Freescale Semiconductor
Data Sheet: Technical Data
An Energy-Efficient Solution by Freescale

Document Number: FXLS8471Q
Rev. 1.5, 06/2015

# FXLS8471Q, 3-Axis, Linear Accelerometer

**FXLS8471Q**



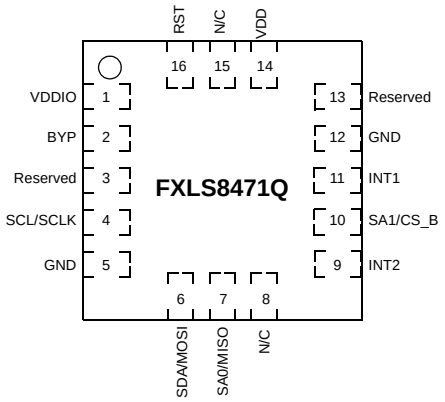

**16 LEAD QFN**
**3 mm x 3 mm x 1 mm**

FXLS8471Q is a small, low-power, 3-axis, linear accelerometer in a 3 mm x 3 mm x 1 mm QFN package. FXLS8471Q has dynamically selectable acceleration full-scale ranges of ±2 *g*/±4 *g*/±8 *g* and 14 bits of resolution. Output data rates (ODR) are programmable from 1.563 Hz to 800 Hz. $I^2C$ and SPI serial digital interfaces are provided along with several user programmable event detection functions that can be used to reduce the overall system power consumption by off-loading the host processor. FXLS8471Q is guaranteed to operate over the extended temperature range of -40 °C to +85 °C.

## Features

- 1.95 V to 3.6 V VDD supply voltage, 1.62 V to 3.6 V VDDIO voltage
- ±2 *g*/±4 *g*/±8 *g* dynamically selectable acceleration full-scale ranges
- Output Data Rates (ODR) from 1.563 Hz to 800 Hz
- Low noise: typically 99 µg/Hz in low-noise mode @ 200-Hz bandwidth
- 14-bit ADC resolution: 0.244 mg/LSB in ±2 *g,* full-scale range
- Embedded programmable acceleration event functions:
  - Freefall and Motion Detection
  - Transient Detection
  - Vector-Magnitude Change Detection
  - Pulse and Tap Detection (Single and Double)
  - Orientation Detection (Portrait/Landscape)
- Programmable automatic ODR change using Auto-Wake and return to Sleep functions to save power.
- 192-byte FIFO buffer, capable of storing up to 32 samples of X/Y/Z data
- Supports 4-wire SPI interface at up to 1 MHz; $I^2C$ Normal (100 kHz) and Fast Modes (400 kHz)
- Integrated accelerometer self-test function

**Top View**




**Pin Connections**

## Target Markets

- Industrial applications: vibration analysis, machine health monitoring, and platform stabilization
- Smartphones, tablets, digital cameras, and personal navigation devices
- Medical applications: patient monitoring, fall detection, and rehabilitation

## Applications

- Shock and vibration monitoring (mechatronic compensation, shipping, and warranty usage logging)
- User interface (menu scrolling by orientation change, tap detection for button replacement)
- Orientation detection (portrait/landscape: up/down, left/right, back/front position identification)
- Gaming and real-time activity analysis (pedometry, freefall and drop detection for hard disk drives and other devices)
- Power management for mobile devices using inertial event detection

Freescale reserves the right to change the detail specifications as may be required to permit improvements in the design of its products.

© 2013-2015 Freescale Semiconductor, Inc. All rights reserved.

| ORDERING INFORMATION | | | |
|---|---|---|---|
| **Part Number** | **Temperature Range** | **Package Description** | **Shipping** |
| FXLS8471QR1 | -40°C to +85°C | QFN-16 | Tape and Reel (1 k) |

## Related Documentation

The FXLS8471Q device features and operations are described in a variety of reference manuals, user guides, and application notes. To find the most-current versions of these documents:

1. Go to the Freescale homepage at:

   http://www.freescale.com/

2. In the Keyword search box at the top of the page, enter the device number FXLS8471Q.

In the Refine Your Result pane on the left, click on the Documentation link.

# Contents


**1 Block Diagram . . . 5**
**2 Pin Description . . . 6**
2.1 Soldering information . . . 7
2.2 Orientation . . . 8
**3 Example FXLS8471Q Driver Code . . . 9**
3.1 Introduction . . . 9
3.2 FXLS8471Q Addresses . . . 9
3.3 Sensor data structure . . . 9
3.4 FXLS8471Q Configuration function . . . 10
3.5 FXLS8471Q Data Read function . . . 11
**4 Terminology . . . 12**
4.1 Sensitivity . . . 12
4.2 Zero-g Offset . . . 12
4.3 Self-Test . . . 12
**5 Device Characteristics . . . 13**
5.1 Mechanical characteristics . . . 13
5.2 Electrical characteristics . . . 14
5.3 Absolute maximum ratings . . . 15
**6 Digital Interfaces . . . 16**
6.1 $I^2C$ interface characteristics . . . 16
6.1.1 General $I^2C$ operation . . . 17
6.1.2 $I^2C$ Read/Write operations . . . 17
6.2 SPI Interface characteristics . . . 19
6.2.1 General SPI operation . . . 19
6.2.2 SPI READ/WRITE operations . . . 20
6.2.3 $I^2C$/SPI auto detection . . . 20
6.2.4 Power supply sequencing and $I^2C$/SPI mode auto-detection . . . 20
**7 Modes of Operation . . . 21**
**8 Embedded Functionality . . . 22**
8.1 Factory calibration . . . 22
8.2 8-bit or 14-bit data . . . 22
8.3 Low-power modes versus high-resolution modes . . . 22
8.4 Auto-Wake/Sleep mode . . . 23
8.5 Freefall and Motion event detection . . . 23
8.5.1 Freefall detection . . . 23
8.5.2 Motion detection . . . 23
8.6 Transient detection . . . 23
8.7 Pulse detection . . . 23
8.8 Orientation detection . . . 24
8.9 Acceleration Vector-Magnitude detection . . . 24
**9 Register Map . . . 25**
**10 Registers by Functional Blocks . . . 28**
10.1 Device configuration . . . 28
10.1.1 STATUS (0x00) register . . . 28
10.1.2 DR_STATUS (0x00) register . . . 28
10.1.3 F_STATUS (0x00) register . . . 29
10.1.4 TRIG_CFG (0x0A) register . . . 30
10.1.5 SYSMOD (0x0B) register . . . 30
10.1.6 INT_SOURCE (0x0C) register . . . 31
10.1.7 WHO_AM_I (0x0D) register . . . 32
10.1.8 CTRL_REG1 (0x2A) register . . . 32
10.1.9 CTRL_REG2 (0x2B) register . . . 33
10.1.10 CTRL_REG3 [Interrupt Control Register] (0x2C) register . . . 35
10.1.11 CTRL_REG4 [Interrupt Enable Register] (0x2D) register . . . 36
10.1.12 CTRL_REG5 [Interrupt Routing Configuration Register] (0x2E) register . . . 37
10.2 Auto-Sleep trigger . . . 39
10.2.1 ASLP_COUNT (0x29) register . . . 39
10.3 Output data registers . . . 41
10.3.1 OUT_X_MSB (0x01), OUT_X_LSB (0x02), OUT_Y_MSB (0x03), OUT_Y_LSB (0x04), OUT_Z_MSB (0x05), OUT_Z_LSB (0x06) registers . . . 41
10.4 FIFO . . . 42
10.4.1 F_SETUP (0x09) register . . . 42
10.5 Sensor data configuration . . . 43

10.5.1 XYZ_DATA_CFG (0x0E) register . . . 43
10.6 High-Pass filter . . . 43
10.6.1 HP_FILTER_CUTOFF (0x0F) register . . . 43
10.7 Portrait/Landscape Detection . . . 45
10.7.1 PL_STATUS (0x10) register . . . 46
10.7.2 PL_CFG (0x11) register . . . 47
10.7.3 PL_COUNT (0x12) register . . . 47
10.7.4 PL_BF_ZCOMP (0x13) register . . . 48
10.7.5 PL_THS_REG (0x14) register . . . 49
10.8 Freefall and Motion detection . . . 49
10.8.1 A_FFMT_CFG (0x15) register . . . 50
10.8.2 A_FFMT_SRC (0x16) register . . . 51
10.8.3 A_FFMT_THS (0x17), A_FFMT_ THS_X_MSB (0x73), A_FFMT_THS_X_LSB (0x74), A_FFMT_THS_Y_MSB (0x75), A_FFMT_THS_Y_LSB (0x76), A_FFMT_THS_Z_MSB (0x77), A_FFMT_THS_Z_LSB (0x78) registers . . . 52
10.8.4 A_FFMT_COUNT (0x18) register . . . 53
10.9 Accelerometer vector-magnitude function . . . 55
10.9.1 A_VECM_CFG (0x5F) register . . . 55
10.9.2 A_VECM_THS_MSB (0x60) register . . . 56
10.9.3 A_VECM_THS_LSB (0x61) register . . . 56
10.9.4 A_VECM_CNT (0x62) register . . . 56
10.9.5 A_VECM_INITX_MSB (0x63) register . . . 57
10.9.6 A_VECM_INITX_LSB (0x64) register . . . 57
10.9.7 A_VECM_INITY_MSB (0x65) register . . . 57
10.9.8 A_VECM_INITY_LSB (0x66) register . . . 57
10.9.9 A_VECM_INITZ_MSB (0x67) register . . . 57
10.9.10 A_VECM_INITZ_LSB (0x68) register . . . 58
10.10 Transient (AC) acceleration detection . . . 58
10.10.1 TRANSIENT_CFG (0x1D) register . . . 58
10.10.2 TRANSIENT_SRC (0x1E) register . . . 59
10.10.3 TRANSIENT_THS (0x1F) register . . . 60
10.10.4 TRANSIENT_COUNT (0x20) register . . . 60
10.11 Pulse detection . . . 61
10.11.1 PULSE_CFG (0x21) register . . . 61
10.11.2 PULSE_SRC (0x22) register . . . 62
10.11.3 PULSE_THSX (0x23) register . . . 62
10.11.4 PULSE_THSY (0x24) register . . . 63
10.11.5 PULSE_THSZ (0x25) register . . . 63
10.11.6 PULSE_TMLT (0x26) register . . . 63
10.11.7 PULSE_LTCY (0x27) register . . . 64
10.11.8 PULSE_WIND (0x28) register . . . 65
10.12 Offset correction . . . 66
10.12.1 OFF_X (0x2F) register . . . 66
10.12.2 OFF_Y (0x30) register . . . 66
10.12.3 OFF_Z (0x31) register . . . 66
**11 Mounting Guidelines for the Quad Flat No-Lead (QFN) Package . . . 67**
11.1 Overview of soldering considerations . . . 67
11.2 Halogen content . . . 67
11.3 PCB mounting recommendations . . . 67
**12 Package Thermal Characteristics . . . 69**
**13 Package . . . 70**
**Appendix A . . . 73**
A.1 Errata . . . 73
A.1.1 SPI Mode Soft-reset using CTRL_REG2 (0x2B), bit 6 . . . 73
**14 Revision History . . . 74**

# 1 Block Diagram

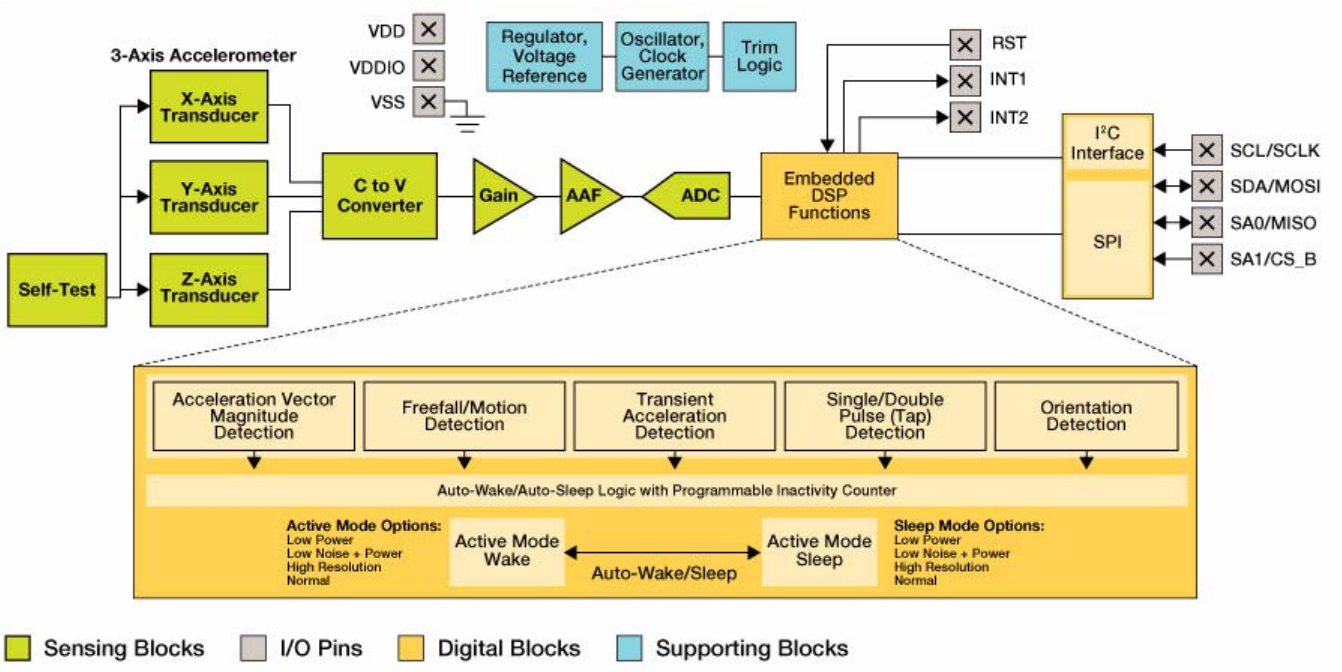
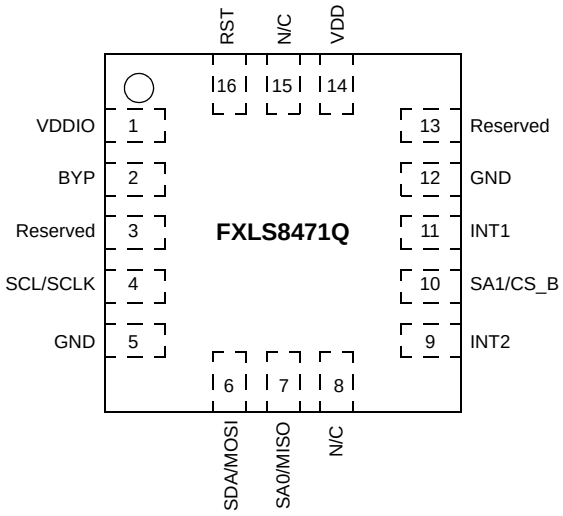



**Figure 1. Block diagram**

# 2 Pin Description




**Top View**
**16 Lead QFN-COL**
**3 mm x 3 mm x 1 mm**

**Figure 2. Pinout diagram**

**Table 1. Pin Description**

| Pin | Name | Function |
|---|---|---|
| 1 | VDDIO | Interface power supply |
| 2 | BYP | Internal regulator output bypass capacitor connection |
| 3 | Reserved | Test reserved, connect to GND |
| 4 | SCL/SCLK | $I^2C$ Serial Clock/SPI Clock |
| 5 | GND | Ground |
| 6 | SDA/MOSI | $I^2C$ Serial Data/SPI Master Out, Slave In |
| 7 | SA0/MISO(1) | $I^2C$ address selection bit 0/SPI Master In, Slave Out |
| 8 | N/C | Internally not connected |
| 9 | INT2 | Interrupt 2 |
| 10 | SA1/CS_B | $I^2C$ address selection bit 1(2)/SPI Chip Select (active low) |
| 11 | INT1 | Interrupt 1 |
| 12 | GND | Ground |
| 13 | Reserved | Test reserved, connect to GND |
| 14 | VDD | Power supply |
| 15 | N/C | Internally not connected |
| 16 | RST | Reset input, active high. Connect to GND if unused |

1. The SA0 pin is also used to select the desired serial interface mode during POR and also after a hard/soft reset event. Please see Section 6.2.3, "$I^2C$/SPI auto detection" for more information
2. See Table 8 for $I^2C$ address options selectable using the SA0 and SA1 pins.

Device power is supplied through the VDD pin. Power-supply, decoupling capacitors (100 nF ceramic plus 4.7 µF or larger bulk) should be placed as close as possible to pin 14 of the device. The digital interface supply voltage (VDDIO) must also be decoupled with a 100 nF ceramic capacitor placed as close as possible to pin 1 of the device.

The digital control signals SCL, SDA, SA0, SA1, and RST are not tolerant of voltages more than VDDIO + 0.3 V. If VDDIO is removed, these pins will clamp any logic signals through their internal ESD protection diodes.

The function and timing of the two interrupt pins (INT1 and INT2) are user-programmable through the I²C/SPI interface. The SDA and SCL I²C connections are open drain and therefore require a pullup resistor as shown in the application diagram in Figure 3. The INT1 and INT2 pins may also be configured for open-drain operation. If they are configured for open drain, external pullup resistors are required.

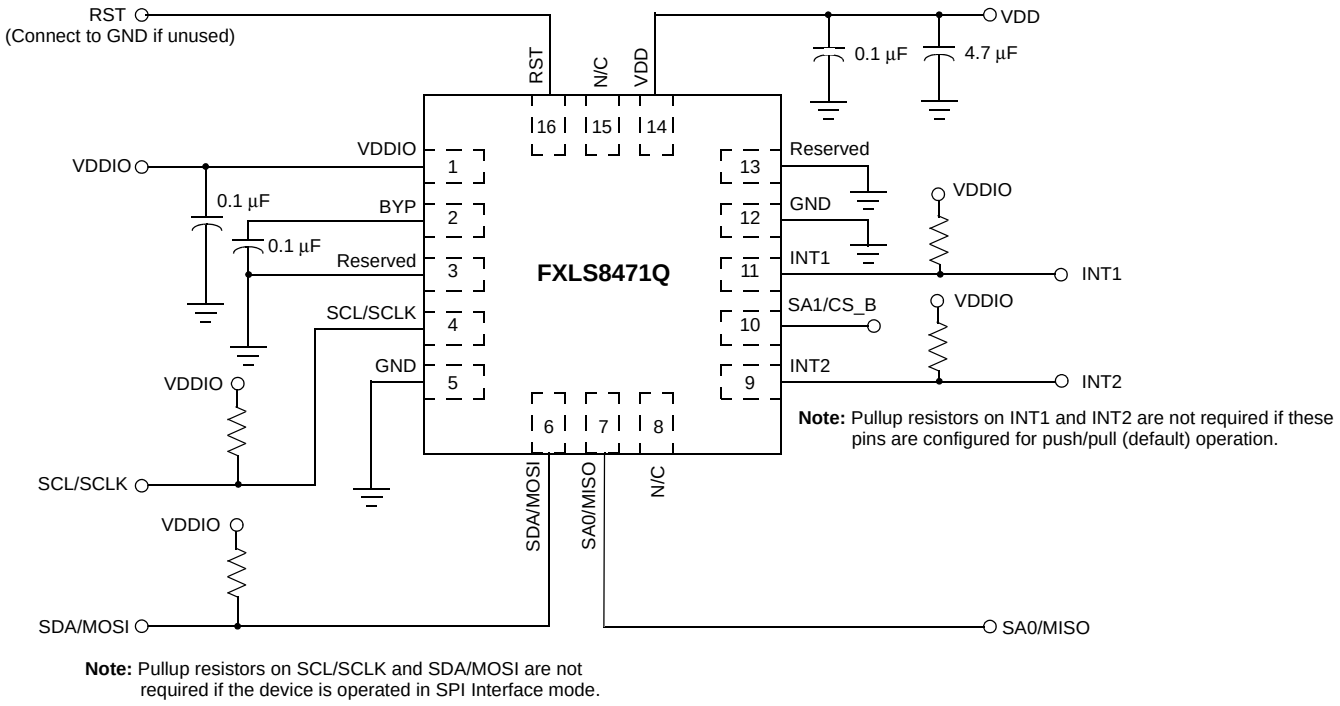



**Figure 3. Electrical connection**

## 2.1 Soldering information

The QFN package is compliant with the RoHS standards. Please refer to Freescale application note AN4077 for more information.

## 2.2 Orientation

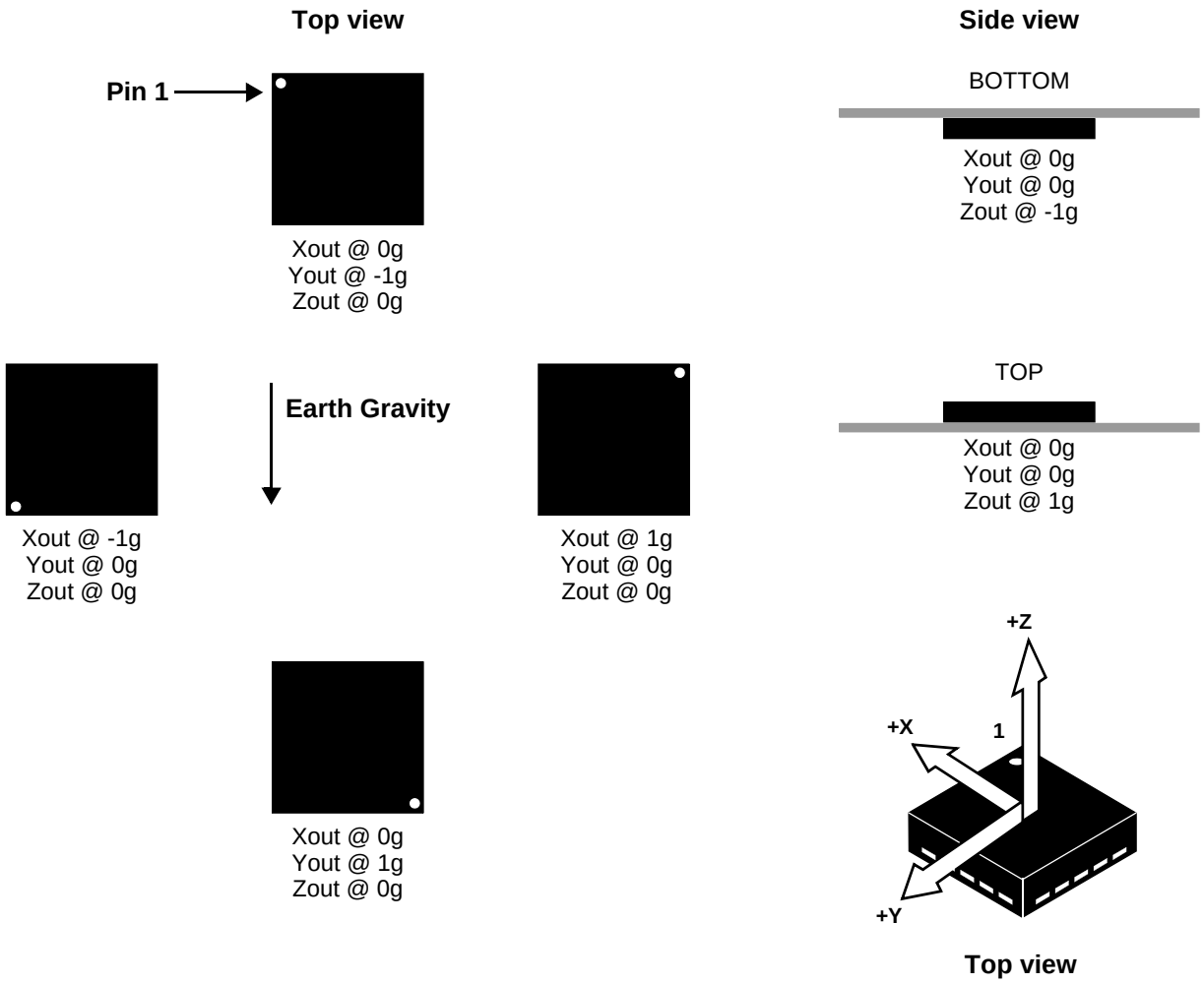



**Figure 4. Component axes orientation and response to gravity stimulus**

# 3 Example FXLS8471Q Driver Code

## 3.1 Introduction

It is very straightforward to configure the FXLS8471Q and start receiving data from the three accelerometerchannels. Unfortunately, since every hardware platform will be different, it is not possible to provide completely portable software drivers. This section therefore provides real FXLS8471Q driver code for a Kinetis uC board running under the MQX operating system. The $I^2C$ functions s_i2c_read_regs and s_i2c_write_regs are not provided here and should be replaced with the corresponding low level $I^2C$ driver code on the development platform.

