MIS2DU12: ultralow-power accelerometer with antialiasing and motion detection

Introduction

This document provides usage information and application hints related to ST's MIS2DU12 motion sensor.

The MIS2DU12 is a 3-axis digital accelerometer system-in-package with a digital I²C/SPI/MIPI I3C® serial interface standard output, performing at 6.1 µA in normal mode at 800 Hz and below 0.47 µA in ultralow-power mode. The accelerometer features smart sleep-to-wake-up (activity) and return-to-sleep (inactivity) functions that allow advanced power saving. The embedded self-test capability allows the user to check that the sensor works in the final application.

The device has a dynamic user-selectable full-scale acceleration range of ±2/±4/±8/±16 *g* and is capable of measuring accelerations with output data rates from 1.6 Hz to 800 Hz. The MIS2DU12 can be configured to generate interrupt signals by using hardware recognition of free-fall events, 6D orientation, tap and double-tap sensing, activity or inactivity, and wake-up events.

The MIS2DU12 has an integrated 128-level first-in, first-out (FIFO) buffer allowing the user to store acceleration and temperature data in order to limit intervention by the host processor.

The MIS2DU12 is available in a small thin plastic, land grid array package (LGA) and it is guaranteed to operate over an extended temperature range from -40°C to +85°C.

The ultrasmall package makes it an ideal solution for all applications where size is a key feature.

1 Pin description

Figure 1. Pin connections

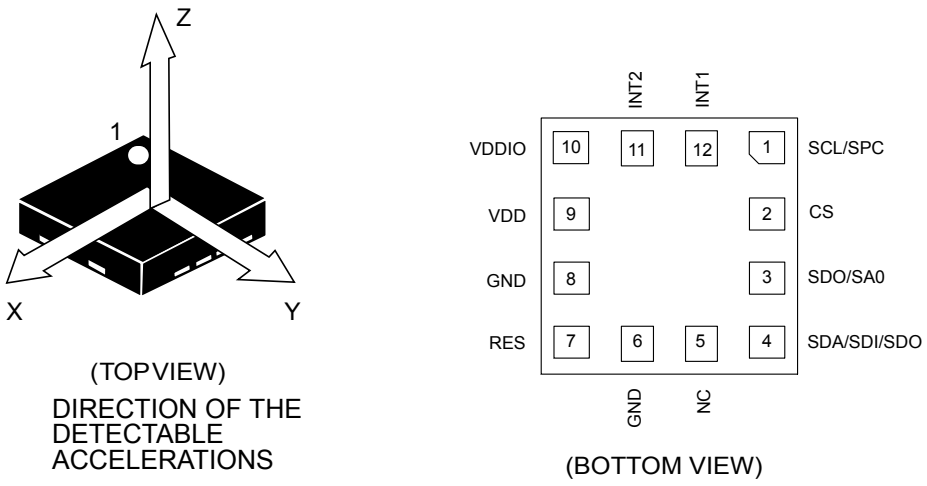


Table 1. Internal pin status

Pin #	Name	Function	Pin status
1	SCL SPC	I²C/MIPI I3C® serial clock (SCL) SPI serial port clock (SPC)	Default: input without internal pull-up
2	CS	SPI/I²C/MIPI I3C® mode selection (1: SPI idle mode / I²C/MIPI I3C® enabled; 0: SPI enabled / I²C/MIPI I3C® disabled)	Default: input with internal pull-up(1)
3	SDO SA0	SPI serial data output (SDO) I²C less significant bit of the device address (SA0)	Default: input with internal pull-up(2)
4	SDA SDI SDO	I²C/MIPI I3C® serial data (SDA) SPI serial data input (SDI) 3-wire interface serial data output (SDO)	Default: (SDA) input without internal pull-up(3)
5	NC	Internally not connected. Can be tied to Vdd, Vdd_IO, or GND	
6	GND	0 V supply	
7	RES	Connect to GND	
8	GND	0 V supply	
9	Vdd	Power supply	
10	Vdd_IO	Power supply for I/O pins	
11	INT2	Interrupt pin 2 Clock input when selected in single data conversion on demand	Default: input with internal pull-down(4)
12	INT1	Interrupt pin 1	Default: input with internal pull-down(5)

1. *The pull-up of the CS pin can be disconnected by setting the CS_PU_DIS bit of register IF_PU_CTRL (0Ch) to 1.*
2. *The pull-up of the SDO/SA0 pin can be disconnected by setting the SDO_PU_DIS bit of register IF_PU_CTRL (0Ch) to 1.*
3. *The internal pull-up of the SDA/SDI/SDO pin can be connected by setting the SDA_PU_EN bit of register IF_PU_CTRL (0Ch) to 1.*
4. *The internal pull-down of the INT2 pin can be disconnected by setting the PD_DIS_INT2 bit in MD2_CFG (20h) to 1.*
5. *The internal pull-down of the INT1 pin can be disconnected by setting the PD_DIS_INT1 bit in IF_CTRL (0Eh) to 1.*

2 Registers

Register name	Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IF_PU_CTRL	0Ch	SDO_PU_DIS	SDA_PU_EN	0(1)	0(1)	0(1)	CS_PU_DIS	0(1)	0(1)
IF_CTRL	0Eh	0(1)	0(1)	0(1)	0(1)	0(1)	PD_DIS_INT1	I3C_DISABLE	I2C_DISABLE
CTRL1	10h	PP_OD	SIM	SW_RESET	IF_ADD_INC	DRDY_PULSED	WU_X_EN	WU_Y_EN	WU_Z_EN
CTRL2	11h	INT1_BOOT	INT1_F_FULL	INT1_F_FTH	INT1_F_OVR	INT1_DRDY	0(1)	0(1)	0(1)
CTRL3	12h	INT2_BOOT	INT2_F_FULL	INT2_F_FTH	INT2_F_OVR	INT2_DRDY	0(1)	ST1	ST0
CTRL4	13h	INACT_ODR1	INACT_ODR0	BDU	0(1)	0(1)	0(1)	SOC	BOOT
CTRL5	14h	ODR3	ODR2	ODR1	ODR0	BW1	BW0	FS1	FS0
FIFO_CTRL	15h	ROUNDING_XYZ	FIFO_DEPTH	0(1)	0(1)	STOP_ON_FTH	FIFO_MODE2	FIFO_MODE1	FIFO_MODE0
FIFO_WTM	16h	-	FTH6	FTH5	FTH4	FTH3	FTH2	FTH1	FTH0
INTERRUPT_CFG	17h	-	INT_SHORT_EN	WAKE_THS_W	-	SLEEP_STATUS_ON_INT	H_LACTIVE	LIR	INTERRUPTS_ENABLE
TAP_THS_X	18h	D4D_EN	D6D_THS1	D6D_THS0	TAP_THS_X_4	TAP_THS_X_3	TAP_THS_X_2	TAP_THS_X_1	TAP_THS_X_0
TAP_THS_Y	19h	TAP_PRIORITY_2	TAP_PRIORITY_1	TAP_PRIORITY_0	TAP_THS_Y_4	TAP_THS_Y_3	TAP_THS_Y_2	TAP_THS_Y_1	TAP_THS_Y_0
TAP_THS_Z	1Ah	TAP_X_EN	TAP_Y_EN	TAP_Z_EN	TAP_THS_Z_4	TAP_THS_Z_3	TAP_THS_Z_2	TAP_THS_Z_1	TAP_THS_Z_0
INT_DUR	1Bh	LATENCY3	LATENCY2	LATENCY1	LATENCY0	QUIET1	QUIET0	SHOCK1	SHOCK0
WAKE_UP_THS	1Ch	SINGLE_DOUBLE_TAP	SLEEP_ON	WK_THS5	WK_THS4	WK_THS3	WK_THS2	WK_THS1	WK_THS0
WAKE_UP_DUR	1Dh	FF_DUR5	WAKE_DUR1	WAKE_DUR0	0(1)	SLEEP_DUR3	SLEEP_DUR2	SLEEP_DUR1	SLEEP_DUR0
FREE_FALL	1Eh	FF_DUR4	FF_DUR3	FF_DUR2	FF_DUR1	FF_DUR0	FF_THS2	FF_THS1	FF_THS0
MD1_CFG	1Fh	INT1_SLEEP_CHANGE	INT1_SINGLE_TAP	INT1_WU	INT1_FF	INT1_DOUBLE_TAP	INT1_6D	WU_DUR_X4	-
MD2_CFG	20h	INT2_SLEEP_CHANGE	INT2_SINGLE_TAP	INT2_WU	INT2_FF	INT2_DOUBLE_TAP	INT2_6D	PD_DIS_INT2	-
WAKE_UP_SRC(2)	21h	-	SLEEP_CHANGE_IA	FF_IA	SLEEP_STATE	WU_IA	X_WU	Y_WU	Z_WU
TAP_SRC(2)	22h	-	TAP_IA	SINGLE_TAP_IA	DOUBLE_TAP_IA	TAP_SIGN	X_TAP	Y_TAP	Z_TAP
SIXD_SRC(2)	23h	-	D6D_IA	ZH	ZL	YH	YL	XH	XL
ALL_INT_SRC(2)	24h	-	INT_GLOBAL	SLEEP_CHANGE_IA_ALL	D6D_IA_ALL	DOUBLE_TAP_ALL	SINGLE_TAP_ALL	WU_IA_ALL	FF_IA_ALL
STATUS(2)	25h	-	-	-	-	-	-	PD_STATUS	DRDY
FIFO_STATUS1(2)	26h	FTH	FIFO_OVR	-	-	-	-	-	-
FIFO_STATUS2(2)	27h	FSS7	FSS6	FSS5	FSS4	FSS3	FSS2	FSS1	FSS0
OUT_X_L(2)	28h	OUTX7	OUTX6	OUTX5	OUTX4	OUTX3	OUTX2	OUTX1	OUTX0
OUT_X_H(2)	29h	OUTX15	OUTX14	OUTX13	OUTX12	OUTX11	OUTX10	OUTX9	OUTX8
OUT_Y_L(2)	2Ah	OUTY7	OUTY6	OUTY5	OUTY4	OUTY3	OUTY2	OUTY1	OUTY0

Register name	Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
OUT_Y_H[2]	2Bh	OUTY15	OUTY14	OUTY13	OUTY12	OUTY11	OUTY10	OUTY9	OUTY8
OUT_Z_L[2]	2Ch	OUTZ7	OUTZ6	OUTZ5	OUTZ4	OUTZ3	OUTZ2	OUTZ1	OUTZ0
OUT_Z_H[2]	2Dh	OUTZ15	OUTZ14	OUTZ13	OUTZ12	OUTZ11	OUTZ10	OUTZ9	OUTZ8
OUT_T_L[2]	2Eh	OUTT7	OUTT6	OUTT5	OUTT4	OUTT3	OUTT2	OUTT1	OUTT0
OUT_T_H[2]	2Fh	OUTT15	OUTT14	OUTT13	OUTT12	OUTT11	OUTT10	OUTT9	OUTT8
TEMP_OUT_L[2]	30h	OUTT7	OUTT6	OUTT5	OUTT4	OUTT3	OUTT2	OUTT1	OUTT0
TEMP_OUT_H[2]	31h	OUTT15	OUTT14	OUTT13	OUTT12	OUTT11	OUTT10	OUTT9	OUTT8
WHO_AM_I[2]	43h	0	1	0	0	0	1	0	1
ST_SIGN	58h	STSIGN2	STSIGN1	STSIGN0	Reserved[3]	Reserved[3]	Reserved[3]	Reserved[3]	Reserved[3]

1. *This bit must be set to 0 for the correct operation of the device.*
2. *Read-only register*
3. *Bits marked as Reserved must not be changed.*

3 Operating modes

The MIS2DU12 provides three possible operating configurations:

- Power-down: almost all the internal blocks of the device are switched off to minimize power consumption.
- Continuous conversion: the device continuously samples the signal at the data rate selected through the ODR[3:0] bits in the CTRL5 register.
- One-shot: the device waits for a trigger signal in order to generate new data.

All the operating modes of the MIS2DU12 can be selected through the ODR[3:0] bits in the CTRL5 register, as shown in the following table.

Table 2. Accelerometer operating modes

ODR[3:0]	Operating mode
0000	Power-down
0001	1.6 Hz (ultralow-power mode)
0010	3 Hz (ultralow-power mode)
0100	6 Hz (ultralow-power mode)
0101	6 Hz (normal mode)
0110	12.5 Hz (normal mode)
0111	50 Hz (normal mode)
1000	100 Hz (normal mode)
1001	200 Hz (normal mode)
1010	400 Hz (normal mode)
1011	800 Hz (normal mode)
1110	One-shot using the INT2 pin
1111	One-shot using the I²C/SPI/MIPI I3C® serial interface (SOC bit in the CTRL4 register)

The device offers a wide Vdd voltage range from 1.62 V to 3.6 V and a Vdd_IO range from 1.62 V to Vdd + 0.1 V.

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 as floating or connected to ground, until Vdd_IO is set. After Vdd_IO is set, the lines connected to the IO pins have to be configured according to their default status described in Table 1. Internal pin status. In order to avoid an unexpected increase in supply current, the input pins that are not pulled-up/pulled-down must be polarized by the host.

Note: Vdd cannot be lower than Vdd_IO. Vdd = 0 V and Vdd_IO "on" is allowed.

When the Vdd power supply is applied, the device performs a 20 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 device is in power-down mode, the digital interfaces (I²C, MIPI I3C®, and SPI) are still 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.

When the device is in continuous conversion mode, two power modes can be selected: normal or ultralow-power. In normal mode the data rates available are from 6 Hz to 800 Hz and the antialiasing filter is active, while in ultralow-power mode the data rates available are 1.6 Hz, 3 Hz and 6 Hz and the antialiasing filter is disabled in order to minimize power consumption.

When the device is in one-shot mode, the trigger event for new data generation can be associated with a "hardware" or "software" event. The sample rate can be customized from "one sample when needed" up to 40 Hz and the antialiasing filter is disabled in order to minimize power consumption.

