integrated BMA456.



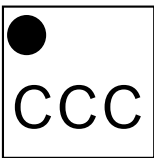
8.3. Landing pattern recommendation

The recommended landing pattern for the BMA456 on customer's PCB is given in the following figure. It is recommended to avoid any wiring underneath the BMA456 (shaded area).

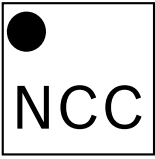


8.4. Marking

Mass production

| Labeling | Name | Symbol | Remark |
|---|---------------------------|--------|---|
|  | Counter ID | CCC | 3 alphanumeric digits, variable to generate trace-code. |
| | Pin 1 identifier top side | ● | -- |
| | | | |

Engineering samples

| Labeling | Name | Symbol | Remark |
|---|---------------------------|--------|--|
|  | Eng. sample ID | E, N | 2 alphanumeric digits, fixed to identify engineering sample, N = "C" |
| | Sample ID | CC | 2 alphanumeric digits, variable to generate trace-code. |
| | Pin 1 identifier top side | ● | -- |
| | | | |

8.5. Soldering guidelines

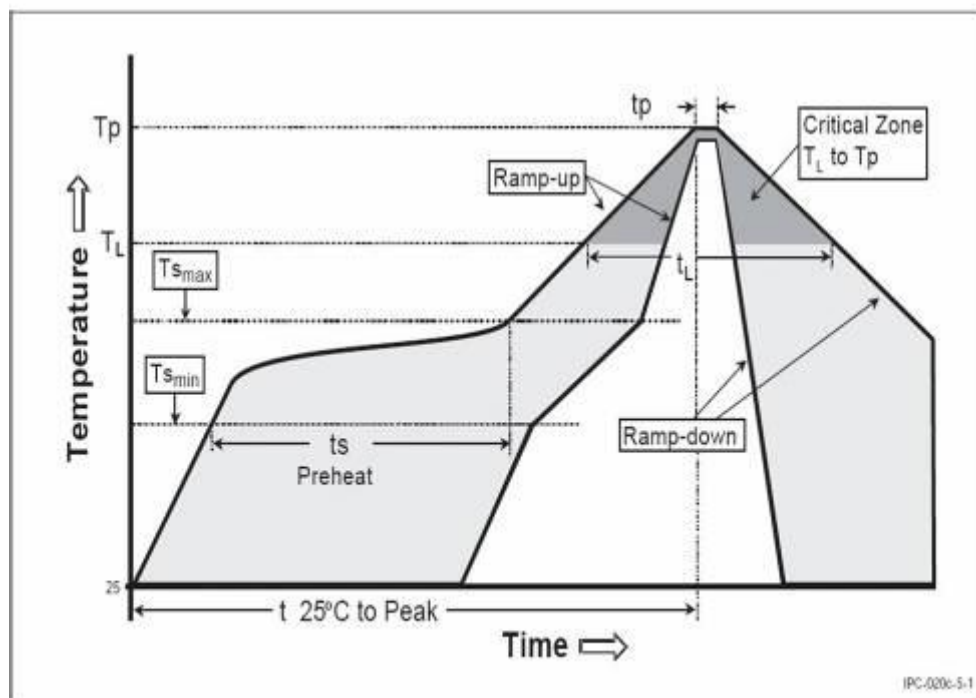
The moisture sensitivity level of the BMA456E sensors corresponds to JEDEC Level 1, see also

- 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 ($T_{s_{max}}$ to T_p) | | 3° C/second max. |
| Preheat <ul style="list-style-type: none"> – Temperature Min ($T_{s_{min}}$) – Temperature Max ($T_{s_{max}}$) – Time ($t_{s_{min}}$ to $t_{s_{max}}$) | | 150 °C 200 °C 60-180 seconds |
| Time maintained above: <ul style="list-style-type: none"> – Temperature (T_L) – Time (t_L) | | 217 °C 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 top side of the package, measured on the package body surface.



