BMA490L

High-performance longevity acceleration sensor

BMA490L – Data Sheet

Document revision	2.1
Document release date	March 2021
Document number	BST-BMA490L-DS000-03
Sales Part number	0273 017 021
Notes	Data and descriptions in this document are subject to change without notice. Product photos and pictures are for illustration purposes only and may differ from the real product appearance

BMA490L - Basic Description

BMA490L is a high-performance longevity acceleration sensor with extended availability of up to ten years[1]. It is a 16 bit, digital, triaxial acceleration sensor with intelligent on-chip motion-triggered interrupt features optimized for industrial applications.

Key features:

• Small package size	LGA package (12 pins), footprint 2mm x 2mm, height 0.95 mm
• Digital Interface	SPI (4-wire, 3-wire), I²C, 2 interrupt pins, V_{DDIO} voltage range: 1.2V to 3.6V
• Programmable functionality	Acceleration ranges ±2g/±4g/±8g/±16g Low-pass filter bandwidths 684Hz -<8Hz up to a max. output data read out of 1.6 kHz
• On-chip FIFO	Integrated FIFO on sensor with 1 kb
• On-chip interrupt features	Any-/No-Motion interrupt
• Ultra-low power	Low current consumption of data acquisition and all integrated features
• (Secondary) Auxilary Interface	Hub for ext. Magnetometer and data synchronization
• ROHS complaint, halogen free	

Typical applications:

- Industrial IoT (IIoT), e.g. predictive maintenance, vibration monitoring
- Logistics, e.g. asset tracking
- Agricultural and industrial robots, e.g. orientation detection, tilt detection
- White goods and home appliances, e.g. vibration monitoring, power management
- Power tools, e.g. power management, device level detection

[1] See longevity disclaimer on the last page of this document.

Index of Contents

1. Specification

Unless stated otherwise, the given values are over lifetime, operating temperature and voltage ranges. Minimum/maximum values are ±3σ.

1.1 Electrical Specification

Table 1: Electrical Parameter specification

OPERATING CONDITIONS						
Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Units**
Acceleration Range	g_{FS2g}	Selectable via serial digital interface		±2		g
	g_{FS4g}			±4		g
	g_{FS8g}			±8		g
	g_{FS16g}			±16		g
Supply Voltage Internal Domains	V_{DD}		1.62	1.8	3.6	V
Supply Voltage I/O Domain	V_{DDIO}		1.2	1.8	3.6	V
Voltage Input Low Level	V_{IL}	SPI & 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}<=2mA, SPI			$0.2V_{DDIO}$	-
		V_{DDIO}<1.62V, I_{OL}<=1.5mA, SPI			$0.2V_{DDIO}$	-
Voltage Output High Level	V_{OH}	V_{DDIO}>=1.62V, I_{OH}<=2mA, SPI	$0.8V_{DDIO}$			-
		V_{DDIO}<=1.62V, I_{OH}<=1.5mA, SPI	$0.8V_{DDIO}$			-
Total Supply Current in Performance mode	I_{DD}	Nominal V_{DD} and V_{DDIO}, 25°C, g_{FS4g}		150		µA
Total Supply Current in Suspend Mode	I_{DDsum}	Nominal V_{DD} and V_{DDIO}, 25°C		3.5		µA
Total Supply Current in Low-power Mode	I_{DDlp1}	Nominal V_{DD} and V_{DDIO}, 25°C 50 Hz ODR		14		µA
Power-Up Time	ts_up				1	ms
Non-volatile memory (NVM) write-cycles	n_{NVM}				15	cycles
Operating Temperature	T_A		-40		+85	°C