It is recommended to set the ODR[3:0] bits in register CTRL5 in power-down (0000) before switching between different operating modes (from continuous to one-shot or vice versa).

Note: When the device is configured in power-down mode (ODR[3:0] = 0000), two consecutive write operations in the FIFO_CTRL register must be separated by at least 1.6 ms.

When passing from power-down mode (ODR[3:0] equal to 0000) to a continuous operating mode (ODR[3:0] different than 0000), it is necessary to wait for the next data-ready event before writing to the CTRL5 and/or FIFO_CTRL registers.

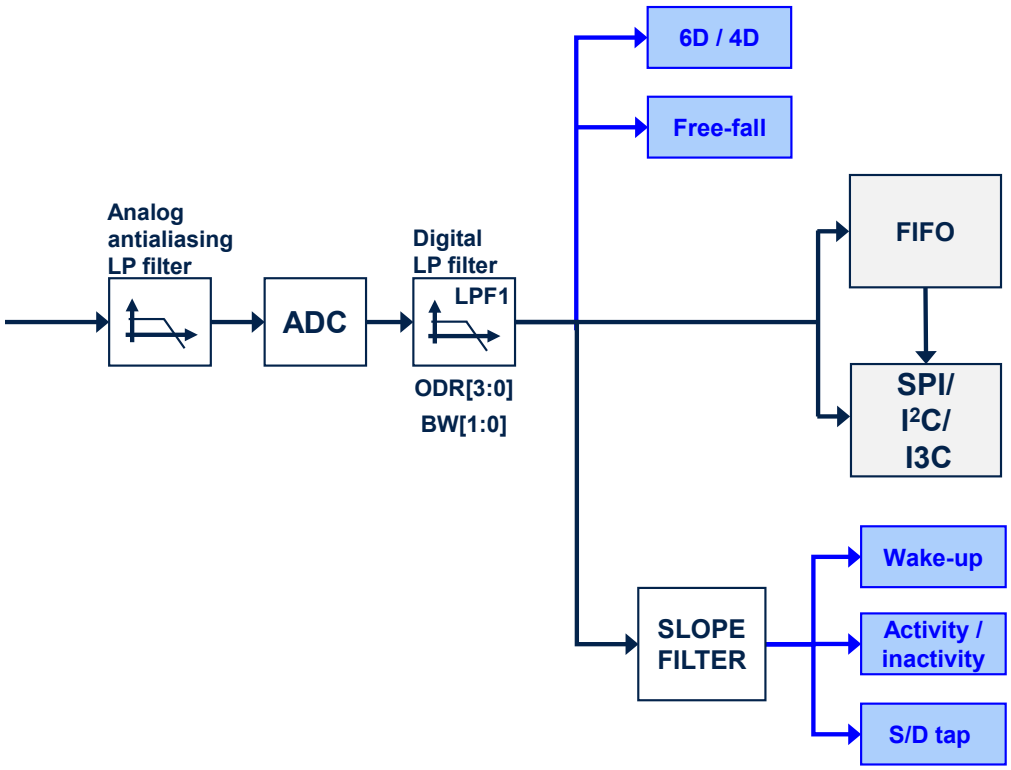
In all other cases, two consecutive write operations in the CTRL5 and/or FIFO_CTRL registers must be separated by a wait time of at least one ODR period +10% to ensure synchronization of the internal blocks in the device. ODR refers to the output data rate value that was configured before the register write operation.

3.1 Accelerometer filtering chain

The accelerometer sampling chain is represented by a cascade of three main blocks: an analog antialiasing low-pass filter, an ADC converter, and a digital low-pass filter (LPF1).

Figure 2. Accelerometer filtering chain shows the accelerometer sampling chain. The analog signal coming from the mechanical parts is filtered by an analog antialiasing low-pass filter before being converted by the ADC.

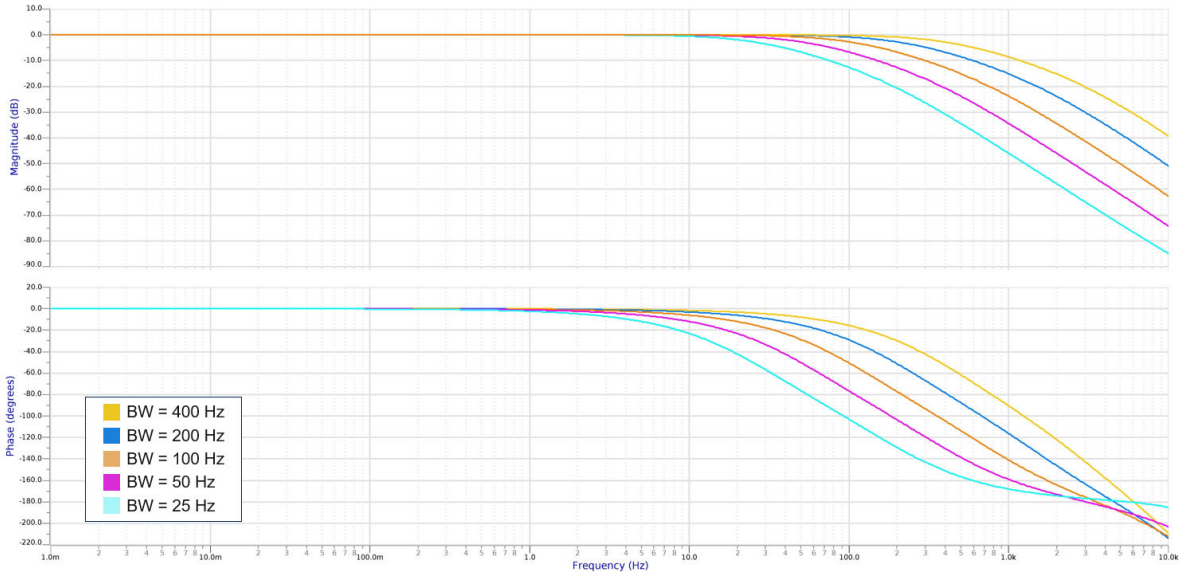
Figure 2. Accelerometer filtering chain



The analog antialiasing filter is enabled in continuous conversion normal mode only. It is disabled in continuous conversion ultralow-power mode and in one-shot mode.

The frequency response (referring to design simulation) of the analog antialiasing filter is shown in Figure 3. The frequency values from 25 Hz to 400 Hz for each curve below refer to the filtering chain bandwidth values (see Table 3 for the complete list of the available cutoff frequencies). For bandwidth values lower than 25 Hz (that is, 12.5 Hz, 6 Hz, 3 Hz), the frequency response is the same as that of 25 Hz.

Figure 3. Antialiasing filter frequency response



The digital LPF1 filter provides different cutoff values based on the accelerometer mode selected, as described in Section 3.2: Continuous conversion and Section 3.3: One-shot.

3.1.1 Accelerometer slope filter

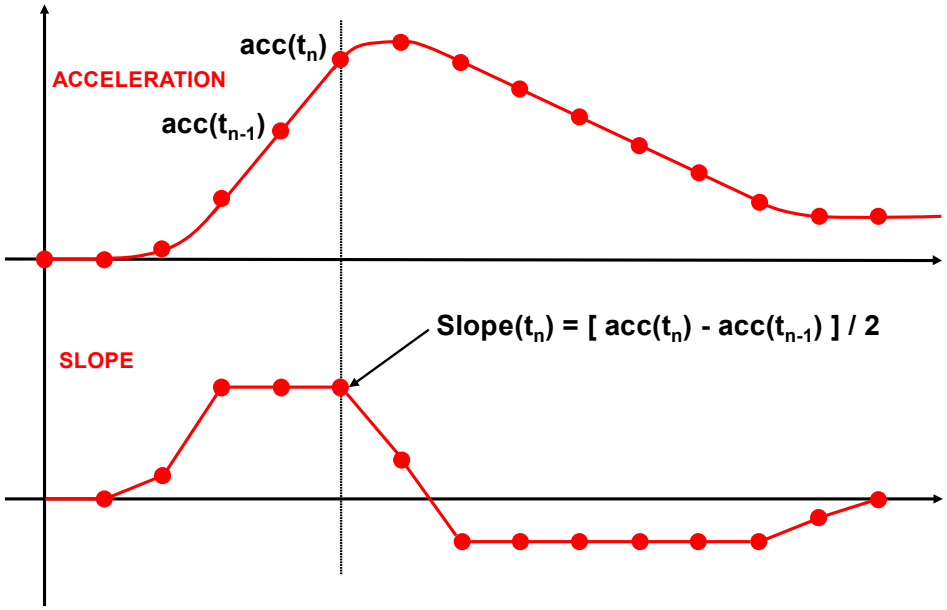
As shown in Figure 4. Accelerometer slope filter, the device embeds a digital slope filter, which can also be used for some embedded features such as single/double-tap recognition, wake-up detection and activity/inactivity.

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 4. Accelerometer slope filter



Note: The first output data generated by the slope filter is not valid and must be discarded.

3.2 Continuous conversion

When one of the continuous conversion data rates is selected through the ODR[3:0] bits in CTRL5, the device continuously samples the signal at the selected data rate.

3.2.1 Normal mode

When continuous-normal mode is set, the filtering chain of MIS2DU12, composed of an antialiasing and a low-pass filter, is enabled. The total bandwidth of the sensor can be configured through the BW[1:0] bits in CTRL5.

Table 3. Filtering chain bandwidth in continuous conversion - normal mode

ODR [Hz]	Cutoff frequency [Hz]	ODR[3:0]	BW[1:0]	Settling time [samples to be discarded]
6	3	0100	11	11
12.5	6	0101	10	5
12.5	3	0101	11	11
25	12.5	0110	01	1
25	6	0110	10	5
25	3	0110	11	11
50	25	0111	00	1
50	12.5	0111	01	2
50	6	0111	10	6
50	3	0111	11	14
100	ODR/2	1000	00	1
100	ODR/4	1000	01	2
100	ODR/8	1000	10	6
100	ODR/16	1000	11	14
200	ODR/2	1001	00	0
200	ODR/4	1001	01	2
200	ODR/8	1001	10	4
200	ODR/16	1001	11	14
400	ODR/2	1010	00	0
400	ODR/4	1010	01	0
400	ODR/8	1010	10	4
400	ODR/16	1010	11	8
800	ODR/2	1011	00	0
800	ODR/4	1011	01	0
800	ODR/8	1011	10	0
800	ODR/16	1011	11	1

In continuous-normal mode, the entire configurable filtering chain is active, but the overall supply current remains very low.

Table 4. Typical RMS noise and supply current in continuous conversion - normal mode [Vdd = 1.8 V, FS = ±8 *g*, BW_{-3db} = ODR/2]

ODR [Hz]	Noise [mg_{rms}]	Supply current [µA]
6	2.6	3.4
12.5	2.9	3.4
25	3.4	3.4
50	4.6	3.4
100	4.7	3.6
200	4.9	4.1
400	5.3	5.0
800	5.8	6.1

3.2.2 Ultralow-power mode

If the application requires extraordinarily low power consumption and a low sampling rate, continuous ultralow-power mode is appropriate. When continuous ultralow-power mode is active, the entire configurable filtering chain is disabled (antialiasing filter also disabled), in order to minimize power consumption, and the cutoff frequency of the system is 400 Hz.

Table 5. Typical RMS noise and supply current in continuous conversion - ultralow-power mode [Vdd = 1.8 V, FS = ±8 *g*, BW = ODR/2]

ODR [Hz]	Noise [mg_{rms}]	Supply current [µA]	ODR[3:0]	BW[1:0]
1.6	5.8	0.47	0001	XX
3	5.8	0.55	0010	XX
6	5.8	0.71	0011	XX

3.3 One-shot

If the application requires extraordinarily low power consumption but a custom data rate up to 40 Hz is needed, one-shot mode can be used. When one-shot mode is selected, the entire configurable filtering chain is disabled (antialiasing filter also disabled) in order to minimize power consumption, and the total bandwidth of the system is 400 Hz.

Figure 5. Single data conversion timing

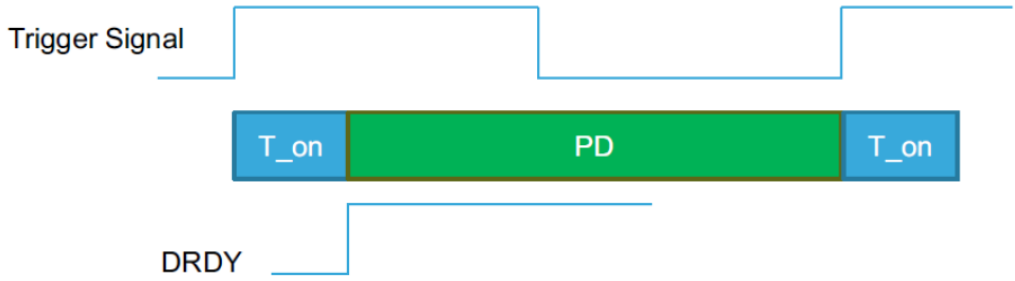


Table 6. Typical turn-on time in one-shot mode using INT2 pin or interface

ODR[3:0]	Operating mode	T_on (typ.)	Max data rate [Hz]
1110	One-shot using the INT2 pin	18 ms	40
1111	One-shot using the I²C/SPI/MIPI I3C® serial interface (SOC bit in the CTRL4 register)	19 ms	40

One-shot mode can use two sampling trigger modes:

- Software: the user starts the acquisition through the I²C / SPI / I3C interface, writing 1 to the SOC bit in the CTRL4 register. The SOC bit in the CTRL4 register is automatically reset when the measurement is finished in order to allow sending a new trigger for the next measurement.
- Hardware: the rising edge signal on the INT2 pin starts a new measurement. The external trigger must be reset to 0 by the user to allow sending a new trigger for the next measurement. The minimum duration for the trigger signal is 100 µs (typ).

New data are stored in output registers (28h - 2Fh) when ready and it is possible to monitor the end of the current measurement, reading the PD_STATUS bit in the STATUS register.

