Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

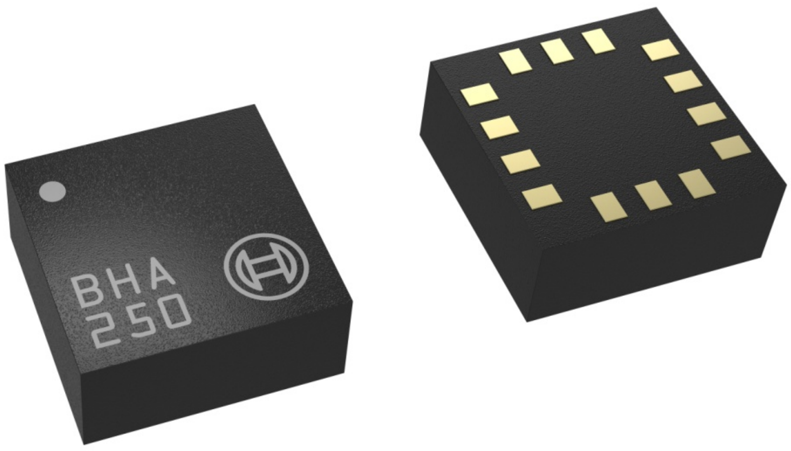
Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

Data Sheet

BHA250 / BHA250B
Ultra low-power sensor hub incl. integrated Accel

Bosch Sensortec

BOSCH
Invented for life



Data Sheet

Document revision	1.2
Document release date	Mar 2017
Document number	BST-BHA250(B)-DS000-01
Technical reference code(s)	BHA250: 0 273 141 231 BHA250B: 0 273 141 310
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.

Features

- All-in-one smart-hub solution for always-on motion sensing at a fraction of current consumption which is commonly required using discrete components.
- 32-bit floating-point microcontroller (Fuser Core). Optimized for data fusion, motion sensing and activity recognition at ultra low power consumption. All in order to offload the power hungry data processing from the main application processor to the smart-hub.
- Powerful BSX sensor fusion library integrated in ROM for lowest design-in effort and fastest time-to-market.
- Additional software and algorithms for RAM processing, provided as ready to use FW patch files. Visit our web site to check available downloads.
- Onboard calculation power for data fusion, 3D- and absolute orientation, rotation vector, quaternions and Euler angles.
- Gesture recognition of significant motion, tilt, pickup, wake up and glance. Enabling customer specific gesture based HMI interfaces for smartphones and wearables.
- Activity recognition of standing, walking, running, biking and in vehicle. Enabling health & fitness applications or any other use case where highly accurate and reliable detection and/or monitoring of user activities is required.
- Step detection and step counting.
- Android 5 / L / Lollipop & Android 6 / M / Marshmallow (non-HiFi) support, incl. batching with dual FIFO buffer for wakeup and non-wakeup events. Implements the full Android sensor stack although an Android OS or any other Android environment is not required.
- High speed I2C interface, with data rates up to 3.4 MBit/s for power-efficient data transfer.
- Highly configurable internal RAM for either feature extension and/or FIFO data buffering.
- SW / FW based functionality. Can be updated, optimized, customized or upgraded with totally new features to support future requirements.
- Smart-hub plus microcontroller, MEMS sensors and software all highly integrated in one 2.2x2.2x0.95 mm3 LGA package with extension interface for additional sensors.

Implemented Sensor Types

With integrated acceleration sensor only:
Accelerometer, Step counter, Step detector, Significant motion, Tilt gesture, Pickup gesture, Wake up gesture, Glance gesture, Activity recognition

With attached gyroscope:
Gravity, Linear acceleration, Gyroscope, Gyroscope uncalibrated, Game rotation vector

With attached magnetometer:
Geomagnetic field, Magnetic field uncalibrated, Orientation, Rotation vector, Geomagnetic rotation vector

General Description

The BHA250(B) is a small, low-power smart-hub with an integrated three axis accelerometer plus a programmable microcontroller, all specifically designed to enable always-on motion sensing. On top it contains software and algorithms for motion-step-, gesture- and activity recognition. The overall concept perfectly matches the requirements of smartphones, wearables or any other application which demands highly accurate, real-time motion data at very low power consumption.
The device integrates our millionfold proven 14bit acceleration sensor with a microcontroller – the new Bosch Sensortec Fuser core. It is bringing you the full Android sensor stack inside your devices – even without having an Android OS or an Android environment. Combining this with the built in computing power and the highly configurable on-board memory the BHA smart-hub offers you a low power solution for motion sensing and data processing.

Target applications

- Activity recognition of standing, walking, running, biking or in vehicle
- HMI interfaces incl. gesture detection of motion, tilt, pickup, wake up and glance
- Step detection and step counting
- Indoor navigation, PDR
- Augmented reality, immersive gaming
- Tilt compensated eCompass and orientation

Target devices

- Mobile phones and tablets
- Wearables such as smart watches, wrist- or neck-bands
- Smart-sports and smart-fitness devices
- Hearables, smart earphones and other head worn devices
- Smart-TV- or AR/VR controllers
- Smart-pens

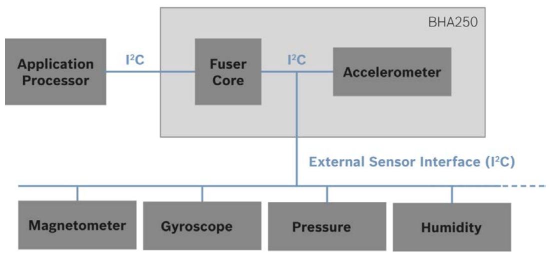


Table of Contents

1. Specification

1.1 Electrical specification

Table 1: Operating Conditions

Operating Conditions BHA						
Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Unit**
Supply Voltage Internal Domains	V_{DD}		1.62	2.4	3.6	V
Supply Voltage I/O Domain	V_{DDIO}		1.6	2.4	3.3	V
Voltage Input Low Level	$V_{IL,a}$		0		$0.3V_{DDIO}$	V
Voltage Input High Level	$V_{IH,a}$		$0.7V_{DDIO}$		V_{DDIO}	V
Voltage Output Low Level	$V_{OL,a}$	I_{OL}=1mA			0.3V	V
Voltage Output High Level	$V_{OH,a}$	I_{OH}=-1mA,	V_{DDIO}-0.3V			V
Operating Temperature	T_A		-40		+85	°C