OUTPUT SIGNAL						
Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Units**
Sensitivity	S_{2g}	g_{FS2g}, T_A=25°C		16384		LSB/g
	S_{4g}	g_{FS4g}, T_A=25°C		8192		LSB/g
	S_{8g}	g_{FS8g}, T_A=25°C		4096		LSB/g
	S_{16g}	g_{FS16g}, T_A=25°C		2048		LSB/g
Sensitivity Temperature Drift	TCS			0.005		%/K
Resolution (in ±2g range)				0.06		mg
Zero-g Offset	Off	Nominal V_{DD} and VDD_{IO}, 25°C, g_{FS4g}		±30		mg
Zero-g Offset Temperature Drift	TCO			0,25		mg/K
Output Data Rate	ODR_{PERF}	Performance mode	12.5		1600	Hz
Output data rate and BW in Performance mode	$ODR_{12.5}$	3dB cutoff frequency of the accelerometer according to ODR with normal filter mode		5.06		Hz
	ODR_{25}			10.12		Hz
	ODR_{50}			20.25		Hz
	ODR_{100}			40.5		Hz
	ODR_{200}			80		Hz
	ODR_{400}			162 (155 for Z axis)		Hz
	ODR_{800}			324 (262 for Z axis)		Hz
	ODR_{1600}			684 (353 for Z axis)		HZ
Output Data Rate	ODR_{LPM}	Low-power mode	0.78		400	Hz
Nonlinearity	NL	Nominal V_{DD} and VDD_{IO}, 25°C, g_{FS4g}		0.5		%FS
Output Noise Density	n_{dens}	Nominal V_{DD} and VDD_{IO}, 25°C, g_{FS4g}		120		µg/√Hz
Temperature sensor Measurement Range	T_s		-40		+80	°C
Temperature Sensor Slope	dT_S			1		K/LSB
Temperature Sensor Offset	OT_S	at 23°C		1		K
Power Supply Rejection Ratio	PSRR			1		mg/50mV

MECHANICAL CHARACTERISTICS						
Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Units**
Cross Axis Sensitivity	S	relative contribution between any two of the three axes		0,7		%
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	V
	V_{DDIO} Pin	-0.3	4	V
Voltage at any Logic Pin	Non-Supply Pin	-0.3	V_{DDIO}+0.3, <4	V
Passive Storage Temp. Range	≤ 65% rel. H.	-50	+150	°C
None-volatile memory (NVM) Data Retention	T = 85°C, after 15 cycles	10		y
Mechanical Shock	Duration ≤ 200µs		10,000	g
	Duration ≤ 1.0ms		2,000	g
	Free fall onto hard surfaces		1.8	m
ESD, at any pin	HBM		2	kV
	CDM		500	V
	MM		200	V

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

3. Quick Start Guide

The purpose of this chapter is to help developers who want to start working with the BMA490L by giving you some very basic hands-on application examples to get started.

Note about using the BMA490L:

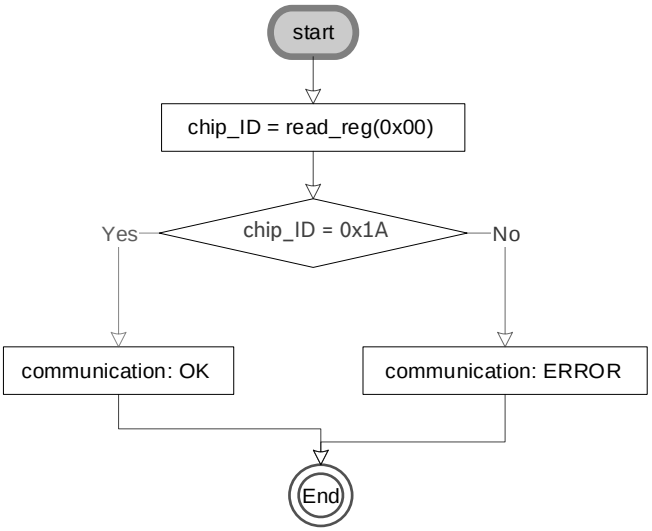
- The communication between application processor and BMA490L will happen either over I2C or SPI interface. For more information about the interfaces, read the related chapter 6 Digital Interfaces.
- Before starting the test, the device has to be properly connected to the master (AP) and powered up. For more information about it, read the related chapter 7 Pin-out and Connection Diagrams.

