LPS27HHW: MEMS pressure sensor with water-resistant package

Introduction

This document is intended to provide usage information and application hints related to ST's LPS27HHW device.

The LPS27HHW is an ultra-compact piezo-resistive absolute pressure sensor which functions as a digital output barometer with a digital I²C / MIPI I3C^{SM} / SPI serial interface standard output. Its operating pressure range is from 260 hPa to 1260 hPa and the device is capable of measuring pressure values with output data rates up to 200 Hz.

The LPS27HHW has an integrated 128-level first-in first-out (FIFO) buffer allowing the user to store data in order to limit intervention by the host processor.

The LPS27HHW is available in a ceramic LGA package with metal lid and is guaranteed to operate over an extended temperature range from -40 °C to +85 °C. The package is holed to allow external pressure to reach the sensing element.

This document does not modify the content of the official datasheet. Please refer to the datasheet for parameter specifications.

1 Pin description

Figure 1. **Pin connections**

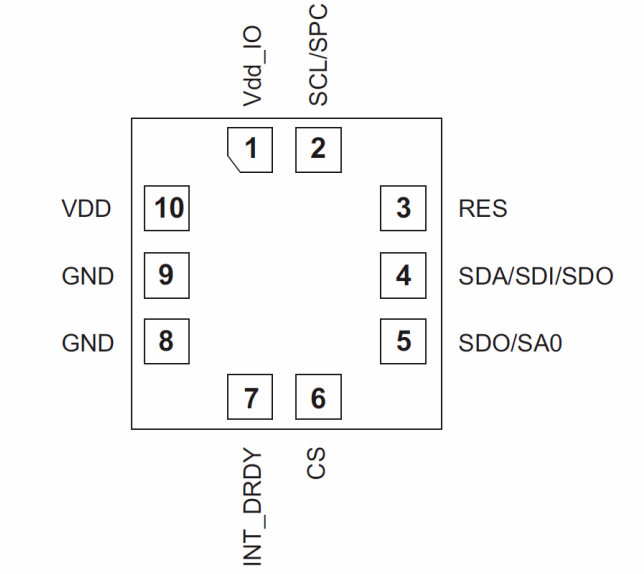


Table 1. **Pin list, functions, and internal status**

Pin number	Name	Function	Internal pin status
1	Vdd_IO	Power supply for I/O pins	
2	SCL SPC	I²C / MIPI I3C^{SM} serial clock (SCL) SPI serial port clock (SPC)	Default: Input without pull-up
3	Reserved	Connect to GND	
4	SDA SDI SDI/SDO	I²C / MIPI I3C^{SM} serial data (SDA) 4-wire SPI serial data input (SDI) 3-wire SPI serial data input/output (SDI/SDO)	Default: Input without pull-up. Pull-up is enabled if bit SDA_PU_EN=1, register IF_CTRL (@0Eh).
5	SDO SA0	4-wire SPI serial data output (SDO) I²C least significant bit of the device address (SA0) MIPI I3C^{SM} least significant bit of the static address (SA0)	Default: Input without pull-up. Pull-up is enabled if bit SDO_PU_EN=1, register IF_CTRL (@0Eh).
6	CS	SPI enable, I²C and MIPI I3C^{SM} / SPI mode selection (1: SPI idle mode / I²C and MIPI I3C^{SM} communication enabled; 0: SPI communication mode / I²C and MIPI I3C^{SM} disabled)	Default: Input without pull-up.
7	INT_DRDY	Interrupt or data-ready	Default: Input with pull-down. Pull-down is disabled if bit PD_DIS_INT1=1, register IF_CTRL (@0Eh).
8	GND	0 V supply	
9	GND	0 V supply	
10	VDD	Power supply	

Note: Internal pull-up value is from 30 kΩ to 50 kΩ, depending on Vdd_IO.

2 Registers

Table 2. Registers

Register name	Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTERRUPT_CFG	0Bh	AUTOREFP	RESET_ARP	AUTOZERO	RESET_AZ	DIFF_EN	LIR	PLE	PHE
THS_P_L	0Ch	THS7	THS6	THS5	THS4	THS3	THS2	THS1	THS0
THS_P_H	0Dh	-	THS14	THS13	THS12	THS11	THS10	THS9	THS8
IF_CTRL	0Eh	INT_EN_I3C	0	0	SDA_PU_EN	SDO_PU_EN	PD_DIS_INT1	I3C_DISABLE	I2C_DISABLE
WHO_AM_I	0Fh	1	0	1	1	0	0	1	1
CTRL_REG1	10h	0	ODR2	ODR1	ODR0	EN_LPFP	LPFP_CFG	BDU	SIM
CTRL_REG2	11h	BOOT	INT_H_L	PP_OD	IF_ADD_INC	0	SWRESET	LOW_NOISE_EN	ONE_SHOT
CTRL_REG3	12h	0	0	INT_F_FULL	INT_F_WTM	INT_F_OVR	DRDY	INT_S1	INT_S0
FIFO_CTRL	13h	0	0	0	0	STOP_ON_WTM	TRIG_MODES	F_MODE1	F_MODE0
FIFO_WTM	14h	0	WTM6	WTM5	WTM4	WTM3	WTM2	WTM1	WTM0
REF_P_L	15h	REFL7	REFL6	REFL5	REFL4	REFL3	REFL2	REFL1	REFL0
REF_P_H	16h	REFL15	REFL14	REFL13	REFL12	REFL11	REFL10	REFL9	REFL8
RPDS_L	18h	RPDS7	RPDS6	RPDS5	RPDS4	RPDS3	RPDS2	RPDS1	RPDS0
RPDS_H	19h	RPDS15	RPDS14	RPDS13	RPDS12	RPDS11	RPDS10	RPDS9	RPDS8
INT_SOURCE	24h	BOOT_ON	0	0	0	0	IA	PL	PH
FIFO_STATUS1	25h	FSS7	FSS6	FSS5	FSS4	FSS3	FSS2	FSS1	FSS0
FIFO_STATUS2	26h	FIFO_WTM_IA	FIFO_OVR_IA	FIFO_FULL_IA	-	-	-	-	-
STATUS	27h	-	-	T_OR	P_OR	-	-	T_DA	P_DA
PRESS_OUT_XL	28h	POUT7	POUT6	POUT5	POUT4	POUT3	POUT2	POUT1	POUT0
PRESS_OUT_L	29h	POUT15	POUT14	POUT13	POUT12	POUT11	POUT10	POUT9	POUT8
PRESS_OUT_H	2Ah	POUT23	POUT22	POUT21	POUT20	POUT19	POUT18	POUT17	POUT16
TEMP_OUT_L	2Bh	TOUT7	TOUT6	TOUT5	TOUT4	TOUT3	TOUT2	TOUT1	TOUT0
TEMP_OUT_H	2Ch	TOUT15	TOUT14	TOUT13	TOUT12	TOUT11	TOUT10	TOUT9	TOUT8
FIFO_DATA_OUT_PRESS_XL	78h	FIFO_P7	FIFO_P6	FIFO_P5	FIFO_P4	FIFO_P3	FIFO_P2	FIFO_P1	FIFO_P0
FIFO_DATA_OUT_PRESS_L	79h	FIFO_P15	FIFO_P14	FIFO_P13	FIFO_P12	FIFO_P11	FIFO_P10	FIFO_P9	FIFO_P8
FIFO_DATA_OUT_PRESS_H	7Ah	FIFO_P23	FIFO_P22	FIFO_P21	FIFO_P20	FIFO_P19	FIFO_P18	FIFO_P17	FIFO_P16
FIFO_DATA_OUT_TEMP_L	7Bh	FIFO_T7	FIFO_T6	FIFO_T5	FIFO_T4	FIFO_T3	FIFO_T2	FIFO_T1	FIFO_T0
FIFO_DATA_OUT_TEMP_H	7Ch	FIFO_T15	FIFO_T14	FIFO_T13	FIFO_T12	FIFO_T11	FIFO_T10	FIFO_T9	FIFO_T8

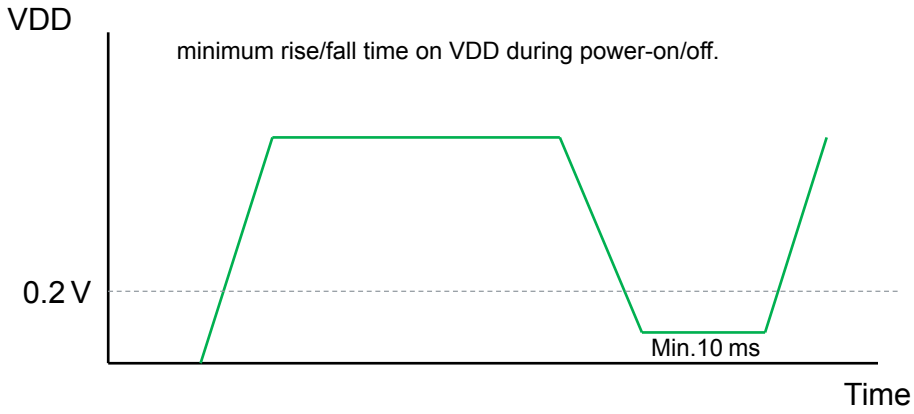
3 Operating / noise modes

The LPS27HHW features three operating modes:

- Power-down mode;
- One-shot mode;
- Continuous mode.

Basically, the LPS27HHW can read environmental data at the very moment the controlling MCU requires it, when configured in one-shot mode, or can keep reading data at predefined frequencies (fixed output data rates, ODRs), when configured in continuous mode. The device offers a wide VDD voltage range from 1.7 V to 3.6 V and a Vdd_IO range from 1.7 V to VDD + 0.1 V. The VDDIO pin can be powered at the same time as the VDD pin or before, however the Vdd_IO level must be less than or equal to VDD + 0.1 V when the device is fully operational (at regime). In order to avoid potential conflicts, during the power-on sequence it is recommended to set the lines connected to the device IO pins to high-impedance state on the host side. Furthermore, to guarantee proper device power-off and the next power-on reset sequence to be successful, it is recommended to drive the VDD line to GND (less than 0.2 V) and keep it at this level for at least 10 ms as illustrated in the figure below. Power timings and profiles need to be taken into account when managing the device power supplies (VDD and Vdd_IO).

Figure 2. VDD power-on/off sequence



- VDD rise/fall time: 10 µs ~ 100 ms, these values represent the minimum and the maximum rise/fall time on VDD during power-on/off.
- VDD must be lower than 0.2 V for at least 10 ms during power-off sequence for correct POR.

After the power supply is applied, the LPS27HHW requires a boot procedure of 4.5 ms (maximum) to load the trimming parameters. After the boot is completed, the device is configured in Power-Down mode, ready to communicate with the host for register configurations and data measurements.

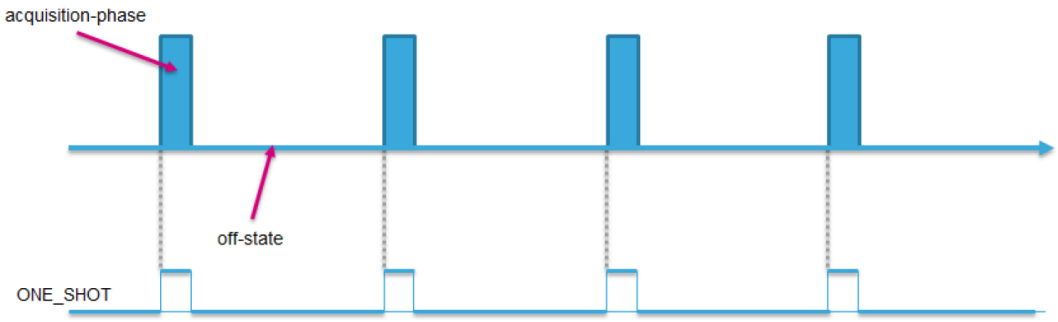
3.1 Power-down mode

Power-down mode is used to put the device in rest condition. When the device is in power-down mode, no data acquisition happens, almost all internal blocks of the device are switched off to minimize current drainage. In power-down mode the LPS27HHW can reach its lowest power consumption achievable while power supplied. While in power-down mode, I²C / MIPI I3C^{SM} / SPI communication serial interfaces are kept active to allow communication with the device and setting configurations. 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 device is in power-down mode when the ODR[2:0] bits of CTRL_REG1(@10h) register are set to ‘000’.

