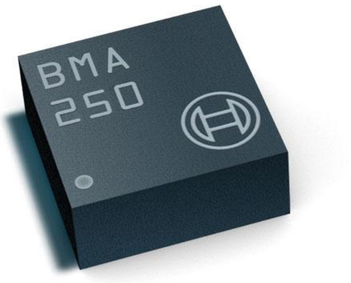
BMA250
Digital, triaxial acceleration sensor

Data sheet

Bosch Sensortec

BOSCH
Invented for life



BMA250 Data sheet

Ordering code	Please contact your Bosch Sensortec representative for the ordering code
Package type	12-pin LGA
Document revision	1.15
Document release date	31 May 2012
Document number	BST-BMA250-DS002-05
Technical reference code(s)	0 273 141 121
Notes	Data in this document are subject to change without notice. Product photos and pictures are for illustration purposes only and may differ from the real product's appearance. Not intended for publishing.

BMA250
Digital, triaxial ±2g to ±16g acceleration sensor with intelligent on-chip motion-triggered interrupt controller

Key features

•	Ultra-Small package	LGA package (12 pins), footprint 2mm x 2mm, height 0.95mm
•	Digital interface	SPI (4-wire, 3-wire), I²C, 2 interrupt pins V_{DDIO} voltage range: 1.2V to 3.6V
•	Programmable functionality	Acceleration ranges ±2g/±4g/±8g/±16g Low-pass filter bandwidths 1kHz - <8Hz
•	On-chip interrupt controller	Motion-triggered interrupt-signal generation for - new data - any-motion (slope) detection - tap sensing (single tap / double tap) - orientation recognition - flat detection - low-g/high-g detection Stand-alone capability (no microcontroller needed)
•	Ultra-low power ASIC	Low current consumption, short wake-up time, Advanced features for system power management
•	RoHS compliant, halogen-free	

Typical applications

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

General description

The BMA250 is a triaxial, low-g acceleration sensor with digital output for consumer market applications. It allows measurements of acceleration in three perpendicular axes. An evaluation circuitry (ASIC) converts the output of a micromechanical acceleration-sensing structure (MEMS) that works according to the differential capacitance principle.

Package and interfaces of the BMA250 have been defined to match a multitude of hardware requirements. Since the sensor features an ultra-small footprint and a flat package it is ingeniously suited for mobile applications.

The BMA250 offers a variable V_{DDIO} voltage range from 1.2V to 3.6V and can be programmed to optimize functionality, performance and power consumption in customer specific applications. In addition it features an on-chip interrupt controller enabling motion-based applications without use of a microcontroller.

The BMA250 senses tilt, motion and shock vibration in cell phones, handhelds, computer peripherals, man-machine interfaces, virtual reality features and game controllers.

Index of Contents

1. Specification

If not stated otherwise, the given values are over lifetime and full performance temperature and voltage ranges, minimum/maximum values are ±3 σ.

Table 1: Parameter Specification

OPERATING CONDITIONS						
Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Units**
Acceleration Range	g_{FS2g}	Selectable via serial digital interface		±2		g
	g_{FS4g}			±4		g
	g_{FS8g}			±8		g
	g_{FS16g}			±16		g
Supply Voltage Internal Domains	V_{DD}		1.62	2.4	3.6	V
Supply Voltage I/O Domain	V_{DDIO}		1.2	2.4	3.6	V
Voltage Input Low Level	V_{IL}	SPI & I²C			$0.3V_{DDIO}$	-
Voltage Input High Level	V_{IH}	SPI & I²C	$0.7V_{DDIO}$			-
Voltage Output Low Level	V_{OL}	V_{DDIO} = 1.62V I_{OL} = 3mA, SPI & I²C			$0.2V_{DDIO}$	-
		V_{DDIO} = 1.2V I_{OL} = 3mA, SPI & I²C			0.23 V_{DDIO}	-
Voltage Output High Level	V_{OH}	V_{DDIO} = 1.62V I_{OL} = 2mA, SPI & I²C	$0.8V_{DDIO}$			-
		V_{DDIO} = 1.2V I_{OL} = 2mA, SPI & I²C	0.62 V_{DDIO}			-
Supply Current in Normal Mode	I_{DD}	Nominal V_{DD} supplies T_A=25°C, bw = 1kHz		139		μA
Supply Current in Low-Power Mode	I_{DDlp}	Nominal V_{DD} supplies T_A=25°C, bw = 1kHz sleep duration ≥ 25ms		7		μA
Supply Current in Suspend Mode	I_{DDsm}	Nominal V_{DD} supplies T_A=25°C		0.5		μA
Wake-Up Time	t_{w_up}	from Low-Power Mode or Suspend Mode, bw = 1kHz		0.8		ms
Start-Up Time	t_{s_up}	POR, bw = 1kHz		2		ms
Operating Temperature	T_A		-40		+85	°C

OUTPUT SIGNAL						
Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Units**
Device Resolution	D_{res}	g_{FS2g}		3.91		mg
Sensitivity	S_{2g}	g_{FS2g}, T_A=25°C		256		LSB/g
	S_{4g}	g_{FS4g}, T_A=25°C		128		LSB/g
	S_{8g}	g_{FS8g}, T_A=25°C		64		LSB/g
	S_{16g}	g_{FS16g}, T_A=25°C		32		LSB/g
Sensitivity Temperature Drift	TCS	g_{FS2g}, -40°C ≤ T_A ≤ +85°C Nominal V_{DD} supplies		±0.02		%/K
Zero-g Offset	Off	g_{FS2g}, T_A=25°C Nominal V_{DD} supplies		±80		mg
Zero-g Offset Temperature Drift	TCO	g_{FS2g}, -40°C ≤ T_A ≤ +85°C Nominal V_{DD} supplies		±1		mg/K
Bandwidth	bw_8	1st order filter, selectable via serial digital interface		8		Hz
	bw_{16}			16		Hz
	bw_{31}			31		Hz
	bw_{63}			63		Hz
	bw_{125}			125		Hz
	bw_{250}			250		Hz
	bw_{500}			500		Hz
	bw_{1000}			1000		Hz
Nonlinearity	NL	best fit straight line		±0.5		%FS
Output Noise	n_{rms}	g_{FS2g}, T_A=25°C Nominal V_{DD} supplies Normal mode		0.8		mg/√Hz
Power Supply Rejection Rate	PSRR	T_A=25°C Nominal V_{DD} supplies			20	mg/V
Temperature Sensor Measurement Range	T_S	T_A=25°C Nominal V_{DD} supplies	-40		+87.5	°C
Temperature Sensor Slope	dT_S	T_A=25°C Nominal V_{DD} supplies		0.5		LSB/K
Temperature Sensor Offset	OT_S	T_A=25°C Nominal V_{DD} supplies		±5		K

MECHANICAL CHARACTERISTICS						
Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Units**
Cross Axis Sensitivity	S	relative contribution between any two of the three axes		1		%
Alignment Error	E_A	relative to package outline		±0.5		°

2. Absolute maximum ratings

Table 2: Absolute maximum ratings

Parameter	Condition	Min	Max	Units
Voltage at Supply Pin	V_{DD} Pin	-0.3	4.25	V
	V_{DDIO} Pin	-0.3	4.25	V
Voltage at any Logic Pad	Non-Supply Pin	-0.3	V_{DDIO}+0.3	V
Passive Storage Temp. Range	≤ 65% rel. H.	-50	+150	°C
Mechanical Shock	Duration ≤ 200µs		10,000	g
	Duration ≤ 1.0ms		2,000	g
	Free fall onto hard surfaces		1.8	m
ESD	HBM, at any Pin		2	kV
	CDM		500	V

3. Block diagram

Figure 1 shows the basic building blocks of the BMA250:

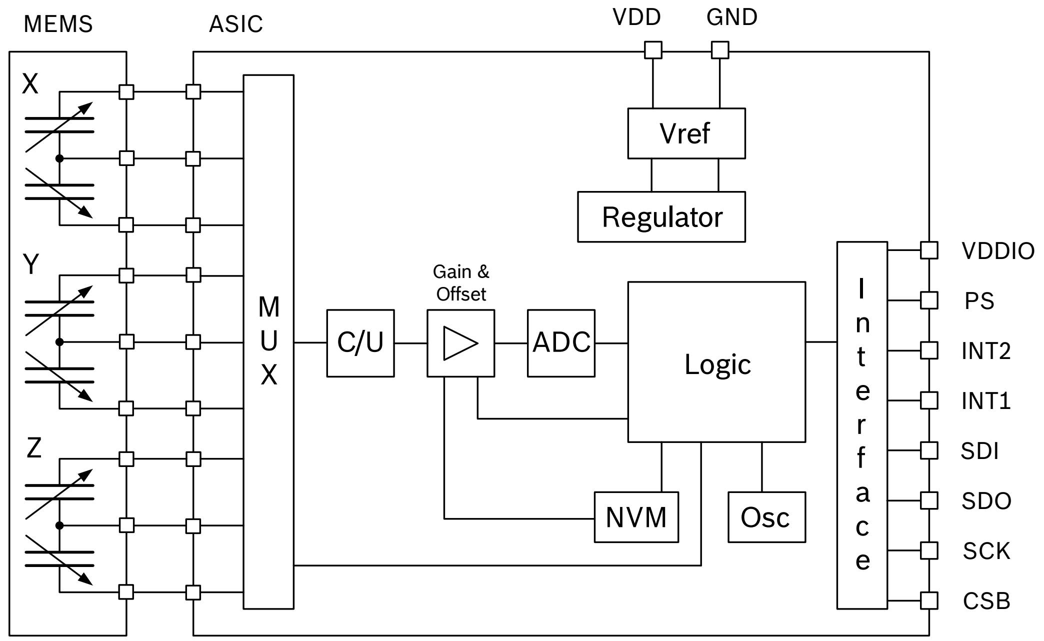


Figure 1: Block diagram of BMA250

4. Functional description

Note: Default values for registers can be found in chapter 5.

4.1 Power management

The BMA250 has two distinct power supply pins:
- V_{DD} is the main power supply for all internal analog and digital functional blocks;
- V_{DDIO} is a separate power supply pin, exclusively used for the supply of the digital interface.

There are no limitations on the voltage levels of both pins relative to each other, as long as each of them lies within its operating range. Furthermore, the device can be completely switched off (V_{DD} = 0V) while keeping the V_{DDIO} supply on (V_{DDIO} > 0V). To switch off the interface supply (V_{DDIO} = 0V) and keep the internal supply on (V_{DD} > 0V) is safe only in normal mode. If the device is in low-power mode or suspend mode while V_{DDIO} = 0V, there is a risk of excess current consumption on the V_{DD} supply (non-destructive).

It is absolutely prohibited to keep any interface at a logical high level when V_{DDIO} is switched off. Such a configuration will permanently damage the device (i.e. if V_{DDIO} = 0 → [SDI & SDO & SCK & CSB] ≠ high).

The device contains a power-on reset (POR) generator. It resets the logic part and the register values after powering-on V_{DD} and V_{DDIO}. There is no limitation on the sequence of switching on both supply voltages. In case the I²C interface shall be used, a direct electrical connection between V_{DDIO} supply and the PS pin is needed in order to ensure reliable protocol selection (see section 4.2 Operational modes).

4.2 Operational modes

Depending on the configuration the BMA250 is able to operate in two different operational modes:

- General mode: The device is acting as a slave on a digital interface (SPI or I²C) and is controlled by the external bus master (e.g. µC). The master gets measurement data and status information from the device through the digital interface. In particular, the master can configure the interrupt controller and read out the interrupt status registers. Moreover, it can freely configure and use the two interrupt pins (INT1, INT2). Several interrupts may be enabled in parallel.

- Dedicated mode: The dedicated mode allows the sensor to be operated as a stand-alone device in a simple µC-less system without abandon of the interrupt functionality. No digital interface is needed and, as a consequence, no measurement data can be read from the device. Instead of the digital interface the internal interrupt engine with its default setting is used. The interrupt status is mapped onto dedicated output pins. One out of three different sub-modes can be chosen: A) orientation recognition, B) tap sensing or C) slope (any-motion) detection. Only one interrupt at a time can be assigned.

The selection of the operational mode is done during start-up or reset by the state of the PS pin. If PS is floating, the dedicated mode is selected. A defined digital state selects the general mode. All pads are in input mode (no output driver active) during the start-up sequence until the operational mode and, in case of the general mode, the interface type is selected. The start-up sequence is run after power-up and after reset.

Figure 2 illustrates the selection of the different operational modes:

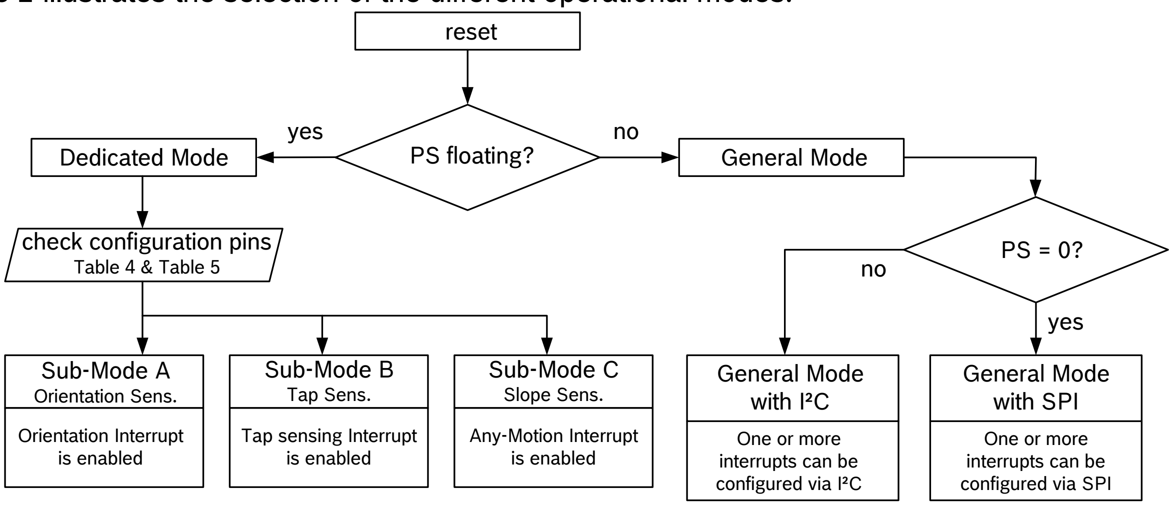


Figure 2: Operational mode selection

4.2.1 General mode

A defined digital state at the PS pin selects the general mode. Its polarity determines the kind of interface to be used:

- PS = GND enables the digital SPI interface
- PS = V_{DDIO} enables the digital I²C interface
- PS = float enables the dedicated mode

4.2.2 Dedicated mode (µC-less or stand-alone mode)

The dedicated mode operates with pre-defined settings of the interrupt engine in order to generate the motion-triggered interrupt-signals, i.e. bandwidth, sleep time, low-power mode, threshold, and hysteresis are use case optimized. Nevertheless some minor configurations can be selected by the user. The dedicated mode is entered if the device is connected according to table 3. During the start-up / power on sequence the PS pin (#11) must float.

Table 3: Entering and operating dedicated mode

VDDIO Pin#3	**NC** Pin#4	**VDD** Pin#7	**GNDIO** Pin#8	**GND** Pin#9	**PS** Pin#11
V_{DDIO}	NC	V_{DD}	GND	GND	**float**

Depending on the configuration of the other device pins according to table 4 the corresponding sub-mode of the dedicated mode is entered. In table 4 and table 5 the unshaded entries represent necessary input values for the corresponding sub-mode selection while the shaded entries represent corresponding output parameters of the events to be detected.

Table 4: Sub-mode selection and specific outputs of the dedicated mode

Sub-Mode	**SDO** Pin#1	**SDx** Pin#2	**INT1** Pin#5	**INT2** Pin#6	**CSB** Pin#10	**SCx** Pin#12
Orientation	output orient1-detect	output orient0-detect	output orient2-detect	output flat-detect	select orient sleep	GND
Tap	output double-detect	output single-detect	GND	select tap type	select tap sleep	V_{DD}
Slope	GND	output motion-detect	V_{DD}	GND	select slope sleep	V_{DD}

Table 5 contains state and description details of the parameters introduced in table 4. Unshaded entries represent input values to be set, shaded entries represent output parameters to be detected.

Table 5: Description of the parameters of table 4

Sub-Mode	Parameter see Table 4	State	Description
Orientation BW = 62.5 Hz	output orient0-detect	low	"upright" for portrait / "left" for landscape
		high	"upside-down" for portrait / "right" for landscape
	output orient1-detect	low	portrait
		high	landscape
	output orient2-detect	low	z-axis upward looking i.e. $\|\theta\| < 90°$ (Fig. 8)
		high	z-axis downward looking i.e. $\|\theta\| > 90°$ (Fig. 8)
	output flat-detect	low	non flat i.e. $\|\theta\| > 19,5°$ (Fig. 8)
		high	flat i.e. $\|\theta\| < 19,5°$ (Fig. 8)
	select orient sleep	GND	Low-Power mode enabled, sleep time = 100ms
		V_{DD}	Low-Power mode enabled, sleep time = 1s
Tap BW = 1k Hz	output double-detect	low	currently no Double-Tap event
		high	Double-Tap event detected
	output single-detect	low	currently no single-tap event
		high	Single-Tap event detected
	select tap type	GND	Single-Tap detection enabled
		V_{DD}	Double-Tap detection enabled
	select tap sleep	GND	Low-Power Mode disabled
		V_{DD}	Low-Power Mode enabled, sleep time = 10ms
Slope BW = 125 Hz	output motion-detect	low	currently no Any-Motion event
		high	Any-Motion event detected
	select slope sleep	GND	Low-Power mode enabled, sleep time = 50ms
		V_{DD}	Low-Power mode enabled, sleep time = 1s

low = GND, high = V_{DDIO}

For more details, refer to chapter 4.3 Power modes and 4.8 Interrupt Controller

- Orientation recognition sub mode → refer to chapter 4.8.7
- Tap sensing sub mode → refer to chapter 4.8.6
- Any-motion (slope) detection) sub mode → refer to chapter 4.8.5

4.3 Power modes

The BMA250 has three different power modes. Besides normal mode, which represents the fully operational state of the device, there are two special energy saving modes: low-power mode and suspend mode.