4 Reading output data

4.1 Startup sequence

Once the device is powered up, it automatically downloads the calibration coefficients from the embedded non-volatile memory to the internal registers. When the boot procedure is completed, that is, after approximately 20 milliseconds, the accelerometer automatically enters power-down. The default status of the pins with both Vdd and Vdd_IO "on" is indicated in Table 1. Internal pin status.

To turn on the accelerometer and gather acceleration data, it is necessary to select one of the operating modes through the ODR[3:0] bits in the CTRL5 register.

Refer to Section 3: Operating modes for a detailed description of data generation.

4.2 Using the status register

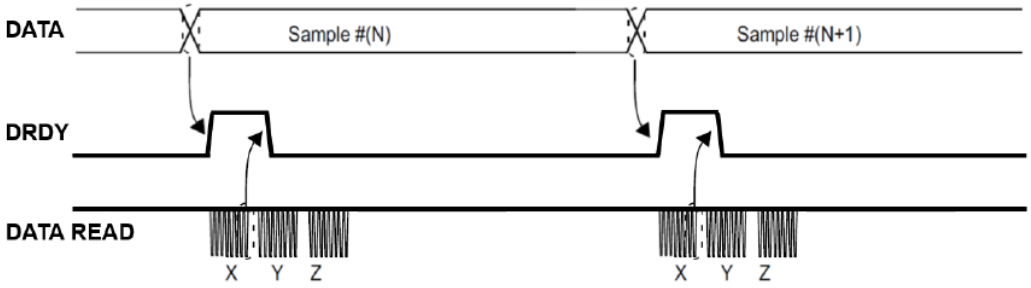
The device is provided with a STATUS (25h) register which can be polled to check when a new set of data is available. The DRDY bit is set to 1 when a new set of data is available from the accelerometer output. The read operations should be performed as follows:

1. Read STATUS.
2. If DRDY = 0, then go to 1.
3. Read OUT_X_L.
4. Read OUT_X_H.
5. Read OUT_Y_L.
6. Read OUT_Y_H.
7. Read OUT_Z_L.
8. Read OUT_Z_H.
9. Data processing
10. Go to 1.

4.3 Using the data-ready signal

The device can be configured to have a hardware signal to determine when a new set of measurement data is available to be read. The data-ready signal is controlled by the DRDY bit of the STATUS register. The signal can be driven to the INT1 pin by setting the INT1_DRDY bit of the CTRL2 register to 1 and to the INT2 pin by setting the INT2_DRDY bit of the CTRL3 register to 1. The data-ready signal rises to 1 when a new set of data has been measured and is available to be read. In DRDY latched mode (bit DRDY_PULSED = 0 in the CTRL1 register), which is the default condition, the signal gets reset when the higher byte of the axis has been read (registers 29h, 2Bh, 2Dh). In DRDY pulsed mode (DRDY_PULSED = 1) the pulse duration is about 90 µs. Pulsed mode is not applied to the DRDY bit which is always latched.

Figure 6. Data-ready signal



4.4 Using the block data update (BDU) feature

If reading the accelerometer data is not synchronized with either the DRDY event bit in the STATUS 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 CTRL4 register.

This feature avoids reading values (most significant and least significant bytes of the output data) related to different samples. In particular, when the BDU is activated, the data registers related to each axis always contain the most recent output data produced by the device; in case the read of a given pair (that is, OUT_X_H and OUT_X_L, OUT_Y_H and OUT_Y_L, OUT_Z_H and OUT_Z_L) is initiated, the refresh for that pair is blocked until both the MSB and LSB of the data are read.

Note: BDU only guarantees that the LSB and MSB bytes of one data axis 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.

4.5 Understanding output data

The measured acceleration data are sent to the OUT_X_H, OUT_X_L, OUT_Y_H, OUT_Y_L, OUT_Z_H, and OUT_Z_L 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 axes is given by the concatenation OUT_X_H & OUT_X_L, OUT_Y_H & OUT_Y_L , OUT_Z_H & OUT_Z_L.

Acceleration data is represented as 12-bit numbers, left-aligned and encoded in two's complement.

After calculating the LSB, it must be multiplied by the proper sensitivity parameter to obtain the corresponding value in m*g*.

Table 7. Sensitivity

Full scale	Sensitivity [m*g*/LSB]
±2 *g*	0.976
±4 *g*	1.952
±8 *g*	3.904
±16 *g*	7.808

4.5.1 Example of output data

Below is a simple example of how to use the LSB data and transform it into m*g*. The values are given under the hypothesis of ideal device calibration (that is, no offset, no gain error, and so forth). Get raw data from the sensor (normal mode, ±2 *g*):

```
OUT_X_L: 60h
OUT_X_H: FDh
OUT_Y_L: 70h
OUT_Y_H: 00h
OUT_Z_L: F0h
OUT_Z_H: 42h
```

Do register concatenation:

```
(OUT_X_H << 8) + OUT_X_L = FD60h [-672 in two's complement]
(OUT_Y_H << 8) + OUT_Y_L = 0070h [+112 in two's complement]
(OUT_Z_H << 8) + OUT_Z_L = 42F0h [+17136 in two's complement]
```

Apply sensitivity (for example, 0.976 at full scale ±2 *g*):

```
X: -672 / 16 * 0.976 = -41 mg
Y: +112 / 16 * 0.976 = +7 mg
Z: +17136 / 16 * 0.976 = +1045 mg
```

5 Interrupt generation and embedded functions

In order to generate an interrupt, the MIS2DU12 device has to be set in an active operating mode (not in power-down) because generation of the interrupt is based on accelerometer data.

The interrupt generator can be configured to detect:

- Free-fall
- Wake-up
- 6D/4D orientation detection
- Single-tap and double-tap sensing
- Activity/inactivity detection

Note: For the embedded functions to work correctly, it is important to configure the functions when the device is in power-down mode.

In order to enable the interrupt generation of the desired embedded function, route the function to the interrupt pins (INT1 and INT2) through the MD1_CFG and MD2_CFG registers and set the INTERRUPTS_ENABLE bit to 1 in the INTERRUPT_CFG register.

Note: The configuration of the INTERRUPTS_ENABLE bit, setting it either to 0 or to 1 in the INTERRUPT_CFG register, must be executed with the device in power-down mode.

The H_LACTIVE bit of the INTERRUPT_CFG 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 the interrupt condition is reached.

The PP_OD bit of CTRL1 allows changing the behavior of the interrupt pins also when the DRDY signal is routed to them from push-pull to open drain. If the PP_OD bit is set to 0 (default value), 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.

It is possible to configure the interrupt level / latched mode, and consequently the duration of the interrupt signal, using the LIR and INT_SHORT_EN bits in the INTERRUPT_CFG register, as shown in the following table.

Table 8. Interrupt mode configurations

LIR	INT_SHORT_EN	Interrupt mode
0	0	Interrupt level mode
1	1	Interrupt latched mode

- **Interrupt level mode**: the interrupt signal is automatically reset when the interrupt condition is no longer verified.
- **Interrupt latched mode:** the interrupt signal is the OR between the interrupt flags to be monitored on the INT1/INT2 pins. The interrupt signal goes high when the interrupt event occurs and is reset when the ALL_INT_SRC register is read. The status flags inside the interrupt source registers are coherent with the behavior of the INT1/INT2 pins. The interrupt generator block is inhibited for one ODR cycle after the interrupt event.

Note: The embedded functions interrupt is generated about 150 µs before the update of the output data registers and the generation of the DRDY signal. For this reason, if the user reads the output data registers synchronously with the occurrence of the embedded functions interrupt, the data that is read may have not been updated yet. To avoid this misalignment, the user should wait 150 µs from the generation of the embedded functions interrupt or wait for the next DRDY signal before reading the output data registers.

When latched mode is enabled, the user should not continuously poll the ALL_INT_SRC or the dedicated source registers, because by reading them, the embedded functions are internally reset. When latched mode is enabled, the user must wait 150 µs from the generation of the embedded functions interrupt or wait for the next DRDY signal before reading the ALL_INT_SRC or the dedicated source registers.

The LIR and INT_SHORT_EN bits in the INTERRUPT_CFG register do not impact the behavior of the DRDY signal. The data-ready signal can be configured to pulsed mode or latched mode based on the value of the DRDY_PULSED bit in the CTRL1 register. When latched mode is enabled, all interrupt signals are latched even if they are not routed to the INT1/INT2 interrupt pins.

5.1 Wake-up interrupt

The wake-up interrupt signal is generated if a certain number of consecutive accelerometer data exceed the configured threshold (Figure 7. Wake-up event recognition). The wake-up feature processes data from the internal slope filter.

Wake-up recognition can be enabled separately for each axis by using the WU_X_EN, WU_Y_EN, and WU_Z_EN bits in the CTRL1 register: for the generation of a wake-up interrupt, the modulus of the accelerometer data of at least one of the enabled axes must be greater than the threshold parameter for a number of samples at least equal to the duration parameter.

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 INTERRUPT_CFG register:

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

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 a duration of 1/ODR (also known as ODR_time), where ODR is the accelerometer output data rate.

By setting the WU_DUR_X4 bit in the MD1_CFG register, the following durations are selectable, instead, by using the WAKE_DUR[1:0] bits of the WAKE_UP_DUR register:

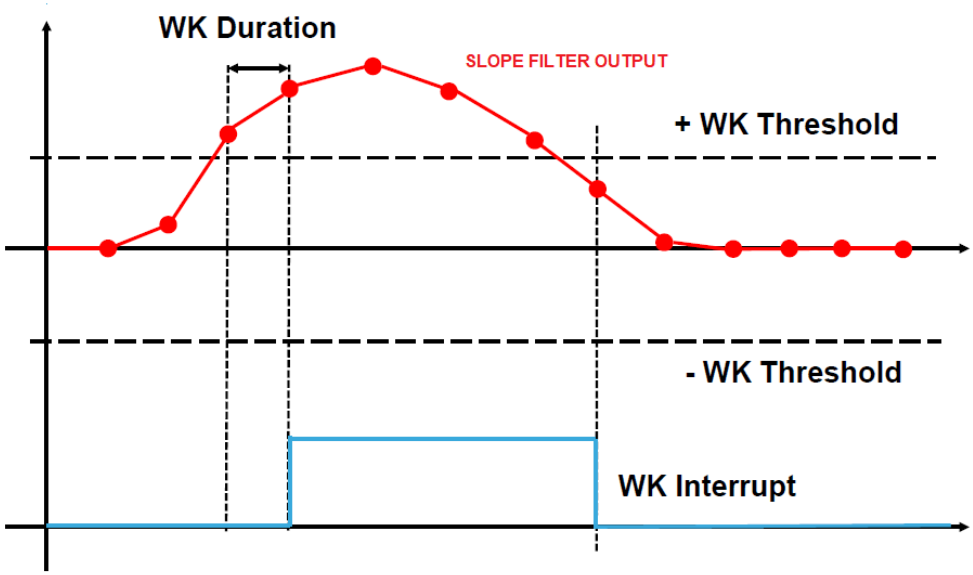
- 00: 3 ODR_time
- 01: 7 ODR_time
- 10: 11 ODR_time
- 11: 15 ODR_time

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 driven to the INT1 (or INT2) pin by setting the INT1_WU (or INT2_WU) bit of the MD1_CFG (or MD2_CFG) register to 1. It can also be checked by reading the WU_IA bit of the WAKE_UP_SRC register or WU_IA_ALL of the ALL_INT_SRC register. The X_WU, Y_WU, Z_WU bits of the WAKE_UP_SRC register indicate which axis/axes has/have triggered the wake-up event.

If the interrupt level mode is enabled (see Table 8. Interrupt mode configurations), the interrupt signal is automatically reset when the filtered data falls below the threshold. If latched mode is enabled (see Table 8. Interrupt mode configurations), once a wake-up event has occurred and the interrupt pin is asserted, it must be reset by reading 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 X_WU, Y_WU, Z_WU bits have to be evaluated (in addition to the WU_IA bit), it is recommended to read the WAKE_UP_SRC register before reading the ALL_INT_SRC register.

Figure 7. **Wake-up event recognition**



The example code that implements the software routine for wake-up event recognition is given below.

1.	Write 00h in CTRL5	// Switch to power-down mode
2.	Write 17h in CTRL1	// Enable wake-up event detection on the X, Y, and Z axes
3.	Write 40h in WAKE_UP_DUR	// Set wake-up duration
4.	Write 02h in WAKE_UP_THS	// Set wake-up threshold
5.	Write 20h in MD1_CFG	// Wake-up interrupt driven to INT1 pin
6.	Write 43h in INTERRUPT_CFG	// Interrupt latched mode // Enable interrupts
7.	Write 90h in CTRL5	// ODR = 200 Hz, normal mode, FS ±2 *g*

Since the duration time is set to zero, the wake-up interrupt signal is generated for each X,Y,Z data from the internal slope filter exceeding the configured threshold. The WU_THS field of the WAKE_UP_THS register is set to 000010, therefore the wake-up threshold is 62.5 m*g* (= 2 * FS / 64).

Since the wake-up functionality is implemented using the slope filter, it is necessary to consider the settling time of this filter if the functionality is enabled. When using the slope filter, the wake-up functionality is based on the comparison of the threshold value with half of the difference between the acceleration data [x,y,z] of the current sample and the acceleration data of the previous one (refer to Section 3.1.1: Accelerometer slope filter). At the very first sample, the slope filter output [x_f,y_f,z_f] is calculated as half of the difference between the first sample (for example, [x,y,z]=[0,0,1*g*]) and an initialization sample with values [x,y,z]=[0,0,0], since a previous sample does not exist yet. For this reason, the first output value of the slope filter (for example, [x_f,y_f,z_f]=[0,0,500m*g*]) could be higher than the threshold value on one of the axes (for example, on the Z-axis), in which case a spurious interrupt event is generated. In order to avoid this spurious interrupt generation, multiple solutions are possible. Three alternative solutions are presented below:

a. Ignore the first generated wake-up interrupt signal.

b. Add a wait time higher than 1 ODR_time before driving the interrupt signal to the INT1/INT2 pin (the status of the interrupt pin must be reset and the eventual wake-up spurious events must be cleared before driving the interrupt signal).