3.2 One-shot mode

One-shot mode is used to execute single data acquisitions at a desired pace. After the acquisition has been completed, the device automatically sets itself to power-down mode. One-shot mode has to be executed while the device is in power-down mode by setting the ONESHOT bit (default value '0') in CTRL_REG2(@11h) to '1'. When this happens, a single data acquisition is executed and read data are made available in the output registers. Once the acquisition is completed and the output registers updated, the device automatically enters again power-down mode and the ONE_SHOT bit is self-cleared (to '0').

Figure 3. One-shot mode



One-shot mode is independent of the programmed output data rate (ODR). This mode depends on the frequency at which the ONESHOT bit is set to 1 by the microcontroller/application processor. The typical time needed for the generation of the new data and the maximum ODR frequency achievable in one-shot mode is given in the following table and strictly depends on the low-noise/low-current mode selected (please refer to Section 3.4 Low-noise / low-current mode configuration).

Table 3. Typical conversion time and maximum ODR in one-shot mode

Mode	Typical data conversion time [ms]	Maximum ODR [Hz]
Low noise	13.2	50
Low current	4.7	200

3.3 Continuous mode

Continuous mode is designed to keep reading data at a specific predefined selectable output data rate (ODR). Output registers are updated with fresher readings every given period according to the selected ODR frequency. Continuous mode ODR selection is made through bit range ODR[2:0], CTRL_REG1(@10h) register. When ODR[2:0] bits are set to a value different than '000' (power-down mode), the device enters Continuous mode and immediately starts to sample pressure and temperature data and put them in the output registers at the selected frequency (Table 4).

Table 4. ODR selection

ODR[2:0]	ODR [Hz]
000	Power-down mode
001	1
010	10
011	25
100	50
101	75
110	100
111	200

3.4 Low-noise / low-current mode configuration

In continuous and one-shot mode a set of configuration options is available. The right trade-off among resolution, output data rate and power consumption has to be identified in order to make the sensor suitable for the specific design requirements. The bit LOW_NOISE_EN, CTRL_REG2(@11h) register, allows selecting two possible configurations:

- When the LOW_NOISE_EN bit is set to '0', low-current mode is selected (default configuration);
- When the LOW_NOISE_EN bit is set to '1', low-noise mode is selected

Note: For proper behavior of the pressure sensor, the LOW_NOISE_EN bit must be changed only when the device is in power-down.

Note: If the selected ODR is 100 Hz or 200 Hz, the LOW_NOISE_EN bit has to be explicitly set to '0', since low-noise mode is not available at these ODRs.

In low-noise mode, the device is optimized to reduce the noise, while in low-current mode the device minimizes current consumption, as shown in Table 5.

Table 5. Low-pass filter bandwidth, noise and power consumption

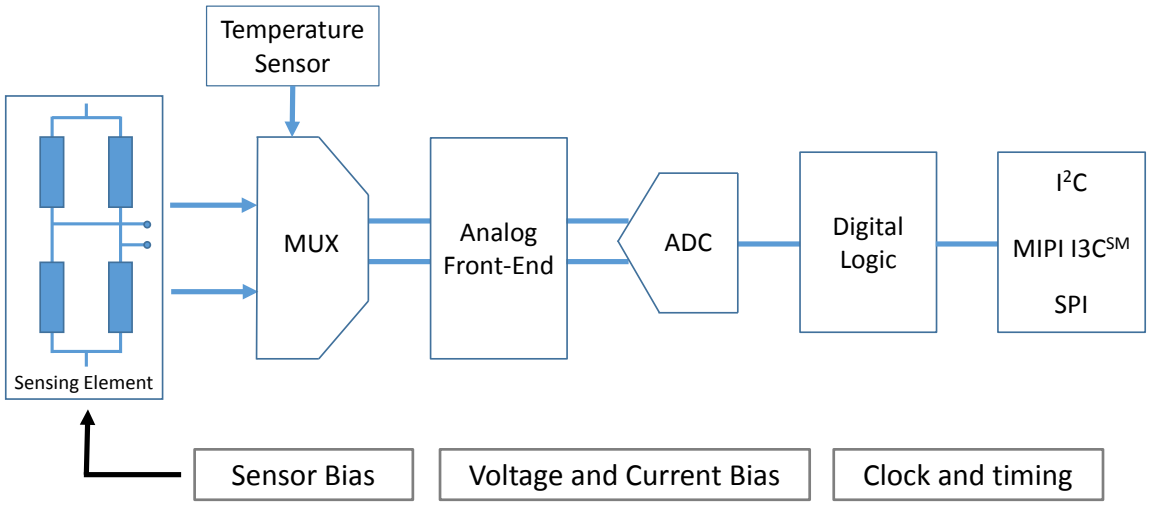
ODR [Hz]	Mode	LPF1 status	Device bandwidth	RMS noise [Pa] (Typical)	Current consumption [µA] (Typical)
1 Hz	Low noise	Disabled	ODR/2	1.7	12
		Enabled	ODR/9	0.9	12
		Enabled	ODR/20	0.6	12
	Low current	Disabled	ODR/2	4.5	4
		Enabled	ODR/9	2.6	4
		Enabled	ODR/20	1.7	4
10 Hz	Low noise	Disabled	ODR/2	1.7	107
		Enabled	ODR/9	0.9	107
		Enabled	ODR/20	0.6	107
	Low current	Disabled	ODR/2	4.5	35
		Enabled	ODR/9	2.6	35
		Enabled	ODR/20	1.7	35
25 Hz	Low noise	Disabled	ODR/2	1.7	265
		Enabled	ODR/9	0.9	265
		Enabled	ODR/20	0.6	265
	Low current	Disabled	ODR/2	4.5	85
		Enabled	ODR/9	2.6	85
		Enabled	ODR/20	1.7	85
50 Hz	Low noise	Disabled	ODR/2	1.7	530
		Enabled	ODR/9	0.9	530
		Enabled	ODR/20	0.6	530
	Low current	Disabled	ODR/2	4.5	170
		Enabled	ODR/9	2.6	170
		Enabled	ODR/20	1.7	170

ODR [Hz]	Mode	LPF1 status	Device bandwidth	RMS noise [Pa] (Typical)	Current consumption [µA] (Typical)
75 Hz	Low noise	Disabled	ODR/2	1.7	726
		Enabled	ODR/9	0.9	726
		Enabled	ODR/20	0.6	726
	Low current	Disabled	ODR/2	4.5	254
		Enabled	ODR/9	2.6	254
		Enabled	ODR/20	1.7	254
100 Hz	Low current	Disabled	ODR/2	4.5	338
		Enabled	ODR/9	2.6	338
		Enabled	ODR/20	1.7	338
200 Hz	Low current	Disabled	ODR/2	4.5	482
		Enabled	ODR/9	2.6	482
		Enabled	ODR/20	1.7	482

4 Sampling chain

The LPS27HHW is a piezo-resistive absolute pressure sensor that works as a digital output barometer. The device comprises a sensing element and an IC interface that communicates through selectable serial protocols SPI, I²C or MIPI I3C^{SM}, from the sensing element to the application.

Figure 4. LPS27HHW architecture block diagram



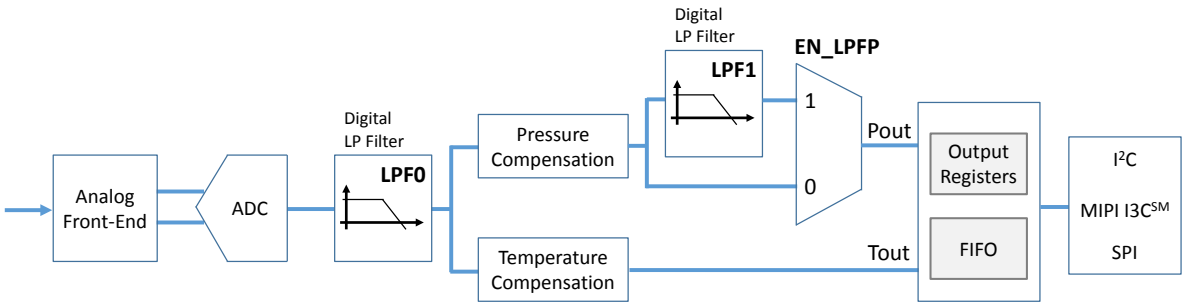
The sensing element, which detects absolute pressure, consists of a suspended membrane manufactured using a dedicated process developed by STMicroelectronics.

The following sections will focus on the "Digital Logic" block, describing the LPS27HHW filtering chain and the entire data path.

4.1 Digital low-pass filters

The LPS27HHW filtering chain embeds two digital low-pass filters. The first one is the LPF0 filter, which is applied to both temperature and pressure data. A second low-pass filter (LPF1) is also present. It can be applied optionally on the pressure readout path (temperature data are not filtered at this stage) when the device is in continuous mode. LPF1 affects data also when one-shot is used. Please note that LPF1 applies to pressure data only.

Figure 5. LPS27HHW digital LP filters block diagram



The LPF1 digital filter can be enabled by configuring the bit EN_LPFP, CTRL_REG1(@10h) register, and the overall device bandwidth can be configured acting on the bit LPFP_CFG, CTRL_REG1(@10h) register, as shown in Table 7. Setting EN_LPFP = 1 enables the filter and diverts its output to the pressure output registers and FIFO buffer. Setting EN_LPFP = 0 resets the filter, too; the LPF1 filter is also reset if the data rate (bit range ODR[2:0], CTRL_REG1(@10h)) or the filter bandwidth (bit LPFP_CFG, CTRL_REG1(@10h) register) is updated.

Table 6. Related registers and bit ranges of the LP filters

Register	Address	Bit	Bit range mask
CTRL_REG1	10h	**EN_LPFP**	00001000b = 08h
CTRL_REG1	10h	**LPFP_CFG**	00000100b = 04h

Table 7. Settings of the LP filters

EN_LPFP	LPFP_CFG	LPF1 filter status	Overall device bandwidth	Max overall settling time[1] (samples to be discarded)
0	x	Disabled, filter reset	ODR/2 (LPF0 only)	0 (first sample correct)
1	0	Enabled	ODR/9	2
1	1	Enabled	ODR/20	2

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

Table 7. Settings of the LP filters indicates the number of samples that should be discarded when the filter is enabled and/or after it's reset, before the filter reaches settling condition and output data can be considered meaningful.