The possible transitions between the power modes are illustrated in figure 3:

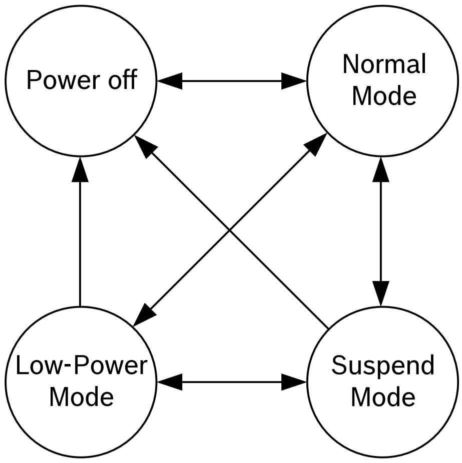


Figure 3: Power mode transition diagram

In normal mode, all parts of the electronic circuit are held powered-up and data acquisition is performed continuously.

In contrast to this, in suspend mode the whole analog part, oscillators included, is powered down. No data acquisition is performed, the only supported operations are reading registers (latest acceleration data are kept) and writing to the *(0x11) suspend* bit or *(0x14) softreset* register. Suspend mode is entered (left) by writing ´1´ (´0´) to the *(0x11) suspend* bit.

In low-power mode, the device is periodically switching between a sleep phase and a wake-up phase. The wake-up phase essentially corresponds to operation in normal mode with complete power-up of the circuitry. During the sleep phase the analog part except the oscillators is powered down. Low-power mode is entered (left) by writing ´1´ (´0´) to the *(0x11) lowpower_en* bit.

During the wake-up phase the number of samples required by any enabled interrupt is processed. If an interrupt is detected, the device stays in the wake-up phase as long as the interrupt condition endures (non-latched interrupt), or until the latch time expires (temporary interrupt), or until the interrupt is reset (latched interrupt). If no interrupt is detected, the device enters the sleep phase.

The duration of the sleep phase is set by the *(0x11) sleep_dur* bits as shown in the following table:

Table 6: Sleep phase duration settings

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

The current consumption of the BMA250 can be calculated according to this formula:

$$I_{DDlp} \approx \frac{t_{sleep} \cdot I_{DDsm} + t_{active} \cdot I_{DD}}{t_{sleep} + t_{active}}.$$

When making an estimation about the length of the wake-up phase t_{active}, the wake-up time, t_{w_up}, has to be considered. Therefore, $t_{active} = t_{ut} + t_{w\_up}$, where t_{ut} is given in table 8. During the wake-up phase all analog modules are held powered-up, while during the sleep phase most analog modules are powered down. As a consequence, a wake-up time of less than 1ms (typ. value 0.8ms) is needed to settle the analog modules in order to get reliable acceleration data.

Table 7 gives an overview of the resulting average supply currents I_{DDlpe} for the different sleep phase durations and a selected bandwidth of 1000Hz, assuming no interrupt is active and thus only one sample per wake-up phase is taken:

Table 7: Average current consumption in low-power mode

Sleep phase duration	Average current consumption
0.5ms	100.5 µA
1ms	78.8 µA
2ms	55.0 µA
4ms	34.5 µA
6ms	25.2 µA
10ms	16.4 µA
25ms	7.4 µA
50ms	4.0 µA
100ms	2.3 µA
500ms	0.9 µA
1s	0.7 µA

4.4 Sensor data

4.4.1 Acceleration data

The width of acceleration data is 10 bits given in two´s complement representation. The 10 bits for each axis are split into an MSB upper part (one byte containing bits 9 to 2) and an LSB lower part (one byte containing bits 1 and 0 of acceleration and a *(0x02, 0x04, 0x06) new_data* flag). Reading the acceleration data registers shall always start with the LSB part. The content of an MSB register is updated by reading the corresponding LSB register (shadowing procedure). The shadowing procedure can be disabled (enabled) by writing ´1´ (´0´) to the bit *shadow_dis*. With disabled shadowing, the content of both MSB and LSB registers is updated by a new value immediately. Unused bits of the LSB registers are fixed to 0. The *(0x02, 0x04, 0x06) new_data* flag of each LSB register is set if the data registers are updated, it is reset if either the corresponding MSB or LSB part is read.

Two different streams of acceleration data are available, unfiltered and filtered. The unfiltered data is sampled with 2kHz. The sampling rate of the filtered data depends on the selected filter bandwidth; it is twice the bandwidth. Which kind of data is stored in the acceleration data registers depends on bit *(0x13) data_high_bw*. If *(0x13) data_high_bw* is ´0´ (´1´), then filtered (unfiltered) data is stored in the registers. Both data streams are separately offset-compensated. Both kinds of data can be processed by the interrupt controller.

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

Table 8: Bandwidth configuration

bw	Bandwidth	Update Time t_{ut}
00xxx	*)	-
01000	7.81Hz	64ms
01001	15.63Hz	32ms
01010	31.25Hz	16ms
01011	62.5Hz	8ms
01100	125Hz	4ms
01101	250Hz	2ms
01110	500Hz	1ms
01111	1000Hz	0.5ms
1xxxx	*)	-

*) Note: Settings 00xxx result in a bandwidth of 7.81 Hz; settings 1xxxx result in a bandwidth of 1000 Hz. It is recommended to actively use the range from ´01000b´ to ´01111b´ only in order to be compatible with future products.

The BMA250 supports four different acceleration measurement ranges. A measurement range is selected by setting the *(0x0F)* range bits as follows:

Table 9: Range selection

Range	Acceleration measurement range	Resolution
0011	±2g	3.91mg/LSB
0101	±4g	7.81mg/LSB
1000	±8g	15.62mg/LSB
1100	±16g	31.25mg/LSB
others	reserved	-

4.4.2 Temperature data

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

The slope of the temperature sensor is 0.5K/LSB, its center temperature is 24°C [*(0x08) temp* = 0x00]. Therefore, the typical temperature measurement range is -40°C up to 87.5°C.

4.5 Self-test

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

The self-test is activated individually for each axis by writing the proper value to the *(0x32) self_test_axis* bits (´01b´ for x-axis, ´10b´ for y-axis, ´11b´ for z-axis, ´00b´ to deactivate self-test). It is possible to control the direction of the deflection through bit *(0x32) self_test_sign*. The excitation occurs in positive (negative) direction if *(0x32) self_test_sign* = ´0b´ (´1b´).

In order to ensure a proper interpretation of the self-test signal it is recommended to perform the self-test for both (positive and negative) directions and then to calculate the difference of the resulting acceleration values. Table 10 shows the minimum differences for each axis. The actually measured signal differences can be significantly larger.

Table 10: Self-test difference values

	x-axis signal	y-axis signal	z-axis signal
resulting minimum difference signal	+0.8 g	+0.8 g	+0.4 g

It is recommended to perform a reset of the device after self-test. 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 600 μs, enable desired interrupts.

4.6 Offset compensation

Offsets in measured signals can have several causes but they are always unwanted and disturbing in many cases. Therefore, the BMA250 offers an advanced set of four digital offset compensation methods which are closely matched to each other. These are slow, fast, and manual compensation, and inline calibration.

The compensation is performed for unfiltered and filtered data independently. It is done by adding a compensation value to the acceleration data coming from the ADC. The result of this computation is saturated if necessary to prevent any overflow errors (the smallest or biggest possible value is set, depending on the sign). However, the public registers used to read and write compensation values have only a width of 8 bits.

An overview of the offset compensation principle is given in figure 4:

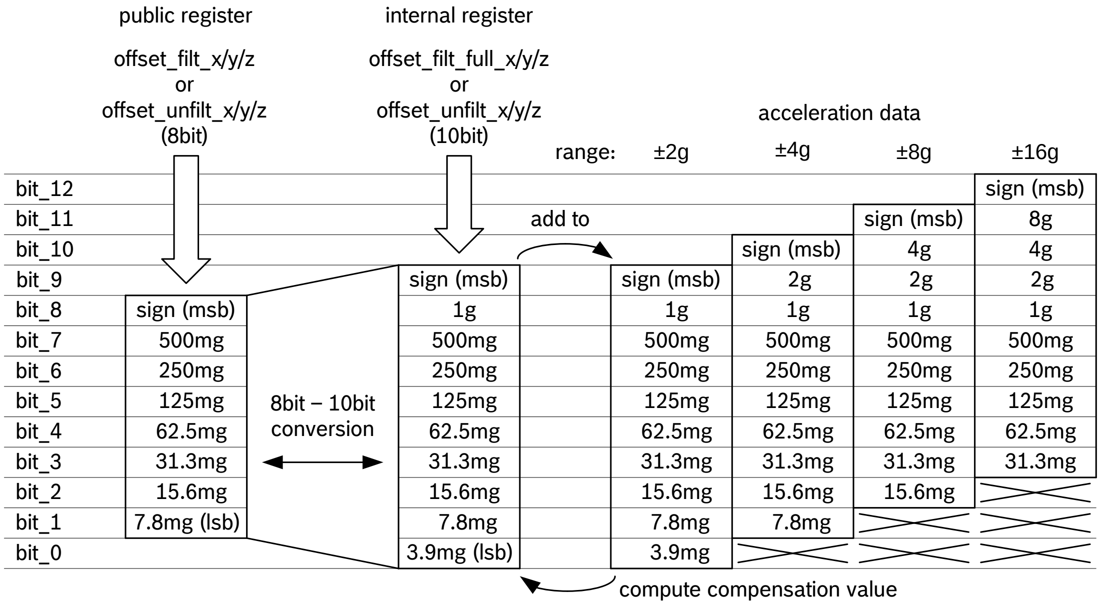


Figure 4: Principle of offset compensation

The meaning of both public and internal registers is the same for all acceleration measurement ranges. Therefore, with measurement ranges other than ±2g, one or more lower significant bits of the internal registers are lost when added to an acceleration value, or are set to zero when the internal compensation value is computed. If a compensation value is too small or too big to fit into the corresponding internal register, it is saturated to prevent an overflow error.

In a similar way the conversion of the internal register value to the public register value (10bit to 8bit) uses saturation.

Summarized, in dependence to the measurement range which has been set, the compensation value, which has been written into the public register will correct the data output according to figure 4.

e.g. ±2g range:
public register = 00000001b → add to acceleration data = ±7.8mg = +2LSB
public register = 00000010b → add to acceleration data = +15.6mg = +4LSB
public register = 00000101b → add to acceleration data = +39.1mg = +10LSB

The public registers are image registers of EEPROM registers. With each image update (see section 4.7 Non-volatile memory for details) the contents of the non-volatile EEPROM registers are written to the public registers. At any time the public register can be over-written by the user. After changing the contents of the public registers by either an image update or manually, all 8bit values are widened to 10bit values and stored in the corresponding internal registers. In the opposite direction, if the value of an internal register changes due to the computation performed by a compensation algorithm, it is converted to an 8bit value and stored in the public register.

For slow and fast offset compensation, the compensation target can be chosen by setting the bits *(0x37) offset_target_x*, *(0x37) offset_target_y*, and *(0x37) offset_target_z* according to table 11:

Table 11: Offset target settings

(0x37) offset_target_x/y/z	**Target value**
00b	0g
01b	+1g
10b	-1g
11b	0g

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

4.6.1 Slow compensation

Slow compensation is a quasi-continuous process which regulates the acceleration value of each axis towards the target value by comparing the current value with the target and adding or subtracting a fixed value depending on the comparison.

The algorithm in detail: If an acceleration value is larger (smaller) than the target value *(0x37) offset_target_x/y/z* for a number of samples (given by the parameter Offset Period see table 12), the internal offset compensation value *(0x38, 0x039, 0x3A) offset_filt_x/y/z* or *(0x3B, 0x03C, 0x3D) offset_unfilt_x/y/z* is decremented (incremented) by 4 LSB.

The public registers *(0x38, 0x039, 0x3A) offset_filt_x/y/z* and *(0x3B, 0x03C, 0x3D) offset_unfilt_x/y/z* are not used for the computations but they are updated with the contents of the internal registers (using saturation if necessary) and can be read by the user.

The compensation period offset_period is set by the *(0x37) cut_off* bit as represented in table 12:

Table 12: Compensation period settings

(0x37) cut_off	**Offset Period**
0b	8
1b	16

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

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

4.6.2 Fast compensation

Fast compensation is a one-shot process by which the compensation value is set in such a way that when added to the raw acceleration, the resulting acceleration value of each axis equals the target value.

The algorithm in detail: An average of 16 consecutive acceleration values is computed and the difference between target value and computed value is written to *(0x38, 0x39, 0x3A) offset_filt_x/y/z* or *(0x3B, 0x3C, 0x3D) offset_unfilt_x/y/z* The public registers *(0x38, 0x39, 0x3A) offset_filt_x/y/z* and *(0x3B, 0x3C, 0x3D) offset_unfilt_x/y/z* are updated with the contents of the internal registers (using saturation if necessary) and can be read by the user.

Fast compensation is triggered for each axis individually by setting the *(0x36) cal_trigger* bits as shown in table 13:

Table 13: Fast compensation axis selection

(0x36) cal_trigger	**Selected Axis**
00b	none
01b	x
10b	y
11b	z

The register *(0x36) cal_trigger* keeps its non-zero value while the fast compensation procedure is running. Slow compensation is blocked as long as fast compensation endures. Bit *(0x36) cal_rdy* is ´0´ when *(0x36) cal_trigger* is not ´00´.

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

4.6.3 Manual compensation

As explained above, the contents of the public compensation registers *(0x38, 0x39, 0x3A) offset_filt_x/y/z* and *(0x3B, 0x3C, 0x3D) offset_unfilt_x/y/z* can be set manually via the digital interface. It is recommended to write into these registers immediately after a new data interrupt in order not to disturb running offset computations.

Writing to the offset compensation registers is not allowed if slow compensation is enabled or if the fast compensation procedure is running.

4.6.4 Inline calibration

For a given application, it is often desirable to calibrate the offset once and to store the compensation values permanently. This can be achieved by using one of the aforementioned offset compensation methods to determine the proper compensation values and then storing these values permanently in the non-volatile memory (EEPROM). See section 4.7 Non-volatile memory for details of the storing procedure.

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

4.7 Non-volatile memory

The entire memory of the BMA250 consists of three different kinds of registers: hard-wired, volatile, and non-volatile. Non-volatile memory is implemented as EEPROM. Part of it can be both read and written by the user. Access to non-volatile memory is only possible through (volatile) image registers.

Altogether, there are eight registers (bytes) of EEPROM which are accessible by the customer. The addresses of the image registers range from 0x38 to 0x3F. While the addresses up to 0x3D are used for offset compensation (see 4.6 Offset Compensation), addresses 0x3E and 0x3F are general purpose registers not linked to any sensor-specific functionality.

The content of the EEPROM is loaded to the image registers after a reset (either POR or softreset) or after a user request which is performed by writing ´1´ to bit *(0x33) nvm_load*. As long as the image update is not yet complete, bit *(0x33) nvm_load* is ´1´, otherwise it is ´0´.

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

Writing to the EEPROM is a three-step procedure:
1. Write the new contents to the image registers.
2. Write ´1´ to bit *(0x33) nvm_prog_mode* in order to unlock the EEPROM.
3. Write ´1´ to bit *(0x33) nvm_prog_trig* and keep ´1´ in bit *(0x33) nvm_prog_mode* in order to trigger the write process.

Writing to the EEPROM always renews the entire EEPROM contents. It is possible to check the write status by reading bit *(0x33) nvm_rdy*. While *(0x33) nvm_rdy* = ´0´, the write process is still enduring; if *(0x33) nvm_rdy* = ´1´, then writing is completed. As long as the write process is ongoing, no power mode change and no image update is allowed. It is forbidden to write to the EEPROM while the image update is running, in low-power mode, and in suspend mode.

4.8 Interrupt controller

Seven interrupt engines are integrated in the BMA250. Each interrupt can be independently enabled and configured. If the condition of an enabled interrupt is fulfilled, the corresponding status bit is set to ´1´ and the selected interrupt pin is activated. There are two interrupt pins, INT1 and INT2; interrupts can be freely mapped to any of these pins. The pin state is a logic ´or´ combination of all mapped interrupts.

The interrupt status registers are updated together with writing new data into the acceleration data registers. If an interrupt is disabled, all active status bits and pins are immediately reset.

All time constants are based upon the typical frequency of the internal oscillator. This is reflected by the bandwidths (bw) as specified in table 1.

4.8.1 General features

An interrupt is cleared depending on the selected interrupt mode, which is common to all interrupts. There are three different interrupt modes: non-latched, latched, and temporary. The mode is selected by the *(0x21) latch_int* bits according to table 14

Table 14: Interrupt mode selection

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

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

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

In the temporary mode an asserted interrupt and selected pin are cleared after a defined period of time. The behaviour of the different interrupt modes is shown graphically in figure 5:

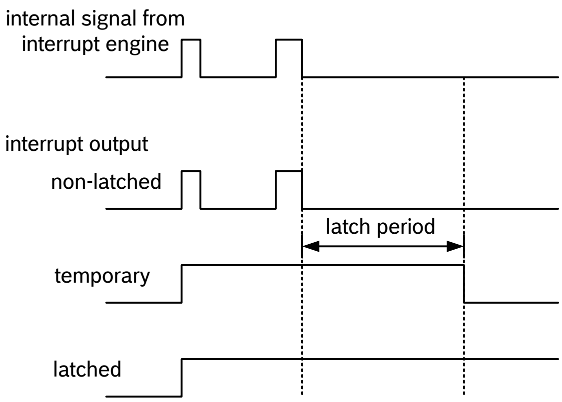


Figure 5: Interrupt modes

Several interrupt engines can use either unfiltered or filtered acceleration data as their input. For these interrupts, the source can be selected with the respective *(0x1E) int_src_...* bits, in details these are *(0x1E) int_src_data*, *(0x1E) int_src_tap*, *(0x1E) int_src_slope*, *(0x1E) int_src_high*, and *(0x1E) int_src_low*. Setting the respective bits to ´0´ (´1´) selects filtered (unfiltered) data as input. For the other interrupts, orientation recognition and flat detection, such a selection is not possible. They always use filtered input data.

It is strongly recommended to set interrupt parameters prior to enabling the interrupt. Changing parameters of an already enabled interrupt may cause unwanted interrupt generation and generation of a false interrupt history. A safe way to change parameters of an enabled interrupt is to keep the following sequence: disable the desired interrupt, change parameters, wait for at least 600 µs, enable the desired interrupt.

