IIS3DWB: ultra-wide bandwidth, low-noise, 3-axis digital vibration sensor

Introduction

This document is intended to provide usage information and application hints related to ST's IIS3DWB vibration sensor.

The IIS3DWB is a high-performance 3-axis MEMS vibration sensor with ultra-wide bandwidth, flat frequency response, low noise and a digital SPI interface standard output.

The device has a dynamic user-selectable full-scale acceleration range of ±2/±4/±8/±16 *g*, a 16-bit data output (26.667 kHz rate) and is capable of measuring accelerations with a signal bandwidth of 6.3 kHz.

Thanks to its ultra-wide bandwidth, flat frequency response up to 6.3 kHz, embedded filtering which eliminates frequency aliasing and its low noise, the IIS3DWB enables out-of-the-box, high quality and cost effective vibration monitoring in industrial applications.

The IIS3DWB is available in a small thin plastic land grid array package (LGA) and is guaranteed to operate over an extended temperature range from -40 °C to +105 °C.

1 Pin description

Figure 1. **Pin connections**

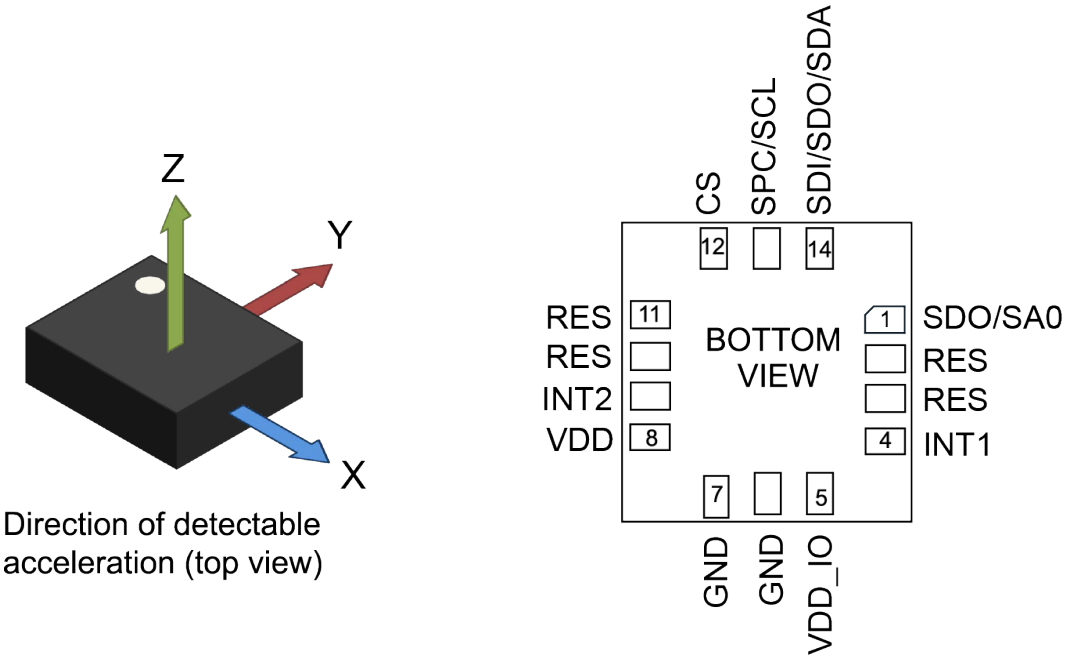


Table 1. Default pin status

Pin#	Name	Function	Default status	Recommended connection
1	SDO/SA0	SPI 4-wire interface serial data output (SDO) I²C least significant bit of the device address (SA0)	Input without pull-up Pull-up is enabled if bit SDO_PU_EN=1 in reg 02h	Application specific
2	RES	Reserved	Input without pull-up	Connect to VDD_IO or GND
3	RES	Reserved	Input without pull-up	Connect to VDD_IO or GND
4	INT1	Programmable interrupt #1	Input with pull-down	Must be set to 0 or left unconnected during device power-up. After device power-up, connection is application specific.
5	VDD_IO	Power supply for I/O pin	-	
6	GND	Ground	-	
7	GND	Ground	-	
8	VDD	Power supply	-	
9	INT2	Programmable interrupt #2	Output forced to GND	Application specific
10	RES	Reserved	Input with pull-up	Connect to VDD_IO or leave pin electrically unconnected and soldered to PCB
11	RES	Reserved	Input with pull-up	Connect to VDD_IO or leave pin electrically unconnected and soldered to PCB
12	CS	I²C/SPI mode selection (1: SPI idle mode / I²C communication enabled; 0: SPI communication mode / I²C disabled)	Input with pull-up Pull-up is disabled if bit I2C_DISABLE=1 in reg 13h	Application specific
13	SPC/SCL	SPI serial port clock (SPC) I²C serial clock (SCL)	Input without pull-up	Application specific
14	SDI/SDO/SDA	SPI serial data input (SDI) 3-wire interface serial data output (SDO) I²C serial data (SDA)	Input without pull-up	Application specific

2 Registers

Table 2. Registers

Register name	Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PIN_CTRL	02h	0	SDO_PU_EN	1	1	1	1	1	1
FIFO_CTRL1	07h	WTM7	WTM6	WTM5	WTM4	WTM3	WTM2	WTM1	WTM0
FIFO_CTRL2	08h	STOP_ON_WTM	0	0	0	0	0	0	WTM8
FIFO_CTRL3	09h	0	0	0	0	BDR_XL_3	BDR_XL_2	BDR_XL_1	BDR_XL_0
FIFO_CTRL4	0Ah	DEC_TS_BATCH_1	DEC_TS_BATCH_0	ODR_T_BATCH_1	ODR_T_BATCH_0	0	FIFO_MODE2	FIFO_MODE1	FIFO_MODE0
COUNTER_BDR_REG1	0Bh	dataready_ pulsed	RST_COUNTER _BDR	0	0	0	CNT_BDR_TH_10	CNT_BDR_TH_9	CNT_BDR_TH_8
COUNTER_BDR_REG2	0Ch	CNT_BDR_TH_7	CNT_BDR_TH_6	CNT_BDR_TH_5	CNT_BDR_TH_4	CNT_BDR_TH_3	CNT_BDR_TH_2	CNT_BDR_TH_1	CNT_BDR_TH_0
INT1_CTRL	0Dh	0	INT1_CNT_BDR	INT1_FIFO_FULL	INT1_FIFO_OVR	INT1_FIFO_TH	INT1_BOOT	0	INT1_DRDY _XL
INT2_CTRL	0Eh	0	INT2_CNT_BDR	INT2_FIFO_FULL	INT2_FIFO_OVR	INT2_FIFO_TH	INT2_DRDY _TEMP	0	INT2_DRDY _XL
WHO_AM_I	0Fh	0	1	1	1	1	0	1	1
CTRL1_XL	10h	XL_EN_2	XL_EN_1	XL_EN_0	0	FS1_XL	FS0_XL	LPF2_XL_EN	0
CTRL3_C	12h	BOOT	BDU	H_LACTIVE	PP_OD	SIM	IF_INC	0	SW_RESET
CTRL4_C	13h	0	0	INT2_on_INT1	0	DRDY _MASK	I2C_disable	0	1AX_TO_ 3REGOUT
CTRL5_C	14h	0	ROUNDING1	ROUNDING0	0	0	0	ST1_XL	ST0_XL
CTRL6_C	15h	0	0	0	0	USR_OFF_W	0	XL_AXIS _SEL_1	XL_AXIS _SEL_0
CTRL7_C	16h	0	0	0	0	0	0	USR_OFF_ ON_OUT	0
CTRL8_XL	17h	HPCF_XL2	HPCF_XL1	HPCF_XL0	HP_REF _MODE_XL	FASTSETTL _MODE_XL	FDS	0	0
CTRL10_C	19h	0	0	TIMESTAMP_EN	0	0	0	0	0
ALL_INT_SRC	1Ah	TIMESTAMP _ENDCOUNT	0	SLEEP_ CHANGE_IA	0	0	0	WU_IA	0
WAKE_UP_SRC	1Bh	0	SLEEP_ CHANGE_IA	0	SLEEP_ STATE_IA	WU_IA	X_WU	Y_WU	Z_WU
STATUS_REG	1Eh	0	0	0	0	0	TDA	0	XLDA
OUT_TEMP_L	20h	Temp7	Temp6	Temp5	Temp4	Temp3	Temp2	Temp1	Temp0
OUT_TEMP_H	21h	Temp15	Temp14	Temp13	Temp12	Temp11	Temp10	Temp9	Temp8
OUTX_L_A	28h	D7	D6	D5	D4	D3	D2	D1	D0
OUTX_H_A	29h	D15	D14	D13	D12	D11	D10	D9	D8
OUTY_L_A	2Ah	D7	D6	D5	D4	D3	D2	D1	D0
OUTY_H_A	2Bh	D15	D14	D13	D12	D11	D10	D9	D8

Register name	Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
OUTZ_L_A	2Ch	D7	D6	D5	D4	D3	D2	D1	D0
OUTZ_H_A	2Dh	D15	D14	D13	D12	D11	D10	D9	D8
FIFO_STATUS1	3Ah	DIFF_FIFO_7	DIFF_FIFO_6	DIFF_FIFO_5	DIFF_FIFO_4	DIFF_FIFO_3	DIFF_FIFO_2	DIFF_FIFO_1	DIFF_FIFO_0
FIFO_STATUS2	3Bh	FIFO_WTM_IA	FIFO_OVR_IA	FIFO_FULL_IA	COUNTER _BDR_IA	FIFO_OVR _LATCHED	0	DIFF_FIFO_9	DIFF_FIFO_8
TIMESTAMP0	40h	T7	T6	T5	T4	T3	T2	T1	T0
TIMESTAMP1	41h	T15	T14	T13	T12	T11	T10	T9	T8
TIMESTAMP2	42h	T23	T22	T21	T20	T19	T18	T17	T16
TIMESTAMP3	43h	T31	T30	T29	T28	T27	T26	T25	T24
SLOPE_EN	56h	0	0	SLEEP_STATUS _ON_INT	SLOPE_FDS	0	0	0	LIR
INTERRUPTS_EN	58h	INTERRUPTS_ ENABLE	0	0	0	0	0	0	0
WAKE_UP_THS	5Bh	0	USR_OFF_ON_WU	WK_THS5	WK_THS4	WK_THS3	WK_THS2	WK_THS1	WK_THS0
WAKE_UP_DUR	5Ch	0	WAKE_DUR1	WAKE_DUR0	WAKE_THS_W	SLEEP_DUR3	SLEEP_DUR2	SLEEP_DUR1	SLEEP_DUR0
MD1_CFG	5Eh	INT1_SLEEP _CHANGE	0	INT1_WU	0	0	0	0	0
MD2_CFG	5Fh	INT2_SLEEP _CHANGE	0	INT2_WU	0	0	0	0	INT2_ TIMESTAMP
INTERNAL_FREQ_FINE	63h	FREQ_FINE7	FREQ_FINE6	FREQ_FINE5	FREQ_FINE4	FREQ_FINE3	FREQ_FINE2	FREQ_FINE1	FREQ_FINE0
X_OFS_USR	73h	X_OFS_USR_7	X_OFS_USR_6	X_OFS_USR_5	X_OFS_USR_4	X_OFS_USR_3	X_OFS_USR_2	X_OFS_USR_1	X_OFS_USR_0
Y_OFS_USR	74h	Y_OFS_USR_7	Y_OFS_USR_6	Y_OFS_USR_5	Y_OFS_USR_4	Y_OFS_USR_3	Y_OFS_USR_2	Y_OFS_USR_1	Y_OFS_USR_0
Z_OFS_USR	75h	Z_OFS_USR_7	Z_OFS_USR_6	Z_OFS_USR_5	Z_OFS_USR_4	Z_OFS_USR_3	Z_OFS_USR_2	Z_OFS_USR_1	Z_OFS_USR_0
FIFO_DATA_OUT_TAG	78h	TAG_SENSOR_4	TAG_SENSOR_3	TAG_SENSOR_2	TAG_SENSOR_1	TAG_SENSOR_0	TAG_CNT_1	TAG_CNT_0	TAG_PARITY
FIFO_DATA_OUT_X_L	79h	D7	D6	D5	D4	D3	D2	D1	D0
FIFO_DATA_OUT_X_H	7Ah	D15	D14	D13	D12	D11	D10	D9	D8
FIFO_DATA_OUT_Y_L	7Bh	D7	D6	D5	D4	D3	D2	D1	D0
FIFO_DATA_OUT_Y_H	7Ch	D15	D14	D13	D12	D11	D10	D9	D8
FIFO_DATA_OUT_Z_L	7Dh	D7	D6	D5	D4	D3	D2	D1	D0
FIFO_DATA_OUT_Z_H	7Eh	D15	D14	D13	D12	D11	D10	D9	D8

3 Operating modes

The IIS3DWB offers two possible operating configurations:

- Power-Down mode;
- Normal mode.

The device offers a wide VDD voltage range from 2.1 V to 3.6 V and a VDDIO range from 1.62 V to VDD+0.1 V. The power-on sequence is not restricted: VDD/VDDIO pins can be either set to power supply level or to ground level (they must not be left floating) and no specific sequence is required for powering them on.

In order to avoid potential conflicts, during the power-on sequence it is recommended to set the lines (on the host side) connected to the device IO pins floating or connected to ground until VDDIO is set. After VDDIO is set, the IO pins have to be configured according to their default status described in Table 1. Default pin status. In order to avoid an unexpected increase in current consumption, the input pins which are not pulled-up/pulled-down must the polarized by the host.

When the VDD power supply is applied, the device performs a 10 ms (maximum) boot procedure to load the trimming parameters. After the boot is completed, the accelerometer is automatically configured in Power-Down mode. To guarantee proper power-off of the device it is recommended to maintain the duration of the VDD line to GND for at least 100 µs.

When the sensor is in Power-Down mode, almost all internal blocks of the device are switched off. The SPI / I²C digital interface remains active to allow communication with the device. The content of the configuration registers is preserved and the output data registers are not updated, keeping the last data sampled in memory before going into Power-Down mode.

The IIS3DWB can be configured in Normal mode by setting to 101b the XL_EN[2:0] bits of the CTRL1_XL register: the sensor provides acceleration data at an output data rate of 26.667 kHz. The device has a selectable full-scale acceleration range of ±2 *g*, ±4 *g*, ±8 *g* or ±16 *g*.