4.2 Data path

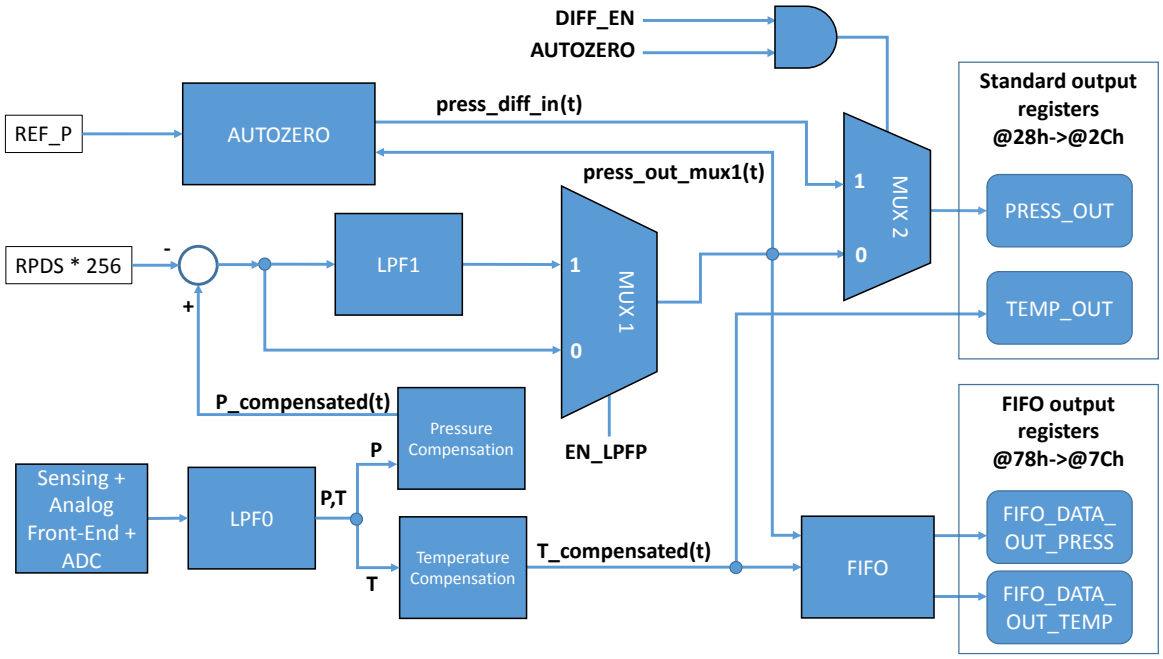
The following block diagram (Figure 6. Data path for output registers (standard and FIFO)) highlights the data path for pressure and temperature data from the sensing element and the analog-to-digital conversion to the standard output registers and FIFO under different operating/setting conditions for filters, FIFO and features like comparison with thresholds and references.

Details for the specific blocks can be found in the related paragraphs.

Important details are:

- Temperature output registers section, both for standard and FIFO, always receive *T_compensated(t)* signal data;
- FIFO buffer pressure section is always filled with signal *press_out_mux1(t)*;
- Standard output register pressure section always receives *press_out_mux1(t)*, except when the AUTOZERO feature is engaged;

Figure 6. Data path for output registers (standard and FIFO)



5 Reading output data

Once the device has been power supplied, it automatically downloads the calibration coefficients from the embedded flash to the internal registers. When the boot procedure completes, i.e. after 4.5 milliseconds (maximum) from power-on, the sensor automatically enters power-down mode.

To turn on the sensor and gather pressure and temperature data through the primary I²C / MIPI I3C^{SM} / SPI interface, it is necessary to select one of the operating modes through the ODR[2:0] bits in CTRL_REG1(@10h) register (continuous mode) or to set the ONESHOT bit to 1 in CTRL_REG2(@11h) (one-shot mode).

Data generated on board can be acquired by the controlling MCU by reads of the output registers.

Reading can be done:

- Asynchronously, polling the output registers at the proper pace;
- Synchronously, leveraging on the data-ready signal;
- Delayed, leveraging on the FIFO buffer, which allows improved power consumption (see Section 9 First-in, first-out (FIFO) buffer).

The basic startup sequence and the available options for data reading are described in the following paragraphs.

As a general rule, it is always recommended to read pressure and temperature samples from standard output (PRESS_OUT_x and TEMP_OUT_x) registers and from FIFO output registers (FIFO_DATA_OUT_PRESS_x and FIFO_DATA_OUT_TEMP_x) as well, starting from the lower address to the higher one, avoiding a different reading order.

5.1 Multi-read / write automatic address increment and rounding features

5.1.1 Automatic address increment feature

Available serial interfaces protocols allow executing single-read/write and multi-read/write register operations.

A single read/write operation is quite simple and requires specifying the register address that is going to be read/written.

A multi-read/write operation allows easily and effectively executing a repeated read/write operation during a single bus transaction. Once the user knows the starting register and the desired number N of repetitions, according to two different behaviors:

- Read N times the same register at the address provided;
- Read a sequence of N registers starting from the address provided;

The IF_ADD_INC bit in CTRL_REG2(@11h) allows switching between the two behaviors by automatically incrementing the read/write address.

Setting IF_ADD_INC = 1 (default) enables automatic address increments in multi-read and multi-write register operations. This allows executing faster and more effective bus transactions to read/write adjacent registers.

For example, executing a multi-read operation of N=5 bytes starting from register PRESS_OUT_XL(@28h) will result in reading with a single transaction all standard output registers: from PRESS_OUT_XL(@28h) to TEMP_OUT_H(@2Ch), without the need for explicitly managing each single address involved.

5.1.2 Address rounding feature

Address rounding is another helpful feature available in the LPS27HHW, related to automatic address increment and designed to facilitate burst readings of standard output and FIFO output samples. It is always enabled when IF_ADD_INC = 1, CTRL_REG2(@11h).

It allows effectively reading multiple times the output register ranges (both standard or FIFO) by executing one single multi-read operation of M*5bytes;

It is available for both output register ranges (5 bytes long):

- STANDARD_OUTPUT (PRESS_OUT_x, TEMP_OUT_x)(@28h-@2Ch);
- FIFO_OUTPUT(@78h-@7Ch);

Reading them starting from the initial register range to the final one 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.

For example, a multi-read of 10 bytes (M=2, 2*5 bytes), starting at @28h will result in reading 2 times the standard output range (from @28h to @2Ch), as for the following sequence:

@28h->@29h->@2Ah->@2Bh->@2Ch->@28h->@29h->@2Ah->@2Bh->@2Ch

This feature is effective in reading/emptying the FIFO buffer data.

5.2 Startup sequence

To turn on the device and gather pressure/temperature data, it is necessary to select one of the operating modes. The following general-purpose sequence can be used to configure the LPS27HHW device:

1. Write CTRL_REG1(@10h) = 3Eh // ODR = 25 Hz, BDU enabled, LPF (ODR/20)

5.3 Using the status register

The device is provided with a STATUS(@27h) register that should be polled to check when a new set of data (a pressure sample and a temperature sample) is available.

The P_DA bit is set to 1 whenever a new sample is available in the pressure output registers; the T_DA bit is set to 1 whenever a new sample is available in the temperature output registers.

The P_DA bit is cleared when the corresponding pressure sample is read (its most significant byte: PRESS_OUT_H(@2Ah).

The T_DA bit is cleared when the corresponding temperature sample is read (its most significant byte: TEMP_OUT_H(@2Ch).

The STATUS(@27h) register also includes the overrun flags: P_OR bit for pressure samples and T_OR bit for temperature samples. They are individually set to '1' when the corresponding sample is generated and the corresponding DA bit is already at 1, meaning the previous sample has been overwritten unread by the later generated new one, hence its value has been lost. The overrun bits are automatically cleared when all the data present inside the device have been read and new data have not been produced in the meantime.

The data-ready signals are represented by the P_DA and T_DA bits, STATUS(@27h) register.

Pressure and temperature data are synchronously generated; hence, bits P_DA and T_DA are synchronously rising at ‘1’ (unless one of the two isn’t already at ‘1’), but not synchronously resetting at ‘0’: it depends on when the respective data are read.

Reading the output registers before 1/ODR time period has expired allows acquiring the data and resetting P_DA and T_DA before an overwrite happens.

For the pressure sensor (for temperature sensor it is similar), the read operation of the output registers should be performed as follows:

1. Read STATUS(@27h);
2. If P_DA = 0, then go to 1;
3. Read PRESS_OUT_XL(@28h);
4. Read PRESS_OUT_L(@29h);
5. Read PRESS_OUT_H(@2Ah);
6. Data processing
7. Go to 1.

If the device is configured in one-shot mode instead of continuous mode, the routine will be stuck at step 1 after one execution, since the device performs a single measurement, sets the P_DA/T_DA bit high and returns to power-down mode. Please note that the ONE_SHOT bit is self-cleared when the device returns to power-down mode. It is possible to trigger another one-shot reading by setting the ONE_SHOT bit to 1 again.

5.4 Using the data-ready signal

The device can be configured to have one HW signal to determine when a new set of measurement data is available to be read and trigger synchronous actions, i.e. read output registers as soon as data are available.

The P_DA signal can be driven to the INT_DRDY pin by setting bit DRDY=1, CTRL_REG3(@12h) register and the bit range INT_S[1:0]=00, CTRL_REG3(@12h) register.

The P_DA pressure data-ready signal resets reading PRESS_OUT_H(@2Ah).

The P_DA pressure data-ready signal is sent to the INT_DRDY pin by an OR logic together with the FIFO interrupt signals, hence it can be required, when INT_DRDY asserts, to read the appropriate set of status registers in range(@24h->@27h), i.e. FIFO_STATUS(@26h) and STATUS(@27h), to identify the occurred event (see Section 8.2 Diverting interrupt events to the INT_DRDY pin).

5.5 Using the block data update (BDU) feature

If reading the data is particularly slow and cannot be synchronized (or it is not required to be) with either the P_DA/T_DA event bit in the STATUS(@27h) register or with the data-ready signal (which can be diverted to the INT_DRDY pin), it is strongly recommended to set the BDU (block data update) bit to 1 in the CTRL_REG1(@10h) register.

This feature avoids reading values (XL, L and H parts of output data) related to different samples. In particular, when the BDU is activated, the output data registers always contain the most recent output data produced by the device, but, in case the read of a given part (e.g. pressure, starting from PRESS_OUT_XL(@28h)) is initiated, the refresh of the remaining bytes for that part (pressure), remains blocked until all XL, L and H parts of the data are read.

The same happens for the temperature section. If TEMP_OUT_L(@2Bh) is read, TEMP_OUT_H(@2Ch) content won't update until read.

BDU applies to both pressure data and temperature data, but manages them separately. When the BDU feature is enabled, pressure and temperature data are separately refreshed depending on when PRESS_OUT_H(@2Ah) / TEMP_OUT_H(@2Ch) is read.

Note: To guarantee the correct behavior of the BDU feature, PRESS_OUT_H(@2Ah) / TEMP_OUT_H(@2Ch), must be the last address read.

The BDU feature also acts on the FIFO_STATUSx(@25h, @26h) registers. When the BDU bit is set to 1, it is mandatory to read FIFO_STATUS1 first and then FIFO_STATUS2.

5.6 Understanding output data

5.6.1 Pressure data

The measured pressure data are sent to the PRESS_OUT_XL(@28h), PRESS_OUT_L(@29h), PRESS_OUT_H(@2Ah) registers. When DIFF_EN = ‘0’, AUTOZERO=‘0’, AUTOREFP= ‘0’, these registers contain, respectively, the least significant byte, the middle significant byte and the most significant byte of the pressure data.

The complete pressure data is given by the concatenation PRESS_OUT_H & PRESS_OUT_L & PRESS_OUT_XL and it is expressed as a binary number.

Pressure data is represented as 24-digit signed 2's complement binary numbers, (each digit referred to as LSB).

To translate the digital representation to its corresponding real number with its SI unit (Pa for pressure), a sensitivity parameter must be applied.

Each pressure sample must be divided by the proper sensitivity parameter (please refer to the datasheet) in order to obtain the corresponding value in hPa:

Psens = 4096 [LSB/hPa]

5.6.2 Example of pressure data

Hereafter is a simple example of how to get the pressure LSB data and transform it into hPa.

1. Get raw data from the sensor:
   - PRESS_OUT_XL(@28h): 1Ah
   - PRESS_OUT_L(@29h): 84h
   - PRESS_OUT_H(@2Ah): 3Eh
2. Do registers concatenation:
   - PRESS_OUT_H & PRESS_OUT_L & PRESS_OUT_XL: 3E841Ah
3. Calculate signed decimal value (from signed 2's complement 24-digit binary format):
   - P[LSB]: +4097050d
4. Apply Psens sensitivity:
   - P[hPa] = +4097050 / 4096 = +1000.2563

5.6.3 Temperature data

The measured temperature data are sent to the TEMP_OUT_L(@2Bh), TEMP_OUT_H(@2Ch) registers. These registers contain, respectively, the least significant byte and the most significant byte of the temperature data.