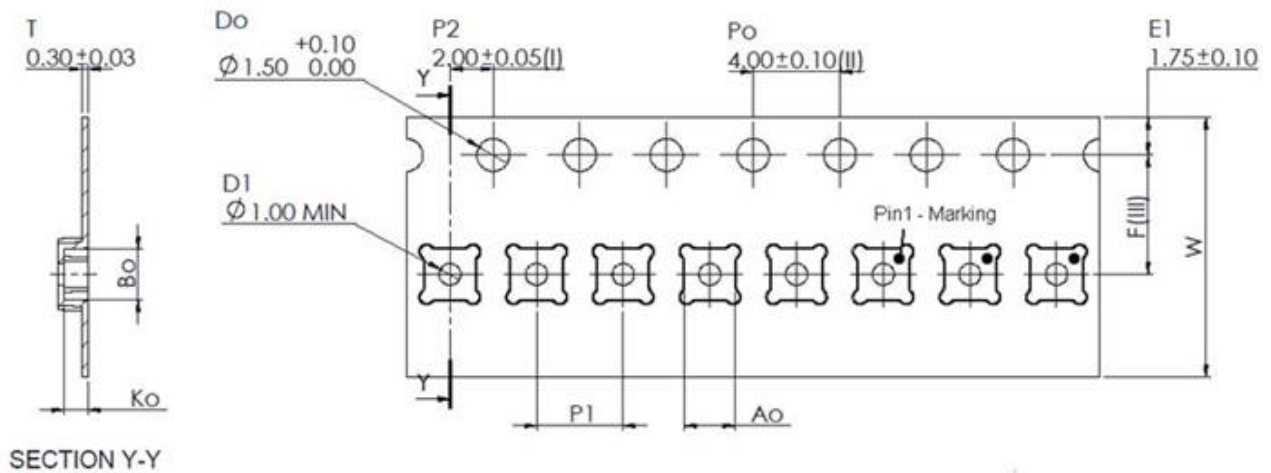


8.6. Handling instructions

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.

**8.7. Tape and Reel specification**

| | | |
|----|-------|---------------|
| Ao | 2.35 | +/- 0.05 |
| Bo | 2.30 | +/- 0.05 |
| Ko | 1.10 | +/- 0.05 |
| F | 5.50 | +/- 0.05 |
| P1 | 4.00 | +/- 0.10 |
| W | 12.00 | +0.30 / -0.10 |



8.8. Environmental safety

The BMA456 sensor meets the requirements of the EC restriction of hazardous substances (RoHS) directive, see also:

Directive 2011/65/EU of the European Parliament and of the Council of 8 September 2011 on the restriction of the use of certain hazardous substances in electrical and electronic equipment.

Halogen content

The BMA456 is halogen-free. For more details on the corresponding analysis results please contact your Bosch Sensortec representative.

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 BMA456.

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 BMA456 product.

9. Legal disclaimer

9.1. Engineering samples

Engineering Samples are marked with an asterisk (*) 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.

9.2. 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 security sensitive systems. Security sensitive systems are those for which a malfunction is expected to lead to bodily harm or significant property damage. In addition, they are not fit for use in products which interact with motor vehicle systems.

The resale and/or use of 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 must monitor the market for the purchased products, particularly with regard to product safety, and inform Bosch Sensortec without delay of all security relevant incidents.

9.3. 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.

10.Document history and modification

| Rev. No | Chapter | Description of modification/changes | Date |
|---------|----------------------|--|-------------|
| 1.0 | | Document creation | 07 Aug 2017 |
| 1.1 | 8.1 | Update | 26 Oct 2017 |
| 1.2 | 4.8, 6.7 6.2, 7.1 | Fixed typos Changed CSB recommendation for I ² C | April 2019 |

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-BMA456-DS000-02
Revision_1.2_April_2019

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Mikroe:

[MIKROE-3440](#)