An active axis of the accelerometer can be selected by using the XL_AXIS_SEL[1:0] bits of the CTRL6_C register, as indicated in the table below. Two operating modes are available:

- 3-axis mode: all three axes (X, Y, Z) are simultaneously active;
- Single-axis mode: only one axis is active (the output value of the other two axes is set to 0).

Table 3. Accelerometer active axis

XL_AXIS_ SEL[1:0]	Active axis
00 (default)	3 axes (XYZ)
01	X-axis
10	Y-axis
11	Z-axis

In both cases the acceleration data of the active axes can be concurrently read from the sensor by using the accelerometer output registers (see Section 4 Reading output data) or the FIFO registers (see Section 6 First-in, first-out (FIFO) buffer)

The active axis has to be selected through the XL_AXIS_SEL[1:0] bits when the device is in Power-Down mode, before setting the device in Normal mode.

In single-axis mode, while the power consumption of IIS3DWB remains the same as 3-axis mode (1.1 mA typ.), the resolution (noise density) of the active axis significantly improves.

Note: Only the SPI interface supports all the device features and capabilities. The I²C interface can be used only in single-axis mode and it is not recommended.

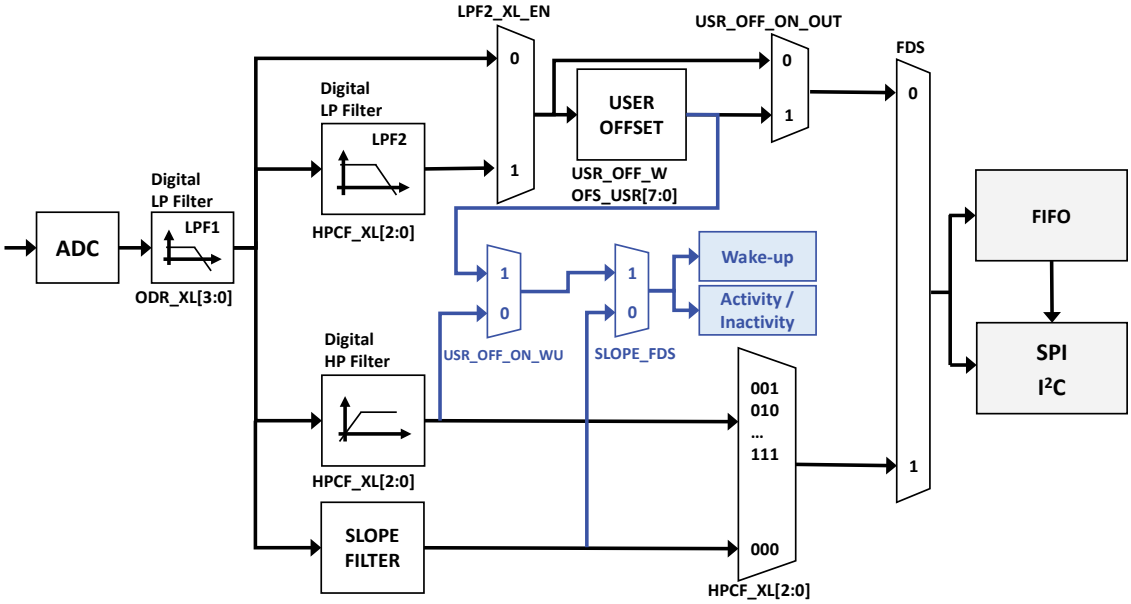
3.1 Accelerometer filtering chain

The IIS3DWB provides a wide bandwidth with very flat frequency response in the pass band and a very high attenuation in the stop band; details about frequency response of the IIS3DWB are given in the device datasheet.

The accelerometer sampling chain is represented by a cascade of three main blocks: an ADC converter, a digital low-pass filter (LPF1) and the composite group of digital filters, as shown in Figure 2. Accelerometer filtering chain.

The analog signal coming from the mechanical parts is converted by the ADC; then, the digital signal is filtered by the digital low-pass filter LPF1 having a cut-off frequency equal to 6.3 kHz.

Figure 2. Accelerometer filtering chain



The composite group of filters composed of a low-pass digital filter (LPF2), a high-pass digital filter and a slope filter processes the digital signal.

The LPF2_XL_EN bit of CTRL1_XL register and the CTRL8_XL register can be used to configure the composite filter group and the overall bandwidth of the accelerometer filtering chain, as shown in Table 4. Accelerometer bandwidth selection. Referring to this table, on the low-pass path side, the Bandwidth columns refer to the LPF1 bandwidth if LPF2_XL_EN = 0; they refer to the LPF2 bandwidth if LPF2_XL_EN = 1. On the high-pass path side, the Bandwidth columns refer to the Slope filter bandwidth if HPCF_XL[2:0] = 000b; they refer to the HP filter bandwidth for all the other configurations.

Table 4. Accelerometer bandwidth selection

FDS	LPF2_XL_EN	HPCF_XL[2:0]	Bandwidth	Max overall settling time[1] (samples to be discarded)
0 (Low-pass path)	0	-	6.3 kHz	12
	1	000	ODR/4	12
		001	ODR/10	12
		010	ODR/20	19
		011	ODR/45	38
		100	ODR/100	75
		101	ODR/200	150
		110	ODR/400	295
		111	ODR/800	595
1 (High-pass path)	-	000	ODR / 4 (slope filter)	13
		001	ODR/10	13
		010	ODR/20	19
		011	ODR/45	38
		100	ODR/100	75
		101	ODR/200	150
		110	ODR/400	295
		111	ODR/800	595

1. *Settling time @ 99% of the final value*

Setting the FDS bit of the CTRL8_XL register to 0, the low-pass path of the composite filter block is selected. If the LPF2_XL_EN bit is set to 0, no additional filter is applied; if the LPF2_XL_EN bit is set to 1, the LPF2 filter is applied in addition to LPF1 and the overall bandwidth of the accelerometer chain can be set by configuring the HPCF_XL[2:0] field of the CTRL8_XL register.

Setting the FDS bit to 1, the high-pass path of the composite filter block is selected: the HPCF_XL[2:0] field is used in order to enable, in addition to the LPF1 filter, either the Slope filter usage (when HPCF_XL[2:0] = 000b) or the digital High-Pass filter (other HPCF_XL[2:0] configurations). The HPCF_XL[2:0] field is also used to select the cutoff frequencies of the HP filter.

The high-pass filter reference mode feature is available for the accelerometer sensor: when this feature is enabled, the current X, Y, Z accelerometer sample is internally stored and subtracted from all subsequent output values. In order to enable the reference mode, both the HP_REF_MODE_XL bit and the FDS bit of the CTRL8_XL register have to be set to 1, and the value of the HPCF_XL[2:0] field must be equal to 111b. When the reference mode feature is enabled, both the LPF2 filter and the HP filter are not available. The first accelerometer output data after enabling the reference mode has to be discarded.

The FASTSETTL_MODE_XL bit of CTRL8_XL register enables the accelerometer LPF2 or HPF fast-settling mode: the selected filter sets the second sample after writing this bit. This feature applies only upon device exit from Power-Down mode.

3.1.1 Accelerometer slope filter

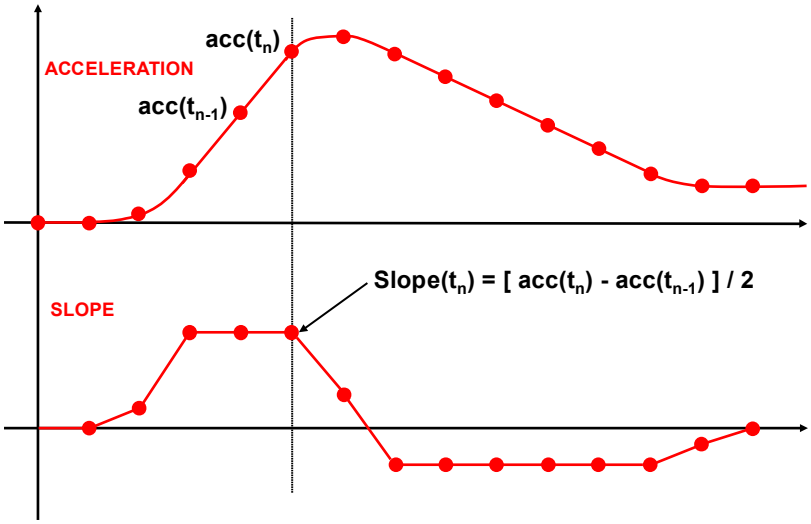
As shown in Figure 2. Accelerometer filtering chain, the device embeds a digital slope filter, which can also be used for the embedded features (wake-up detection and activity/inactivity).

The slope filter output data is computed using the following formula:

$$slope(t_n) = [\ acc(t_n) - acc(t_{n-1})\ ] / 2$$

An example of a slope data signal is illustrated in the following figure.

Figure 3. Accelerometer slope filter



3.2 Accelerometer turn-on/off time

The accelerometer reading chain contains low-pass filtering to improve signal-to-noise performance. For this reason, it is necessary to take into account the settling time of the filters when the accelerometer operating mode is switched.

Table 4. Accelerometer bandwidth selection provides the maximum (worst case) turn-on time (when switching from Power-Down mode to Normal mode) in terms of samples to be discarded for the various configurations of the accelerometer filtering chain.

Maximum turn-off time when switching from Normal mode to Power-Down mode is 1 µs.

4 Reading output data

4.1 Startup sequence

Once the device is powered up, it automatically downloads the calibration coefficients from the embedded flash to the internal registers. When the boot procedure is completed, i.e. after approximately 10 milliseconds, the accelerometer automatically enters Power-Down mode.

To turn on the accelerometer and gather acceleration data through the SPI / I²C interface, it is necessary to configure the XL_EN[2:0] bits of the CTRL1_XL register to 101b.

The following general-purpose sequence can be used to configure the accelerorometer:

1. Write INT1_CTRL = 01h // Acc data-ready interrupt on INT1
2. Write CTRL6_C = 00h // 3-axis mode enabled
3. Write CTRL1_XL = A0h // Enable accelerometer (ODR = 26.667 kHz ; FS = ±2 *g*)

4.2 Using the status register

The device is provided with a STATUS_REG register which should be polled to check when a new set of data is available. The XLDA bit is set to 1 when a new set of data is available at the accelerometer output.

The read of the accelerometer output registers (in 3-axis mode) should be performed as follows:

1. Read STATUS_REG
2. If XLDA = 0, then go to 1
3. Read OUTX_L_A
4. Read OUTX_H_A
5. Read OUTY_L_A
6. Read OUTY_H_A
7. Read OUTZ_L_A
8. Read OUTZ_H_A
9. Data processing
10. Go to 1

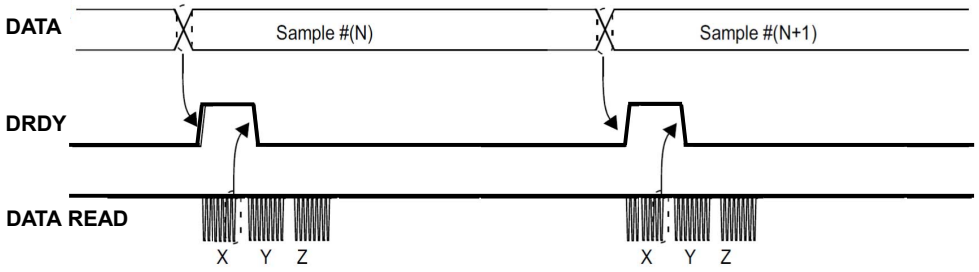
4.3 Using the data-ready signal

The device can be configured to have one hardware signal to determine when a new set of measurement data is available to be read.

The accelerometer sensor data-ready signal is represented by the XLDA bit of the STATUS_REG register. This signal can be driven to the INT1 pin by setting the INT1_DRDY_XL bit of the INT1_CTRL register to 1 and to the INT2 pin by setting the INT2_DRDY_XL bit of the INT2_CTRL register to 1.

The data-ready signal rises to 1 when a new set of data has been generated and it is available to be read. The data-ready signal can be either latched or pulsed: if the dataready_pulsed bit of the COUNTER_BDR_REG1 register is set to 0 (default value), then the data-ready signal is latched and the interrupt is reset when the higher part of one of the enabled channels is read (registers 29h, 2Bh, 2Dh). If the dataready_pulsed bit of the COUNTER_BDR_REG1 register is set to 1, then the data-ready is pulsed and the duration of the pulse observed on the interrupt pins is 18.75 µs (typ.). Pulsed mode is not applied to the XLDA bit which is always latched.

Figure 4. Data-ready signal



4.3.1 DRDY mask functionality

Setting the DRDY_MASK bit of the CTRL4_C register to 1, the accelerometer data-ready signal is masked until the settling of the sensor filters is completed.

When FIFO is active and the DRDY_MASK bit is set to 1, accelerometer invalid samples stored in FIFO can be equal to 7FFFh, 7FFEh or 7FFDh. In this way, a tag is applied to the invalid samples stored in the FIFO buffer so that they can be easily identified and discarded during data post-processing.

Note: The DRDY_MASK bit acts only on the accelerometer LPF1 digital filter settling time.

4.4 Using the block data update (BDU) feature

If reading the accelerometer data is particularly slow and cannot be synchronized (or it is not required) with the XLDA bit in the STATUS_REG register or with the DRDY signal driven to the INT1/INT2 pins, it is strongly recommended to set the BDU (Block Data Update) bit to 1 in the CTRL3_C register.

This feature avoids reading values (most significant and least significant parts of output data) related to different samples. In particular, when the BDU is activated, the data registers related to each channel always contain the most recent output data produced by the device, but, in case the read of a given pair (i.e. OUTX_H_A and OUTX_L_A, OUTY_H_A and OUTY_L_A, OUTZ_H_A and OUTZ_L_A) is initiated, the refresh for that pair is blocked until both MSB and LSB parts of the data are read.

Note: BDU only guarantees that the LSB part and MSB part have been sampled at the same moment. For example, if the reading speed is too slow, X and Y can be read at T1 and Z sampled at T2.

The BDU feature also acts on the FIFO_STATUS1 and FIFO_STATUS2 registers. When the BDU bit is set to 1, it is mandatory to read FIFO_STATUS1 first and then FIFO_STATUS2.

4.5 Understanding output data

The measured acceleration data are sent to the OUTX_H_A, OUTX_L_A, OUTY_H_A, OUTY_L_A, OUTZ_H_A, and OUTZ_L_A registers. These registers contain, respectively, the most significant part and the least significant part of the acceleration signals acting on the X, Y, and Z axes.