The complete temperature data is given by the concatenation of TEMP_OUT_H & TEMP_OUT_L registers and it is expressed as binary signed number by two's complement representation.

Temperature data is represented as 16-bit signed binary number, each digit referred to as LSB.

To translate the digital representation to its corresponding real number with its SI unit (°C, Celsius degrees, for temperature), a sensitivity parameter for temperature must be applied.

Each temperature sample must be divided by the proper sensitivity parameter (please refer to the datasheet) in order to obtain the corresponding value in °C:

Tsens = 100 [LSB/°C]

5.6.4 Example of temperature data

Hereafter is a simple example of how to get the temperature LSB data and transform it into °C.

1. Get raw data from the sensor:
   - TEMP_OUT_L(@2Bh): 7Bh
   - TEMP_OUT_H(@2Ch): FEh
2. Do register concatenation:
   - TEMP_OUT_H & TEMP_OUT_L: FE7Bh
3. Calculate signed decimal value (from 16-bit signed represented by two's complement format):
   - T[LSB]: -389d
4. Apply Tsens sensitivity:
   - T[°C] = -389 / 100 = -3.89

6 Reboot and software reset

After the device is powered up, the LPS27HHW performs a 4.5 ms (maximum) boot procedure to load the trimming parameters. After the boot is completed, the device is automatically configured in power-down mode.

During the boot time the registers are not accessible. Anyway, in order to check when the boot procedure is completed, the user can read the BOOT_ON bit of register INT_SOURCE (@24h). If this bit is equal to 1, the boot is running, when it goes to 0, the boot is ended.

After power-up, when the BOOT bit of the CTRL_REG2(@11h) register is set to 1, the trimming parameters are reloaded and the registers RPDS_L(@18h) and RPDS_H(@19h) are reset to 0.

No toggle of the device power lines is required. After the reboot is completed, the device enters in power-down mode (regardless of the selected operating mode) and the BOOT bit is self-cleared to '0'. The BOOT_ON bit described above can be used to check when the reboot procedure has ended.

If the reset to the default value of the device registers is required, it can be performed by setting the SWRESET bit of the CTRL_REG2(@11h) register to 1. When this bit is set to 1, the following registers are reset to their default value:

- INTERRUPT_CFG(@0Bh);
- THS_P_L(@0Ch);
- THS_P_H(@0Dh);
- IF_CTRL(@0Eh);
- CTRL_REG1(@10h);
- CTRL_REG2(@11h);
- CTRL_REG3(@12h);
- FIFO_CTRL(@13h);
- FIFO_WTM(@14h);
- INT_SOURCE(@24h);
- FIFO_STATUS1(@25h);
- FIFO_STATUS2(@26h);
- STATUS(@27h).

The software reset procedure can take a few tens of µs; the status of the reset is signaled by the status of the SWRESET bit of the CTRL_REG2(@11h) register. Once the reset is completed, this bit is automatically set low.

In order to avoid conflicts, the reboot and the software reset must not be executed at the same time (do not set to 1 at the same time both the BOOT bit and SWRESET bit of the CTRL_REG2(@11h) register).

The flow must be performed serially as shown in the example below:

1. Set the BOOT bit of the CTRL_REG2(@11h) register to 1;
2. Wait 4.5 ms (or wait until the BOOT_ON bit of the INT_SOURCE (@24h) register returns to 0);
3. Set the SWRESET bit of the CTRL_REG2(@11h) register to 1;
4. Wait 50 µs (or wait until the SWRESET bit of the CTRL_REG2(@11h) register returns to 0).

7 Offset compensation (OPC - one-point calibration)

If, after the soldering of the component, a residual pressure offset is still present, it can be removed with a one-point calibration (OPC), leveraging on the RPDS registers which can store the residual offset value to be removed.

The calibration offset value is expected to be stored as a signed 16-bit value expressed as 2's complement in the RPDS_L(@18h) and RPDS_H(@19h) registers.

The default value for RPDS is 0 (zero).

The content of the RPDS registers is always automatically subtracted from the compensated pressure output and provided to the standard output pressure registers PRESS_OUT_x(@28h, 29h and 2Ah) and FIFO. It is provided as the difference between the measured pressure and the content of the RPDS registers (@18h, @19h) multiplied by 256, when DIFF_EN = '0', AUTOZERO='0', AUTOREFP= '0' (see Section 4.2 Data path).

8 Interrupt modes

The LPS27HHW has a built-in configurable interrupt generator block, which allows generating interrupt events based on pressure samples and comparisons to customizable references and thresholds.

Interrupt event signals can be selected and enabled; they are made available at a dedicated status register (INT_SOURCE(@24h)) to monitor them.

Additionally, other signals are generated by the FIFO buffer subsystem which can be used as event signals as well and monitored through the FIFO_STATUS1(@25h) and FIFO_STATUS2(@26h) registers (see Section 9.2.2 Monitoring the FIFO buffer status).

All the above interrupt signals can be singularly selected by a controlling register, CTRL_REG3(@12h), for their diversion to the interrupt pin INT_DRDY.

The interrupt generator block control is managed through the INTERRUPT_CFG(@0Bh) register.

The interrupt events related to pressure and temperature sampling are the following:

- Data-ready: new data available;
- Threshold-based.

The events generated by the FIFO buffer are the following:

- FIFO watermark;
- FIFO full;
- FIFO overrun.

8.1 Interrupt events related to pressure and temperature sampling

8.1.1 Data-ready

If data generation is enabled, it is possible to identify when a new pressure or new temperature data value has been generated and is ready to be read by monitoring the STATUS(@27) register bits.

Every time a new pressure data value is ready, the bit P_DA, STATUS(@27h) register, is set to ‘1’.

Every time a new temperature data value is generated, the bit T_DA in STATUS(@27h) register is set to ‘1’.

Pressure and temperature data are synchronously generated, see Section 5.3 Using the status register.

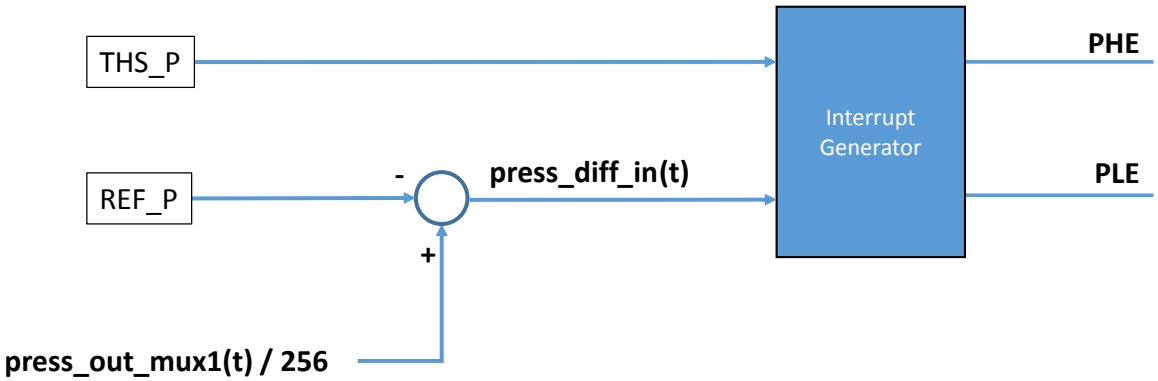
The content of P_DA can be diverted to the DRDY_INT pin. Diversion is enabled by bit DRDY, CTRL_REG3(@12h), together with the multiplexing control bit range INT_S[1:0]=’00’, CTRL_REG3(@12h) (see Section 8.2 Diverting interrupt events to the INT_DRDY pin).

8.1.2 Threshold-based differential interrupt

The interrupt generator in LPS27HHW allows generating an interrupt event based on a *user-defined pressure threshold value*, THS_P, which can be stored in two dedicated registers THS_P_L(@0Ch) and THS_P_H(@0Dh).

The interrupt generator generates interrupt signals based on a comparison executed on the differential signal *press_diff_in* and the *user-defined pressure threshold* stored in THS_P, as illustrated in Figure 7. Interrupt generator.

Figure 7. Interrupt generator



Signal *press_diff_in* is the difference between compensated pressure samples taken at MUX1 output (*press_out_mux1*, see Figure 6) and an instantaneous sample of the same signal taken at enabling time and stored in dedicated registers REF_P_L (@15h) and REF_P_H (@16h).

The instantaneous sample is automatically stored in read-only REF_P registers, REF_P_L(@15h) and REF_P_H(@16h), when one of the two differential modes (AUTOZERO mode or AUTOREFP mode, they will be described later) is engaged.

Only the compensated pressure signal 16 most significant bits are stored in REF_P and used to obtain differential signal *press_diff_in*.

press_diff_in(t) = press_out_mux1(t)/256 – REF_P

with *REF_P = press_out_mux1(t=Differential Mode_Engaging_Time) / 256.*

Note: Differential Mode_Engaging_Time is the instant when AUTOZERO mode or AUTOREFP mode are engaged.

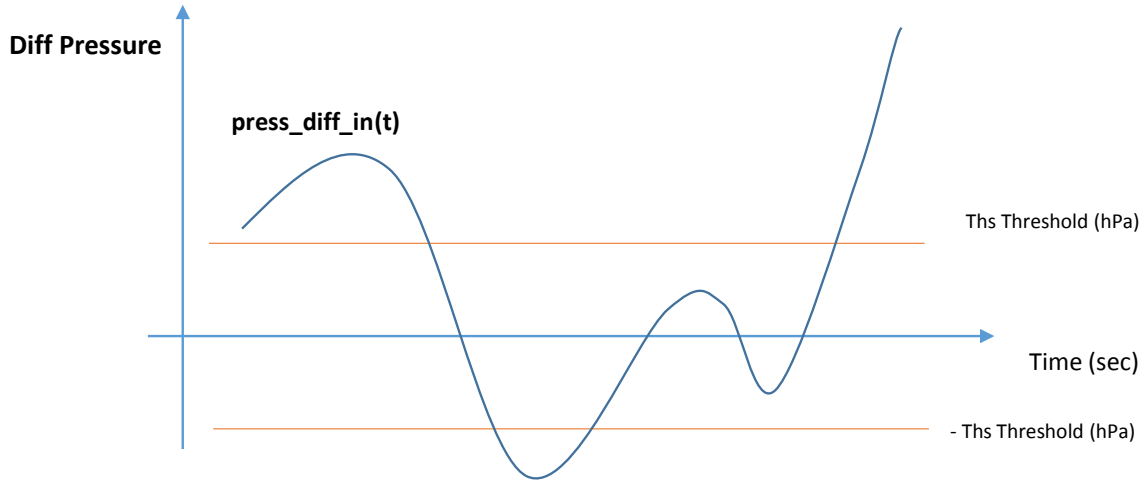
Basically, thresholds (THS_P) and references (REF_P), in combination with *press_out_mux1(t)*, are used for comparisons: it is important to note that THS_P is a differential pressure threshold, not an absolute pressure threshold. Thresholds and references have to be considered as the 2 most significant bytes with respect to the standard pressure samples which are 24-bit (or 3 bytes) long.

When enabled, the device executes a comparison between press_diff_in samples and THS_P at each new sample arrival (1/ODR period). In this way, user-defined pressure threshold (THS_P), identifies three areas in which each press_diff_in sample can fall:

- above +THS_P,
- below -THS_P
- between -THS_P and +THS_P.

The interrupt generator will allow selecting the generation of two different interrupt signals (PLE and PHE) which rise when the *press_diff_in(t)* signal falls below –THS_P and/or above +THS_P areas (see Figure 8).

Figure 8. Differential interrupt input signal and threshold relationship



The desired interrupt threshold quantity used for pressure interrupt generation, expressed as hPa, has to be converted to binary as a 15-bit unsigned right-justified notation and must be stored in THS_P_H(@0Dh) for MSByte and THS_P_L(@0Ch) for LSByte.