1.2 Electrical and physical characteristics, measurement performance

All parameters defined for operating conditions (unless otherwise specified)

Table 2: Electrical characteristics Fuser Core

Operating Conditions Fuser Core						
Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Units**
Regulator Output Voltage	V_{REG}		1.0	1.1	1.2	V
Power on Reset Threshold	V_{POR}	VREG>VPOR		VREG-125mV		V
Current Consumption, Run1	I_{RUN}	0°C to +40°C, (1)		800		uA
Current Consumption, Normal Operation2	I_{OPER}	0°C to +40°C, (2)		350		uA
Current Consumption, Sleep3	I_{SLEEP}	0°C to +40°C, (3)		40		uA
Current Consumption, Deep Sleep4	I_{DSLEEP}	0°C to +40°C, (4)		7		uA
Current Consumption, Idle5	I_{IDLE}	0°C to +40°C, (5)		6		uA

Notes:

(1) Current consumption when CPU is running and executing from ROM.

(2) Current consumption in normal operation is average consumption for 9DoF Sensor Fusion with ODR of 100 Hz
(3) Sleep mode is entered when CPU and I2C are idle and timer and system clock are enabled
(4) In Deep Sleep mode, only timer is enabled while system clock is disabled
(5) In Idle mode, no operations are performed, all oscillators are disabled

Table 3: Electrical characteristics accelerometer

OPERATING CONDITIONS ACCELEROMETER						
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

OUTPUT SIGNAL ACCELEROMETER						
Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Units**
Resolution				14		bit
Sensitivity	S_{2g}	g_{FS2g}, T_A=25°C		4096		LSB/g
	S_{4g}	g_{FS4g}, T_A=25°C		2048		LSB/g
	S_{8g}	g_{FS8g}, T_A=25°C		1024		LSB/g
	S_{16g}	g_{FS16g}, T_A=25°C		512		LSB/g
Sensitivity Temperature Drift	TCS_a	g_{FS2g}, Nominal V_{DD} supplies		±0.02		%/K
Sensitivity Supply Volt. Drift	$S_{VDD,a}$	g_{FS2g}, T_A=25°C, $V_{DD\_min} \leq V_{DD} \leq V_{DD\_max}$		0.05		%/V
Zero-g Offset	Off	g_{FS2g}, T_A=25°C, nominal V_{DD} supplies, over life-time		±80		mg
Zero-g Offset Temperature Drift	TCO_a	g_{FS2g}, Nominal V_{DD} supplies		±1		mg/K
Zero-g Offset Supply Volt. Drift	$Off_{VDD,a}$	g_{FS2g}, T_A=25°C, $V_{DD\_min} \leq V_{DD} \leq V_{DD\_max}$		0.5		mg/V
Nonlinearity	NL_A	Best fit straight line, g_{FS2g}		±0.5		%FS
Output Noise Density	$n_{rms,a}$	g_{FS2g}, T_A=25°C, nominal V_{DD} supplies, Normal mode		150		μg/√Hz

1.3 Absolute maximum ratings

Table 4: 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	3.6	V
Voltage at any Logic Pin	Non-Supply Pin	-0.3	V_{DDIO}+0.3	V
Passive Storage Temp. Range	≤65% rel. H.	-50	+150	°C
None-volatile memory (NVM) Data Retention	T = 85°C, after 15 cycles	10		y
Mechanical Shock	Duration 200 µs, half sine		10,000	g
	Duration 1.0 ms, half sine		2,000	g
	Free fall onto hard surfaces		1.8	m
ESD	HBM, at any Pin		2	kV
	CDM		500	V
	MM		100	V

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

2. Pin Connections and description

Figure 1: Pin Connections

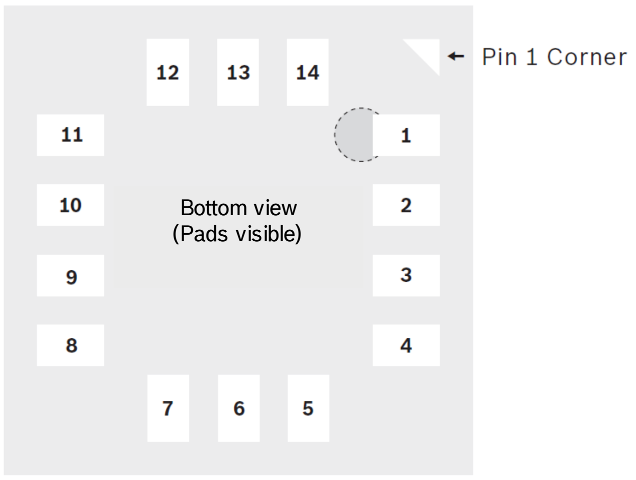


Table 5: Pin description

Pin	Name	Description
1	INT	Host interrupt
2	SCK	I^{2}C serial clock (Host interface)
3	ASCK	I^{2}C Master serial clock, for connecting to external sensors
4	ASDA	I^{2}C master serial data, for connecting to external sensors
5	VREG	Regulator filter capacitor connection
6	GPIO1	Application specific I/O pin[1)]
7	RESV1	Do not connect pin (reserved)
8	GPIO2	Application specific I/O pin
9	GND	Analog power supply ground
10	SA_GPIO7	Select I^{2}C address & Application specific I/O pin refer to section 4.1 page 14
11	GNDIO	Digital I/O power supply ground
12	VDD	Analog power supply voltage (1.71V ... 3.6V)
13	VDDIO	Digital I/O power supply voltage (1.6 ... 3.3 V)
14	SDA	I^{2}C serial data (Host interface)

1) GPIO1 is driven low at power up until firmware download is completed and BHA is initialized.