The complete output data for the X, Y, Z channels is given by the concatenation OUTX_H_A & OUTX_L_A, OUTY_H_A & OUTY_L_A, OUTZ_H_A & OUTZ_L_A and it is expressed as a two's complement number.

Acceleration data are represented as 16-bit numbers.

4.5.1 Examples of output data

Table 5. Content of output data registers vs. acceleration (FS_XL = ±2 *g*) provides a few basic examples of the accelerometer data that is read in the data registers when the device is subjected to a given acceleration.

The values listed in the following table are given under the hypothesis of perfect device calibration (i.e. no offset, no gain error, …).

Table 5. Content of output data registers vs. acceleration (FS_XL = ±2 *g*)

Acceleration values	Register address	
	OUTX_H_A (29h)	OUTX_L_A (28h)
0 *g*	00h	00h
350 m*g*	16h	69h
1 *g*	40h	09h
-350 m*g*	E9h	97h
-1 *g*	BFh	F7h

4.6 Accelerometer offset registers

The device provides accelerometer offset registers (X_OFS_USR, Y_OFS_USR, Z_OFS_USR) which can be used for zero-*g* offset correction or, in general, to apply an offset to the accelerometer output data.

The accelerometer offset block can be enabled by setting the USR_OFF_ON_OUT bit of the CTRL7_C register. The offset value set in the offset registers is internally subtracted from the measured acceleration value for the respective axis; internally processed data are then sent to the accelerometer output register and to the FIFO (if enabled). These register values are expressed as an 8-bit word in two's complement and must be in the range [-127, 127].

The weight [*g*/LSB] to be applied to the offset register values is independent of the accelerometer selected full scale and can be configured using the USR_OFF_W bit of the CTRL6_C register:

- 2^{-10} *g*/LSB if the USR_OFF_W bit is set to 0;
- 2^{-6} *g*/LSB if the USR_OFF_W bit is set to 1.

4.7 Rounding function

The rounding function can be used to auto address the device registers for a circular burst-mode read. Basically, with a multiple read operation the address of the register that is being read goes automatically from the first register to the last register of the pattern and then goes back to the first one.

In order to enable the rounding function for the accelerometer output registers (from OUTX_L_A (28h) to OUTZ_H_A (2Dh)), the ROUNDING[1:0] bits of the CTRL5_C register have to be set to 01b.

5 Interrupt generation

Interrupt generation is based on accelerometer data, so, for interrupt-generation purposes, the accelerometer sensor has to be set to Normal mode.

The interrupt generator can be configured to detect wake-up events and for activity/inactivity events.

The interrupt signals, together with the FIFO interrupt signals, can be independently driven to the INT1 and INT2 interrupt pins or checked by reading the dedicated source register bits.

The H_LACTIVE bit of the CTRL3_C register must be used to select the polarity of the interrupt pins. If this bit is set to 0 (default value), the interrupt pins are active high and they change from low to high level when the related interrupt condition is verified. Otherwise, if the H_LACTIVE bit is set to 1 (active low), the interrupt pins are normally at high level and they change from high to low when interrupt condition is reached.

The PP_OD bit of CTR3_C allows changing the behavior of the interrupt pins from push-pull to open drain. If the PP_OD bit is set to 0, the interrupt pins are in push-pull configuration (low-impedance output for both high and low level). When the PP_OD bit is set to 1, only the interrupt active state is a low-impedance output.

5.1 Interrupt pin configuration

The device is provided with two pins that can be activated to generate either data-ready or interrupt signals. The functionality of these pins is selected through the MD1_CFG and INT1_CTRL registers for the INT1 pin, and through the MD2_CFG and INT2_CTRL registers for the INT2 pin.

A brief description of these interrupt control registers is given in the following summary; the default value of their bits is equal to 0, which corresponds to ‘disable’. In order to enable the routing of a specific interrupt signal on the pin, the related bit has to be set to 1.

Table 6. INT1_CTRL register

b7	b6	b5	b4	b3	b2	b1	b0
0	INT1_ CNT_BDR	INT1_ FIFO_FULL	INT1_ FIFO_OVR	INT1_ FIFO_TH	INT1_ BOOT	0	INT1_ DRDY_XL

- INT1_CNT_BDR: FIFO COUNTER_BDR_IA interrupt on INT1
- INT1_FIFO_FULL: FIFO full flag interrupt on INT1
- INT1_FIFO_OVR: FIFO overrun flag interrupt on INT1
- INT1_FIFO_TH: FIFO threshold interrupt on INT1
- INT1_BOOT: Boot interrupt on INT1
- INT1_DRDY_XL: Accelerometer data-ready on INT1

Table 7. MD1_CFG register

b7	b6	b5	b4	b3	b2	b1	b0
INT1_SLEEP _CHANGE	0	INT1_WU	0	0	0	0	0

- INT1_SLEEP_CHANGE: Activity/inactivity recognition event interrupt on INT1
- INT1_WU: Wake-up interrupt on INT1

Table 8. **INT2_CTRL register**

b7	b6	b5	b4	b3	b2	b1	b0
0	INT2_ CNT_BDR	INT2_ FIFO_FULL	INT2_ FIFO_OVR	INT2_ FIFO_TH	INT2_DRDY _TEMP	0	INT2_ DRDY_XL

- INT2_CNT_BDR: FIFO COUNTER_BDR_IA interrupt on INT2
- INT2_FIFO_FULL: FIFO full flag interrupt on INT2
- INT2_FIFO_OVR: FIFO overrun flag interrupt on INT2
- INT2_FIFO_TH: FIFO threshold interrupt on INT2
- INT2_DRDY_TEMP: Temperature data-ready on INT2
- INT2_DRDY_XL: Accelerometer data-ready on INT2

Table 9. **MD2_CFG register**

b7	b6	b5	b4	b3	b2	b1	b0
INT2_SLEEP _CHANGE	0	INT2_WU	0	0	0	0	INT2_ TIMESTAMP

- INT2_SLEEP_CHANGE: Activity/inactivity recognition event interrupt on INT2
- INT2_WU: Wake-up interrupt on INT2
- INT2_TIMESTAMP: timestamp overflow alert interrupt on INT2

If multiple interrupt signals are routed on the same pin (INTx), the logic level of this pin is the “OR” combination of the selected interrupt signals. In order to know which event has generated the interrupt condition, the related source registers have to be read:

- WAKE_UP_SRC, ALL_INT_SRC (for wake-up and activity/inactivity functions)
- STATUS_REG (for data-ready signals)
- FIFO_STATUS2 (for FIFO)

The INT2_on_INT1 pin of CTRL4_C register allows driving all the enabled interrupt signals in logic “OR” on the INT1 pin (by setting this bit to 1). When this bit is set to 0, the interrupt signals are divided between the INT1 and INT2 pins.

The wake-up and activity/inactivity interrupts have to be enabled by setting the INTERRUPTS_ENABLE bit in the INTERRUPTS_EN register.

The LIR bit of the SLOPE_EN register enables the latched interrupt for the basic interrupt functions: when this bit is set to 1 and the interrupt flag is sent to the INT1 pin and/or INT2 pin, the interrupt remains active until the ALL_INT_SRC register or the corresponding source register is read, and it is reset at the next ODR cycle. The latched interrupt is enabled on a function only if a function is routed to the INT1 or INT2 pin: if latched mode is enabled but the interrupt signal is not driven to the interrupt pins, the latch feature does not take effect.

Note: If latched mode is enabled (LIR = 1), it is not recommended to continuously poll the ALL_INT_SRC or the dedicated source registers, because by reading them the embedded functions are internally reset; a synchronous (with interrupt event) read of the source registers is recommended in this case.

5.2 Wake-up interrupt

The wake-up feature can be implemented using either the slope filter (see Section 3.1.1 Accelerometer slope filter for more details) or the high-pass digital filter, as illustrated in Figure 2. Accelerometer filtering chain. The filter to be applied can be selected using the SLOPE_FDS bit of the SLOPE_EN register: if this bit is set to 0 (default value), the slope filter is used; if it's set to 1, the HPF digital filter is used. Moreover, it is possible to configure the wake-up feature as an absolute wake-up with respect to a programmable position. This can be done by setting both the SLOPE_FDS bit of the SLOPE_EN register and the USR_OFF_ON_WU bit of the WAKE_UP_THS register to 1. Using this configuration, the input data for the wake-up function comes from the low-pass filter path and the programmable position is subtracted as an offset. The programmable position can be configured through the X_OFS_USR, Y_OFS_USR and Z_OFS_USR registers (refer to Section 4.6 Accelerometer offset registers for more details).

The wake-up interrupt signal is generated if a certain number of consecutive filtered data exceed the configured threshold (Figure 5. Wake-up interrupt (using the slope filter)).

The unsigned threshold value is defined using the WK_THS[5:0] bits of the WAKE_UP_THS register; the value of 1 LSB of these 6 bits depends on the selected accelerometer full scale and on the value of the WAKE_THS_W bit of the WAKE_UP_DUR register:

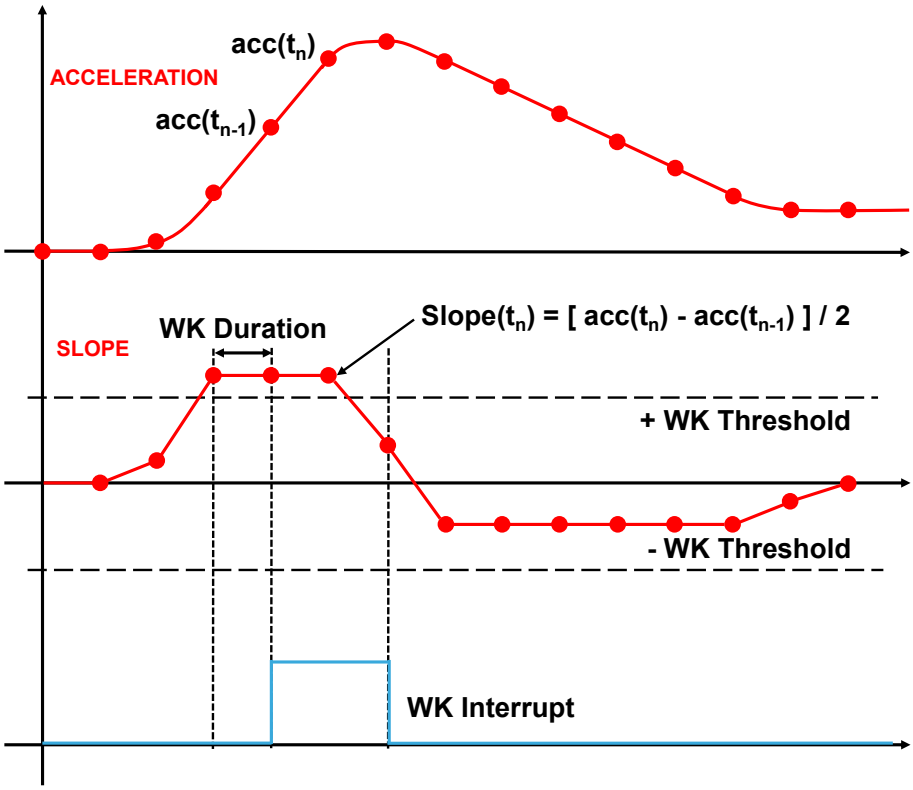
- If WAKE_THS_W = 0, 1 LSB = FS_XL / 2^6;
- If WAKE_THS_W = 1, 1 LSB = FS_XL / 2^8.

The threshold is applied to both positive and negative data: for wake-up interrupt generation, the absolute value of the filtered data must be bigger than the threshold.

The duration parameter defines the minimum duration of the wake-up event to be recognized; its value is set using the WAKE_DUR[1:0] bits of the WAKE_UP_DUR register: 1 LSB corresponds to 1/ODR_XL time, where ODR_XL is the accelerometer output data rate. It is important to appropriately define the duration parameter to avoid unwanted wake-up interrupts due to spurious spikes of the input signal.

This interrupt signal can be enabled by setting the INTERRUPTS_ENABLE bit in the INTERRUPTS_EN register to 1 and can be driven to the two interrupt pins by setting to 1 the INT1_WU bit of the MD1_CFG register or the INT2_WU bit of the MD2_CFG register; it can also be checked by reading the WU_IA bit of the WAKE_UP_SRC or ALL_INT_SRC register. The X_WU, Y_WU, Z_WU bits of the WAKE_UP_SRC register indicate which axes have triggered the wake-up event.

Figure 5. **Wake-up interrupt (using the slope filter)**



If latch mode is disabled (LIR bit of SLOPE_EN is set to 0), the interrupt signal is automatically reset when the filtered data falls below the threshold. If latch mode is enabled and the wake-up interrupt signal is driven to the interrupt pins, once a wake-up event has occurred and the interrupt pin is asserted, it must be reset by reading the WAKE_UP_SRC register or the ALL_INT_SRC register. The X_WU, Y_WU, Z_WU bits are maintained at the state in which the interrupt was generated until the read is performed, and released at the next ODR cycle. In case the WU_X, WU_Y, WU_Z bits have to be evaluated (in addition to the WU_IA bit), it is recommended to directly read the WAKE_UP_SRC register (do not use ALL_INT_SRC register for this specific case). If latch mode is enabled but the interrupt signal is not driven to the interrupt pins, the latch feature does not take effect.

A basic SW routine for wake-up event recognition using the high-pass digital filter is given below.

1. Write A0h to CTRL1_XL // Enable accelerometer (ODR = 26.667 kHz ; FS = ±2 *g*)
2. Write 11h to SLOPE_EN // Enable latch mode and digital high-pass filter
3. Write 80h to INTERRUPTS_EN // Enable interrupt function
4. Write 00h to WAKE_UP_DUR // No duration and selection of wake-up threshold weight (1 LSB = FS_XL / 2^6)
5. Write 02h to WAKE_UP_THS // Set wake-up threshold
6. Write 20h to MD1_CFG // Wake-up interrupt driven to INT1 pin

Since the duration time is set to zero, the wake-up interrupt signal is generated for each X,Y,Z filtered data exceeding the configured threshold. The WK_THS field of the WAKE_UP_THS register is set to 000010b, therefore the wake-up threshold is 62.5 m*g* (= 2 * FS_XL / 2^6).

If the wake-up functionality is implemented using the slope/high-pass digital filter, it is necessary to consider the settling time of the filter just after this functionality is enabled. For example, when using the slope filter (but a similar consideration can be done for the high-pass digital filter usage) the wake-up functionality is based on the comparison of the threshold value with half of the difference of the acceleration of the current (x,y,z) sample and the previous one (refer to Section 3.1.1 Accelerometer slope filter).