The binary unsigned integer to be stored in THS_P can be computed starting from its physical value, expressed as hPa as follows:

THS_P [LSB] = abs (interrupt pressure threshold [hPa]) * 4096 [LSB/hPa] / 256 = abs (interrupt pressure threshold [hPa]) * 16

Example:

Suppose desired interrupt pressure threshold = 10 hPa;

THS_P = 10d * 16 = 160d = 00A0h

values to store in THS_P are:

THS_P_H = 00h; THS_P_L = A0h

Differential interrupt can be used by selecting between two different modes:

- AUTOREFP;
- AUTOZERO;

Both of them result in:

- storage in REF_P of current sample of *press_out_mux1* at engaging time. As for above description.
- enabling interrupt generation for above positive threshold and/or below negative threshold.

The output available in the standard output registers will be different according to the two different modes (see Section 4.2 Data path):

- AUTOZERO: standard output registers will report the differential *press_diff_in(t)* signal;
- AUTOREFP: standard output registers will keep reporting the usual output (as for all other configurations *press_out_mux1(t)* signal);

To enable the above interrupt modes, the bit DIFF_EN, INTERRUPT_CFG(@0Bh) register must be set to ‘1’ and the desired threshold values must be stored in the THS_P_L(@0Ch) and THS_P_H(@0Dh) registers.

Furthermore, the PHE bit or PLE bit (or both of them) in INTERRUPT_CFG(@0Bh) have to be set to 1 to enable, respectively, the interrupt generation on the positive or negative events.

Finally, according to the AUTOZERO or AUTOREFP choice, bit AUTOREFP and AUTOZERO need to be set to ‘1’ (see other details in Section 8.1.2.1 AUTOZERO mode and Section 8.1.2.2 AUTOREFP mode).

8.1.2.1 *AUTOZERO mode*

AUTOZERO mode when engaged:

- instantly triggers the storage of current pressure measurements in dedicated registers (REF_P);
- reports to standard output registers, pressure part, (@28h->@2Ah) the current pressure decreased by the REF_P stored sample (*press_diff_in(t)* signal);
- detects when pressure variations with respect to REF_P are over the user-defined differential threshold (±THS_P) and generates separate interrupt signals (PLE, PHE) accordingly.

At the instant it is engaged (t = t_AUTOZERO), the current measured pressure sample is stored in the REF_P(@15h, @16h) registers and used as the pressure reference. From the engaging time on, the output pressure registers PRESS_OUT(@28h, @29h and @2Ah) are filled with:

PRESS_OUT(t) = *press_out_mux1*(t)/256 – REF_P

Where:

REF_P = *press_out_mux1*(t = t_AUTOZERO)/256

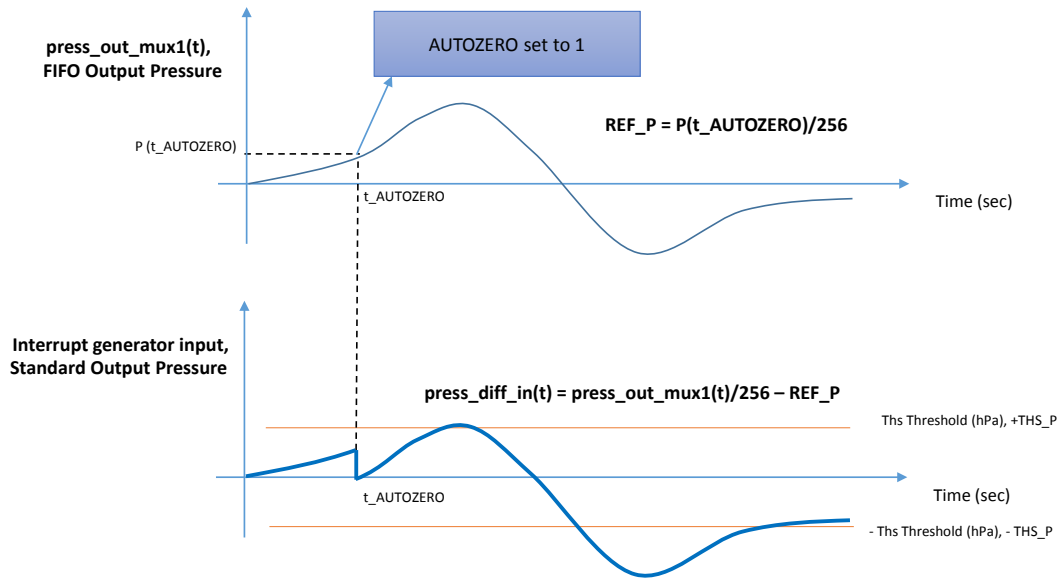
The same value is diverted to the interrupt generator and used as input for the interrupt generation:

press_diff_in(t) = press_out_mux1(t)/256 – REF_P

Hence, what is available in the standard output registers (PRESS_OUT) for pressure is a differential pressure, the same used for interrupt generation.

In case the differential pressure value reported in the standard output registers (PRESS_OUT[LSB]) has to be converted to the corresponding value in hPa, it can be done by dividing the PRESS_OUT[LSB] by a factor equal to 16 [LSB/hPa].

Figure 9. Differential interrupt AUTOZERO mode, outputs and thresholds



To enable the AUTOZERO differential interrupt feature, bits AUTOZERO, DIFF_EN, PLE and/or PHE, need to be set to ‘1’.

After the first conversion, the AUTOZERO bit is automatically set back to ‘0’, but the Autozero mode remains engaged. To disable the engaged Autozero feature and return back to normal mode, an explicit action is required: the bit RESET_AZ, INTERRUPT_CFG(@0Bh) register has to be set to ‘1’. The RESET_AZ bit, autonomously sets back to ‘0’, too.

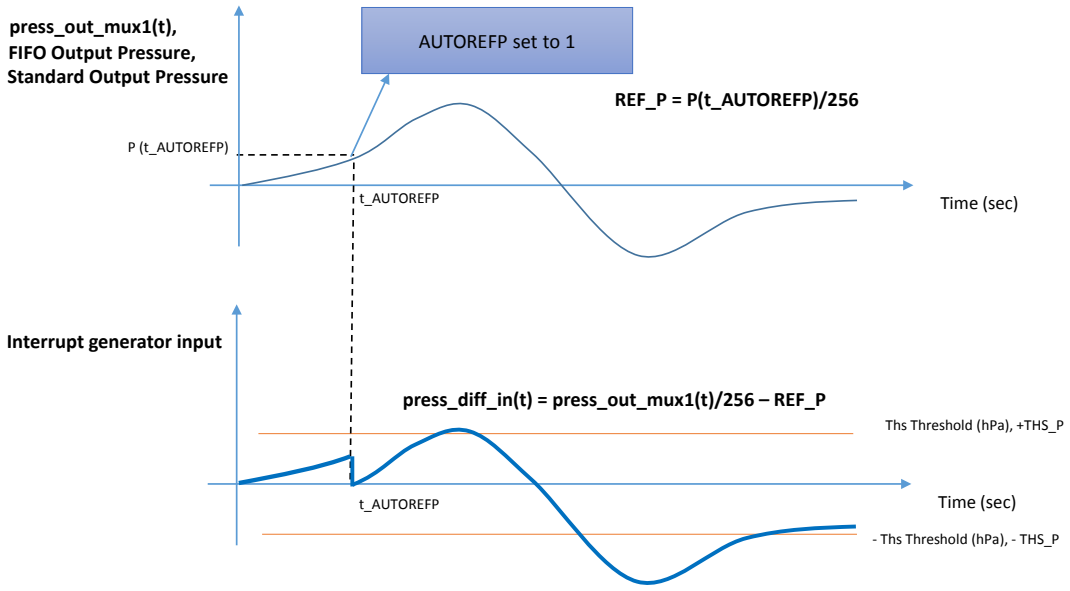
The Autozero feature disengagement effects are: resets REF_P to default value 0 (zero), content of standard output registers switches back to default signal: PRESS_OUT(t) = *press_out_mux1*(t).

8.1.2.2 *AUTOREFP mode*

The AUTOREFP mode, when engaged:

- instantly triggers the storage of the current pressure measurements in dedicated registers (REF_P);
- reports to standard output registers, pressure part, (@28h->@2Ah) the *press_out_mux1(t)* signal;
- detects when pressure variations with respect to REF_P are over the user-defined threshold (+/-THS_P) and generates separate interrupt signals (PLE, PHE) accordingly.

Figure 10. Differential interrupt AUTOREFP mode, outputs and thresholds



At the instant it is engaged (t = t_AUTOREFP), the current measured pressure sample is stored in the REF_P(@15h, @16h) registers and used as the pressure reference. From the engaging time on, the output pressure registers PRESS_OUT(@28h, @29h and @2Ah) keep being filled, as usual, with:

PRESS_OUT(t) = *press_out_mux1*(t)

The signal used as input for the interrupt generation is:

press_diff_in(t) = *press_out_mux1*(t)/256 – REF_P

Where:

REF_P = *press_out_mux1*(t = t_AUTOREFP)/256

Hence, what is available in the standard output registers (PRESS_OUT) for pressure is the usual pressure signal: this is different with respect to what happens for AUTOZERO mode.

To enable the AUTOREFP differential interrupt feature, bits AUTOREFP, DIFF_EN, PLE and/or PHE, need to be set to ‘1’.

After the first conversion, the AUTOREFP bit is automatically set back to ‘0’, but the AUTOREFP mode remains engaged. To disable the engaged AUTOREFP feature and return back to normal mode, an explicit action is required: the bit RESET_ARP, INTERRUPT_CFG(@0Bh) register has to be set to ‘1’. The RESET_ARP bit, autonomously sets back to ‘0’, too.

The AUTOREFP feature disengagement effect is: resets REF_P to default value 0 (zero).

8.1.2.3 *Interrupt latching*

Threshold-based differential interrupts offer the option to latch the content of the INT_SOURCE(@24h) register. This option is controlled through bit LIR, INTERRUPT_CFG(@0Bh).

Latching has the effect to keep the content of the register bits “frozen”, even if the condition which triggered their rise is no longer valid, until the INT_SOURCE register has been read.

If latched, when the IA bit rises, the latch feature keeps it at ‘1’ and “freezes” the PL and PH content, too, until the INT_SOURCE register is read.

When latching is not enabled, the IA value and PL and PH values as well keep updating every 1/ODR time period according to *press_diff_int(t)* and THS_P. Latch behavior propagates to the INT_DRDY pin if the IA, or PL, or PH signals are diverted to it.