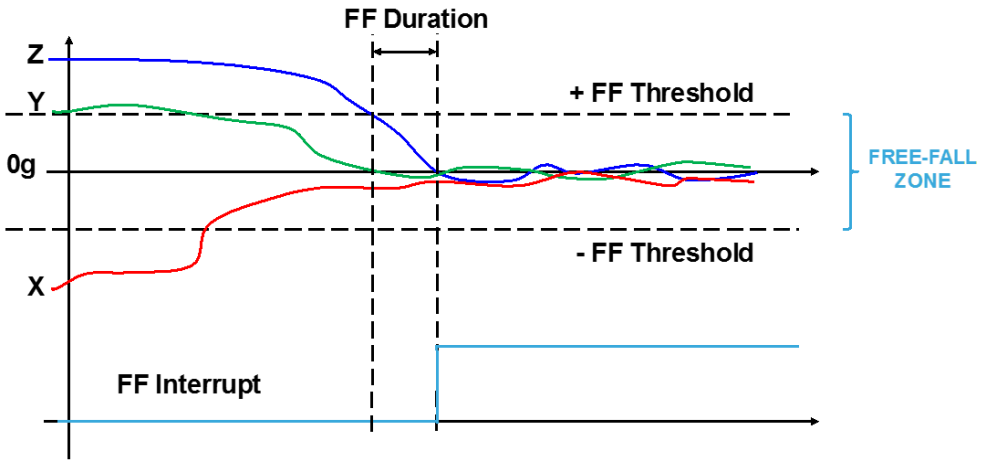
c. Set the duration parameter of the wake-up feature to at least 1 ODR_time; then, if needed, set it to 0 after 1 ODR_time.

The settling time in terms of samples to be discarded when changing the operating mode of the device must be considered, too, in order to avoid spurious interrupt generation. Refer to Section 3.2: Continuous conversion for the settling time at the various operating modes.

5.2 Free-fall interrupt

Free-fall detection refers to a specific register configuration that allows recognizing when the device is in free-fall: the acceleration measured along all the axes goes to zero. In a real case, a “free-fall zone" is defined around the zero-*g* level where all the accelerations are small enough to generate the interrupt. Configurable threshold and duration parameters are associated with free-fall event detection: the threshold parameter defines the free-fall zone amplitude; the duration parameter defines the minimum duration of the free-fall interrupt event to be recognized (Figure 8. Free-fall interrupt)

Figure 8. Free-fall interrupt



The free-fall event signal can be routed to the INT1(or INT2) pin by setting the INT1_FF (or INT2_FF) bit of the MD1_CFG (or MD2_CFG) register to 1; it can also be checked by reading the FF_IA bit of the WAKE_UP_SRC register or FF_IA_ALL of the ALL_INT_SRC register.

If the interrupt level mode is enabled (see Table 8), the interrupt signal is automatically reset when the free-fall condition is no longer verified. If latched mode is enabled (see Table 8), once a free-fall event has occurred and the interrupt pin is asserted, it must be reset by reading the FF_IA bit of the WAKE_UP_SRC register or the FF_IA_ALL bit of the ALL_INT_SRC register.

The free-fall detection parameters can be modified by configuring the FREE_FALL (contains bits FF_THS[2:0] and FF_DUR[4:0]) and WAKE_UP_DUR (contains MSB of duration parameter - FF_DUR5) registers. The threshold value can be set through the FF_THS[2:0] bits and is described in Table 9. Free-fall threshold values. The values given in this table are valid for any accelerometer full-scale configuration.

Table 9. Free-fall threshold values

FREE_FALL register - FF_THS[2:0]	Threshold value
000	~156 mg
001	~219 mg
010	~250 mg
011	~312 mg
100	~344 mg
101	~406 mg
110	~469 mg
111	~500 mg

A basic software routine for free-fall event recognition is given below.

1. Write 00h in CTRL5 // Switch to power-down mode
2. Write 00h in WAKE_UP_DUR // Set event duration (FF_DUR5 = 0)
3. Write 33h in FREE_FALL // Set FF threshold (FF_THS[2:0] = 011)
4. Write 10h in MD1_CFG // FF interrupt driven to INT1 pin
5. Write 43h in INTERRUPT_CFG // Interrupt latched mode
// Enable interrupts
6. Write 90h in CTRL5 // ODR = 200 Hz, normal mode, FS ±2 *g*

The sample code exploits a threshold set to ~312 m*g* for free-fall recognition and the event is notified by hardware through the INT1 pin. The FF_DUR[5:0] field of the FREE_FALL / WAKE_UP_DUR registers is configured to ignore events that are shorter than 6/ODR = 6/200 Hz = 30 ms in order to avoid false detections.

5.3 6D/4D orientation detection

The MIS2DU12 device provides the capability to detect the orientation of the device in space, enabling easy implementation of energy-saving procedures and automatic image rotation for mobile devices.

5.3.1 6D orientation detection

Six orientations of the device in space can be detected; the interrupt signal is asserted when the device switches from one orientation to another. The interrupt is not reasserted as long as the position is maintained. 6D interrupt is generated when only one axis exceeds a selected threshold and the acceleration values measured from the other two axes are lower than the threshold: the ZH, ZL, YH, YL, XH, XL bits of the SIXD_SRC register indicate which axis has triggered the 6D event.

In more detail:

Table 10. SIXD_SRC register

b7	b6	b5	b4	b3	b2	b1	b0
0	D6D_IA	ZH	ZL	YH	YL	XH	XL

The D6D_IA bit is set high when the device switches from one orientation to another.

- ZH (YH, XH) is set high when the face perpendicular to the Z (Y,X) axis is almost flat and the acceleration measured on the Z (Y,X) axis is positive and in the absolute value bigger than the threshold.
- ZL (YL, XL) is set high when the face perpendicular to the Z (Y,X) axis is almost flat and the acceleration measured on the Z (Y,X) axis is negative and in the absolute value bigger than the threshold.

The D6D_THS[1:0] bits of the TAP_THS_X register are used to select the threshold value used to detect the change in device orientation. The threshold values given in Table 11. Threshold for 4D/6D function are valid for each accelerometer full-scale value.

Table 11. Threshold for 4D/6D function

D6D_THS[1:0]	Threshold value [degrees]
00	80
01	70
10	60
11	50

This interrupt signal can be driven to the INT1 (or INT2) pin by setting the INT1_6D (or INT2_6D) bit of the MD1_CFG (or MD2_CFG) register to 1; it can also be checked by reading the D6D_IA bit of the SIXD_SRC register or the D6D_IA_ALL bit of the ALL_INT_SRC register.

If the interrupt level mode is enabled (see Table 8. Interrupt mode configurations), the interrupt signal is active only for a duration of 1/ODR then it is automatically deasserted (ODR is the accelerometer output data rate). If latched mode is enabled (see Table 8. Interrupt mode configurations), once an orientation change has occurred and the interrupt pin is asserted, a read of the ALL_INT_SRC register clears the request and the device is ready to recognize a different orientation. The XL, XH, YL, YH, ZL, ZH bits are not affected by the interrupt mode configuration: they correspond to the current state of the device when the SIXD_SRC register is read.

Referring to the six possible cases illustrated in Figure 9. 6D recognized orientations, the content of the SIXD_SRC register for each position is shown in Table 12. SIXD_SRC register for 6D positions.

Figure 9. 6D recognized orientations

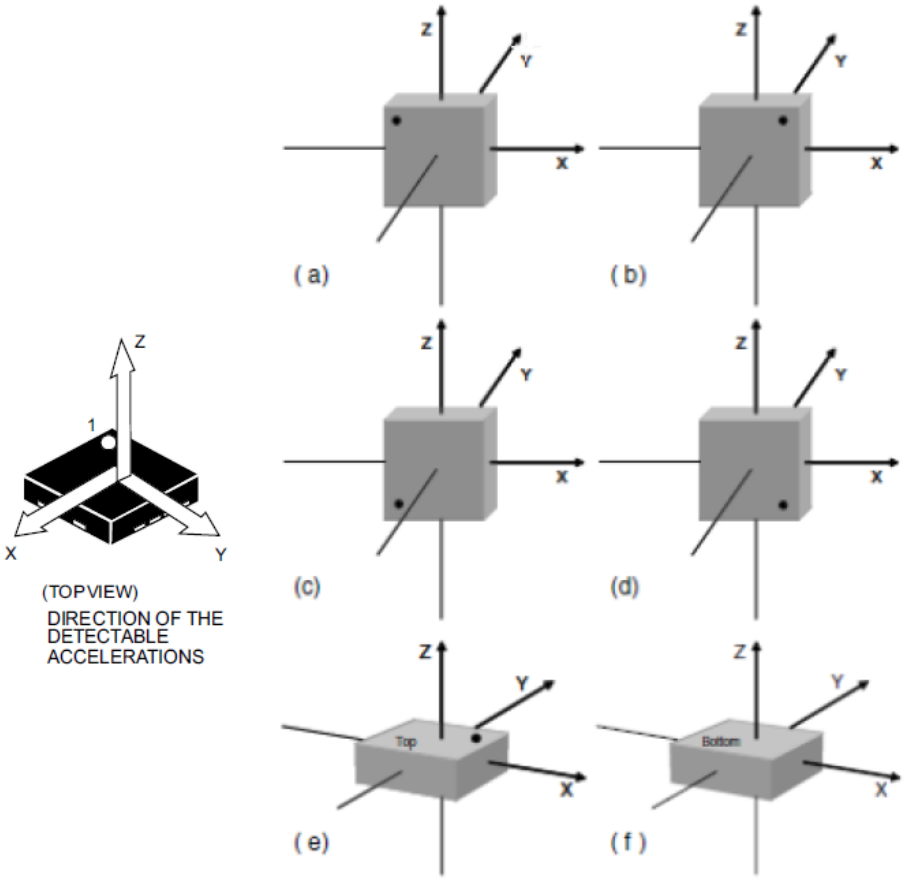


Table 12. SIXD_SRC register for 6D positions

Case	D6D_IA	ZH	ZL	YH	YL	XH	XL
(a)	1	0	0	0	0	0	1
(b)	1	0	0	0	1	0	0
(c)	1	0	0	1	0	0	0
(d)	1	0	0	0	0	1	0
(e)	1	1	0	0	0	0	0
(f)	1	0	1	0	0	0	0

The following example implements a software routine for 6D orientation detection:

1.	Write 00h in CTRL5	// Switch to power-down mode
2.	Write 40h in TAP_THS_X	// Set 6D threshold (D6D_THS[1:0] = 10 = 60 degrees)
3.	Write 04h in MD1_CFG	// 6D interrupt driven to INT1 pin
4.	Write 01h in INTERRUPT_CFG	// Interrupt level mode // Enable interrupts
5.	Write 90h in CTRL5	// ODR = 200 Hz, low-power mode, FS ±2 *g*

5.3.2 4D orientation detection

The 4D direction function is a subset of the 6D function especially defined to be implemented in mobile devices for portrait and landscape computation. It can be enabled by setting the D4D_EN bit of the TAP_THS_X register.

In this configuration, Z-axis position detection is disabled, therefore reducing position recognition to cases (a), (b), (c), and (d) of Table 12. SIXD_SRC register for 6D positions.

5.4 Single-tap and double-tap recognition

The single-tap and double-tap recognition functions featured in the MIS2DU12 help to create a man-machine interface with little software loading. The device can be configured to output an interrupt signal on a dedicated pin when tapped in any direction.

If the sensor is exposed to a single input stimulus, it generates an interrupt request on the interrupt pin INT1 / INT2. A more advanced feature allows the generation of an interrupt request when a double input stimulus with programmable time between the two events is recognized, enabling a mouse button-like function. In the MIS2DU12 device the single-tap and double-tap recognition functions use the slope filter to detect tap events.

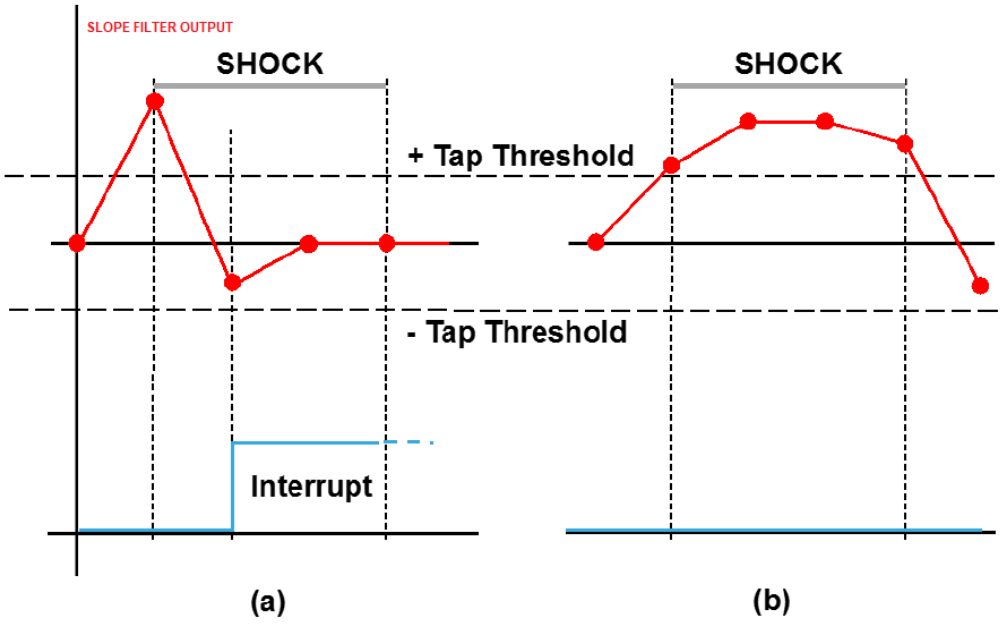
This function can be fully programmed by the user in terms of expected amplitude and timing of the internal slope filtered data by means of a dedicated set of registers.