2.1 Connection Diagram

Figure 2: Reference Diagram

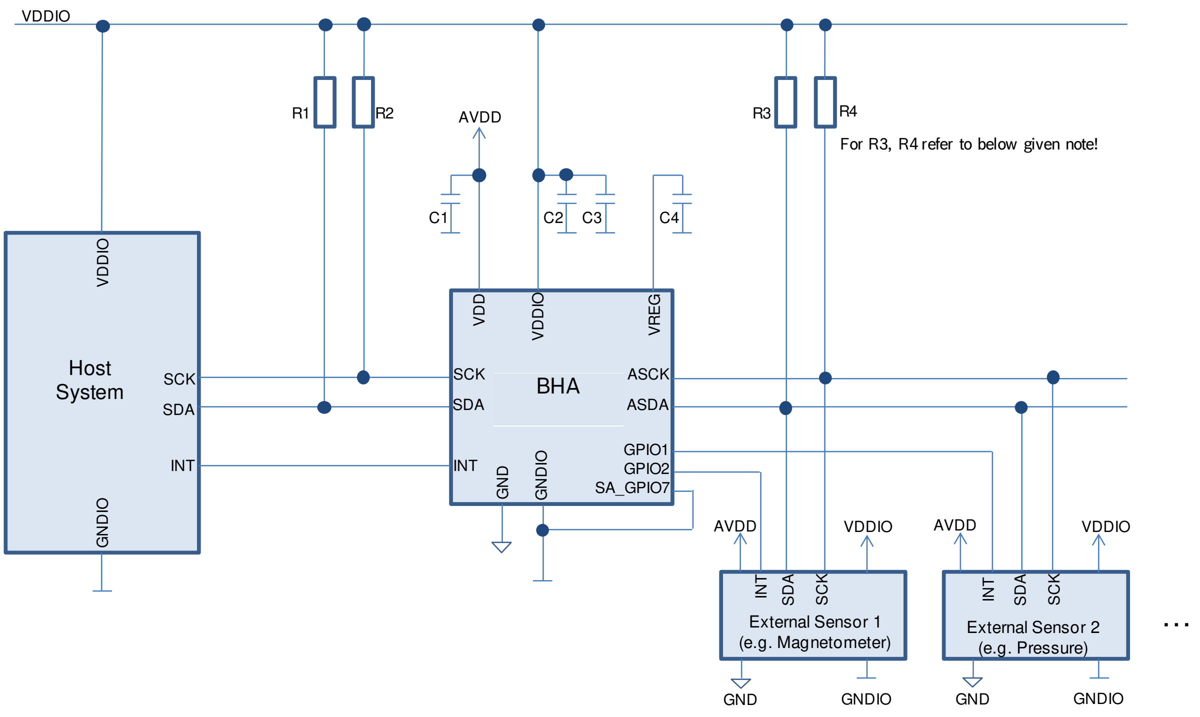


Table 6: Typical values for external circuit components

Component	Value	Remarks
R1	4.7 kΩ	Pull-up resistor for SDA, Host Interface
R2	4.7 kΩ	Pull-up resistor for SCK, Host Interface
R3	4.7 kΩ	Pull-up resistor for ASDA, Aux Interface
R4	4.7 kΩ	Pull-up resistor for ASCK, Aux Interface
C1	100nF	Filter capacitor AVDD
C2	1 µF	Filter capacitor VDDIO
C3	100 nF	Filter capacitor VDDIO
C4	470 nF	Filter capacitor VREG

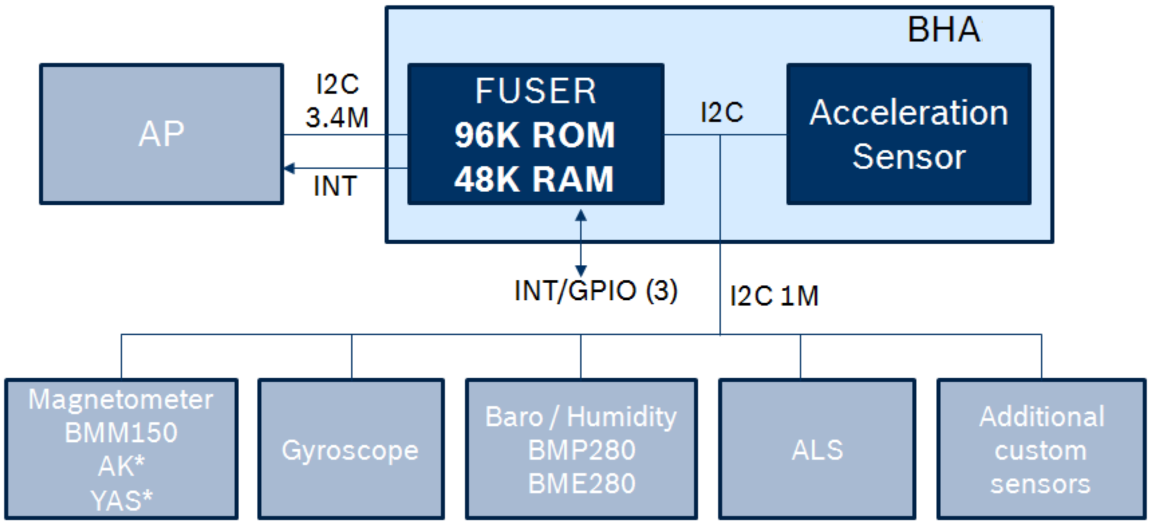
NOTE: R3 and R4 are mandatory, even if no external sensor is attached.