Figure 11. Latch disabled (LIR = 0): interrupt behavior

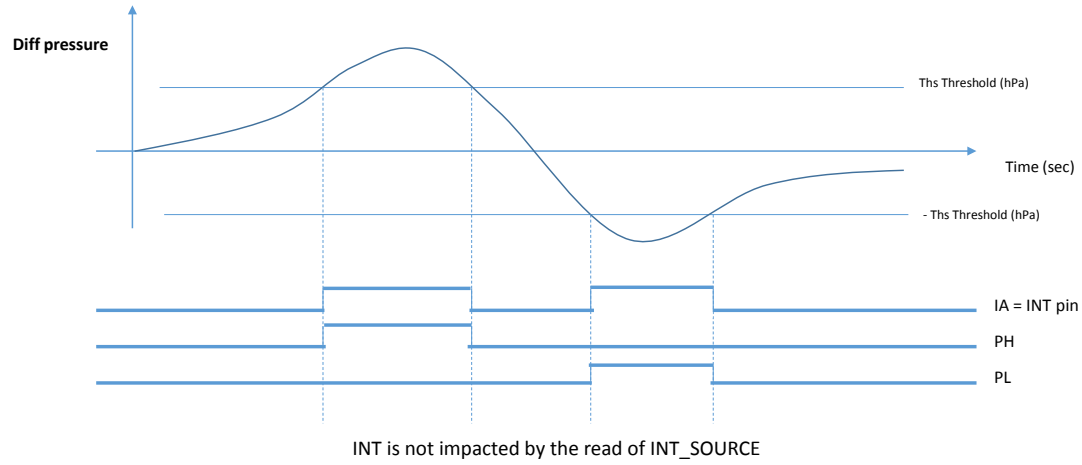
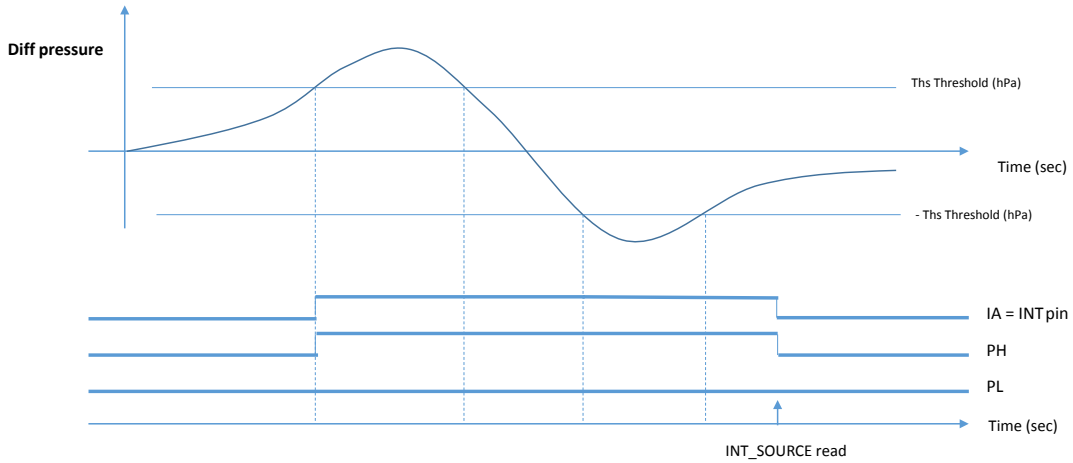


Figure 12. Latch enabled (LIR = 1): interrupt behavior



8.1.3 Interrupt events related to FIFO status

With the LPS27HHW pressure sensor, when FIFO is running, it generates, according to its selected operating mode, a set of event signals which allow monitoring its status and which are available in the FIFO_STATUS2(@26h) register (see Section 9 First-in, first-out (FIFO) buffer).

It is possible to select the following events for diversion to the INT_DRDY pin by properly configuring the CTRL_REG3(@12h) register (see Section 8.2 Diverting interrupt events to the INT_DRDY pin):

- FIFO full condition: INT_F_FULL set to '1';
- FIFO watermark level reached: INT_F_WTM set to '1';
- FIFO overrun: INT_F_OVR set to '1'.

8.1.4 Interrupt events for FIFO triggered modes

It is possible to leverage on differential interrupt events related to pressure sample values (see Section 8.1.2 Threshold-based differential interrupt) to trigger transitions in FIFO triggered modes.

For the following FIFO buffer modes:

- Continuous (Dynamic-Stream)-to-FIFO mode,
- Bypass-to-Continuous (Dynamic-Stream) mode,
- Bypass-to-FIFO mode,

the bit IA, INT_SOURCE(@24h) register, with: IA = (PL or PH) , INT_SOURCE(@24h) is used as an internal trigger event signal to drive the switch from one FIFO behavior to the following one according to the selected mode (see Section 9.3.2 Triggered FIFO modes).

8.2 Diverting interrupt events to the INT_DRDY pin

The INT_DRDY output pin allows selecting “Push-Pull” or “Open-Drain” configuration by acting on bit PP_OD, CTRL_REG2(@11h). The default is 0: "Push-Pull".

The interrupt electrical signal logic on INT_DRDY, not the status register logic, can be selected acting on bit INT_H_L, CTRL_REG2(@11h). The default is 0: "Active-High", meaning INT_DRDY at voltage level “High” when the interrupt is asserted.

The following interrupt events are generated when the related device feature is running:

1. Data generation (pressure data-ready);
2. FIFO status;
3. Threshold-based differential interrupt;

and are available in the following dedicated status registers:

1. STATUS(@27h);
2. FIFO_STATUS2(@26h);
3. INT_SOURCE(@24h);

They can be singularly enabled for being diverted to the INT_DRY pin.

The interrupt events are multiplexed to INT_DRDY and OR-ed according to the architecture displayed in Figure 13. Diverting interrupt events to the INT_DRDY pin depending on the settings for the multiplexer MUX controlled by bit range INT_S[1:0], CTRL_REG3(@12h) register.

Signal selection can be done through dedicated bits in the CTRL_REG3(@12h) register.

Divertible signals and relative diversion to the INT_DRDY enabling bit are listed in the following table.

Table 8. Settings for INT_DRDY configuration

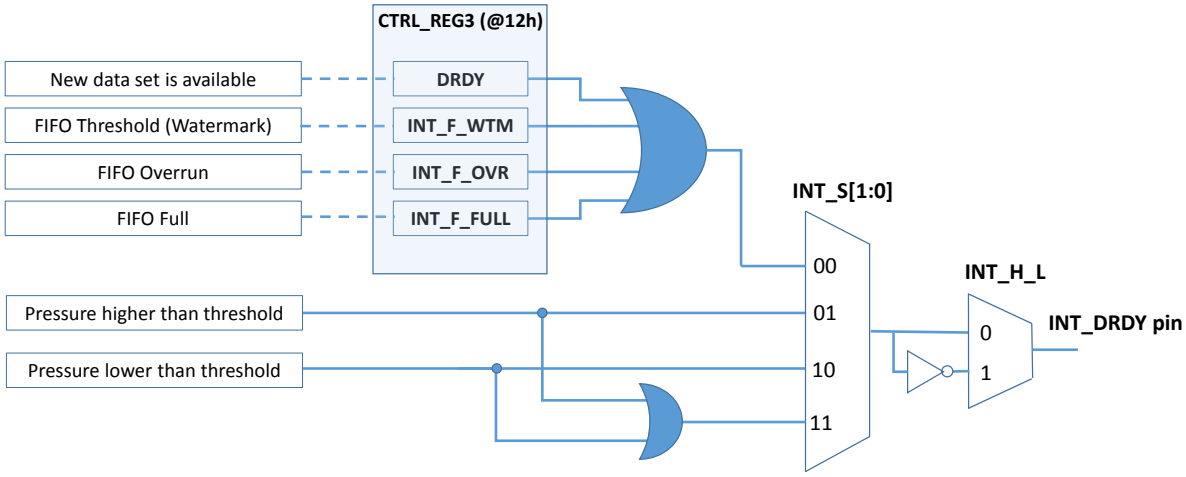
Interrupt event signal	Diversion to INT_DRDY pin, enabled by	Event
P_DA, STATUS(@27h)	DRDY and INT_S[1:0], CTRL_REG3(@12h)	Pressure data ready
FIFO_FULL_IA, FIFO_STATUS2(@26h)	INT_F_FULL and INT_S[1:0], CTRL_REG3(@12h)	FIFO full
FIFO_OVR_IA, FIFO_STATUS2(@26h);	INT_F_OVR and INT_S[1:0], CTRL_REG3(@12h)	FIFO Overrun
FIFO_WTM_IA, FIFO_STATUS2(@26h)	INT_F_WTM and INT_S[1:0], CTRL_REG3(@12h)	FIFO watermark
IA, INT_SOURCE(@24h)	(PLE OR PHE), INTERRUPT_CFG(@0Bh) and INT_S[1:0], CTRL_REG3(@12h)	Any threshold on differential interrupt
PL, INT_SOURCE(@24h)	PLE, INTERRUPT_CFG(@0Bh) AND INT_S[1:0], CTRL_REG3(@12h)	Negative threshold
PH, INT_SOURCE(@24h)	PHE, INTERRUPT_CFG(@0Bh) AND INT_S[1:0], CTRL_REG3(@12h)	Positive threshold

Signal results to be separated: selecting INT_S[1:0]=’00’ allows receiving data-ready and FIFO status events; other INT_S[1:0] settings allow selecting threshold-based differential interrupt signals, as given in the following table.

Table 9. Multiplexing interrupt events on the INT_DRDY pin

INT_S1	INT_S0	INT_DRDY pin multiplexing configuration
0	0	Data-ready signal / FIFO status events
0	1	Pressure high (P_high)
1	0	Pressure low (P_low)
1	1	Pressure low OR high

Figure 13. Diverting interrupt events to the INT_DRDY pin



Every time an interrupt event is caught on INT_DRDY, according to the current multiplexing and enabled signals, the corresponding set of status registers has to be read to detect what event has reached the pin and act accordingly.

Threshold-based differential interrupts can be latched INT_SOURCE(@24h), hence in this case in order to de-assert the interrupt signal, reading the INT_SOURCE(@24h) register would be mandatory.

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

The LPS27HHW provides an embedded 128-slot deep, first-in, first-out (FIFO) buffer to store pressure and temperature data. The FIFO buffer allows limiting intervention by the host processor and facilitates post-processing data for event or pattern detection and analysis. FIFO usage allows remarkable power saving for the system: the host CPU can wake up only when notified by FIFO events, burst the significant data out from the FIFO, return to sleep. The FIFO buffer can work according to six different selectable modes that guarantee a high-level of flexibility during application development:

- Bypass mode
- FIFO mode
- Continuous (Dynamic-Stream) mode
- Bypass-to-FIFO mode
- Bypass-to-Continuous (Dynamic-Stream) mode
- Continuous (Dynamic-Stream)-to-FIFO mode

The device interrupt generator monitors the FIFO operating parameters which are used for generating FIFO-related interrupt signals that are available in dedicated registers. FIFO-related interrupt signals, in turn, are divertible to an interrupt dedicated output pin (INT_DRDY) for usage connected to CPU input ports.

9.1 FIFO description

The FIFO buffer is able to store up to 128 “data sample sets”. Each “data sample set” (dss, from now on) is sized at 5 bytes and is built on a couple of pressure (24-bit) and temperature (16-bit) synchronous samples (24-bit + 16-bit = 5 bytes), sampled at the same time. The dss consists of 5 bytes and is the logic atomic information for FIFO.

When FIFO is enabled, the samples from the pressure and temperature sensors are also diverted to feed the FIFO buffer at the currently selected output data rate (ODR). The data path for feeding standard output registers and FIFO buffer is common. The dss feeding FIFO are always the data coming out from the LPF blocks (see Section 4.2 Data path). While FIFO is fed, the current dss data remain available in the standard output registers.

Usually dss available in the standard output registers PRESS_OUT_XL(@28h), PRESS_OUT_L(@29h), PRESS_OUT_H(@2Ah), TEMP_OUT_L(@2Bh), TEMP_OUT_H(@2Ch), are the same data sets filling the FIFO buffer (but when interrupt AUTOZERO is enabled, AUTOZERO=1 and DIFF_EN=1, in this case what is available in the standard outputs is different from what is sent to FIFO).

The number of FIFO slots can be limited according to user needs by enabling the "Stop on Watermark level" feature which allows setting a custom level of fullness lower than the allowed maximum of 128 dss.

FIFO behaves as a resizable circular buffer: each new dss is placed in the first available empty FIFO slot until the buffer reaches fullness level. Once the fullness level has been reached, the oldest value is overwritten or FIFO stops filling according to the selected operating mode. Reading a dss from FIFO_OUTPUT registers removes it from the FIFO buffer. Each time FIFO is fed by a new dss or a dss is read, FIFO and its status registers are updated accordingly.

9.2 FIFO settings and control

Upon device power-up, FIFO is not enabled: the pressure and temperature data are not stored in the FIFO, but stored in the standard output pressure and temperature registers only. The FIFO can be configured and controlled by setting two registers:

- FIFO_CTRL(@13h): for FIFO mode and FIFO trigger mode selection and enabling FIFO depth limit (watermark);
- FIFO_WTM(@14h): for setting the FIFO watermark level.

The FIFO buffer status can be monitored by:

- FIFO_STATUS1(@25h): for reading the FIFO stored data level during operation;
- FIFO_STATUS2(@26h): for reading the FIFO status during operation.