At the very first sample, the slope filter output is calculated as half of the difference of the current sample [e.g. (x,y,z) = (0,0,1*g*)] with the previous one which is (x,y,z)=(0,0,0) since it doesn't exist. For this reason, on the z-axis the first output value of the slope filter is (1*g* - 0)/2=500 m*g* and it could be higher than the threshold value in which case a spurious interrupt event is generated. The interrupt signal is kept high for 1 ODR then it goes low.

In order to avoid this spurious interrupt generation, multiple solutions are possible. Hereafter are two alternative solutions (for the slope filter case):

a. Ignore the first generated wake-up signal;

b. Add a wait time higher than 1 ODR before driving the interrupt signal to the INT1/2 pin.

5.3 Activity/Inactivity recognition

The working principle of the Activity/Inactivity embedded function is similar to wake-up. If no movement condition is detected for a programmable time, an inactivity/stationary condition event is generated; otherwise, when the accelerometer data exceed the configurable threshold, an Activity/Motion condition event is generated.

The Activity/Inactivity recognition function is enabled when the INTERRUPTS_ENABLE bit of the INTERRUPTS_EN register is set to 1.

The Activity/Inactivity recognition function can be implemented using either the slope filter (see Section 3.1.1 Accelerometer slope filter for more details) or the high-pass digital filter, as illustrated in Figure 2. Accelerometer filtering chain. The filter to be applied can be selected using the SLOPE_FDS bit of the SLOPE_EN register: if this bit is set to 0 (default value), the slope filter is used; if it is set to 1, the high-pass digital filter is used.

This function can be fully programmed by the user in terms of expected amplitude and timing of the filtered data by means of a dedicated set of registers (Figure 6. Activity/Inactivity recognition (using the slope filter)).

The unsigned threshold value is defined using the WK_THS[5:0] bits of the WAKE_UP_THS register; the value of 1 LSB of these 6 bits depends on the selected accelerometer full scale and on the value of the WAKE_THS_W bit of the WAKE_UP_DUR register:

- if WAKE_THS_W = 0, 1 LSB = FS_XL / 2^6;
- if WAKE_THS_W = 1, 1 LSB = FS_XL / 2^8.

The threshold is applied to both positive and negative filtered data.

The duration of the Inactivity status to be recognized is defined by the SLEEP_DUR[3:0] bits of the WAKE_UP_DUR register: 1 LSB corresponds to 512/ODR_XL time, where ODR_XL is the accelerometer output data rate. If the SLEEP_DUR[3:0] bits are set to 0000b, the duration of the Inactivity status to be recognized is equal to 16 / ODR_XL time.

When the Inactivity status is detected, the interrupt is set high for 1/ODR_XL[s] period then it is automatically deasserted.

When filtered data on one axis becomes bigger than the threshold for a configurable time, the Activity status is detected. The duration of the Activity status to be recognize is defined by the WAKE_DUR[1:0] bits of the WAKE_UP_DUR register. 1 LSB corresponds to 1 / ODR_XL time, where ODR_XL is the accelerometer output data rate.

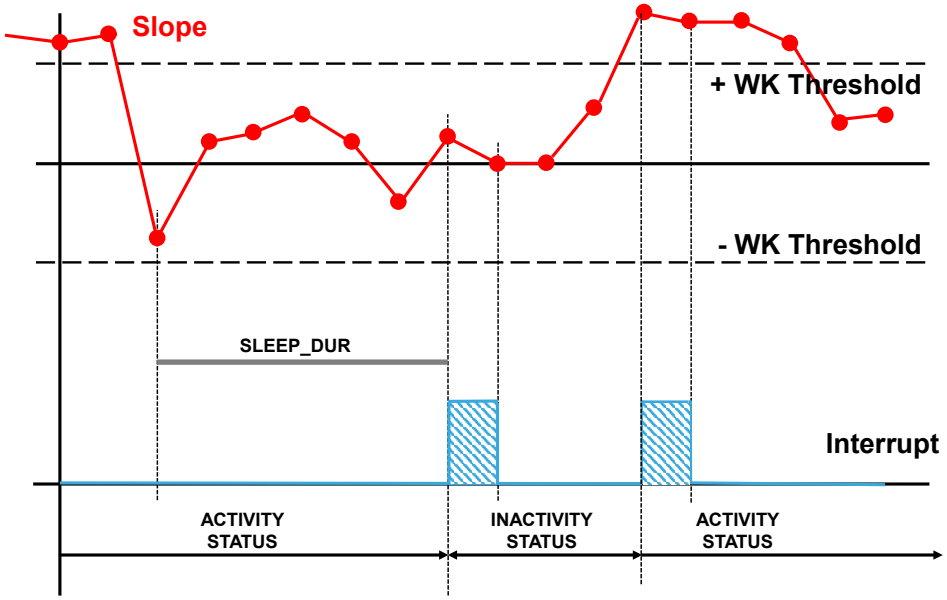
When the Activity status is detected, the interrupt is set high for 1/ODR_XL[s] period then it is automatically deasserted.

Once the Activity/Inactivity detection function is enabled, the status can be driven to the two interrupt pins by setting to 1 the INT1_SLEEP_CHANGE bit of the MD1_CFG register or the INT2_SLEEP_CHANGE bit of the MD2_CFG register; it can also be checked by reading the SLEEP_CHANGE_IA bit of the WAKE_UP_SRC or ALL_INT_SRC register.

The SLEEP_CHANGE_IA bit is by default in pulsed mode. Latched mode can be selected by setting the LIR bit of the SLOPE_EN register to 1 and the the INT1_SLEEP_CHANGE of the MD1_CFG register or INT2_SLEEP_CHANGE of the MD2_CFG register to 1. The SLEEP_STATE_IA bit of the WAKE_UP_SRC register is not affected by the LIR configuration: it corresponds to the current state of the device when the WAKE_UP_SRC register is read.

By setting the SLEEP_STATUS_ON_INT bit of the SLOPE_EN register to 1, the signal routed to the INT1 or INT2 pins is configured to be the Activity/Inactivity state (SLEEP_STATE bit of WAKE_UP_SRC register) instead of the sleep-change signal: it goes high during Inactivity state and it goes low during Activity state. Latched mode is not supported in this configuration.

Figure 6. **Activity/Inactivity recognition (using the slope filter)**



A basic SW routine for Activity/Inactivity detection is as follows:

1.	Write A0h to CTRL1_XL	// Enable accelerometer (ODR = 26.667 kHz ; FS = ±2 *g*)
2.	Write 02h to WAKE_UP_DUR	// Set duration for Inactivity detection
		// Select Activity/Inactivity threshold resolution and duration
3.	Write 02h to WAKE_UP_THS	// Set Activity/Inactivity threshold
4.	Write 00h to SLOPE_EN	// Select sleep-change notification
		// Select slope filter
5.	Write 80h to INTERRUPTS_EN	// Enable interrupt
6.	Write 80h to MD1_CFG	// Activity/Inactivity interrupt driven to INT1 pin

In this example the WK_THS field of the WAKE_UP_THS register is set to 000010b, therefore the Activity/Inactivity threshold is 62.5 mg (= 2 * FS_XL / 2^6 since the WAKE_THS_W bit of the WAKE_UP_DUR register is set to 0).

Before Inactivity detection, the X,Y,Z slope data must be smaller than the configured threshold for a period of time defined by the SLEEP_DUR field of the WAKE_UP_DUR register: this field is set to 0010b, corresponding to 0.038 s (= 2 * 512 / ODR_XL). After this period of time has elapsed, the Inactivity status is detected.

The Activity status is detected as soon as the slope data of (at least) one axis are bigger than the threshold for one sample, since the WAKE_DUR[1:0] bits of the WAKE_UP_DUR register are configured to 00b.

5.4 Boot status

After the device is powered up, it performs a 10 ms (maximum) boot procedure to load the trimming parameters. After the boot is completed, the accelerometer is automatically configured in Power-Down mode. During the boot time the registers are not accessible.

After power up, the trimming parameters can be re-loaded by setting the BOOT bit of the CTRL3_C register to 1.

No toggle of the device power lines is required and the content of the device control registers is not modified, so the device operating mode doesn't change after boot. If the reset to the default value of the control registers is required, it can be performed by setting the SW_RESET bit of the CTRL3_C register to 1. When this bit is set to 1, the following registers are reset to their default value:

- PIN_CTRL (02h);
- FIFO_CTRL1 (07h) through FIFO_CTRL4 (0Ah);
- COUNTER_BDR_REG1 (0Bh) and COUNTER_BDR_REG2 (0Ch);
- INT1_CTRL (0Dh) and INT2_CTRL (0Eh);
- CTRL1_XL (10h) through CTRL10_C (19h);
- FIFO_STATUS1 (3Ah) and FIFO_STATUS2 (3Bh);
- SLOPE_EN (56h);
- WAKE_UP_THS (5Bh) and WAKE_UP_DUR(5Ch);
- MD1_CFG (5Eh) and MD2_CFG (5Fh);
- X_OFS_USR (73h), Y_OFS_USR (74h) and Z_OFS_USR (75h).

The SW_RESET procedure can take 50 µs; the status of reset is signaled by the status of the SW_RESET bit of the CTRL3_C register: once the reset is completed, this bit is automatically set low.

The boot status signal is driven to the INT1 interrupt pin by setting the INT1_BOOT bit of the INT1_CTRL register to 1: this signal is set high while the boot is running and it is set low again at the end of the boot procedure.

The reboot flow is as follows:

1. Set the accelerometer in Power-Down mode;
2. Set INT1_BOOT bit of INT1_CTRL register to 1 [optional];
3. Set BOOT bit of CTRL3_C register to 1;
4. Monitor reboot status, three possibilities:
   a. Wait 10 ms;
   b. Monitor INT1 pin until it returns to 0 (step 2. is mandatory in this case);
   c. Poll BOOT bit of CTRL3_C until it returns to 0.

Reset flow is as follows:

1. Set the accelerometer in Power-down mode;
2. Set to 1 the SW_RESET bit of CTRL3_C to 1;
3. Monitor software reset status, two possibilities:
   a. Wait 50 µs
   b. Poll SW_RESET bit of CTRL3_C until it returns to 0.

In order to avoid conflicts, the reboot and the sw reset must not be executed at the same time (do not set to 1 at the same time both the BOOT bit and SW_RESET bit of CTRL3_C register). The above flows must be performed serially.

5.5 Timestamp

Together with sensor data, the device can provide timestamp information.

To enable this functionality the TIMESTAMP_EN bit of the CTRL10_C register has to be set to 1. The time step count is given by the concatenation of the TIMESTAMP3 & TIMESTAMP2 & TIMESTAMP1 & TIMESTAMP0 registers and is represented as a 32-bit unsigned number.

The nominal timestamp resolution is 12.5 µs. It is possible to get the actual timestamp resolution value through the FREQ_FINE[7:0] bits of the INTERNAL_FREQ_FINE register, which contains the difference in percentage of the actual ODR (and timestamp rate) with respect to the nominal value.

$$t_{actual}[s] = \frac{1}{80000 \cdot (1 + 0.0015 \cdot FREQ\_FINE)}$$

Similarly, it is possible to get the actual output data rate by using the following formula:

$$ODR_{actual}[Hz] = 26667 * (1 + (0.0015 * INTERNAL\_FREQ\_FINE))$$

If the accelerometer is in Power-Down mode, the timestamp counter does not work and the timestamp value is frozen at the last value.

When the maximum value 4294967295 LSB (equal to FFFFFFFFh) is reached corresponding to approximately 30 hours, the counter is automatically reset to 00000000h and continues to count. The timer count can be reset to zero at any time by writing the reset value AAh in the TIMESTAMP2 register.

The TIMESTAMP_ENDCOUNT bit of the ALL_INT_SRC goes high 3.2 ms before the occurrence of a timestamp overrun condition. This flag is reset when the ALL_INT_SRC register is read. It is also possible to route this signal on the INT2 pin (37.5 µs duration pulse) by setting the INT2_TIMESTAMP bit of MD2_CFG to 1.

The timestamp can be batched in FIFO (see Section 6 First-in, first-out (FIFO) buffer for details).

6 First-in, first-out (FIFO) buffer

In order to limit intervention by the host processor and facilitate post-processing data for event recognition, the IIS3DWB embeds a 3 kbyte first-in, first-out buffer (FIFO).

The FIFO can be configured to store the following data:

- accelerometer sensor data;
- timestamp data;
- temperature sensor data.

FIFO is divided into 512 words of 7 bytes each. A FIFO word contains one byte with TAG information and 6 bytes of data: the overall FIFO buffer dimension is equal to 3584 bytes and can contain 3072 bytes of data. The TAG byte contains the information indicating which data is stored in the FIFO data field and other useful information.

Writing data in the FIFO is triggered by the accelerometer data-ready signal. Data can be retrieved from the FIFO through six dedicated registers, from address 79h to 7Eh: FIFO_DATA_OUT_X_L, FIFO_DATA_OUT_X_H, FIFO_DATA_OUT_Y_L, FIFO_DATA_OUT_Y_H, FIFO_DATA_OUT_Z_L, FIFO_DATA_OUT_Z_H.

The reconstruction of a FIFO stream is a simple task thanks to the FIFO_TAG field of FIFO_DATA_OUT_TAG register that allows recognizing the meaning of a word in FIFO.

Six different FIFO operating modes can be chosen through the FIFO_MODE[2:0] bits of the FIFO_CTRL4 register:

- Bypass mode;
- FIFO mode;
- Continuous mode;
- Continuous-to-FIFO mode;
- Bypass-to-Continuous mode;
- Bypass-to-FIFO mode.

To monitor the FIFO status (full, overrun, number of samples stored, etc…), two dedicated registers are available: FIFO_STATUS1 and FIFO_STATUS2.

Programmable FIFO threshold can be set in FIFO_CTRL1 and FIFO_CTRL2 using the WTM[8:0] bits.

FIFO full, FIFO threshold and FIFO overrun events can be enabled to generate dedicated interrupts on the two interrupt pins (INT1 and INT2) through the INT1_FIFO_FULL, INT1_FIFO_FTH and INT1_FIFO_OVR bits of the INT1_CTRL register, and through the INT2_ FIFO_FULL, INT2_FIFO_FTH and INT2_FIFO_OVR bits of the INT2_CTRL register.