First application setup examples algorithms:

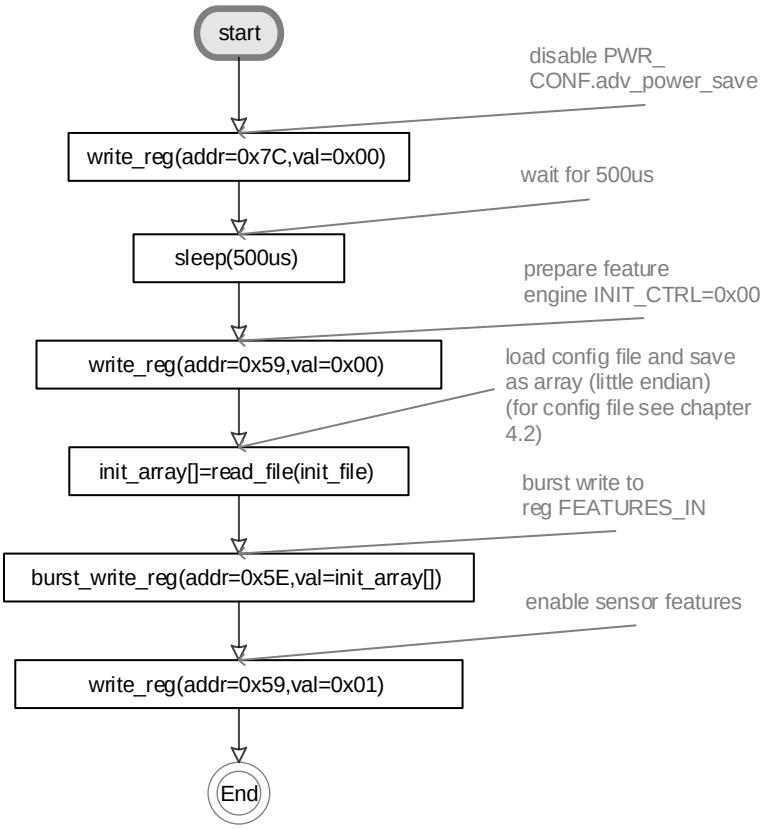
After correct power up by setting the correct voltage to the appropriate external pins, the BMA490L enters automatically into the Power On Reset (POR) sequence. In order to properly make use of the BMA490L, certain steps from host processor side are needed. The most typical operations will be explained in the following application examples in form of flow-diagrams.

Example 1: ***Testing communication with the BMA490L and initializing feature engine***

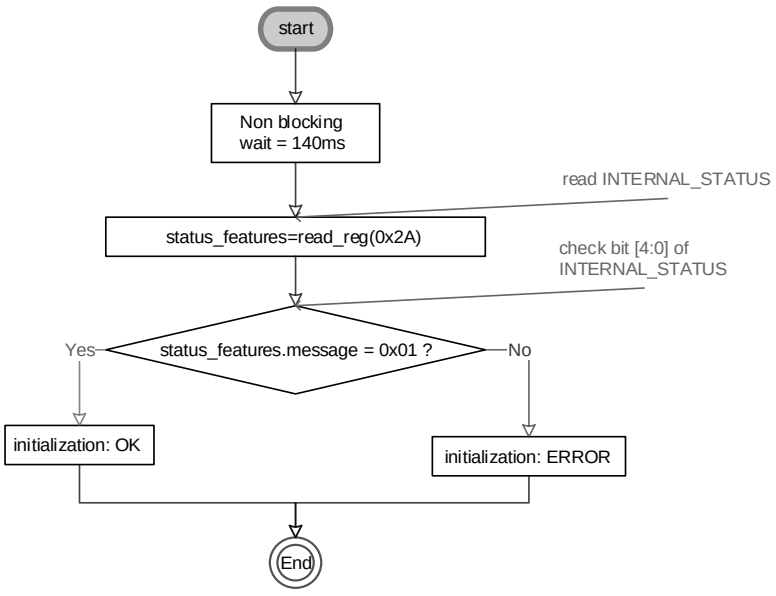
a. -reading chip id (checking correct communication)