The recommended accelerometer ODR for single and double-tap recognition is 400 Hz or higher.

5.4.1 Single-tap

If the device is configured for single-tap event detection, an interrupt is generated when the internal slope filtered data exceeds the programmed threshold and returns below it within the shock time window. If the interrupt level mode is enabled (see Table 8. Interrupt mode configurations), the interrupt is kept high for the duration of the quiet window. If latched mode is enabled (see Table 8. Interrupt mode configurations) on the single-tap interrupt signal, the interrupt is kept high until the ALL_INT_SRC register is read. The SINGLE_DOUBLE_TAP bit of WAKE_UP_THS has to be set to 0 in order to enable single-tap recognition only. In case (a) of Figure 10. Single-tap event recognition the single-tap event has been recognized, while in case (b) the tap has not been recognized because the signal falls below the threshold after the shock time window has expired.

Figure 10. Single-tap event recognition



5.4.2 Double tap

If the device is configured for double-tap event detection, an interrupt is generated when, after a first tap, a second tap is recognized. The recognition of the second tap occurs only if the event satisfies the rules defined by the shock, the latency and the quiet time windows.

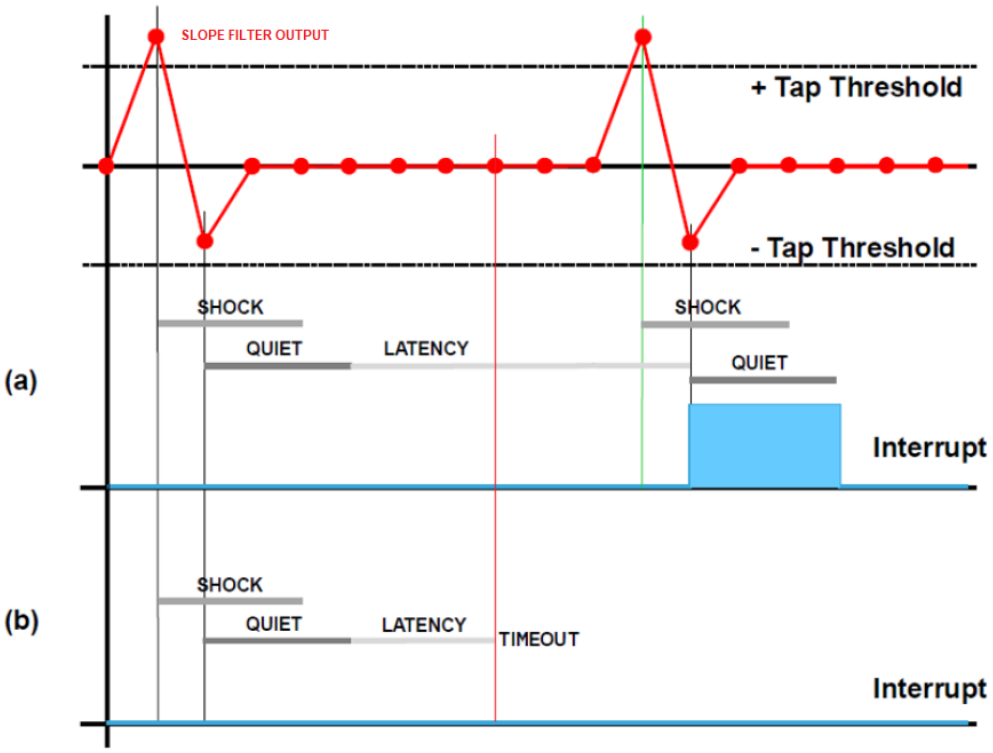
In particular, after the first tap has been recognized, the second tap detection procedure is delayed for an interval defined by the quiet time. This means that after the first tap has been recognized, the second tap detection procedure starts only if the internal slope filtered data exceeds the threshold after the quiet window but before the latency window has expired. In case (a) of Figure 11. Double-tap event recognition - interrupt level mode is enabled, a double-tap event has been correctly recognized, while in case (b) the interrupt has not been generated because the internal slope filtered data exceeds the threshold after the latency window interval has expired.

Once the second tap detection procedure is initiated, the second tap is recognized with the same rule as the first: the internal slope filtered data must return below the threshold before the shock window has expired.

It is important to appropriately define the quiet window to avoid unwanted taps due to spurious bouncing of the input signal.

In the double-tap case, if interrupt level is enabled (see Table 8. Interrupt mode configurations), the interrupt is kept high for the duration of the quiet window. If latched mode is enabled, the interrupt is kept high until the ALL_INT_SRC register is read.

Figure 11. Double-tap event recognition - interrupt level mode is enabled



5.4.3 Single-tap and double-tap recognition configuration

The MIS2DU12 device can be configured to output an interrupt signal when tapped (once or twice) in any direction: the TAP_X_EN, TAP_Y_EN and TAP_Z_EN bits of the TAP_THS_Z register must be set to 1 to enable the tap recognition on X, Y, Z directions, respectively.

The TAP_THS_X_[4:0], TAP_THS_Y_[4:0], TAP_THS_Z_[4:0] bits of the TAP_THS_X, TAP_THS_Y, TAP_THS_Z registers are used to select the unsigned threshold value used to detect the tap event. The value of 1 LSB of these 5 bits depends on the selected accelerometer full scale: 1 LSB = FS/32. The unsigned threshold is applied to both positive and negative internal slope filtered data.

The user can also define the "priority report" of the single / double-tap interrupt event through the TAP_PRIORITY_[2:0] bits in register TAP_THS_Y. This feature is useful in case of a contemporary tap event on more than one axis because only the tap event related to the axis having higher priority is given in the source register.

The shock time window defines the maximum duration of the overthreshold event: the acceleration must return below the threshold before the shock window has expired, otherwise the tap event is not detected. The SHOCK[1:0] bits of the INT_DUR register are used to set the shock time window value: the default value of these bits is 00 and corresponds to a duration of 4/ODR, where ODR is the accelerometer output data rate. If the SHOCK[1:0] bits are set to a different value, 1 LSB corresponds to a duration of 8/ODR.

In the double-tap case, the quiet time window defines the time after the first tap recognition in which there must not be any overthreshold. When the interrupt level mode is enabled (see Table 8. Interrupt mode configurations), the quiet time also defines the length of the interrupt pulse (for both single and double-tap). The QUIET[1:0] bits of the INT_DUR register are used to set the quiet time window value: the default value of these bits is 00 and corresponds to a duration of 2/ODR, where ODR is the accelerometer output data rate. If the QUIET[1:0] bits are set to a different value, 1 LSB corresponds to a duration of 4/ODR.

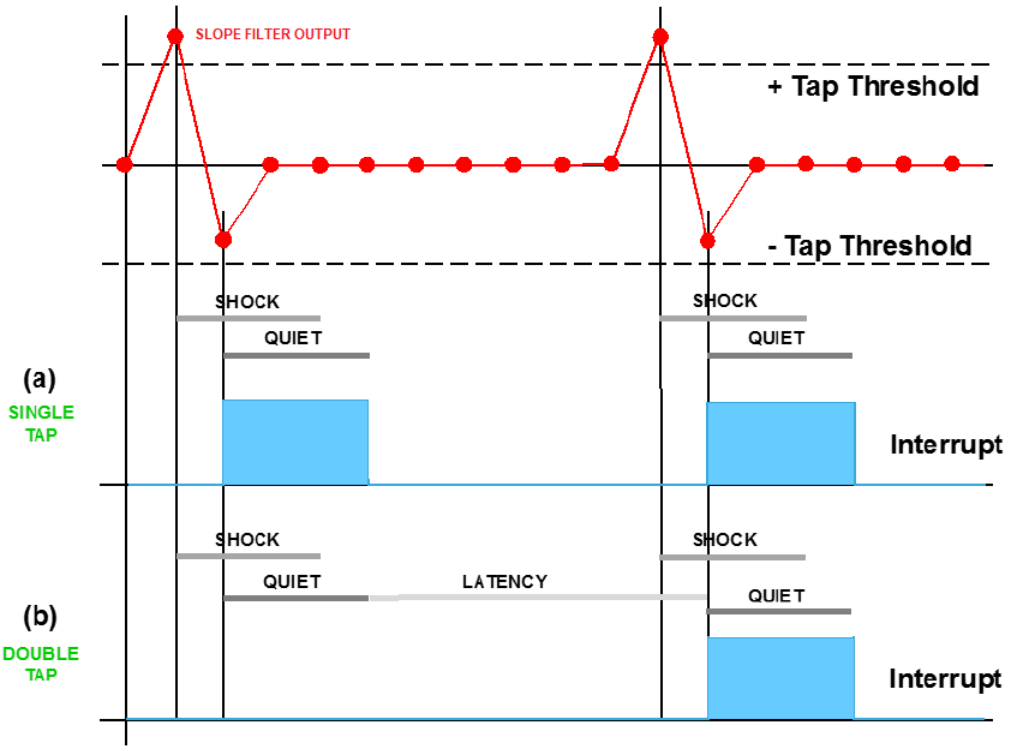
In the double-tap case, the latency time window defines the maximum time between two consecutive detected taps. The latency time period starts just after the completion of the quiet time of the first tap. The LATENCY[3:0] bits of the INT_DUR register are used to set the latency time window value: the default value of these bits is 0000 and corresponds to a duration of 16/ODR, where ODR is the accelerometer output data rate. If the LATENCY[3:0] bits are set to a different value, 1 LSB corresponds to a duration of 32/ODR.

Figure 12. Single and double-tap recognition - interrupt level mode is enabled illustrates a single-tap event (a) and a double-tap event (b). These interrupt signals can be driven to the INT1 (or INT2) pin by setting the INT1_SINGLE_TAP (or INT2_SINGLE_TAP) bit of the MD1_CFG (or MD2_CFG) register to 1 for the single-tap case, and setting the INT1_DOUBLE_TAP (or INT2_DOUBLE_TAP) bit of the MD1_CFG (or MD2_CFG) register to 1 for the double-tap case.

Configurable parameters for tap recognition functionality are the tap threshold and the shock, quiet and latency time windows. Valid ODRs are 400 Hz, 800 Hz.

No single/double-tap interrupt is generated if the accelerometer is in inactivity status (see Section 5.5: Activity/ inactivity recognition for more details).

Figure 12. **Single and double-tap recognition - interrupt level mode is enabled**



The tap interrupt signals can also be checked by reading the TAP_SRC register, described in Table 13.

Table 13. **TAP_SRC register**

b7	b6	b5	b4	b3	b2	b1	b0
0	TAP_IA	SINGLE_TAP _IA	DOUBLE_TAP _IA	TAP_SIGN	X_TAP	Y_TAP	Z_TAP

- TAP_IA is set high when a single-tap or double-tap event has been detected.
- SINGLE_TAP_IA is set high when a single tap has been detected.
- DOUBLE_TAP_IA is set high when a double tap has been detected.
- TAP_SIGN indicates the acceleration sign when the tap event is detected. It is set low in case of positive sign and it is set high in case of negative sign.
- X_TAP (Y_TAP, Z_TAP) is set high when the tap event has been detected on the X (Y, Z) axis.

Single and double-tap recognition work independently. Setting the SINGLE_DOUBLE_TAP bit of WAKE_UP_THS to 0, only single-tap recognition is enabled: double-tap recognition is disabled and cannot be detected. When the SINGLE_DOUBLE_TAP bit is set to 1, both single and double-tap recognition are enabled, and the single-tap event is always recognized first, followed by the double-tap event.

If the interrupt level mode is enabled, the interrupt is kept high for the duration of the quiet window.

If latched mode is enabled, the value assigned to SINGLE_DOUBLE_TAP also affects the behavior of the interrupt signal: when it is set to 0, latched mode is applied to the single-tap interrupt signal; when it is set to 1, latched mode is applied to the double-tap interrupt signal only. The latched interrupt signal is kept high until the ALL_INT_SRC register is read. The TAP_SIGN, X_TAP, Y_TAP, Z_TAP 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 TAP_SIGN, X_TAP, Y_TAP, Z_TAP bits have to be evaluated (in addition to the TAP_IA bit), it is recommended to read the TAP_SRC register before reading the ALL_INT_SRC register.

5.4.4 Single-tap example

The following example code implements a software routine for single-tap detection.

1.	Write 00h in CTRL5	Switch to power-down mode
2.	Write 09h in TAP_THS_X	// Set tap threshold for X-axis
3.	Write E9h in TAP_THS_Y	// Set tap threshold for Y-axis // Set TAP priority Z-Y-X
4.	Write E9h in TAP_THS_Z	// Enable tap detection on X, Y, Z-axis // Set tap threshold for Z-axis
5.	Write 06h in INT_DUR	// Set quiet and shock time windows
6.	Write 00h in WAKE_UP_THS	// Only single-tap enabled (SINGLE_DOUBLE_TAP = 0)
7.	Write 40h in MD1_CFG	// Single-tap interrupt driven to INT1 pin
8.	Write 43h in INTERRUPT_CFG	// Interrupt latched mode // Enable interrupts
9.	Write A0h in CTRL5	// ODR = 400 Hz, normal mode, FS ±2 *g*

In this example the threshold for each axis is set to 01001, therefore the tap threshold is 562.5 m*g* (= 9 * FS / 32).

The SHOCK field of the INT_DUR register is set to 10: an interrupt is generated when the internal slope filtered data exceeds the programmed threshold and returns below it within 40 ms (= 2 * 8 / ODR) corresponding to the shock time window.

The QUIET field of the INT_DUR register is set to 01: since latched mode is disabled, the interrupt is kept high for the duration of the quiet window, therefore 10 ms (= 1 * 4 / ODR).

5.4.5 Double-tap example

The example code that implements the software routine for double-tap detection is given below.