3. Overview

The BHA Sensor hub is a small multichip system in a LGA package consisting of

- a 32-bit floating-point microcontroller (Fuser Core) optimized for sensor fusion and activity recognition
- 96 KByte of ROM including the BSX sensor fusion library
- 48 KByte of RAM for
    - feature extension (e.g. for additional drivers of externally attached sensors)
    - local data buffering (implementing a wake-up and a non-wake-up FIFO as defined in Android
    - feature updates (allowing the updates of features implemented in RAM or ROM to meet future requirement)
- a high speed I^{2}C host interface, with data rates up to 3.4 MBit/s and a host interrupt line
- a fast I^{2}C sensor interface, with data rates up to 1 MBit/s for connection of external sensors
- up to 3 additional GPIO pins

Figure 3: Block Diagram



With these integrated hardware and software features, the low power consumption and the sensor extension interface, the BHA provides an ideal all-in-one solution for always-on sensor applications.

Without any additionally attached sensors the BHA provides a three degrees of freedom (3-DoF) acceleration sensor out of the box, implementing the following Android[1] sensor types:

- Accelerometer
- Step counter
- Step detector
- Significant motion
- Tilt detector
- Pickup gesture
- Wake up gesture
- Glance gesture
- Activity recognition[2] of standing, walking, running, biking, in vehicle

By attaching an external magnetometer to the sensor interface (and configuring the RAM firmware patch to include the sensor driver for the magnetometer) the BHA provides additionally the following sensor types:

- Gravity
- Linear acceleration
- Geomagnetic field
- Magnetic field uncalibrated
- Orientation
- Rotation vector
- Geomagnetic rotation vector

offering a robust eCompass solution to the user.

With further attachment of additional sensors to the sensor interface, as e.g.

- Gyroscope
- Barometic pressure
- Humidity
- Ambient temperature
- Proximity
- Ambient Light

the BHA can provide the full Android sensor stack to the application.

[1] See http://source.android.com/devices/sensors/sensor-types.html for details on defined Android Sensor Types.

[2] Activity recognition is also implemented as a Sensor Type in BHA250, despite not being defined in Android's "sensors.h", but in "activity_recognition.h".

4. Physical Interfaces

4.1 Host interface

According to the interface concept introduced from Android 5 onwards, the BHA provides a high speed I²C interface and a single interrupt line as main interface to the application processor.
The available GPIO pins can be used to implement additional interrupt lines, in case this is necessary for specific applications.

The host interface is implemented as an I²C slave interface, as described in the I²C bus specification created from NXP[3] and implements data transfer rates up to 3.4 Mbit/s in the high-speed mode.
The I²C bus consists of 2 wires, SCK (Serial Clock) and SDA (Serial Data). Both bus lines are bi-directional. The BHA250 can be connected to this bus via SDA and SCL pads with open drain drivers within the device. The bus lines must be externally connected to a positive supply voltage (VDDIO) via a pull-up resistor or current-source.
A data transfer via the I²C slave interface is always initiated by the host. The I²C slave interface can operate as either a transmitter or receiver only, if a valid device address has been received from the host.
The BHA250 responds to device addresses, depending on the logic level applied on the SA_GPIO7. To select the corresponding I2C address keep the desired level for min 10 ns after reset release as described in Table 7. By default there are 2 application specific I/O pins GPIO1 and GPIO2 available and recommended. Special cases might require additional I/O pins. Therefore SA_GPIO was designed to be operated as a third application specific I/O pin, once the I2C address was successfully selected. For details and technical support please refer to corresponding application notes or contact our regional offices, distributors and sales representatives.

Table 7: I2C address selection

SA_GPIO7	I2C address
HIGH	0x29
LOW	0x28

The address and data are transferred between master and slave serially through the data line (SDA) in an 8-bit oriented transfer format. The transfer is synchronized by the serial clock line (SCK). The supported transfer formats are single byte read, multiple byte read, single byte write, multiple byte write. The data line (SDA) can be driven either by the host or the BHA. The serial clock line (SCK) is driven by the host only.

Figure 3 illustrates an example of how to write data to registers in single-byte or multiple-byte mode.

Figure 4: I²C write example

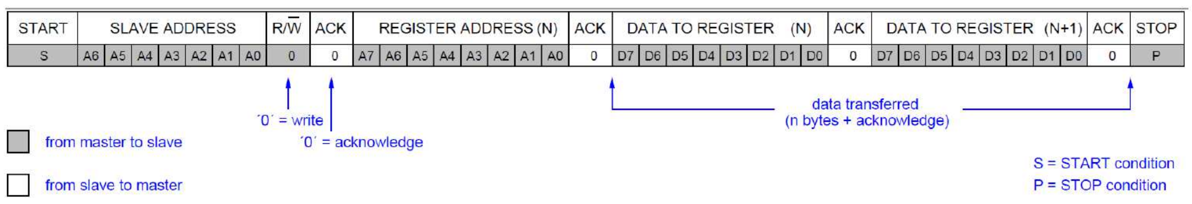
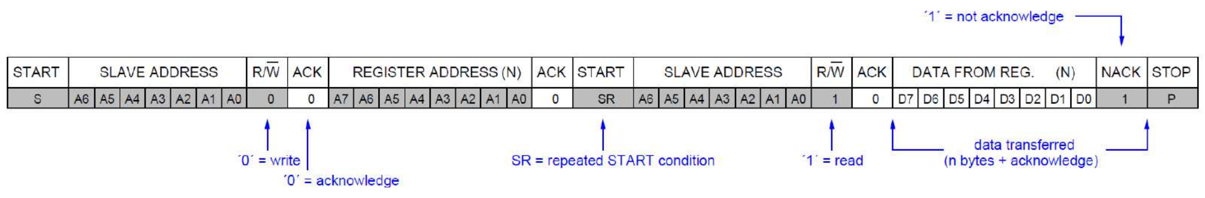


Figure 4 illustrates an example of how to read data to registers in single-byte or multiple-byte mode.