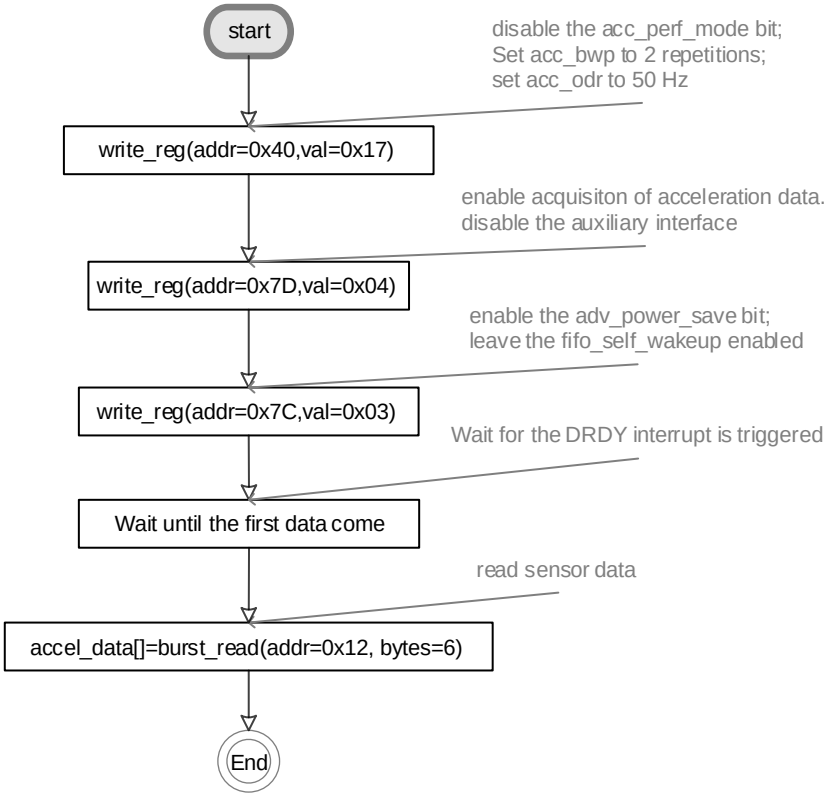
b. -performing initialization sequence (interrupt feature engine)



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



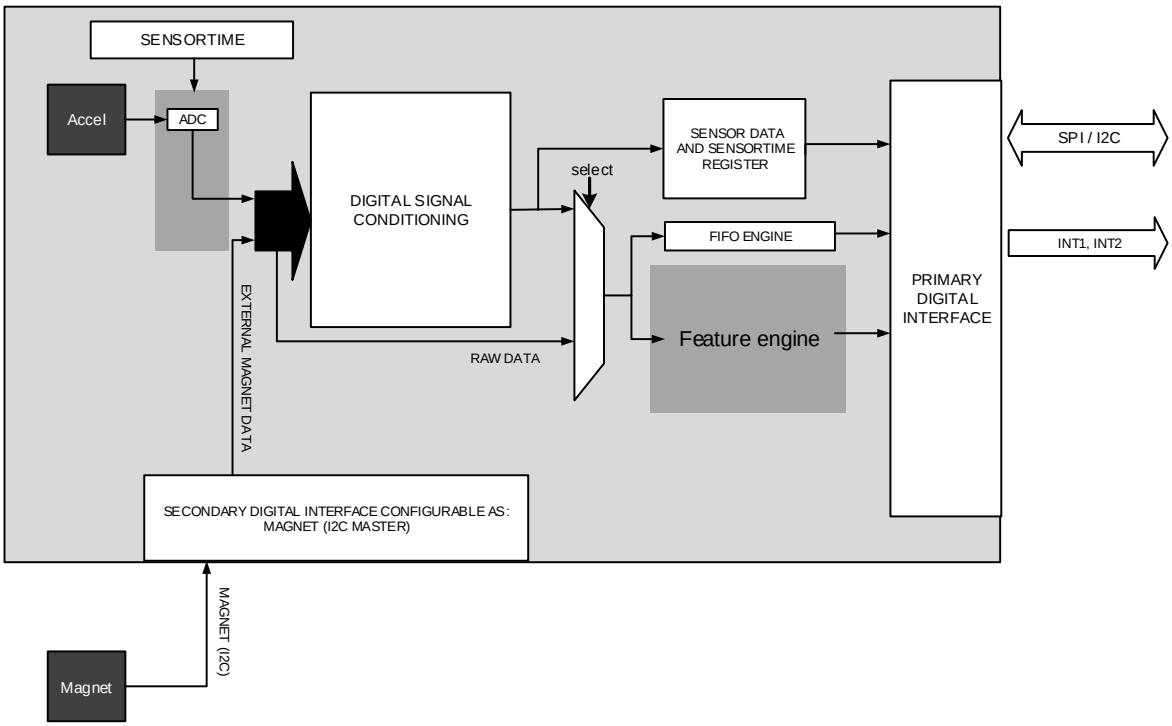
Example 2: ***Reading acceleration data from BMA490L (example: low power mode)***
-setting data processing parameters (power, bandwidth, range) and reading sensor data