6.1 FIFO registers

The FIFO buffer is managed by:

- Six control registers: FIFO_CTRL1, FIFO_CTRL2, FIFO_CTRL3, FIFO_CTRL4, COUNTER_BDR_REG1, COUNTER_BDR_REG2;
- Two status registers: FIFO_STATUS1 and FIFO_STATUS2;
- Seven output registers (tag + data): FIFO_DATA_OUT_TAG, FIFO_DATA_OUT_X_L, FIFO_DATA_OUT_X_H, FIFO_DATA_OUT_Y_L, FIFO_DATA_OUT_Y_H, FIFO_DATA_OUT_Z_L, FIFO_DATA_OUT_Z_H;
- Some additional bits to route FIFO events to the two interrupt lines: INT1_CNT_BDR, INT1_FIFO_FULL, INT1_FIFO_OVR, INT1_FIFO_TH bits of the INT1_CTRL register and INT2_CNT_BDR, INT2_FIFO_FULL, INT2_FIFO_OVR, INT2_FIFO_TH bits of the INT2_CTRL register.

6.1.1 FIFO_CTRL1

The FIFO_CTRL1 register contains the lower part of the 9-bit FIFO watermark threshold level. For the complete watermark threshold level configuration, consider also the WTM8 bit of the FIFO_CTRL2 register. 1 LSB value of the FIFO threshold level is referred to as a FIFO word (7 bytes).

The FIFO watermark flag (FIFO_WTM_IA bit in the FIFO_STATUS2 register) rises when the number of bytes stored in the FIFO is equal to or higher than the watermark threshold level.

In order to limit the FIFO depth to the watermark level, the STOP_ON_WTM bit must be set to 1 in the FIFO_CTRL2 register.

Table 10. FIFO_CTRL1 register

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WTM7	WTM6	WTM5	WTM4	WTM3	WTM2	WTM1	WTM0

6.1.2 FIFO_CTRL2

Table 11. FIFO_CTRL2 register

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
STOP_ ON_WTM	0	0	0	0	0	0	WTM8

The FIFO_CTRL2 register contains the upper part of the 9-bit FIFO watermark threshold level (WTM8 bit). For the complete watermark threshold level configuration, consider also the WTM[7:0] bits of the FIFO_CTRL1 register. The register contains the bit STOP_ON_WTM which allows limiting the FIFO depth to the watermark level.

6.1.3 FIFO_CTRL3

Table 12. FIFO_CTRL3 register

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	BDR_XL_3	BDR_XL_2	BDR_XL_1	BDR_XL_0

The FIFO_CTRL3 register contains the fields to select the writing frequency in FIFO.

The following table indicates the allowed configurations for the BDR_XL[3:0] bits; other configuration values are not allowed.

Table 13. Accelerometer batching data rate

BDR_XL[3:0]	Batching data rate [Hz]
0000	Not batched in FIFO
1010	26667

6.1.4 FIFO_CTRL4

The FIFO_CTRL4 register contains the fields to select the decimation factor for timestamp batching in FIFO and the batching data rate for the temperature sensor.

The timestamp writing rate corresponds to the accelerometer batching data rate divided by the decimation factor specified in the DEC_TS_BATCH_[1:0] field. The programmable decimation factors are indicated in the table below.

Table 14. Timestamp batching data rate

DEC_TS_BATCH[1:0]	Timestamp batching data rate [Hz]
00	Not batched in FIFO (default)
01	BDR_XL[Hz] (no decimation)
10	BDR_XL[Hz] / 8
11	BDR_XL[Hz] / 32

The temperature batching data rate is configurable through the ODR_T_BATCH_[1:0] field as shown in the table below.

Table 15. Temperature sensor batching data rate

ODR_T_BATCH[1:0]	Temperature batching data rate [Hz]
00	Not batched in FIFO (default)
11	104

The FIFO_CTRL4 register also contains the FIFO operating modes bits. FIFO operating modes are described in Section 6.3 FIFO modes.

Table 16. FIFO_CTRL4 register

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DEC_TS_ BATCH_1	DEC_TS_ BATCH_0	ODR_T_ BATCH_1	ODR_T_ BATCH_0	0	FIFO_ MODE2	FIFO_ MODE1	FIFO_ MODE0

6.1.5 COUNTER_BDR_REG1

Since the FIFO might contain accelerometer, timestamp and temperature data, the FIFO provides a way to synchronize the FIFO reading on the basis of the accelerometer actual number of samples stored in FIFO: the BDR counter.

The BDR counter can be configured through the COUNTER_BDR_REG1 and COUNTER_BDR_REG2 registers.

Table 17. COUNTER_BDR_REG1 register

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
-	RST_COUNTER_BDR	0	0	0	CNT_BDR_TH_10	CNT_BDR_TH_9	CNT_BDR_TH_8

The RST_COUNTER_BDR bit can be asserted to reset the BDR counter: it is automatically reset to zero.

The user can select the threshold which generates the COUNTER_BDR_IA event in the FIFO_STATUS2 register. Once the internal BDR counter reaches the threshold, the COUNTER_BDR_IA bit is set to 1. The threshold is configurable through the CNT_BDR_TH_[10:0] bits. The upper part of the field is contained in register COUNTER_BDR_REG1. 1 LSB value of the CNT_BDR_TH threshold level is referred to as one accelerometer sample (X, Y and Z data).

6.1.6 COUNTER_BDR_REG2

The COUNTER_BDR_REG2 register contains the lower part of the BDR-counter threshold.

Table 18. COUNTER_BDR_REG2 register

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CNT_BDR_TH_7	CNT_BDR_TH_6	CNT_BDR_TH_5	CNT_BDR_TH_4	CNT_BDR_TH_3	CNT_BDR_TH_2	CNT_BDR_TH_1	CNT_BDR_TH_0

6.1.7 FIFO_STATUS1

The FIFO_STATUS1 register, together with the FIFO_STATUS2 register, provides information about the number of samples stored in the FIFO. 1 LSB value of the DIFF_FIFO level is referred to as a FIFO word (7 bytes).

Table 19. FIFO_STATUS1 register

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DIFF_FIFO_7	DIFF_FIFO_6	DIFF_FIFO_5	DIFF_FIFO_4	DIFF_FIFO_3	DIFF_FIFO_2	DIFF_FIFO_1	DIFF_FIFO_0

6.1.8 FIFO_STATUS2

The FIFO_STATUS2 register, together with the FIFO_STATUS1 register, provides information about the number of samples stored in the FIFO and about the current status (watermark, overrun, full, BDR counter) of the FIFO buffer.

Table 20. FIFO_STATUS2 register

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FIFO_ WTM_IA	FIFO_ OVR_IA	FIFO_ FULL_IA	COUNTER _BDR_IA	FIFO_OVR_ LATCHED	0	DIFF_ FIFO_9	DIFF_ FIFO_8

- FIFO_WTM_IA represents the watermark status. This bit goes high when the number of FIFO words (7 bytes each) already stored in the FIFO is equal to or higher than the watermark threshold level. The watermark status signal can be driven to the two interrupt pins by setting to 1 the INT1_FIFO_TH bit of the INT1_CTRL register or the INT2_FIFO_TH bit of the INT2_CTRL register.
- FIFO_OVR_IA goes high when the FIFO is completely filled and at least one sample has already been overwritten to store the new data. This signal can be driven to the two interrupt pins by setting to 1 the INT1_FIFO_OVR bit of the INT1_CTRL register or the INT2_FIFO_OVR bit of the INT2_CTRL register.
- FIFO_FULL_IA goes high when the next set of data that will be stored in FIFO will make the FIFO completely full (i.e. DIFF_FIFO_9 = 1) or generate a FIFO overrun. This signal can be driven to the two interrupt pins by setting to 1 the INT1_FIFO_FULL bit of the INT1_CTRL register or the INT2_FIFO_FULL bit of the INT2_CTRL register.
- COUNTER_BDR_IA represents the BDR-counter status. This bit goes high when the number of accelerometer batched samples reaches the BDR-counter threshold level configured through the CNT_BDR_TH_[10:0] bits of the COUNTER_BDR_REG1 and COUNTER_BDR_REG2 registers. The COUNTER_BDR_IA bit is automatically reset when the FIFO_STATUS2 register is read. The BDR-counter status can be driven to the two interrupt pins by setting to 1 the INT1_CNT_BDR bit of the INT1_CTRL register or the INT2_CNT_BDR bit of the INT2_CTRL register.
- FIFO_OVR_LATCHED, as FIFO_OVR_IA, goes high when the FIFO is completely filled and at least one sample has already been overwritten to store the new data. The difference between the two flags is that FIFO_OVR_LATCHED is reset when the FIFO_STATUS2 register is read, whereas the FIFO_OVR_IA is reset when at least one FIFO word is read. This allows detecting a FIFO overrun condition during reading data from FIFO.
- DIFF_FIFO_[9:8] contains the upper part of the number of unread words stored in the FIFO. The lower part is represented by the DIFF_FIFO_[7:0] bits in FIFO_STATUS1. The value of the DIFF_FIFO_[9:0] field corresponds to the number of 7-byte words in the FIFO.

Register content is updated synchronously to the FIFO write and read operations.

Note: The BDU feature also acts on the FIFO_STATUS1 and FIFO_STATUS2 registers. When the BDU bit is set to 1, it is mandatory to read FIFO_STATUS1 first and then FIFO_STATUS2.

6.1.9 FIFO_DATA_OUT_TAG

By reading the FIFO_DATA_OUT_TAG register, it is possible to understand to which sensor the data of the current reading belongs and to check if data are consistent.

Table 21. FIFO_DATA_OUT_TAG register

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TAG_ SENSOR_4	TAG_ SENSOR_3	TAG_ SENSOR_2	TAG_ SENSOR_1	TAG_ SENSOR_0	TAG_ CNT_1	TAG_ CNT_0	TAG_ PARITY

- TAG_SENSOR_[4:0] field identifies the sensors stored in the 6 data bytes (Table 22);
- TAG_CNT_[1:0] field identifies the FIFO time slot (described in next sections);
- TAG_PARITY bit recognizes if the content of the FIFO_DATA_OUT_TAG register is corrupted.

The table below contains all the possible values and associated type of sensor for the TAG_SENSOR_[4:0] field.

Table 22. TAG_SENSOR field and associated sensor

TAG_SENSOR_[4:0]	Sensor name
0x02	Accelerometer
0x03	Temperature
0x04	Timestamp

The TAG_PARITY bit can be used to check the content of the FIFO_DATA_OUT_TAG register. In order to do this, the user can implement the following routine:

1. Read the FIFO_DATA_OUT_TAG register;
2. Count the number of bits equal to 1;
3. If the number of bits equal to 1 is even, then the FIFO_DATA_OUT_TAG content is reliable, otherwise it is unreliable.

6.1.10 FIFO_DATA_OUT

Data can be retrieved from the FIFO through six dedicated registers, from address 79h to address 7Eh: FIFO_DATA_OUT_X_L, FIFO_DATA_OUT_X_H, FIFO_DATA_OUT_Y_L, FIFO_DATA_OUT_Y_H, FIFO_DATA_OUT_Z_L, FIFO_DATA_OUT_Z_H.

The FIFO output registers content depends on the sensor category and type, as described in the next section.

6.2 FIFO batched sensors

As previously described, the FIFO can be configured to store the following data:

- accelerometer sensor data;
- timestamp data;
- temperature sensor data.

In this section, details about batched sensors will be presented.

6.2.1 Accelerometer data in FIFO

The accelerometer batching data rate can be configured through the BDR_XL_[3:0] bits of the FIFO_CTRL3 register.

The accelerometer sensor defines the FIFO time base. This means that each one of the other data stored in FIFO (timestamp, temperature) can be associated to a time base slot defined by the accelerometer sensor. A batching event of the accelerometer sensor increments the TAG counter (TAG_CNT field of FIFO_DATA_OUT_TAG register): this counter is composed of two bits and its value is continuously incremented (from 00b to 11b) to identify different time slots.

The FIFO word format of the accelerometer sensor is presented in the table below, representing the device addresses from 78h to 7Eh.

Table 23. Main sensors output data format in FIFO

TAG	X_L	X_H	Y_L	Y_H	Z_L	Z_H

6.2.2 Timestamp and temperature data in FIFO

In addition to accelerometer data, it is possible to store in FIFO the data of the following sensors:

- Temperature sensor (ODR_T_BATCH_[1:0] bits of the FIFO_CTRL4 register must be configured properly);
- Timestamp sensor: it stores the timestamp corresponding to a FIFO time slot (TIMESTAMP_EN bit of the CTRL10_C register must be set to 1 and the DEC_TS_BATCH_[1:0] bits of the FIFO_CTRL4 register must be configured properly).

Temperature and timestamp sensors cannot trigger a write in FIFO. Their registers are written in FIFO when an accelerometer data-ready signal event occurs.

The temperature output data format in FIFO is presented in the following table.

Table 24. Temperature output data format in FIFO

Data	FIFO_DATA_OUT registers
TEMPERATURE[7:0]	FIFO_DATA_OUT_X_L
TEMPERATURE[15:8]	FIFO_DATA_OUT_X_H
0	FIFO_DATA_OUT_Y_L
0	FIFO_DATA_OUT_Y_H
0	FIFO_DATA_OUT_Z_L
0	FIFO_DATA_OUT_Z_H

The timestamp output data format in FIFO is presented in the following table.

Table 25. **Timestamp output data format in FIFO**

Data	FIFO_DATA_OUT registers
TIMESTAMP[7:0]	FIFO_DATA_OUT_X_L
TIMESTAMP[15:8]	FIFO_DATA_OUT_X_H
TIMESTAMP[23:16]	FIFO_DATA_OUT_Y_L
TIMESTAMP[31:24]	FIFO_DATA_OUT_Y_H
0	FIFO_DATA_OUT_Z_L[3:0]
0	FIFO_DATA_OUT_Z_L[7:4]
BDR_XL	FIFO_DATA_OUT_Z_H[3:0]
0	FIFO_DATA_OUT_Z_H[7:4]

6.3 FIFO modes

The IIS3DWB FIFO buffer can be configured to operate in six different modes, selectable through the FIFO_MODE_[2:0] field of the FIFO_CTRL4 register. The available configurations ensure a high level of flexibility and extend the number of functions usable in application development.

Bypass, FIFO, Continuous, Continuous-to-FIFO, Bypass-to-Continuous, and Bypass-to-FIFO modes are described in the following paragraphs.