An extra register controlling FIFO-related features is CTRL_REG3(@12h) which is used for selecting FIFO_STATUS register bit signals to be diverted to the INT_DRDY pin (see Section 8.2 Diverting interrupt events to the INT_DRDY pin).

Table 10. List of registers involved in managing FIFO

FIFO control registers			
Register	**@address**	**Bit range**	**Bit range mask**
FIFO_CTRL	13h	**F_MODE[1:0]**	00000011b = 03h
FIFO_CTRL	13h	**TRIG_MODES**	00000100b = 04h
FIFO_CTRL	13h	**STOP_ON_WTM**	00001000b = 08h
FIFO_WTM	14h	**WTM[6:0]**	01111111b = 7Fh
CTRL_REG3	12h	**INT_F_OVR**	00001000b = 08h
CTRL_REG3	12h	**INT_F_WTM**	00010000b = 10h
CTRL_REG3	12h	**INT_F_FULL**	00100000b = 20h

Table 11. FIFO modes and FIFO trigger mode selection

TRIG_MODES	F_MODE1	F_MODE0	FIFO mode selection
x	0	0	Bypass mode: turn off and reset FIFO
0	0	1	FIFO mode
0	1	x	Continuous (Dynamic-Stream) mode
1	0	1	Bypass-to-FIFO mode
1	1	0	Bypass-to-Continuous (Dynamic-Stream) mode
1	1	1	Continuous (Dynamic-Stream)-to-FIFO mode

9.2.1 Limiting FIFO depth: stop-on-watermark level

The full FIFO buffer can store up to max. depth = 128 depth levels of dss, or slots.

The FIFO depth can be logically sized, limited by the stop-on-watermark feature. When enabled, it logically resizes FIFO, or the number of its available slots, reducing its depth. It works by defining a watermark level and enabling it.

Defining the required FIFO depth limit is performed by setting bit range WTM[6:0], FIFO_WTM(@14h) register, its maximum value is 127d = 7Fh.

When the stop-on-watermark feature is enabled, FIFO depth level will then be equal to the value stored in bit range WTM[6:0]. The watermark limit is enabled by setting bit STOP_ON_WTM= ‘1’, FIFO_CTRL(@13h) register.

The watermark level needs to be set and enabled before enabling FIFO; it cannot be changed while FIFO is already running.

Table 12. **Watermark settings**

Register	@address	Bit range	Bit range mask
FIFO_CTRL	13h	**STOP_ON_WTM**	00001000b = 08h
FIFO_WTM	14h	**WTM[6:0]**	01111111b = 7Fh

9.2.2 Monitoring the FIFO buffer status

The FIFO buffer status can be monitored by reading the dedicated registers. FIFO_STATUSx(@25h, @26h) are read-only registers that allow monitoring the current FIFO status.

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.

The FSS[7:0] bit range, FIFO_STATUS1(@25h) register provides information about the current number of data set samples (dss) stored in the FIFO buffer, hence:

- FSS is equal to ‘00000001b' when 1 data set is stored in the FIFO;
- FSS is equal to ‘10000000b’ when 128 data sets are stored in the FIFO.

The FIFO_WTM_IA bit, FIFO_STATUS2(@26h) register is a watermark monitor and is asserted while the current FIFO buffer filling level is equal to or higher than the level defined by WTM[6:0], FIFO_WTM(@14h). The FIFO current filling level, is available in FSS[7:0], FIFO_STATUS1(@25h). The monitor works even if the stop-on-watermark feature is not enabled (STOP_ON_WTM=’0’).

The FIFO_WTM_IA bit value reflects the comparison result between FSS[7:0] and WTM[6:0], FIFO_WTM(@14h) bit range value. A comparison is enabled and the monitor bit is working when WTM[6:0] is set to a value between 1d and 127d. If bit range WTM[6:0]=‘0’, then the watermark monitor is de-asserted: FIFO_WTM_IA=‘0’, FIFO_STATUS2(@26h).

FIFO_WTM_IA=’1’, FIFO_STATUS2(@26), if and while the number of stored sample sets in the FIFO buffer FSS[7:0], FIFO_STATUS1@25) is greater than or equal to the watermark level selected by bit range WTM[6:0], FIFO_CTRL(@14h).

The FIFO_FULL_IA bit, FIFO_STATUS2(@26h), goes to '1' if and while FIFO is completely filled and no dss in FIFO have been overwritten, which means it will remain at level '1' for a duration of 1/ODR.

The FIFO_OVR_IA bit, FIFO_STATUS2(@26h), goes to ‘1’ if the FIFO buffer is full and at least one dss in the FIFO has been overwritten by the last generated dss.

While FIFO is running, the FIFO_STATUSx registers bits keep updating with the current status: they are reset when FIFO is reset.

All three status bits in FIFO_STATUS2(@26h) register can be singularly selected for diversion to the INT_DRY pin. They are helpful to trigger actions related to the specific buffer conditions and FIFO operating modes. See Section 8 Interrupt modes for details.

Table 13. **FIFO buffer status monitoring registers and bit ranges**

Register	@address	Bit range	Bit range mask
FIFO_STATUS1	25h	**FSS [7:0]**	11111111b = FFh
FIFO_STATUS2	26h	**FIFO_WTM_IA**	10000000b = 80h
FIFO_STATUS2	26h	**FIFO_OVR_IA**	01000000b = 40h
FIFO_STATUS2	26h	**FIFO_FULL_IA**	00100000b = 20h

9.3 FIFO buffer modes

Three main FIFO buffer behaviors are available:

- Bypass mode: the FIFO buffer is disabled and arriving dss are not filling it, hence bypassing the buffer;
- FIFO mode: arriving dss are filling all the available buffer slots until its selected size is completely filled (hence full or to stop-on-watermark level is reached) and no more empty slots are available at which point the buffer stops filling and keeps stored dss in it until an action is taken. Once filled, newer dss are lost (not stored in the FIFO buffer)
- Continuous (or Dynamic-Stream) mode: when arriving dss are filling all the available buffer slots until its selected size is completely filled (hence full or watermark level is reached) and no more empty slots are available at which point new arriving dss will start to replace older stored data. Once filled, older dss are lost (replaced in the FIFO buffer by newer dss);

FIFO buffer behaviors can be singularly selected (non-triggered modes), default, or as a sequence of two behaviors with the transition from the first to the second one triggered by an event signal (triggered modes). Triggered mode combinations can be selected by setting bit: TRIG_MODES='1', FIFO_CTRL(@13h) and by bit range F_MODE[1:0], FIFO_CTRL(@13h), see Table 11. FIFO modes and FIFO trigger mode selection. The event signal is the interrupt event generated when the threshold-based differential interrupt generator is enabled: IA, INT_SOURCE(@24h) register.

9.3.1 Non-triggered FIFO modes

9.3.1.1 Bypass mode

In Bypass mode, (TRIG_MODES & F_MODE[1:0] = 'x00', TRIGMODES='X', FIFO_CTRL(@13h)). When Bypass mode is selected, the full FIFO buffer content is cleared and all is cleared and reset. The FIFO buffer is not operational and it remains empty. If the device is operational, the pressure and temperature values are sent to the PRESS_OUT_x and TEMP_OUT_x registers (standard output) only. Switching to Bypass mode must be used in order to stop and to reset the FIFO buffer and its counters. This is the mandatory step to switch between different FIFO operating modes.

Figure 14. FIFO Bypass mode

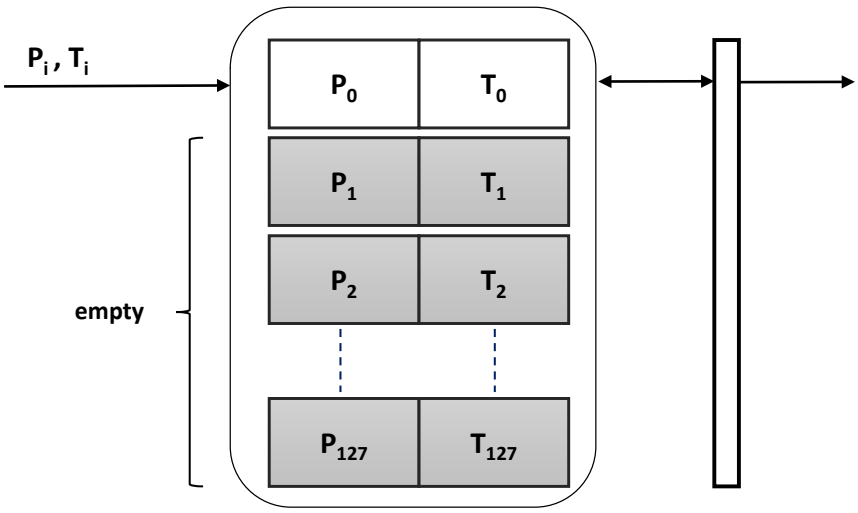


Table 14. Settings for FIFO buffer Bypass mode

Register	@address	Bit range	Value	Mask	Mode
FIFO_CTRL	13h	TRIG_MODES & F_MODE[1:0]	X00	00000111b = 07h	Bypass mode

9.3.1.2 *FIFO mode*

In FIFO mode, (TRIG_MODES & F_MODE[1:0] = '001', FIFO_CTRL(@13h)), the acquired dss generated at the current ODR pressure and temperature are stored in the FIFO buffer.

The FIFO buffer keeps filling until the FIFO is full (128 slots) or the selected and enabled watermark level is reached (if STOP_ON_WTM=1, FIFO_CTRL(@13h)).

When one of the two above conditions is true, the FIFO filling process will stop and data in FIFO will no longer be updated and will remain stored, unchanged until another action is taken. At this point, the FIFO content can be read. FIFO cannot be filled again until the buffer reset operation is executed. This is done by switching it to “Bypass mode”, F_MODE[1:0]='00’, FIFO_CTRL(@13h). To use FIFO mode again once it has stopped, the FIFO reset step is mandatory.

After this reset command, it is possible to re-engage FIFO mode by setting F_MODE[1:0] = ‘01’, FIFO_CTRL(@13h).

In FIFO mode, if the stop-on-watermark feature isn’t enabled (STOP_ON_WTM=0, FIFO_CTRL(@13h)), the FIFO buffer fillling status can be monitored by checking FIFO_STATUS2(@26h), bit FIFO_FULL_IA. On the other hand, if the stop-on-watermark feature is enabled (STOP_ON_WTM=1, FIFO_CTRL(@13h)), limited filling can be monitored by FIFO_STATUS2(@26h), bit FIFO_WTM_IA status.

Figure 15. FIFO mode with disabled stop-on-watermark feature

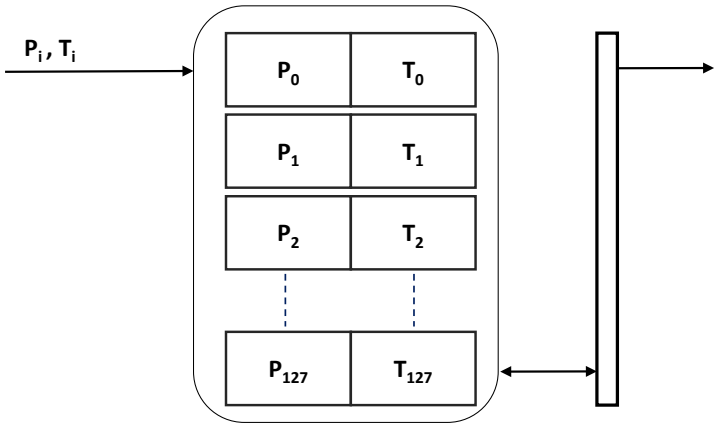


Figure 16. FIFO mode with enabled stop-on-watermark feature

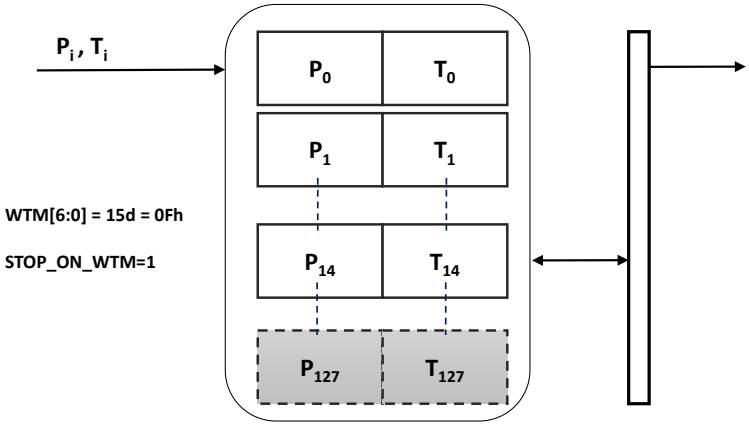


Table 15. Settings for FIFO buffer FIFO mode

Register	@address	Bit range	Value	Mask	Mode
FIFO_CTRL	13h	TRIG_MODES & F_MODE[1:0]	001	00000111b = 07h	FIFO mode

9.3.1.3 *Continuous (or Dynamic-Stream) mode*

When Continuous (or Dynamic-Stream) mode is enabled through the following settings: (TRIG_MODES & F_MODE[1:0] = ‘01x’, FIFO_CTRL(@13h)), the FIFO buffer starts running and the newer dss feed the available FIFO buffer slots. Once the FIFO is full (or the watermark level is reached if the stop-on-watermark feature has been configured and enabled), each new arriving dss will overwrite the older dss stored in the buffer: hence this older dss will be unread.