Further steps:
The BMA490L has many more capabilities that are described in this document and include FIFO, power saving modes, synchronization capabilities with host processor, data synchronization and integration with third party sensors, many interrupts generation and features like any motion/no motion.

4. Functional Description

4.1 Block Diagram



4.2 Supply Voltage and Power Management

BMA490L has two distinct power supply pins:

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

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

4.3 Device Initialization

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

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

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

4.4 Power Modes

The power state of the BMA490L is controlled through the registers PWR_CONF and PWR_CTRL. The Register PWR_CTRL enables and disables the accelerometer and the auxiliary sensor. The Register PWR_CONF controls which power state the sensors enter if they are enabled or disabled in the Register PWR_CTRL. The power state impacts the behavior of the sensor with respect to start-up time, available functions, etc. but not the sensor data quality. The sensor data quality is controlled in the Registers ACC_CONF.

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

Low Power Mode: This power configuration aggressively reduces power of the device as much as possible. The low power mode configuration is activated through enabling PWR_CONF.adv_power_save=0b1 and disabling ACC_CONF.acc_perf_mode=0b0. In this configuration these externally user visible features may not be available:

- Register writes need an inter-write-delay of at least **450 µs**.
- The sensors log data into the FIFO in performance and low power mode. When the FIFO watermark interrupt is active, the FIFO is accessible for reading in low power mode until a burst read operation on Register FIFO_DATA completes when PWR_CONF.fifo_self_wakeup=0b1. When PWR_CONF.fifo_self_wakeup=0b0, the user needs to disable advanced power save mode (PWR_CONF.adv_power_save=0b0) and wait for 250 µs before reading the FIFO.
- To read out FIFO data w/o a FIFO watermark interrupt, the advanced power save configuration needs to be disabled (PWR_CONF.adv_power_save=0b0)

Table 3: Examples with the optimal power configurations:

Usecase	ACC_CONF.acc_perf_mode	PWR_CONF.adv_power_save	PWR_CTRL.acc_en	Power consumption
Configuration mode	x	0	x	
Suspend (lowest power mode)	x	1	0	suspend power
Performance mode	1	x	1	Accel works in continuous mode
Low power mode	0	1	1	Depends on ACC_CONF

The PWR_CTRL register is used to enable and disable sensors. Per default, all sensors are disabled. Acceleration sensor must be enabled by setting PWR_CTRL.acc_en=0b1.
The auxiliary sensor functionality is supported only when the auxiliary interface is connected for the auxiliary sensor operation. If the auxiliary interface is not used for auxiliary sensor operation, then the auxiliary sensor interface must remain disabled by setting PWR_CTRL.aux_en=0b0 (default).

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

- Set the auxiliary sensor interface to suspend in Register PWR_CTRL.aux_en=0b0. Changing the auxiliary sensor interface power mode to suspend does not imply any mode change in the auxiliary sensor.
- The auxiliary sensor part itself must be put into suspend mode by writing the respective configuration bits of the auxiliary sensor part. The power mode of the auxiliary sensor part is

controlled by setting the BMA490L auxiliary sensor interface into manual mode by AUX_IF_CONF.aux_manual_en=0b1 and then communicating with the auxiliary sensor part through the BMA490L registers AUX_RD_ADDR, AUX_WR_ADDR, and AUX_WR_DATA. For details see Chapter 4.9.