6.3.1 Bypass mode

When Bypass mode is enabled, the FIFO is not used, the buffer content is cleared, and it remains empty until another mode is selected. Bypass mode is selected when the FIFO_MODE_[2:0] bits are set to 000b. Bypass mode must be used in order to stop and reset the FIFO buffer when a different mode is intended to be used. Note that by placing the FIFO buffer into Bypass mode, the whole buffer content is cleared.

6.3.2 FIFO mode

In FIFO mode, the buffer continues filling until it becomes full. Then it stops collecting data and the FIFO content remains unchanged until a different mode is selected.

Follow these steps for FIFO mode configuration:

1. Enable the sensor data to be stored in FIFO with the corresponding batching data rate;
2. Set the FIFO_MODE_[2:0] bits in the FIFO_CTRL4 register to 001b to enable FIFO mode.

When this mode is selected, the FIFO starts collecting data. The FIFO_STATUS1 and FIFO_STATUS2 registers are updated according to the number of samples stored.

When the FIFO is full, the DIFF_FIFO_9 bit of the FIFO_STATUS2 register is set to 1 and no more data are stored in the FIFO buffer. Data can be retrieved by reading all the FIFO_DATA_OUT (from 78h to 7Eh) registers for the number of times specified by the DIFF_FIFO_[9:0] bits of the FIFO_STATUS1 and FIFO_STATUS2 registers.

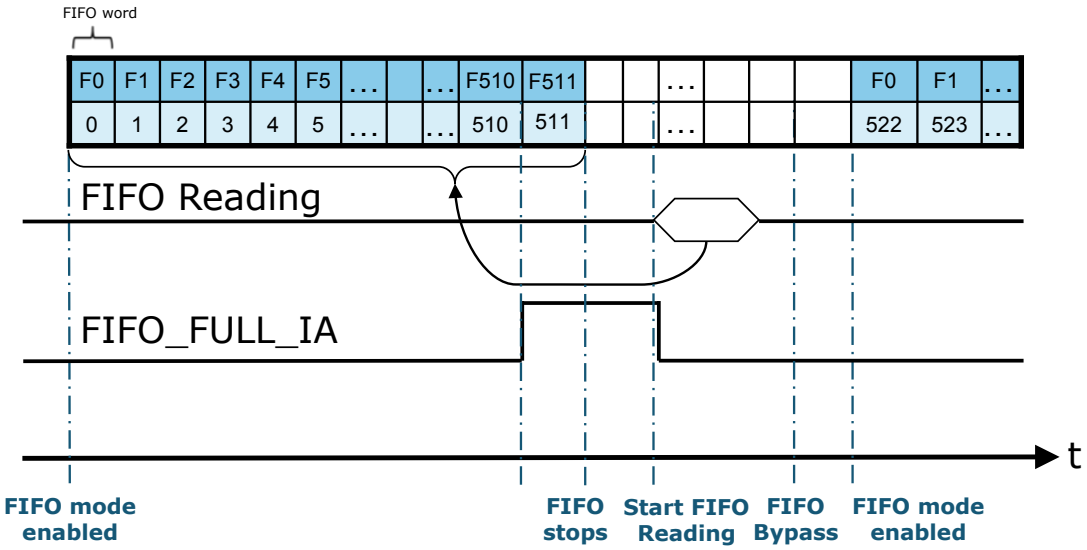
Using the FIFO_WTM_IA bit of the FIFO_STATUS2 register, data can also be retrieved when a threshold level (WTM[8:0] in FIFO_CTRL1 and FIFO_CTRL2 registers) is reached if the application requires a lower number of samples in the FIFO.

If the STOP_ON_WTM bit of the FIFO_CTRL2 register is set to 1, the FIFO size is limited to the value of the WTM[8:0] bits in the FIFO_CTRL1 and FIFO_CTRL2 registers. In this case, the FIFO_FULL_IA bit of the FIFO_STATUS2 register is set high when the number of samples in FIFO will reach or exceed the WTM[8:0] value on the next FIFO write operation.

Communication speed is not very important in FIFO mode because the data collection is stopped and there is no risk of overwriting data already acquired. Before restarting the FIFO mode, it is necessary to set to Bypass mode first in order to completely clear the FIFO content.

Figure 7. FIFO mode (STOP_ON_WTM = 0) shows an example of FIFO mode usage; in this example, just accelerometer data are stored in the FIFO. In these conditions, the number of accelerometer samples that can be stored in the FIFO buffer is 512. The FIFO_FULL_IA bit of the FIFO_STATUS2 register goes high just after the level labeled as 510 to notify that the FIFO buffer will be completely filled at the next FIFO write operation. After the FIFO is full (FIFO_DIFF_9 = 1), the data collection stops.

Figure 7. FIFO mode (STOP_ON_WTM = 0)



6.3.3 Continuous mode

In Continuous mode, the FIFO continues filling. When the buffer is full, the FIFO index restarts from the beginning, and older data are replaced by the new data. The oldest values continue to be overwritten until a read operation frees FIFO slots. The host processor reading speed is important in order to free slots faster than new data is made available. To stop this configuration, Bypass mode must be selected.

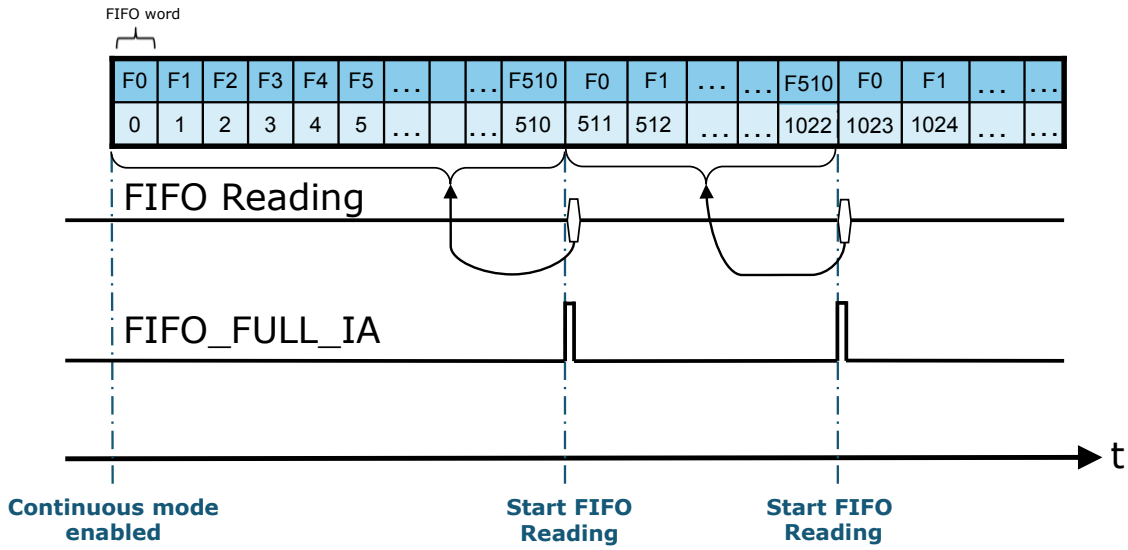
Follow these steps for Continuous mode configuration:

1. Enable the sensor data to be stored in FIFO with the corresponding batching data rate;
2. Set the FIFO_MODE_[2:0] bits in the FIFO_CTRL4 register to 110b to enable FIFO mode.

When this mode is selected, the FIFO collects data continuously. The FIFO_STATUS1 and FIFO_STATUS2 registers are updated according to the number of samples stored. When the next FIFO write operation will make the FIFO completely full or generate a FIFO overrun, the FIFO_FULL_IA bit of the FIFO_STATUS2 register goes to 1. The FIFO_OVR_ IA and FIFO_OVR_LATCHED bits in the FIFO_STATUS2 register indicates when at least one FIFO word has been overwritten to store the new data. Data can be retrieved after the FIFO_FULL_IA event by reading the FIFO_DATA_OUT (from 78h to 7Eh) registers for the number of times specified by the DIFF_FIFO_[9:0] bits in the FIFO_STATUS1 and FIFO_STATUS2 registers. Using the FIFO_WTM_IA bit of the FIFO_STATUS2 register, data can also be retrieved when a threshold level (WTM[8:0] in the FIFO_CTRL1 and FIFO_CTRL2 registers) is reached. If the STOP_ON_WTM bit of the FIFO_CTRL2 register is set to 1, the FIFO size is limited to the value of the WTM[8:0] bits in the FIFO_CTRL1 and FIFO_CTRL2 registers. In this case, the FIFO_FULL_IA bit of the FIFO_STATUS2 register goes high when the number of samples in FIFO will reach or overcome the WTM[8:0] value on the next FIFO write operation.

Figure 8. Continuous mode shows an example of the Continuous mode usage. In the example, just accelerometer sensor data are stored in the FIFO and the FIFO samples are read on the FIFO_FULL_IA event and faster than 1 * ODR so that no data is lost. In these conditions, the number of samples stored is 511.

Figure 8. Continuous mode



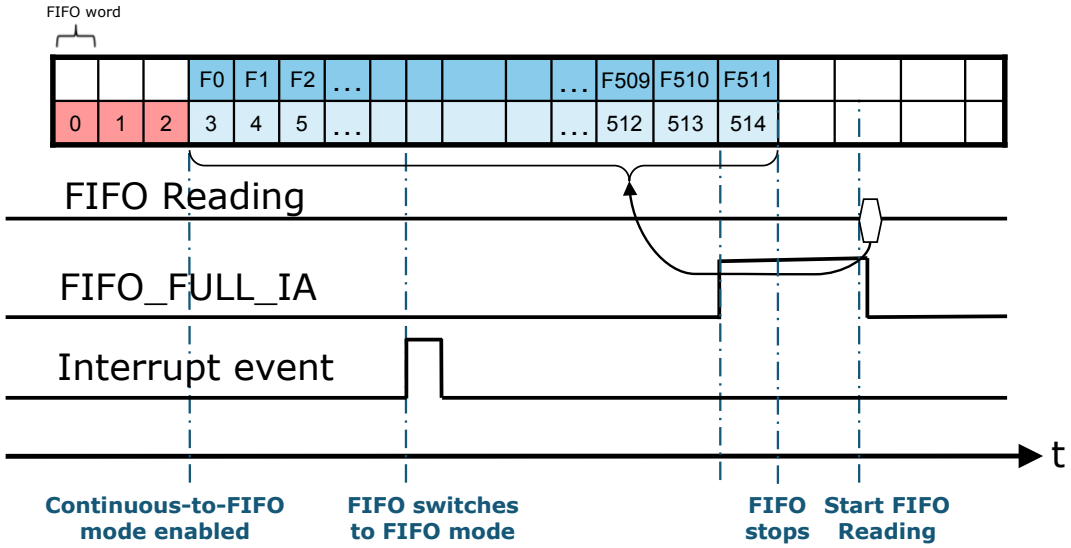
6.3.4 Continuous-to-FIFO mode

This mode is a combination of the Continuous and FIFO modes previously described. In Continuous-to-FIFO mode, the FIFO buffer starts operating in Continuous mode and switches to FIFO mode when a wake-up event condition occurs. Wake-up event detection has to be properly configured and the INT2_WU bit of the MD2_CFG register has to be set to 1.

The event condition can be one of the following:

Continuous-to-FIFO mode is sensitive to the edge of the interrupt signal. At the first interrupt event, FIFO changes from Continuous mode to FIFO mode and maintains it until Bypass mode is set.

Figure 9. Continuous-to-FIFO mode



Follow these steps for Continuous-to-FIFO mode configuration:

1. Configure the wake-up event detection as previously described;
2. Enable the sensor data to be stored in FIFO with the corresponding batching data rate;
3. Set the FIFO_MODE_[2:0] bits in the FIFO_CTRL4 register to 011b to enable FIFO Continuous-to-FIFO mode.

In Continuous-to-FIFO mode the FIFO buffer continues filling. When the FIFO will be full or overrun at the next FIFO write operation, the FIFO_FULL_IA bit goes high.

If the STOP_ON_WTM bit of the FIFO_CTRL2 register is set to 1, the FIFO size is limited to the value of the WTM[8:0] bits in the FIFO_CTRL1 and FIFO_CTRL2 registers. In this case, the FIFO_FULL_IA bit of the FIFO_STATUS2 register goes high when the number of samples in FIFO will reach or exceed the WTM[8:0] value on the next FIFO write operation.

When the trigger event occurs, two different cases can be observed:

1. If the FIFO buffer is already full, it stops collecting data at the first sample after the event trigger. The FIFO content is composed of the samples collected before the event.
2. If FIFO buffer is not full yet, it continues filling until it becomes full and then it stops collecting data.

Continuous-to-FIFO can be used in order to analyze the history of the samples which have generated a wake-up interrupt. The standard operation is to read the FIFO content when the FIFO mode is triggered and the FIFO buffer is full and stopped.

6.3.5 Bypass-to-Continuous mode

This mode is a combination of the Bypass and Continuous modes previously described. In Bypass-to-Continuous mode, the FIFO buffer starts operating in Bypass mode and switches to Continuous mode when a wake-up event condition occurs. Wake-up event detection has to be properly configured and the INT2_WU bit of the MD2_CFG register has to be set to 1.

Bypass-to-Continuous mode is sensitive to the edge of the interrupt signal: at the first interrupt event, FIFO changes from Bypass mode to Continuous mode and maintains it until Bypass mode is set.

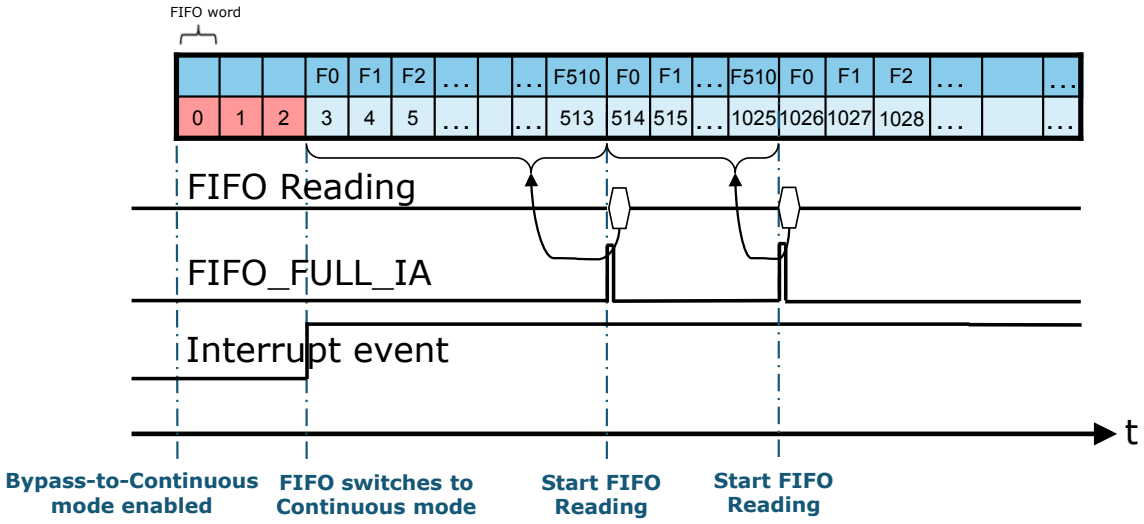
Follow these steps for Bypass-to-Continuous mode configuration:

1. Configure the wake-up event detection as previously described;
2. Enable the sensor data to be stored in FIFO with the corresponding batching data rate;
3. Set the FIFO_MODE[2:0] bits in the FIFO_CTRL4 register to 100b to enable FIFO Bypass-to-Continuous mode.