4.8.2 Mapping (inttype to INT Pin#)

The mapping of interrupts to the interrupt pins #05 or #06 is done by registers *(0x19)* to *(0x1B)*. Setting *(0x19) int1_*"inttyp" to ´1´ (´0´) maps (unmaps) "inttyp" to pin #5 (INT1), correspondingly setting *(0x1B) int2_*"inttyp" to ´1´ (´0´) maps (unmaps) "inttyp" to pin #6 (INT2).

Note: "inttyp" to be replaced with the precise notation, given in the memory map in chapter 5.

Example: For flat interrupt (int1_flat): Setting *(0x19) int1_flat* to ´1´ maps *int1_flat* to pin #5 (INT1).

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

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

If *(0x20) int1_lvl* = ´1´ (´0´) / *(0x20) int2_lvl* = ´1´ (´0´), then pin #05 (INT1) / pin #06 (INT2) is active ´1´ (´0´). In addition to that, also the electric type of the interrupt pins can be selected. By setting bits *(0x20) int1_od* / *(0x20) int2_od* to ´0´, the interrupt pin output type gets push-pull, by setting the configuration bits to ´1´, the output type gets open-drive.

Remark: Due to their use for sub-mode selection in dedicated mode, the states of both INT pins are not defined during the first 2 ms after power-up.

4.8.4 New data interrupt

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

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

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

4.8.5 Any-motion (slope) detection

Any-motion detection uses the slope between successive acceleration signals to detect changes in motion. An interrupt is generated when the slope (absolute value of acceleration difference) exceeds a preset threshold. It is cleared as soon as the slope falls below the threshold. The principle is made clear in figure 6.

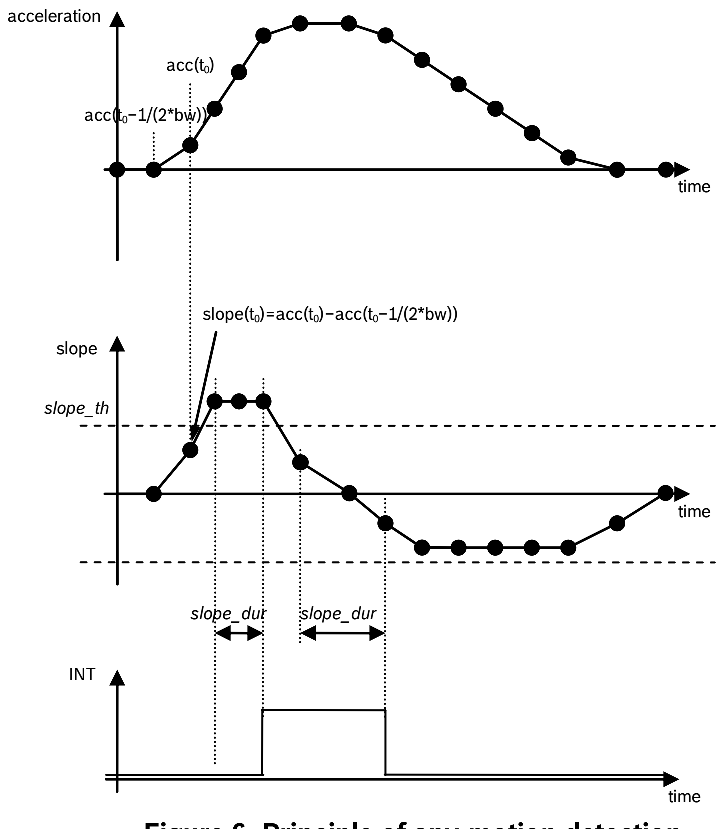


Figure 6: Principle of any-motion detection

The threshold is set with the value of register *(0x28) slope_th*. 1 LSB of *(0x28) slope_th* corresponds to 1 LSB of acceleration data. Therefore, an increment of *(0x28) slope_th* is 3.91 mg in 2g-range (7.81 mg in 4g-range, 15.6 mg in 8g-range and 31.3 mg in 16g-range). And the maximum value is 996 mg in 2g-range (1.99g in 4g-range, 3.98g in 8g-range and 7.97g in 16g-range).

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

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

4.8.5.1 Enabling (disabling) for each axis

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

4.8.5.2 Axis and sign information of any motion interrupt

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

4.8.5.3 Serial interface and dedicated wake-up mode

When serial interface is active, any-motion detection logic is enabled if any of the axis specific *(0x16) slope_en_...* register bits are set. To disable the any-motion interrupt, clear all the axis specific *(0x16) slope_en_...* bits.
In the dedicated wake-up mode (see chapter 4.2.2), all three axes are enabled for any-motion detection whether the individual axis enable bits are set or not.

4.8.6 Tap sensing

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

Only one of the tap interrupts can be enabled at the same time. Single tap interrupt is enabled (disabled) by writing ´1´ (´0´) to bit *(0x16) s_tap_en*. Double tap interrupt is enabled (disabled) by writing ´1´ (´0´) to bit *(0x16) d_tap_en*. If one tries to enable both interrupts by writing ´1´ to

(0x16) s_tap_en and *(0x16) d_tap_en*, then only *(0x16) d_tap_en* keeps the value ´1´ and the double tap interrupt is enabled.
The status of the single tap interrupt is stored in bit *(0x09) s_tap_int*, the status of the double tap interrupt is stored in bit *(0x09) d_tap_int*.

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

In figure 7 the meaning of the different timing parameters is visualized:

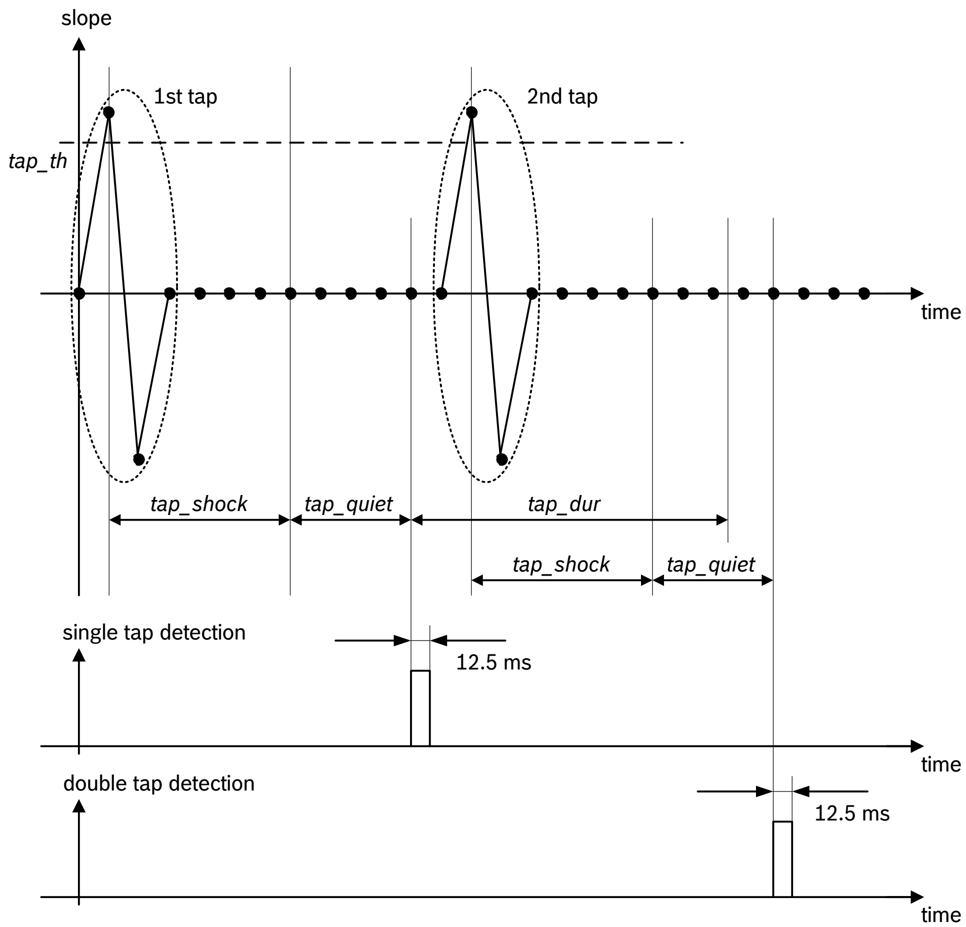


Figure 7: Timing of tap detection

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

4.8.6.1 Single tap detection

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

4.8.6.2 Double tap detection

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

4.8.6.3 Selecting the timing of tap detection

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

The length of *(0x2A) tap_dur* can be selected by setting the *(0x2A) tap_dur* bits according to table 15:

Table 15: Selection of *tap_dur*

(0x2A) tap_dur	**length of *tap_dur***
000b	50 ms
001b	100 ms
010b	150 ms
011b	200 ms
100b	250 ms
101b	375 ms
110b	500 ms
111b	700 ms

4.8.6.4 Axis and sign information of tap sensing

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

The axis which triggered the interrupt is indicated by bits *(0x0B) tap_first_x*, *(0x0B) tap_first_y*, and *(0x0B) tap_first_z*.
The bit corresponding to the triggering axis contains a ´1´ while the other bits hold a ´0´. These bits are cleared together with clearing the interrupt status.

4.8.6.5 Tap sensing in low power mode

In low-power mode, a limited number of samples is processed after wake-up to decide whether an interrupt condition is fulfilled. The number of samples is selected by bits *(0x2B) tap_samp* according to table 16.

Table 16: Meaning of *(0x2B) tap_samp*

(0x2B) tap_samp	**Number of Samples**
00b	2
01b	4
10b	8
11b	16

4.8.7 Orientation recognition

The orientation recognition feature informs on an orientation change of the sensor with respect to the gravitational field vector ‘g’. The measured acceleration vector components with respect to the gravitational field are defined as shown in figure 8.

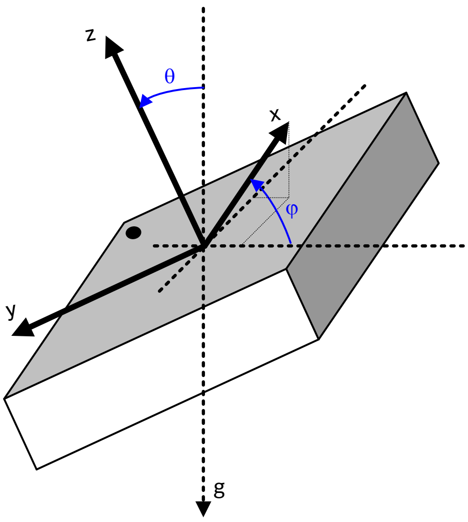


Figure 8: Definition of vector components

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

$$acc\_x = 1g \cdot \sin\theta \cdot \cos\varphi$$
$$acc\_y = -1g \cdot \sin\theta \cdot \sin\varphi$$
$$acc\_z = 1g \cdot \cos\theta$$
$$\rightarrow acc\_y/acc\_x = -\tan\varphi$$

Depending on the magnitudes of the acceleration vectors the orientation of the device in the space is determined and stored in the three *(0x0C) orient* bits. These bits may not be reset in the sleep phase of low-power mode. There are three orientation calculation modes with different thresholds for switching between different orientations: symmetrical, high-asymmetrical, and low-asymmetrical. The mode is selected by setting the *(0x2C) orient_mode* bits as given in table 17.

Table 17: Orientation mode settings

(0x2C) orient_mode	**Orientation Mode**
00b	symmetrical
01b	high-asymmetrical
10b	low-asymmetrical
11b	symmetrical

For each orientation mode the *(0x0C) orient* bits have a different meaning as shown in table 18 to table 20:

Table 18: Meaning of the *(0x0C) orient* bits in symmetrical mode

(0x0C) orient	**Name**	**Angle**	**Condition**
x00	portrait upright	315° < φ < 45°	|acc_y| < |acc_x| - 'hyst' and acc_x - 'hyst'' ≥ 0
x01	portrait upside down	135° < φ < 225°	|acc_y| < |acc_x| - 'hyst' and acc_x + 'hyst' < 0
x10	landscape left	45° < φ < 135°	|acc_y| ≥ |acc_x| + 'hyst' and acc_y < 0
x11	landscape right	225° < φ < 315°	|acc_y| ≥ |acc_x| + 'hyst' and acc_y ≥ 0

Table 19: Meaning of the *(0x0C) orient* bits in high-asymmetrical mode

(0x0C) orient	**Name**	**Angle**	**Condition**
x00	portrait upright	297° < φ < 63°	|acc_y| < 2·|acc_x| - 'hyst' and acc_x - 'hyst' ≥ 0
x01	portrait upside down	117° < φ < 243°	|acc_y| < 2·|acc_x| - 'hyst' and acc_x + 'hyst' < 0
x10	landscape left	63° < φ < 117°	|acc_y| ≥ 2·|acc_x| + 'hyst' and acc_y < 0
x11	landscape right	243° < φ < 297°	|acc_y| ≥ 2·|acc_x| + 'hyst' and acc_y ≥ 0

Table 20: Meaning of the *(0x0C) orient* bits in low-asymmetrical mode

(0x0C) orient	Name	Angle	Condition
x00	portrait upright	333° < φ < 27°	|acc_y| < 0.5·|acc_x| - 'hyst' and acc_x - 'hyst' ≥ 0
x01	portrait upside down	153° < φ < 207°	|acc_y| < 0.5·|acc_x| - 'hyst' and acc_x + 'hyst' < 0
x10	landscape left	27° < φ < 153°	|acc_y| ≥ 0.5·|acc_x| + 'hyst' and acc_y < 0
x11	landscape right	207° < φ < 333°	|acc_y| ≥ 0.5·|acc_x| + 'hyst' and acc_y ≥ 0

In the preceding tables, the parameter 'hyst' stands for a hysteresis, which can be selected by setting the *(0x0C) orient_hyst* bits. 1 LSB of *(0x0C) orient_hyst* always corresponds to 62.5 mg, in 2g-range, 125 mg in 4g-range, 250 mg in 8g-range and 500 mg in 16g-range.. It is important to note that by using a hysteresis ≠ 0 the actual switching angles become different from the angles given in the tables since there is an overlap between the different orientations.

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

Figure 9 shows the typical switching conditions between the four different orientations for the symmetrical mode (i.e. without hysteresis):

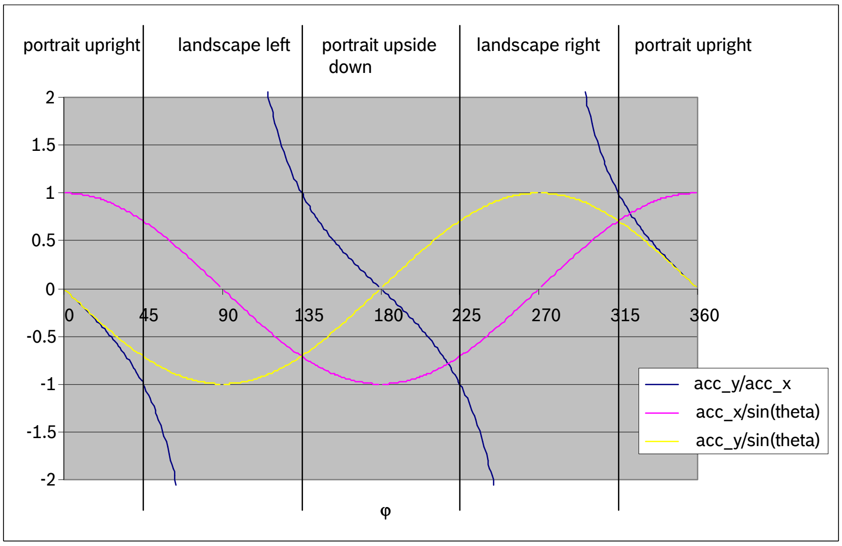


Figure 9: Typical orientation switching conditions w/o hysteresis

The orientation interrupt is enabled (disabled) by writing ´1´ (´0´) to bit *(0x16) orient_en*. The interrupt is generated if the value of *(0x0C) orient* has changed. It is automatically cleared after one stable period of the *(0x0C) orient* value. The interrupt status is stored in the *(0x09) orient_int* bit.

If temporary or latched interrupt mode is used, after the generation of the interrupt the changed *(0x0C) orient* value is kept fixed as long as the interrupt persists (e. g. until the latch time expires or the interrupt is reset). After clearing the interrupt, the *(0x0C) orient* is only updated with the next following value change (i.e. with the next occurring interrupt). In order to ensure the continuous availability of up-to-date orientation data it is therefore optimal to use the non-latched interrupt. It is strongly advised against using latched interrupt mode or temporary interrupt mode with latch times above 50 ms for orient recognition.

4.8.7.1 Orientation blocking

The change of the *(0x0C) orient* value and – as a consequence – the generation of the interrupt can be blocked according to conditions selected by setting the value of the *(0x2C) orient_blocking* bits as described by table 21.

Table 21: Blocking conditions for orientation recognition

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

The theta blocking is defined by the following inequality:

$$|\tan\theta| < \frac{\sqrt{blocking\_theta}}{8}.$$

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

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

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

4.8.8 Flat detection

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

The condition for the device to be in the flat position is

$$|\tan\theta| < \frac{\sqrt{parameter\_theta}}{8}.$$

Like *blocking_theta*, used with orientation recognition, the *parameter_theta* stands for a user-defined setting. In this case the content of the *(0x2E) flat_theta* bits. The possible flat angles also range from 0° to 44.8°. To ensure proper operation, *parameter_theta* has to be less than or equal to *blocking_theta*.

The flat interrupt is enabled (disabled) by writing ´1´ (´0´) to bit *(0x16) flat_en*. The flat interrupt is generated if the flat value has changed and the new value is stable for at least the time given by the *(0x2F) flat_hold_time* bits. The flat value is stored in the *(0x0C) flat* bit if the interrupt is enabled. This value is ´1´ if the device is in the flat position, it is ´0´ otherwise. The content of the *(0x0C) flat* bit is changed only if the interrupt is generated. The interrupt is automatically cleared after one sample period. Its status is stored in the *(0x09) flat_int* bit.

If temporary or latched interrupt mode is used, after the generation of the interrupt the changed *(0x0C) flat* value is kept fixed as long as the interrupt persists (e. g. until the latch time expires or the interrupt is reset). After clearing the interrupt, the *(0x0C) flat* value is only updated with the next following value change (i.e. with the next occurring interrupt).