Table 4: Current consumption in low power mode

Current Consumption[2] [µA] depending on number of averaged samples in low power mode								
ODR	No Avg	Avg 2	Avg 4	Avg 8	Avg 16	Avg 32	Avg 64	Avg 128
ODR 0.78	3	3	3	4	4	5	7	12
ODR_1.56	3	3	3	4	4	6	10	15
ODR_3.125	4	4	4	6	8	12	21	39
ODR_6.25	4	5	6	8	13	22	40	77
ODR_12.5	6	7	9	14	23	40	77	152
ODR_25	8	11	14	24	43	79	152	152
ODR_50	14	18	27	45	83	152	152	152
ODR_100	22	32	51	87	152	152	152	152
ODR_200	42	60	97	152	152	152	152	152
ODR_400	80	118	152	152	152	152	152	152

4.5 Sensor Data

4.5.1 Acceleration Data

The width of acceleration data is 16 bits given in two´s complement representation in the registers DATA_8 to DATA_13. The 16 bits for each axis are split into an MSB upper part and an LSB lower part. Reading the acceleration data registers shall always start with the LSB part. In order to ensure the integrity of the acceleration data, the content of an MSB register is locked by reading the corresponding LSB register (shadowing procedure).

4.5.2 Filter Settings

The accelerometer digital filter can be configured through the Register ACC_CONF.

Note:

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

[2] Current consumption based on limited lab measurements. Only for reference.

4.5.3 Accelerometer data processing for performance mode

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

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

Table 5: 3dB cutoff frequency of the accelerometer according to ODR with normal filter mode

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

The noise is also depending on the filter settings and ODR, see table below.

Table 6: Accelerometer noise in mg according to ODR with normal filter mode (range +/- 4g) (based on device measurement)

ODR in Hz	**25**	**50**	**100**	**200**	**400**
RMS-Noise (typ.) [mg]	0.5	0.7	0.9	1.3	1.7

4.5.4 Accelerometer data processing for low power mode

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

averaged samples = $2^{(Val(acc\_bwp))}$
skipped samples = (1600/ODR)-averaged samples

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

4.5.5 Data Ready Interrupt

This interrupt fires whenever a new data sample set from accelerometer, or auxiliary sensor is complete. This allows a low latency data readout. In non-latched mode, the interrupt and the flag in Register INT_STATUS_1 are cleared automatically after 1/(3200Hz). If this automatic clearance is unwanted, latched-mode can be used.
In order to enable/use the data ready interrupt map it on the desired interrupt pin via INT_MAP_DATA.

4.5.6 Temperature Sensor

The temperature sensor has 8 bits. The temperature value is defined in Register TEMPERATURE and updated every 1.28 s.

Table 7: The temperature sensor is always on, when the accelerometer sensor is active.

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

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

4.5.7 Sensor Time

The BMA490L supports the concept of sensortime. Its core element is a free running counter with a width of 24 bits. It increments with a resolution of 39.0625us. The user can access the current state of the counter by reading registers SENSORTIME_0 to SENSORTIME_2.
All sensor events e.g. updates of data registers are synchronous to this sensor time register as defined in the table below. With every update of the data register or the FIFO, a bit *m* in the registers SENSORTIME_0 to SENSORTIME_2 toggles where *m* depends on the output data rate for the data register and the output data rate and the FIFO downsampling rate for the FIFO. The table below shows which bit toggles for which update rate of data register and FIFO

Table 8: Bit m in sensor_time with Resolution in [s]

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

Table 9: Bit m in sensor_time with Resolution in [ms]

Bit *m* in sensor_time	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Resolution [ms]	1280	640	320	160	80	40	20	10
Update rate [Hz]	0.78	1.56	3.125	6.25	12.5	25	50	100