1.	Write 00h in CTRL5	Switch to power-down mode
2.	Write 0Ch in TAP_THS_X	// Set tap threshold for X-axis
3.	Write ECh in TAP_THS_Y	// Set tap threshold for Y-axis // Set TAP priority Z-Y-X
4.	Write ECh in TAP_THS_Z	// Enable tap detection on X, Y, Z-axis // Set tap threshold for Z-axis
5.	Write 7Fh in INT_DUR	// Set duration, quiet, and shock time windows
6.	Write 80h in WAKE_UP_THS	// Single and double-tap enabled // (SINGLE_DOUBLE_TAP = 1)
7.	Write 08h in MD1_CFG	// Single and double-tap interrupt driven to INT1 pin
8.	Write 43h in INTERRUPT_CFG	// Interrupt latched mode // Enable interrupts
9.	Write A0h in CTRL5	// ODR = 400 Hz, normal mode, FS ±2 *g*

In this example the threshold for each axis is set to 01100, therefore the tap threshold is 750 m*g* (= 12 * FS / 32).

For interrupt generation, during the first and the second tap the internal slope filtered data must return below the threshold before the shock window has expired. The SHOCK field of the INT_DUR register is set to 11, therefore the shock time is 60 ms (= 3 * 8 / ODR).

For interrupt generation, after the first tap recognition there must not be any internal slope filtered data overthreshold during the quiet time window. Furthermore, since latched mode is disabled, the interrupt is kept high for the duration of the quiet window. The QUIET field of the INT_DUR register is set to 11, therefore the quiet time is 30 ms (= 3 * 4 / ODR).

For the maximum time between two consecutive detected taps, the LAT field of the INT_DUR register is set to 0111, therefore the duration time is 560 ms (= 7 * 32 / ODR).

5.5 Activity/inactivity recognition

The activity/inactivity recognition function allows reducing system power consumption and developing new smart applications.

The activity/inactivity function is enabled by setting the SLEEP_ON bit of the WAKE_UP_THS register to 1. This bit must be set only when the device is in power-down mode. If the sleep state condition is detected, the MIS2DU12 automatically goes to the ultralow-power ODR selected by the INACT_ODR[1:0] bits in the CTRL4 register. The MIS2DU12 wakes up from the sleep state as soon as a wake-up event has been detected, switching to the operating mode and ODR configured in the CTRL5 register.

With this feature the system may be efficiently switched from low-power consumption to full performance and vice versa depending on user-selectable acceleration events, thus ensuring power saving and flexibility.

This function can be fully programmed by the user in terms of the expected amplitude and timing of the slope filtered data by means of a dedicated set of registers (Figure 13. Activity/inactivity recognition).

The unsigned threshold value is defined using the WK_THS[5:0] bits in 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 INTERRUPT_CFG 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 slope filtered data.

When a certain number of consecutive X,Y,Z slope filtered data is smaller than the configured threshold, the ODR[3:0] bits of the CTRL5 register are bypassed (inactivity) and the accelerometer is internally set to the ultralow-power ODR selected by the INACT_ODR[1:0] bits in CTRL4 although the content of CTRL5 is left untouched. 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 a duration of 512/ODR, where ODR is the accelerometer output data rate configured by the ODR[3:0] bits of the CTRL5 register. The default value is 0000 and corresponds to a duration of 16/ODR.

During the inactivity status of the device, the SLEEP_STATE bit in the WAKE_UP_SRC register is set high. Every time the device status changes from activity to inactivity or vice versa, the SLEEP_CHANGE_IA bit in WAKE_UP_SRC is set for about 1/ODR_ACT + 17 ms if the device status changes from activity to inactivity (for an ODR less than 50 Hz, 37 ms) and 17 ms vice versa. This bit can be routed to the INT1 (or INT2) pin alternatively:

- If the SLEEP_STATUS_ON_INT bit in INTERRUPT_CFG (17h) is set to 1, then the SLEEP_STATE bit in WAKE_UP_SRC is routed to the pin.
- If the SLEEP_STATUS_ON_INT bit in INTERRUPT_CFG (17h) is set to 0, then the SLEEP_CHANGE_IA bit in WAKE_UP_SRC is routed to the pin.

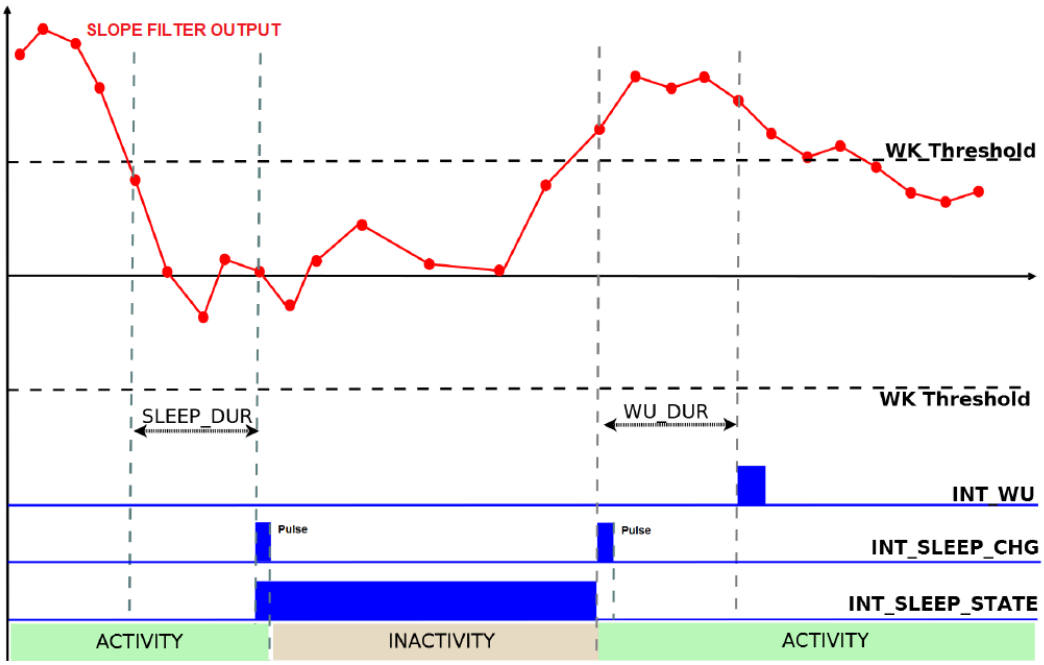
In any case, the INT1_SLEEP_CHANGE (or INT2_SLEEP_CHANGE) bit of the MD1_CFG (or MD2_CFG) register has to be set to 1. Note that these signals are not compatible with the interrupt mode described in Table 8. Interrupt mode configurations.

Note: When in inactivity state, in order to change the value of the CTRL5 register (and eventually to change the operating mode), it is necessary to disable the INTERRUPTS_ENABLE bit of the INTERRUPT_CFG register first. If the INTERRUPTS_ENABLE bit needs to remain enabled, it is necessary to follow this procedure to change the value of the CTRL5 register: disable the INTERRUPTS_ENABLE bit, switch to power-down mode, re-enable the INTERRUPTS_ENABLE bit, and finally change the value of the CTRL5 register as desired.

When a single sample of slope filtered data on one axis becomes bigger than the threshold, the CTRL5 register settings are immediately restored (activity). The wake-up interrupt event can be delayed in function of the value of the WU_DUR[1:0] bits of the WAKE_UP_DUR register: 1 LSB corresponds to a duration of 1/ODR, where ODR is the accelerometer output data rate. In order to generate the interrupt at the same time as the inactivity/activity event, WU_DUR[1:0] have to be set to 0.

When the wake-up event is detected, the interrupt is set high for a duration of 1/ODR, then it is automatically deasserted (the WU_IA event on the pin must be routed by setting the INT1_WU (or INT2_WU) bit of the MD1_CFG (or MD2_CFG) register to 1).

Figure 13. **Activity/inactivity recognition**



The following routine describes the implementation of activity/inactivity detection.

1. Write 00h in CTRL5 // Switch to power-down mode
2. Write C0h in CTRL4 // Select ODR during inactivity status
3. Write 17h in CTRL1 // Enable wake-up event detection on the X, Y, and Z axes
4. Write 42h in WAKE_UP_DUR // Set duration for inactivity detection
// Set duration for wake-up detection
5. Write 42h in WAKE_UP_THS // Set activity/inactivity threshold
// Enable activity/inactivity detection
6. Write 20h in MD1_CFG // Activity (wake-up) interrupt driven to INT1 pin
7. Write 01h in INTERRUPT_CFG // Enable interrupts
8. Write 90h in CTRL5 // ODR = 200 Hz, normal mode, FS ±2 *g*

In this example the WU_THS field of the WAKE_UP_THS register is set to 000010, therefore the activity/ inactivity threshold is 62.5 mg (= 2 * FS / 64).

Before inactivity detection, the X,Y,Z slope filtered 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 0010, corresponding to 5.12 s (= 2 * 512 / ODR). After this period of time has elapsed, the accelerometer ODR is internally set to 6 Hz in ultralow-power mode since the INACT_ODR field of the CTRL4 register is set to 11.

The activity status is detected and the CTRL1 register settings immediately restored if the slope filtered data of (at least) one axis is bigger than the threshold and the wake-up interrupt was notified after an interval defined by the WU_DUR field of the WAKE_UP_DUR register: this field is set to 10, corresponding to 10 ms (= 2 * 1 / ODR).

The following routine describes how to route the sleep change event to the INT1 pin.

1. Write 00h in CTRL5 // Switch to power-down mode
2. Write C0h in CTRL4 // Select ODR during inactivity status
3. Write 42h in WAKE_UP_DUR // Set duration for inactivity detection
// Set duration for wake-up detection
4. Write 42h in WAKE_UP_THS // Set activity/inactivity threshold
// Enable activity/inactivity detection
5. Write 80h in MD1_CFG // Sleep change interrupt driven to INT1 pin
6. Write 01h in INTERRUPT_CFG // Enable interrupts
7. Write 90h in CTRL5 // ODR = 200 Hz, normal mode, FS ±2 *g*

The following routine describes how to route the sleep status event to the INT1 pin.

1. Write 00h in CTRL5 // Switch to power-down mode
2. Write C0h in CTRL4 // Select ODR during inactivity status
3. Write 42h in WAKE_UP_DUR // Set duration for inactivity detection
// Set duration for wake-up detection
4. Write 42h in WAKE_UP_THS // Set activity/inactivity threshold
// Enable activity/inactivity detection
5. Write 80h in MD1_CFG // Sleep change interrupt driven to INT1 pin
6. Write 09h in INTERRUPT_CFG // Route sleep status instead of sleep change
// Enable interrupts
7. Write 90h in CTRL5 // ODR = 200 Hz, normal mode, FS ±2 *g*

The last two examples are similar to the first one except for the event routed to the INT1 pin.

5.5.1 Stationary/motion detection

Stationary/motion detection is a particular case of the activity/inactivity functionality described in Section 5.5 in which no ODR / power mode changes occur when a sleep condition (equivalent to a stationary condition) is detected. Stationary/motion detection is activated by setting the INACT_ODR[1:0] = 00 bits in the CTRL4 register.

5.6 Boot status

After the device is powered up, the MIS2DU12 performs a 20 ms boot procedure to load the trimming parameters. After the boot is completed, the accelerometer is automatically configured in power-down mode.

After power-up, the trimming parameters can be reloaded by setting the BOOT bit of the CTRL4 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 does not change after boot. During the boot duration, the device operating mode must not be changed.

If a reset to the default value of the control registers is required (registers addresses: 0Ch, 0Eh, and from 10h to 20h), it can be performed by setting the SW_RESET bit of the CTRL1 register to 1. The software reset procedure can take 50 µs: all the control registers involved in the software reset procedure cannot be accessed for read/ write operations during this period.

The boot status signal is driven to the INT1 (or INT2) interrupt pin by setting the INT1_BOOT (or INT2_BOOT) bit of the CTRL2 (or CTRL3) register to 1. The signal goes to 1 while a boot is taking place, and returns to 0 when it is done.

The flow must be performed serially (from ANY operating mode) as shown in the example below:

1. Set the device in power-down mode.
2. Set the SW_RESET bit to 1.
3. Wait 50 µs.
4. Check that the SW_RESET bit of the CTRL1 register has returned to 0.
5. Set the device in the desired operating mode.
6. Set the BOOT bit to 1.
7. Wait 20 ms.
8. Check that the BOOT bit of the CTRL4 register has returned to 0.

Note: The SIM bit of the CTRL1 register is not reset during the software reset procedure.

6 First-in first-out (FIFO) buffer

In order to limit intervention by the host processor and facilitate post-processing data for recognition of events, the MIS2DU12 embeds a first-in, first-out buffer (FIFO) for each of the three output axes, X, Y, Z, and temperature.

FIFO use allows consistent power saving for the system, it can wake up only when needed and burst the significant data out from the FIFO.

The FIFO buffer can work according to six different modes that guarantee a high level of flexibility during application development: bypass mode, FIFO mode, continuous mode, bypass-to-continuous, continuous to-FIFO mode and bypass-to-FIFO.

A programmable watermark level and the FIFO full event can be enabled to generate dedicated interrupts on the INT1 or INT2 pins.

It is possible to store the data in FIFO both when the device is configured in continuous conversion mode and when it is configured in one-shot mode.

Note: When using the FIFO with the device in one-shot mode, it is necessary to configure FIFO mode (through the FIFO_MODE[2:0] bits in the FIFO_CTRL register) before triggering new data generation.

6.1 FIFO description

The FIFO buffer is able to store up to 128 acceleration and temperature samples stored at 12-bit resolution.

The data sample set consists of 8 bytes (OUT_X_L, OUT_X_H, OUT_Y_L, OUT_Y_H, OUT_Z_L, OUT_Z_H, OUT_T_L, and OUT_T_H) and they are released to the FIFO at the selected output data rate defined by the ODR[3:0] bits in register CTRL5. Data can be retrieved from the FIFO through the eight output registers from address 28h to 2Fh.