The meaning of the *(0x2F) flat_hold_time* bits can be seen from table 22.

Table 22: Meaning of flat_hold_time

(0x2F) flat_hold_time	**Time**
00b	0
01b	512 ms
10b	1024 ms
11b	2048 ms

4.8.9 Low-g interrupt

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

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

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

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

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

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

4.8.10 High-g interrupt

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

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

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

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

4.8.10.1 Axis and sign information of high-g interrupt

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

5. Register description

5.1 General remarks

The entire communication with the device is performed by reading from and writing to registers (exception: dedicated mode, see chapter 4.2.2). Registers have a width of 8 bits; they are mapped to a common space of 64 addresses from *(0x00)* up to *(0x3F)*. Within the used range there are several registers which are either completely or partially marked as ‘reserved’. Any reserved bit is ignored when it is written and no specific value is guaranteed when read. It is recommended not to use registers at all which are completely marked as ‘reserved’. Furthermore it is recommended to mask out (logical *and* with zero) reserved bits of registers which are partially marked as reserved.

Registers with addresses from *(0x00)* up to *(0x0E)* are read-only. Any attempt to write to these registers is ignored. There are bits within some registers that connected with an action to be done and, therefore, are intended for write-only access, e. g. *(0x21) reset_int* or the entire *(0x14) softreset* register. Such bits always give ´0´ when read.

5.2 Register map

Register Address	Default Value	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x3F	0x00	*reserved*							
0x3E	0x00	*reserved*							
0x3D	0x00	offset_unfilt_z<7:0>							
0x3C	0x00	offset_unfilt_y<7:0>							
0x3B	0x00	offset_unfilt_x<7:0>							
0x3A	0x00	offset_filt_z<7:0>							
0x39	0x00	offset_filt_y<7:0>							
0x38	0x00	offset_filt_x<7:0>							
0x37	0x00	*reserved*	offset_target_z<1:0>		offset_target_y<1:0>		offset_target_x<1:0>		cut_off
0x36	0x10	offset_reset	cal_trigger<1:0>		cal_rdy	*reserved*	hp_z_en	hp_y_en	hp_x_en
0x35	0x00	*reserved*							
0x34	0x00	*reserved*					i2c_wdt_en	i2c_wdt_sel	spi3
0x33	0x04	*reserved*				nvm_load	nvm_rdy	nvm_prog_trig	nvm_prog_mode
0x32	0x70	*reserved*					self_test_sign	self_test_axis	
0x31	0x01	*reserved*							
0x30	0x00	*reserved*							
0x2F	0x10	*reserved*		flat_hold_time<1:0>		*reserved*			
0x2E	0x08	*reserved*		flat_theta<5:0>					
0x2D	0x08	*reserved*		orient_theta<5:0>					
0x2C	0x18	*reserved*	orient_hyst<2:0>			orient_blocking<1:0>		orient_mode<1:0>	
0x2B	0x0A	tap_samp<1:0>		*reserved*	tap_th<4:0>				
0x2A	0x04	tap_quiet	tap_shock	*reserved*			tap_dur<2:0>		
0x29	0x00	*reserved*							
0x28	0x14	slope_th<7:0>							
0x27	0x00	*reserved*						slope_dur<1:0>	
0x26	0xC0	high_th<7:0>							
0x25	0x0F	high_dur<7:0>							
0x24	0x81	high_hy<1:0>		*reserved*			low_mode	low_hy<1:0>	
0x23	0x30	low_th<7:0>							
0x22	0x09	low_dur<7:0>							
0x21	0x00	reset_int	*reserved*			latch_int<3:0>			
0x20	0x05	*reserved*				int2_od	int2_lvl	int1_od	int1_lvl
0x1F	0x00	*reserved*							
0x1E	0x00	*reserved*		int_src_data	int_src_tap	*reserved*	int_src_slope	int_src_high	int_src_low
0x1D	0x00	*reserved*							
0x1C	0x00	*reserved*							
0x1B	0x00	int2_flat	int2_orient	int2_s_tap	int2_d_tap	*reserved*	int2_slope	int2_high	int2_low
0x1A	0x00	int2_data	*reserved*						int1_data
0x19	0x00	int1_flat	int1_orient	int1_s_tap	int1_d_tap	*reserved*	int1_slope	int1_high	int1_low
0x18	0x00	*reserved*							
0x17	0x00	*reserved*			data_en	low_en	high_en_z	high_en_y	high_en_x
0x16	0x00	flat_en	orient_en	s_tap_en	d_tap_en	*reserved*	slope_en_z	slope_en_y	slope_en_x
0x15	0x00	*reserved*							
0x14	0x00	softreset							
0x13	0x00	data_high_bw	shadow_dis	*reserved*					
0x12	0x00	*reserved*							
0x11	0x00	suspend	lowpower_en	*reserved*	sleep_dur<3:0>				*reserved*
0x10	0x1F	*reserved*			bw<4:0>				
0x0F	0x03	*reserved*				range<3:0>			
0x0E	0x00	*reserved*							
0x0D	0x00	*reserved*							
0x0C	0x00	flat	orient[2:0]			high_sign	high_first_z	high_first_y	high_first_x
0x0B	0x00	tap_sign	tap_first_z	tap_first_y	tap_first_x	slope_sign	slope_first_z	slope_first_y	slope_first_x
0x0A	0x00	data_int	*reserved*						
0x09	0x00	flat_int	orient_int	s_tap_int	d_tap_int	*reserved*	slope_int	high_int	low_int
0x08	0x00	temp<7:0>							
0x07	0x00	acc_z_msb<9:2>							
0x06	0x00	acc_z_lsb<1:0>		0					new_data_z
0x05	0x00	acc_y_msb<9:2>							
0x04	0x00	acc_y_lsb<1:0>		0					new_data_y
0x03	0x00	acc_x_msb<9:2>							
0x02	0x00	acc_x_lsb<1:0>		0					new_data_x
0x01	n/a	*reserved*							
0x00	0x03	Chip ID							

blue	w/r
green	write only
yellow	read only
grey	reserved

5.3 Chip ID

Register *(0x00)* *Chip ID* contains the chip identification number.

Table 23: Chip identification number, register *(0x00)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	0	0	1	1

Register *(0x01)* *is reserved*

5.4 Acceleration data

Register *(0x02)* contains the LSB part of x-axis acceleration data and the new data flag for the x-axis.

Table 24: LSB part of x-axis acceleration, register *(0x02)*

(0x02) **Bit**	**Name**	**Description**
Bit 7	*acc_x_lsb <1>*	Bit 1 of x-axis acceleration data
Bit 6	*acc_x_lsb <0>*	Bit 0 of x-axis acceleration data **= x LSB**
Bit 5	-	(fixed to 0)
Bit 4	-	(fixed to 0)
Bit 3	-	(fixed to 0)
Bit 2	-	(fixed to 0)
Bit 1	-	(fixed to 0)
Bit 0	*new_data_x*	New data flag of x-axis

Register *(0x03)* contains the MSB part of x-axis acceleration data.

Table 25: MSB part of x-axis acceleration, register *(0x03)*

(0x03) **Bit**	**Name**	**Description**
Bit 7	*acc_x_msb <9>*	Bit 9 of x-axis acceleration data **= x MSB**
Bit 6	*acc_x_msb <8>*	Bit 8 of x-axis acceleration data
Bit 5	*acc_x_msb <7>*	Bit 7 of x-axis acceleration data
Bit 4	*acc_x_msb <6>*	Bit 6 of x-axis acceleration data
Bit 3	*acc_x_msb <5>*	Bit 5 of x-axis acceleration data
Bit 2	*acc_x_msb <4>*	Bit 4 of x-axis acceleration data
Bit 1	*acc_x_msb <3>*	Bit 3 of x-axis acceleration data
Bit 0	*acc_x_msb <2>*	Bit 2 of x-axis acceleration data

Register ***(0x04)*** contains the LSB part of y-axis acceleration data and the new data flag for the y-axis.

Table 26: LSB part of y-axis acceleration, register *(0x04)*

(0x04) **Bit**	**Name**	**Description**
Bit 7	*acc_y_lsb <1>*	Bit 1 of y-axis acceleration data
Bit 6	*acc_y_lsb <0>*	Bit 0 of y-axis acceleration data **= y LSB**
Bit 5	-	(fixed to 0)
Bit 4	-	(fixed to 0)
Bit 3	-	(fixed to 0)
Bit 2	-	(fixed to 0)
Bit 1	-	(fixed to 0)
Bit 0	*new_data_y*	New data flag of y-axis

Register ***(0x05)*** contains the MSB part of acceleration data for the y-axis.

Table 27: MSB part of y-axis acceleration, register *(0x05)*

(0x05) **Bit**	**Name**	**Description**
Bit 7	*acc_y_msb <9>*	Bit 9 of y-axis acceleration data **= y MSB**
Bit 6	*acc_y_msb <8>*	Bit 8 of y-axis acceleration data
Bit 5	*acc_y_msb <7>*	Bit 7 of y-axis acceleration data
Bit 4	*acc_y_msb <6>*	Bit 6 of y-axis acceleration data
Bit 3	*acc_y_msb <5>*	Bit 5 of y-axis acceleration data
Bit 2	*acc_y_msb <4>*	Bit 4 of y-axis acceleration data
Bit 1	*acc_y_msb <3>*	Bit 3 of y-axis acceleration data
Bit 0	*acc_y_msb <2>*	Bit 2 of y-axis acceleration data

Register ***(0x06)*** contains the LSB part of acceleration data and the new data flag for the z-axis.

Table 28: LSB part of y-axis acceleration, register *(0x06)*

(0x06) **Bit**	**Name**	**Description**
Bit 7	*acc_z_lsb <1>*	Bit 1 of z-axis acceleration data
Bit 6	*acc_z_lsb <0>*	Bit 0 of z-axis acceleration data **= z LSB**
Bit 5	-	(fixed to 0)
Bit 4	-	(fixed to 0)
Bit 3	-	(fixed to 0)
Bit 2	-	(fixed to 0)
Bit 1	-	(fixed to 0)
Bit 0	*new_data_z*	New data flag of z-axis

Register *(0x07)* contains the MSB part of acceleration data for the z-axis.

Table 29: MSB part of z-axis acceleration, register *(0x07)*

(0x07) **Bit**	**Name**	**Description**
Bit 7	*acc_z_msb <9>*	Bit 9 of z-axis acceleration data **= z MSB**
Bit 6	*acc_z_msb <8>*	Bit 8 of z-axis acceleration data
Bit 5	*acc_z_msb <7>*	Bit 7 of z-axis acceleration data
Bit 4	*acc_z_msb <6>*	Bit 6 of z-axis acceleration data
Bit 3	*acc_z_msb <5>*	Bit 5 of z-axis acceleration data
Bit 2	*acc_z_msb <4>*	Bit 4 of z-axis acceleration data
Bit 1	*acc_z_msb <3>*	Bit 3 of z-axis acceleration data
Bit 0	*acc_z_msb <2>*	Bit 2 of z-axis acceleration data

5.5 Temperature data

Register *(0x08) temp* contains temperature data in two's complement representation. Center temperature = 24 °C → i.e. *(0x08) temp* = 00000000b
1 LSB increment of temperature sensor is 0.5 °C (0.9 °F).

Table 30: Temperature data, register *(0x08)*

Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Temp <7>	*Temp <6>*	*Temp <5>*	*Temp <4>*	*Temp <3>*	*Temp <2>*	*Temp <1>*	*Temp <0>*

5.6 Status registers

Register *(0x09)* contains the states of several interrupts.

Table 31: Interrupt status, register *(0x09)*

(0x09) **Bit**	**Name**	**Description**
Bit 7	*flat_int*	Flat interrupt status
Bit 6	*orient_int*	Orientation interrupt status
Bit 5	*s_tap_int*	Single tap interrupt status
Bit 4	*d_tap_int*	Double tap interrupt status
Bit 3	*- reserved -*	reserved
Bit 2	*slope_int*	Slope interrupt status
Bit 1	*high_int*	High-g interrupt status
Bit 0	*low_int*	Low-g interrupt status

Register ***(0x0A)*** contains the status of the new data interrupt.

Table 32: New data status, register *(0x0A)*

(0x0A) **Bit**	**Name**	**Description**
Bit 7	*data_int*	New data interrupt status
Bit 6	*- reserved -*	reserved
Bit 5	*- reserved -*	reserved
Bit 4	*- reserved -*	reserved
Bit 3	*- reserved -*	reserved
Bit 2	*- reserved -*	reserved
Bit 1	*- reserved -*	reserved
Bit 0	*- reserved -*	reserved

Register ***(0x0B)*** contains the sign and triggering axis information for the tap and slope interrupts. Here tap interrupt comprises both single and double tap interrupt.

Table 33: Tap and slope interrupts status, register *(0x0B)*

(0x0B) **Bit**	**Name**	**Description**
Bit 7	*tap_sign*	Sign of 1^{st} tap that triggered the interrupt (´0´=positive, ´1´=negative)
Bit 6	*tap_first_z*	´1´ indicates that z-axis is triggering axis of tap interrupt
Bit 5	*tap_first_y*	´1´ indicates that y-axis is triggering axis of tap interrupt
Bit 4	*tap_first_x*	´1´ indicates that x-axis is triggering axis of tap interrupt
Bit 3	*slope_sign*	Sign of slope that triggered the interrupt (´0´=positive, ´1´=negative)
Bit 2	*slope_first_z*	´1´ indicates that z-axis is triggering axis of slope interrupt
Bit 1	*slope_first_y*	´1´ indicates that y-axis is triggering axis of slope interrupt
Bit 0	*slope_first_x*	´1´ indicates that x-axis is triggering axis of slope interrupt

Register ***(0x0C)*** contains the flat and orientation status, and the sign and triggering axis information for the high-g interrupt. Registers *(0x0D)* and *(0x0E)* are *reserved*.

Table 34: Flat and orientation Status, register *(0x0C)*

(0x0C) **Bit**	**Name**	**Description**
Bit 7	*flat*	flat detection (´1´ if flat condition is fulfilled, ´0´ otherwise)
Bit 6	*orient <2>*	orientation value of z-axis (´0´ if upward looking, ´1´ if downward looking)
Bit 5	*orient <1>*	orientation value of x-y plane (´00´=portrait upright, ´01´=portrait upside-down, ´10´=landscape left, ´11´=landscape right)
Bit 4	*orient <0>*	
Bit 3	*high_sign*	Sign of slope that triggered the interrupt (´0´=positive, ´1´=negative)
Bit 2	*high_first_z*	´1´ indicates that z-axis is triggering axis of high-g interrupt
Bit 1	*high_first_y*	´1´ indicates that y-axis is triggering axis of high-g interrupt
Bit 0	*high_first_x*	´1´ indicates that x-axis is triggering axis of high-g interrupt

Registers ***(0x0D)*** and ***(0x0E)*** are *reserved*.

5.7 g-range selection

Register *(0x0F)* contains the selection of the g-range. Proper settings for *(0x0F) range* are ´0011b´ (selects ±2g range), ´0101b´ (selects ±4g range), ´1000b´ (selects ±8g range), ´1100b´ (selects ±16g range). All other settings are irregular; if such a setting is used, ±2g range is selected. Default value of *(0x0F) range* (after reset) is ´0011b´.

Table 35: g-range, register *(0x0F)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
reserved	*reserved*	*reserved*	*reserved*	*range <3>*	*range <2>*	*range <1>*	*range <0>*

5.8 Bandwidths

Register *(0x10)* contains the selection of the bandwidth for filtered acceleration data. Settings for *(0x10) bw* are ´00xxxb´ (bandwidth = 7.81 Hz), ´01000b´ (bandwidth = 7.81 Hz), ´01001b´ (bandwidth = 15.63 Hz), ´01010b´ (bandwidth = 31.25 Hz), ´01011b´ (bandwidth = 62.5 Hz), ´01100b´ (bandwidth = 125 Hz), ´01101b´ (bandwidth = 250 Hz), ´01110b´ (bandwidth = 500 Hz), ´01111b´ (bandwidth = 1000 Hz), ´1xxxxb´ (bandwidth = 1000 Hz). Default value of *(0x10) bw* (after reset) is ´11111b´. It is recommended to actively use the range from ´01000b´ to ´01111b´ only in order to be compatible with future products.

Table 36: Bandwidths, register *(0x10)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
reserved	*reserved*	*reserved*	*bw <4>*	*bw <3>*	*bw <2>*	*bw <1>*	*bw <0>*

5.9 Power modes

Register *(0x11)* contains the configuration of the power modes. *(0x11) suspend* = ´1´ (´0´) sets (resets) suspend mode; default value of *(0x11) suspend* is ´0´.
(0x11) lowpower_en = ´1´ (´0´) sets (resets) low-power mode, default value of *(0x11) lowpower_en* is ´0´.

The settings for *(0x11) sleep_dur* are ´0000b´ to ´0101b´ (sleep phase duration = 0.5 ms), ´0110b´ (sleep phase duration = 1 ms), ´0111b´ (sleep phase duration = 2 ms), ´1000b´ (sleep phase duration = 4 ms), ´1001b´ (sleep phase duration = 6 ms), ´1010b´ (sleep phase duration = 10 ms), ´1011b´ (sleep phase duration = 25 ms), ´1100b´ (sleep phase duration = 50 ms), ´1101b´ (sleep phase duration = 100 ms), ´1110b´ (sleep phase duration = 500 ms), ´1111b´ (sleep phase duration = 1 s). Default value of *(0x11) sleep_dur* is ´0000b´.

Table 37: Power modes, register *(0x11)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
suspend	*lowpower_en*	*reserved*	*sleep_dur<3>*	*sleep_dur<2>*	*sleep_dur<1>*	*sleep_dur<0>*	*reserved*

5.10 Special control settings

Register *(0x12)* is *reserved*.

Register *(0x13)* contains settings for the configuration of the acceleration data acquisition and the data output format.

(0x13) data_high_bw = ´0´ (´1´) selects filtered (unfiltered) acceleration data to be written into the data registers (0x02) to (0x07). Default value of *(0x13) data_high_bw* is ´0´.
(0x13) shadow_dis = ´0´ (´1´) enables (disables) the shadowing procedure. Shadowing means that the MSB register is updated by reading the corresponding LSB register. Default value of *(0x13) shadow_dis* is ´0´.