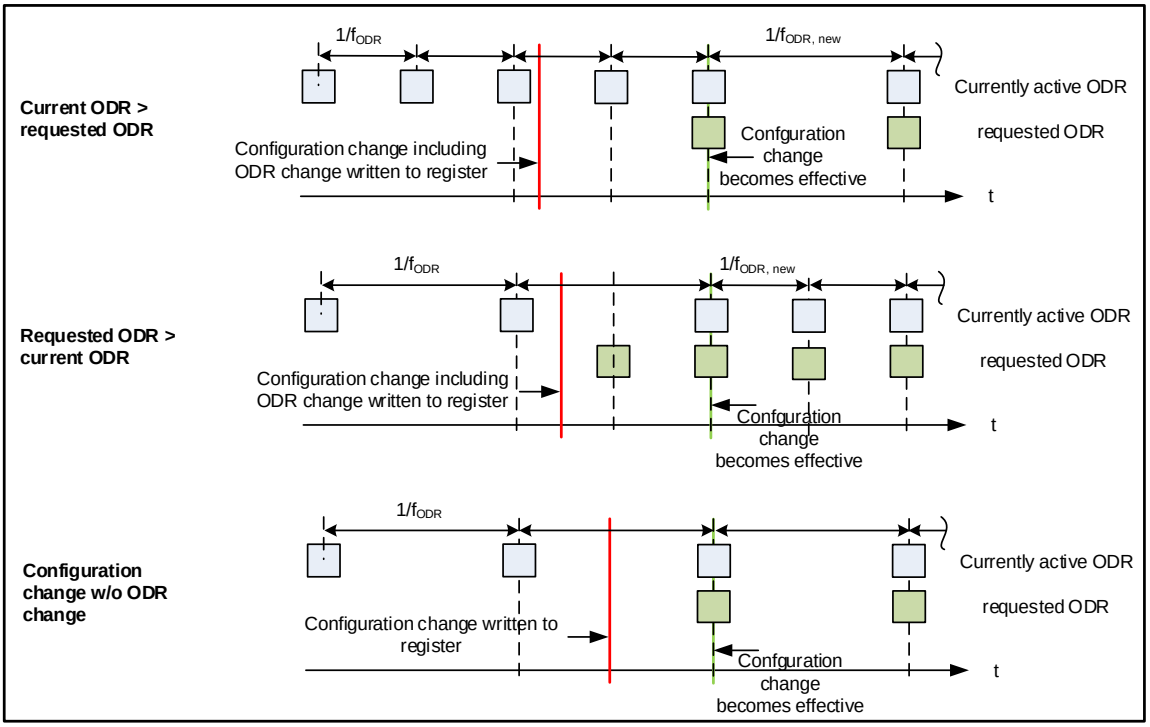
Bit *m* in sensor_time	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Resolution [ms]	5	2.5	1.250	0.625	0.3125	0.156	0.078	0.039
Update rate [Hz]	200	400	800	1600	3200			

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

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

4.5.8 Configuration Changes

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



Due to filter settling, some invalid samples can be suppressed in addition after a configuration change.

4.6 FIFO

The device supports the following FIFO operating modes:

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

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

- FIFO full interrupt
- FIFO watermark interrupt

FIFO is enabled with FIFO_CONFIG_1.fifo_acc_en=0b1 (to enable FIFO for accelerometer data, 0b0= disabled), or set FIFO_CONFIG_1.fifo_aux_en=0b1 (to enable the FIFO for the auxiliary interface (magnetometer), 0b0=disabled).

4.6.1 Frames

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

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

Header mode

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

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

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

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

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

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

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

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

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

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

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

Headerless mode

When the data rates of all enabled sensor elements are identical, the FIFO header may be disabled in FIFO_CONFIG_1.fifo_header_en.

The headerless mode supports only regular frames. To be able to distinguish frames from each other, all frames must have the same size. For this reason, any change in configuration that have an impact to frame size or order of data within a frame will cause an instant flush of FIFO, restarting capturing of data with the new settings.
If the auxiliary sensor interface is enabled, the number of auxiliary sensor bytes in a FIFO frame is always AUX_IF_CONF.aux_rd_burst bytes (see chapter 4.8). If the burst length is less than 8, BMA490L will pad the values read form the auxiliary sensor. E.g. if AUX_IF_CONF.aux_rd_burst=0b01 (2 Bytes), a frame with auxiliary sensor and accelerometer data will look like

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

4.6.2 Conditions and Details

Frame rates

The frame sampling rate of the FIFO is defined by the maximum output data rate of the sensors enabled for FIFO sampling. The FIFO sampling configuration is set in register FIFO_CONFIG_0 to FIFO_CONFIG_1. It is possible to select filtered or pre-filtered data as an input to the FIFO. If un-filtered data are selected in register FIFO_DOWNS.acc_fifo_filt_data for the accelerometer, the sample rate is 1600 Hz. The input data rate to the FIFO can be reduced by selecting a down-sampling factor 2^k in register FIFO_DOWNS.acc_fifo_downs, where k=[0,1..7].

FIFO Overflow

In the case of overflow the FIFO can either stop recording data or overwrite the oldest data. The behavior is controlled by Register FIFO_CONFIG_0.fifo_stop_on_full. When FIFO_CONFIG_0.fifo_stop_on_full *=0b0,* the FIFO logic may delete the oldest frames. If header mode is enabled, the skip frame is prepended at the next FIFO readout, when the free FIFO space falls below the maximum size frame.
If FIFO_CONFIG_0.fifo_stop_on_full =0b1, the newest frame may be discarded, if the free FIFO space falls below the maximum size frame. If header mode is enabled, a skip frame is prepended at the next FIFO readout (which is **not** the position where the frame(s) have been discarded).
During a FIFO read operation of the host, no data at the FIFO tail may be dropped. If the host reads the FIFO with a slower rate than it is filled, it may happen that the sensor needs to drop new data, even when FIFO_CONFIG_0.fifo_stop_on_full =0b0. These events are recorded in the Register ERR_REG.fifo_err.

Control frames

Control frames are only supported in header mode. There are a number of control frames defined through the *fh_parm* field. These are shown in below.
A skip frame indicates the number of skipped frames after a FIFO overrun occurred, a sensortime frame contains the sensortime when the last sampled frame stored in the FIFO is read, a FIFO input config frames indicates a change in sensor configuration which affects the sensor data.
The FIFO fill level is contained in registers FIFO_LENGTH_1.fifo_byte_counter_13_8 and FIFO_LENGTH_0.fifo_byte_counter_7_0 and includes the control frames, with the exception of the sensortime frame.

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

Skip Frame (fh_parm=0x0):
In the case of FIFO overflows, a skip_frame is prepended to the FIFO content, when read out next time. The data for the frame consists of one byte and contains the number of skipped frames. When more than 0xFF frames have been skipped, 0xFF is returned. A skip frame is expected always as first frame in a FIFO read burst.

Sensortime Frame (fh_parm=0x1):
The data for the sensortime frame consists content of the Register SENSORTIME_0 to SENSORTIME_2 when the last byte of the last sample frame was read. A sensortime frame is always expected as last frame in the FIFO. A sensortime frame is only sent if the FIFO becomes empty during the burst read. A sensortime frame does not consume memory in the FIFO. Sensortime frames are enabled (disabled) by setting FIFO_CONFIG_0.fifo_time_en to 0b1 (0b0).

Fifo_Input_Config Frame (fh_parm=0x2):
Whenever the filter configuration of the FIFO input data sources changes, a FIFO input config frame is inserted into the FIFO, before the configuration change becomes active. E.g. when the bandwidth for the accelerometer filter is changed in Register ACC_CONF, a FIFO input config frame is inserted before the first frame with accelerometer data with the new bandwidth configuration. The FIFO input config frame contains one byte of data with the format

Bit	7	6	5	4	3	2	1	0
Content	reserved		aux_ if_ch	aux_ conf_ch	reserved	reserved	acc_ range_c h	acc_ conf_ch

aux_if_ch: A write to Register AUX_IF_CONF, AUX_RD_ADDR, or AUX_WR_ADDR becomes active.
aux_conf_ch: A write to Register AUX_CONF becomes active.
acc_range_ch: A write to Register ACC_RANGE becomes active.
acc_conf_ch: A write to Register ACC_CONF or acc_FIFO_filt_data or acc_FIFO_downsampling in Register FIFO_DOWNS becomes active.