While the fullness status and zero slots are available, the overwriting cycle will continue until an action is taken on FIFO which can be:

- a (multi-)read operation;
- a FIFO reset.

To avoid losing data sets, a read operation should be performed at a pace faster than the ODR which will result in free FIFO slots. The reading speed of the host processor is important in order to free slots faster than new data sets are made available.

To stop Continuous mode configuration, Bypass mode must be selected which will result in stopping and resetting FIFO.

While Continuous mode is active, the FIFO collects data continuously and the FIFO_STATUS1 and FIFO_STATUS2 registers keep updating according to the number of stored dss.

When the next FIFO write operation renders the FIFO completely full, the FIFO_FULL_IA bit, FIFO_STATUS2(@26h) register will go to 1 and remains at ‘1’ for a duration of 1/ODR.

The next and successive arriving dss (if no slot has not been released in the meantime) will overwrite the oldest dss and generate a FIFO overrun condition. The FIFO_OVR_IA bit, FIFO_STATUS2(@26h) register rises to indicate when at least one FIFO dss has been overwritten to store the new data.

Data can be retrieved after the FIFO_FULL_IA event by reading the FIFO_DATA_OUT_x (from 78h to 7Ch) registers for the number of times specified by the content of the FIFO_STATUS1 register. Otherwise, leveraging on the FIFO_WTM_IA bit, FIFO_STATUS2(@26h) register, data can alternatively be retrieved when a threshold level, (WTM[6:0] in FIFO_WTM(@14h) register, is reached.

If the stop-on-watermark feature is enabled, bit STOP_ON_WTM=‘1’, FIFO_CTRL(@13h) register, the FIFO buffer size is limited to a customizable depth (number of slots or *dss*),equal to the value stored in the bit range WTM[6:0], FIFO_WTM(@14h) register. In this case, bit FIFO_WTM_IA, FIFO_STATUS2(@26h) register will rise to ‘1’, when the number of samples in FIFO reaches or surpasses the WTM[6:0] value.

If, after FIFO_WTM_IA has risen, a read operation happens which reduces the amount of stored dss to a level lower than the one defined by WTM[6:0], the FIFO_WTM_IA will de-assert.

When empty slots are no longer available, the WTM[6:0] level is exceeded and an overwrite happens. A soon as this condition occurs, the FIFO_OVR_IA bit, FIFO_STATUS2(@26h) register will rise to ‘1’.

9.3.2 Triggered FIFO modes

Triggered modes are a combination, a sequence of two, of the previously mentioned FIFO buffer behaviors, with the switch between the first and the second one triggered by one of the interrupt event conditions made available by the interrupt settings. Once the second behavior has been set, this condition will remain until an action is taken on FIFO. Three triggered modes are available.

The triggered modes are dependent on the interrupt active signal edge, bit IA, INT_SOURCE(@24h).

The IA signal is the output of the interrupt generator. The interrupt generator can be configured to toggle the IA signal according to different conditions and settings (see Section 8 Interrupt modes).

9.3.2.1 *Bypass-to-FIFO mode*

In Bypass-to-FIFO mode (TRIG_MODES & F_MODE[1:0] = '101', FIFO_CTRL(@13h)), the FIFO works in "Bypass mode" until an interrupt trigger event is generated, then it switches to FIFO mode. The trigger event is set through INTERRUPT_CFG(@0Bh) register.

When the interrupt is triggered, bit IA, INT_SOURCE(@24h) rises to '1' and the FIFO switches from Bypass to FIFO mode.

When the interrupt is de-asserted, the IA bit, INT_SOURCE(@24h) is equal to '0', the FIFO doesn't automatically switch back to Bypass mode: an action on it has to be taken according to the FIFO mode description.

Figure 17. Bypass-to-FIFO mode

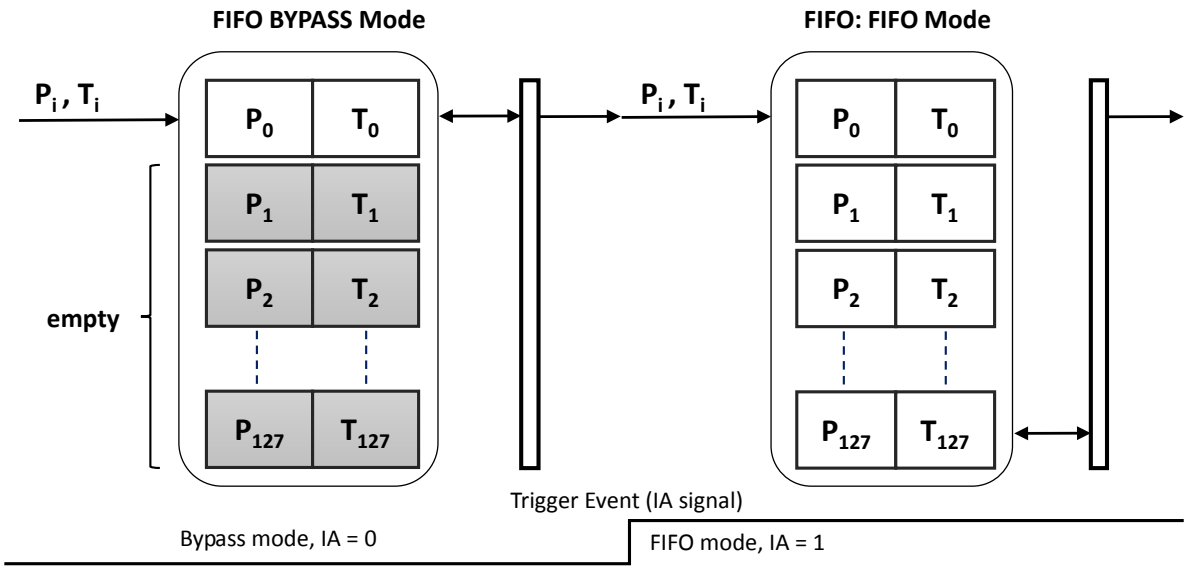


Table 16. Settings for FIFO buffer Bypass-to-FIFO mode

Register	@address	Bit range	Value	Mask	Mode
FIFO_CTRL	13h	TRIG_MODES & F_MODE[1:0]	101	00000111b = 07h	Bypass-to-FIFO mode

9.3.2.2 *Bypass-to-Continuous (Dynamic-Stream) mode*

When Bypass-to-Continuous (Dynamic-Stream) mode (TRIG_MODES & FMODE[2:0] = ‘110’, FIFO_CTRL(@13h)) is selected, the FIFO is set to Bypass mode and stays in this condition until a trigger event is generated. When this happens FIFO automatically switches to Continuous (Dynamic-Stream) mode. The trigger event is set through the INTERRUPT_CFG(@0Bh) register.

If the interrupt is triggered, the IA bit, INT_SOURCE(@24h) is equal to '1', and the FIFO switches from Bypass to Continuous (Dynamic-Stream) mode and stays in it until an action on FIFO is taken. FIFO doesn’t automatically switch back to Bypass mode if the interrupt is de-asserted (IA bit, INT_SOURCE(@24h) goes back to '0'). An action on FIFO has to be taken according to the Continuous mode description.

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

Table 17. Settings for FIFO buffer Bypass-to-Continuous mode

Register	@address	Bit range	Value	Mask	Mode
FIFO_CTRL	13h	TRIG_MODES & F_MODE[1:0]	110	00000111b = 07h	Bypass-to-Continuous mode

9.3.2.3 *Continuous (Dynamic-Stream)-to-FIFO mode*

In Continuous (Dynamic-Stream)-to-FIFO mode (TRIG_MODES & F_MODE[1:0] = ‘111’, FIFO_CTRL(@13h)) , the FIFO buffer works in Continuous (Dynamic-Stream) behavior until a trigger event is generated (IA signal), then it switches to FIFO mode behavior. The trigger event is set through INTERRUPT_CFG(@0Bh). If the interrupt is triggered, the bit IA=’1’, INT_SOURCE(@24h) is asserted, and the FIFO switches from Continuous (Dynamic-Stream) to FIFO mode behavior. When the interrupt is de-asserted, the IA=’0’, INT_SOURCE(@24h), the FIFO doesn’t automatically switch back to Continuous (Dynamic-Stream) behavior: an action to it has to be taken according to the Continuous (Dynamic-Stream) behavior description.

Table 18. Settings for FIFO buffer Continuous-to-FIFO mode

Register	@address	Bit range	Value	Mask	Mode
FIFO_CTRL	13h	TRIG_MODES & F_MODE[1:0]	111	00000111b = 07h	Continuous-to-FIFO mode

9.4 Retrieving data from FIFO

The action of consuming data from FIFO is performed by reading FIFO output registers. Data sample sets in FIFO are readable on a one-by-one basis, through the FIFO_DATA_OUT_x continuous register range: from register (@78h) to register (@7Ch).

Every time a dss is read from the FIFO, the remaining oldest entry is placed in the FIFO_DATA_OUT_x registers and status registers are updated accordingly. The FIFO_STATUS1 register content reports the number of remaining dss currently stored in FIFO.

Both single-read and multiple-read operations can be performed. It is recommended to avoid executing incomplete dss readings.

FIFO output registers (FIFO_DATA_OUT_x) support the rounding feature. The rounding feature allows executing register multi-reads without the need to manage the addresses, but it specifies the starting address and the number of bytes to be read. It defines the process of multiple reads where the address is automatically updated by the interface (I²C / MIPI I3C^{SM} / SPI) and it rolls back to 28h when register 2Ch is reached (see Section 5.1 Multi-read / write automatic address increment and rounding features).

A single complete dss can be read by a 5-byte multi-read operation executed against the FIFO_DATA_OUT_x registers (starting @78h). It returns the 5 bytes of the current oldest dss stored in FIFO.

Similarly, M (M=1,2, …, 128) data sets can be extracted from FIFO with a single bus transaction through an M*5-byte multi-read operation executed against the FIFO_DATA_OUT_x registers starting from FIFO_DATA_OUT_PRESS_XL(@78h). It is performed leveraging on the automatic address increment and rounding features, which are enabled by setting bit IF_ADD_INC=1, CTRL_REG2(@11h).

To read all the FIFO levels (128 data sample sets) stored in a full FIFO by a multiple-read operation, 640 bytes (5 output registers by 128 levels) have to be read, starting at FIFO_DATA_OUT_PRESS_XL (@78h)).

Revision history

Table 19. **Document revision history**

Date	Version	Changes
10-Sep-2019	1	Initial release

Contents

List of tables

List of figures