[3] See http://www.nxp.com/documents/user_manual/UM10204.pdf for details

Figure 5: I^{2}C read example



4.2 Sensor interface

The BHA implements a fast-mode plus I^{2}C master interface for connections of external sensors.
This sensor interface is directly connected to the internal acceleration sensor and also available on the auxiliary serial clock (ASCK) and auxiliary serial data (ASDA) pins of the BHA.
The bus lines must be externally connected to a positive supply voltage (VDDIO) via a pull-up resistor or current-source, even if no additional external sensor is attached to the device, in order to enable the proper I^{2}C communication between the Fuser core and the integrated BMA2x2 acceleration sensor.

A common use-case of the sensor interface is the connection of an external magnetometer.
The following external magnetometers are currently supported:

Table 8: Supported Magnetometers

Vendor	**Device**
Bosch Sensortec	BMM150
Asahi Kasei	AK09911/12
Yamaha	YAS532/537

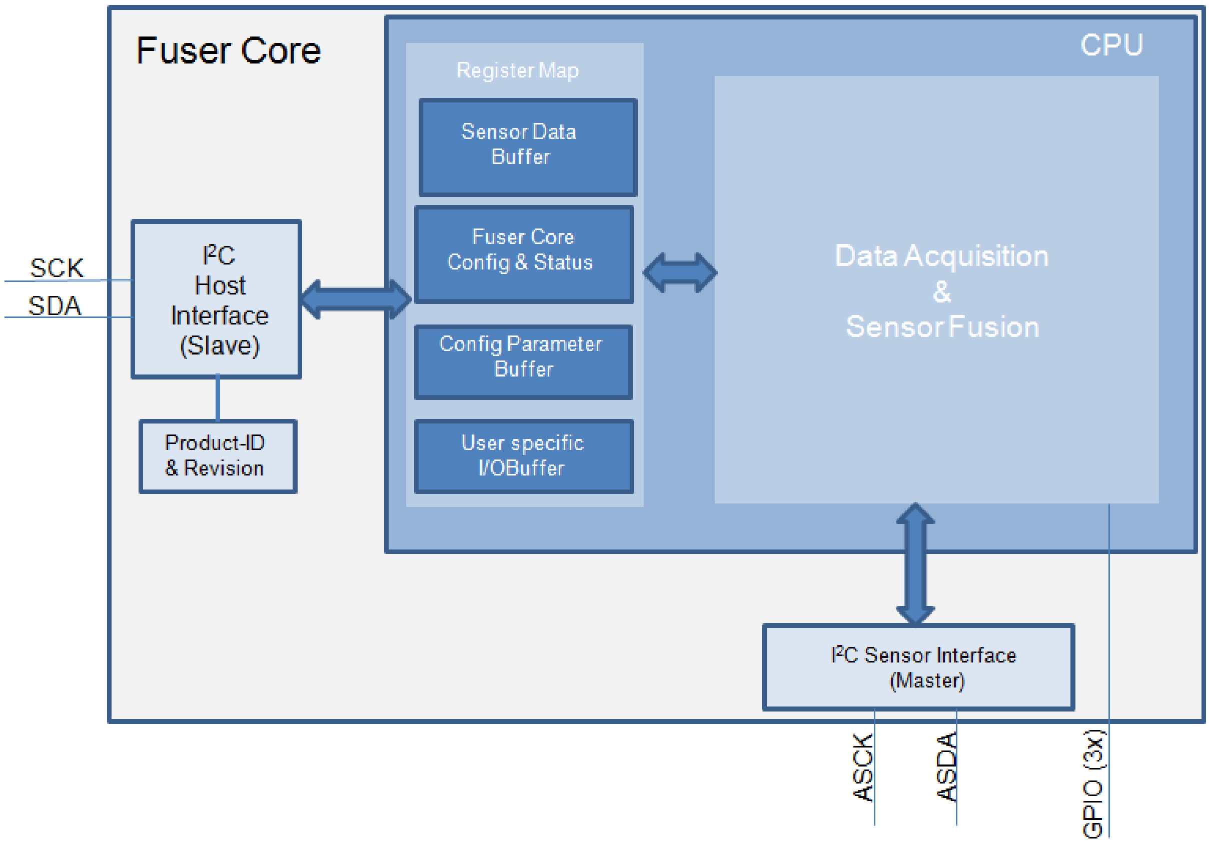
Alternative magnetometers can be supported on customer request.

5. Data interface

5.1 General overview

Figure 5 provides a general overview of the BHA data interface. The software running on the Fuser core obtains the raw sensor from the I²C sensor interface, performs the necessary computations and provides the results into a register map, which forms the main I/O interface to the host from a programmer's point of view.

Figure 6: BHA data interface



The register map consists of 4 main sections:

- Sensor Data Buffer
- Fuser Core Config & Status Buffer
- Configuration Parameter I/O
- User specific I/O Buffer

The **Sensor Data Buffer** consists of 50 register (I²C addresses 0x00:0x31) providing an interface to the Fuser Core's internal Event FiFOs which contain the sensor event data.
Per default the data of both FIFOs (the wake-up and the non-wake-up FIFO) will be mapped to the Sensor Data Buffer, so that the host can read all available data in a burst and identify and separate the data afterwards.
The FIFO_FLUSH register of the Fuser Core Config & Status Buffer can be used to adjust the behavior of the sensor data buffer in a more specific way.

The **Fuser Core & Status Buffer** consist of a register set, which allows the host to control the fundamental behavior of the fuser core as well as getting information on the current status.