The new sample set is placed in the first empty FIFO slot until the buffer is full, therefore, the oldest value is overwritten.

Table 14. FIFO buffer full representation (128th sample set stored at 12-bit)

Output registers	28h	29h	2Ah	2Bh	2Ch	2Dh	2Eh	2Fh
FIFO index	FIFO sample set	FIFO sample set	FIFO sample set	FIFO sample set	FIFO sample set	FIFO sample set	FIFO sample set	FIFO sample set
FIFO(0)	Xl(0)	Xh(0)	Yl(0)	Yh(0)	Zl(0)	Zh(0)	Tl(0)	Th(0)
FIFO(1)	Xl(1)	Xh(1)	Yl(1)	Yh(1)	Zl(1)	Zh(1)	Tl(1)	Th(1)
FIFO(2)	Xl(2)	Xh(2)	Yl(2)	Yh(2)	Zl(2)	Zh(2)	Tl(2)	Th(2)
...	...	...	...	...	...	...	...	...
FIFO(126)	Xl(126)	Xh(126)	Yl(126)	Yh(126)	Zl(126)	Zh(126)	Tl(126)	Th(126)
FIFO(127)	Xl(127)	Xh(127)	Yl(127)	Yh(127)	Zl(127)	Zh(127)	Tl(127)	Th(127)

Table 15. FIFO buffer full representation (129th sample set stored at 12-bit and 1st sample discarded)

Output registers	28h	29h	2Ah	2Bh	2Ch	2Dh	2Eh	2Fh
FIFO index	FIFO sample set	FIFO sample set	FIFO sample set	FIFO sample set	FIFO sample set	FIFO sample set	FIFO sample set	FIFO sample set
FIFO(0)	Xl(1)	Xh(1)	Yl(1)	Yh(1)	Zl(1)	Zh(1)	Tl(1)	Th(1)
FIFO(1)	Xl(2)	Xh(2)	Yl(2)	Yh(2)	Zl(2)	Zh(2)	Tl(2)	Th(2)
FIFO(2)	Xl(3)	Xh(3)	Yl(3)	Yh(3)	Zl(3)	Zh(3)	Tl(3)	Th(3)
...	...	...	...	...	...	...	...	...
FIFO(126)	Xl(127)	Xh(127)	Yl(127)	Yh(127)	Zl(127)	Zh(127)	Tl(127)	Th(127)
FIFO(127)	Xl(128)	Xh(128)	Yl(128)	Yh(128)	Zl(128)	Zh(128)	Tl(128)	Th(128)

Table 14. FIFO buffer full representation (128th sample set stored at 12-bit) represents the FIFO full status when 128 samples are stored in the buffer while Table 15. FIFO buffer full representation (129th sample set stored at 12-bit and 1st sample discarded) represents the next step when the 129th sample is inserted into FIFO and the 1st sample is overwritten. The new oldest sample set is made available in the output registers.

When FIFO is enabled and the mode is different from bypass, the MIS2DU12 output registers always contain the oldest FIFO sample set.

In order to maximize the number of data collected in FIFO, it is possible to enable 2x depth mode, writing the FIFO_DEPTH bit in FIFO_CTRL (15h). If 2x depth mode is enabled, the most significant 8 bits for each accelerometer axis are stored in FIFO: each FIFO word contains two consecutive samples. The temperature data is not stored in FIFO but is available in the TEMP_OUT_L (30h) and TEMP_OUT_H (31h) registers (see Table 16. FIFO buffer full representation (256th sample set stored with 2x depth enabled)).

In 2x depth mode (FIFO_DEPTH = 1), the automatic wraparound feature applies from address 0x28 to 0x2D and back to 0x28. In this case, the ROUNDING_XYZ bit must not be set to 1.

Table 16. FIFO buffer full representation (256th sample set stored with 2x depth enabled)

Output registers	28h	29h	2Ah	2Bh	2Ch	2Dh	2Eh	2Fh
FIFO index	FIFO sample set	FIFO sample set	FIFO sample set	FIFO sample set	FIFO sample set	FIFO sample set	FIFO sample set	FIFO sample set
FIFO(0)	Xh(0)	Yh(0)	Zh(0)	Xh(1)	Yh(1)	Zh(1)	-	-
FIFO(1)	Xh(2)	Yh(2)	Zh(2)	Xh(3)	Yh(3)	Zh(3)	-	-
FIFO(2)	Xh(4)	Yh(4)	Zh(4)	Xh(5)	Yh(5)	Zh(5)	-	-
...	...	...	...	...	...	...	...	...
FIFO(126)	Xh(252)	Yh(252)	Zh(252)	Xh(253)	Yh(253)	Zh(253)	-	-
FIFO(127)	Xh(254)	Yh(254)	Zh(254)	Xh(255)	Yh(255)	Zh(255)	-	-

6.2 FIFO registers

The FIFO buffer is managed by two different registers, one allows enabling and configuring the FIFO behavior, the other one provides information about the buffer status. A few other registers are used to route FIFO events to the INT1 and INT2 pins to interrupt the application processor. These are discussed in Section 6.3: FIFO interrupts.

6.2.1 FIFO_CTRL

The FIFO_CTRL register contains the mode in which the FIFO is set. At reset, by default, the FIFO mode is bypass, which means that the FIFO is off; the FIFO is enabled and starts storing the samples as soon as the mode is set to a mode other than bypass.

Table 17. FIFO_CTRL register

b7	b6	b5	b4	b3	b2	b1	b0
ROUNDING_XYZ	FIFO_DEPTH	-	-	STOP_ON_FTH	FIFO_MODE2	FIFO_MODE1	FIFO_MODE0

- FIFO_MODE[2:0] bits select the FIFO buffer behavior:
  - FIFO_MODE[2:0] = 000: bypass mode (FIFO turned off)
  - FIFO_MODE[2:0] = 001: FIFO mode
  - FIFO_MODE[2:0] = 110: continuous mode
  - FIFO_MODE[2:0] = 011: continuous-to-FIFO mode
  - FIFO_MODE[2:0] = 100: bypass-to-continuous mode
  - FIFO_MODE[2:0] = 111: bypass-to-FIFO mode
- STOP_ON_FTH bit: allows limiting the FIFO depth to the threshold level (see Section 6.2.2: FIFO_WTM for setting the threshold).
- FIFO_DEPTH bit: if 1, enables 2x depth mode.
- ROUNDING_XYZ bit: enables the automatic wraparound feature for a circular continuous read from OUT_X_L (28h) to OUT_Z_H (2Dh).

6.2.2 FIFO_WTM

Table 18. FIFO_WTM register

b7	b6	b5	b4	b3	b2	b1	b0
-	FTH6	FTH5	FTH4	FTH3	FTH2	FTH1	FTH0

- FTH[6:0] bits: FIFO watermark threshold, maximum value is 127.

6.2.3 FIFO_STATUS1

Table 19. FIFO_STATUS1 register

b7	b6	b5	b4	b3	b2	b1	b0
FTH	FIFO_OVR	-	-	-	-	-	-

- FTH bit: shows the FIFO watermark status (0: FIFO filling is lower than WTM; 1: FIFO filling is equal to or higher than WTM)
- FIFO_OVR bit: shows the FIFO overrun status (1 if FIFO has overwritten data).

6.2.4 FIFO_STATUS2

Table 20. **FIFO_STATUS2 register**

b7	b6	b5	b4	b3	b2	b1	b0
FSS7	FSS6	FSS5	FSS4	FSS3	FSS2	FSS1	FSS0

- FSS[7:0] bits show the number of unread data stored in FIFO.

Table 21. **FIFO_SAMPLES behavior assuming FTH[6:0] = 15 (stored at 12-bit)**

FIFO_FTH	FSS7 (FIFO full)	FIFO_OVR	Diff[6:0]	Unread FIFO samples	Timing
0	0	0	0000000	0	t_0
0	0	0	0000001	1	t_0 + 1/ODR
0	0	0	0000010	2	t_0 + 2/ODR
...	...	...	...	...	...
0	0	0	0001110	14	t_0 + 14/ODR
1	0	0	0001111	15	t_0 + 15/ODR
...	...	...	...	...	...
1	0	0	1111111	127	t_0 + 127/ODR
1	1	0	0000000	128	t_0 + 128/ODR
1	1	1	0000000	128	t_0 + 129/ODR

Table 22. **FIFO_SAMPLES behavior assuming FTH[6:0] = 15 (stored with 2x depth mode enabled)**

FIFO_FTH	FSS7 (FIFO full)	FIFO_OVR	Diff[6:0]	Unread FIFO samples	Timing
0	0	0	0000000	0	t_0
0	0	0	0000001	2	t_0 + 2/ODR
0	0	0	0000010	4	t_0 + 4/ODR
...	...	...	...	...	...
0	0	0	0001110	28	t_0 + 28/ODR
1	0	0	0001111	30	t_0 + 30/ODR
...	...	...	...	...	...
1	0	0	1111111	254	t_0 + 254/ODR
1	1	0	0000000	256	t_0 + 256/ODR
1	1	1	0000000	256	t_0 + 258/ODR

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.

6.3 FIFO interrupts

There are three specific FIFO events that can be routed to the pins in order to interrupt the main processor: FIFO threshold, FIFO full, and FIFO overrun.

All FIFO events can be routed to the INT1 and INT2 pins.

6.3.1 FIFO threshold

The FIFO threshold is a configurable feature that can be used to generate a specific interrupt in order to know when the FIFO buffer contains at least the number of samples defined as the threshold level. The user can select the desired level in a range from 0 to 127 using the FTH[6:0] field in the FIFO_WTM register.

If the number of entries in FIFO (FSS[7:0]) is greater than or equal to the value programmed in FTH[6:0], the FTH bit is set high in the FIFO_STATUS register.

FSS[7:0] increases by one step at the ODR frequency if 2x depth mode is disabled (otherwise one step at the ODR/2 frequency) and decreases by one step every time that a sample set read is performed by the user.

The threshold flag (FTH) can be routed to the INT1 and INT2 pins to provide a dedicated interrupt for the application processor that can, as a consequence, consume less power between interrupts. The INT1_F_FTH bit of the CTRL2 register and the INT2_F_FTH bit of the CTRL3 register are dedicated to this task.

6.3.2 FIFO full

It is possible to configure the device to generate an interrupt whenever the FIFO becomes full. To do so, just set the INT1_F_FULL bit of the CTRL2 register to 1 (or the INT2_F_FULL bit of the CTRL3 register to 1). To avoid losing samples, the host processor reading speed is important in order to free slots faster than new data is made available.

In order to reduce the possibility of losing data due to occurrences of the overrun condition, it is not recommended to use the INTx_F_FULL signal to read FIFO. It is recommended to use the INTx_F_FTH signal by setting FIFO_WTM = total fifo size - 1.

6.3.3 FIFO overrun

It is possible to configure the device to generate an interrupt if the overrun event occurs in FIFO. To do so, just set the INT1_F_OVR bit of the CTRL2 register to 1 (or the INT2_F_OVR bit of the CTRL3 register to 1).

6.3.4 FIFO empty

When FSS[7:0] is equal to 00000000, FIFO is empty. When FIFO is emptied, a special value is used in order to recognize the empty condition and no duplicated samples are read.

In high-resolution (12 bits) batch mode, the least significant 4 bits of data read during the empty condition are 1111. In 2X depth mode, in the empty condition the values stored in FIFO are equal to 7Fh in all the output registers. The accelerometer data stored in FIFO are limited to 7Eh in order to differentiate the empty condition.

6.4 FIFO modes

The MIS2DU12 FIFO buffer can be configured to operate in six different modes selectable by the FIFO_MODE[2:0] field in the FIFO_CTRL register. Available configurations ensure a high-level of flexibility and extend the number of usable functions in application development.

Bypass, FIFO, continuous, bypass-to-continuous, continuous-to-FIFO, and bypass-to-FIFO modes are described in the following sections.

When the FIFO operating mode must be changed at the same time as the ODR of the sensor, adhere to the following procedure:

1. Change the ODR as desired.
2. Clear the DRDY signal by reading the output registers and wait for the next DRDY signal.
3. Set the FIFO in bypass mode.
4. Wait 100 µs.
5. Set the FIFO in the desired operating mode.

6.4.1 Bypass mode

When bypass mode is enabled, the FIFO is not operational: buffer content is cleared and the FIFO buffer remains empty until another mode is selected.

Bypass mode is activated by setting the FIFO_MODE[2:0] field to 000 in the FIFO_CTRL register and the configuration takes about 100 µs before being active in continuous conversion mode. Otherwise, if the device is in one-shot mode, it is recommended to read back the register in order to verify the FIFO bypass configuration.

Bypass mode must be used in order to stop and reset the FIFO buffer when a different operating mode or a different FIFO mode is intended to be used. Note that placing the FIFO buffer in bypass mode clears the entire content of the buffer.

Note: After FIFO bypass mode is set (by configuring the FIFO_MODE[2:0] bits to 000 in the FIFO_CTRL register), it is necessary to wait at least 100 µs before configuring a different FIFO mode through the FIFO_MODE[2:0] bits.

6.4.2 FIFO mode

In FIFO mode, the buffer continues filling until full (128 sample sets stored). As soon as the FSS7 bit of the FIFO_STATUS2 register goes to 1, the FIFO stops collecting data and its content remains unchanged until a different mode is selected.

FIFO mode is activated by setting the FIFO_MODE[2:0] field to 001 in the FIFO_CTRL register.

By selecting this mode, FIFO starts data collection and FSS[7:0] changes according to the number of sets of samples stored. At the end of the procedure, the FFS7 bit rises to 1, and data can then be retrieved, performing a 128 sample set read from the output registers. Communication speed is not so important in FIFO mode because data collection is stopped and there is no risk of overwriting acquired data. Before restarting FIFO mode, at the end of the reading procedure it is necessary to exit bypass mode.