Table 38: Acceleration data acquisition & data output format, register *(0x13)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
data_high_bw	*shadow_dis*	*reserved*	*reserved*	*reserved*	*reserved*	*reserved*	*reserved*

Register *(0x14)* is the *softreset* register. A user-triggered reset (softreset) of the sensor is performed after writing ´0xB6´ to the softreset register. After that reset all registers return to their default values. Reading *(0x14) softreset* returns 0x00.

Register *(0x15)* is *reserved*.

5.11 Interrupt settings

Registers *(0x16)* and *(0x17)* contain the enable bits for the interrupts. Default value of each enable bit is ´0´.

Table 39: Interrupt setting, register *(0x16)*

(0x16) Bit	Name	Description
Bit 7	*flat_en*	´1´ (´0´) enables (disables) flat interrupt
Bit 6	*orient_en*	´1´ (´0´) enables (disables) orientation interrupt
Bit 5	*s_tap_en*	´1´ (´0´) enables (disables) single tap interrupt
Bit 4	*d_tap_en*	´1´ (´0´) enables (disables) double tap interrupt
Bit 3	*- reserved -*	reserved
Bit 2	*slope_en_z*	´1´ (´0´) enables (disables) slope interrupt for z-axis
Bit 1	*slope_en_y*	´1´ (´0´) enables (disables) slope interrupt for y-axis
Bit 0	*slope_en_x*	´1´ (´0´) enables (disables) slope interrupt for x-axis

Table 40: Interrupt setting, register *(0x17)*

***(0x17)* Bit**	**Name**	**Description**
Bit 7	*- reserved -*	reserved
Bit 6	*- reserved -*	reserved
Bit 5	*- reserved -*	reserved
Bit 4	*data_en*	´1´ (´0´) enables (disables) new data interrupt
Bit 3	*low_en*	´1´ (´0´) enables (disables) low-g interrupt
Bit 2	*high_en_z*	´1´ (´0´) enables (disables) high-g interrupt for z-axis
Bit 1	*high_en_y*	´1´ (´0´) enables (disables) high-g interrupt for y-axis
Bit 0	*high_en_x*	´1´ (´0´) enables (disables) high-g interrupt for x-axis

Register *(0x18)* is *reserved*.

Registers *(0x19)* to *(0x1B)* contain the mapping of interrupts onto the interrupt pins. Default value of each mapping bit is ´0´.

Table 41: Interrupt mapping, register *(0x19)*

***(0x19)* Bit**	**Name**	**Description**
Bit 7	*int1_flat*	´1´ (´0´) maps (unmaps) flat interrupt to INT1 pin
Bit 6	*int1_orient*	´1´ (´0´) maps (unmaps) orientation interrupt to INT1 pin
Bit 5	*int1_s_tap*	´1´ (´0´) maps (unmaps) single tap interrupt to INT1 pin
Bit 4	*int1_d_tap*	´1´ (´0´) maps (unmaps) double tap interrupt to INT1 pin
Bit 3	*- reserved -*	reserved
Bit 2	*int1_slope*	´1´ (´0´) maps (unmaps) slope interrupt to INT1 pin
Bit 1	*int1_high*	´1´ (´0´) maps (unmaps) high-g interrupt to INT1 pin
Bit 0	*int1_low*	´1´ (´0´) maps (unmaps) low-g interrupt to INT1 pin

Table 42: Interrupt mapping, register *(0x1A)*

***(0x1A)* Bit**	**Name**	**Description**
Bit 7	*int2_data*	´1´ (´0´) maps (unmaps) new data interrupt to INT2 pin
Bit 6	*- reserved -*	reserved
Bit 5	*- reserved -*	reserved
Bit 4	*- reserved -*	reserved
Bit 3	*- reserved -*	reserved
Bit 2	*- reserved -*	reserved
Bit 1	*- reserved -*	reserved
Bit 0	*int1_data*	´1´ (´0´) maps (unmaps) new data interrupt to INT1 pin

Table 43: Interrupt mapping, register *(0x1B)*

***(0x1B)* Bit**	**Name**	**Description**
Bit 7	*int2_flat*	´1´ (´0´) maps (unmaps) flat interrupt to INT2 pin
Bit 6	*int2_orient*	´1´ (´0´) maps (unmaps) orientation interrupt to INT2 pin
Bit 5	*int2_s_tap*	´1´ (´0´) maps (unmaps) single tap interrupt to INT2 pin
Bit 4	*int2_d_tap*	´1´ (´0´) maps (unmaps) double tap interrupt to INT2 pin
Bit 3	*- reserved -*	reserved
Bit 2	*int2_slope*	´1´ (´0´) maps (unmaps) slope interrupt to INT2 pin
Bit 1	*int2_high*	´1´ (´0´) maps (unmaps) high-g interrupt to INT2 pin
Bit 0	*int2_low*	´1´ (´0´) maps (unmaps) low-g interrupt to INT2 pin

Registers *(0x1C)* and *(0x1D)* are *reserved*.

Register *(0x1E)* contains the data source definition for those interrupts with selectable data source. Default value of each data source selection bit is ´0´.

Table 44: Interrupt data source definition, register *(0x1E)*

***(0x1E)* Bit**	**Name**	**Description**
Bit 7	*- reserved -*	reserved
Bit 6	*- reserved -*	reserved
Bit 5	*int_src_data*	´1´ (´0´) selects unfiltered (filtered) data for the new data interrupt
Bit 4	*int_src_tap*	´1´ (´0´) selects unfiltered (filtered) data for the single tap and double tap interrupts
Bit 3	*- reserved -*	reserved
Bit 2	*int_src_slope*	´1´ (´0´) selects unfiltered (filtered) data for the slope interrupt
Bit 1	*int_src_high*	´1´ (´0´) selects unfiltered (filtered) data for the high-g interrupt
Bit 0	*int_src_low*	´1´ (´0´) selects unfiltered (filtered) data for the low-g interrupt

Register *(0x1F)* is *reserved*.

Register *(0x20)* contains the behavioural configuration (electrical behaviour) of the interrupt pins. Default value of *(0x20) int1_od* and *(0x20) int2_od* is ´0´. Default value of *(0x20) int1_lvl* and *(0x20) int2_lvl* is ´1´.

Table 45: Electrical behaviour of interrupt pin, register *(0x20)*

***(0x20)* Bit**	**Name**	**Description**
Bit 7	*- reserved -*	reserved
Bit 6	*- reserved -*	reserved
Bit 5	*- reserved -*	reserved
Bit 4	*- reserved -*	reserved
Bit 3	*int2_od*	´0´ selects push-pull, ´1´ selects open drive for INT2 pin
Bit 2	*int2_lvl*	´0´ (´1´) selects active level ´0´ (´1´) for INT2 pin
Bit 1	*int1_od*	´0´ selects push-pull, ´1´ selects open drive for INT1
Bit 0	*int1_lvl*	´0´ (´1´) selects active level ´0´ (´1´) for INT1 pin

Register *(0x21)* contains the interrupt reset bit and the interrupt mode selection. Writing ´1´ to *(0x21) reset_int* resets any latched interrupt.

The settings for *(0x21) latch_int* are ´0000b´ (non-latched), ´0001b´ (temporary, 250 ms), ´0010b´ (temporary, 500 ms), ´0011b´ (temporary, 1 s), ´0100b´ (temporary, 2 s), ´0101b´ (temporary, 4 s), ´0110b´ (temporary, 8 s), ´0111b´ (latched), ´1000b´ (non-latched), ´1001b´ (temporary, 500 μs), ´1010b´ (temporary, 500 μs), ´1011b´ (temporary, 1 ms), ´1100b´ (temporary, 12.5 ms), ´1101b´ (temporary, 25 ms), ´1110b´ (temporary, 50 ms), ´1111b´ (latched).
Default value of *(0x21) latch_int* is ´0000b´.

Table 46: Interrupt reset bit and interrupt mode selection, register *(0x21)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
reset_int	*reserved*	*reserved*	*reserved*	*latch_ int<3>*	*latch_ int<2>*	*latch_ int<1>*	*latch_ int<0>*

Register *(0x22)* contains the delay time definition for the low-g interrupt. The physical delay time can be computed from the content of *(0x22) low_dur* according to:
delay [ms] = [*(0x22) low_dur* + 1] • 2 ms.

Possible delay times range from 2 ms to 512 ms. Default value of *(0x22) low_dur* is 0x09, corresponding to a delay of 20 ms.

Table 47: Delay time definition for the low-g interrupt, register *(0x22)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
low_ dur<7>	*low_ dur<6>*	*low_ dur<5>*	*low_ dur<4>*	*low_ dur<3>*	*low_ dur<2>*	*low_ dur<1>*	*low_ dur<0>*

Register *(0x23)* contains the threshold definition for the low-g interrupt. An LSB of *(0x23) low_th* corresponds to an actual acceleration of 7.81 mg. Therefore, the threshold ranges from 0 g to 1.992 g. Default value of *(0x23) low_th* is 0x30, corresponding to an acceleration of 375 mg.

Table 48: Threshold definition for the low-g interrupt, register *(0x23)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
low_ th<7>	*low_ th<6>*	*low_ th<5>*	*low_ th<4>*	*low_ th<3>*	*low_ th<2>*	*low_ th<1>*	*low_ th<0>*

Register *(0x24)* contains the low-g interrupt mode selection, the low-g interrupt hysteresis setting, and the high-g interrupt hysteresis setting. Setting *(0x24) low_mode* to ´0´ (´1´) selects ´single´ mode (´sum´ mode). Default value is ´0´ (´single´ mode).

(0x24) low_hy sets the hysteresis of the low-g interrupt. An LSB of *(0x24) low_hy* corresponds to an acceleration difference of 125 mg. Default value of *(0x24) low_hy* is ´01b´.

(0x24) high_hy sets the hysteresis of the high-g interrupt. The meaning of an LSB of *(0x24) high_hy* depends on the selected g-range. It corresponds to an acceleration difference of 125 mg in 2g-range, 250 mg in 4g-range, 500 mg in 8g-range, and 1000mg in 16g-range.
Default value of *(0x24) high_hy* is ´10b´.

Table 49: Threshold definition for the low-g interrupt, register *(0x24)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
high_hy<1>	*high_hy<0>*	*reserved*	*reserved*	*reserved*	*low_mode*	*low_hy<1>*	*low_hy<0>*

Register *(0x25)* contains the delay time definition for the high-g interrupt. The physical delay time can be computed from the content of *(0x25) high_dur* according to
delay [ms] = [*(0x25) high_dur* + 1] • 2 ms. Possible delay times range from 2 ms to 512 ms.
Default value of *(0x25) high_dur* is 0x0F, corresponding to a delay of 32 ms.

Table 50: Delay time definition for the high-g interrupt, register *(0x25)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
high_dur<7>	*high_dur<6>*	*high_dur<5>*	*high_dur<4>*	*high_dur<3>*	*high_dur<2>*	*high_dur<1>*	*high_dur<0>*

Register *(0x26)* contains the threshold definition for the high-g interrupt. The meaning of an LSB of *(0x26) high_th* depends on the selected g-range. It corresponds to 7.81 mg in 2g-range, 15.63 mg in 4g-range, 31.25 mg in 8g-range, and 62.5 mg in 16g-range.
Default value of *(0x26) high_th* is 0xC0.

Table 51: Threshold definition for the high-g interrupt, register *(0x26)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
high_th<7>	*high_th<6>*	*high_th<5>*	*high_th<4>*	*high_th<3>*	*high_th<2>*	*high_th<1>*	*high_th<0>*

Register *(0x27)* contains the definition of the number of samples to be evaluated for the slope interrupt (any-motion detection). The number of samples is *N = (0x27) slope_dur* + 1.
Default value of *(0x27) slope_dur* is ´00b´.

Table 52: Samples number definition for the slope interrupt, register *(0x27)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
reserved	*reserved*	*reserved*	*reserved*	*reserved*	*reserved*	*slope_dur<1>*	*slope_dur<0>*

Register ***(0x28)*** contains the threshold definition for the slope interrupt. An LSB of *(0x28) slope_th* corresponds to an LSB of acceleration data. Its meaning therefore depends on the selected g-range. Default value of *(0x28) slope_th* is 0x14.

Table 53: Slope threshold for the slope interrupt, register *(0x28)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
slope_ th<7>	*slope_ th<6>*	*slope_ th<5>*	*slope_ th<4>*	*slope_ th<3>*	*slope_ th<2>*	*slope_ th<1>*	*slope_ th<0>*

Register ***(0x29)*** is *reserved*.

Register ***(0x2A)*** contains the timing definitions for the single tap and double tap interrupts.

(0x2A) tap_quiet = ´0´ (´1´) selects a quiet duration of 30 ms (20 ms). The default value of *(0x2A) tap_quiet* is ´0´.

(0x2A) tap_shock = ´0´ (´1´) selects a shock duration of 50 ms (75 ms). The default value of *(0x2A) tap_shock* is ´0´.

(0x2A) tap_dur selects the length of the time window for the second shock event (for double tap detection). The settings for *(0x2A) tap_dur* are ´000b´ (50 ms), ´001b´ (100 ms), ´010b´ (150 ms), ´011b´ (200 ms), ´100b´ (250 ms), ´101b´ (375 ms), ´110b´ (500 ms), ´111b´ (700 ms). The default value of *(0x2A) tap_dur* is ´100b´.

Table 54: Tap Quiet duration and tap shock duration, register *(0x2A)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
tap_ quiet	*tap_ shock*	*reserved*	*reserved*	*reserved*	*tap_ dur<2>*	*tap_ dur<1>*	*tap_ dur<0>*

Register ***(0x2B)*** contains the definition of the number of samples to be processed after wake-up in low-power mode and the threshold definition for the single and double tap interrupts. *(0x2B) tap_samp* selects the number of samples that are processed after wake-up in the low-power mode. The settings for *(0x2B) tap_samp* are ´00b´ (2 samples), ´01b´ (4 samples), ´10b´ (8 samples), and ´11b´ (16 samples). Default value of *(0x2B) tap_samp* is ´00b´.

The meaning of an LSB of *(0x2B) tap_th* depends on the selected g-range. It corresponds to an acceleration difference of 62.5mg in 2g-range, 125mg in 4g-range, 250mg in 8g-range, and 500mg in 16g-range. Default value of *(0x2B) tap_th* is 0x0A.

Table 55: Samples number after wake-up and threshold tap interrupt, register *(0x2B)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
tap_ samp<1>	*tap_ samp<0>*	*reserved*	*tap_ th<4>*	*tap_ th<3>*	*tap_ th<2>*	*tap_ th<1>*	*tap_ th<0>*

Register ***(0x2C)*** contains the definition of hysteresis, blocking, and mode for the orientation interrupt. *(0x2C) orient_hyst* sets the hysteresis of the orientation interrupt; 1 LSB always corresponds to 62.5 mg, in any g-range (i.e. increment is independent from g-range setting). Default value of *(0x2C) orient_hyst* is ´001b´.

(0x2C) orient_blocking selects the kind of blocking that is used for the generation of the orientation interrupt. The settings for *(0x2C) orient_blocking* are ´00b´ (no blocking), ´01b´ (theta blocking), ´10b´ (theta blocking or slope in any axis > 0.2 g), and ´11b´ (*orient* value not stable for at least 100 ms or theta blocking or slope in any axis > 0.4 g). Default value of *(0x2C) orient_blocking* is ´10b´.

(0x2C) orient_mode sets the thresholds for switching between the different orientations. The settings for *(0x2C) orient_mode* are ´00b´ (symmetrical), ´01b´ (high-asymmetrical), ´10b´ (low-asymmetrical), ´11b´ (symmetrical). Default value of *(0x2C) orient_mode* is ´00b´.

Table 56: Hysteresis, Blocking for Orientation Interrupt, Register *(0x2C)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
reserved	*orient_ hyst<2>*	*orient_ hyst<1>*	*orient_ hyst<0>*	*orient_ blocking<1>*	*orient_ blocking<0>*	*orient_ mode<1>*	*orient_ mode<0>*

Register ***(0x2D)*** contains the definition of the theta blocking angle for the orientation interrupt. *(0x2D) orient_theta* defines a blocking angle between 0° and 44.8° as described in section "4.8.1.7 Orientation blocking". Default value of *(0x2D) orient_theta* is 0x08.

Table 57: Theta blocking angle, register *(0x2D)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
reserved	*reserved*	*orient_ theta<5>*	*orient_ theta<4>*	*orient_ theta<3>*	*orient_ theta<2>*	*orient_ theta<1>*	*orient_ theta<0>*

Register ***(0x2E)*** contains the definition of the flat threshold angle for the flat interrupt. *(0x2E) flat_theta* defines a blocking angle between 0° and 44.8° as described in section"4.8.8 Flat detection". Default value of *(0x2E) flat_theta* is 0x08.

Table 58: Flat threshold angle, register *(0x2E)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
reserved	*reserved*	*flat_ theta<5>*	*flat_ theta<4>*	*flat_ theta<3>*	*flat_ theta<2>*	*flat_ theta<1>*	*flat_ theta<0>*

Register ***(0x2F)*** contains the definition of the flat hold time. *(0x2F) flat_hold_time* defines the time a new flat value has to be at least stable for before the interrupt is generated. The settings for *(0x2F) flat_hold_time* are ´00b´ (0), ´01b´ (512 ms), ´10b´ (1024 ms), ´11b´ (2048 ms). Default value of *(0x2F) flat_hold_time* is ´01b´.

Table 59: Flat threshold angle, register *(0x2F)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
reserved	*reserved*	*flat_hold_ time<1>*	*flat_hold_ time<0>*	*reserved*	*reserved*	*reserved*	*reserved*

Register *(0x30) and (0x31)* are *reserved*.

5.12 Self-test

Register *(0x32)* contains the settings for the activation of the sensor self-test.

(0x32) self_test_sign sets the sign of the electrostatic excitation. The settings for *(0x32) self_test_sign* are ´0´ (positive sign) and ´1´ (negative sign). Default value of *(0x32) self_test_sign* is ´0´.

(0x32) self_test_axis defines the axis which shall be excited. Only one axis can be excited at the same time. The settings for *(0x32) self_test_axis* are ´00b´ (no self-test), ´01´ (x-axis), ´10´ (y-axis), and ´11´ (z-axis). Default value of *(0x32) self_test_axis* is ´00b´.

Table 60: Sensor self-test, register *(0x32)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
reserved	*reserved*	*reserved*	*reserved*	*reserved*	*self_test _sign*	*self_test _axis<1>*	*self_test _axis<0>*

5.13 Non-volatile memory control (EEPROM control)

Register *(0x33)* contains the control settings for the non-volatile memory (EEPROM). *(0x33) nvm_load* is used to perform a user-defined image update. Writing ´1´ *(0x33) nvm_load* starts the update procedure. The value ´1´ is kept as long as the update procedure runs, afterwards it is reset to ´0´.

(0x33) nvm_rdy contains the status of writing the EEPROM. *(0x33) nvm_rdy* is ´0´ as long as writing the EEPROM endures, it is ´1´ if currently no write access is performed and, therefore, a new write access can be initiated.

Writing ´1´to *(0x33) nvm_prog_trig* triggers writing the EEPROM. The EEPROM can only be written if it was unlocked before.

Writing ´1´to *(0x33) nvm_prog_mode* unlocks the EEPROM.

Table 61: EEPROM control settings, register *(0x33)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
reserved	*reserved*	*reserved*	*reserved*	*nvm_load*	*nvm_rdy*	*nvm_prog _trig*	*nvm_prog _mode*

5.14 Interface configuration

Register *(0x34)* contains the settings for the digital interfaces. Writing ´1´to *(0x34) i2c_wdt_en* enables the watchdog at the SDI pin (= SDA for I²C) if I²C is selected. Default value of *(0x34) i2c_wdt_en* is ´0´.

(0x34) i2c_wdt_sel selects the I²C data pad watchdog timer period. The settings for *(0x34) i2c_wdt_sel* are ´0´ (1 ms) and ´1´ (50 ms). Default value of *(0x34) i2c_wdt_sel* is ´0´.

(0x34) spi3 selects the SPI mode. The settings for *(0x34) spi3* are ´0´ (4-wire SPI) and ´1´ (3-wire SPI). Default value of *(0x34) spi3* is ´0´.

Table 62: EEPROM control settings, register *(0x34)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
reserved	*reserved*	*reserved*	*reserved*	*reserved*	*i2c_wdt _en*	*i2c_wdt _sel*	*spi3*

Register *(0x35)* is *reserved*.

5.15 Offset compensation

Register *(0x36)* contains settings for the offset compensation in general, for fast offset compensation, and for slow offset compensation. Writing ´1´to *(0x36) offset_reset* sets all offset compensation registers *(0x38 to 0x3D)* to zero.

Default value of *(0x36) offset_reset* is ´0´.

(0x36) cal_trigger starts the fast compensation process for the specified axis. The settings for *(0x36) cal_trigger* are ´00b´ (no axis selected), ´01b´ (x-axis), ´10b´ (y-axis), ´11b´ (z-axis). A non-zero value is kept until the fast compensation procedure is finished. Default value of *(0x36) cal_trigger* is ´00b´.

(0x36) cal_rdy indicates the state of the fast compensation. *(0x36) cal_rdy* is ´0´ when *(0x36) cal_trigger* has a nonzero value, otherwise *(0x36) cal_rdy* is ´1´.

Writing ´1´ (´0´) to *(0x36) hp_z_en* enables (disables) slow offset compensation for the z-axis.
Writing ´1´ (´0´) to *(0x36) hp_y_en* enables (disables) slow offset compensation for the y-axis.

Writing ´1´ (´0´) to *(0x36) hp_x_en* enables (disables) slow offset compensation for the x-axis. Default value for each of *(0x36) hp_x_en, (0x36) hp_y_en,* and *(0x36) hp_x_en* is ´0´, respectively.

Table 63: Offset compensation, fast offset compensation, register *(0x36)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
offset _reset	*cal_ trigger<1>*	*cal_ trigger<0>*	*cal_rdy*	*reserved*	*hp_z_en*	*hp_y_en*	*hp_x_en*

Register *(0x37)* contains settings for the offset compensation in general, and for slow offset compensation. *(0x37) offset_target_z* sets the target value for the offset compensation of the z-axis.

(0x37) offset_target_y sets the target value for the offset compensation of the y-axis.
(0x37) offset_target_x sets the target value for the offset compensation of the x-axis.
The settings for *(0x37) offset_target_x, (0x37) offset_target_y,* and *(0x37) offset_target_z* are ´00b´ (0 g), ´01b´ (+1 g), ´10b´ (-1 g), and ´11b´ (0 g). Default value of each of *(0x37) offset_target_x, (0x37) offset_target_y,* and *(0x37) offset_target_z* is ´00b´, respectively.

(0x37) cut_off defines the number of samples for comparison by the slow offset compensation. The settings for *(0x37) cut_off* are ´0´ (8 samples) and ´1´ (16 samples). The default value of *(0x37) cut_off* is ´0´.

Table 64: Offset compensation, slow offset compensation, register *(0x37)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
reserved	*offset_tar get_z<1>*	*offset_tar get_z<0>*	*offset_tar get_y<1>*	*offset_tar get_y<0>*	*offset_tar get_x<1>*	*offset_tar get_x<0>*	*cut_off*

Register *(0x38)* contains the compensation value for filtered data for the x-axis. The contents of each of the registers *(0x38)* to *(0x3D)* is added to the corresponding acceleration data; it can be set either automatically by one of the implemented compensation algorithms or manually. These registers are image registers of registers in the EEPROM; the content of the EEPROM is copied to them after every reset.

Table 65: Filtered data compensation for the x-axis, register *(0x38)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
offset_ filt_x<7>	*offset_ filt_x<6>*	*offset_ filt_x<5>*	*offset_ filt_x<4>*	*offset_ filt_x<3>*	*offset_ filt_x<2>*	*offset_ filt_x<1>*	*offset_ filt_x<0>*

Register ***(0x39)*** contains the compensation value for filtered data for the y-axis.

Table 66: Filtered data compensation for the y-axis, register *(0x39)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
offset_ filt_y<7>	*offset_ filt_y<6>*	*offset_ filt_y<5>*	*offset_ filt_y<4>*	*offset_ filt_y<3>*	*offset_ filt_y<2>*	*offset_ filt_y<1>*	*offset_ filt_y<0>*

Register ***(0x3A)*** contains the compensation value for filtered data for the z-axis.

Table 67: Filtered data compensation for the z-axis, register *(0x3A)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
offset_ filt_z<7>	*offset_ filt_z<6>*	*offset_ filt_z<5>*	*offset_ filt_z<4>*	*offset_ filt_z<3>*	*offset_ filt_z<2>*	*offset_ filt_z<1>*	*offset_ filt_z<0>*

Register ***(0x3B)*** contains the compensation value for unfiltered data for the x-axis.

Table 68: Unfiltered data compensation for the x-axis, register *(0x3B)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
offset_ unfilt_x <7>	*offset_ unfilt_x <6>*	*offset_ unfilt_x <5>*	*offset_ unfilt_x <4>*	*offset_ unfilt_x <3>*	*offset_ unfilt_x <2>*	*offset_ unfilt_x <1>*	*offset_ unfilt_x <0>*

Register ***(0x3C)*** contains the compensation value for unfiltered data for the y-axis.

Table 69: Unfiltered data compensation for the x-axis, register *(0x3C)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
offset_ unfilt_y <7>	*offset_ unfilt_y <6>*	*offset_ unfilt_y <5>*	*offset_ unfilt_y <4>*	*offset_ unfilt_y <3>*	*offset_ unfilt_y <2>*	*offset_ unfilt_y <1>*	*offset_ unfilt_y <0>*

Register ***(0x3D)*** contains the compensation value for unfiltered data for the z-axis.

Table 70: Unfiltered data compensation for the y-axis, register *(0x3D)*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
offset_ unfilt_z <7>	*offset_ unfilt_z <6>*	*offset_ unfilt_z <5>*	*offset_ unfilt_z <4>*	*offset_ unfilt_z <3>*	*offset_ unfilt_z <2>*	*offset_ unfilt_z <1>*	*offset_ unfilt_z <0>*

Registers ***(0x3E)*** **and** ***(0x3F)*** are image registers of registers in the EEPROM. They are not linked to any sensor-specific functionality.

6. Digital interfaces

The BMA250 supports two serial digital interface protocols for communication as a slave with a host device (when operating in general mode): SPI and I²C. The active interface is selected by the state of the Pin#11 (PS) 'protocol select' pin: ´0´ (´1´) selects SPI (I²C). For details see section 4.2 Operational modes.

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

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

Table 71: Mapping of the interface pins

Pin#	Name	use w/ SPI	use w/ I²C	Description
1	SDO	SDO	address	SPI: Data Output (4-wire mode) I²C: Used to set LSB of I²C address
2	SDx	SDI	SDA	SPI: Data Input (4-wire mode) Data Input / Output (3-wire mode) I²C: Serial Data
10	CSB	CSB	unused	Chip Select (enable)
12	SCx	SCK	SCL	SPI: Serial Clock I²C: Serial Clock

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

Table 72: Electrical specification of the interface pins

Parameter	Symbol	Condition	Min	Typ	Max	Units
PS Impedance for Tri-state Detection	R_{TS}		1			MΩ
	C_{TS}				10	pF
PS Impedance for Non-Tri-state	R_{NTS}				5	kΩ
Pull-up Resistance	R_{up}	Internal Pull-up Resistance to VDDIO	70	120	190	kΩ
Pull-down Resistance	R_{down}	Internal Pull-down Resistance to GND	12	20	32	kΩ
Input Capacitance	C_{in}			5	10	pF
I²C Bus Load Capacitance (max. drive capability)	C_{I2C_Load}				400	pF

6.1 Serial peripheral interface (SPI)

The timing specification for SPI of the BMA250 is given in the following table:

Table 73: SPI timing

Parameter	Symbol	Condition	Min	Max	Units
Clock Frequency	f_{SPI}	Max. Load on SDI or SDO = 25pF		10	MHz
SCK Low Pulse	t_{SCKL}		20		ns
SCK High Pulse	t_{SCKH}		20		ns
SDI Setup Time	t_{SDI_setup}		20		ns
SDI Hold Time	t_{SDI_hold}		20		ns
SDO Output Delay	t_{SDO_OD}	Load = 25pF		30	ns
		Load = 250pF, V_{DDIO} = 2.4V		40	ns
CSB Setup Time	t_{CSB_setup}		20		ns
CSB Hold Time	t_{CSB_hold}		40		ns

The following figure shows the definition of the SPI timings given in table 73:

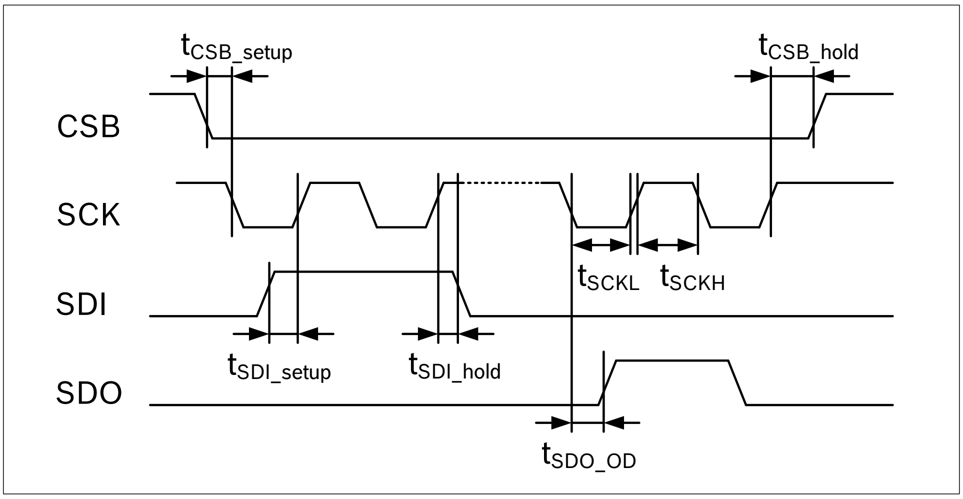


Figure 10: SPI timing diagram

The SPI interface of the BMA250 is compatible with two modes, ´00´ and ´11´. The automatic selection between [CPOL = ´0´ and CPHA = ´0´] and [CPOL = ´1´ and CPHA = ´1´] is done based on the value of SCK after a falling edge of CSB.

Two configurations of the SPI interface are supported by the BMA250: 4-wire and 3-wire. The same protocol is used by both configurations. The device operates in 4-wire configuration by default. It can be switched to 3-wire configuration by writing ´1´ to *(0x34) spi3*. Pin SDI is used as the common data pin in 3-wire configuration.

For single byte read as well as write operations, 16-bit protocols are used. The BMA250 also supports multiple-byte read operations.

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

The basic write operation waveform for 4-wire configuration is depicted in figure 11. During the entire write cycle SDO remains in high- impedance state.

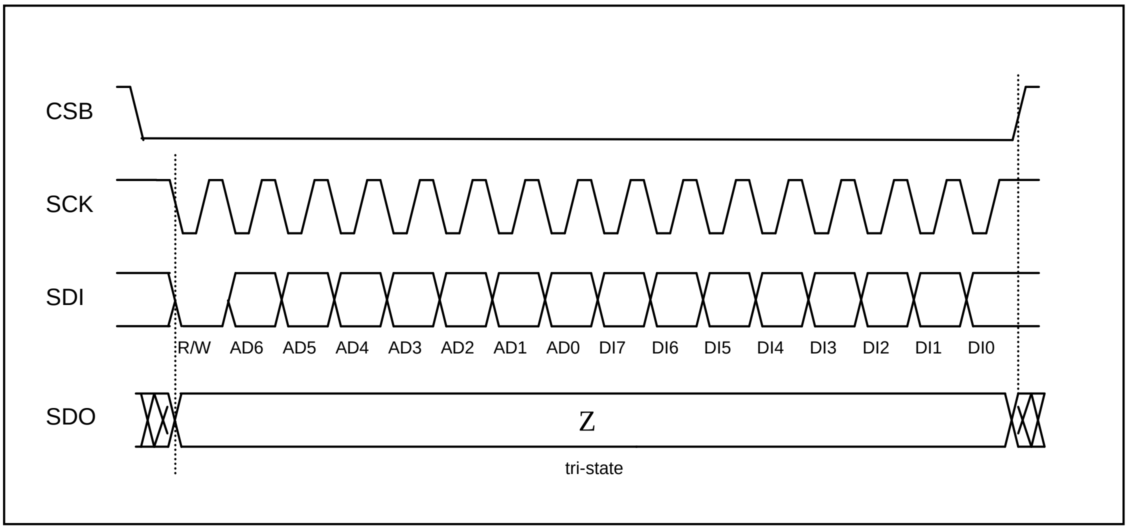


Figure 11: 4-wire basic SPI write sequence (mode ´11´)

The basic read operation waveform for 4-wire configuration is depicted in figure 12:

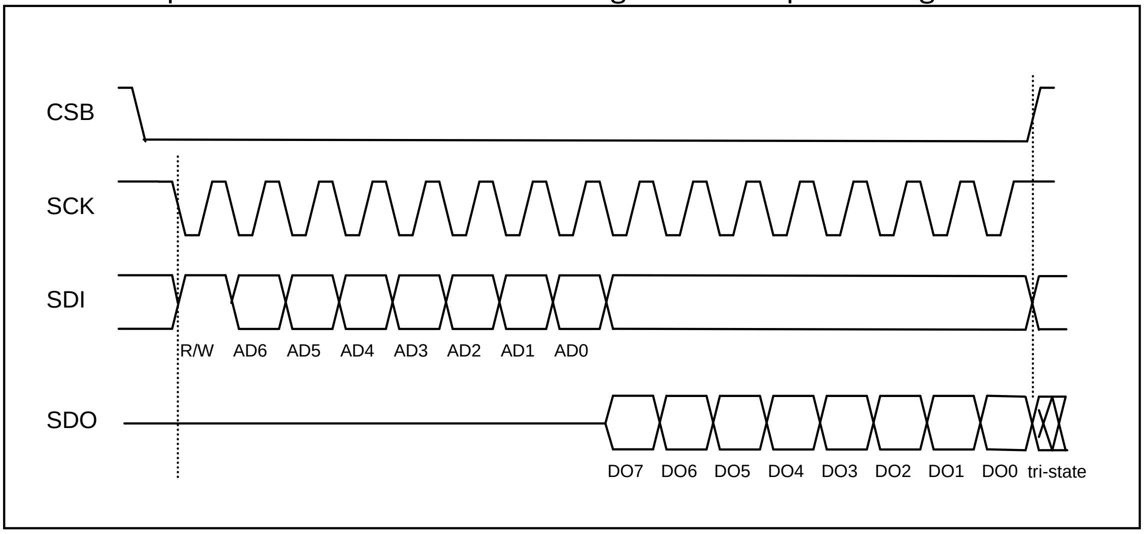


Figure 12: 4-wire basic SPI read sequence (mode ´11´)

The data bits are used as follows:

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

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

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

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

The principle of multiple read is shown in figure 13:

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

Figure 13: SPI multiple read

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

The protocol as such is the same in 3-wire configuration as it is in 4-wire configuration. The basic operation waveform (read or write access) for 3-wire configuration is depicted in figure 14:

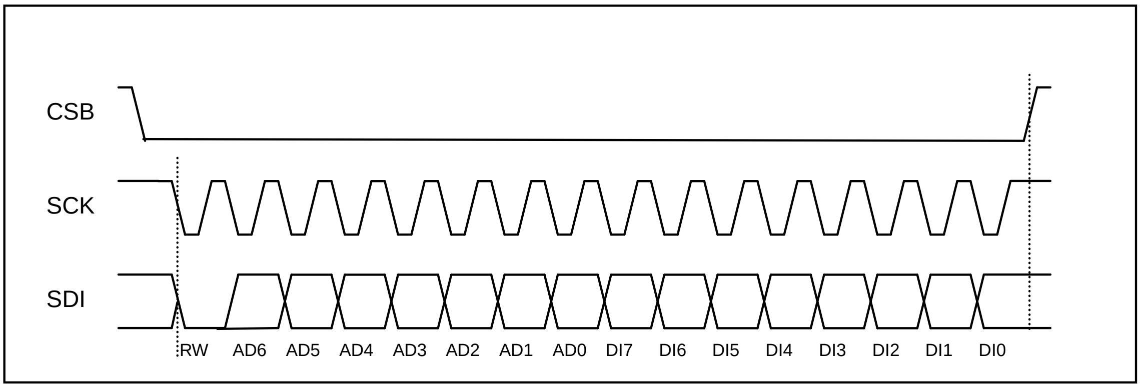


Figure 14: 3-wire basic SPI read or write sequence (mode ´11´)

6.2 Inter-Integrated Circuit (I²C)

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

The I²C interface of the BMA250 is compatible with the I²C Specification UM10204 Rev. 03 (19 June 2007), available at http://www.nxp.com. The BMA250 supports I²C standard mode and fast mode, only 7-bit address mode is supported. For V_{DDIO} = 1.2V to 1.8V the guaranteed voltage output levels are slightly relaxed as described in the Parameter Specification (table 1).

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

The timing specification for I²C of the BMA250 is given in table 74:

Table 74: I²C timings

Parameter	Symbol	Condition	Min	Max	Units
Clock Frequency	f_{SCL}			400	kHz
SCL Low Period	t_{LOW}		1.3		μs
SCL High Period	t_{HIGH}		0.6		
SDA Setup Time	t_{SUDAT}		0.1		
SDA Hold Time	t_{HDDAT}		0.0		
Setup Time for a repeated Start Condition	t_{SUSTA}		0.6		
Hold Time for a Start Condition	t_{HDSTA}		0.6		
Setup Time for a Stop Condition	t_{SUSTO}		0.6		
Time before a new Transmission can start	t_{BUF}		1.3		

Figure 15 shows the definition of the I²C timings given in table 74:

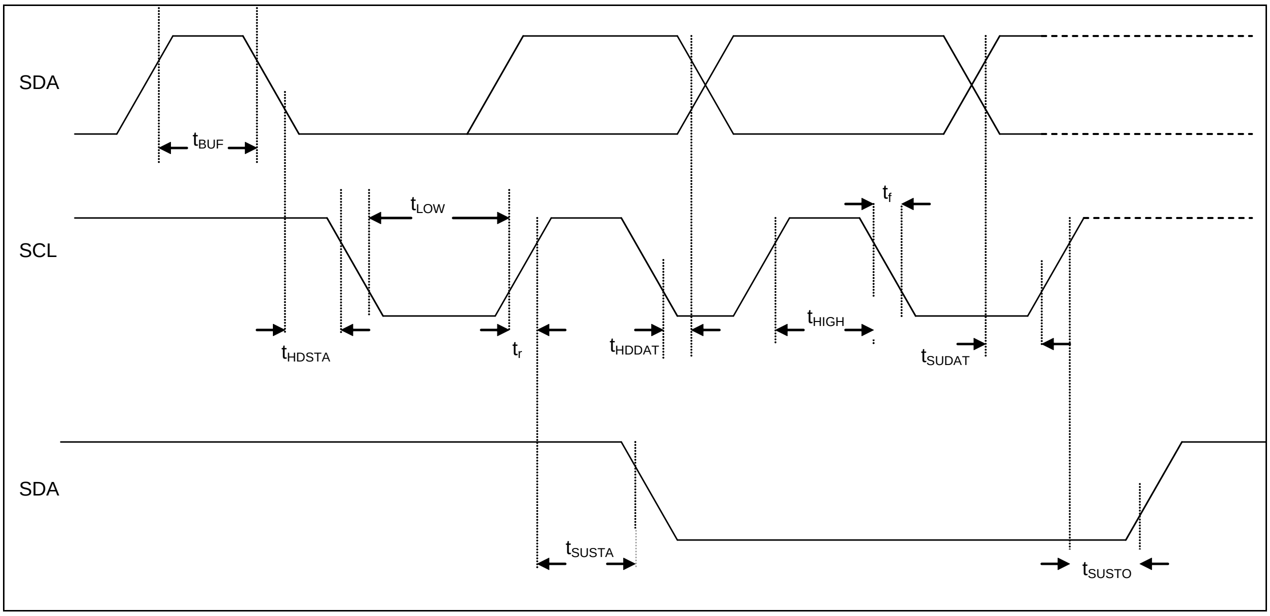


Figure 15: I²C timing diagram

The I²C protocol works as follows:

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

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

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

In the following diagrams these abbreviations are used:

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

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

I²C write access:

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

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

Example of an I²C write access:

				Control byte		Data byte		
Start	Slave Adress	RW	ACKS	Register adress (0x10)	ACKS	Data (0x09)	ACKS	Stop
S	0 0 1 1 0 0 0	0		0 0 0 1 0 0 0 0		X X X X X X X X		P

Figure 16: I²C write

I²C read access:

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

A read sequence consists of a one-byte I²C write phase followed by the I²C read phase. The two parts of the transmission must be separated by a repeated start condition (Sr). The I²C write phase addresses the slave and sends the register address to be read. After slave acknowledges the transmission, the master generates again a start condition and sends the slave address together with a read bit (RW = 1). Then the master releases the bus and waits for the data bytes to be read out from slave. After each data byte the master has to generate an acknowledge bit (ACK = 0) to enable further data transfer. A NACKM (ACK = 1) from the master stops the data being transferred from the slave. The slave releases the bus so that the master can generate a STOP condition and terminate the transmission.
The register address is automatically incremented and, therefore, more than one byte can be sequentially read out. Once a new data read transmission starts, the start address will be set to the register address specified in the latest I²C write command. By default the start address is set at 0x00. In this way repetitive multi-bytes reads from the same starting address are possible.

In order to prevent the I²C slave of the device to lock-up the I²C bus, a watchdog timer (WDT) is implemented. The WDT observes internal I²C signals and resets the I²C interface if the bus is locked-up by the BMA250. The activity and the timer period of the WDT can be configured through the bits *(0x34) i2c_wdt_en* and *(0x34) i2c_wdt_sel*.

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

Example of an I²C read access:

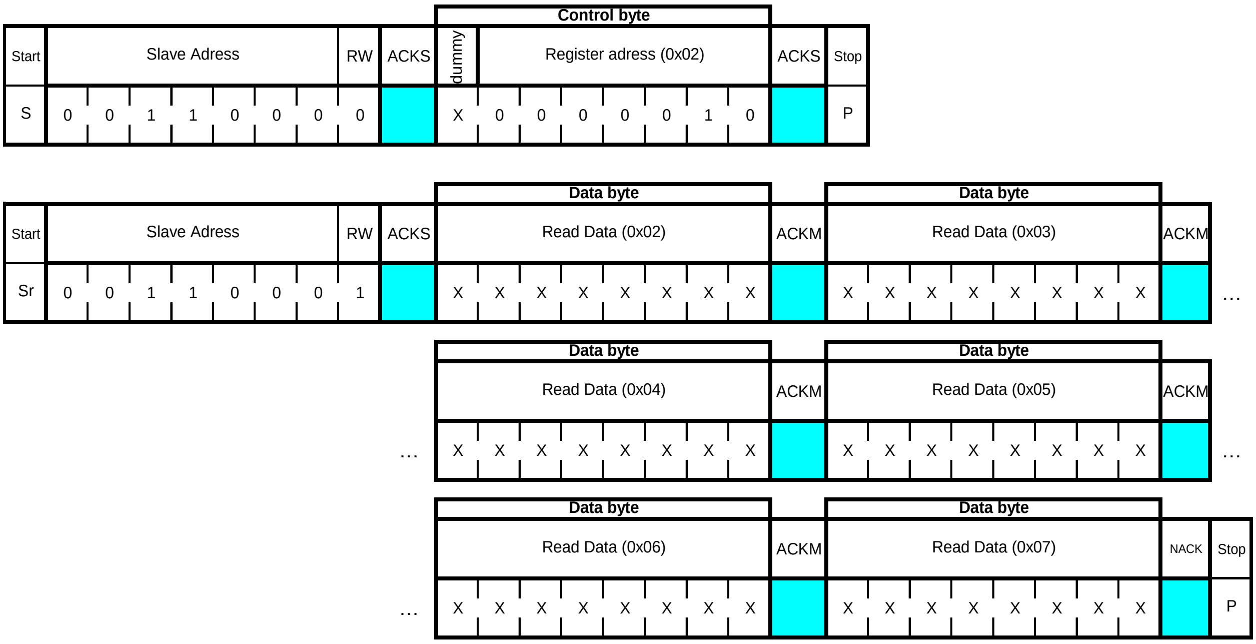


Figure 17: I²C multiple read

7. Pin-out and connection diagram

7.1 Pin-out

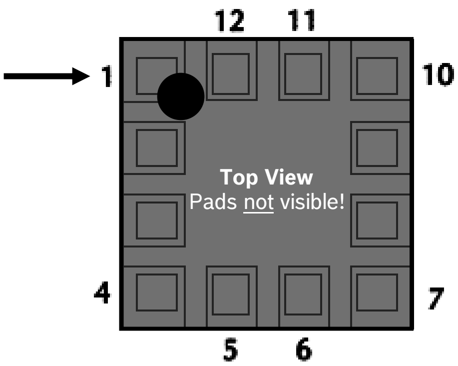


Figure 18: Pin-out top view

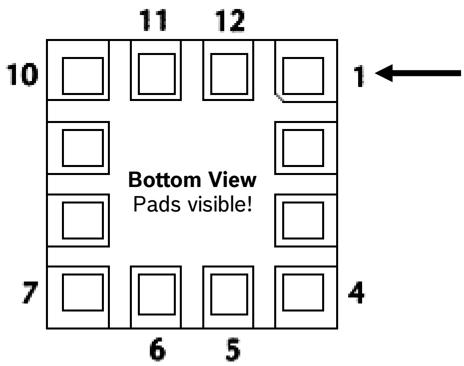


Figure 19 Pin-out bottom view

Table 75: Pin description

Pin#	Name	I/O Type	Description	Connect to		
				in SPI 4W	In SPI 3W	in I²C
1	SDO	Digital out	Serial data output in SPI Address select in I²C mode see chapter 6.2	SDO	DNC (float)	GND for default addr.
2	SDx	Digital I/O	SDA serial data I/O in I²C SDI serial data input in SPI 4W SDA serial data I/O in SPI 3W	SDI	SDA	SDA
3	VDDIO	Supply	Digital I/O supply voltage (1.2V … 3.6V)	V_{DDIO}	V_{DDIO}	V_{DDIO}
4	NC	--		GND	GND	GND
5	INT1	Digital out	Interrupt output 1	INT1	INT1	INT1
6	INT2	Digital out	Interrupt output 2	INT2	INT2	INT2
7	VDD	Supply	Power supply for analog & digital domain (1.62V … 3.6V)	V_{DD}	V_{DD}	V_{DD}
8	GNDIO	Ground	Ground for I/O	GND	GND	GND
9	GND	Ground	Ground for digital & analog	GND	GND	GND
10	CSB	Digital in	Chip select for SPI mode	CSB	CSB	DNC (float)
11	PS	Digital in	Protocol select (GND = SPI, V_{DDIO} = I²C, float = µC-less). Pin must not float unless dedicated mode is used, see chapter 4.2.2	GND	GND	V_{DDIO}
12	SCx	Digital in	SCK for SPI serial clock SCL for I²C serial clock	SCK	SCK	SCL

7.2 Connection diagram 4-wire SPI

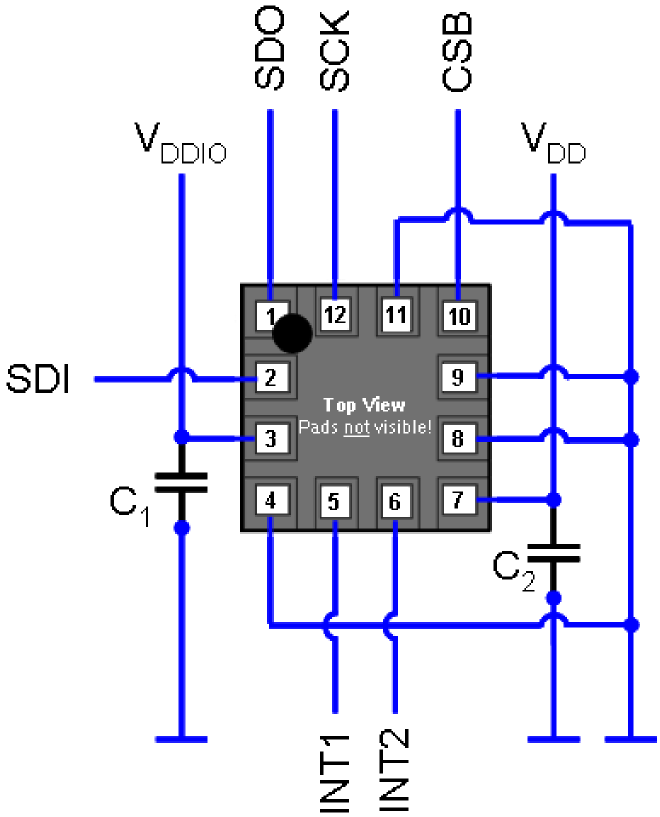


Figure 20: 4-wire SPI connection

7.3 Connection diagram 3-wire SPI

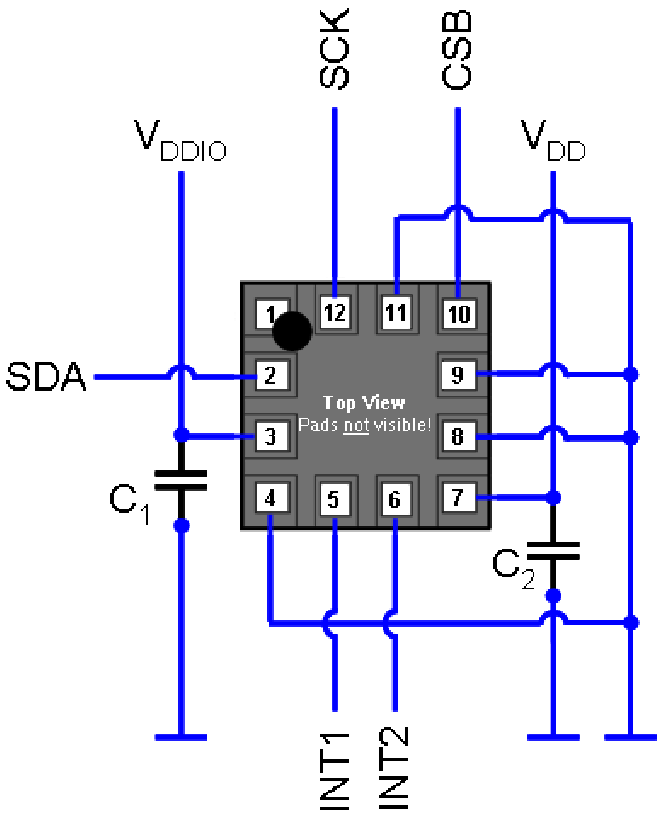


Figure 21: 3-wire SPI connection

7.4 Connection diagram I²C

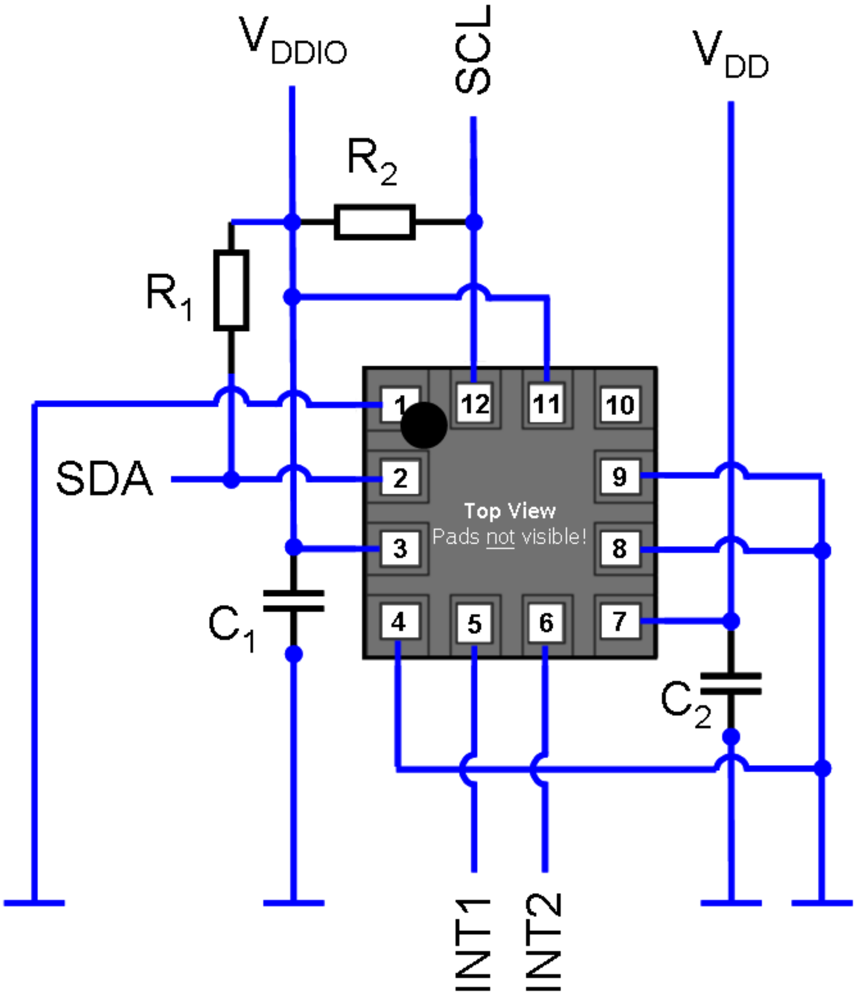


Figure 22: I²C connection

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

8. Package

8.1 Outline dimensions

The sensor housing is a standard LGA package. It is compliant with JEDEC Standard MO-229 Type VGGD-3. Its dimensions are the following.

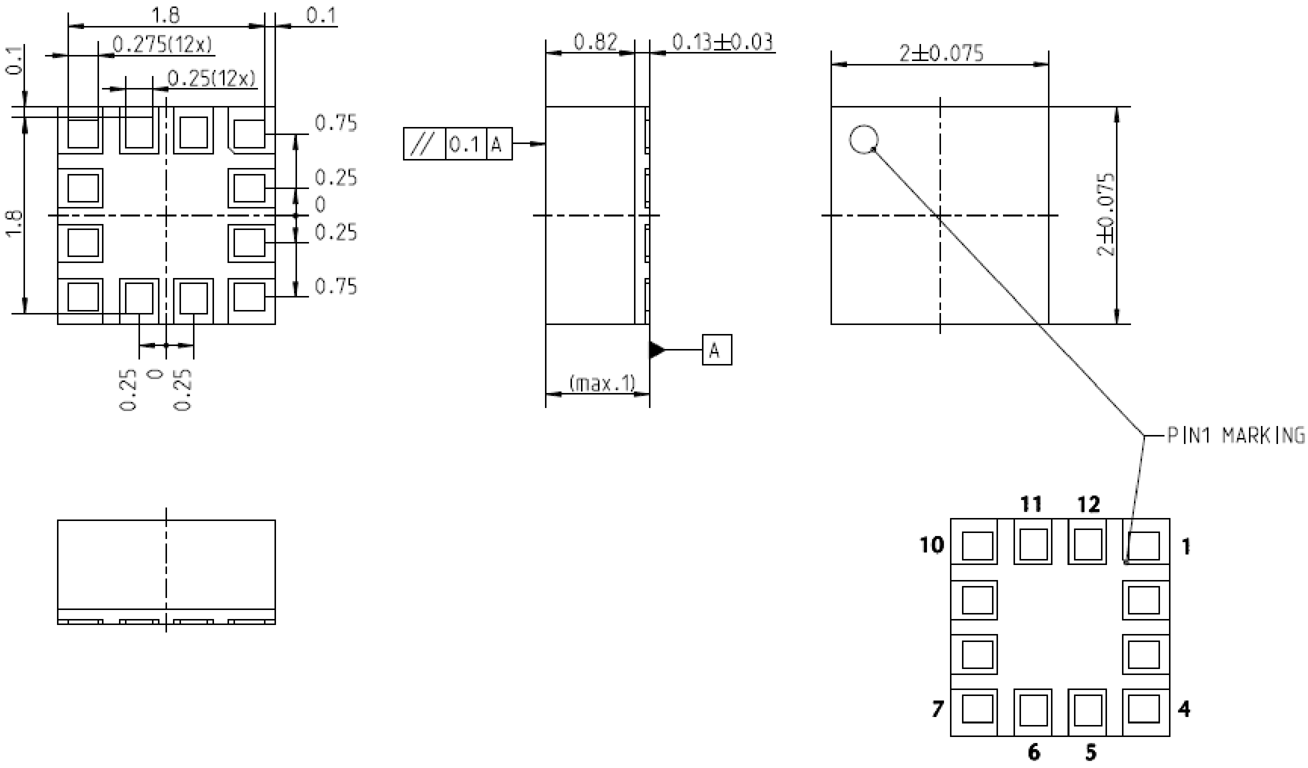


Figure 23: Package outline dimensions

8.2 Sensing axes orientation

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

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

- ± 0g for the X channel
- ± 0g for the Y channel
- + 1g for the Z channel

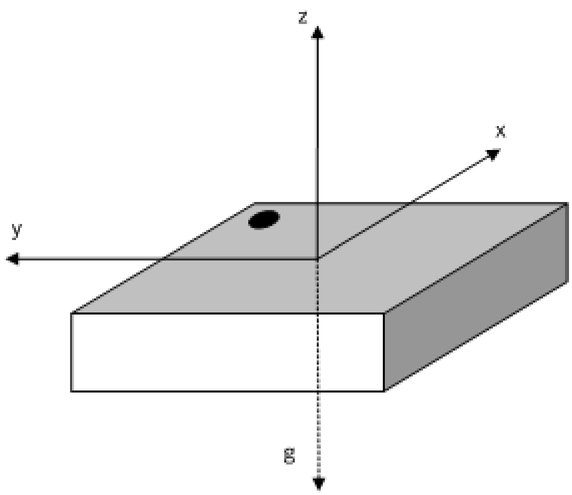
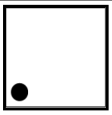
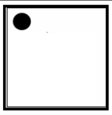
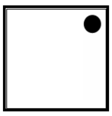
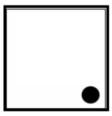


Figure 24: Orientation of sensing axis

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

Table 76: Output signals depending on sensor orientation

Sensor Orientation (gravity vector ↓)						
Output Signal X	0g / 0LSB	1g/256LSB	0g / 0LSB	-1g/-256LSB	0g / 0LSB	0g / 0LSB
Output Signal Y	-1g/-256LSB	0g / 0LSB	+1g / 256LSB	0g / 0LSB	0g / 0LSB	0g / 0LSB
Output Signal Z	0g / 0LSB	0g / 0LSB	0g / 0LSB	0g / 0LSB	1g/256LSB	-1g/-256LSB

8.3 Landing pattern recommendation

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

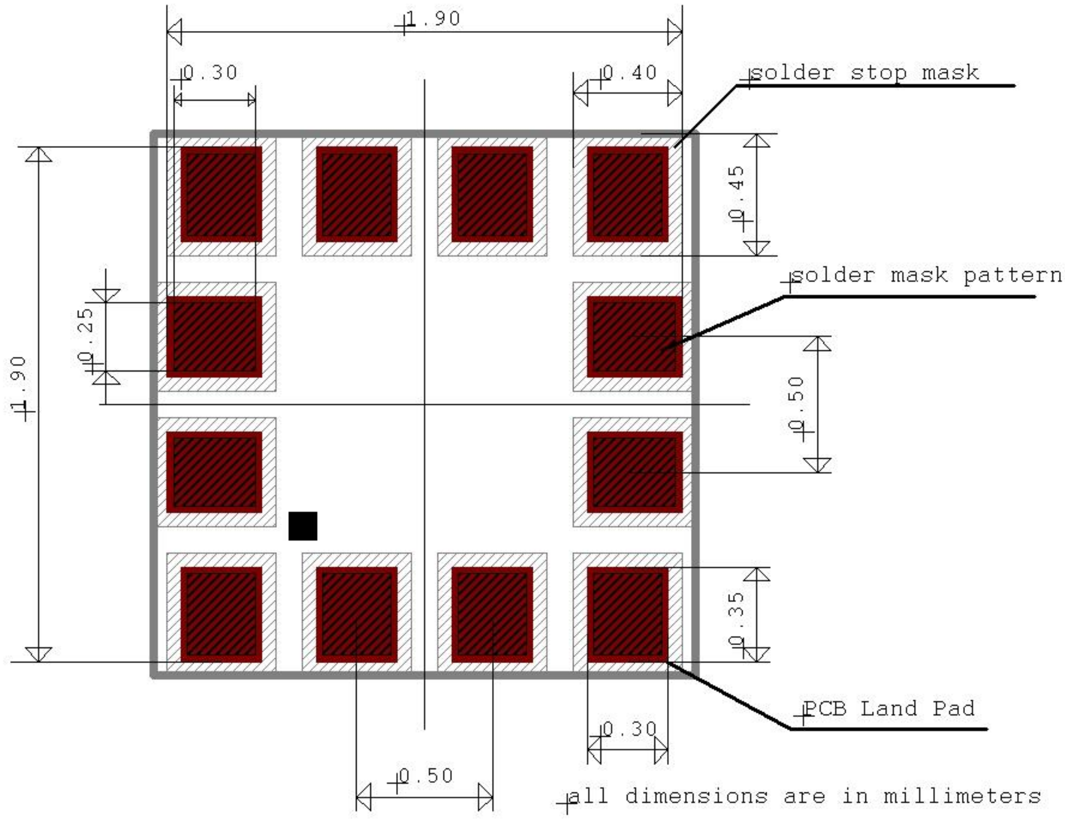
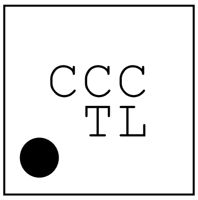


Figure 25: Landing patterns relative to the device pins, dimensions are in mm

8.4 Marking

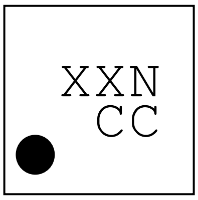
8.4.1 Mass production samples

Table 77: Marking of mass production samples

<table>
<tr><th>Labeling</th><th>Name</th><th>Symbol</th><th>Remark</th></tr>
<tr><td rowspan="4">
</td><td>Lot counter</td><td>CCC</td><td>3 alphanumeric digits, variable
to generate mass production trace-code</td></tr>
<tr><td>Product number</td><td>T</td><td>1 alphanumeric digit, fixed
to identify product type, T = “8”</td></tr>
<tr><td>Sub-con ID</td><td>L</td><td>1 alphanumeric digit, variable
to identify sub-con (L = “A” or L = “U” or L = “P”)</td></tr>
<tr><td>Pin 1 identifier</td><td>●</td><td>--</td></tr>
</table>

8.4.2 Engineering samples

Table 78: Marking of engineering samples

<table>
<tr><th>Labeling</th><th>Name</th><th>Symbol</th><th>Remark</th></tr>
<tr><td rowspan="4">
</td><td>Eng. sample ID</td><td>N</td><td>1 alphanumeric digit, fixed
to identify engineering sample, N = “e”</td></tr>
<tr><td>Sample ID</td><td>XX</td><td>2 alphanumeric digits, variable
to generate trace-code</td></tr>
<tr><td>Counter ID</td><td>CC</td><td>2 alphanumeric digits, variable
to generate trace-code</td></tr>
<tr><td>Pin 1 identifier</td><td>●</td><td>--</td></tr>
</table>

8.5 Soldering guidelines

The moisture sensitivity level of the BMA250 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 (Ts_{max} to Tp)		3° C/second max.
Preheat – Temperature Min (Ts_{min}) – Temperature Max (Ts_{max}) – Time (ts_{min} to ts_{max})		 150 °C 200 °C 60-180 seconds
Time maintained above: – Temperature (T_L) – Time (t_L)		 217 °C 60-150 seconds
Peak/Classification Temperature (Tp)		260 °C
Time within 5 °C of actual Peak Temperature (tp)		20-40 seconds
Ramp-Down Rate		6 °C/second max.
Time 25 °C to Peak Temperature		8 minutes max.

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

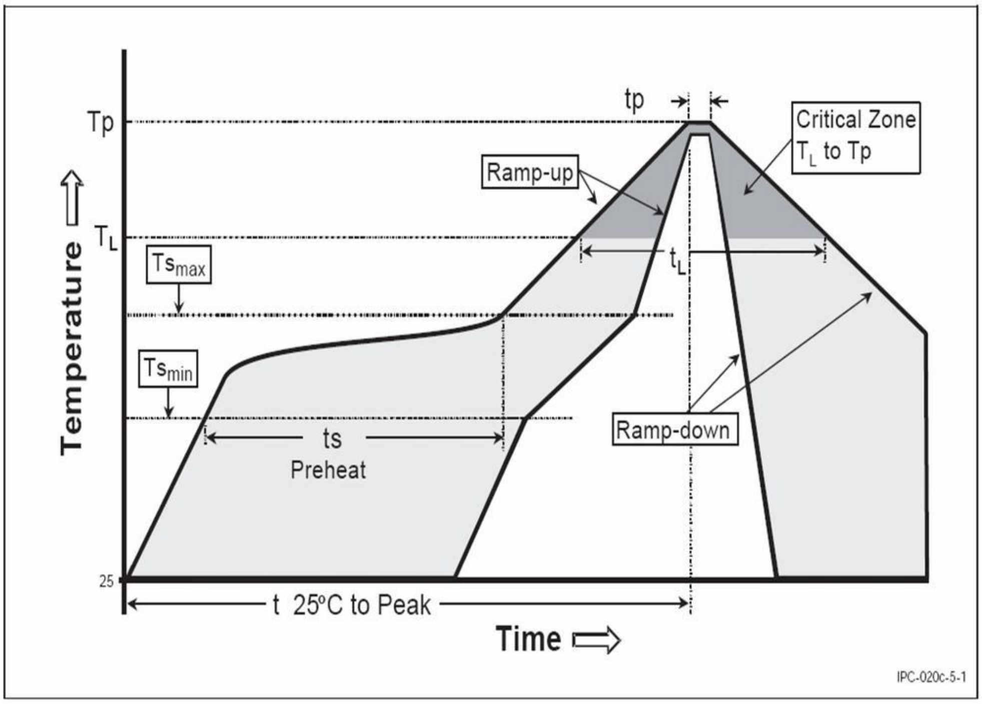


Figure 26: Soldering profile

8.6 Handling instructions

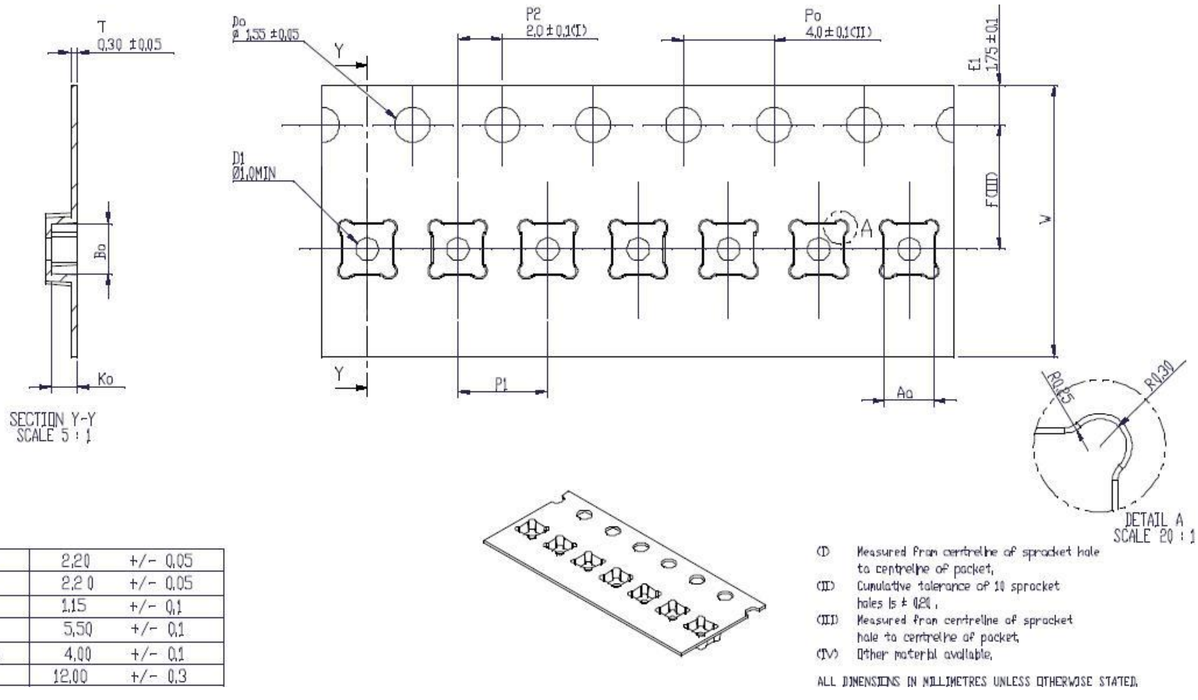
Micromechanical sensors are designed to sense acceleration with high accuracy even at low amplitudes and contain highly sensitive structures inside the sensor element. The MEMS sensor can tolerate mechanical shocks up to several thousand g's. However, these limits might be exceeded in conditions with 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

The BMA250 is shipped in a standard cardboard box.
The box dimension for 1 reel is: L x W x H = 35cm x 35cm x 6cm
BMA250 quantity: 10,000pcs per reel, please handle with care.



Ao	2.20	+/- 0.05
Bo	2.20	+/- 0.05
Ko	1.15	+/- 0.1
F	5.50	+/- 0.1
P1	4.00	+/- 0.1
W	12.00	+/- 0.3

Figure 27: Tape and reel dimensions in mm

8.7.1 Orientation within the reel

→ Processing direction →

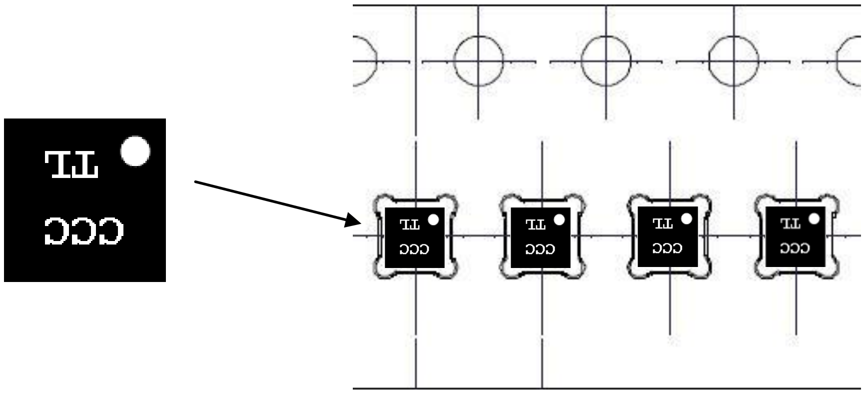


Figure 28: Orientation of the BMA250 devices relative to the tape

8.8 Environmental safety

The BMA250 sensor meets the requirements of the EC restriction of hazardous substances (RoHS) directive, see also:

Directive 2002/95/EC of the European Parliament and of the Council of 27 January 2003 on the restriction of the use of certain hazardous substances in electrical and electronic equipment.

8.8.1 Halogen content

Results of chemical analysis indicate that the BMA250 contains less than 900ppm (by weight) of Fluorine, Chlorine, Iodine and Bromine (i.e. < 50ppm per each substance). Therefore the BMA250 can be regarded as halogen-free. For more details on the analysis results please contact your Bosch Sensortec representative.

8.8.2 Internal package structure

Within the scope of Bosch Sensortec's ambition to improve its products and secure the mass product supply, Bosch Sensortec qualifies additional sources (e.g. 2^{nd} source) for the LGA package of the BMA250.

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 BMA250 product.

9. Legal disclaimer

9.1 Engineering samples

Engineering Samples are marked with “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

Revision	Chapter	Description of modification/changes	Date
0.8		Document release	17 December 2010
0.9	1	Update table 1	26 January 2011
	4.2.2	Added missing table numbers	
	4.3	Update table 7	
	4.4.1	Update range register 0x0F	
1.0		Document rev. 1.0 update / no changes	03 March 2011
1.05	4.8.3	Typo correction, int1_od, int2_od	17 June 2011
	5.11	Typo correction, register 0x25	
	5.11	Typo correction in table 53 description	
1.10	5.2	Typo correction register map	02 November 2011
1.15	4.8.7	Update orientation interrupt	31 May 2012
	6.2	Update I2C address selection	

Bosch Sensortec GmbH
Gerhard-Kindler-Strasse 8
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-BMA250-DS002-05
Version_1.15_052012