The Configuration Parameter I/O interface provides a window in the various configuration options of the sensor system. In consists of the 16 Byte deep Par_Read_Buffer (I^2C addresses 0x3B:0x4A) and the 8 Byte deep Par_Write_Buffer (I^2C addresses 0x5C:0x63) and some additional registers for selecting and mapping the desired parameters into these 2 buffers.

The **User Specific I/O Buffer** is reserved for application specific purposes and can be used to serve the needs of individual applications. It consist of 3 different I^2C address areas (0x4B:0x4F, 0x56:0x5B and 0x65:0x6B) where the first one is read-only, while the others are read-write for the host.

5.2 Register Map

Table 9: BHA Register Map

I2C Adress	Register Name	Access mode	Map section
0x00 – 0x31	Buffer Out[00:49]	Read only	Sensor Data Buffer
0x32	FIFO Flush	Read write	Fuser Core Config & Status
0x33	*Reserved*		
0x34 0x35 0x36 0x37	Chip Control Host Status Int Status Chip Status	Read write Read only Read only Read only	Fuser Core Config & Status
0x38 0x39	Bytes Remaining LSB Bytes Remaining MSB	Read only Read only	Sensor Data Buffer
0x3A	Parameter Acknowledge	Read only	Config Parameter I/O Interface
0x3B – 0x4A	Parameter Read Buffer[0:15]	Read only	Config Parameter I/O Interface
0x4B – 0x4F	GP20 – GP24	Read only	User specific I/O
0x50 – 0x53	*Reserved*		
0x54	Parameter Page Select	Read write	Config Parameter I/O Interface
0x55	Host Interface Control	Read write	Fuser Core Config & Status
0x56 – 0x5B	GP31 – GP36	Read write	User specific I/O
0x5C – 0x63	Parameter Write Buffer[0:7]	Read write	Config Parameter I/O Interface
0x64	Parameter Request	Read write	Config Parameter I/O Interface
0x65 – 0x6B	GP46 – GP52	Read write	User specific I/O
0x6C – 0x6F	Host IRQ Timestamp	Read only	Fuser Core Config & Status
0x70 – 0x71 0x72 – 0x73	ROM Version RAM Version	Read only	Fuser Core Config & Status
0x74 – 0x8F	*Reserved*		

0x90 0x91	Product ID Revision ID	Read only Read only	Chip specific IDs
0x92 – 0x93	*Reserved*		
0x94 – 0x95 0x96 0x97 – 0x9A	Upload Address Upload Data Upload CRC	Read write Read write Read only	Fuser Core Config & Status (Firmware upload interface)
0x9B	Reset Request	Read write	Fuser Core Config & Status

6. Device Initialization and Startup

The procedure in order to initialization and startup the BHA until it reaches its normal operation mode consist mainly of the following steps:

1. Power on or reset the device
2. Wait for Interrupt
3. Upload the Firmware (RAM patch)
4. Switch into main execution mode
5. Wait for Interrupt
6. Configure the sensors and meta events
7. Configure the FIFO buffers
8. Configure the host interrupt setting

Once this procedure is successfully finished, the host can go into sleep mode and wait for the BHA's interrupt, according to the defined conditions (see step 6).

If the host receives an interrupt request from the BHA250, it can simply read out the FIFO buffer and parse the obtained data. (See section 13 for details on how to read the FIFO buffer)

6.1 Reset

The BHA does not provide a specific hardware pin for a reset. A reset can be triggered due to

- Power On Reset
- Watchdog Reset
- Host initiated Reset Request

In order to trigger a reset request, the host has to write a 1 into the Reset_Request register (Address 0x9B in the register map). This bit automatically clears to 0 after reset.

6.2 Boot Mode

The ROM is split into two parts, a small boot loader and the larger set of libraries and drivers which can be used by a RAM-based firmware or "patch."
It is this latter part of the ROM which provides most of the functionality required for sensor fusion, host interface interactions, data batching, and so on. However, without a RAM patch, none of these more advanced behaviors can occur. This is where boot loading comes in.

When the BHA first comes out of reset it executes the ROM boot loader. The boot loader performs the default initialization of the BHA, apply factory trim values, initialization of the host interrupt line, etc, generates an interrupt request to the host and goes into halt mode.

In halt mode, the host may directly load a RAM patch using the firmware update interface registers (Address 0x94-0x9A in the register map) in the Fuser Core Config & Status block.

After the firmware upload procedure is finished successfully, the host can switch the BHA250 into the main execution mode by writing a 1 to bit 0 (CPU_Run_Request) of the Chip Control register (Address 0x34). A successful execution of the CPU_Run_Request can be detected by checking the RAM Version registers (Address 0x72-0x73). Before execution of the RAM patch, the RAM Version registers will contain 0.

6.3 Main Execution Mode

Once in this mode, the full Android host interface and sensor suite is available. The BHA indicates its readiness by inserting an initialized meta event in the FIFO. The host should wait for this before attempting to query or configure sensors or other features.

If an incorrect RAM patch has been loaded (for example, is built for a different sensor suite), the FIFO will instead contain one or more Sensor Error or Error meta events.

In the nominal case, however, the host is now free to query which sensors are present by reading the Sensor Status bits, learn the details of each sensor by querying the Sensor Information parameters, load any Warm Start values using the Algorithm Warm Start parameters, and/or configure sensors to start generating output using the Sensor Configuration parameters.

The host may also wish to configure which meta events will appear in the FIFOs, such as FIFO Overflow, Watermark, or many others. It can specify whether certain meta events can cause an immediate host interrupt, or are batched until later.

Finally, the host may wish to configure the optional Watermark values using the FIFO Control parameter. This allows the host to be informed that either one or both of the FIFOs have reached a level at which the host shall read its contents to avoid data is loss. This is especially useful when the Application Processor is asleep.