In order to serve the FIFO full (FSS7 bit) event as soon as possible, it is recommended to route it to the pin in order to generate an interrupt, which is then be managed by a specific handler:

1. Set INT1_F_FULL to 1: enables the FIFO_FULL interrupt.
2. Set FIFO_MODE[2:0] = 001: enables FIFO mode.
3. When the FIFO full event interrupt is generated or the FSS7 bit is high (polling mode), read data from the output registers.

When FIFO mode is enabled, the buffer starts to collect data and fills all 128 slots (from F0 to F127) at the selected output data rate. When the buffer is full the FSS7 bit goes high and data collection is permanently stopped; the user can decide to read FIFO content at any time because it is maintained unchanged until bypass mode is selected.

The read procedure may be performed inside an interrupt handler triggered by a FIFO FULL condition (FSS7) and it is composed of 128 sample sets of 6 bytes (8 with temperature) for a total of 768 bytes (1024 bytes with temperature) and retrieves data starting from the oldest sample stored in FIFO (F0). The bypass mode setting resets FIFO and allows the user to enable FIFO mode again.

6.4.3 Continuous mode

In continuous mode, FIFO continues filling, when the buffer is full, the FIFO index restarts from the beginning and older data is replaced by current data. The oldest values continue to be overwritten until a read operation frees FIFO slots. The host processor reading speed is most important in order to free slots faster than new data is made available. The FIFO_MODE[2:0] bit field in bypass configuration is used to stop this mode.

Follow these steps for FIFO continuous configuration, which sets a threshold to generate an interrupt in order to trigger a read by the application processor:

1. Set FTH[6:0] to the desired threshold value to trigger the FIFO threshold event.
2. Set INT1_F_FTH to 1: enables the FIFO threshold interrupt.
3. Activate continuous mode by setting the FIFO_MODE[2:0] field to 110 in the FIFO_CTRL register.
4. When the FIFO threshold event flag (FIFO_WTM_IA bit in the FIFO_STATUS1 register) goes high and the FIFO threshold event interrupt is generated, read data from the output registers.

When continuous mode is enabled, the FIFO buffer continuously fills (from F0 to F127) at the selected output data rate (or half the output data rate if 2x depth mode is selected). When the buffer reaches the desired threshold value, the FTH interrupt goes high, and the application processor may read all FIFO samples as soon as possible to avoid loss of data and to limit intervention by the host processor, which increases system efficiency. In the case at least one sample is overwritten to store the new data, the FIFO overrun flag (FIFO_OVR_IA bit) goes high. See Section 6.5: Retrieving data from FIFO for more details on FIFO reading speed.

When a read command is sent to the device, the content of the output registers is moved to the SPI/I²C/MIPI I3C® register and the current oldest FIFO value is shifted into the output registers in order to allow the next read operation.

6.4.4 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 the selected interrupt (that is, wake-up, free-fall, single or double-tap, 6D/4D, or any combination of these) occurs and is maintained until bypass mode is set.

This mode can be used in order to analyze the history of samples that generated an interrupt; the standard operation is to read FIFO content when a FIFO mode is triggered and the FIFO buffer is full and stopped.

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

1. Configure the desired interrupt generator by following the instructions in Section 5: Interrupt generation and embedded functions.
2. Activate continuous-to-FIFO mode by setting the FIFO_MODE[2:0] field to 011 in the FIFO_CTRL register.

Note: When the requested event takes place, the FIFO mode change is triggered if and only if the event flag is routed to the INT1 pin.

While in continuous mode the FIFO buffer continues filling. When the requested event takes place, the FIFO mode changes; then, as soon as the buffer becomes full, the FSS7 bit of the FIFO_STATUS2 register is set high, the next samples overwrite the oldest, and the FIFO stops collecting data.

6.4.5 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 in bypass mode and switches to continuous mode when the selected interrupt (that is, wake-up, free-fall, single or double-tap, 6D/4D, or any combination of these) occurs and is maintained until bypass mode is set. Follow these steps for bypass-to-continuous mode configuration:

1. Configure the desired interrupt generator by following the instructions in Section 5: Interrupt generation and embedded functions.
2. Set FTH[6:0] to the desired threshold value to trigger the FIFO threshold event.
3. Set INT1_F_FTH to 1: enables the FIFO threshold interrupt.
4. Activate bypass-to-continuous mode by setting the FIFO_MODE[2:0] field to 100 in the FIFO_CTRL register.
5. When the FIFO threshold event interrupt is generated, read data from the output registers. The sample that generated the trigger is available in FIFO.

When the FTH interrupt is generated, data is read from the accelerometer output registers. The sample that generated the trigger is available in FIFO.

The FIFO is initially in bypass mode, so no samples enter the FIFO buffer. As soon as an event occurs (for example, a wake-up or a free-fall event) the FIFO switches to continuous mode and starts to store the samples at the configured data rate. When the programmed threshold is reached, the FTH interrupt goes high, and the application processor may start reading all FIFO samples (128 sample sets of 6 bytes, 8 bytes with temperature) as soon as possible to avoid loss of data.

If the FTH flag was set, it goes to 0 as soon as the first FIFO set is read, creating space for new data. Since the FIFO is still in continuous mode, the FIFO eventually reaches the threshold again and the situation repeats.

6.4.6 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 an event condition occurs.

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, which is maintained until bypass mode is set.

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

1. Configure the desired interrupt generator by following the instructions in Section 5: Interrupt generation and embedded functions.
2. Set INT1_F_FULL to 1: enables the FIFO_FULL interrupt.
3. Activate bypass-to-FIFO mode by setting the FIFO_MODE[2:0] field to 111 in the FIFO_CTRL register.
4. When the FIFO full event interrupt is generated, read data from the output registers. The sample that generated the trigger is available in FIFO.

When the FIFO_FULL interrupt is generated, data is read from the accelerometer output registers. The sample that generated the trigger is available in FIFO.

The FIFO is initially in bypass mode, so no samples enter the FIFO buffer. As soon as an event occurs (for example, a wake-up or a free-fall event) the FIFO switches to FIFO mode and starts to store the samples at the configured data rate. When the buffer is full, the FIFO_FULL interrupt goes high, and the application processor may start reading all FIFO samples (128 sample sets of 6 bytes, 8 bytes with temperature).

6.5 Retrieving data from FIFO

When the FIFO mode is different from bypass, reading the output registers (28h to 2Fh) returns the oldest FIFO sample set.

Whenever the output registers are read, their content is moved to the SPI/I²C/MIPI I3C® output buffer. FIFO slots are ideally shifted up one level in order to release room for receiving a new sample and the output registers load the current oldest value stored in the FIFO buffer.

The whole FIFO content is retrieved by performing 128 read operations from the accelerometer and temperature standard output registers.

The size of the data stored in FIFO is dependent on the selected power mode.

Data can be retrieved from FIFO using every read byte combination in order to increase application flexibility (ex: 768 (or 1024) single byte read, 128 reads of 6 (or 8) bytes, 1 multiple read of 768 (or 1024) bytes, and so forth).

It is recommended to read all FIFO slots in a multiple byte read of 768 (or 1024) bytes (6 or 8 output registers by 128 slots). The automatic wraparound feature is available when reading the accelerometer and temperature output registers through a multiple byte read: reading them starting from the initial register (28h) to the final one (2Fh) results in the auto-increment feature providing, as the next address being read, the initial address range once the last address range has been read (the device rolls back to 28h when the register 2Fh is reached).

In case it is not necessary to read the temperature data stored in FIFO, the wraparound feature can be limited to accelerometer output registers only by setting the ROUNDING_XYZ bit of the FIFO_CTRL register to 1. The device rolls back to 28h when the register 2Dh is reached.

The I²C speed is lower than SPI and it needs about 29 clock pulses to start communication (start, slave address, register address+write, restart, register address+read) plus an additional 9 clock pulses for every byte to read (total of 83 clock pulses). So, in the case of standard I²C mode being used (max rate 100 kHz), a single sample set read takes 830 µs while total FIFO download takes about 69.41 ms (29 + 9 * 768 clock pulses).

In the case of the SPI, instead, 8 clock pulses are required only once at the very beginning to get started (for the register address, including the r/w bit) plus an additional 8 clock pulses for every byte to read. With a 2 MHz clock a single sample set read would take 28 µs, while total FIFO download takes about 3.08 ms.

If this recommendation were followed, using a standard I²C (100 kHz) the complete FIFO read (69.41 ms) is taking about 56/ODR with ODR at 800 Hz. Using a SPI @ 2 MHz (10 MHz is the maximum supported by the device) the complete FIFO read would take about two periods of data generation (2/ODR) with ODR at 800 Hz.

So, in order to not lose samples, the application reads samples before the FIFO becomes full, setting a threshold and using the FTH interrupt (see Section 6.3: FIFO interrupts).

Table 23. Example: threshold function of ODR reading 768 byte

ODR (Hz)	FTH_THS (I²C @ 100 kHz)	FTH_THS (I²C @ 400 kHz)	FTH_THS (SPI @ 2 MHz)
12.5	128	128	128
25	73	128	128
50	36	128	128
100	17	73	128
200	8	36	128
400	4	17	104
800	1	8	51

7 Temperature sensor

The MIS2DU12 is provided with an internal temperature sensor that is suitable for ambient temperature measurement.

If the sensor is in power-down mode, the temperature sensor is off and shows the last value measured.

The DRDY bit in the STATUS register is set high when a new set of data is available, including temperature.

The temperature is available in OUT_T_L (2Eh), OUT_T_H (2Fh) stored as two's complement data, left-justified in 12-bit mode, and duplicated in registers TEMP_OUT_L (30h) - TEMP_OUT_H (31h). If FIFO is enabled and 2x depth mode is disabled, registers OUT_T_L (2Eh), OUT_T_H (2Fh) are filled with the last temperature data in FIFO.

Refer to the table below for temperature sensor details.

Table 24. Temperature sensor characteristics

Parameter	Typ.(1)
Data at 25°C	0 LSB
Temperature refresh rate	ODR
Temperature sensor output change vs. temperature	0.045(2) °C/LSB

1. *Typical specifications are not guaranteed.*
2. *12-bit resolution*

7.1 Example of temperature data calculation

Table 25. Output data registers content vs. temperature 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 (that is, no offset, no gain error, and so forth).

Table 25. Output data registers content vs. temperature

Temperature values	OUT_T_H	OUT_T_L
23.92°C	FEh	80h
25.0°C	00h	00h
26.08°C	01h	80h

8 Self-test

Self-test mode allows checking the sensor functionality without moving it, applying an actuation force to the sensor and simulating a definite input acceleration.

The procedures for positive and negative self-tests are as follows:

Positive self-test

1. Set the STSIGN[2:0] bits in the ST_SIGN register to 110 (note: do not change the value of the other bits in the ST_SIGN register).
2. Set the device in power-down mode (ODR[3:0] = 0000 in the CTRL5 register) and wait 10 ms.
3. Set the ST[1:0] bits in the CTRL3 register to 10.
4. Set the CTRL5 register to 92h (ODR = 200 Hz, BW = ODR/2, FS = ±8 *g*) and wait 100 ms.
5. Read 5 output samples, compute the average for each axis, and save the result in OUT1.
6. Set the device in power-down mode (ODR[3:0] = 0000 in the CTRL5 register) and wait 10 ms.
7. Set the ST[1:0] bits in the CTRL3 register to 01.
8. Set the CTRL5 register to 92h (ODR = 200 Hz, BW = ODR/2, FS = ±8 *g*) and wait 100 ms.
9. Read 5 output samples, compute the average for each axis, and save the result in OUT2.
10. Set the device in power-down mode (ODR[3:0] = 0000 in the CTRL5 register) and wait 10 ms.
11. Set the ST[1:0] bits in the CTRL3 register to 00.
12. Self-test deviation is |OUT2 - OUT1|. Compute the difference in absolute value for each axis and verify that it falls within the given range.

Negative self-test

1. Set the STSIGN[2:0] bits in the ST_SIGN register to 001 (note: do not change the value of the other bits in the ST_SIGN register).
2. Set the device in power-down mode (ODR[3:0] = 0000 in the CTRL5 register) and wait 10 ms.
3. Set the ST[1:0] bits in the CTRL3 register to 10.
4. Set the CTRL5 register to 92h (ODR = 200 Hz, BW = ODR/2, FS = ±8 *g*) and wait 100 ms.
5. Read 5 output samples, compute the average for each axis, and save the result in OUT1.
6. Set the device in power-down mode (ODR[3:0] = 0000 in the CTRL5 register) and wait 10 ms.
7. Set the ST[1:0] bits in the CTRL3 register to 01.
8. Set the CTRL5 register to 92h (ODR = 200 Hz, BW = ODR/2, FS = ±8 *g*) and wait 100 ms.
9. Read 5 output samples, compute the average for each axis, and save the result in OUT2.
10. Set the device in power-down mode (ODR[3:0] = 0000 in the CTRL5 register) and wait 10 ms.
11. Set the ST[1:0] bits in the CTRL3 register to 00.
12. Self-test deviation is |OUT2 - OUT1|. Compute the difference in absolute value for each axis and verify that it falls within the given range.

Notes:

1. Keep the device still during the self-test procedures.
2. Refer to the datasheet for minimum and maximum values.

Revision history

Table 26. **Document revision history**

Date	Version	Changes
12-Mar-2026	1	Initial release

Contents

List of tables

List of figures

IMPORTANT NOTICE – READ CAREFULLY

STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice.

In the event of any conflict between the provisions of this document and the provisions of any contractual arrangement in force between the purchasers and ST, the provisions of such contractual arrangement shall prevail.

The purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST’s terms and conditions of sale in place at the time of order acknowledgment.

The purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of the purchasers’ products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

If the purchasers identify an ST product that meets their functional and performance requirements but that is not designated for the purchasers' market segment, the purchasers shall contact ST for more information.

ST and the ST logo are trademarks of ST. For additional information about ST trademarks, refer to www.st.com/trademarks. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2026 STMicroelectronics – All rights reserved