Once the trigger condition appears and the buffer switches to Continuous mode, the FIFO buffer continues filling. When the next stored set of data will make the FIFO full or overrun, the FIFO_FULL_IA bit is set high.

Bypass-to-Continuous can be used in order to start the acquisition when the configured interrupt is generated.

Figure 10. Bypass-to-Continuous mode



6.3.6 Bypass-to-FIFO mode

This mode is a combination of the Bypass and FIFO modes previously described. In Bypass-to-FIFO mode, the FIFO buffer starts operating in Bypass mode and switches to FIFO mode when a wake-up event condition occurs. Wake-up event detection has to be properly configured and the INT2_WU bit of the MD2_CFG register has to be set to 1.

Bypass-to-FIFO mode is sensitive to the edge of the interrupt signal. At the first interrupt event, FIFO changes from Bypass mode to FIFO mode and maintains it until Bypass mode is set.

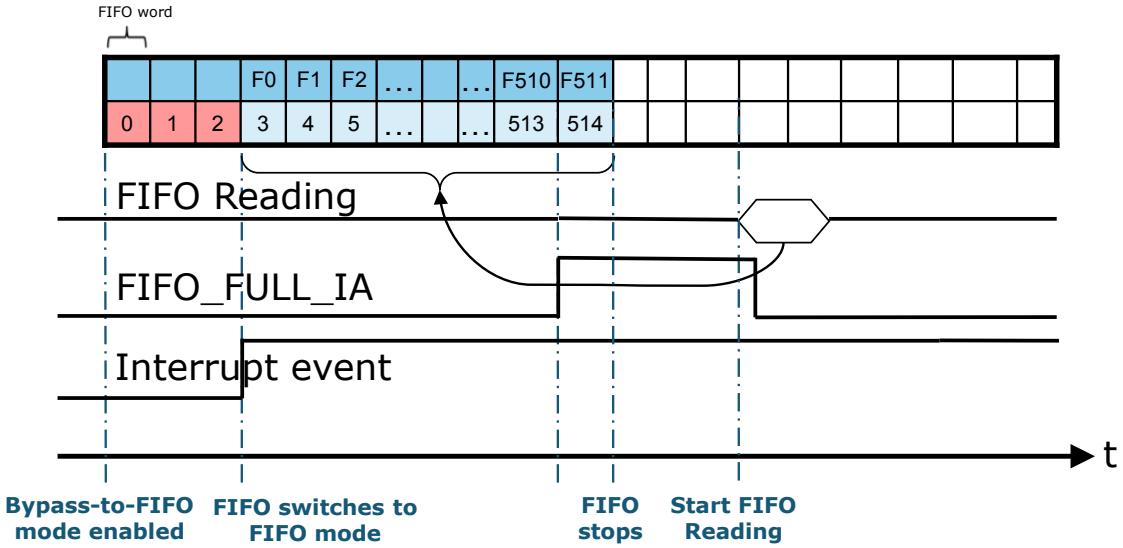
Follow these steps for Bypass-to-FIFO mode configuration:

1. Configure the wake-up event detection as previously described;
2. Enable the sensor data to be stored in FIFO with the corresponding batching data rate;
3. Set the FIFO_MODE_[2:0] bits in the FIFO_CTRL4 register to 111b to enable FIFO Bypass-to-FIFO mode.

Once the trigger condition appears and the buffer switches to FIFO mode, the FIFO buffer starts filling. When the next stored set of data will make the FIFO full or overrun, the FIFO_FULL_IA bit is set high and the FIFO stops.

Bypass-to-FIFO can be used in order to analyze the history of the samples which have generated an interrupt.

Figure 11. Bypass-to-FIFO mode



6.4 Retrieving data from the FIFO

When FIFO is enabled and the mode is different from Bypass, reading the FIFO output registers return the oldest FIFO sample set. Whenever these registers are read, their content is moved to the SPI / I²C output buffer.

FIFO slots are ideally shifted up one level in order to release room for a new sample, and the FIFO output registers load the current oldest value stored in the FIFO buffer.

The recommended way to retrieve data from the FIFO is the following:

1. Read the FIFO_STATUS1 and FIFO_STATUS2 registers to check how many words are stored in the FIFO. This information is contained in the DIFF_FIFO_[9:0] bits.
2. For each word in FIFO, read the FIFO word (tag and output data) and interpret it on the basis of the FIFO tag.
3. Go to step 1.

The entire FIFO content is retrieved by performing a certain number of read operations from the FIFO output registers until the buffer becomes empty (DIFF_FIFO_[9:0] bits of the FIFO_STATUS1 and FIFO_STATUS2 register are equal to 0).

It is recommended to avoid reading from FIFO when it is empty.

FIFO output data must be read with multiple of 7 bytes reads starting from the FIFO_DATA_OUT_TAG register. The rounding function from address FIFO_DATA_OUT_Z_H to FIFO_DATA_OUT_TAG is done automatically in the device, in order to allow reading many words with a unique multiple read operation.

6.5 FIFO watermark threshold

The FIFO threshold is a functionality of the IIS3DWB FIFO which can be used to check when the number of samples in the FIFO reaches a defined watermark threshold level.

The bits WTM[8:0] in the FIFO_CTRL1 and FIFO_CTRL2 registers contain the watermark threshold level. The resolution of the WTM[8:0] field is 7 bytes, corresponding to a complete FIFO word. So, the user can select the desired level in a range between 0 and 511.

The bit FIFO_WTM_IA in the FIFO_STATUS2 register represents the watermark status. This bit is set high if the number of words in the FIFO reaches or exceeds the watermark level. FIFO size can be limited to the threshold level by setting the STOP_ON_WTM bit in the FIFO_CTRL2 register to 1.

Figure 12. FIFO threshold (STOP_ON_WTM = 0)

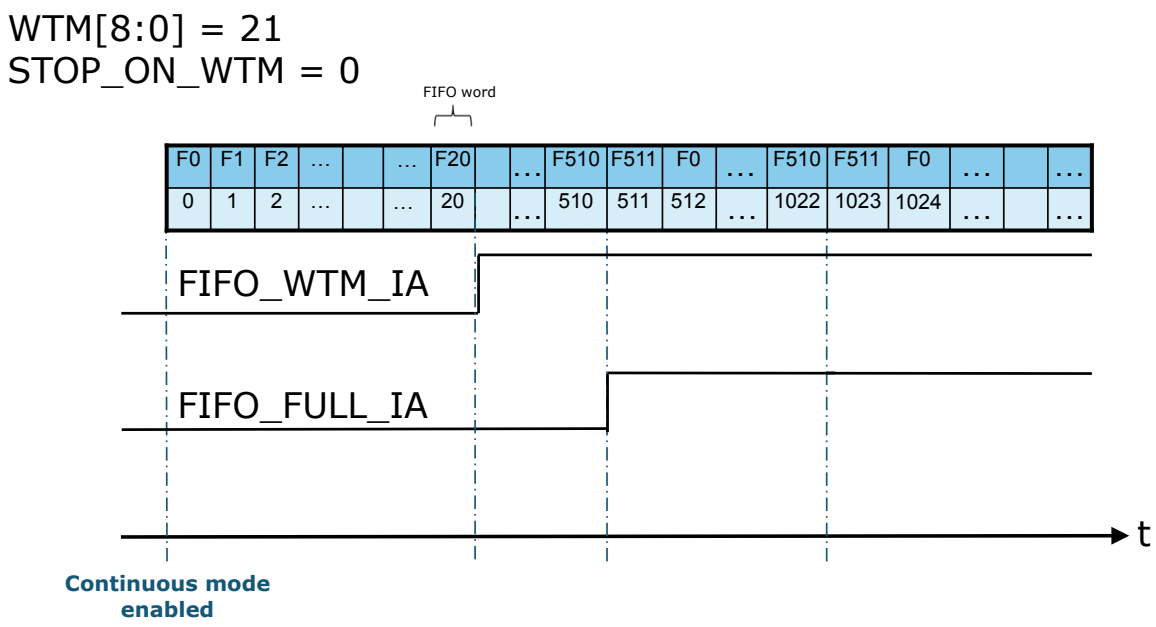


Figure 12. FIFO threshold (STOP_ON_WTM = 0) shows an example of FIFO threshold level usage when just accelerometer data are stored. The STOP_ON_WTM bit set to 0 in the FIFO_CTRL2 register. The threshold level is set to 21 through the WTM[8:0] bits. The FIFO_WTM_IA bit of the FIFO_STATUS2 register rises after the 21st level has been reached (21 words in the FIFO). Since the STOP_ON_WTM bit is set to 0, the FIFO will not stop at the 21st set of data, but will keep storing data until the FIFO_FULL_IA flag is set high.

Figure 13. **FIFO threshold (STOP_ON_WTM = 1) in FIFO mode**

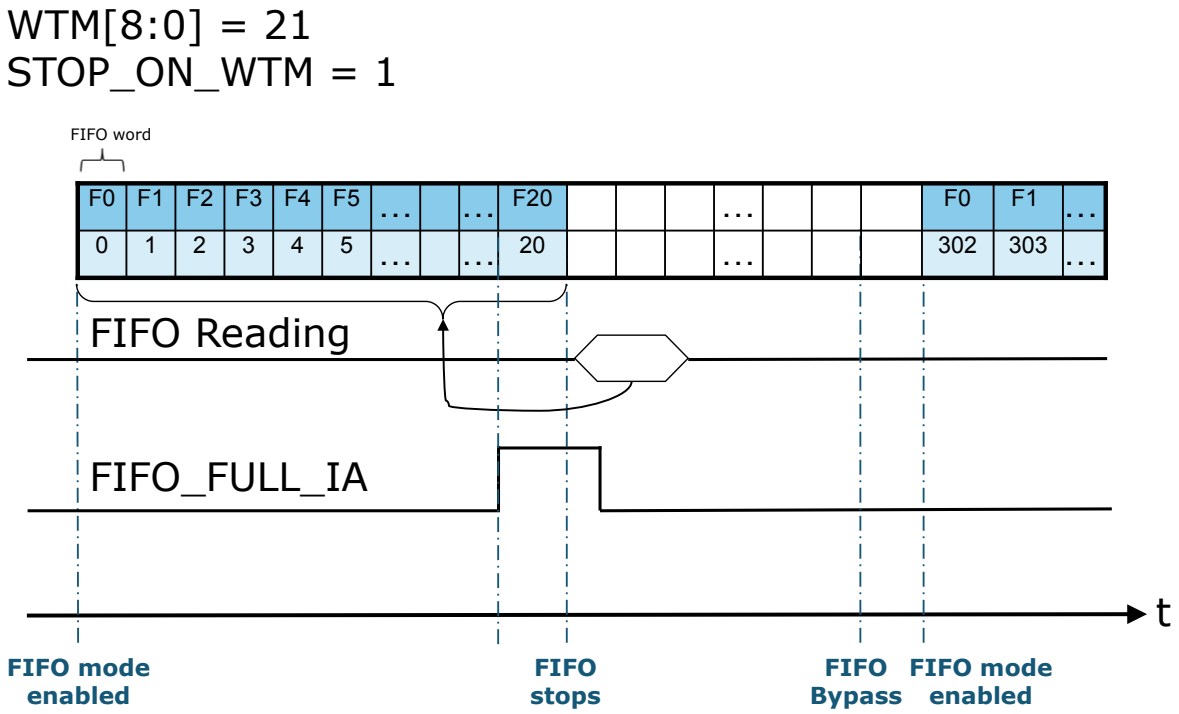


Figure 13. FIFO threshold (STOP_ON_WTM = 1) in FIFO mode shows an example of FIFO threshold level usage in FIFO mode with the STOP_ON_WTM bit set to 1 in the FIFO_CTRL2 register. Just accelerometer data are stored in this example. The threshold level is set to 21 through the WTM[8:0] bits and defines the current FIFO size. In FIFO mode, data are stored in the FIFO buffer until the FIFO is full. The FIFO_FULL_IA bit of the FIFO_STATUS2 register rises when the next data stored in the FIFO will generate the FIFO full or overrun condition. The FIFO_WTM_IA bit of the FIFO_STATUS2 register goes high when the FIFO is full.

Figure 14. **FIFO threshold (STOP_ON_WTM = 1) in Continuous mode**

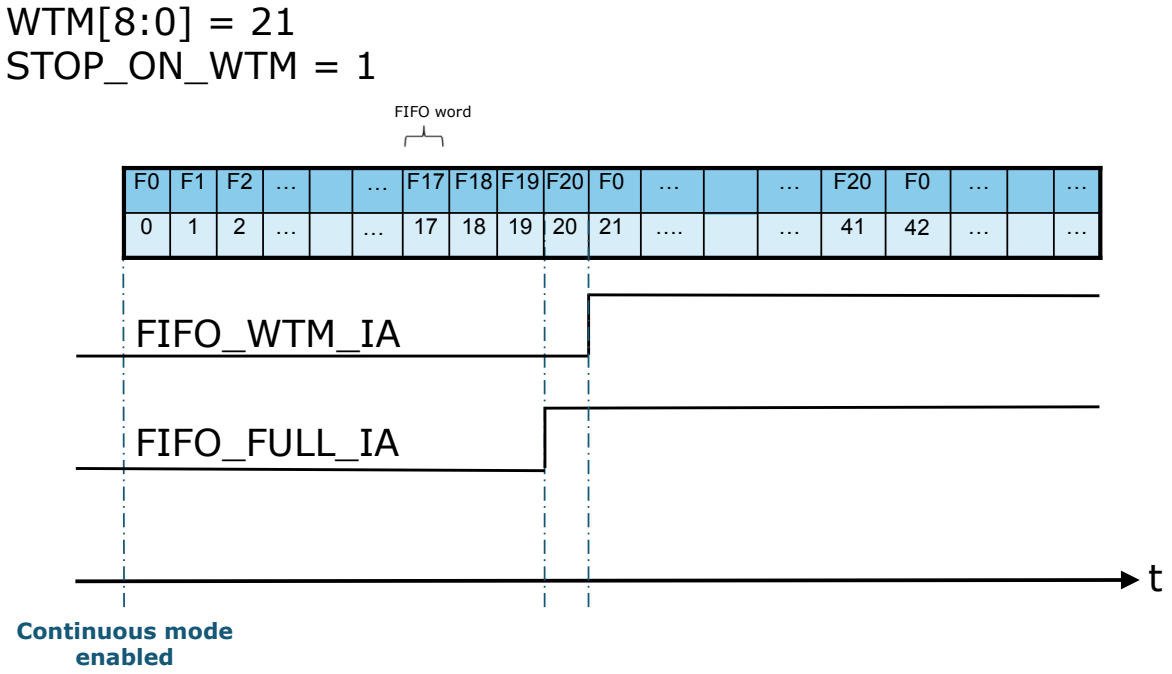


Figure 14. FIFO threshold (STOP_ON_WTM = 1) in Continuous mode shows an example of FIFO threshold level usage in Continuous mode with the STOP_ON_WTM bit set to 1 in the FIFO_CTRL2 register. Just accelerometer data are stored in this example. The threshold level is set to 21 through the WTM[8:0] bits. The FIFO_FULL_IA bit of the FIFO_STATUS2 register rises when the next data stored in the FIFO will make the FIFO full. The FIFO_WTM_IA bit of the FIFO_STATUS2 goes high when the FIFO is full. If data are not retrieved from FIFO, new data (labeled as sample 21) will override the older data stored in FIFO (labeled as sample F0).

6.6 Timestamp correlation

It is possible to reconstruct the timestamp of FIFO stream with three different approaches:

1. Basic, using only timestamp sensor information;
2. Memory-saving, based on the TAG_CNT field in FIFO_DATA_OUT_TAG
3. Hybrid, based on combined usage of the TAG_CNT field and decimated timestamp sensor

The basic approach guarantees the highest precision in timestamp reconstruction but wastes a lot of memory space available in FIFO. The timestamp sensor is written in FIFO at each time slot. If the overrun condition occurs, the correct procedure to retrieve the data from FIFO is to discard each data read before a new timestamp sensor.

The memory-saving approach uses only the TAG_CNT information and, when the TAG_CNT value increases, the timestamp stored at the software layer should be updated as follows:

$$timestamp = timestamp\left(i-1\right) + \frac{1}{BDR\_XL}$$

The memory-saving approach allows the user to maximize the data stored in FIFO. With this method all the timestamp correlation is forwarded to the application processor.

This approach is not recommended when the overrun condition can occur.

The hybrid approach is a trade-off and a combination of the two previous solutions. The timestamp is configured to be written in FIFO with decimation. When the TAG_CNT value increases, the timestamp stored at the software layer should be updated as in the memory-saving approach, while when the timestamp sensor is read, the timestamp stored at the software layer should be realigned with the correct value from the sensor.

7 Temperature sensor

The device is provided with an internal temperature sensor that is suitable for ambient temperature measurement.

If the accelerometer sensor is in Power-Down mode, the temperature sensor is off.

When the accelerometer is enabled, the output data rate of the temperature sensor is 104 Hz.

For the temperature sensor, the data-ready signal is represented by the TDA bit of the STATUS_REG register. The signal can be driven to the INT2 pin by setting the INT2_DRDY_TEMP bit of the INT2_CTRL register to 1.

The temperature data is given by the concatenation of the OUT_TEMP_H and OUT_TEMP_L registers and it is represented as a number of 16 bits in two's complement format with a sensitivity of 256 LSB/°C. The output zero level corresponds to 25 °C. Absolute temperature accuracy can be improved, reducing the effect of temperature offset, by performing OPC (one-point calibration) at room temperature (25 °C).

Temperature sensor data can also be stored in FIFO (see Section 6 First-in, first-out (FIFO) buffer for details).

7.1 Example of temperature data calculation

The following table provides a few basic examples of the data that is read from the temperature data registers at different ambient temperature values. The values listed in this table are given under the hypothesis of perfect device calibration (i.e. no offset, no gain error,....).

Table 26. Content of output data registers vs. temperature

Temperature values	Register address	
	OUT_TEMP_H (21h)	OUT_TEMP_L (20h)
0 °C	E7h	00h
25 °C	00h	00h
50 °C	19h	00h

8 Self-test

The embedded self-test functions allow checking the device functionality without moving it.

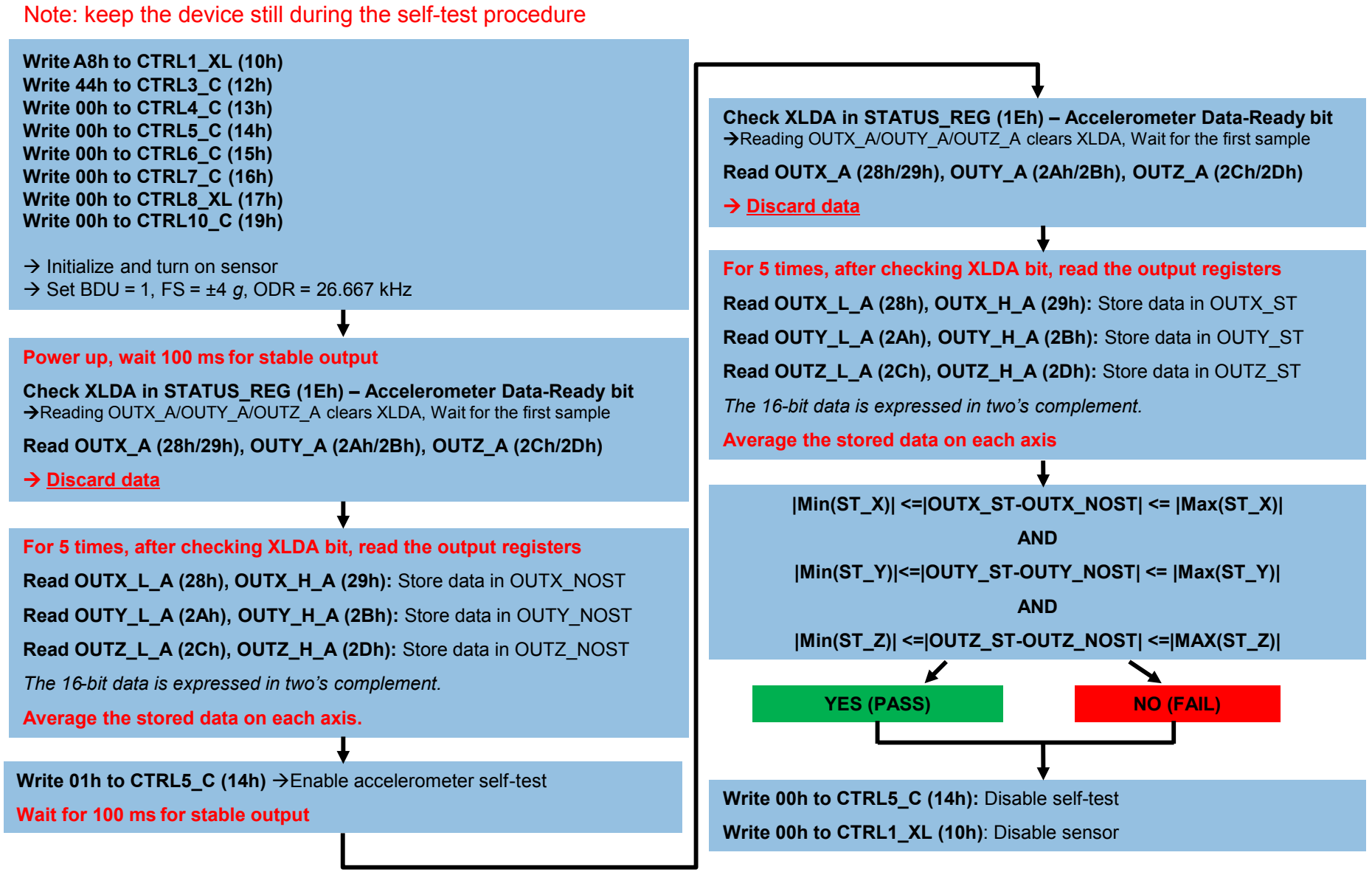
When the accelerometer self-test is enabled, an actuation force is applied to the sensor, simulating a definite input acceleration. In this case, the sensor outputs exhibit a change in their DC levels which are related to the selected full scale through the sensitivity value.

The accelerometer self-test function is off when the ST[1:0]_XL bits of the CTRL5_C register are programmed to 00b; it is enabled when the ST[1:0]_XL bits are set to 01b (positive sign self-test) or 10b (negative sign self-test).

When the accelerometer self-test is activated, the sensor output level is given by the algebraic sum of the signals produced by the acceleration acting on the sensor and by the electrostatic test-force.

The complete accelerometer self-test procedure is indicated in Figure 15. Accelerometer self-test procedure.

Figure 15. **Accelerometer self-test procedure**



Revision history

Table 27. **Document revision history**

Date	Version	Changes
03-Mar-2020	1	Initial release
04-Aug-2020	2	Updated Section 3 Operating modes Updated Section 5.5 Timestamp

Contents

1 **Pin description** 2
2 **Registers** 4
3 **Operating modes** 6
3.1 Accelerometer filtering chain 7
3.1.1 Accelerometer slope filter 9
3.2 Accelerometer turn-on/off time 9
4 **Reading output data** 10
4.1 Startup sequence 10
4.2 Using the status register 10
4.3 Using the data-ready signal 11
4.3.1 DRDY mask functionality 11
4.4 Using the block data update (BDU) feature 11
4.5 Understanding output data 12
4.5.1 Examples of output data 12
4.6 Accelerometer offset registers 12
4.7 Rounding function 12
5 **Interrupt generation** 13
5.1 Interrupt pin configuration 13
5.2 Wake-up interrupt 15
5.3 Activity/Inactivity recognition 18
5.4 Boot status 20
5.5 Timestamp 21
6 **First-in, first-out (FIFO) buffer** 22
6.1 FIFO registers 22
6.1.1 FIFO_CTRL1 23
6.1.2 FIFO_CTRL2 23
6.1.3 FIFO_CTRL3 23
6.1.4 FIFO_CTRL4 24
6.1.5 COUNTER_BDR_REG1 25

6.1.6 COUNTER_BDR_REG2 . . . 25
6.1.7 FIFO_STATUS1 . . . 25
6.1.8 FIFO_STATUS2 . . . 26
6.1.9 FIFO_DATA_OUT_TAG . . . 27
6.1.10 FIFO_DATA_OUT . . . 27
6.2 FIFO batched sensors . . . 28
6.2.1 Accelerometer data in FIFO . . . 28
6.2.2 Timestamp and temperature data in FIFO . . . 28
6.3 FIFO modes . . . 29
6.3.1 Bypass mode . . . 29
6.3.2 FIFO mode . . . 30
6.3.3 Continuous mode . . . 31
6.3.4 Continuous-to-FIFO mode . . . 32
6.3.5 Bypass-to-Continuous mode . . . 33
6.3.6 Bypass-to-FIFO mode . . . 34
6.4 Retrieving data from the FIFO . . . 35
6.5 FIFO watermark threshold . . . 36
6.6 Timestamp correlation . . . 39
7 Temperature sensor . . . 40
7.1 Example of temperature data calculation . . . 40
8 Self-test . . . 41
Revision history . . . 43
Contents . . . 44
List of tables . . . 46
List of figures . . . 47

List of tables

Table 1. Default pin status . . . 3
Table 2. Registers . . . 4
Table 3. Accelerometer active axis . . . 6
Table 4. Accelerometer bandwidth selection . . . 8
Table 5. Content of output data registers vs. acceleration (FS_XL = ±2 *g*) . . . 12
Table 6. INT1_CTRL register . . . 13
Table 7. MD1_CFG register . . . 13
Table 8. INT2_CTRL register . . . 14
Table 9. MD2_CFG register . . . 14
Table 10. FIFO_CTRL1 register . . . 23
Table 11. FIFO_CTRL2 register . . . 23
Table 12. FIFO_CTRL3 register . . . 23
Table 13. Accelerometer batching data rate . . . 23
Table 14. Timestamp batching data rate . . . 24
Table 15. Temperature sensor batching data rate . . . 24
Table 16. FIFO_CTRL4 register . . . 24
Table 17. COUNTER_BDR_REG1 register . . . 25
Table 18. COUNTER_BDR_REG2 register . . . 25
Table 19. FIFO_STATUS1 register . . . 25
Table 20. FIFO_STATUS2 register . . . 26
Table 21. FIFO_DATA_OUT_TAG register . . . 27
Table 22. TAG_SENSOR field and associated sensor . . . 27
Table 23. Main sensors output data format in FIFO . . . 28
Table 24. Temperature output data format in FIFO . . . 28
Table 25. Timestamp output data format in FIFO . . . 29
Table 26. Content of output data registers vs. temperature . . . 40
Table 27. Document revision history . . . 43

List of figures

Figure 1. Pin connections . . . 2
Figure 2. Accelerometer filtering chain . . . 7
Figure 3. Accelerometer slope filter . . . 9
Figure 4. Data-ready signal . . . 11
Figure 5. Wake-up interrupt (using the slope filter) . . . 16
Figure 6. Activity/Inactivity recognition (using the slope filter) . . . 19
Figure 7. FIFO mode (STOP_ON_WTM = 0) . . . 30
Figure 8. Continuous mode . . . 31
Figure 9. Continuous-to-FIFO mode . . . 32
Figure 10. Bypass-to-Continuous mode . . . 33
Figure 11. Bypass-to-FIFO mode . . . 34
Figure 12. FIFO threshold (STOP_ON_WTM = 0) . . . 36
Figure 13. FIFO threshold (STOP_ON_WTM = 1) in FIFO mode . . . 37
Figure 14. FIFO threshold (STOP_ON_WTM = 1) in Continuous mode . . . 38
Figure 15. Accelerometer self-test procedure . . . 42