7. Device Configuration

A set of registers (the Fuser Core Config & Status block) can be used to configure the fundamental behavior of the CPU core and the host interface (see section 10 for a detailed description of the specific registers). Besides this basic configuration, the full flexibility of the BHA is offered through the Configuration Parameter I/O interface.

The Configuration Parameter I/O interface, is provided through registers of the BHA and consists of

- Parameter_Read_Buffer[0:15] (0x3B – 0x4A)
  in order to read a specific parameter set out the BHA's config parameter area
- Parameter_Write_Buffer[0:7] (0x5C – 0x63)
  in order to write a specific parameter set into the BHA's config parameter area
- Parameter_Page_Select (0x54), Parameter_Request (0x64), Parameter_Acknowledge (0x3A) for the required handshaking.

In general, the Configuration Parameter I/O interface basically copies a specific parameter set either from the parameter area into the read buffer (read access) or from the write buffer into the specified parameter area (write access).

The procedure for a **read** access works a follow:

In order to get the a copy of the desired parameter inside the read buffer, the host requests a parameter set by writing the requested page into the Parameter_Page_Select register and the desired parameter set into the Parameter_Request register.
Afterwards the host waits for an acknowledgement, by polling the Parameter_Acknowledge register until it matches the desired parameter number (or indicates an error). The acknowledgment indicates that the Parameter_Read_Buffer has been updated with the values of the requested parameter.
The host can read more parameters within the same page by writing a new Parameter Request register value, polling for a match in the Parameter Acknowledge register, then reading the new parameter's value from the Parameter Read Buffer area.
The host ends the parameter transfer procedure by writing the Parameter Page Select register with 0.

The procedure for a **write** access works a follow:

The host writes the new data for a specific parameter set into the Parameter_Write_Buffer. In order to address the specific dataset it writes the desired parameter page into the Parameter_Page_Select register and the specific parameter set into the Parameter_Request register.
Afterwards the host waits for an acknowledgement, by polling the Parameter_Acknowledge register until it matches the desired parameter number (or indicates an error). The acknowledgment indicates that the Parameter_Write_Buffer has been copied inside the addressed parameter set.
The host may write another parameter in the same page by repeating the procedure.
The host ends the parameter transfer by writing a 0 into the Parameter_Request register.

A detailed description of the various parameters and their organization into several parameter pages is given in section 11 Parameter I/O Description.

8. FIFOs and Events

Understanding the concept of FIFOs and Events is fundamental for proper operation of the BHA. Both elements are implemented into the device in order to meet the requirements of Android.

Every piece of information the BHA delivers to the host is treated as an Event and placed into a FIFO.

FIFOs

BHA provides two FIFOs: a wakeup and a non-wakeup FIFO.

The **non-wakeup FIFO** will never trigger an interrupt request to the host when the host is in sleep mode (in default configuration).
(The host should inform the BHA about using the AP_SUSPENDED bit (bit 5) in the Host_Interface_Control register (0x55))
If the non-wakeup FIFO is full, while the host is in sleep mode, the non-wakeup FIFO is allowed to overflow, discarding the oldest data to make room for new data as they arrive.

The **wakeup FIFO** may trigger an interrupt request, depending on the current configuration, for different reasons, even if the host is in sleep mode. One obvious reason is to avoid, that the wakeup FIFO overflows (configured by the wakeup FIFO watermark level setting) or the events in the FIFO become too old (configured by the max report latency setting).
There are more reasons, see section 11 and 13 for further details.

Events

In order to implement a generalized and efficient handling mechanism for sensor data the concept of sensor events is used within the BHA.

A sensor event consists of the sensor ID of the virtual sensor generating the event and, the data according to the data type of the specific sensor. It is placed into a FIFO, when it occurs.
Events can be generated continuously, e.g. if a virtual sensor is setup to produce data samples on a configured data rate, or as single events, e.g. when a step or significant motion is detected.

To make use of the event concept in a generalized way, the virtual sensor IDs – which are originating from (and thus are identical to) the virtual sensor definitions in the Android CDD – are extended by additional IDs not necessary related to virtual sensors.

In a first step, each virtual sensor gets a second sensor ID in order to distinguish wakeup from non-wakeup events.
In a second step, additional event IDs are introduced in order to handle non sensor related information, like timestamps and meta events.

A detailed description of all available event IDs is provided in the following sections.

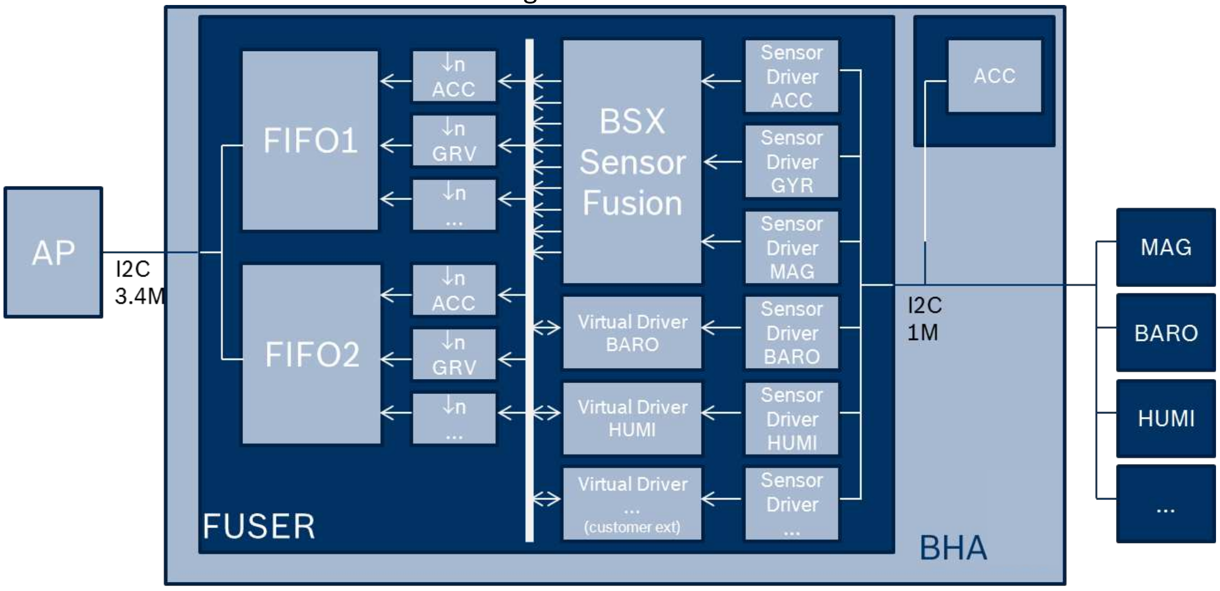
Using this event concept, the BHA's output data will be sent in a continuous stream, with each event (e.g. a sensor sample) uniquely identified. Because many sensors produce output at the same time, the timestamp event is only introduced once into the FIFOs at the start of a series of sensor samples that occurred at the same time. This saves space in the buffer.