## 3.2 FXLS8471Q Addresses

This section lists the $I^2C$ address of the FXLS8471Q. The $I^2C$ address depends on the logic level of FXLS8471Q pins SA0 and SA1 so the $I^2C$ address may be 0x1C, 0x1D, 0x1E or 0x1F.

**Example 1.**

```
// FXLS8471Q I2C address
#define FXLS8471Q_SLAVE_ADDR      0x1E   // with pins SA0=0, SA1=0
```

Some of the key FXLS8471Q internal register addresses are listed below.

**Example 2.**

```
// FXLS8471Q internal register addresses
#define FXLS8471Q_STATUS          0x00
#define FXLS8471Q_WHOAMI          0x0D
#define FXLS8471Q_XYZ_DATA_CFG    0x0E
#define FXLS8471Q_CTRL_REG1       0x2A
#define FXLS8471Q_WHOAMI_VAL      0x6A
```

The reference driver here does a block read of the FXLS8471Q status byte plus three 16-bit accelerometer channels.

**Example 3.**

```
// number of bytes to be read from FXLS8471Q
#define FXLS8471Q_READ_LEN 7// status plus 3 accelerometer channels
```

## 3.3 Sensor data structure

The high and low bytes of the three accelerometer are placed into a structure of type SRAWDATA containing three signed short integers.

**Example 4.**

```
typedef struct
{
  int16_t  x;
  int16_t  y;
  int16_t  z;
} SRAWDATA;
```

## 3.4 FXLS8471Q Configuration function

This function configures the FXLS8471Q for a 200-Hz ODR. The code is self-explanatory and can be easily customized for different settings.

**Example 5.**

```
// function configures FXLS8471Q accelerometer sensor
static _mqx_int s_FXLS8471Q_start(MQX_FILE_PTR aFP)
{
  uint8_t databyte;

  // read and check the FXLS8471Q WHOAMI register
  if (s_i2c_read_regs(aFP, FXLS8471Q_SLAVE_ADDR, FXLS8471Q_WHOAMI, &databyte,
    (uint8_t) 1) != 1)
  {
    return (I2C_ERROR);
  }
  if (databyte != FXLS8471Q_WHOAMI_VAL)
  {
    return (I2C_ERROR);
  }

  // write 0000 0000 = 0x00 to accelerometer control register 1 to place FXLS8471Q into
  // standby
  // [7-1] = 0000 000
  // [0]: active=0
  databyte = 0x00;
  if (s_i2c_write_regs(aFP, FXLS8471Q_SLAVE_ADDR, FXLS8471Q_CTRL_REG1, &databyte,
    (uint8_t) 1) != 1)
  {
    return (I2C_ERROR);
  }

  // write 0000 0001= 0x01 to XYZ_DATA_CFG register
  // [7]: reserved
  // [6]: reserved
  // [5]: reserved
  // [4]: hpf_out=0
  // [3]: reserved
  // [2]: reserved
  // [1-0]: fs=01 for accelerometer range of +/-4g with 0.488mg/LSB
  databyte = 0x01;
  if (s_i2c_write_regs(aFP, FXLS8471Q_SLAVE_ADDR, FXLS8471Q_XYZ_DATA_CFG,
    &databyte, (uint8_t) 1) != 1)
  {
    return (I2C_ERROR);
  }

  // write 0001 0101b = 0x15 to accelerometer control register 1
  // [7-6]: aslp_rate=00
  // [5-3]: dr=010 for 200Hz data rate
  // [2]: lnoise=1 for low noise mode
  // [1]: f_read=0 for normal 16 bit reads
  // [0]: active=1 to take the part out of standby and enable sampling
  databyte = 0x15;
  if (s_i2c_write_regs(aFP, FXLS8471Q_SLAVE_ADDR, FXLS8471Q_CTRL_REG1, &databyte,
    (uint8_t) 1) != 1)
```

```
  {
    return (I2C_ERROR);
  }

  // normal return
  return (I2C_OK);
}
```

## 3.5 FXLS8471Q Data Read function

This function performs a block read of the status and acceleration data and places the bytes read into the structures of type SRAWDATA as signed short integers.

**Example 6.**

```
// read status and the three channels of accelerometer data from
// FXLS8471Q (7 bytes)
int16_t ReadAccel(SRAWDATA *pAccelData)
{
  MQX_FILE_PTR fp;                          // I2C file pointer
  uint8_t Buffer[FXLS8471Q_READ_LEN];       // read buffer

  // read FXLS8471Q_READ_LEN=7 bytes (status byte and the three channels of data)
  if (s_i2c_read_regs(fp, FXLS8471Q_SLAVE_ADDR, FXLS8471Q_STATUS, Buffer,
    FXLS8471Q_READ_LEN) == FXLS8471Q_READ_LEN)
  {
    // copy the 14 bit accelerometer byte data into 16 bit words
    pAccelData->x = ((Buffer[1] << 8) | Buffer[2])>> 2;
    pAccelData->y = ((Buffer[3] << 8) | Buffer[4])>> 2;
    pAccelData->z = ((Buffer[5] << 8) | Buffer[6])>> 2;
  }
  else
  {
    // return with error
    return (I2C_ERROR);
  }

  // normal return
  return (I2C_OK);
}
```

# 4 Terminology

## 4.1 Sensitivity

The sensitivity is represented in LSB/g. In 2-*g* mode the sensitivity is 4096 LSB/g. In 4-*g* mode the sensitivity is 2048 LSB/ *g* and in 8-*g* mode the sensitivity is 1024 LSB/g.

## 4.2 Zero-g Offset

Zero-g Offset describes the deviation of an actual output signal from the ideal output signal if the sensor is stationary. A sensor stationary on a horizontal surface will measure 0 *g* in X-axis and 0 *g* in Y-axis whereas the Z-axis will measure 1 *g*. A deviation from ideal value in this case is called Zero-*g* offset. Offset is to some extent a result of stress on the MEMS sensor and therefore the offset can slightly change after mounting the sensor onto a printed circuit board or exposing it to extensive mechanical stress.

## 4.3 Self-Test

Self-Test can be used to verify the transducer functionality without applying an external mechanical stimulus. When Self-Test is activated, an electrostatic actuation force is applied to the sensor, simulating a small acceleration. In this case, the sensor outputs will exhibit a change in their DC levels which are related to the selected full scale through the device sensitivity. When Self-Test is activated, the device output level is given by the algebraic sum of the signals produced by the acceleration acting on the sensor and by the electrostatic self-test force.

# 5 Device Characteristics

## 5.1 Mechanical characteristics

**Table 2. Mechanical characteristics @ VDD = 2.5 V, VDDIO = 1.8 V T = 25°C unless otherwise noted.**

| Parameter | Test Conditions | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Measurement range(1) | ±2 *g* mode | $FS_{ACC}$ | | ±2 | | *g* |
| | ±4 *g* mode | | | ±4 | | |
| | ±8 *g* mode | | | ±8 | | |
| Sensitivity | ±2 *g* mode | $SEN_{ACC}$ | | 4096 | | LSB/*g* |
| | | | | 0.244 | | mg/LSB |
| | ±4 *g* mode | | | 2048 | | LSB/*g* |
| | | | | 0.488 | | mg/LSB |
| | ±8 *g* mode | | | 1024 | | LSB/*g* |
| | | | | 0.976 | | mg/LSB |
| Sensitivity change with temperature(1) | ±2 *g*, ±4 *g*, ±8 *g* modes | $TCS_{ACC}$ | | ±0.01 | | %/°C |
| Sensitivity accuracy | @ 25°C | $SEN\text{-}TOL_{ACC}$ | | ±2.5 | | $\%SEN_{ACC}$ |
| Zero-*g* level offset accuracy(2) | ±2 *g*, ±4 *g*, ±8 *g* modes | $OFF_{ACC}$ | | ±20 | | mg |
| Zero-*g* level offset accuracy post-board mount(3) | ±2 *g*, ±4 *g*, ±8 *g* modes | $OFF\text{-}PBM_{ACC}$ | | ±30 | | mg |
| Zero-*g* level change versus temperature | -40°C to 85°C(1) | $TCO_{ACC}$ | | ±0.2 | | mg/°C |
| Nonlinearity (deviation from straight line)(4)(5) | Over ±1 *g* range normal mode | $NL_{ACC}$ | | ±0.5 | | $\%FS_{ACC}$ |
| Self-Test output change(6)<br>X<br>Y<br>Z | Set to ±2 *g* mode | $STOC_{ACC}$ | <br>+192<br>+270<br>+1275 | | | LSB |
| Output noise density(4)(7) | ODR = 400 Hz, normal mode | $ND_{ACC\text{-}NM}$ | | 126 | | µg/√Hz |
| | ODR = 400 Hz, low-noise mode(1) | $ND_{ACC\text{-}LNM}$ | | 99 | | µg/√Hz |
| Operating temperature range | | Top | -40 | | +85 | °C |

1. Dynamic range is limited to ±4 *g* when in the low-noise mode.
2. Before board mount.
3. Post-board mount offset specifications are based on a 2-layer PCB design.
4. Evaluation only.
5. After post-board mount corrections for sensitivity, cross axis and offset. Refer to AN4399 for more information.
6. Self-test is only exercised along one direction for each sensitive axis.
7. Measured using earth's gravitational field (1 *g*) with the device oriented horizontally (+Z axis up) and stationary.

## 5.2 Electrical characteristics

**Table 3. Electrical characteristics @ VDD = 2.5 V, VDDIO = 1.8 V T = 25°C unless otherwise noted.**

| Parameter | Test Conditions | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Supply voltage | | VDD | 1.95 | 2.5 | 3.6 | V |
| Interface supply voltage | | VDDIO | 1.62 | 1.8 | 3.6 | V |
| Low-power acceleration mode | ODR = 12.5 Hz | $Idd_{ACC-LPM}$ | | 8 | | µA |
| | ODR = 100 Hz | | | 35 | | |
| | ODR = 400 Hz | | | 130 | | |
| Normal acceleration mode | ODR = 50 Hz | $Idd_{ACC-NM}$ | | 35 | | µA |
| | ODR = 200 Hz | | | 130 | | |
| | ODR = 800 Hz | | | 240 | | |
| Current during boot sequence, 0.9 mS max duration using recommended regulator bypass capacitor | VDD = 2.5 V | $Idd_{BOOT}$ | | | 3 | mA |
| Value of capacitor on BYP pin | -40°C to 85°C | $C_{BYP}$ | 75 | 100 | 470 | nF |
| Standby mode current @ 25°C | Standby mode | $Idd_{STBY}$ | | 2 | | µA |
| Standby mode current over-temperature range | Standby mode | $Idd_{STBY}$ | | | 10 | µA |
| Digital high-level input voltage RST pin | | $VIH_{RST}$ | 1.04 | | | V |
| Digital low-level input voltage RST pin | | $VIL_{RST}$ | | | 0.68 | V |
| Digital high-level input voltage SCL, SDA, SA0, SA1 | | VIH | 0.75*VDDIO | | | V |
| Digital low-level input voltage SCL, SDA, SA0, SA1 | | VIL | | | 0.3*VDDIO | V |
| High-level output voltage INT1, INT2 | $I_O$ = 500 µA | VOH | 0.9*VDDIO | | | V |
| Low-level output voltage INT1, INT2 | $I_O$ = 500 µA | VOL | | | 0.1*VDDIO | V |
| Low-level output voltage SDA | $I_O$ = 500 µA | $VOL_{SDA}$ | | | 0.1*VDDIO | V |
| SCL, SDA pin leakage | 25°C | | | 1.0 | | nA |
| | -40°C to 85°C | | | 4.0 | | |
| SCL, SDA pin capacitance | | | | 3 | | pf |
| VDD rise time | | | 0.001 | | 1000 | ms |
| Boot time(1) | | $T_{BOOT}$ | | | 1000 | µs |
| Turn-on time 1(2) | | $T_{POR \rightarrow ACT}$ | | 2/ODR + 2 | | ms |
| Turn-on time 2(3) | | $T_{STBY \rightarrow ACT}$ | | 2/ODR + 1 | | ms |
| Operating temperature range | | $T_{OP}$ | -40 | | +85 | °C |

1. Time from VDDIO on and VDD > VDD min until $I^2C$/SPI interface ready for operation.
2. Time to obtain valid data from power-down mode to Active mode.
3. Time to obtain valid data from Standby mode to Active mode.

## 5.3 Absolute maximum ratings

Stresses above those listed as “absolute maximum ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device under these conditions is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

**Table 5. Maximum ratings**

| Rating | Symbol | Value | Unit |
|---|---|---|---|
| Maximum acceleration (all axes, 100 µs) | $g_{max}$ | 5,000 | *g* |
| Supply voltage, IO voltage | $VDD_{max}$/ $VDDIO_{max}$ | -0.3 to +3.6 | V |
| Input voltage on any control pin (SA0/MISO, SA1/CS_B, SCL/SCLK, SDA/MOSI, RST) | $VIN_{max}$ | -0.3 to VDDIO + 0.3 | V |
| Drop-Test height | $D_{drop}$ | 1.8 | m |
| Storage temperature range | $T_{STG}$ | -40 to +125 | °C |

**Table 6. ESD and latchup protection characteristics**

| Rating | Symbol | Value | Unit |
|---|---|---|---|
| Human Body Model | HBM | ±2000 | V |
| Machine Model | MM | ±200 | V |
| Charge Device Model | CDM | ±500 | V |
| Latchup current at T = 85°C | $I_{LU}$ | ±100 | mA |



This device is sensitive to mechanical shock. Improper handling can cause permanent damage to the part or cause the part to otherwise fail.



This device is sensitive to ESD, improper handling can cause permanent damage to the part.

# 6 Digital Interfaces

## 6.1 I²C interface characteristics

**Table 7. I²C slave timing values(1)**

| Parameter | Symbol | I²C Fast Mode | | Unit |
|---|---|---|---|---|
| | | Min | Max | |
| SCL Clock Frequency | $f_{SCL}$ | 0 | 400 | kHz |
| Bus Free Time between STOP and START condition | $t_{BUF}$ | 1.3 | | µs |
| (Repeated) START Hold Time | $t_{HD;STA}$ | 0.6 | | µs |
| (Repeated) START Setup Time | $t_{SU;STA}$ | 0.6 | | µs |
| STOP Condition Setup Time | $t_{SU;STO}$ | 0.6 | | µs |
| SDA Data Hold Time | $t_{HD;DAT}$ | 0.05 | 0.9(2) | µs |
| SDA Valid Time(3) | $t_{VD;DAT}$ | | 0.9(2) | µs |
| SDA Valid Acknowledge Time(4) | $t_{VD;ACK}$ | | 0.9(2) | µs |
| SDA Setup Time | $t_{SU;DAT}$ | 100 | | ns |
| SCL Clock Low Time | $t_{LOW}$ | 1.3 | | µs |
| SCL Clock High Time | $t_{HIGH}$ | 0.6 | | µs |
| SDA and SCL Rise Time | $t_r$ | 20 + 0.1 $C_b$(5) | 300 | ns |
| SDA and SCL Fall Time | $t_f$ | 20 + 0.1 $C_b$(5) | 300 | ns |
| Pulse width of spikes on SDA and SCL that must be suppressed by internal input filter | $t_{SP}$ | 0 | 50 | ns |

1. All values referred to VIH (min) and VIL (max) levels.
2. This device does not stretch the LOW period ($t_{LOW}$) of the SCL signal.
3. $t_{VD;DAT}$ = time for Data signal from SCL LOW to SDA output.
4. $t_{VD;ACK}$ = time for Acknowledgement signal from SCL LOW to SDA output (HIGH or LOW, depending on which one is worse).
5. $C_b$ = total capacitance of one bus line in pF.

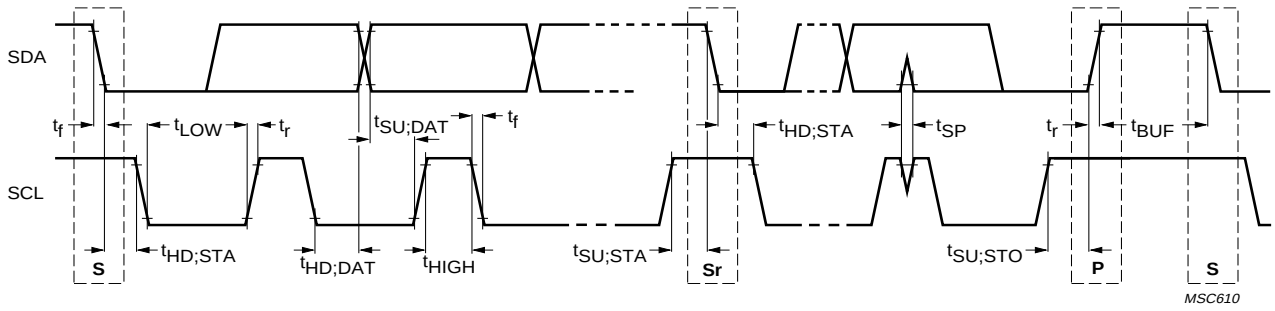



**Figure 5. I²C slave timing diagram**

### 6.1.1 General I²C operation

There are two signals associated with the I²C bus: the Serial Clock Line (SCL) and the Serial Data line (SDA). The latter is a bidirectional line used for sending and receiving the data to/from the interface. External pullup resistors connected to VDDIO are required for SDA and SCL. When the bus is free both the lines are high. The I²C interface is compliant with fast mode (400 kHz), and normal mode (100 kHz) I²C standards. Operation at frequencies higher than 400 kHz is possible, but depends on several factors including the pullup resistor values, and total bus capacitance (trace + device capacitance). See Table 8 for more information.

A transaction on the bus is started through a start condition (ST) signal, which is defined as a HIGH-to-LOW transition on the data line while the SCL line is held HIGH. After the ST signal has been transmitted by the master, the bus is considered busy. The next byte of data transmitted contains the slave address in the first seven bits, and the eighth bit, the read/write bit, indicates whether the master is receiving data from the slave or transmitting data to the slave. When an address is sent, each device in the system compares the first seven bits after the ST condition with its own address. If they match, the device considers itself addressed by the master. The 9th clock pulse, following the slave address byte (and each subsequent byte) is the acknowledge (ACK). The transmitter must release the SDA line during the ACK period. The receiver must then pull the data line low so that it remains stable low during the high period of the acknowledge clock period.

The number of bytes per transfer is unlimited. If a receiver can't receive another complete byte of data until it has performed some other function, it can hold the clock line, SCL low to force the transmitter into a wait state. Data transfer only continues when the receiver is ready for another byte and releases the data line. This delay action is called clock stretching. Not all receiver devices support clock stretching. Not all master devices recognize clock stretching. This part does not use clock stretching.

A low to high transition on the SDA line while the SCL line is high is defined as a stop condition (SP) signal. A write or burst write is always terminated by the master issuing the SP signal. A master should properly terminate a read by not acknowledging a byte at the appropriate time in the protocol. A master may also issue a repeated start signal (SR) during a transfer

The slave addresses that may be assigned to the FXLS8471Q part are 0x1C, 0x1D, 0x1E, or 0x1F. The selection is made through the logic level of the SA1 and SA0 inputs.

**Table 8. I²C slave address**

| SA1 | SA0 | Slave address |
|---|---|---|
| 0 | 0 | 0x1E |
| 0 | 1 | 0x1D |
| 1 | 0 | 0x1C |
| 1 | 1 | 0x1F |

### 6.1.2 I²C Read/Write operations

#### Single-byte read

The master (or MCU) transmits a start condition (ST) to the FXLS8471Q, followed by the slave address, with the R/W bit set to “0” for a write, and the FXLS8471Q sends an acknowledgement. Then the master (or MCU) transmits the address of the register to read and the FXLS8471Q sends an acknowledgement. The master (or MCU) transmits a repeated start condition (SR), followed by the slave address with the R/W bit set to “1” for a read from the previously selected register. The FXLS8471Q then acknowledges and transmits the data from the requested register. The master does not acknowledge (NAK) the transmitted data, but transmits a stop condition to end the data transfer.

#### Multiple-byte read

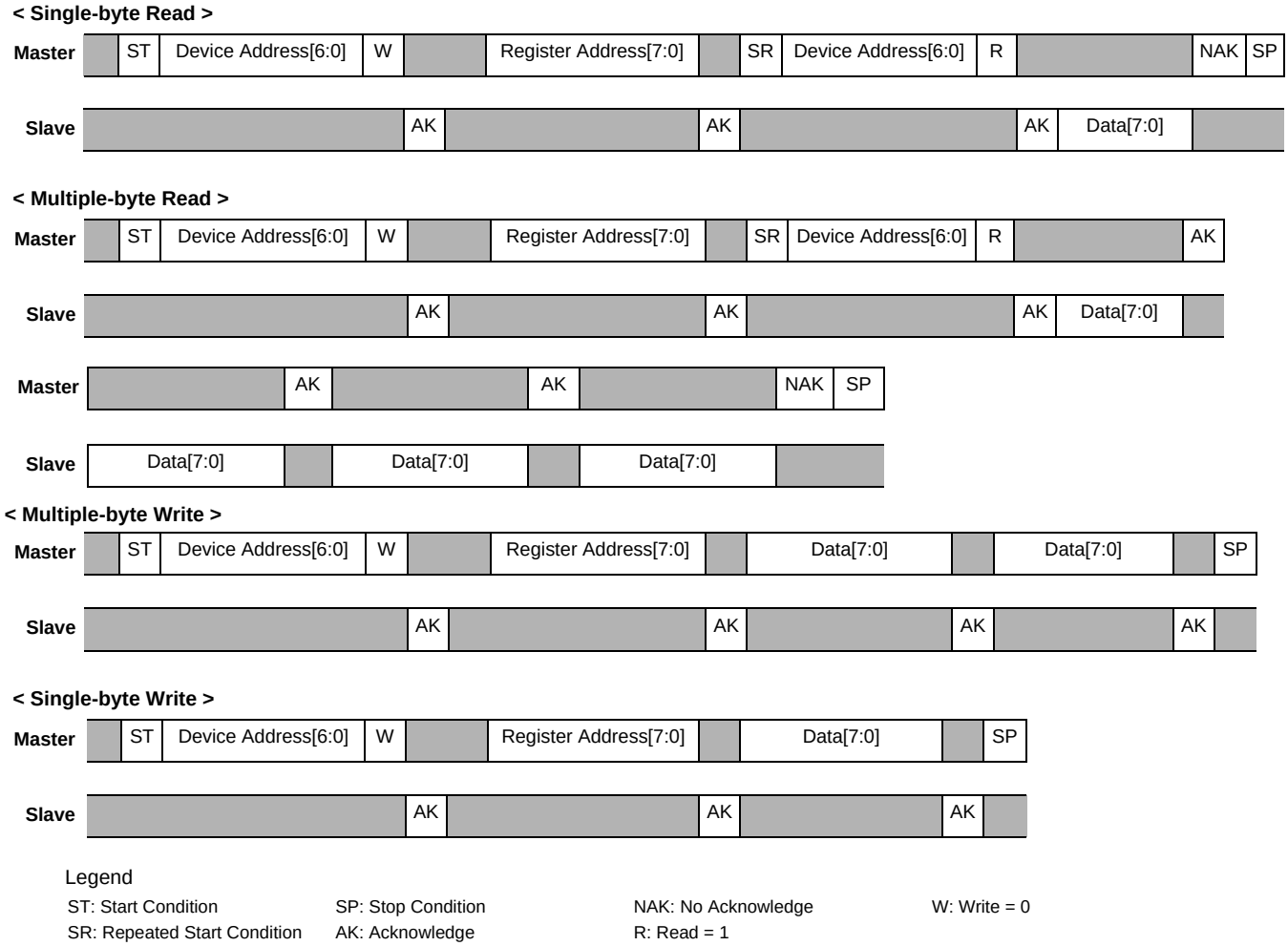
When performing a multi-byte or “burst” read, the FXLS8471Q automatically increments the register address read pointer after a read command is received. Therefore, after following the steps of a single-byte read, multiple bytes of data can be read from sequential registers after each FXLS8471Q acknowledgment (AK) is received until a no acknowledge (NAK) occurs from the master followed by a stop condition (SP) signaling an end of transmission.

#### Single-byte write

To start a write command, the master transmits a start condition (ST) to the FXLS8471Q, followed by the slave address with the R/W bit set to “0” for a write, and the FXLS8471Q sends an acknowledgement. Then the master (or MCU) transmits the address of the register to write to, and the FXLS8471Q sends an acknowledgement. Then the master (or MCU) transmits the 8-bit data to write to the designated register and the FXLS8471Q sends an acknowledgement that it has received the data. Since this transmission is complete, the master transmits a stop condition (SP) to end the data transfer. The data sent to the FXLS8471Q is now stored in the appropriate register.

### Multiple-byte write

The FXLS8471Q automatically increments the register address write pointer after a write command is received. Therefore, after following the steps of a single-byte write, multiple bytes of data can be written to sequential registers after each FXLS8471Q acknowledgment (ACK) is received.




Legend

| | | | |
|---|---|---|---|
| ST: Start Condition | SP: Stop Condition | NAK: No Acknowledge | W: Write = 0 |
| SR: Repeated Start Condition | AK: Acknowledge | R: Read = 1 | |

**Figure 6. I$^2$C timing diagram**

## 6.2 SPI Interface characteristics

SPI interface is a classical master/slave serial port. The FXLS8471Q is always considered as the slave and thus is never initiating the communication.

Table 9 and Figure 7 describe the timing requirements for the SPI system.

**Table 9. SPI timing**

| Function | Symbol | Min | Max | Unit |
|---|---|---|---|---|
| Operating Frequency | Of | — | 1 | MHz |
| SCLK Period | tSCLK | 1000 | — | ns |
| SCLK High time | tCLKH | 500 | — | ns |
| SCLK Low time | tCLKL | 500 | — | ns |
| CS_B lead time | tSCS | 65 | — | ns |
| CS_B lag time | tHCS | 65 | — | ns |
| MOSI data setup time | tSET | 25 | — | ns |
| MOSI data hold time | tHOLD | 75 | — | ns |
| MISO data valid (after SCLK low edge) | tDDLY | — | 500 | ns |
| Width CS High | tWCS | 100 | — | ns |

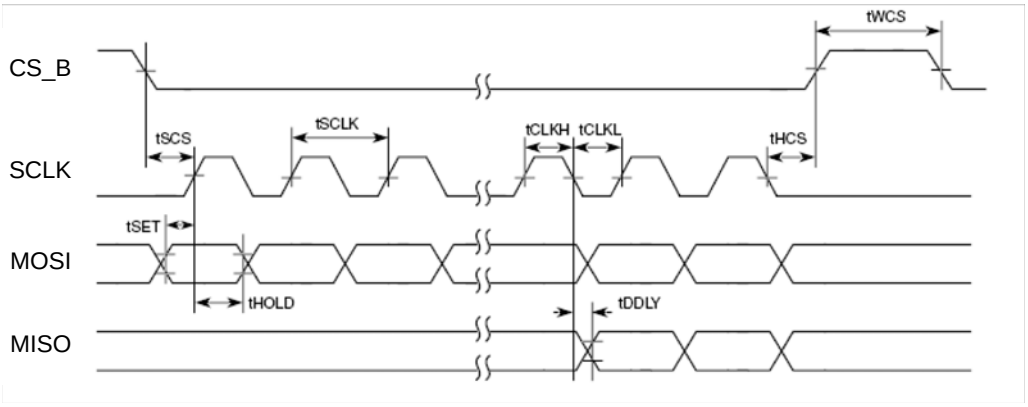



**Figure 7. SPI Timing Diagram**

### 6.2.1 General SPI operation

The CS_B pin is driven low at the start of a SPI transaction, held low for the duration of the transaction, and driven high after the transaction is complete. During a transaction the master toggles the SPI clock (SCLK) and transmits data on the MOSI pin.

A write operation is initiated by transmitting a 1 for the R/W bit. Then the 8-bit register address, ADDR[7:0] is encoded in the first and second serialized bytes. Data to be written starts in the third serialized byte. The order of the bits is as follows:

Byte 0: R/W,ADDR[6],ADDR[5],ADDR[4],ADDR[3],ADDR[2],ADDR[1],ADDR[0],

Byte 1: ADDR[7],X,X,X,X,X,X,X,

Byte 2: DATA[7],DATA[6],DATA[5],DATA[4],DATA[3],DATA[2],DATA[1],DATA[0].

Multiple bytes of DATA may be transmitted. The X indicates a bit that is ignored by the part. The register address is auto-incremented so that the next clock edges will latch the data for the next register. When desired, the rising edge on CS_B stops the SPI communication.

The FXLS8471Q SPI configuration is as follows:

- Polarity: rising/falling
- Phase: sample/setup
- Order: MSB first

Data is sampled during the rising edge of SCLK and set up during the falling edge of SCLK.

### 6.2.2 SPI READ/WRITE operations

A READ operation is initiated by transmitting a 0 for the R/W bit. Then the 8-bit register address, ADDR[7:0] is encoded in the first and second serialized bytes. Subsequent bits are ignored by the part. The read data is deserialized from the MISO pin.

Similarly a WRITE operation is initiated by transmitting a 1 for the R/W bit. After the first and second serialized bytes multiple-data bytes can be transmitted into consecutive registers, starting from the indicated register address in ADDR[7:0].

An SPI transaction is started by asserting the CS_B pin (high-to-low transition), and ended by deasserting the CS_B pin (low-to-high transition).

| | | | | | |
|---|---|---|---|---|---|
| R/W bit followed by ADDR [6:0] | ADDR[7] followed by 7 “don’t care” bits | Data0* | Data1 | — | Datan |

* Data bytes must be transmitted to the slave (FXLS8471Q) using the MOSI pin by the master when R/W = 1. Data bytes will be transmitted by the slave (FXLS8471Q) to the master using the MISO pin when R/W = 0. The first 2 bytes are always transmitted by the master using the MOSI pin. That is, a transaction is always initiated by master. 1

**Figure 8. SPI single-burst READ/WRITE transaction diagram**

The registers embedded inside FXLS8471Q are accessed through either an $I^2C$, or a SPI serial interface. To enable either interface the VDDIO line must be connected to the interface supply voltage. If VDD is not present and VDDIO is present FXLS8471Q is in shutdown mode and communications on the interface are ignored. If VDDIO is held high, VDD can be powered off and the communications pins will be in a high impedance state. This will allow communications to continue on the bus with other devices.

**Table 10. Serial interface pin descriptions**

| Pin Name | Pin Description |
|---|---|
| VDDIO | Digital interface power |
| SA1/CS_B | $I^2C$ second least significant bit of device address/SPI chip select |
| SCL/SCLK | $I^2C$/SPI serial clock |
| SDA/MOSI | $I^2C$ serial data/SPI master serial data out slave serial data in |
| SA0/MISO | $I^2C$ least significant bit of the device address/SPI master serial data in slave out |

### 6.2.3 $I^2C$/SPI auto detection

**Table 11. $I^2C$/SPI auto detection**

| SA0 | Slave address |
|---|---|
| GND | $I^2C$ |
| VDDIO | $I^2C$ |
| Floating | SPI |

FXLS8471Q employs an interface mode, auto-detection circuit that will select either $I^2C$ or SPI interface mode based on the state of the SA0 pin during power up or when exiting reset. Once set for $I^2C$ or SPI operation, the device will remain in $I^2C$ or SPI mode until the device is reset or powered down and the auto-detection process is repeated. Please note that when SPI interface mode is desired, care must be taken to ensure that no other slave device drives the common SA0/MISO pin during the 1 ms period after a hard or soft reset or powerup event.

### 6.2.4 Power supply sequencing and $I^2C$/SPI mode auto-detection

FXLS8471Q does not have any specific power supply sequencing requirements between VDD and VDDIO voltage supplies to ensure normal operation. To ensure correct operation of the $I^2C$/SPI auto-detection function, VDDIO should be applied before or at the same time as VDD. If this order cannot be maintained, the user should either toggle the RST line or power cycle the VDD rail in order to force the auto-detect function to restart and correctly identify the desired interface. FXLS8471Q will indicate completion of the reset sequence by toggling the INT1 pin from logic high to low to high over a 500 ns period. If the INT1 pin was already low prior to the reset event, it will only go high.

# 7 Modes of Operation

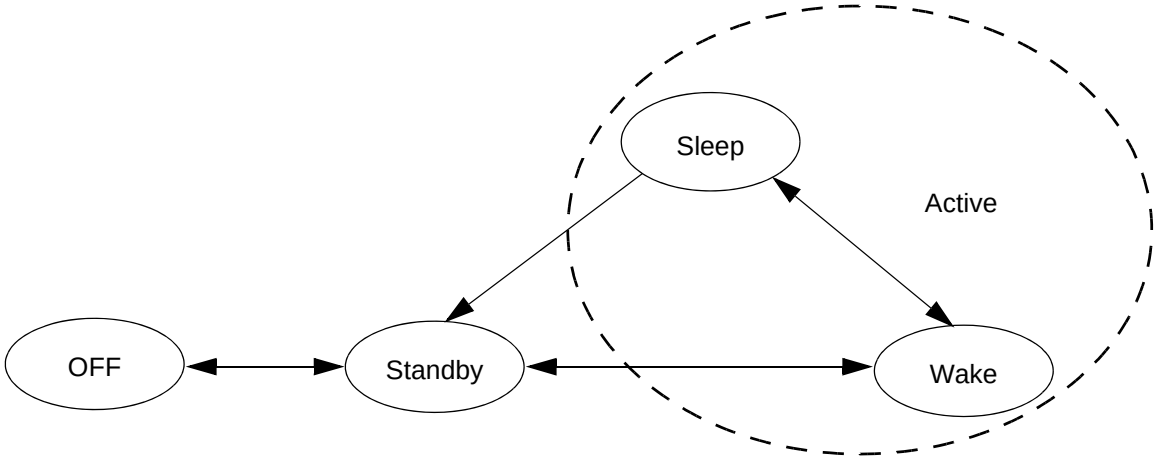



**Figure 9. FXLS8471Q power mode transition diagram**

**Table 12. Mode of operation description**

| Mode | $I^2C$/SPI Bus state | VDD | VDDIO | Function description |
|---|---|---|---|---|
| OFF | Powered down | <1.8 V | VDDIO can be > VDD | The device is powered off. All analog and digital blocks are shutdown. $I^2C$ bus inhibited. |
| Standby | $I^2C$/SPI communication with FXLS8471Q is possible | ON | VDDIO = High<br>VDD = High<br>Active bit is cleared | Only digital blocks are enabled.<br>Analog subsystem is disabled. Internal clocks disabled. |
| Active (Wake/Sleep) | $I^2C$/SPI communication with FXLS8471Q is possible | ON | VDDIO = High<br>VDD = High<br>Active bit is set | All blocks are enabled (digital and analog). |

All register contents are preserved when transitioning from Active to Standby mode, but some registers are reset when transitioning from Standby to Active. These registers are noted in Table 13, “Register Address Map,” on page 25. The Sleep and Wake modes are active modes. For more information on how to use the Sleep and Wake modes and configuring the device to transition between them, please refer to Section 8, “Embedded Functionality” or Freescale application note AN4074.

# 8 Embedded Functionality

FXLS8471Q is a low-power, digital output 3-axis acceleration sensor with both $I^2C$ and SPI interface options. Extensive embedded functionality is provided to detect inertial events at low power, with the ability to notify the host processor via either of the two programmable interrupt pins. The embedded functionality includes:

- 8-bit or 14-bit accelerometer data with an option for high-pass filtered output data
- Four different oversampling options for the output data. The oversampling settings allow the end user to optimize the resolution versus power consumption trade-off in a given application.
- A low-noise accelerometer mode that functions independently of the oversampling modes for even higher resolution
- Low-power auto-wake/sleep function for conserving power in portable battery powered applications
- Accelerometer pulse detection circuit which can be used to detect directional single and double taps
- Accelerometer directional motion and freefall event detection with programmable threshold and debounce time
- Acceleration transient detection with programmable threshold and debounce time. Transient detection can employ either a high-pass filter or use the difference between reference and current sample values.
- Orientation detection with programmable hysteresis for smooth transitions between portrait and landscape orientations
- Accelerometer vector-magnitude change event detection with programmable reference, threshold, and debounce time values

Many different configurations of the above functions are possible to suit the needs of the end application. Separate application notes are available to further explain the different configuration settings and potential use cases.

## 8.1 Factory calibration

FXLS8471Q is factory calibrated for sensitivity and offset on each axis. The trim values are stored in Non-Volatile Memory (NVM). On startup, the trim parameters are read from NVM and applied to the internal compensation circuitry. After mounting the device to the PCB, the user may further adjust the accelerometer offsets through the OFF_X/Y/Z registers. For more information on accelerometer calibration, refer to Freescale application note AN4069.

## 8.2 8-bit or 14-bit data

The measured acceleration data is stored in the OUT_X_MSB, OUT_X_LSB, OUT_Y_MSB, OUT_Y_LSB, OUT_Z_MSB, and OUT_Z_LSB registers as 2's complement 14-bit numbers. The most significant 8-bits of each axis are stored in the OUT_X, Y, Z_MSB registers, so applications needing only 8-bit results simply read these three registers and ignore the OUT_X,Y, Z_LSB registers. To do this, the *f_read* mode bit in CTRL_REG1 must be set.

When the full-scale range is set to 2 *g*, the measurement range is -2 *g* to +1.999 *g*, and each count corresponds to 0.244 mg at ±14-bits resolution. When the full-scale is set to 8 *g*, the measurement range is -8 *g* to +7.996 *g*, and each count corresponds to 0.976 mg. The resolution is reduced by a factor of 64 if only the 8-bit results are used (CTRL_REG1[*f_read*] = 1). For further information on the different data formats and modes, please refer to Freescale application note AN4076.

## 8.3 Low-power modes versus high-resolution modes

FXLS8471Q can be optimized for lower power or higher resolution of the accelerometer output data. High resolution is achieved by setting the lnoise bit in register 0x2A. This improves the resolution (by lowering the noise), but be aware that the dynamic range becomes fixed at ±4 *g* when this bit is set. This will affect all internal embedded functions (scaling of thresholds, etc.) and reduce noise. Another method for improving the resolution of the data is through oversampling. One of the oversampling schemes of the output data can be activated when CTRL_REG2[*mods*] = 2'b10 which will improve the resolution of the output data without affecting the internal embedded functions or fixing the dynamic range.

There is a trade-off between low power and high resolution. Low power can be achieved when the oversampling rate is reduced. When CTRL_REG2[*mods*] = 2'b10, the lowest power is achieved, at the expense of higher noise. In general, the lower the selected ODR and OSR, the lower the power consumption. For more information on how to configure the device in low-power or high-resolution modes and understand the benefits and trade-offs, please refer to Freescale application note AN4075.

## 8.4 Auto-Wake/Sleep mode

FXLS8471Q can be configured to transition between sample rates (with their respective current consumptions) based on the status of the embedded interrupt event generators in the device. The advantage of using the Auto-Wake/Sleep is that the system can automatically transition to a higher sample rate (higher current consumption) when needed but spends the majority of the time in the Sleep mode (lower current) when the device does not require higher sampling rates. Auto-Wake refers to the device being triggered by one of the interrupt event functions to transition to a higher sample rate. This may also interrupt the processor to transition from a sleep mode to a higher power mode.

Sleep mode occurs when none of the enabled interrupt event functions has detected an interrupt within the user-defined, time-out period. The device will then transition to the specified lower sample rate. It may also alert the processor to go into a lower power mode to save power during this period of inactivity. Please refer to AN4074 for more detailed information on configuring the Auto-Wake/Sleep function.

## 8.5 Freefall and Motion event detection

FXLS8471Q integrates a programmable threshold based acceleration detection function capable of detecting either motion or freefall events depending upon the configuration. For further details and examples on using the embedded freefall and motion detection functions, please refer to Freescale application note AN4070.

### 8.5.1 Freefall detection

The detection of “Freefall” involves the monitoring of the X, Y, and Z axes for the condition where the acceleration magnitude is **below** a user-specified threshold for a user-definable amount of time. Typically, the usable threshold ranges are between ±100 mg and ±500 mg.

### 8.5.2 Motion detection

Motion detection is often used to alert the main processor that the device is currently in use. When the acceleration exceeds a set threshold for a set amount of time, the motion interrupt is asserted. A motion can be a fast moving shake or a slow moving tilt. This will depend on the threshold and timing values configured for the event. The motion detection function can analyze static acceleration changes or faster jolts. The timing value is set by a configurable debounce counter. The debounce counter acts like a filter to indicate whether the condition exists for longer than a set amount of time (that is, 100 ms or longer). There is also directional data available in the source register to detect the direction of the motion that generated the interrupt. This is useful for applications such as directional shake or flick detection, and can also assist gesture detection algorithms by indicating that a motion gesture has started.

## 8.6 Transient detection

FXLS8471Q integrates an acceleration transient detection function that incorporates a high-pass filter. Acceleration data goes through the high-pass filter, eliminating the DC tilt offset and low frequency acceleration changes. The high-pass filter cutoff can be set by the user to four different frequencies which are dependent on the selected Output Data Rate (ODR). A higher cutoff frequency ensures that DC and slowly changing acceleration data will be filtered out, allowing only the higher frequencies to pass. The transient detection feature can be used in the same manner as the motion detection by bypassing the high-pass filter. There is an option in the configuration register to do this. This adds more flexibility to cover the various customer use cases.

Many applications use the accelerometer’s static acceleration readings (that is, tilt) which measure the change in acceleration due to gravity only. These functions benefit from acceleration data being filtered with a low-pass filter where high-frequency data is considered noise. However, there are many functions where the accelerometer must analyze dynamic acceleration. Functions such as tap, flick, shake and step counting are based on the analysis of the change in the dynamic acceleration. The transient detection function can be routed to either interrupt pin through bit 5 in CTRL_REG5 register (0x2E). Registers 0x1D – 0x20 are used for configuring the transient detection function. The source register contains directional data to determine the direction of the transient acceleration, either positive or negative. For further information of the embedded transient detection function along with specific application examples and recommended configuration settings, refer to Freescale application note AN4461.

## 8.7 Pulse detection

FXLS8471Q has embedded single/double and directional pulse detection. This function employs several timers for programming the pulse width time and the latency between pulses. The detection thresholds are independently programmable for each axis. The acceleration data input to the pulse detection circuit can be put through both high and low-pass filters, allowing for greater flexibility in discriminating between pulse and tap events. The PULSE_SRC register provides information on the axis, direction (polarity), and single/double event status for the detected pulse or tap. For more information on how to configure the device for pulse detection, please refer to Freescale application note AN4072.

## 8.8 Orientation detection

FXLS8471Q has an embedded orientation detection algorithm with the ability to detect all six orientations. The transition angles and hysteresis are programmable, allowing for a smooth transition between portrait and landscape orientations.

The angle at which the device no longer detects the orientation change is referred to as the “Z-lockout angle”. The device operates down to 29° from the flat position. All angles are accurate to ±2°.

For further information on the orientation detection function refer to Freescale application note, AN4068.

## 8.9 Acceleration Vector-Magnitude detection

FXLS8471Q incorporates an acceleration vector-magnitude change detection block that can be configured to generate an interrupt when the acceleration magnitude exceeds a preset threshold for a programmed debounce time. The function can be configured to operate in absolute or relative modes, and can also act as a wake to sleep/sleep to wake source. This function is useful for detecting acceleration transients when operated in absolute mode, or for detecting changes in orientation when operated in relative mode, refer to Freescale application note AN4692.

# 9 Register Map

**Table 13. Register Address Map**

| Name | Type | Register Address | Auto-Increment Address | | | | Default Hex Value | Comment | |
|---|---|---|---|---|---|---|---|---|---|
| | | | STATUS[*f_mode*] = 00, CTRL_REG1[*f_read*] = 0 | STATUS[*f_mode*] > 00, CTRL_REG1[*f_read*] = 0 | STATUS[*f_mode*] = 00, CTRL_REG1[*f_read*] = 1 | STATUS[*f_mode*] > 00, CTRL_REG1[*f_read*] = 1 | | | |
| STATUS[1][2] | R | 0x00 | 0x01 | | | | 0x00 | Real-time, data-ready status or FIFO status (DR_STATUS or F_STATUS) | |
| OUT_X_MSB[1][2] | R | 0x01 | 0x02 | 0x01 | 0x03 | 0x01 | Data | [7:0] are 8 MSBs of 14-bit sample. | Root pointer to XYZ FIFO data. |
| OUT_X_LSB[1][2] | R | 0x02 | 0x03 | | 0x00 | | Data | [7:2] are 6 LSBs of 14-bit sample | |
| OUT_Y_MSB[1][2] | R | 0x03 | 0x04 | | 0x05 | 0x00 | Data | [7:0] are 8 MSBs of 14-bit sample | |
| OUT_Y_LSB[1][2] | R | 0x04 | 0x05 | | 0x00 | | Data | [7:2] are 6 LSBs of 14-bit sample | |
| OUT_Z_MSB[1][2] | R | 0x05 | 0x06 | | 0x00 | | Data | [7:0] are 8 MSBs of 14-bit sample | |
| OUT_Z_LSB[1][2] | R | 0x06 | 0x00 | | 0x00 | | Data | [7:2] are 6 LSBs of 14-bit sample | |
| Reserved | — | 0x07-0x08 | — | | | | — | Reserved, do not modify | |
| F_SETUP[1][3] | R/W | 0x09 | 0x0A | | | | 0x00 | FIFO setup | |
| TRIG_CFG | R/W | 0x0A | 0x0B | | | | 0x00 | FIFO event trigger configuration register | |
| SYSMOD[1][2] | R | 0x0B | 0x0C | | | | Output | Current system mode | |
| INT_SOURCE[1][2] | R | 0x0C | 0x0D | | | | Output | Interrupt status | |
| WHO_AM_I[1] | R | 0x0D | 0x0E | | | | 0x6A | Device ID | |
| XYZ_DATA_CFG[1][4] | R/W | 0x0E | 0x0F | | | | 0x00 | Acceleration dynamic range and filter enable settings | |
| HP_FILTER_CUTOFF[1][4] | R/W | 0x0F | 0x10 | | | | 0x00 | Pulse detection high-pass and low-pass filter enable bits. High-pass filter cutoff frequency selection | |
| PL_STATUS[1][2] | R | 0x10 | 0x11 | | | | 0x00 | Landscape/Portrait orientation status | |
| PL_CFG[1][4] | R/W | 0x11 | 0x12 | | | | 0x83 | Landscape/Portrait configuration. | |
| PL_COUNT[1][3] | R/W | 0x12 | 0x13 | | | | 0x00 | Landscape/Portrait debounce counter | |
| PL_BF_ZCOMP[1][4] | R/W | 0x13 | 0x14 | | | | 0x00 | Back/Front Trip angle threshold | |
| PL_THS_REG[1][4] | R/W | 0x14 | 0x15 | | | | 0x1A | Portrait to Landscape Trip Threshold angle and hysteresis settings | |
| A_FFMT_CFG[1][4] | R/W | 0x15 | 0x16 | | | | 0x00 | Freefall/Motion function configuration | |
| A_FFMT_SRC[1][2] | R | 0x16 | 0x17 | | | | 0x00 | Freefall/Motion event source register | |
| A_FFMT_THS[1][3] | R/W | 0x17 | 0x18 | | | | 0x00 | Freefall/Motion threshold register | |

**Table 13. Register Address Map (Continued)**

| | | | | | |
|---|---|---|---|---|---|
| A_FFMT_COUNT(1)(3) | R/W | 0x18 | 0x19 | 0x00 | Freefall/Motion debounce counter |
| Reserved | — | 0x19-0x1C | — | — | Reserved, do not modify |
| TRANSIENT_CFG(1)(4) | R/W | 0x1D | 0x1E | 0x00 | Transient function configuration |
| TRANSIENT_SRC(1)(2) | R | 0x1E | 0x1F | 0x00 | Transient event status register |
| TRANSIENT_THS(1)(3) | R/W | 0x1F | 0x20 | 0x00 | Transient event threshold |
| TRANSIENT_COUNT(1)(3) | R/W | 0x20 | 0x21 | 0x00 | Transient debounce counter |
| PULSE_CFG(1)(4) | R/W | 0x21 | 0x22 | 0x00 | Pulse function configuration |
| PULSE_SRC(1)(2) | R | 0x22 | 0x23 | 0x00 | Pulse function source register |
| PULSE_THSX(1)(3) | R/W | 0x23 | 0x24 | 0x00 | X-axis pulse threshold |
| PULSE_THSY(1)(3) | R/W | 0x24 | 0x25 | 0x00 | Y-axis pulse threshold |
| PULSE_THSZ(1)(3) | R/W | 0x25 | 0x26 | 0x00 | Z-axis pulse threshold |
| PULSE_TMLT(1)(4) | R/W | 0x26 | 0x27 | 0x00 | Time limit for pulse detection |
| PULSE_LTCY(1)(4) | R/W | 0x27 | 0x28 | 0x00 | Latency time for second pulse detection |
| PULSE_WIND(1)(4) | R/W | 0x28 | 0x29 | 0x00 | Window time for second pulse detection |
| ASLP_COUNT(1)(4) | R/W | 0x29 | 0x2A | 0x00 | In activity counter setting for Auto-Sleep |
| CTRL_REG1(1)(4) | R/W | 0x2A | 0x2B | 0x00 | System ODR, accelerometer OSR, operating mode |
| CTRL_REG2(1)(4) | R/W | 0x2B | 0x2C | 0x00 | Self-Test, Reset, accelerometer OSR and Sleep mode settings |
| CTRL_REG3(1)(4) | R/W | 0x2C | 0x2D | 0x00 | Sleep mode interrupt wake enable, interrupt polarity, push-pull/open-drain configuration |
| CTRL_REG4(1)(4) | R/W | 0x2D | 0x2E | 0x00 | Interrupt enable register |
| CTRL_REG5(1)(4) | R/W | 0x2E | 0x2F | 0x00 | Interrupt pin (INT1/INT2) map |
| OFF_X(1)(4) | R/W | 0x2F | 0x30 | 0x00 | X-axis accelerometer offset adjust |
| OFF_Y(1)(4) | R/W | 0x30 | 0x31 | 0x00 | Y-axis accelerometer offset adjust |
| OFF_Z(1)(4) | R/W | 0x31 | 0x32 | 0x00 | Z-axis accelerometer offset adjust |
| Reserved | R/W | 0x32-0x5E | — | — | Reserved, do not modify |
| A_VECM_CFG | R/W | 0x5F | 0x60 | 0x00 | Acceleration vector-magnitude configuration register |
| A_VECM_THS_MSB | R/W | 0x60 | 0x61 | 0x00 | Acceleration vector-magnitude threshold MSB |

**Table 13. Register Address Map (Continued)**

| | | | | | |
|---|---|---|---|---|---|
| A_VECM_THS_LSB | R/W | 0x61 | 0x62 | 0x00 | Acceleration vector-magnitude threshold LSB |
| A_VECM_CNT | R/W | 0x62 | 0x63 | 0x00 | Acceleration vector-magnitude debounce count |
| A_VECM_INITX_MSB | R/W | 0x63 | 0x64 | 0x00 | Acceleration vector-magnitude X-axis reference value MSB |
| A_VECM_INITX_LSB | R/W | 0x64 | 0x65 | 0x00 | Acceleration vector-magnitude X-axis reference value LSB |
| A_VECM_INITY_MSB | R/W | 0x65 | 0x66 | 0x00 | Acceleration vector-magnitude Y-axis reference value MSB |
| A_VECM_INITY_LSB | R/W | 0x66 | 0x67 | 0x00 | Acceleration vector-magnitude Y-axis reference value LSB |
| A_VECM_INITZ_MSB | R/W | 0x67 | 0x68 | 0x00 | Acceleration vector-magnitude Z-axis reference value MSB |
| A_VECM_INITZ_LSB | R/W | 0x68 | 0x69 | 0x00 | Acceleration vector-magnitude Z-axis reference value LSB |
| Reserved | — | 0x69-0x72 | — | — | Reserved, do not modify |
| A_FFMT_THS_X_MSB | R/W | 0x73 | 0x74 | 0x00 | X-axis FMT threshold MSB |
| A_FFMT_THS_X_LSB | R/W | 0x74 | 0x75 | 0x00 | X-axis FFMT threshold LSB |
| A_FFMT_THS_Y_MSB | R/W | 0x75 | 0x76 | 0x00 | Y-axis FFMT threshold MSB |
| A_FFMT_THS_Y_LSB | R/W | 0x76 | 0x77 | 0x00 | Y-axis FFMT threshold LSB |
| A_FFMT_THS_Z_MSB | R/W | 0x77 | 0x78 | 0x00 | Z-axis FFMT threshold MSB |
| A_FFMT_THS_Z_LSB | R/W | 0x78 | 0x79 | 0x00 | Z-axis FFMT threshold LSB |
| Reserved | — | 0x79 - 0xFF | — | — | Reserved, do not modify |

1. Register contents are preserved when transitioning from Active to Standby mode.
2. Register contents are reset when transitioning from Standby to Active mode.
3. Register contents can be modified anytime in Standby or Active mode. A write to this register will cause a reset of the corresponding internal system debounce counter.
4. Modification of this register's contents can only occur when device is in Standby mode, except the FS[1:0] bit fields in CTRL_REG1 register.

## NOTE

The auto-increment addressing is only enabled when registers are read using burst-read mode when the device is configured for $I^2C$ or SPI. The auto-increment address is automatically reset to 0x00 in $I^2C$ mode when a stop condition is detected. In SPI mode there is no stop condition and the auto-increment address is not automatically reset to 0x00.

# 10 Registers by Functional Blocks

## 10.1 Device configuration

### 10.1.1 STATUS (0x00) register

**Table 14. STATUS register**

| DR_STATUS or F_STATUS | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 15. STATUS Description**

| Field | Description |
|---|---|
| F_SETUP[*f_mode*] = 2'b00 | register 0x00 → DR_STATUS |
| F_SETUP[*f_mode*] > 2'b00 | register 0x00 → F_STATUS |

The STATUS register aliases allow for the contiguous burst read of both status and current acceleration sample/FIFO data using the auto-increment addressing mechanism in both 8- and 14-bit modes.

### 10.1.2 DR_STATUS (0x00) register

Data-Ready Status when F_SETUP[*f_mode*] = 0x00

This STATUS register provides the acquisition status information on a per-sample basis, and reflects real-time updates to the OUT_X, OUT_Y, and OUT_Z registers.

When the FIFO subsystem data output register driver is disabled (F_SETUP[*f_mode*] = 2'b00), this register indicates the real-time status information of the accelerometer X, Y, and Z axes sample data.

**Table 16. DR_STATUS register**

| zyxow | zow | yow | xow | zyxdr | zdr | ydr | xdr |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 17. DR_STATUS description**

| Field | Description |
|---|---|
| zyxow | *zyxow* is set to 1 whenever new data is acquired before completing the retrieval of the previous set. This event occurs when the content of at least one acceleration data register (that is, OUT_X, OUT_Y, and OUT_Z) has been overwritten. *zyxow* is cleared when the high-bytes of the acceleration data (OUT_X_MSB, OUT_Y_MSB, and OUT_Z_MSB) are read.<br>X, Y, Z-axis data overwrite.<br>0: No data overwrite has occurred<br>1: Previous X, Y, Z data was overwritten by new X, Y, Z data before it was completely read |
| zow | *zow* is set to 1 whenever a new Z-axis acquisition is completed before the retrieval of the previous data. When this occurs the previous data is overwritten. zow is cleared anytime OUT_Z_MSB register is read.<br>Z-axis data overwrite.<br>0: No data overwrite has occurred<br>1: Previous Z-axis data was overwritten by new Z-axis data before it was read |
| yow | *yow* is set to 1 whenever a new Y-axis acquisition is completed before the retrieval of the previous data. When this occurs the previous data is overwritten. yow is cleared anytime OUT_Y_MSB register is read.<br>Y-axis data overwrite.<br>0: No data overwrite has occurred<br>1: Previous Y-axis data was overwritten by new Y-axis data before it was read |
| xow | *xow* is set to 1 whenever a new X-axis acquisition is completed before the retrieval of the previous data. When this occurs the previous data is overwritten. xow is cleared anytime OUT_X_MSB register is read.<br>X-axis data overwrite.<br>0: No data overwrite has occurred<br>1: Previous X-axis data was overwritten by new X-axis data before it was read |

**Table 17. DR_STATUS description (Continued)**

| | |
|---|---|
| zyxdr | *zyxdr* signals that a new acquisition for any of the enabled channels is available. *zyxdr* is cleared when the high-bytes of the acceleration data (OUT_X_MSB, OUT_Y_MSB, OUT_Z_MSB) are read.<br>X, Y, Z-axis new data ready.<br>0: No new set of data ready<br>1: New set of data is ready |
| zdr | *zdr* is set to 1 whenever a new Z-axis data acquisition is completed. *zdr* is cleared anytime the OUT_Z_MSB register is read.<br>Z-axis new data available.<br>0: No new Z-axis data is ready<br>1: New Z-axis data is ready |
| ydr | *ydr* is set to 1 whenever a new Y-axis data acquisition is completed. *ydr* is cleared anytime the OUT_Y_MSB register is read.<br>Y-axis new data available. Default value: 0<br>0: No new Y-axis data ready<br>1: New Y-axis data is ready |
| xdr | *xdr* is set to 1 whenever a new X-axis data acquisition is completed. *xdr* is cleared anytime the OUT_X_MSB register is read.<br>X-axis new data available. Default value: 0<br>0: No new X-axis data ready<br>1: New X-axis data is ready |

### 10.1.3 F_STATUS (0x00) register

FIFO Status when F_SETUP[*f_mode*] = 0x00 > 0x00.

If the FIFO subsystem data output register driver is enabled, the status register indicates the current status information of the FIFO subsystem.

**Table 18. F_STATUS register**

| f_ovf | f_wmrk_flag | f_cnt[5:0] |
|---|---|---|
| 0 | 0 | 0 |

**Table 19. FIFO flag event descriptions**

| f_ovf | f_wmrk_flag | Event description |
|---|---|---|
| 0 | X | No FIFO overflow events detected. |
| 1 | X | FIFO overflow event detected. |
| X | 0 | No FIFO watermark event detected. |
| X | 1 | A FIFO Watermark event was detected indicating that a FIFO sample count greater than watermark value has been reached.<br>If F_SETUP[*f_mode*] = 2'b11, a FIFO trigger event was detected |

The *f_ovf* and *f_wmrk_flag* flags remain asserted while the event source is still active, but the user can clear the FIFO interrupt bit in the interrupt source register (INT_SOURCE) by reading the F_STATUS register. In this case, the INT_SOURCE[*src_fifo*] bit will be set again when the next data sample enters the FIFO.

Therefore the *f_ovf* bit will remain asserted while the FIFO has overflowed and the *f_wmrk_flag* bit will remain asserted while the *f_cnt* value is equal to or greater than then *f_wmrk* value.

**Table 20. FIFO sample count bit description**

| Field | Description |
|---|---|
| f_cnt[5:0] | These bits indicate the number of acceleration samples currently stored in the FIFO buffer. Count 6'b000000 indicates that the FIFO is empty.<br>FIFO sample counter. Default value 6'b000000.<br>(6'b000001 to 6'b100000 indicates 1 to 32 samples stored in FIFO |

### 10.1.4 TRIG_CFG (0x0A) register

FIFO trigger configuration register. After the interrupt flag of the enabled event in TRIG_CFG is set, the FIFO (when configured in Trigger mode) is gated at the time of the interrupt event preventing the further collection of data samples. This allows the host processor to analyze the data leading up to the event detection (up to 32 samples). For detailed information on how to utilize the FIFO and the various trigger events, please see AN4073 available on the Freescale website.

**Table 21. TRIG_CFG register**

| — | — | trig_trans | trig_lndprt | trig_pulse | trig_ffmt | trig_a_vecm | — |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 22. TRIG_CFG bit descriptions**

| Field | Description |
|---|---|
| trig_trans | Transient interrupt FIFO trigger enable. |
| trig_lndprt | Landscape/Portrait orientation interrupt FIFO trigger enable. |
| trig_pulse | Pulse interrupt FIFO trigger enable |
| trig_ffmt | Freefall/motion interrupt FIFO trigger enable |
| trig_a_vecm | Acceleration vector-magnitude FIFO trigger enable. |

### 10.1.5 SYSMOD (0x0B) register

**Table 23. SYSMOD register**

| fgerr | fgt[4:0] | sysmod[1:0] |
|---|---|---|

**Table 24. SYSMOD bit description**

| Field | Description |
|---|---|
| fgerr | FIFO gate error. Default value: 0.<br>0: No FIFO gate error detected.<br>1: FIFO gate error was detected.<br>Emptying the FIFO buffer clears the *fgerr* bit in the SYSMOD register.<br>See CTRL_REG3 [Interrupt CTRL register] (0x2C) for more information on configuring the FIFO Gate function. |
| fgt[4:0] | Number of ODR time units since *fgerr* was asserted. Reset when *fgerr* is cleared |
| sysmod[1:0] | System mode. Default value: 0.<br>00: Standby mode<br>01: Wake mode<br>10: Sleep mode |

The system mode register indicates the current device operating mode. Applications using the Auto-Sleep/Auto-Wake mechanism should use this register to synchronize their application with the device operating mode. The system mode register also indicates the status of the FIFO gate error flag and the time elapsed since the FIFO gate error flag was asserted.

### 10.1.6 INT_SOURCE (0x0C) register

Interrupt source register. The bits that are set (logic ‘1’) indicate which function has asserted its interrupt and conversely bits that are cleared (logic ‘0’) indicate which function has not asserted its interrupt.

Reading the INT_SOURCE register does not clear any interrupt status bits (except for src_a_vecm, see below); the respective interrupt flag bits are reset by reading the appropriate source register for the function that generated the interrupt.

**Table 25. INT_SOURCE register**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| src_aslp | src_fifo | src_trans | src_lndprt | src_pulse | src_ffmt | src_a_vecm | src_drdy |

**Table 26. INT_SOURCE bit descriptions**

| Field | Description |
|---|---|
| src_aslp | Auto-Sleep/Wake interrupt status bit: logic ‘1’ indicates that an interrupt event that can cause a Wake to Sleep or Sleep to Wake system mode transition has occurred and logic ‘0’ indicates that no Wake to Sleep or Sleep to Wake system mode transition interrupt event has occurred.<br>The “Wake-to-Sleep” transition occurs when a period of inactivity that exceeds the user-specified time limit (ASLP_COUNT) has been detected, thus causing the system to transition to a user-specified low ODR setting.<br>A “Sleep-to-Wake” transition occurs when the user-specified interrupt event has awakened the system, thus causing the system to transition to the user-specified higher ODR setting.<br>Reading the SYSMOD register will clear the *src_aslp* bit. |
| src_fifo | FIFO interrupt status bit: logic ‘1’ indicates that a FIFO interrupt event such as an overflow or watermark (F_STATUS[*f_cnt*] = F_STATUS[*f_wmrk*]) event has occurred and logic ‘0’ indicates that no FIFO interrupt event has occurred.<br>This bit is cleared by reading the F_STATUS register. |
| src_trans | Transient interrupt status bit: logic ‘1’ indicates that an acceleration transient value greater than user-specified threshold has occurred. and logic ‘0’ indicates that no transient event has occurred.<br>This bit is asserted whenever TRANSIENT_SRC[*ea*] is asserted and the functional block interrupt has been enabled.<br>This bit is cleared by reading the TRANSIENT_SRC register. |
| src_lndprt | Landscape/Portrait orientation interrupt status bit: logic ‘1’ indicates that an interrupt was generated due to a change in the device orientation status and logic ‘0’ indicates that no change in orientation status was detected.<br>This bit is asserted whenever PL_STATUS[*newlp*] is asserted and the functional block interrupt has been enabled.<br>This bit is cleared by reading the PL_STATUS register. |
| src_pulse | Pulse interrupt status bit: logic ‘1’ indicates that an interrupt was generated due to single- and/or double- pulse event and logic ‘0’ indicates that no pulse event was detected.<br>This bit is asserted whenever PULSE_SRC[*ea*] is asserted and the functional block interrupt has been enabled.<br>This bit is cleared by reading the PULSE_SRC register. |
| src_ffmt | Freefall/motion interrupt status bit: logic ‘1’ indicates that the freefall/motion function interrupt is active and logic ‘0’ indicates that no freefall or motion event was detected.<br>This bit is asserted whenever PULSE_SRC[*ea*] is asserted and the functional block interrupt has been enabled.<br>This bit is cleared by reading the A_FFMT_SRC register. |
| src_a_vecm | Accelerometer vector-magnitude interrupt status bit: logic ‘1’ indicates that an interrupt was generated due to acceleration vector-magnitude function and logic ‘0’ indicates that no interrupt has been generated. This bit is cleared by reading this register (INT_SOURCE). |
| src_drdy | Data-ready interrupt status bit. In acceleration only mode this bit indicates that new accelerometer data is available to read.<br>The *src_drdy* interrupt flag is cleared by reading out the acceleration data from the OUT_X, OUT_Y, and OUT_Z registers.<br>This data can be burst read using a 6-byte burst read starting from the address 0x01 (OUT_X_MSB). |

### 10.1.7 WHO_AM_I (0x0D) register

**Table 27. WHO_AM_I register**

| who_am_i[7:0] |
|---|
| 0x6A |

Device identification register. This register contains the device identifier which is set to 0x6A.

### 10.1.8 CTRL_REG1 (0x2A) register

**NOTE**

Except for Standby mode selection, the device must be in Standby mode to change any of the fields within CTRL_REG1 (0x2A).

**Table 28. CTRL_REG1 register**

| aslp_rate[1:0] | dr[2:0] | lnoise | f_read | active |
|---|---|---|---|---|
| 0 | 3'b001 | 0 | 0 | 0 |

**Table 29. CTRL_REG1 bit descriptions**

| Field | Description |
|---|---|
| aslp_rate[1:0] | Configures the auto-wake sample frequency when the device is in Sleep mode.<br>See Table 30 for more information. |
| dr[2:0] | Output Data Rate (ODR) selection.<br>See Table 31 for more information. |
| lnoise | Reduced noise and full-scale range mode (analog gain times 2).<br>0: Normal mode<br>1: Reduced noise mode; Note that the FSR setting is restricted to a ±4 *g* in this mode (lnoise = 1). |
| f_read | Fast-read mode: Data format is limited to the 8-bit MSB for accelerometer output data. The auto-address pointer will skip over the LSB addresses for each axes sample data when performing a burst read operation.<br>0: Normal mode<br>1: Fast-read mode |
| active | Standby/Active.<br>0: Standby mode<br>1: Active mode |

**Table 30. Sleep mode poll rate description**

| aslp_rate1 | aslp_rate0 | Frequency (Hz) |
|---|---|---|
| 0 | 0 | 50 |
| 0 | 1 | 12.5 |
| 1 | 0 | 6.25 |
| 1 | 1 | 1.56 |

It is important to note that when the device is in Auto-Sleep mode, the system ODR and data rate for all the system functional blocks is overridden by the sleep data rate set by the aslp_rate field..

Table 31 shows the various system output data rates (ODR) that may be selected using the dr[2:0] bits.

**Table 31. System Output Data Rate selection**

| dr2 | dr1 | dr0 | ODR (Hz) | Period (ms) |
|---|---|---|---|---|
| 0 | 0 | 0 | 800.0 | 1.25 |
| 0 | 0 | 1 | 400.0 | 2.5 |
| 0 | 1 | 0 | 200.0 | 5 |
| 0 | 1 | 1 | 100.0 | 10 |
| 1 | 0 | 0 | 50.0 | 20 |
| 1 | 0 | 1 | 12.5 | 80 |
| 1 | 1 | 0 | 6.25 | 160 |
| 1 | 1 | 1 | 1.5625 | 640 |

The *active* bit selects between Standby mode and Active mode. The default value is 0 (Standby mode) on reset.

The *lnoise* bit selects between normal full dynamic range mode and a high sensitivity, low-noise mode. In low-noise mode the maximum signal that can be measured is ±4 g. Note: Any thresholds set above 4 *g* will not be reached.

The *f_read* bit selects between normal and fast-read modes where the auto-increment counter will also skip over the LSB data bytes when *f_read* = 1. All of the acceleration data MSB's can be read out with a single 3-byte burst read starting at the OUT_X_MSB register when f_read = 1.

**NOTE**

The *f_read* bit can only be changed while F_SETUP[*f_mode*] = 0.

### 10.1.9 CTRL_REG2 (0x2B) register

**Table 32. CTRL_REG2 register**

| st | rst | — | smods[1:0] | slpe | mods[1:0] |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |

**Table 33. CTRL_REG2 bit descriptions**

| Field | Description |
|---|---|
| st | The *st* bit activates the accelerometer self-test function. When *st* is set to 1, a change will occur in the device output levels for each axis, allowing the host application to check the functionality of the transducer and measurement signal chain.<br>Self-Test Enable:<br>0: Self-Test disabled<br>1: Self-Test enabled. |
| rst | The *rst* bit is used to initiate a software reset. The reset mechanism can be enabled in both Standby and Active modes. When the *rst* bit is set, the boot mechanism resets all functional block registers and loads the respective internal registers with their default values. Note that the current revision of FXLS8471Q silicon, as identified by a WHO_AM_I value of 0x6A, has an errata associated with the software reset mechanism when the device is operated in SPI mode. Refer to Appendix A.1 for further information and a suggested work-around. After setting the *rst* bit, the system will automatically transition to Standby mode. Therefore, if the system was already in Standby mode, the reboot process will immediately begin; else if the system was in Active mode the boot mechanism will automatically transition the system from Active mode to Standby mode, only then can the reboot process begin. A system reset can also be initiated by pulsing the external RST pin high.<br>The $I^2C$ and SPI communication systems are also reset to avoid corrupted data transactions. The host application should allow 1 ms between issuing a software (setting *rst* bit) or hardware (pulsing RST pin) reset and attempting communications with the device over the $I^2C$ or SPI interfaces. When the SPI interface mode is desired and multiple devices are present on the bus, make sure that the bus is quiet (all slave device MISO pins are high-z) during this 1 ms period to ensure the device does not inadvertently enter $I^2C$ mode. See Section 6.2.3 for further information about the interface mode auto-detection circuit.<br>At the end of the boot process, the *rst* bit is hardware cleared.<br>0: Device reset disabled<br>1: Device reset enabled. |
| smods[1:0] | Sleep mode power scheme selection. See Table 34 for more information. |
| slpe(1) | Auto-Sleep mode enable:<br>0: Auto-Sleep is not enabled<br>1: Auto-Sleep is enabled. |
| mods[1:0] | Accelerometer OSR selection. This setting, along with the ODR selection determines the Active mode power and RMS noise for acceleration measurements. See Table 34 for more information. |

1. When SLPE = 1, a transition between Sleep mode and Wake mode results in a FIFO flush and a reset of internal functional block counters. All functional block status information is preserved except where otherwise indicated. For further information, refer to the CTRL_REG3 register description (*fifo_gate* bit).

**Table 34. CTRL_REG2[mods] oversampling modes**

| (s)mods1 | (s)mods0 | Power mode |
|---|---|---|
| 0 | 0 | Normal |
| 0 | 1 | Low Noise, Low Power |
| 1 | 0 | High Resolution |
| 1 | 1 | Low Power |

**Table 35. Oversampling Ratio versus oversampling mode**

| Accelerometer OSR | | | | |
|---|---|---|---|---|
| ODR (Hz) | Normal | Low Noise, Low Power | High Resolution | Low Power |
| 1.5625 | 128 | 32 | 1024 | 16 |
| 6.25 | 32 | 8 | 256 | 4 |
| 12.5 | 16 | 4 | 128 | 2 |
| 50 | 4 | 4 | 32 | 2 |
| 100 | 4 | 4 | 16 | 2 |
| 200 | 4 | 4 | 8 | 2 |
| 400 | 4 | 4 | 4 | 2 |
| 800 | 2 | 2 | 2 | 2 |

### 10.1.10 CTRL_REG3 [Interrupt Control Register] (0x2C) register

**Table 36. CTRL_REG3 register**

| fifo_gate | wake_trans | wake_lndprt | wake_pulse | wake_ffmt | wake_en_a_vecm | ipol | pp_od |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 37. CTRL_REG3 bit descriptions**

| Field | Description |
|---|---|
| fifo_gate | 0: FIFO gate is bypassed. FIFO is flushed upon the system mode transitioning from Wake-to-Sleep mode or from Sleep-to-Wake mode.<br>1: The FIFO input buffer is blocked when transitioning from “Wake-to-Sleep” mode or from “Sleep-to-Wake” mode until the FIFO is flushed.(1) Although the system transitions from “Wake-to-Sleep” or from “Sleep-to-Wake” the contents of the FIFO buffer are preserved, new data samples are ignored until the FIFO is emptied by the host application.<br>If the *fifo_gate* bit is set to logic ‘1’ and the FIFO buffer is not emptied before the arrival of the next sample, then the SYSMOD[*fgerr*] will be asserted. The SYSMOD[*fgerr*] bit remains asserted as long as the FIFO buffer remains un-emptied.<br>Emptying the FIFO buffer clears the SYS_MOD[*fgerr*] register. |
| wake_tran | 0: Transient function is disabled in Sleep mode<br>1: Transient function is enabled in Sleep mode and can generate an interrupt to wake the system |
| wake_lndprt | 0: Orientation function is disabled Sleep mode.<br>1: Orientation function is enabled in Sleep mode and can generate an interrupt to wake the system |
| wake_pulse | 0: Pulse function is disabled in Sleep mode<br>1: Pulse function is enabled in Sleep mode and can generate an interrupt to wake the system |
| wake_ffmt | 0: Freefall/motion function is disabled in Sleep mode<br>1: Freefall/motion function is enabled in Sleep mode and can generate an interrupt to wake the system |
| wake_en_a_vecm | 0: Acceleration vector-magnitude function is disabled in Sleep mode<br>1: Acceleration vector-magnitude function is enabled in Sleep mode and can generate an interrupt to wake the system |
| ipol | The *ipol* The bit selects the logic polarity of the interrupt signals output on the INT1 and INT2 pins.<br>INT1/INT2 interrupt logic polarity:<br>0: Active low (default)<br>1: Active high |
| pp_od | INT1/INT2 push-pull or open-drain output mode selection. The open-drain configuration can be used for connecting multiple interrupt signals on the same interrupt line but will require an external pullup resistor to function correctly.<br>0: Push-pull (default)<br>1: Open-drain |

1. The FIFO contents are flushed whenever the system ODR changes in order to prevent the mixing of FIFO data from different ODR periods.

### 10.1.11 CTRL_REG4 [Interrupt Enable Register] (0x2D) register

**Table 38. CTRL_REG4 register**

| int_en_aslp | int_en_fifo | int_en_trans | int_en_lndprt | int_en_pulse | int_en_ffmt | int_en_a_vecm | int_en_drdy |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 39. Interrupt Enable Register bit descriptions**

| Field | Description |
|---|---|
| int_en_aslp | Sleep interrupt enable<br>0: Auto-Sleep/Wake interrupt disabled<br>1: Auto-Sleep/Wake interrupt enabled |
| int_en_fifo | FIFO interrupt enable<br>0: FIFO interrupt disabled<br>1: FIFO interrupt enabled |
| int_en_trans | Transient interrupt enable<br>0: Transient interrupt disabled<br>1: Transient interrupt enabled |
| int_en_lndprt | Orientation interrupt enable<br>0: Orientation (Landscape/Portrait) interrupt disabled<br>1: Orientation (Landscape/Portrait) interrupt enabled |
| int_en_pulse | Pulse interrupt enable<br>0: Pulse detection interrupt disabled<br>1: Pulse detection interrupt enabled |
| int_en_ffmt | Freefall/motion interrupt enable<br>0: Freefall/motion interrupt disabled<br>1: Freefall/motion interrupt enabled |
| int_en_a_vecm | Acceleration vector-magnitude interrupt enable<br>0: Acceleration vector-magnitude interrupt disabled<br>1: Acceleration vector-magnitude interrupt enabled |
| int_en_drdy | Data-ready interrupt enable<br>0: Data-ready interrupt disabled<br>1: Data-ready interrupt enabled |

The corresponding functional block interrupt enable bit allows the functional block to route its event detection flag to the system's interrupt controller. The interrupt controller routes the enabled interrupt signals to either the INT1 or INT2 pins depending on the settings made in CTRL_REG5.

### 10.1.12 CTRL_REG5 [Interrupt Routing Configuration Register] (0x2E) register

**Table 40. CTRL_REG5 register**

| int_cfg_aslp | int_cfg_fifo | int_cfg_trans | int_cfg_lndprt | int_cfg_pulse | int_cfg_ffmt | int_cfg_a_vecm | int_cfg_drdy |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 41. Interrupt Routing Configuration bit descriptions**

| Field | Description |
|---|---|
| int_cfg_aslp | Sleep interrupt routing<br>0: Interrupt is routed to INT2 pin<br>1: Interrupt is routed to INT1 pin |
| int_cfg_fifo | FIFO interrupt routing<br>0: Interrupt is routed to INT2 pin<br>1: Interrupt is routed to INT1 pin |
| int_cfg_trans | Transient detection interrupt routing<br>0: Interrupt is routed to INT2 pin<br>1: Interrupt is routed to INT1 pin |
| int_cfg_lndprt | Orientation detection interrupt routing<br>0: Interrupt is routed to INT2 pin<br>1: Interrupt is routed to INT1 pin |
| int_cfg_pulse | Pulse detection interrupt routing<br>0: Interrupt is routed to INT2 pin<br>1: Interrupt is routed to INT1 pin |
| int_cfg_ffmt | Freefall/motion detection interrupt routing<br>0: Interrupt is routed to INT2 pin<br>1: Interrupt is routed to INT1 pin |
| int_cfg_a_vecm | Acceleration vector-magnitude interrupt routing<br>0: Interrupt is routed to INT2 pin<br>1: Interrupt is routed to INT1 pin. |
| int_cfg_drdy | INT1/INT2 configuration.<br>0: Interrupt is routed to INT2 pin<br>1: Interrupt is routed to INT1 pin. |

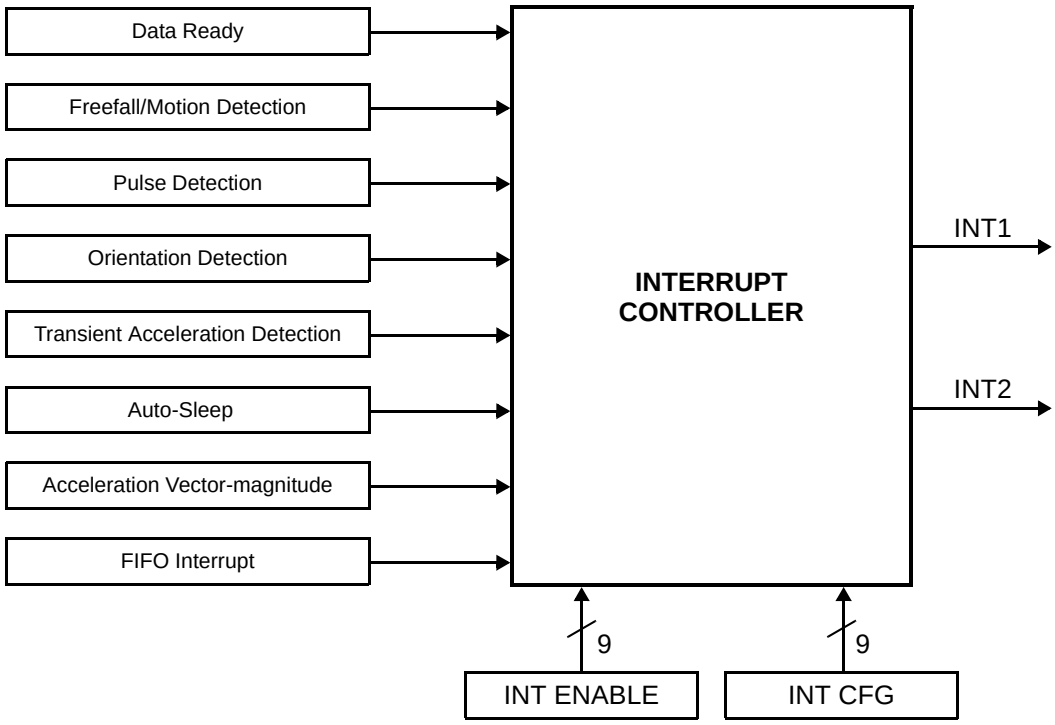



**Figure 10. Interrupt controller block diagram**

The system's interrupt controller uses the corresponding bit field in the CTRL_REG5 register to determine the routing for the INT1 and INT2 interrupt pins. For example, if the *int_cfg_drdy* bit value is logic '0' the functional block's interrupt is routed to INT2, and if the bit value is logic '1' then the interrupt is routed to INT1. All interrupt signals routed to either INT1 or INT2 are logically OR'ed together as illustrated in Figure 11, thus one or more functional blocks can assert an interrupt pin simultaneously; therefore a host application responding to an interrupt should read the INT_SOURCE register to determine the source(s) of the interrupt(s).

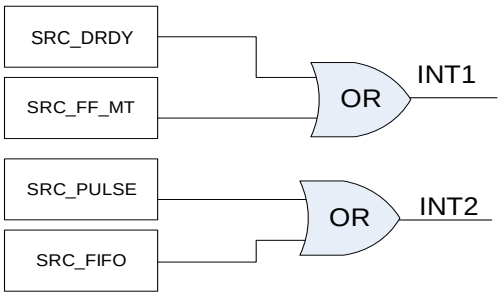



**Figure 11. INT1/INT2 PIN Control Logic**

## 10.2 Auto-Sleep trigger

### 10.2.1 ASLP_COUNT (0x29) register

The ASLP_COUNT register sets the minimum time period of event flag inactivity required to initiate a change from the current active mode ODR value specified in CTRL_REG1[*dr*] to the Sleep mode ODR value specified in CTRL_REG1[*aslp_rate*], provided that CTRL_REG2[*slpe*] = 1.

See Table 45 for functional blocks that may be monitored for inactivity in order to trigger the return-to-sleep event.

**Table 42. ASLP_COUNT register**

| aslp_cnt[7:0] |
|---|
| 8'b00000000 |

**Table 43. ASLP_COUNT bit description**

| Field | Description |
|---|---|
| aslp_cnt[7:0] | See Table 44 for details |

**Table 44. ASLP_COUNT relationship with ODR**

| Output Data Rate (ODR) | Maximum inactivity time (s) | ODR time step (ms) | ASLP_COUNT step (ms) |
|---|---|---|---|
| 800 | 81 | 1.25 | 320 |
| 400 | 81 | 2.5 | 320 |
| 200 | 81 | 5 | 320 |
| 100 | 81 | 10 | 320 |
| 50 | 81 | 20 | 320 |
| 12.5 | 81 | 80 | 320 |
| 6.25 | 81 | 160 | 320 |
| 1.56 | 63 | 640 | 640 |

**Table 45. Sleep/Wake mode gates and triggers**

| Interrupt source | Event restarts time and delays Return-to-Sleep | Event will Wake-from-Sleep |
|---|---|---|
| SRC_FIFO | Yes | No |
| SRC_TRANS | Yes | Yes |
| SRC_LNDPRT | Yes | Yes |
| SRC_PULSE | Yes | Yes |
| SRC_FFMT | Yes | Yes |
| SRC_ASLP | No* | No* |
| SRC_AVECM | Yes | Yes |

* If the *fifo_gate* bit is set to logic '1', the assertion of the SRC_ASLP interrupt does not prevent the system from transitioning to Sleep or from Wake mode; instead it prevents the FIFO buffer from accepting new sample data until the host application flushes the FIFO buffer.

The interrupt sources listed in Table 45 affect the auto-sleep, return to sleep and wake from sleep mechanism only if they have been previously enabled. The functional block event flags that are bypassed while the system is in Auto-Sleep mode are temporary disabled *(see Section 10.1.10, "CTRL_REG3 [Interrupt Control Register] (0x2C) register," on page 35 for more information)* and are automatically re-enabled when the device returns from Auto-Sleep mode (that is, wakes up), except for the data ready function.

If any of the interrupt sources listed under the Return-to-Sleep column is asserted before the sleep counter reaches the value specified in ASLP_COUNT, then all sleep mode transitions are terminated and the internal sleep counter is reset. If none of the interrupts listed under the Return-to-Sleep column are asserted within the time limit specified by the ASLP_COUNT register, the system will transition to the Sleep mode and use the ODR value specified in CTRL_REG1[*aslp_rate*].

If any of the interrupt sources listed under the “Wake-from-Sleep” column is asserted, then the system will transition out of the low sample rate Auto-Sleep mode to the user-specified fast sample rate provided the user-specified wake event function is enabled in register CTRL_REG3.

If the Auto-Sleep interrupt is enabled, a transition from Active mode to Sleep mode and vice-versa will generate an interrupt.

If CTRL_REG3[*fifo_gate*] = 1, transitioning to Auto-Sleep mode will preserve the FIFO contents, set SYSMOD[*fgerr*] (FIFO Gate error), and stop new acquisitions. The system will wait for the FIFO buffer to be emptied by the host application before new samples can be acquired.

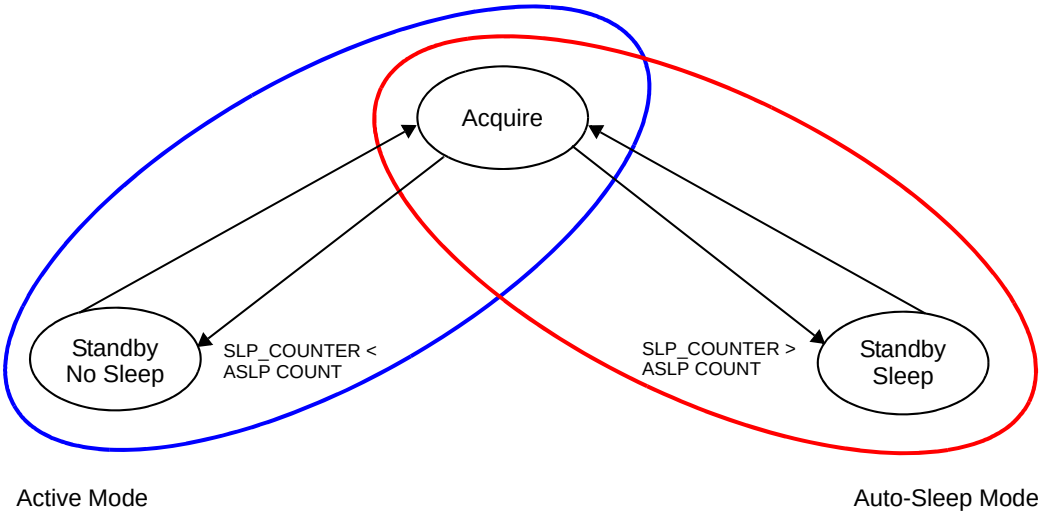



**Figure 12. Auto-Sleep state transition diagram**

## 10.3 Output data registers

### 10.3.1 OUT_X_MSB (0x01), OUT_X_LSB (0x02), OUT_Y_MSB (0x03), OUT_Y_LSB (0x04), OUT_Z_MSB (0x05), OUT_Z_LSB (0x06) registers

These registers contain the X-axis, Y-axis, and Z-axis 14-bit left-justified sample data expressed as 2's complement numbers.

The sample data output registers store the current sample data if the FIFO buffer function is disabled, but if the FIFO buffer function is enabled the sample data output registers then point to the head of the FIFO buffer which contains up to the previous 32 X, Y, and Z data samples.

The data is read out in the following order: Xmsb, Xlsb, Ymsb, Ylsb, Zmsb, Zlsb for CTRL_REG1[*f_read*] = 0, and Xmsb, Ymsb, Zmsb for CTRL_REG1[*f_read*] = 1. Similarly, for CTRL_REG1[*f_read*] = 1, only the MSB's of the acceleration data are read out in the same axis order.

If the CTRL_REG1[*f_read*] bit is set, auto-increment will skip over the LSB registers. This will shorten the data acquisition from 6 bytes to 3 bytes. If the LSB registers are directly addressed, the LSB information can still be read regardless of the CTRL_REG1[*f_read*] register setting.

If the FIFO data output register driver is enabled (F_SETUP[*f_mode*] > 00), register 0x01 points to the head of the FIFO buffer, while registers 0x02, 0x03, 0x04, 0x05, 0x06 return a value of zero when read directly.

The DR_STATUS registers, OUT_X_MSB, OUT_X_LSB, OUT_Y_MSB, OUT_Y_LSB, OUT_Z_MSB, and OUT_Z_LSB are located in the auto-incrementing address range of 0x00 to 0x06, allowing all of the acceleration data to be read in a single-burst read of 6 bytes starting at the OUT_X_MSB register.

**Table 46. OUT_X_MSB register**

| xd[13:6] |
|---|

**Table 47. OUT_X_LSB register**

| xd[5:0] | — | — |
|---|---|---|

**Table 48. OUT_Y_MSB register**

| yd[13:6] |
|---|

**Table 49. OUT_Y_LSB register**

| yd[5:0] | — | — |
|---|---|---|

**Table 50. OUT_Z_MSB register**

| zd[13:6] |
|---|

**Table 51. OUT_Z_LSB register**

| zd[5:0] | — | — |
|---|---|---|

## 10.4 FIFO

### 10.4.1 F_SETUP (0x09) register

**Table 52. F_SETUP register**

| f_mode[1:0] | f_wmrk[5:0] |
|---|---|
| 0 | 6b'000000 |

**Table 53. F_SETUP bit descriptions**

| Field | Description |
|---|---|
| f_mode[1:0](1)(2)(3) | FIFO buffer operating mode.<br>**00:** FIFO is disabled.<br>**01:** FIFO contains the most recent samples when overflowed (circular buffer). Oldest sample is discarded to be replaced by new sample.<br>**10:** FIFO stops accepting new samples when overflowed.<br>**11:** FIFO trigger mode.<br>The FIFO is flushed whenever the FIFO is disabled, during an automatic ODR change (Auto-Wake/Sleep), or on a transition from Standby mode to Active mode.<br>Disabling the FIFO (*f_mode* = 2'b00) resets the F_STATUS[*f_ovf*], F_STATUS[*f_wmrk_flag*], F_STATUS[*f_cnt*] status flags to zero.<br>A FIFO overflow event (that is, F_STATUS[*f_cnt]* = 32) will assert the F_STATUS[*f_ovf*] flag. |
| f_wmrk[5:0](2) | FIFO sample count watermark.<br>These bits set the number of FIFO samples required to trigger a watermark interrupt. A FIFO watermark event flag F_STATUS[*f_wmk_flag*] is raised when FIFO sample count F_STATUS[*f_cnt]* value is equal to or greater than the *f_ wmrk* watermark.<br>Setting the *f_wmrk* to 6'b000000 will disable the FIFO watermark event flag generation.<br>This field is also used to set the number of pre-trigger samples in trigger mode (*f_mode* = 2'b11). |

1. This bit field can be written in Active mode.
2. This bit field can be written in Standby mode.
3. The FIFO mode (*f_mode*) cannot be switched between operational modes (01, 10 and 11).

A FIFO sample count exceeding the watermark event does not stop the FIFO from accepting new data.

The FIFO update rate is dictated by the selected system ODR. In Active mode the ODR is set by CTRL_REG1[*dr*] and when Auto-Sleep is active, the ODR is set by CTRL_REG1[*aslp_rate*] bit fields.

When data is read from the FIFO buffer, the oldest sample data in the buffer is returned and also deleted from the front of the FIFO, while the FIFO sample count is decremented by one. It is assumed that the host application will use the $I^2C$ or SPI burst read transactions to dump the FIFO contents. If the FIFO X, Y, and Z data is not completely read in one burst read transaction, the next read will start at the next FIFO location X-axis data. If the Y or Z data is not read out in the same burst transaction as the X-axis data, it will be lost.

In Trigger mode, the FIFO is operated as a circular buffer and will contain up to the 32 most recent acceleration data samples. The oldest sample is discarded and replaced by the current sample, until a FIFO trigger event occurs. After a trigger event occurs, the FIFO will continue to accept samples only until overflowed, after which point the newest sample data is discarded. For more information on using the FIFO buffer and the various FIFO operating modes, please refer to Freescale application note AN4073.

## 10.5 Sensor data configuration

### 10.5.1 XYZ_DATA_CFG (0x0E) register

The XYZ_DATA_CFG register is used to configure the desired acceleration full-scale range, and also to select whether the output data is passed through the high-pass filter.

**Table 54. XYZ_DATA_CFG register**

| — | — | — | hpf_out | — | — | fs[1:0] |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 55. XYZ_DATA_CFG bit descriptions**

| Field | Description |
|---|---|
| hpf_out | Enable high-pass filter on acceleration output data<br>1: Output data is high-pass filtered<br>0: High-pass filter is disabled. |
| fs[1:0] | Accelerometer full-scale range selection. See Table 56 |

**Table 56.**

| fs[1] | fs[0] | Full-Scale range |
|---|---|---|
| 0 | 0 | ±0.244 mg/LSB |
| 0 | 1 | ±0.488 mg/LSB |
| 1 | 0 | ±0.976 mg/LSB |
| 1 | 1 | Reserved |

## 10.6 High-Pass filter

### 10.6.1 HP_FILTER_CUTOFF (0x0F) register

High-pass filter cutoff frequency setting register.

**Table 57. HP_FILTER_CUTOFF register**

| — | — | pulse_hpf_byp | pulse_lpf_en | — | — | sel[1:0] |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 58. HP_FILTER_CUTOFF bit descriptions**

| Field | Description |
|---|---|
| pulse_hpf_byp | Bypass high-pass filter for pulse processing function<br>0: HPF enabled for pulse processing<br>1: HPF bypassed for pulse processing |
| pulse_lpf_en | Enable low-pass filter for pulse processing function<br>0: LPF disabled for pulse processing<br>1: LPF enabled for pulse processing |
| sel[1:0] | HPF cutoff frequency selection<br>See Table 59. |

**Table 59. HP_FILTER_CUTOFF**

| High-Pass cutoff frequency (Hz) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| ODR (Hz) | sel = 2'b00 | | | | sel = 2'b01 | | | |
| | Normal | LPLN | High resolution | Low power | Normal | LPLN | High resolution | Low power |
| 800 | 16 | 16 | 16 | 16 | 8 | 8 | 8 | 8 |
| 400 | 16 | 16 | 16 | 8 | 8 | 8 | 8 | 4 |
| 200 | 8 | 8 | 16 | 4 | 4 | 4 | 8 | 2 |
| 100 | 4 | 4 | 16 | 2 | 2 | 2 | 8 | 1 |
| 50 | 2 | 2 | 16 | 1 | 1 | 1 | 8 | 0.5 |
| 12.5 | 2 | 0.5 | 16 | 0.25 | 1 | 0.25 | 8 | 0.125 |
| 6.25 | 2 | 0.25 | 16 | 0.125 | 1 | 0.125 | 8 | 0.063 |
| 1.56 | 2 | 0.063 | 16 | 0.031 | 1 | 0.031 | 8 | 0.016 |
| ODR (Hz) | sel = 2'b10 | | | | sel = 2'b11 | | | |
| | Normal | LPLN | High resolution | Low power | Normal | LPLN | High resolution | Low power |
| 800 | 4 | 4 | 4 | 4 | 2 | 2 | 2 | 2 |
| 400 | 4 | 4 | 4 | 2 | 2 | 2 | 2 | 1 |
| 200 | 2 | 2 | 4 | 1 | 1 | 1 | 2 | 0.5 |
| 100 | 1 | 1 | 4 | 0.5 | 0.5 | 0.5 | 2 | 0.25 |
| 50 | 0.5 | 0.5 | 4 | 0.25 | 0.25 | 0.25 | 2 | 0.125 |
| 12.5 | 0.5 | 0.125 | 4 | 0.063 | 0.25 | 0.063 | 2 | 0.031 |
| 6.25 | 0.5 | 0.063 | 4 | 0.031 | 0.25 | 0.031 | 2 | 0.016 |
| 1.56 | 0.5 | 0.016 | 4 | 0.008 | 0.25 | 0.008 | 2 | 0.004 |

## 10.7 Portrait/Landscape Detection

The FXLS8471Q is capable of detecting six orientations: Landscape Left, Landscape Right, Portrait Up, and Portrait Down with Z-lockout feature as well as Face Up and Face Down orientation as shown in Figures 13, 14 and 15. For more details on the meaning of the different user-configurable settings and for example code, please refer to Freescale application note AN4068.

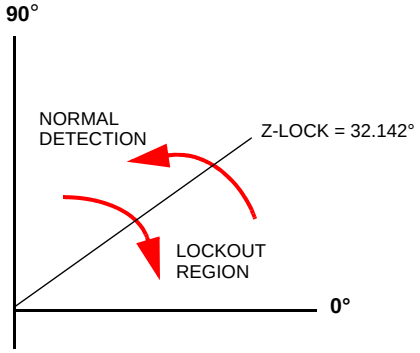



**Figure 13. Illustration of Z-tilt angle lockout transition**

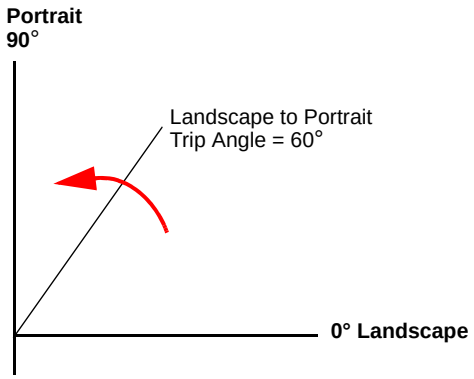



**Figure 14. Illustration of landscape to portrait transition**

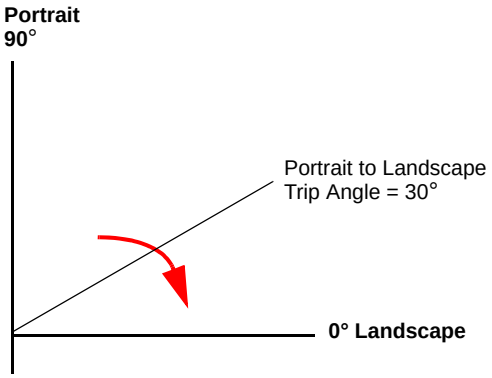



**Figure 15. Illustration of portrait to landscape transition**

### 10.7.1 PL_STATUS (0x10) register

This status register can be read to get updated information on any change in orientation by reading bit 7, or the specifics of the orientation by reading the other bits. For further understanding of Portrait Up, Portrait Down, Landscape Left, Landscape Right, Back and Front orientations please refer to Figure 15. The interrupt is cleared when reading the PL_STATUS register.

**Table 60. PL_STATUS register**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| newlp | lo | — | — | — | lapo[1] | lapo[0] | bafro |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 61. PL_STATUS bit descriptions**

| Field | Description |
|---|---|
| newlp | Landscape/Portrait status change flag.<br>0: No change<br>1: BAFRO and/or LAPO and/or Z-tilt lockout value has changed |
| lo | Z-tilt angle lockout.<br>0: Lockout condition has not been detected.<br>1: Z-tilt lockout trip angle has been exceeded. Lockout condition has been detected. |
| lapo[1:0](1) | Landscape/Portrait orientation.<br>00: Portrait up: equipment standing vertically in the normal orientation<br>01: Portrait down: equipment standing vertically in the inverted orientation<br>10: Landscape right: equipment is in landscape mode to the right<br>11: Landscape left: equipment is in landscape mode to the left. |
| bafro | Back or front orientation.<br>0: Front: equipment is in the front facing orientation.<br>1: Back: equipment is in the back facing orientation. |

1. The default powerup state is *bafro*(Undefined), *lapo*(Undefined), and no lockout for orientation function.

The *newlp* bit is set to 1 after the first orientation detection after a Standby to Active transition, and whenever a change in *lo*, *bafro*, or *lapo* occurs. The *newlp* bit is cleared anytime the PL_STATUS register is read. *lapo, bafro* and *lo* continue to change when *newlp* is set. The current orientation is locked if the absolute value of the acceleration experienced on any of the three axes is greater than 1.25 *g*.

### 10.7.2 PL_CFG (0x11) register

This register enables the Portrait/Landscape function and sets the behavior of the debounce counter.

**Table 62. PL_CFG register**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| dbcntm | pl_en | — | — | — | — | — | — |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 63. PL_CFG bit descriptions**

| Field | Description |
|---|---|
| dbcntm | Debounce counter mode selection.<br>0: Decrements debounce whenever condition of interest is no longer valid.<br>1: Clears counter whenever condition of interest is no longer valid. |
| pl_en | Portrait/Landscape detection enable.<br>0: Portrait/Landscape detection is disabled.<br>1: Portrait/Landscape detection is enabled. |

### 10.7.3 PL_COUNT (0x12) register

This register sets the debounce count for the orientation state transition. The minimum debounce latency is determined by the system ODR value and the value of the PL_COUNT register. Any change to the system ODR or a transition from Active to Standby (or vice-versa) resets the internal landscape/portrait internal debounce counters.

**Table 64. PL_COUNT register**

| |
|---|
| dbnce[7:0] |
| 8'b00000000 |

**Table 65. PL_Count Relationship with the ODR**

| ODR (Hz) | Max time range (s) | | | | Time step (ms) | | | |
|---|---|---|---|---|---|---|---|---|
| | Normal | LPLN | High resolution | Low power | Normal | LPLN | High resolution | Low power |
| 800 | 0.319 | 0.319 | 0.319 | 0.319 | 1.25 | 1.25 | 1.25 | 1.25 |
| 400 | 0.638 | 0.638 | 0.638 | 0.638 | 2.5 | 2.5 | 2.5 | 2.5 |
| 200 | 1.28 | 1.28 | 0.638 | 1.28 | 5 | 5 | 2.5 | 5 |
| 100 | 2.55 | 2.55 | 0.638 | 2.55 | 10 | 10 | 2.5 | 10 |
| 50 | 5.1 | 5.1 | 0.638 | 5.1 | 20 | 20 | 2.5 | 20 |
| 12.5 | 5.1 | 20.4 | 0.638 | 20.4 | 20 | 80 | 2.5 | 80 |
| 6.25 | 5.1 | 20.4 | 0.638 | 40.8 | 20 | 80 | 2.5 | 160 |
| 1.56 | 5.1 | 20.4 | 0.638 | 40.8 | 20 | 80 | 2.5 | 160 |

### 10.7.4 PL_BF_ZCOMP (0x13) register

Back/Front and Z-tilt angle compensation register

**Table 66. PL_BF_ZCOMP register**

| bkfr[1:0] | — | — | — | zlock[2:0] |
|---|---|---|---|---|
| 2'b10 | 0 | 0 | 0 | 3'b100 |

**Table 67. PL_BF_ZCOMP bit descriptions**

| Field | Description |
|---|---|
| zlock[2:0] | Z-lock angle threshold. range is from approximately 13° to 44°. Step size is approximately 4°. See Table 68 for more information.<br>Default value: **0x04** → ~**28°**.<br>Maximum value: **0x07** → **~44°**. |
| bkfr[1:0] | Back/front trip angle threshold. See Table 69 for more information.<br>Default: **2'b10** → **±70°**. Step size is 5°.<br>Range: ±(65° to 80°). |

**Table 68. Z-lockout angle definitions**

| zlock | Resultant angle (min) for positions between Landscape and Portrait | Resultant angle (max) for ideal Landscape or Portrait |
|---|---|---|
| 0x00 | 13.6° | 14.5° |
| 0x01 | 17.1° | 18.2° |
| 0x02 | 20.7° | 22.0° |
| 0x03 | 24.4° | 25.9° |
| 0x04 | 28.1° | 30.0° |
| 0x05 | 32.0° | 34.2° |
| 0x06 | 36.1° | 38.7° |
| 0x07 | 40.4° | 43.4° |

**Table 69. Back/Front orientation definitions**

| bkfr | Back → Front Transition | Front → Back Transition |
|---|---|---|
| 00 | Z < 80° or Z > 280° | Z > 100° and Z < 260° |
| 01 | Z < 75° or Z > 285° | Z > 105° and Z < 255° |
| 10 | Z < 70° or Z > 290° | Z > 110° and Z < 250° |
| 11 | Z < 65° or Z > 295° | Z > 115° and Z < 245° |

### 10.7.5 PL_THS_REG (0x14) register

Portrait to landscape trip threshold registers.

**Table 70. PL_THS_REG register**

| pl_ths[4:0] | hys[2:0] |
|---|---|
| 5'b01000 | 3'b100 |

**Table 71. Threshold angle lookup table**

| pl_ths[4:0] value | Threshold angle (approx.) |
|---|---|
| 0x07 | 15° |
| 0x09 | 20° |
| 0x0C | 30° |
| 0x0D | 35° |
| 0x0F | 40° |
| 0x10 | 45° |
| 0x13 | 55° |
| 0x14 | 60° |
| 0x17 | 70° |
| 0x19 | 75° |

**Table 72. Trip angles versus hysteresis settings**

| hys[2:0] value | Landscape to Portrait trip angle | Portrait to Landscape trip angle |
|---|---|---|
| 0 | 45° | 45° |
| 1 | 49° | 41° |
| 2 | 52° | 38° |
| 3 | 56° | 34° |
| 4 | 59° | 31° |
| 5 | 62° | 28° |
| 6 | 66° | 24° |
| 7 | 69° | 21° |

**Table 73. Portrait/Landscape ideal orientation definitions**

| Position | Description |
|---|---|
| PU | y ~ -1 *g*, x ~ 0 |
| PD | y ~ +1 *g*, x ~ 0 |
| LR | y ~ 0, x ~ +1 *g* |
| LL | y ~ 0, x ~ -1 *g* |

## 10.8 Freefall and Motion detection

The Freefall/Motion detection block can be configured to detect low-*g* (freefall) or high-*g* (motion) events utilizing the A_FFMT_CFG[*a_ffmt_oae*] bit.

In low-*g* detect mode (A_FFMT_CFG[*a_ffmt_oae*] = 0) a low-*g* condition will need to occur on all enabled axes (ex. X, Y and Z) for the A_FFMT_SRC[*a_ffmt_ea*] bit to be affected. And, in high-*g* detect mode (A_FFMT_CFG[*a_ffmt_oae*] = 1) a high-*g* condition occurring in any of the enabled axes (ex. X, Y or Z) will suffice to affect the A_FFMT_SRC[*a_ffmt_ea*] bit.

The detection threshold(s) are programed in register 0x17 (A_FFMT_THS) for common threshold operation, and 0x73-0x78 (A_FFMT_THS_X/Y/Z) for individual axis threshold operation.

A_FFMT_CFG[*a_ffmt_ele*] bit determines the behavior of A_FFMT_SRC[*a_ffmt_ea*] bit in response to the desired acceleration event (low-g/high-g). When A_FFMT_CFG[*a_ffmt_ele*] = 1, the freefall or motion event is latched and the

A_FFMT_SRC[*a_ffmt_ea*] flag can only be cleared by reading the A_FFMT_SRC register. When A_FFMT_CFG[*a_ffmt_ele*] = 0, freefall or motion events are not latched, and the A_FFMT_SRC[*a_ffmt_ea*] bit reflects the real-time status of the event detection.

A_FFMT_THS[*a_ffmt_dbcntm*] bit determines the debounce filtering behavior of the logic which sets the A_FFMT_SRC[*a_ffmt_ea*] bit. See Figure 17 for details.

It is possible to enable/disable each axis used in the freefall/motion detection function by configuring bits A_FFMT_CFG[*a_ffmt_xefe*], A_FFMT_CFG[*a_ffmt_yefe*], and A_FFMT_CFG[*a_ffmt_zefe*].

The freefall/motion detection function has the option to use a common 7-bit unsigned threshold for each of the X, Y, Z axes, or individual unsigned 13-bit thresholds for each axis. When A_FFMT_THS_X_MSB[*a_ffmt_ths_xyz_en*] = 0, the 7-bit threshold value stored in register 0x17 is used as a common 7-bit threshold for the X, Y, and Z axes. When *a_ffmt_ths_xyz_en* = 1, each axis may be programmed with an individual 13-bit threshold (stored in the A_FFMT_X/Y/Z MSB and LSB registers).

### 10.8.1 A_FFMT_CFG (0x15) register

Freefall/motion configuration register.

**Table 74. A_FFMT_CFG register**

| a_ffmt_ele | a_ffmt_oae | a_ffmt_zefe | a_ffmt_yefe | a_ffmt_xefe | — | — | — |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 75. A_FFMT_CFG bit descriptions**

| Field | Description |
|---|---|
| a_ffmt_ele | *a_ffmt_ele* denotes whether the enabled event flag will be latched in the A_FFMT_SRC register or the event flag status in the A_FFMT_SRC will indicate the real-time status of the event. If *a_ffmt_ele* bit is set to a logic '1', then the event flags are frozen when the *a_ffmt_ea* bit gets set, and are cleared by reading the A_FFMT_SRC source register.<br>Default value: 0<br>0: Event flag latch disabled<br>1: Event flag latch enabled |
| a_ffmt_oae | *a_ffmt_oae* bit allows the selection between motion (logical OR combination of high-*g* X, Y, Z-axis event flags) and freefall (logical AND combination of low-*g* X, Y, Z-axis event flags) detection.<br>Motion detect/freefall detect logic selection.<br>Default value: 0 (freefall flag)<br>0: Freefall flag (Logical AND combination of low-*g* X, Y, Z-axis event flags)<br>1: Motion flag (Logical OR combination of high-*g* X, Y, Z event flags) |
| a_ffmt_zefe | *a_ffmt_zefe* enables the detection of a high- or low-*g* event when the measured acceleration data on Z-axis is above/below the threshold set in the A_FFMT_THS register. If the *a_ffmt_ele* bit is set to logic '1' in the A_FFMT_CFG register, new event flags are blocked from updating the A_FFMT_SRC register.<br>Default value: 0<br>0: Event detection disabled<br>1: Raise event flag on measured Z-axis acceleration above/below threshold. |
| a_ffmt_yefe | *a_ffmt_yefe* enables the detection of a high- or low-*g* event when the measured acceleration data on Y-axis is above/below the threshold set in the A_FFMT_THS register. If the *a_ffmt_ele* bit is set to logic '1' in the A_FFMT_CFG register, new event flags are blocked from updating the A_FFMT_SRC register.<br>Default value: 0<br>0: Event detection disabled<br>1: Raise event flag on measured Y-axis acceleration above/below threshold. |
| a_ffmt_xefe | *a_ffmt_xefe* enables the detection of a high- or low-*g* event when the measured acceleration data on X-axis is above/below the threshold set in the A_FFMT_THS register. If the *a_ffmt_ele* bit is set to logic '1' in the A_FFMT_CFG register, new event flags are blocked from updating the A_FFMT_SRC register.<br>Default value: 0<br>0: Event detection disabled<br>1: Raise event flag on measured X-axis acceleration above/below threshold. |

### 10.8.2 A_FFMT_SRC (0x16) register

Freefall/motion source register. Read-only register.

This register keeps track of the acceleration event which is triggering (or has triggered, in case of A_FFMT_CFG[*a_ffmt_ele*] = 1) the event flag. In particular A_FFMT_SRC[*a_ffmt_ea*] is set to a logic '1' when the logical combination of acceleration event flags specified in A_FFMT_CFG register is true. This bit is used in combination with the values in CTRL_REG4[*int_en_ffmt*] and CTRL_REG5[*int_cfg_ffmt*] register bits to generate the freefall/motion interrupts.

**Table 76. A_FFMT_SRC register**

| a_ffmt_ea | — | a_ffmt_zhe | a_ffmt_zhp | a_ffmt_yhe | a_ffmt_yhp | a_ffmt_xhe | a_ffmt_xhp |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 77. A_FFMT_SRC bit descriptions**

| Field | Description |
|---|---|
| a_ffmt_ea | Event active flag. Default value: 0<br>0: No event flag has been asserted<br>1: One or more event flag has been asserted. See the description of the A_FFMT_CFG[*a_ffmt_oae*] bit to determine the effect of the 3-axis event flags on the *a_ffmt_ea* bit. |
| a_ffmt_zhe | Z-high event flag. Default value: 0<br>0: Event detected<br>1: Z-high event has been detected<br>This bit always reads zero if the *a_ffmt_zefe* control bit is set to zero |
| a_ffmt_zhp | Z-high event polarity flag. Default value: 0<br>0: Z event was positive *g*<br>1: Z event was negative *g*<br>This bit read always zero if the *a_ffmt_zefe* control bit is set to zero |
| a_ffmt_yhe | Y-high event flag. Default value: 0<br>0: No event detected<br>1: Y-high event has been detected<br>This bit read always zero if the *a_ffmt_yefe* control bit is set to zero |
| a_ffmt_yhp | Y-high event polarity flag. Default value: 0<br>0: Y event detected was positive *g*<br>1: Y event was negative *g*<br>This bit always reads zero if the *a_ffmt_yefe* control bit is set to zero |
| a_ffmt_xhe | X-high event flag. Default value: 0<br>0: No event detected<br>1: X-high event has been detected<br>This bit always reads zero if the *a_ffmt_xefe* control bit is set to zero |
| a_ffmt_xhp | X-high event polarity flag. Default value: 0<br>0: X event was positive *g*<br>1: X event was negative *g*<br>This bit always reads zero if the *a_ffmt_xefe* control bit is set to zero |

### 10.8.3 A_FFMT_THS (0x17), A_FFMT_ THS_X_MSB (0x73), A_FFMT_THS_X_LSB (0x74), A_FFMT_THS_Y_MSB (0x75), A_FFMT_THS_Y_LSB (0x76), A_FFMT_THS_Z_MSB (0x77), A_FFMT_THS_Z_LSB (0x78) registers

Freefall/motion detection threshold registers.

**Table 78. A_FFMT_THS (0x17) register**

| a_ffmt_dbcntm | ths[6:0] |
|---|---|
| 0 | 7'b0000000 |

**Table 79. A_FFMT_THS (0x17) bit descriptions**

| Field | Description |
|---|---|
| a_ffmt_dbcntm | The ASIC uses *a_ffmt_dbcntm* to set the acceleration FFMT debounce counter clear mode independent of the value of the *a_ffmt_ths_xyz_en*.<br>*a_ffmt_dbcntm* bit configures the way in which the debounce counter is reset when the inertial event of interest is momentarily not true.<br>When *a_ffmt_dbcntm* bit is a logic '1', the debounce counter is cleared to 0 whenever the inertial event of interest is no longer true (part b, Figure 17) while if the *a_ffmt_dbcntm* bit is set to logic '0' the debounce counter is decremented by 1 whenever the inertial event of interest in longer true<br>(part c, Figure 17) until the debounce counter reaches 0 or the inertial event of interest become active.<br>The decrementing of the debounce counter acts to filter out irregular spurious events which might impede the correct detection of inertial events. |
| ths[6:0] | Freefall/motion detection threshold: default value: 7'b0000000. Resolution is fixed at 63 mg/LSB. |

**Table 80. A_FFMT_THS_X_MSB (0x73) register**

| a_ffmt_ths_xyz_en | a_ffmt_ths_x[12:6] |
|---|---|
| 0 | 7'b0000000 |

**Table 81. A_FFMT_THS_X_MSB (0x73) bit descriptions**

| Field | Description |
|---|---|
| a_ffmt_ths_xyz_en | For *a_ffmt_ths_xyz_en* = 0 the ASIC uses the *ffmt_ths*[6:0] value located in register x17[6:0] as a common threshold for the X, Y, and Z-axis acceleration detection. The common unsigned 7-bit acceleration threshold has a fixed resolution of 63 mg/LSB, with a range of 0-127 counts.<br>For *a_ffmt_ths_xyz_en* = 1 the ASIC ignores the common 7-bit G_FFMT_THS value located in register x17 when executing the FFMT function, and the following independent threshold values are used for each axis:<br>A_FFMT_THS_X_MSB and A_FFMT_THS_X_LSB are used for the X-axis acceleration threshold,<br>A_FFMT_THS_Y_MSB and A_FFMT_THS_Y_LSB for the Y-axis acceleration threshold,<br>A_FFMT_THS_Z_MSB and A_FFMT_THS_Z_LSB for the Z-axis acceleration threshold.<br>The A_FFMT_THS_X/Y/Z thresholds are 13-bit unsigned values that have the same resolution as the accelerometer output data determined by XYZ_DATA_CFG fs [1:0]. The *a_ffmt_ths_xyz_en* and *a_ffmt_trans_ths_en* bits must not be enabled simultaneously. |
| a_ffmt_ths_x[12:6] | 7-bit MSB of X-axis acceleration threshold |

**Table 82. A_FFMT_THS_X_LSB (0x74) register**

| a_ffmt_ths_x[5:0] | — | — |
|---|---|---|
| 6'b000000 | 0 | 0 |

**Table 83. A_FFMT_THS_Y_MSB (0x75) register**

| a_ffmt_trans_ths_en | a_ffmt_ths_y[12:6] |
|---|---|
| 0 | 7'b0000000 |

**Table 84. A_FFMT_THS_Y_LSB (0x76) register**

| a_ffmt_ths_y[5:0] | — | — |
|---|---|---|
| 6'b000000 | 0 | 0 |

**Table 85. A_FFMT_THS_Z_MSB (0x77) register**

| | |
|---|---|
| — | a_ffmt_ths_z[12:6] |
| 0 | 7’b0000000 |

**Table 86. A_FFMT_THS_Z_LSB (0x78) register**

| | | |
|---|---|---|
| a_ffmt_ths_z[5:0] | — | — |
| 6’b000000 | 0 | 0 |

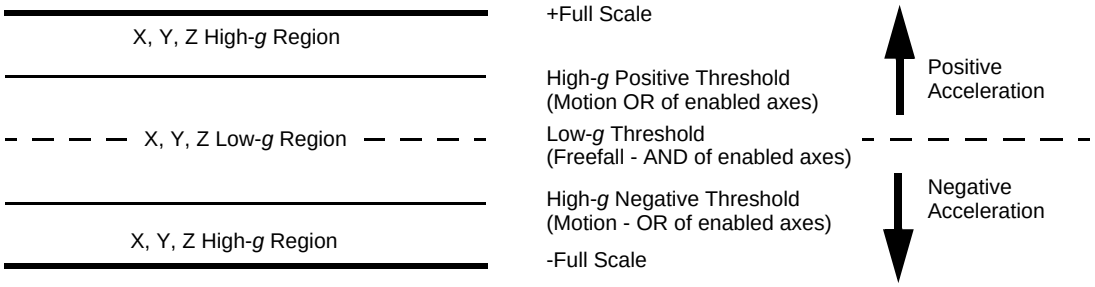



**Figure 16. A_FFMT_THS high and low-*g* level**

A_FFMT_THS contains the unsigned 7-bit threshold value used by the freefall/motion detection functional block and is used to detect either low-*g* (freefall) or high-*g* (motion) events depending on the setting of G_FFMT_CFG[*f_ffmt_oae*]. If *g_ffmt_oae* = 0, the event is detected when the absolute value of all the enabled axes are below the threshold value. When *g_ffmt_oae* = 1, the event is detected when the absolute value of any of the enabled axes is above the threshold value (see Figure 16 for an illustration of the freefall/motion event detection thresholds). If A_FFMT_THS_X_MSB[*a_ffmt_ths_xyz_en*] = 1, the behavior is identical, except that each axis may be programmed with an individual 13-bit threshold (stored in the A_FFMT_X/Y/Z MSB and LSB registers).

### 10.8.4 A_FFMT_COUNT (0x18) register

Debounce count register for freefall/motion detection events

This register sets the number of debounce counts for acceleration sample data matching the user-programmed conditions for either a freefall or motion detection event required before the interrupt is triggered.

**Table 87. A_FFMT_COUNT register**

| |
|---|
| a_ffmt_count[7:0] |
| 8’b00000000 |

**Table 88. A_FFMT_COUNT bit description**

| Field | Description |
|---|---|
| a_ffmt_count[7:0] | *a_ffmt_count* defines the minimum number of debounce sample counts required for the detection of a freefall or motion event. A_FFMT_THS[*ffmt_dbcntm*] determines the behavior of the counter when the condition of interest is momentarily not true. |

When the internal debounce counter reaches the A_FFMT_COUNT value a freefall/motion event flag is set. The debounce counter will never increase beyond the A_FFMT_COUNT value. The time step used for the debounce sample count depends on the ODR chosen (see Table 89).

**Table 89. A_FFMT_COUNT relationship with the ODR**

| ODR (Hz) | Max time range (s) | | | | Time step (ms) | | | |
|---|---|---|---|---|---|---|---|---|
| | Normal | LPLN | High resolution | Low power | Normal | LPLN | High resolution | LP |
| 800 | 0.319 | 0.319 | 0.319 | 0.319 | 1.25 | 1.25 | 1.25 | 1.25 |
| 400 | 0.638 | 0.638 | 0.638 | 0.638 | 2.5 | 2.5 | 2.5 | 2.5 |
| 200 | 1.28 | 1.28 | 0.638 | 1.28 | 5 | 5 | 2.5 | 5 |
| 100 | 2.55 | 2.55 | 0.638 | 2.55 | 10 | 10 | 2.5 | 10 |
| 50 | 5.1 | 5.1 | 0.638 | 5.1 | 20 | 20 | 2.5 | 20 |
| 12.5 | 5.1 | 20.4 | 0.638 | 20.4 | 20 | 80 | 2.5 | 80 |
| 6.25 | 5.1 | 20.4 | 0.638 | 40.8 | 20 | 80 | 2.5 | 160 |
| 1.56 | 5.1 | 20.4 | 0.638 | 40.8 | 20 | 80 | 2.5 | 160 |

For example, an ODR of 100 Hz and a A_ FFMT_COUNT value of 15 would result in minimum debounce response time of 150 ms.

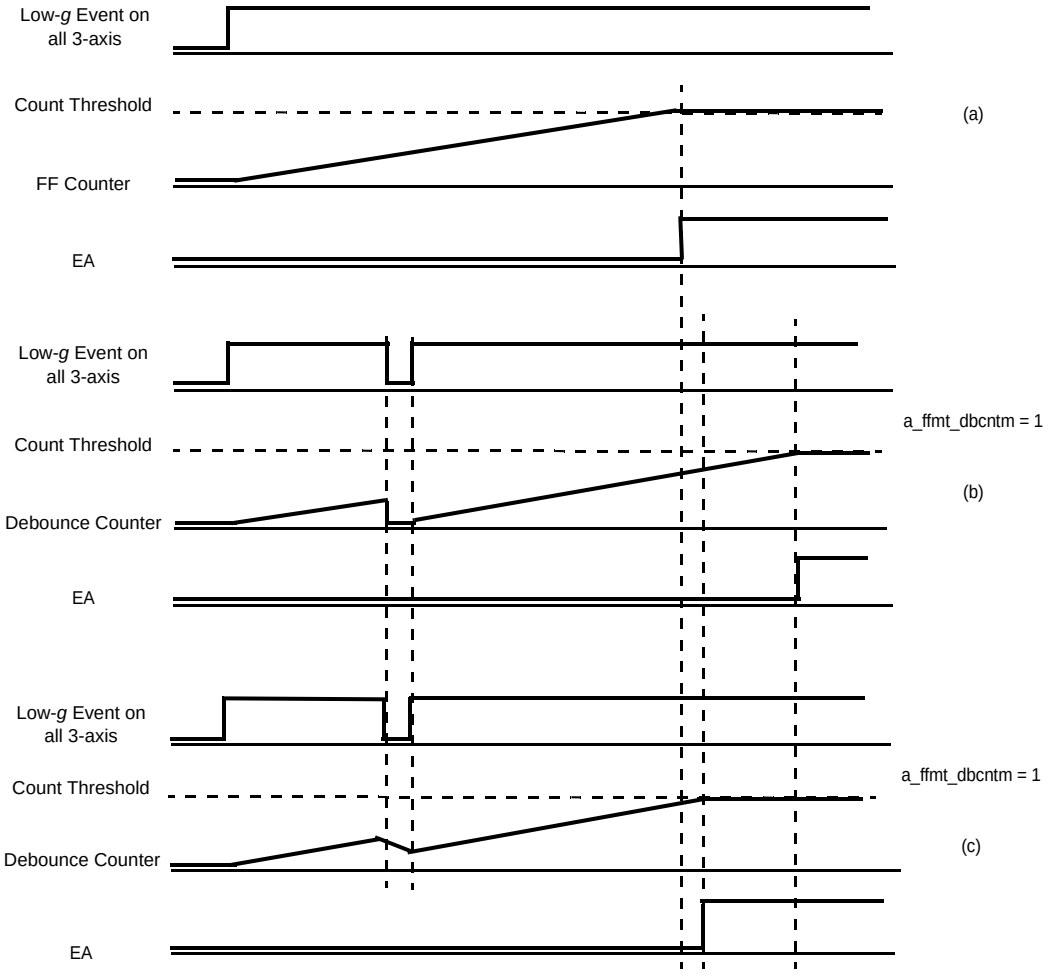



**Figure 17. Behavior of the A_FFMT debounce counter in relation to the a_ffmt_dbcntm setting**

## 10.9 Accelerometer vector-magnitude function

The accelerometer vector-magnitude function is an inertial event detection function available to assist host software algorithms in detecting motion events.

If $\sqrt{(a_x_out - a_x_ref)^2 + (a_y_out - a_y_ref)^2 + (a_z_out - a_z_ref)^2}$ > A_VECM_THS for a time period greater than the value stored in A_VECM_CNT, the vector-magnitude change event flag is triggered.

a_x_out, a_y_out, and a_z_out are the current accelerometer output values, and a_x_ref, a_y_ref, and a_z_ref are the reference values stored internally in the ASIC for each axis or in A_VECM_INIT_X/Y/Z registers if A_VECM_CFG[*a_vecm_initm*] is set.

Please note that the x_ref, y_ref, and z_ref values are not directly visible to the host application through the register interface. Please refer to Freescale application note 4458.

### 10.9.1 A_VECM_CFG (0x5F) register

**Table 90. A_VECM_CFG register**

| — | a_vecm_ele | a_vecm_initm | a_vecm_updm | a_vecm_en | — | — | — |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 91. A_VECM_CFG bit descriptions**

| Field | Description |
|---|---|
| a_vecm_ele | Control bit *a_vecm_ele* defines the event latch enable mode. Event latching is disabled for *a_vecm_ele* = 0. In this case, the vector-magnitude interrupt flag is in updated real-time and is cleared when the condition for triggering the interrupt is no longer true. The setting and clearing of the event flag is controlled by the A_VECM_CNT register's programmed debounce time.<br>For *a_vecm_ele* = 1, the interrupt flag is latched in and held until the host application reads the INT_SOURCE register (0x0C). |
| a_vecm_initm | Control bit *a_vecm_initm* defines how the initial reference values (x_ref, y_ref, and z_ref) are chosen.<br>For *a_vecm_initm* = 0 the function uses the current x/y/z accelerometer output data at the time when the vector magnitude function is enabled.<br>For *a_vecm_initm* = 1 the function uses the data from A_VECM_INIT_X/Y/Z registers as the initial reference values. |
| a_vecm_updm | Control bit *a_vecm_updm* defines how the reference values are updated once the vector-magnitude function has been triggered.<br>For *a_vecm_updm* = 0, the function updates the reference value with the current x, y, and z accelerometer output data values.<br>For *a_vecm_updm* = 1, the function does not update the reference values when the interrupt is triggered. Instead the function continues to use the reference values that were loaded when the function was enabled. If both *a_vecm_initm* and *a_vecm_updm* are set to logic '1', the host software can manually update the reference values in real time by writing to the A_VECM_INITX,Y,Z registers. |
| a_vecm_en | The accelerometer vector-magnitude function is enabled by setting *a_vecm_en* = 1, and disabled by clearing this bit (default). The reference values are loaded with either the current X/Y/Z acceleration values or the values stored in the A_VECM_INIT_X/Y/Z registers, depending on the state of the *a_vecm_initm* bit.<br>**Note:** The vector-magnitude function will only perform correctly up to a maximum ODR of 400 Hz. |

### 10.9.2 A_VECM_THS_MSB (0x60) register

**Table 92. A_VECM_THS_MSB register**

| | | | |
|---|---|---|---|
| a_vecm_dbcntm | — | — | a_vecm_ths[12:8] |
| 0 | 0 | 0 | 5'b00000 |

**Table 93. A_VECM_THS_MSB bit descriptions**

| Field | Description |
|---|---|
| a_vecm_dbcntm | Control bit *a_vecm_dbcntm* defines how the debounce timer is reset when the condition for triggering the interrupt is no longer true.<br>When *a_vecm_dbcntm* = 0 the debounce counter is decremented by 1 when the vector-magnitude result is below the programmed threshold value.<br>When *a_vecm_dbcntm* = 1 the debounce counter is cleared when the vector-magnitude result is below the programmed threshold value. |
| a_vecm_ths[12:8] | Five MSBs of the 13-bit unsigned A_VECM_THS value. The resolution is equal to the selected accelerometer resolution set in XYZ_DATA_CFG[*fs*] |

### 10.9.3 A_VECM_THS_LSB (0x61) register

**Table 94. A_VECM_THS_LSB register**

| |
|---|
| a_vecm_ths[7:0] |
| 8'b00000000 |

### 10.9.4 A_VECM_CNT (0x62) register

**Table 95. A_VECM_CNT register**

| |
|---|
| a_vecm_cnt[7:0] |
| 8'b00000000 |

**Table 96. A_VECM_CNT bit description**

| Field | Description |
|---|---|
| a_vecm_cnt[7:0] | Vector-magnitude function debounce count value. |

The debounce timer period is determined by the ODR selected in CTRL_REG1; it is equal to the number indicated in A_VECM_CNT register times 1/ODR. For example, a value of 16 in A_VECM_CNT with an ODR setting of 400 Hz will result in a debounce period of 40 ms.

### 10.9.5 A_VECM_INITX_MSB (0x63) register

**Table 97. A_VECM_INITX_MSB register**

| | | |
|---|---|---|
| — | — | a_vecm_initx[13:8] |
| 0 | 0 | 6'b000000 |

**Table 98. A_VECM_INITX_MSB bit description**

| Field | Description |
|---|---|
| a_vecm_initx[13:8] | Most significant 6 bits of the signed 14-bit initial X-axis value to be used as ref_x when A_VECM_CFG[*a_vecm_initm*] = 1. The resolution is determined by the settings made in XYZ_DATA_CFG[*fs*], and is equal to the accelerometer resolution. |

### 10.9.6 A_VECM_INITX_LSB (0x64) register

**Table 99. A_VECM_INITX_LSB register**

| |
|---|
| a_vecm_initx[7:0] |
| 8'b00000000 |

**Table 100. A_VECM_INITX_LSB bit description**

| Field | Description |
|---|---|
| a_vecm_initx[7:0] | LSB of the signed 14-bit initial X-axis value to be used as ref_x when A_VECM_CFG[*a_vecm_initm*] = 1. The resolution is determined by the settings made in XYZ_DATA_CFG[*fs*], and is equal to the accelerometer resolution. |

### 10.9.7 A_VECM_INITY_MSB (0x65) register

**Table 101. A_VECM_INITY_MSB register**

| | | |
|---|---|---|
| — | — | a_vecm_inity[13:8] |
| 0 | 0 | 6'b000000 |

**Table 102. A_VECM_INITY_MSB bit description**

| Field | Description |
|---|---|
| a_vecm_inity[13:8] | Most significant 6 bits of the signed 14-bit initial Y-axis value to be used as ref_y when A_VECM_CFG[*a_vecm_initm*] = 1. The resolution is determined by the settings made in XYZ_DATA_CFG[*fs*], and is equal to the accelerometer resolution. |

### 10.9.8 A_VECM_INITY_LSB (0x66) register

**Table 103. A_VECM_INITY_LSB register**

| |
|---|
| a_vecm_inity[7:0] |

**Table 104. A_VECM_INITY_LSB bit description**

| Field | Description |
|---|---|
| a_vecm_inity[7:0] | LSB of the signed 14-bit initial Y-axis value to be used as ref_y when A_VECM_CFG[*a_vecm_initm*] = 1. The resolution is determined by the settings made in XYZ_DATA_CFG[*fs*], and is equal to the accelerometer resolution. |

### 10.9.9 A_VECM_INITZ_MSB (0x67) register

**Table 105. A_VECM_INITZ_MSB register**

| | | |
|---|---|---|
| — | — | a_vecm_initz[13:8] |
| 0 | 0 | 6'b000000 |

**Table 106. A_VECM_INITZ_MSB bit description**

| Field | Description |
|---|---|
| a_vecm_initz[13:8] | Most significant 6 bits of the signed 14-bit initial Z-axis value to be used as ref_z when A_VECM_CFG[*a_vecm_initm*] = 1. The resolution is determined by the settings made in XYZ_DATA_CFG[*fs*], and is equal to the accelerometer resolution. |

### 10.9.10 A_VECM_INITZ_LSB (0x68) register

**Table 107. A_VECM_INITZ_LSB register**

| a_vecm_initz[7:0] |
|---|
| 8'b00000000 |

**Table 108. A_VECM_INITZ_LSB bit description**

| Field | Description |
|---|---|
| a_vecm_initz[7:0] | LSB of the signed 14-bit initial Z-axis value to be used as ref_z when A_VECM_CFG[*a_vecm_initm*] = 1. The resolution is determined by the settings made in XYZ_DATA_CFG[*fs*], and is equal to the accelerometer resolution. |

## 10.10 Transient (AC) acceleration detection

The transient detection function is similar to the freefall/motion detection function with the exception that a high-pass filter can be used to eliminate the DC offset from the acceleration data. There is an option to disable the high-pass filter, which causes the transient detection function to work in a similar manner to the motion detection function.

The transient detection function can be configured to signal an interrupt when the high-pass filtered acceleration delta values for any of the enabled axes exceeds the threshold programmed in TRANSIENT_THS for the debounce time programmed in TRANSIENT_COUNT. For more information on how to use and configure the transient detection function please refer to Freescale application note AN4461.

### 10.10.1 TRANSIENT_CFG (0x1D) register

**Table 109. TRANSIENT_ CFG register**

| — | — | — | tran_ele | tran_zefe | tran_yefe | tran_xefe | tran_hpf_byp |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 110. TRANSIENT_ CFG bit descriptions**

| Field | Description |
|---|---|
| tran_ele | Transient event flag latch enable. Default value: 0<br>0: Event flag latch disabled: the transient interrupt flag reflects the real-time status of the function.<br>1: Event flag latch enabled: the transient interrupt event flag is latched and a read of the TRANSIENT_SRC register is required to clear the event flag. |
| tran_zefe | Z-axis transient event flag enable. Default value: 0<br>0: Z-axis event detection disabled<br>1: Z-axis event detection enabled. Raise event flag on Z-axis acceleration value greater than threshold. |
| tran_yefe | Y-axis transient event flag enable. Default value: 0<br>0: Y-axis event detection disabled<br>1: Y-axis event detection enabled. Raise event flag on Y-axis acceleration value greater than threshold. |
| tran_xefe | X-axis transient event flag enable. Default value: 0<br>0: X-axis event detection disabled<br>1: X-axis event detection enabled. Raise event flag on X-axis acceleration value greater than threshold. |
| tran_hpf_byp | Transient function high-pass filter bypass. Default value: 0<br>0: High-pass filter is applied to accelerometer data input to the transient function.<br>1: High-pass filter is not applied to accelerometer data input to the transient function. |

### 10.10.2 TRANSIENT_SRC (0x1E) register

Transient event flag source register. This register provides the event status of the enabled axes and polarity (directional) information.

**Table 111. TRANSIENT_ CFG register**

| — | tran_ea | tran_zef | tran_zpol | tran_yef | tran_ypol | tran_xef | trans_xpol |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 112. TRANSIENT_SRC bit descriptions**

| Field | Description |
|---|---|
| tran_ea | Transient event active flag. Default value: 0<br>0: No transient event active flag has been asserted.<br>1: One or more transient event active flags has been asserted. |
| tran_zef | Z-axis transient event active flag. Default value: 0<br>0: Z-axis event flag is not active.<br>1: Z-axis event flag is active; Z-axis acceleration has exceeded the programmed threshold for the debounce time specified in TRANS_COUNT. |
| tran_zpol | Z-axis event flag polarity.<br>0: Z-axis event was above positive threshold value.<br>1: Z-axis event was below negative threshold value. |
| tran_yef | Y-axis transient event active flag. Default value: 0<br>0: Y-axis event flag is not active.<br>1: Y-axis event flag is active; Y-axis acceleration has exceeded the programmed threshold for the debounce time specified in TRANS_COUNT. |
| tran_ypol | Y-axis event flag polarity.<br>0: Y-axis event was above positive threshold value.<br>1: Y-axis event was below negative threshold value. |
| tran_xef | X-axis transient event active flag. Default value: 0<br>0: X-axis event flag is not active.<br>1: X-axis event flag is active; X-axis acceleration has exceeded the programmed threshold for the debounce time specified in TRANS_COUNT. |
| tran_xpol | X-axis event flag polarity.<br>0: X-axis event was above positive threshold value.<br>1: X-axis event was below negative threshold value. |

When TRANSIENT_CFG[*tran_ele*] = 1, the TRANSIENT_SRC event flag(s) and polarity bits are latched when the interrupt event is triggered, allowing the host application to determine which event flag(s) originally triggered the interrupt. When TRANSIENT_CFG[*tran_ele*] = 0, events which occur after the event that originally triggered the interrupt will update the flag and polarity bits, but once set, the flags can only be cleared by reading the TRANSIENT_SRC register.

### 10.10.3 TRANSIENT_THS (0x1F) register

The TRANSIENT_THS register determines the debounce counter behavior and also sets the transient event detection threshold.It is possible to use A_FFMT_THS_X/Y/Z MSB and LSB registers to set transient acceleration thresholds for individual axes using the *a_ffmt_trans_ths_en* bit in A_FFMT_THS_Y_MSB register.

**Table 113. TRANSIENT_THS register**

| tr_dbcntm | tr_ths[6:0] |
|---|---|
| 0 | 7'b0000000 |

**Table 114. TRANSIENT_THS bit descriptions**

| Field | Description |
|---|---|
| tr_dbcntm | Debounce counter mode selection.<br>0: Decrements debounce counter when the transient event condition is not true during the current ODR period.<br>1: Clears debounce counter when the transient event condition is not true during the current ODR period. |
| tr_ths[6:0] | Transient event threshold. This register has a resolution of 63 mg/LSB regardless of the full-scale range setting made in XYZ_DATA_CFG[*fs*]. If CTRL_REG1[*lnoise*] = 1, the maximum acceleration measurement range is ±4 *g*. |

The tr_ths[6:0] value is a 7-bit unsigned number, with a fixed resolution of 63 mg/LSB corresponding to a ±8 *g* measurement range. The resolution does not change with the full-scale range setting made in XYZ_DATA_CFG[*fs*]. If CTRL_REG1[*lnoise*] = 1, the measurement range is fixed at ±4 *g*, regardless of the settings made in XYZ_DATA_CFG.

### 10.10.4 TRANSIENT_COUNT (0x20) register

The TRANSIENT_COUNT register sets the minimum number of debounce counts needed to trigger the transient event interrupt flag when the measured acceleration value exceeds the threshold set in TRANSIENT_THS for any of the enabled axes.

**Table 115. TRANSIENT_COUNT register**

| tr_count[7:0] |
|---|
| 8'b00000000 |

**Table 116. TRANSIENT_COUNT bit description**

| Field | Description |
|---|---|
| tr_count[7:0) | Transient function debounce count value. |

The time step for the transient detection debounce counter is set by the value of the system ODR and power mode as shown in Table 117.

**Table 117. TRANSIENT_COUNT relationship with the ODR**

| ODR (Hz) | Max time range (s) | | | | Time step (ms) | | | |
|---|---|---|---|---|---|---|---|---|
| | Normal | LPLN | High resolution | Low power | Normal | LPLN | High resolution | Low power |
| 800 | 0.319 | 0.319 | 0.319 | 0.319 | 1.25 | 1.25 | 1.25 | 1.25 |
| 400 | 0.638 | 0.638 | 0.638 | 0.638 | 2.5 | 2.5 | 2.5 | 2.5 |
| 200 | 1.28 | 1.28 | 0.638 | 1.28 | 5 | 5 | 2.5 | 5 |
| 100 | 2.55 | 2.55 | 0.638 | 2.55 | 10 | 10 | 2.5 | 10 |
| 50 | 5.1 | 5.1 | 0.638 | 5.1 | 20 | 20 | 2.5 | 20 |
| 12.5 | 5.1 | 20.4 | 0.638 | 20.4 | 20 | 80 | 2.5 | 80 |
| 6.25 | 5.1 | 20.4 | 0.638 | 40.8 | 20 | 80 | 2.5 | 160 |
| 1.56 | 5.1 | 20.4 | 0.638 | 40.8 | 20 | 80 | 2.5 | 160 |

An ODR of 100 Hz and a TRANSIENT_COUNT value of 15, when accelerometer OSR is set to normal using CTRL_REG2, would result in minimum debounce response time of 150 ms.

## 10.11 Pulse detection

### 10.11.1 PULSE_CFG (0x21) register

This register configures the pulse event detection function.

**Table 118. PULSE_CFG register**

| pls_dpa | pls_ele | pls_zdpefe | pls_zspefe | pls_ydpefe | pls_yspefe | pls_xdpefe | pls_xspefe |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 119. PULSE_CFG bit descriptions**

| Field | Description |
|---|---|
| pls_dpa | Double-pulse abort.<br>0: Double-pulse detection is not aborted if the start of a pulse is detected during the time period specified by the PULSE_LTCY register.<br>1: Setting the *pls_dpa* bit momentarily suspends the double-tap detection if the start of a pulse is detected during the time period specified by the PULSE_LTCY register and the pulse ends before the end of the time period specified by the PULSE_LTCY register. |
| pls_ele | Pulse event flag latch enable. When enabled, a read of the PULSE_SRC register is needed to clear the event flag.<br>0: Event flag latch disabled<br>1: Event flag latch enabled |
| pls_zdpefe | Event flag enable on double-pulse event on Z-axis.<br>0: Event detection disabled<br>1: Raise event flag on detection of double-pulse event on Z-axis |
| pls_zspefe | Event flag enable on single-pulse event on Z-axis.<br>0: Event detection disabled<br>1: Raise event flag on detection of single-pulse event on Z-axis |
| pls_ydpefe | Event flag enable on double-pulse event on Y-axis.<br>0: Event detection disabled<br>1: Raise event flag on detection of double-pulse event on Y-axis |
| pls_yspefe | Event flag enable on single-pulse event on Y-axis.<br>0: Event detection disabled<br>1: Raise event flag on detection of single-pulse event on Z-axis. |
| pls_xdpefe | Event flag enable on double-pulse event on X-axis.<br>0: Event detection disabled<br>1: Raise event flag on detection of double-pulse event on X-axis. |
| pls_xspefe | Event flag enable on single-pulse event on X-axis.<br>0: Event detection disabled<br>1: Raise event flag on detection of single-pulse event on X-axis. |

### 10.11.2 PULSE_SRC (0x22) register

This register indicates the status bits for the pulse detection function.

**Table 120. PULSE_SRC register**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| pls_src_ea | pls_src_axz | pls_src_axy | pls_src_axx | pls_src_dpe | pls_src_polz | pls_src_poly | pls_src_polx |

**Table 121. PULSE_SRC bit descriptions**

| Field | Description |
|---|---|
| pls_src_ea | Event active flag.<br>0: No interrupt has been generated<br>1: One or more interrupt events have been generated |
| pls_src_axz | Z-axis event flag.<br>0: No interrupt.<br>1: Z-axis event has occurred |
| pls_src_axy | Y-axis event flag.<br>0: No interrupt.<br>1: Y-axis event has occurred |
| pls_src_axx | X-axis event flag.<br>0: No interrupt.<br>1: X-axis event has occurred. |
| pls_src_dpe | Double pulse on first event.<br>0: Single-pulse event triggered interrupt.<br>1: Double-pulse event triggered interrupt. |
| pls_src_polz | Pulse polarity of Z-axis event.<br>0: Pulse event that triggered interrupt was positive.<br>1: Pulse event that triggered interrupt was negative. |
| pls_src_poly | Pulse polarity of Y-axis event.<br>0: Pulse event that triggered interrupt was positive.<br>1: Pulse event that triggered interrupt was negative. |
| pls_src_polx | Pulse polarity of X-axis event.<br>0: Pulse event that triggered interrupt was positive.<br>1: Pulse event that triggered interrupt was negative. |

### 10.11.3 PULSE_THSX (0x23) register

**Table 122. PULSE_THSX register**

| | |
|---|---|
| — | pls_thsx[6:0] |
| 0 | 7'b0000000 |

**Table 123. PULSE_THSX bit description**

| Field | Description |
|---|---|
| pls_thsx[6:0] | Pulse threshold for X-axis. |

The PULSE_THSX, PULSE_THSY and PULSE_THSZ registers define the thresholds used by the system to start the pulse-event detection procedure. Threshold values for each axis are unsigned 7-bit numbers with a fixed resolution of 0.063 *g*/LSB, corresponding to an 8 *g* acceleration full-scale range. The full-scale range is fixed at 8 *g* for the pulse detection function, regardless of the settings made in XYZ_DATA_CFG[*fs*].

### 10.11.4 PULSE_THSY (0x24) register

**Table 124. PULSE_THSY register**

| — | pls_thsy[6:0] |
|---|---|
| 0 | 7'b0000000 |

**Table 125. PULSE_THSY bit description**

| Field | Description |
|---|---|
| pls_thsy[6:0] | Pulse threshold for Y-axis. |

### 10.11.5 PULSE_THSZ (0x25) register

**Table 126. PULSE_THSZ register**

| — | pls_thsz[6:0] |
|---|---|
| 0 | 7'b0000000 |

**Table 127. PULSE_THSZ bit description**

| Field | Description |
|---|---|
| pls_thsz[6:0] | Pulse threshold for Z-axis. |

### 10.11.6 PULSE_TMLT (0x26) register

**Table 128. PULSE_TMLT register**

| pls_tmlt[7:0] |
|---|
| 8'b00000000 |

**Table 129. PULSE_TMLT bit description**

| Field | Description |
|---|---|
| pls_tmlt[7:0] | *pls_tmlt*[7:0] defines the maximum time interval that can elapse between the start of the acceleration on the selected channel exceeding the specified threshold and the end when the channel acceleration goes back below the specified threshold. |

Minimum time step for the pulse-time limit is defined in Tables 130 and 131. Maximum time for a given ODR is "Minimum time step x 255".

**Table 130. Time step for PULSE_TMLT with HP_FILTER_CUTOFF[*pls_hpf_en*] = 1**

| ODR (Hz) | Max time range (s) | | | | Time step (ms) | | | |
|---|---|---|---|---|---|---|---|---|
| | Normal | LPLN | High resolution | Low power | Normal | LPLN | High resolution | Low power |
| 800 | 0.319 | 0.319 | 0.319 | 0.319 | 1.25 | 1.25 | 1.25 | 1.25 |
| 400 | 0.638 | 0.638 | 0.638 | 0.638 | 2.5 | 2.5 | 2.5 | 2.5 |
| 200 | 1.28 | 1.28 | 0.638 | 1.28 | 5 | 5 | 2.5 | 5 |
| 100 | 2.55 | 2.55 | 0.638 | 2.55 | 10 | 10 | 2.5 | 10 |
| 50 | 5.1 | 5.1 | 0.638 | 5.1 | 20 | 20 | 2.5 | 20 |
| 12.5 | 5.1 | 20.4 | 0.638 | 20.4 | 20 | 80 | 2.5 | 80 |
| 6.25 | 5.1 | 20.4 | 0.638 | 40.8 | 20 | 80 | 2.5 | 160 |
| 1.56 | 5.1 | 20.4 | 0.638 | 40.8 | 20 | 80 | 2.5 | 160 |

**Table 131. Time step for PULSE_TMLT with HP_FILTER_CUTOFF[*pls_hpf_en*] = 0**

| ODR (Hz) | Max time range (s) | | | | Time step (ms) | | | |
|---|---|---|---|---|---|---|---|---|
| | Normal | LPLN | High resolution | Low power | Normal | LPLN | High resolution | Low power |
| 800 | 0.159 | 0.159 | 0.159 | 0.159 | 0.625 | 0.625 | 0.625 | 0.625 |
| 400 | 0.159 | 0.159 | 0.159 | 0.319 | 0.625 | 0.625 | 0.625 | 1.25 |
| 200 | 0.319 | 0.319 | 0.159 | 0.638 | 1.25 | 1.25 | 0.625 | 2.5 |
| 100 | 0.638 | 0.638 | 0.159 | 1.28 | 2.5 | 2.5 | 0.625 | 5 |
| 50 | 1.28 | 1.28 | 0.159 | 2.55 | 5 | 5 | 0.625 | 10 |
| 12.5 | 1.28 | 5.1 | 0.159 | 10.2 | 5 | 20 | 0.625 | 40 |
| 6.25 | 1.28 | 5.1 | 0.159 | 10.2 | 5 | 20 | 0.625 | 40 |
| 1.56 | 1.28 | 5.1 | 0.159 | 10.2 | 5 | 20 | 0.625 | 40 |

Therefore an ODR setting of 400 Hz, when accelerometer OSR is set to normal using CTRL_REG2, would result in a maximum pulse-time limit of (0.625 ms * 255) = 159 ms.

### 10.11.7 PULSE_LTCY (0x27) register

**Table 132. PULSE_LTCY register**

| pls_ltcy[7:0] |
|---|
| 8'b00000000 |

**Table 133. PULSE_LTCY bit description**

| Field | Description |
|---|---|
| pls_ltcy[7:0] | *pls_ltcy*[7:0] defines the time interval that starts after the first pulse detection where the pulse-detection function ignores the start of a new pulse. |

Minimum time step for the pulse latency is defined in Tables 134 and 135. Maximum time is "(time step @ ODR and power mode) x 255".

**Table 134. Time step for PULSE_LTCY with HP_FILTER_CUTOFF[*pls_hpf_en*] = 1**

| ODR (Hz) | Max time range (s) | | | | Time step (ms) | | | |
|---|---|---|---|---|---|---|---|---|
| | Normal | LPLN | High resolution | Low power | Normal | LPLN | High resolution | Low power |
| 800 | 0.638 | 0.638 | 0.638 | 0.638 | 2.5 | 2.5 | 2.5 | 2.5 |
| 400 | 1.276 | 1.276 | 1.276 | 1.276 | 5 | 5 | 5 | 5 |
| 200 | 2.56 | 2.56 | 1.276 | 2.56 | 10 | 10 | 5 | 10 |
| 100 | 5.1 | 5.1 | 1.276 | 5.1 | 20 | 20 | 5 | 20 |
| 50 | 10.2 | 10.2 | 1.276 | 10.2 | 40 | 40 | 5 | 40 |
| 12.5 | 10.2 | 40.8 | 1.276 | 40.8 | 40 | 160 | 5 | 160 |
| 6.25 | 10.2 | 40.8 | 1.276 | 81.6 | 40 | 160 | 5 | 320 |
| 1.56 | 10.2 | 40.8 | 1.276 | 81.6 | 40 | 160 | 5 | 320 |

**Table 135. Time step for PULSE_LTCY with HP_FILTER_CUTOFF[*pls_hpf_en*] = 0**

| ODR (Hz) | Max time range (s) | | | | Time step (ms) | | | |
|---|---|---|---|---|---|---|---|---|
| | Normal | LPLN | High resolution | Low power | Normal | LPLN | High resolution | Low power |
| 800 | 0.318 | 0.318 | 0.318 | 0.318 | 1.25 | 1.25 | 1.25 | 1.25 |
| 400 | 0.318 | 0.318 | 0.318 | 0.638 | 1.25 | 1.25 | 1.25 | 2.5 |
| 200 | 0.638 | 0.638 | 0.318 | 1.276 | 2.5 | 2.5 | 1.25 | 5 |
| 100 | 1.276 | 1.276 | 0.318 | 2.56 | 5 | 5 | 1.25 | 10 |
| 50 | 2.56 | 2.56 | 0.318 | 5.1 | 10 | 10 | 1.25 | 20 |
| 12.5 | 2.56 | 10.2 | 0.318 | 20.4 | 10 | 40 | 1.25 | 80 |
| 6.25 | 2.56 | 10.2 | 0.318 | 20.4 | 10 | 40 | 1.25 | 80 |
| 1.56 | 2.56 | 10.2 | 0.318 | 20.4 | 10 | 40 | 1.25 | 80 |

### 10.11.8 PULSE_WIND (0x28) register

**Table 136. PULSE_WIND register**

| pls_wind[7:0] |
|---|
| 8'b00000000 |

**Table 137. PULSE_WIND bit description**

| Field | Description |
|---|---|
| pls_wind[7:0] | *pls_wind*[7:0] defines the maximum interval of time that can elapse after the end of the latency interval in which the start of the second pulse event must be detected provided the device has been configured for double pulse detection. The detected second pulse width must be shorter than the time limit constraint specified by the PULSE_TMLT register, but the end of the double pulse need not finish within the time specified by the PULSE_WIND register. |

The time step for the pulse-window counter varies with the selected ODR and power modes as defined in Tables 138 and 139. The maximum time value is equal to (time step @ ODR and power mode) x 255.

**Table 138. Time step for PULSE_WIND with HP_FILTER_CUTOFF[*pls_hpf_en*] = 1**

| ODR (Hz) | Max time range (s) | | | | Time step (ms) | | | |
|---|---|---|---|---|---|---|---|---|
| | Normal | LPLN | High resolution | Low power | Normal | LPLN | High resolution | Low power |
| 800 | 0.638 | 0.638 | 0.638 | 0.638 | 2.5 | 2.5 | 2.5 | 2.5 |
| 400 | 1.276 | 1.276 | 1.276 | 1.276 | 5 | 5 | 5 | 5 |
| 200 | 2.56 | 2.56 | 1.276 | 2.56 | 10 | 10 | 5 | 10 |
| 100 | 5.1 | 5.1 | 1.276 | 5.1 | 20 | 20 | 5 | 20 |
| 50 | 10.2 | 10.2 | 1.276 | 10.2 | 40 | 40 | 5 | 40 |
| 12.5 | 10.2 | 40.8 | 1.276 | 40.8 | 40 | 160 | 5 | 160 |
| 6.25 | 10.2 | 40.8 | 1.276 | 81.6 | 40 | 160 | 5 | 320 |
| 1.56 | 10.2 | 40.8 | 1.276 | 81.6 | 40 | 160 | 5 | 320 |

**Table 139. Time step for PULSE_WIND with HP_FILTER_CUTOFF[*pls_hpf_en*] = 0**

| ODR (Hz) | Max time range (s) | | | | Time step (ms) | | | |
|---|---|---|---|---|---|---|---|---|
| | Normal | LPLN | High resolution | Low power | Normal | LPLN | High resolution | Low power |
| 800 | 0.318 | 0.318 | 0.318 | 0.318 | 1.25 | 1.25 | 1.25 | 1.25 |
| 400 | 0.318 | 0.318 | 0.318 | 0.638 | 1.25 | 1.25 | 1.25 | 2.5 |
| 200 | 0.638 | 0.638 | 0.318 | 1.276 | 2.5 | 2.5 | 1.25 | 5 |
| 100 | 1.276 | 1.276 | 0.318 | 2.56 | 5 | 5 | 1.25 | 10 |
| 50 | 2.56 | 2.56 | 0.318 | 5.1 | 10 | 10 | 1.25 | 20 |
| 12.5 | 2.56 | 10.2 | 0.318 | 20.4 | 10 | 40 | 1.25 | 80 |
| 6.25 | 2.56 | 10.2 | 0.318 | 20.4 | 10 | 40 | 1.25 | 80 |
| 1.56 | 2.56 | 10.2 | 0.318 | 20.4 | 10 | 40 | 1.25 | 80 |

## 10.12 Offset correction

The 8-bit 2's complement offset correction registers are used to remove the sensor zero *g* offset on the X, Y, and Z axes after device board mount. The resolution of the offset registers is 2 mg per LSB, with an effective offset adjustment range of -256 mg to +254 mg for each axis.

For more information on how to calibrate the 0 *g* offset, please refer to Freescale application note AN4069.

### 10.12.1 OFF_X (0x2F) register

**Table 140. OFF_X register**

| off_x[7:0] |
|---|
| 8'b00000000 |

**Table 141. OFF_X bit description**

| Field | Description |
|---|---|
| off_x[7:0] | X-axis offset correction value expressed as an 8-bit 2's complement number. |

### 10.12.2 OFF_Y (0x30) register

**Table 142. OFF_Y register**

| off_y[7:0] |
|---|
| 8'b00000000 |

**Table 143. OFF_Y bit description**

| Field | Description |
|---|---|
| off_y[7:0] | Y-axis offset correction value expressed as an 8-bit 2's complement number. |

### 10.12.3 OFF_Z (0x31) register

**Table 144. OFF_Z register**

| off_z[7:0] |
|---|
| 8'b00000000 |

**Table 145. OFF_Z bit description**

| Field | Description |
|---|---|
| off_z[7:0] | Z-axis offset correction value expressed as an 8-bit 2's complement number. |

# 11 Mounting Guidelines for the Quad Flat No-Lead (QFN) Package

Printed Circuit Board (PCB) layout is a critical portion of the total design. The footprint for the surface mount packages must be the correct size to ensure proper solder connection interface between the PCB and the package. With the correct footprint, the packages will self-align when subjected to a solder reflow process.

These guidelines are for soldering and mounting the Quad Flat No-Lead (QFN) package inertial sensors to PCBs. The purpose is to minimize the stress on the package after board mounting. The FXLS8471Q uses the QFN package platform. This section describes suggested methods of soldering these devices to the PCB for consumer applications.

Freescale application note AN1902, “Quad Flat Pack No-Lead (QFN) Micro Dual Flat Pack No-Lead (DFN)” discusses the QFN package used by the FXLS8471Q, PCB design guidelines for using QFN packages and temperature profiles for reflow soldering.

## 11.1 Overview of soldering considerations

Information provided here is based on experiments executed on QFN devices. As they cannot represent exact conditions present at a customer site, the information provided herein should be used for guidance only and further process and design optimizations are recommended to develop an application specific solution. It should be noted that with the proper PCB footprint and solder stencil designs, the package will self-align during the solder reflow process.

## 11.2 Halogen content

This package is designed to be Halogen Free, exceeding most industry and customer standards. Halogen Free means that no homogeneous material within the assembled package will contain chlorine (Cl) in excess of 700 ppm or 0.07% weight/weight or bromine (Br) in excess of 900 ppm or 0.09% weight/weight.

## 11.3 PCB mounting recommendations

1. The PCB land should be designed with Non-Solder Mask Defined (NSMD) as shown in Figure 18 and Figure 19.
2. No additional via pattern underneath package.
3. PCB land pad is 0.8 mm by 0.3 mm as shown in Figure 18 and Figure 19.
4. Solder mask opening = PCB land pad edge + 0.113 mm larger all around.
5. Stencil opening = PCB land pad -0.015 mm smaller all around = 0.77 mm by 0.27 mm.
6. Stencil thickness is 100 or 125 µm.
7. Do not place any components or vias at a distance less than 2 mm from the package land area. This may cause additional package stress if it is too close to the package land area.
8. Signal traces connected to pads should be as symmetric as possible. Put dummy traces on the NC pads in order to have same length of exposed trace for all pads.
9. Use a standard pick and place process and equipment. Do not use a hand soldering process.
10. Do not use a screw down or stacking to fix the PCB into an enclosure as this could bend the PCB, putting stress on the package.
11. The PCB should be rated for the multiple lead-free reflow condition with max 260°C temperature.
12. No copper traces on top layer of PCB under the package. This will cause planarity issues with board mount. Freescale QFN sensors are compliant with Restrictions on Hazardous Substances (RoHS), having halide-free molding compound (green) and lead-free terminations. These terminations are compatible with tin-lead (Sn-Pb) as well as tin-silver-copper (Sn-Ag-Cu) solder paste soldering processes. Reflow profiles applicable to those processes can be used successfully for soldering the devices.

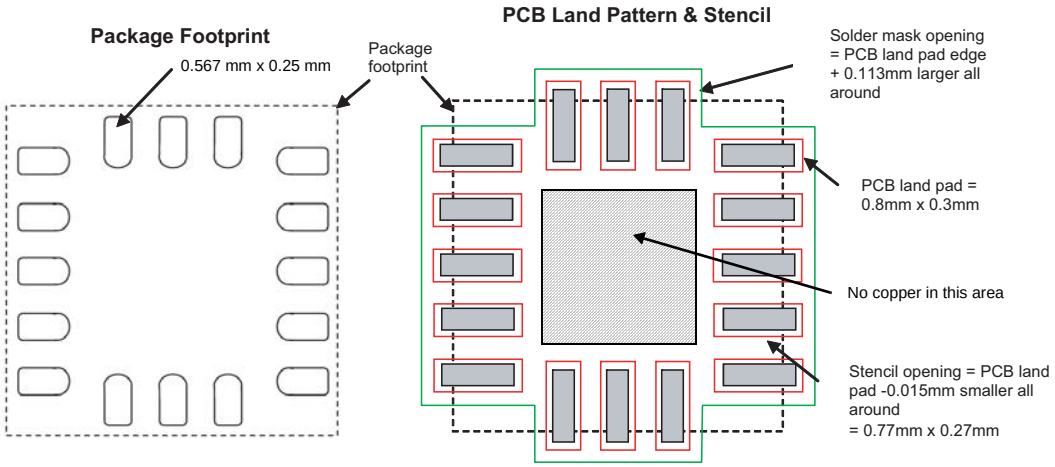



**Figure 18. Recommended PCB land pattern, solder mask, and stencil opening near package footprint**

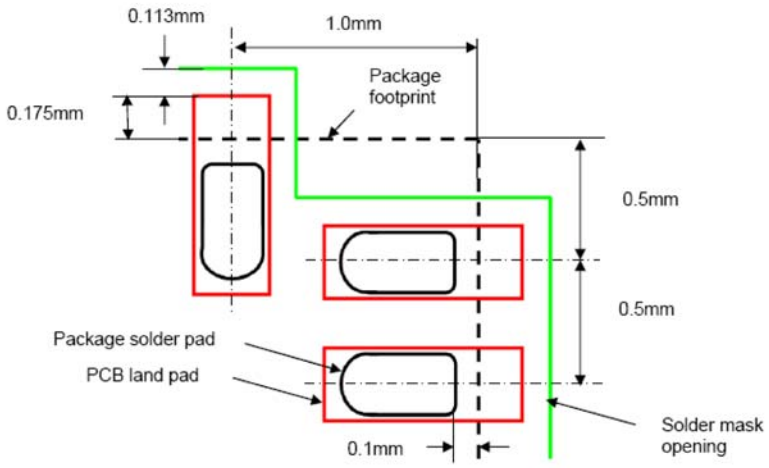



**Figure 19. Detailed dimensions**

# 12 Package Thermal Characteristics

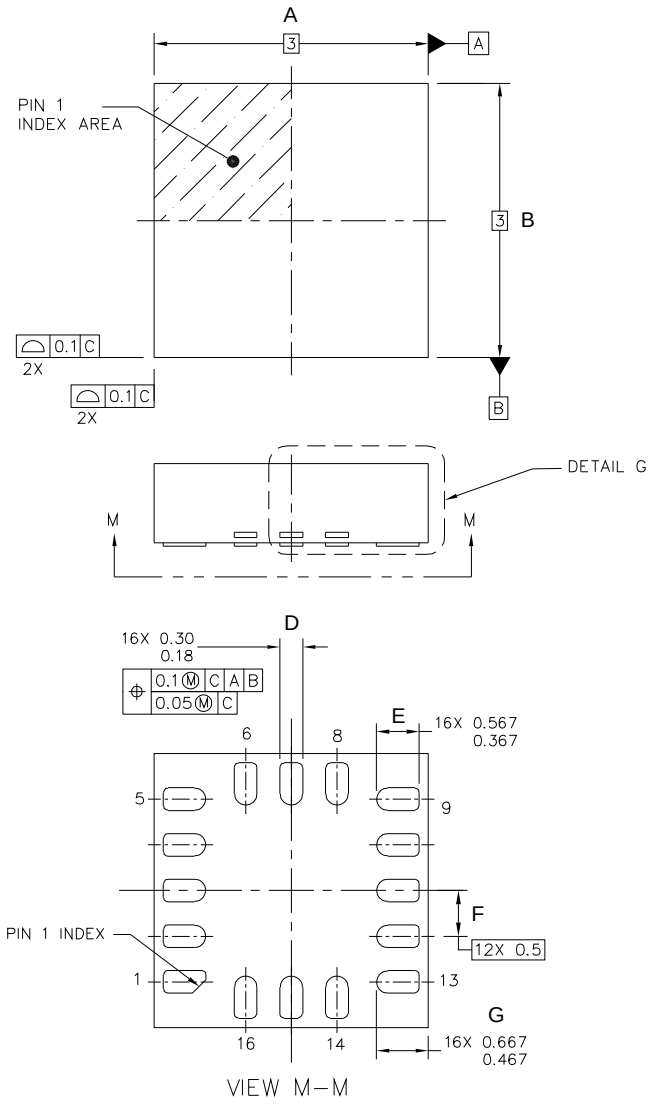
**Table 146. Thermal resistance data**

| Rating | Description | Symbol | Value | Unit |
|---|---|---|---|---|
| Junction-to-ambient, natural convection[1][2] | Single-layer board | $R_{\theta JA}$ | 163 | °C/W |
| Junction-to-ambient, natural convection[1][3] | Four-layer board (two signals, two planes) | | 70 | |
| Junction-to-board[4] | | $R_{\theta JB}$ | 33 | °C/W |
| Junction-to-case (top)[5] | | $R_{\theta JCTop}$ | 84 | °C/W |
| Junction-to-package (top)[6] | Natural convection | $\Psi_{JT}$ | 6 | °C/W |

1. Junction temperature is a function of die size, on-chip power dissipation, package thermal resistance, mounting site (board) temperature, ambient temperature, air flow, power dissipation of other components on the board, and board thermal resistance.
2. Per JEDEC JESD51-2 with the single-layer board (JESD51-3) horizontal.
3. Per JEDEC JESD51-6 with the board (JESD51-7) horizontal.
4. Thermal resistance between the die and the printed circuit board per JEDEC JESD51-8. Board temperature is measured on the top surface of the board near the package.
5. Thermal resistance between the die and the case top surface as measured by the cold plate method (MIL SPEC-883 Method 1012.1).
6. Thermal characterization parameter indicating the temperature difference between package top and the junction temperature per JEDEC JESD51-2. When Greek letters are not available, the thermal characterization parameter is written as Psi-JT.

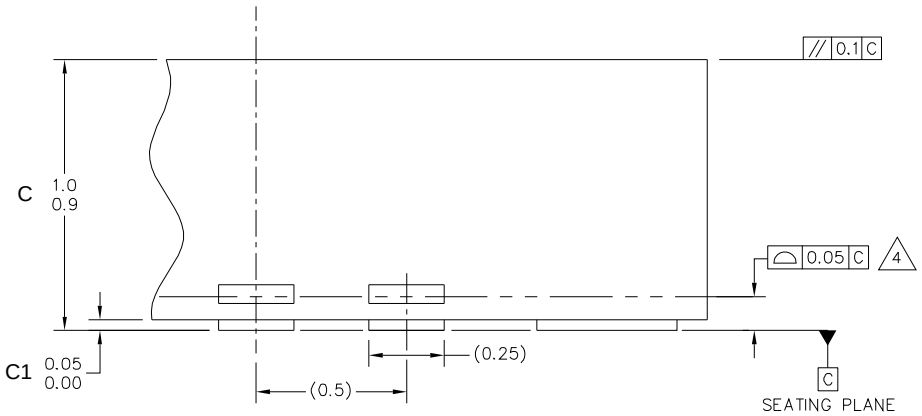
# 13 Package

This drawing is located at http://cache.freescale.com/files/shared/doc/package_info/98ASA00063D.pdf.




| © FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED. | MECHANICAL OUTLINE | PRINT VERSION NOT TO SCALE |
|---|---|---|
| TITLE: QUAD FLAT NO LEAD COL PACKAGE (QFN–COL) 16 TERMINAL, 0.5 PITCH (3 X 3 X 1.0) | DOCUMENT NO: 98ASA00063D | REV: A |
| | CASE NUMBER: 2077–02 | 20 OCT 2011 |
| | STANDARD: NON JEDEC | |

**CASE 2077-02**
**ISSUE A**
**16-LEAD QFN**




DETAIL G

| © FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED. | MECHANICAL OUTLINE | PRINT VERSION NOT TO SCALE |
|---|---|---|
| TITLE: QUAD FLAT NO LEAD COL PACKAGE (QFN-COL) 16 TERMINAL, 0.5 PITCH (3 X 3 X 1.0) | DOCUMENT NO: 98ASA00063D | REV: A |
| | CASE NUMBER: 2077-02 | 20 OCT 2011 |
| | STANDARD: NON JEDEC | |

**CASE 2077-02**
**ISSUE A**
**16-LEAD QFN**

**Table 147. Package dimensions (mm)**

| Symbol | Description | Min | Typ | Max |
|---|---|---|---|---|
| A | Package width | 2.9 | 3 | 3.1 |
| B | Package length | 2.9 | 3 | 3.1 |
| C | Package thickness | 0.9 | 0.98 | 1 |
| C1 | Lead finger (pad) seating height | 0 | — | 0.05 |
| D | Lead finger (pad) width | 0.18 | 0.25 | 0.30 |
| E | Lead length | 0.367 | 0.467 | 0.567 |
| F | Lead finger-finger (pad-pad) pitch | — | 0.5 | — |
| G | Package edge to inner edge of lead | 0.467 | 0.567 | 0.667 |

NOTES:

1. ALL DIMENSIONS ARE IN MILLIMETERS.
2. INTERPRET DIMENSIONS AND TOLERANCES PER ASME Y14.5M–1994.
3. THIS IS NON JEDEC REGISTERED PACKAGE.
4. COPLANARITY APPLIES TO ALL LEADS.
5. MIN. METAL GAP SHOULD BE 0.2MM.

| © FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED. | MECHANICAL OUTLINE | PRINT VERSION NOT TO SCALE |
|---|---|---|
| TITLE: QUAD FLAT NO LEAD COL PACKAGE (QFN–COL) 16 TERMINAL, 0.5 PITCH (3 X 3 X 1.0) | DOCUMENT NO: 98ASA00063D | REV: A |
| | CASE NUMBER: 2077–02 | 20 OCT 2011 |
| | STANDARD: NON JEDEC | |

**CASE 2077-02**
**ISSUE A**
**16-LEAD QFN**

# Appendix A

## A.1 Errata

### A.1.1 SPI Mode Soft-reset using CTRL_REG2 (0x2B), bit 6

***Description:***

Following a soft-reset command, issued by setting CTRL_REG2[*rst*] = 1, certain device-specific parameters do not get updated correctly from NVM, causing inaccurate data output and incorrect WHOAMI (0x0D) register content. This behavior happens only in SPI mode. In $I^2C$ mode the device works as advertised.

***Workaround:***

Avoid using soft-reset in SPI mode by alternately utilizing the hardware RESET pin.

# 14 Revision History

**Table 148. Revision history**

| Revision number | Revision date | Description of changes |
|---|---|---|
| 1.0 | 8/2013 | • Initial data sheet. |
| 1.1 | 8/2013 | • Global update: “counts/g” changed to “LSB/g” throughout document.<br>• Table 2: Updated Min values for Self-Test output change, X, Y, and Z from +249, +335, and +1680 to +192, +270, and +1275 respectively.<br>• Appendix A.1.1: Corrected register name in Description paragraph. |
| 1.2 | 11/2014 | • Added Ordering Information table on page 2. |
| 1.3 | 01/2015 | • Added callouts to the package drawing and dimension table following package drawing. |
| 1.4 | 03/2015 | • Table 147: Added Typ dimensions for Symbol C and E. |
| 1.5 | 06/2015 | • Added G callout to package drawing and dimension table following package drawing. |

***How to Reach Us:***

**Home Page:**
freescale.com

**Web Support:**
freescale.com/support

Information in this document is provided solely to enable system and software implementers to use Freescale products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document.

Freescale reserves the right to make changes without further notice to any products herein. Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer’s technical experts. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: freescale.com/salestermsandconditions.

Freescale, the Freescale logo, and the Energy Efficient Solutions logo are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. All other product or service names are the property of their respective owners.
© 2013-2015 Freescale Semiconductor, Inc.

Document Number: FXLS8471Q
Rev. 1.5
06/2015