GENERAL DESCRIPTION

The MC3419 is a small form factor, integrated digital output 3-axis accelerometer with a feature set optimized for cell phones and IoT product motion sensing. Applications include user interface control, gaming motion input, electronic compass tilt compensation for cell phones, game controllers, remote controls and portable media products.

The MC3419 features a dedicated motion block which implements embedded algorithms to support “any motion” and shake detection, tilt/flip and tilt 35 position detection.

The EV3419A is a prebuilt circuit board with MC3419 LGA-12 3-axes sensor. The MC3419 has internal sample rate from 0.5 to 1000 samples / second and measures acceleration with a wide usage range, from +/-2g up to +/-16g, and 16-bit high precision ADC output, which is easy to fit on top of the microcontroller, such as an Arduino. The accelerometer communicates via I2C/SPI and gives out motion detection or sample acquisition conditions to trigger an interrupt toward an MCU.

The sensor data is easily readable by connecting DVDD to 3.3V, GND to ground, and SCL/SDA pins to your Arduino I2C clock and data pin respectively. Download the MC3419 library from GitHub onto the board, run the example sketch, and then sensor data shortly comes out in raw data count and SI unit accelerometer measurements. An easy-to-use demonstration on EV3419A using the Arduino platform is depicted in this document.

MC3419 FEATURES

Range, Sampling & Power
- ±2,4,8,12 or 16g range
- 16-bit single sample resolution
- 16-bit resolution with FIFO
- 0.5 to 1000 Hz Output Data Rate
- 4 μA typical Standby current
- Low typical active current

Simple System Integration
- I2C interface, up to 1 MHz
- SPI Interface, up to 10 MHz
- 2×2×0.92 mm 12-pin LGA package
- Single-chip 3D silicon MEMS
- RoHS compliant
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>General Operation</td>
<td>3</td>
</tr>
<tr>
<td>1.1</td>
<td>Pinouts</td>
<td>3</td>
</tr>
<tr>
<td>1.2</td>
<td>Power Pins</td>
<td>3</td>
</tr>
<tr>
<td>1.3</td>
<td>I2C Pins</td>
<td>4</td>
</tr>
<tr>
<td>1.4</td>
<td>SPI Pins</td>
<td>5</td>
</tr>
<tr>
<td>1.5</td>
<td>Interrupt Pins</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>Assembly and Test</td>
<td>6</td>
</tr>
<tr>
<td>2.1</td>
<td>I2C Interface</td>
<td>6</td>
</tr>
<tr>
<td>2.2</td>
<td>SPI Interface</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>Demo</td>
<td>8</td>
</tr>
<tr>
<td>3.1</td>
<td>Download the Driver from GitHub</td>
<td>8</td>
</tr>
<tr>
<td>3.2</td>
<td>Load the Demo</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>Library Reference</td>
<td>12</td>
</tr>
<tr>
<td>4.1</td>
<td>Create MCUBE_MC34X9 Object</td>
<td>12</td>
</tr>
<tr>
<td>4.2</td>
<td>Initialize and Configure Sensor</td>
<td>12</td>
</tr>
<tr>
<td>4.3</td>
<td>Set Range</td>
<td>12</td>
</tr>
<tr>
<td>4.4</td>
<td>Read Range</td>
<td>12</td>
</tr>
<tr>
<td>4.5</td>
<td>Set Sampling Rate</td>
<td>12</td>
</tr>
<tr>
<td>4.6</td>
<td>Read Sampling Rate</td>
<td>13</td>
</tr>
<tr>
<td>4.7</td>
<td>Config Motion Feature</td>
<td>13</td>
</tr>
<tr>
<td>4.8</td>
<td>Config Interrupt Mode</td>
<td>13</td>
</tr>
<tr>
<td>4.9</td>
<td>Read Raw Count Data</td>
<td>14</td>
</tr>
<tr>
<td>5</td>
<td>Downloads</td>
<td>15</td>
</tr>
<tr>
<td>5.1</td>
<td>MC3419 Accelerometer Datasheet and Quick Start Guide</td>
<td>15</td>
</tr>
<tr>
<td>5.2</td>
<td>MC34X9 Driver at GitHub</td>
<td>15</td>
</tr>
<tr>
<td>5.3</td>
<td>All Other mCube Documentation</td>
<td>15</td>
</tr>
<tr>
<td>6</td>
<td>Schematics</td>
<td>16</td>
</tr>
<tr>
<td>7</td>
<td>Bill of Materials</td>
<td>17</td>
</tr>
<tr>
<td>8</td>
<td>Fabrication Print</td>
<td>18</td>
</tr>
<tr>
<td>9</td>
<td>Revision History</td>
<td>19</td>
</tr>
<tr>
<td>10</td>
<td>Legal</td>
<td>20</td>
</tr>
</tbody>
</table>
1 GENERAL OPERATION

1.1 PINOUTS

1: SDO
2: INT1
3: AVDD
4: VPP
5: INT2
10: SCS
9: DVDD
8: GND
7: SDA
6: SCL

1.2 POWER PINS

- **DVDD** – 3.3V Power Supply Input
- **GND** – Ground Pin for Power and Logic
- **R7**: The current draw from the sensor can be measured by putting an ammeter in place of R7.
- In the following demonstration, an Arduino DUE is used to illustrate on how to test the evaluation board with a microcontroller.
- Please be advised that if an Arduino UNO is used instead, hardware modification on Arduino UNO MUST be made for it to output at 3.3V. (WARNING: attempting to power the part at 5V is likely to damage it.) By default, Arduino UNO operates at 5V, which is higher than the maximum voltage rating for the evaluation board. Please refer to an excellent tutorial on modifying Arduino UNO to output at 3.3V: [https://learn.adafruit.com/arduino-tips-tricks-and-techniques/3-3v-conversion](https://learn.adafruit.com/arduino-tips-tricks-and-techniques/3-3v-conversion)
### 1.3 I2C PINS

- Connect the **SCL** (I2C clock pin) to your microcontroller’s I2C clock line.
- Connect the **SDA** (I2C data pin) to your microcontroller’s I2C data line.

**R4, R5**: If using I2C and I2C pull-up resistors are needed for your application then install ~4.7KΩ resistors into R4 (SCL clock pin) and R5 (SDA data pin) which are not installed by factory default. In addition, besides soldering resistors on R4/R5, you can add axial lead 4.7K ohm resistors to the SDA and SCL pin respectively. It will work the same either way.

**NOTE**: DO NOT install more than one setup pull-up resistors per I2C bus.
1.4 SPI PINS

With an SPI connection, there is always one master device (usually a microcontroller) which controls the peripheral devices. Typically, there are four wires commonly connected to all the devices:

Connect the SCS (Slave Chip Select) to the pin on the device that the master can use to enable and disable SPI cycles.

Connect the SCL (Serial clock) to the pin where the clock pulses synchronize data transmission generated by the master.

Connect SDO to the pin where the Slave sends data to the master (Master Input, Slave Output).

Connect SDA to the pin where the Master sends data to the peripherals (Master Output, Slave Input).

1.5 INTERRUPT PINS

INT - HW interrupt signal pin. This pin will be triggered by the device when data is ready to read, or a motion event is detected by the accelerometer. (Not currently supported in the library for the interrupt pin, so please check the datasheet for the I2C commands and related registers).

R6: If using the sensor interrupt signal as open-drain, then install pull-up resistor ~4.7KΩ into R6 (not installed by default).
2 ASSEMBLY AND TEST

Please note that the SPI and I2C interfaces cannot both be active at the same time as the clock (SCK) and data (SDA) are shared between the two protocols.

2.1 I2C INTERFACE

The EV3419A evaluation board can be easily wired to any microcontroller. This example shows a typical Arduino DUE platform. For other microcontrollers, be sure it has I2C with repeated-start support, then port the code. Please refer to the illustration below to connect the related pins.

- Connect DVDD to the power supply, 3.3V. (WARNING: attempting to power the part at a voltage exceeds the maximum rating of 3.6V is likely to damage it.)
- Connect GND to common power/data ground.
- Connect the SCL pin to the I2C clock SCL pin on your Arduino.
- Connect the SDA pin to the I2C data SDA pin on your Arduino.

The MC3419 has a default I2C address of 0x4C and it can be changed to 0x6C by tying the SDO pin to VDD.
2.2 SPI INTERFACE

The EV3419A evaluation board can be easily wired to any microcontroller. This example shows a typical Arduino DUE platform. Please refer to the illustration below for connecting the related pins and then port the code to get the raw X, Y, Z sensor data.

- Connect **DVDD** to the power supply, **3.3V**. (WARNING: attempting to power the part at a voltage exceeds the maximum rating of 3.6V is likely to damage it.)
- Connect **GND** to common power/data ground.
- Connect **SCL** to ICSP-3 as Serial Clock.
- Connect **SDO** to ICSP-1 as Master Input, Slave Output.
- Connect **SDA** to ICSP-4 as Master Output, Slave Input.
- Connect **SCS** to digital I/O pin **10** as Slave Chip Select.
3 DEMO

3.1 Download the Driver from GitHub

To begin reading sensor data, you will need to download the MC3419 Library from the GitHub repository and simply click URL to clone or download the driver file.

https://github.com/mcubemems/mCube_mc34x9_arduino_driver

Download with ZIP format from link above and check the zip file consisting of MC34X9.cpp, MC34X9.h and examples directory.

An excellent tutorial on Arduino library installation is located at:

https://www.arduino.cc/en/guide/libraries#toc4

You may need to create the library subfolder if it is your first library files. Then just restart the IDE.
### 3.2 LOAD THE DEMO

Open File->Examples->MC34X9-> MC34X9_demo and upload to your Arduino while it is wired to the sensor.

If you need the sensor running on SPI, please configure the bus as SPI in the MC34X9_demo.ino shown as below. Otherwise, the default is I2C bus.

```cpp
/*** bSpi: I2C/SPI bus selection. SPI: 0, I2C: 1 ***/
const uint8_t bSpi = 1;
```

Setup I2C interface as 2.1 will get I2C address 0x4C.

Setup SPI interface as 2.2 will get SPI select pin 10.

```cpp
/* Chip Select & Address */
uint8_t chipSelect = 0;
const uint8_t SPIChipSelectPin = 10; // SPI chipSelectPin
const uint8_t I2CAddress = 0x4C; // I2C address
```
Now open the serial terminal window at 115,200 baud rate speed to begin the test.

You will see the output from the serial terminal showing the current range scale and resolution of the sensor in the first three lines followed by two lines of output sensor data at some output data rate which depict “raw count” data for line 1: X: 174 Y: 21 Z: 4074 with 8G range, 16bit ADC resolution. Line 2 indicates the SI units for measuring acceleration as X: 0.42 m/s^2 Y: 0.05 m/s^2 Z: 9.75 m/s^2.

To enable FIFO mode, which is controlled by the configurations below. FIFO size could be set up to maximum 31 samples. Enable FIFO threshold interrupt will automatically enable FIFO feature.

```c
/*** FIFO control ***/
int FIFO_THRE_SIZE = 30;
/* FIFO Interrupt */
const bool enableFifoThrINT = false;
/* For FIFO feature, enable FIFO interrupt will automatically enable FIFO feature */
bool enableFIFO = false;
```
To control embedded motion blocks, which is enabled by the configurations below. Enable individual motion mode will also enable corresponding interrupt.

*These modes can only be enabled separately.*

```c
/*** Motion control ***/
/* Enabling motion feature below also enables corresponded motion interrupt */
const bool enableTILT = false;
const bool enableFLIP = false;
const bool enableANYM = true;
const bool enableSHAKE = false;
const bool enableTILT_35 = false;
```
4 LIBRARY REFERENCE

4.1 CREATE MCUBE_MC34X9 OBJECT

You can create the MCUBE_MC34X9 object with:

```cpp
MC34X9 MC34X9_acc = MC34X9();
```

4.2 INITIALIZE AND CONFIGURE SENSOR

Initialize and configure the sensor with:

```cpp
MC34X9_acc.start(bSpi, chipSelect);
```

Wake up sensor with your own configuration, it will follow the factory default setting:

```cpp
MC34X9_acc.wake();
```

Stop sensor to change setting:

```cpp
MC34X9_acc.stop();
```

4.3 SET RANGE

Set the accelerometer max range to ±2g, ±4g, ±8g, ±16g or ±12g with:

```cpp
MC34X9_acc.SetRangeCtrl(MC34X9_RANGE_8G);
```

4.4 READ RANGE

Read the current range with:

```cpp
MC34X9_acc.GetRangeCtrl();
```

It returns: 0 for ±2g, 1 for ±4g, 2 for ±8g, 3 for ±16g, 4 for ±12g.

4.5 SET SAMPLING RATE

Set the accelerometer sampling rate with:

```cpp
MC34X9_acc.SetSampleRate(MC34X9_SR_DEFAULT_1000Hz);
```
4.6 READ SAMPLING RATE

Read the current sampling rate with:

```c
MC34X9_acc.GetSampleRate();
```

It returns sampling rate from 25 ~ 1000 Hz.

4.7 CONFIG MOTION FEATURE

To configure motion feature, open MC34X9.h

Set the threshold and debounce value used by the tilt and flip logic for activity detection:

```c
#define s_bCfgFTThr 100
#define s_bCfgFTDebounce 50
```

Set the threshold and debounce value used by the any motion logic for activity detection:

```c
#define s_bCfgANYMThr 200
#define s_bCfgANYMDebounce 100
```

Set the threshold, duration and count value used by the shake logic for activity detection:

```c
#define s_bCfgShakeThr 300
#define s_bCfgShakeP2PDuration 10
#define s_bCfgShakeCount 1
```

Set the threshold and duration value used by the tilt35 logic for activity detection:

```c
#define s_bCfgTILT35Thr 20
#define s_bCfgTILT35Timer MC34X9_TILT35_2p0
```

4.8 CONFIG INTERRUPT MODE

Configure the FIFO interrupt mode with:

```c
MC34X9_acc.SetFIFOINTCtrl(false, false, enableFifoThrINT);
```
Configure the motion interrupt mode with:

```c
MC34X9_acc.SetINTCtrl(enableTILT, enableFLIP, enableANYM, enableSHAKE, enableTILT_35);
```

MC34X9 has 5 motion interrupt modes – TILT | FILP | ANYM | SHAKE | TILT35.

*These modes can only be enabled separately.*

### 4.9 READ RAW COUNT DATA

Read the raw count data and SI unit measurement with:

```c
MC34X9_acc.readRawAccel();
```
5  DOWNLOADS

5.1  MC3419 ACCELEROMETER DATASHEET AND QUICK START GUIDE

https://mcubemems.com/product/mc3419-3-axis-accelerometer/

5.2  MC34X9 DRIVER AT GITHUB

https://github.com/mcubemems/mCube_mc34x9_arduino_driver

5.3  ALL OTHER MCUBE DOCUMENTATION

https://mcubemems.com/resources-support/resources/
Above is a schematic on EV3419A. This is the factory preset when receiving the part. For other options, please refer to the following table:

<table>
<thead>
<tr>
<th>Interface</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPI or I2C 0x4C <strong>(Factory default)</strong></td>
<td>100KΩ</td>
<td>DNI¹</td>
<td>10KΩ</td>
</tr>
<tr>
<td>SPI or I2C 0x6C</td>
<td>100KΩ</td>
<td>10KΩ</td>
<td>DNI</td>
</tr>
</tbody>
</table>

**R4, R5:** Install ~4.7KΩ as the I2C pull-ups if using I2C mode and there is no I2C pull-ups installed. (DNI by default)

Note: It is recommended not to install more than one pull-ups per I2C bus.

**R6:** Install ~4.7KΩ pull-up resistor if setting the sensor interrupt pin to open-drain. (DNI by default)

**R7:** Sensor’s driving current can be measured by putting an ammeter in place of R7.

The physical location of the resistor is in the diagram in Section 8.

¹ DNI: Do Not Install
## 7 BILL OF MATERIALS

<table>
<thead>
<tr>
<th>Item</th>
<th>Part</th>
<th>Value</th>
<th>Package</th>
<th>Vendor</th>
<th>P/N</th>
<th>Install</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>C1</td>
<td>0.1µF</td>
<td>CAP-0603</td>
<td>Walsin</td>
<td>0603B104K500</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>J1</td>
<td>DIL10-A</td>
<td>DIL10-A</td>
<td>-</td>
<td>-</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>R1</td>
<td>100K</td>
<td>RES-0603</td>
<td>Walsin</td>
<td>WR06X103JTL</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>R3</td>
<td>10K</td>
<td>RES-0603</td>
<td>Walsin</td>
<td>WR06X102JTL</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>R7</td>
<td>0R</td>
<td>RES-0603</td>
<td>Walsin</td>
<td>WR06X000PTL</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>U1</td>
<td>MC3419</td>
<td>LGA-12_2.0 x 2.0</td>
<td>mCube</td>
<td>MC3419</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>R2</td>
<td>10K</td>
<td>RES-0603</td>
<td>Walsin</td>
<td>WR06X472JTL</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>R4</td>
<td>4.7K</td>
<td>RES-0603</td>
<td>Walsin</td>
<td>WR06X472JTL</td>
<td>No</td>
</tr>
<tr>
<td>9</td>
<td>R5</td>
<td>4.7K</td>
<td>RES-0603</td>
<td>Walsin</td>
<td>WR06X472JTL</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>R6</td>
<td>4.7K</td>
<td>RES-0603</td>
<td>Walsin</td>
<td>WR06X472JTL</td>
<td>No</td>
</tr>
<tr>
<td>11</td>
<td>R9</td>
<td>10K</td>
<td>RES-0603</td>
<td>Walsin</td>
<td>WR06X472JTL</td>
<td>No</td>
</tr>
</tbody>
</table>
8 FABRICATION PRINT

NOTE: All dimensions are in millimeters.
# REVISION HISTORY

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2020-02-05</td>
<td>APS-045-0033v1.0</td>
<td>First release.</td>
</tr>
</tbody>
</table>
10 LEGAL

1. M-CUBE reserves the right to make corrections, modifications, enhancements, improvements and other changes to its products and to this document at any time and discontinue any product without notice. The information contained in this document has been carefully checked and is believed to be accurate. However, M-CUBE shall assume no responsibilities for inaccuracies and make no commitment to update or to keep current the information contained in this document.

2. M-CUBE products are designed only for commercial and normal industrial applications and are not suitable for other purposes, such as: medical life support equipment; nuclear facilities; critical care equipment; military/aerospace; automotive; security or any other applications, the failure of which could lead to death, personal injury or environmental or property damage. Use of the products in unsuitable applications are at the customer's own risk and expense.

3. M-CUBE shall assume no liability for incidental, consequential or special damages or injury that may result from misapplication or improper use of operation of the product.

4. No license, express or implied, by estoppel or otherwise, to any intellectual property rights of M-CUBE or any third party is granted under this document.

5. M-CUBE makes no warranty or representation of non-infringement of intellectual property rights of any third party with respect to the products. M-CUBE specifically excludes any liability to the customers or any third party regarding infringement of any intellectual property rights, including the patent, copyright, trademark or trade secret rights of any third party, relating to any combination, machine, or process in which the M-CUBE products are used.

6. Examples of use described herein are provided solely to guide use of M-CUBE products and merely indicate targeted characteristics, performance and applications of products. M-CUBE shall assume no responsibility for any intellectual property claims or other problems that may result from applications based on the examples described herein.

7. Information described in this document including parameters, application circuits and its constants and calculation formulas, programs and control procedures are provided for the purpose of explaining typical operation and usage. “Typical” parameters that may be provided in M-CUBE data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters including “Typical,” must be validated for each customer application by customer’s technical experts. In no event shall the information described be regarded as a guarantee of conditions or characteristics of the products. Therefore, the customer should evaluate the design sufficiently as whole system under the consideration of various external or environmental conditions and determine their application at the customer’s own risk. M-CUBE shall assume no responsibility or liability for claims, damages, costs and expenses caused by the customer or any third party, owing to the use of the above information.

is a trademark of M-CUBE, Inc.

M-CUBE and the M-CUBE logo are trademarks of M-CUBE, Inc.,

All other product or service names are the property of their respective owners.

© M-CUBE, Inc. 2020. All rights reserved.