9. Functional description

9.1 Dataflow of Sensor Fusion

The integrated Fuser Core receives *raw sensor data* from the connected sensors and provides *calibrated (virtual) sensor data* to the application processor. The raw sensor data flows from the sensor with a maximum ODR 200 Hz to the fuser core. The BSX library runs sensor fusion (when required) using this high speed data, to avoid loss in signal quality. The results are subsampled to the ODR required by the host processor.

Figure 7: Dataflow



9.2 Supported data rates of BSX Sensor Fusion Engine

The following output data rates configuration can be selected by the host processor, these are support in both sensor only and data fusion operating modes:

Table 10: Supported BSX output data rates

BSX output data rate
200Hz
100Hz
50Hz
25Hz 12.5Hz

The actual output data rate requested by Android will be provided according to the Android requirements and derived from the above mentioned internal data rates. I.e., the actual output data rate will be in the range of 90%...210% of the requested data rate. Output samples are generated by subsampling from a suitable data rate from Table 10.

If multiple virtual sensors with different output data rates are requested by Android, the internal data rate will be selected such that all output data rates can be generated according to the Android requirements.

9.3 Gesture recognition

Android defines 3 gestures have to be implemented in the system, but leaves the functional implementation open to the device provider. The BHA allows individual gestures to be implemented and mapped on the above mentioned system gestures. By default the firmware of the BHA performs the following mapping of gestures to the virtual gesture sensors:

- **Wakeup Gesture**

  Double Tap on the device

- **Pickup Gesture**

  Pickup the device from a surface (table) and hold it in a 45° angle in relation to gravity

- **Glance Gesture**

  Move (Slide) the phone left and right on the surface (table) without lifting it up

9.4 Power Modes and Current Consumption

The power modes of the Fuser Core and the connected sensors are configured automatically, depending on which virtual sensors are requested by the host, and the resulting fusion mode.

After startup, the list of requested sensors is empty, i.e. the firmware switches the connected sensors into standby mode and then also sets the processor to sleep. Once a virtual sensor has been requested by the host, the requested physical sensors is enabled and the BSX library is set into a working mode that supports the requested sensors.

In general, the uC can be in operation or in sleep. Each interrupt will wake the uC out of sleep, e.g. to process a new data sample from a sensor. When the processing is complete the uC returns to sleep again.

The current consumption of the Fuser Core in sleep mode is ~7µA, in full operation it is ~800µA. The actual average current consumption therefore depends of the amount of time the uC is in full operation mode, which in turn depends on the selected operation mode.

Current Consumption per operation mode

The following exemplary average current consumptions can be reached in the various operation mode of the BHA, within an isolated use case consideration. The total value includes both the processing in the Fuser Core and the MEMS Sensor power consumption (including the intrinsic ASIC and an estimate for the external magnetometer sensors, where required).

Please note that there is no linear addition of the exemplary values given in the following table if the use cases are not isolated but combined. In this case the resulting total current consumption is always lower than its single fractions.

Table 11: Power consumption vs. operating mode

		Current Consumption (µA Typical)
Use Case	**Device**	**BHA250**
Significant Motion	Accelerometer	50
	Fuser Core	50
	Total	**100**
Step Counting	Accelerometer	50
	Fuser Core	50
	Total	**100**
Activity recognition	Accelerometer	50
	Fuser Core	150
	Total	**200**
Rotation Vector eCompass	Accelerometer	200
	Magnetometer	300
	Fuser Core	300
	Total	**800**
Standby	Accelerometer	3
	Magnetometer	1
	Fuser Core	7
	Total	**11**

9.5 Virtual Sensors

Virtual sensors are the interface to the Android application layer and are directly requested from there. The BHA supports all Virtual Sensors as defined in the Android CDD. Based on the Android specification virtual sensor are completely independent. So each virtual sensor has its own data rate (delay), type, and trigger mode.

Virtual sensors are implemented as software modules within the firmware running on the Fuser Core. Each virtual sensor SW module may access a physical sensor via its sensor driver, or it may use other SW modules (e.g. the BSX library) in order to derive processed data based on physical sensors.

The virtual sensors generate sensor events, which may be continuously (e.g. samples at a configured data rate) or single events (on-change, one-shot, or special; e.g. when a step or significant motion has been detected). These events are represented as data packets of a specific virtual sensor data type and are put into the output FIFO of the BHA.

Each of the sensors will be supported as wakeup and non-wakeup sensor and has a fixed ID which allows distinguishing a wakeup from a non-wakeup version. Each version of a sensor has independent sample rate and report latency values.

The supported virtual sensors are listed the following table: