Freescale Semiconductor
Application Note

Document Number: AN4458
Rev. 0, 06/2012

Using the Xtrinsic FXOS8700CQ Accelerometer and Magnetometer Vector-Magnitude Function

by: Talat Ozyagcilar
Applications Engineer

1 Introduction

This application note demonstrates the FXOS8700CQ vector-magnitude function for both the accelerometer and magnetometer with use-case examples. Use cases where this application note is useful include:

Accelerometer vector-magnitude function:

- Detection of linear acceleration exceeding a threshold
- Detection of change of tilt angle exceeding a threshold.

Magnetometer vector-magnitude function:

- Detection of magnetic jamming.

1.1 Keywords

Vector Magnitude, Motion Detection, Tilt Detection, Magnetic Field Strength, Linear-Acceleration Detection, Tilt-Angle Detection, Holster, Smart Cover, Inertial, Vector Magnitude

Contents

1.2 Related Documentation

The FXOS8700CQ 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 FXOS8700CQ.
3. In the Refine Your Result pane on the left, click on the Documentation link.

2 Vector-Magnitude Functions

The accelerometer vector-magnitude function raises an event flag[1] when the magnitude of the current accelerometer measurement minus a programmable reference accelerometer reading exceeds a programmable threshold for a programmable period of time:

$$\sqrt{(a\_x\_out - a\_x\_ref)^2 + (a\_y\_out - a\_y\_ref)^2 + (a\_z\_out - a\_z\_ref)^2} > A\_VECM\_THS \qquad \textbf{Eqn. 1}$$

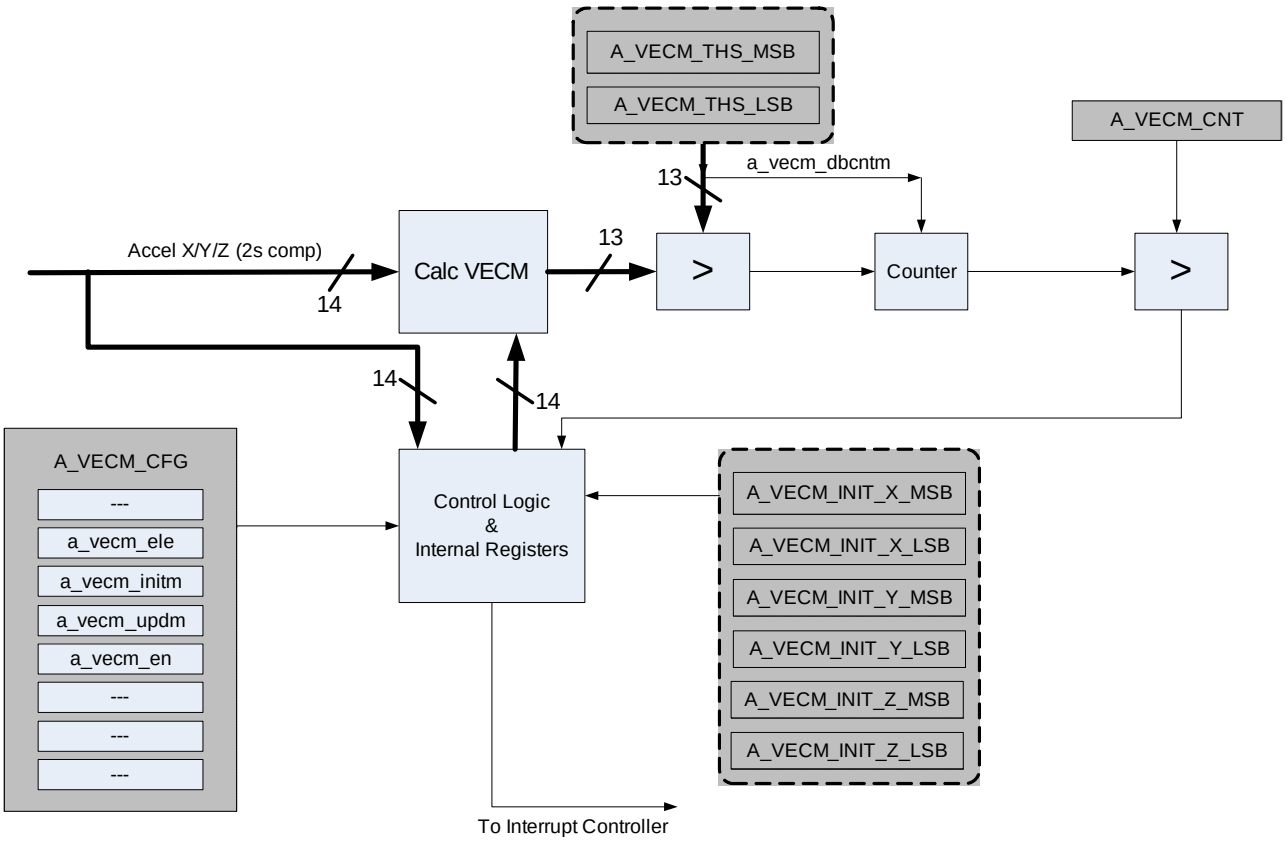

Figure 1. Accelerometer vector-magnitude function block diagram

1. An event flag sets src_a_vecm bit in the INT_SOURCE register, which can be set up to set/clear INT1/INT2 pins.

Similarly, the magnetometer vector-magnitude function raises an event flag[1] when the magnitude of the magnetometer measurement minus a programmable reference magnetic reading exceeds a programmable threshold for a programmable period of time:

$$\sqrt{(m\_x\_out - m\_x\_ref)^2 + (m\_y\_out - m\_y\_ref)^2 + (m\_z\_out - m\_z\_ref)^2} > M\_VECM\_THS \quad \textbf{Eqn. 2}$$

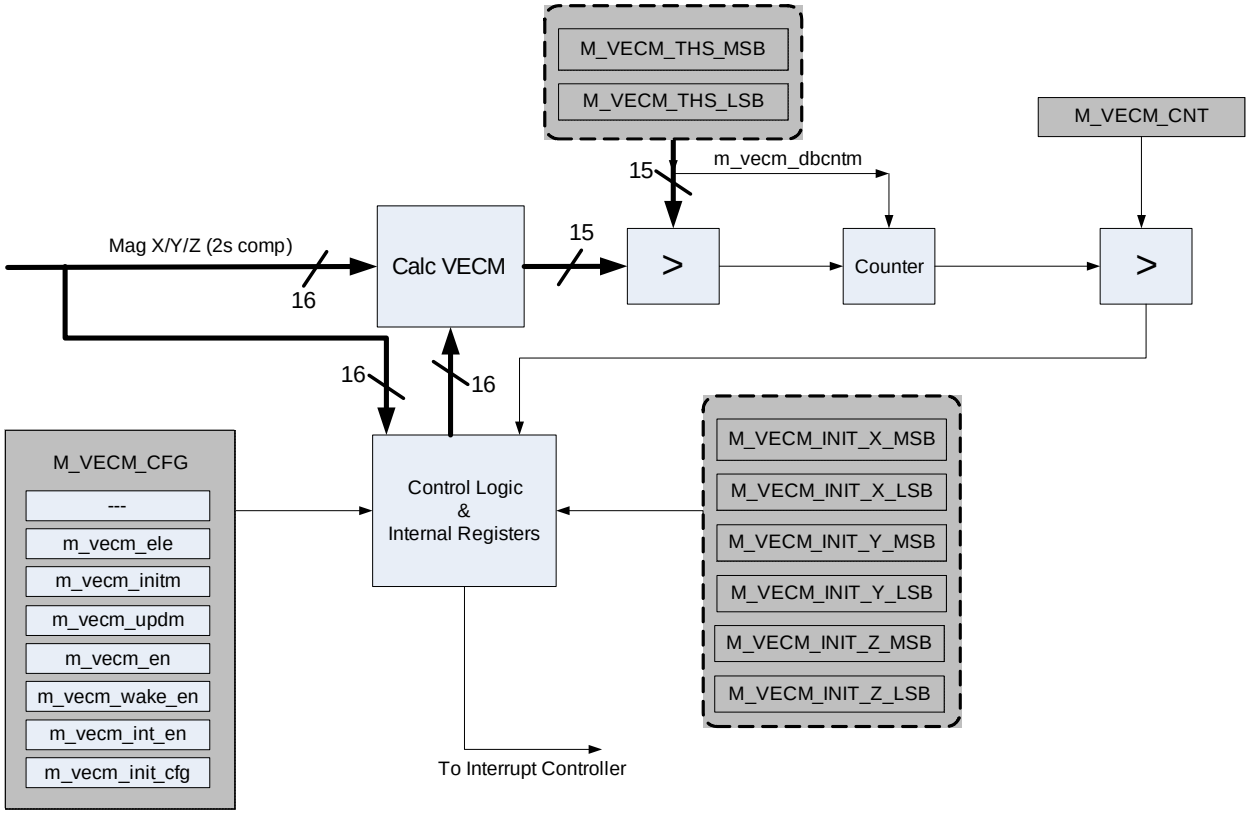

Figure 2. Magnetometer vector-magnitude function block diagram

Equations 1 and 2 are tests of the form:

$$|\boldsymbol{p} - \boldsymbol{q}| = \sqrt{(\boldsymbol{p} - \boldsymbol{q}) \cdot (\boldsymbol{p} - \boldsymbol{q})} > r \quad \textbf{Eqn. 3}$$

where $\boldsymbol{p}$ is the accelerometer or magnetometer reading, $\boldsymbol{q}$ is the programmable offset and r is the event threshold.

Equation 3 can be rewritten as:

$$\sqrt{\boldsymbol{p} \cdot \boldsymbol{p} + \boldsymbol{q} \cdot \boldsymbol{q} - 2\boldsymbol{p} \cdot \boldsymbol{q}} = \sqrt{p^2 + q^2 - 2pq\cos\alpha} > r \quad \textbf{Eqn. 4}$$

where p and q are the magnitudes of the vectors $\boldsymbol{p}$ and $\boldsymbol{q}$, and α is the angle between the vectors $\boldsymbol{p}$ and $\boldsymbol{q}$.

1. An event flag sets src_m_vecm bit in the M_INT_SOURCE register, which can be set up to set/clear INT1/INT2 pins.

For the special case where ***p*** and ***q*** have the same magnitude, |***p***| = |***q***|, Equation 4 can be written as:

$$\sqrt{2p^2 - 2p^2 \cos\alpha} = \sqrt{4p^2 \sin^2\left(\frac{\alpha}{2}\right)} = \left|2p \sin\left(\frac{\alpha}{2}\right)\right| > r \qquad \textit{Eqn. 5}$$

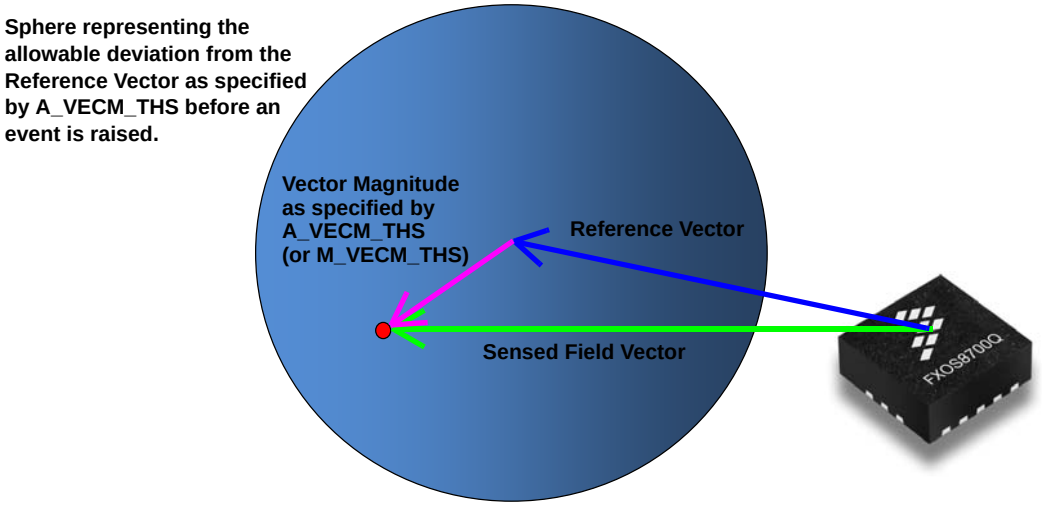

Figure 3. Operating principle of the vector-magnitude feature

The operating principle of this function is depicted in Figure 3. The vector-magnitude function simply compares the magnitude of the difference between the sensed field vector and the reference field vector against the magnitude specified by *A_VECM_THS* (or *M_VECM_THS*). If the former is greater than the latter for a minimum duration of time specified by *A_VECM_CNT* (or *M_VECM_CNT*), then an event is raised.

2.1 Accelerometer vector-magnitude function

The accelerometer vector-magnitude function use cases below are discussed:

- Detection of linear acceleration exceeding a threshold
- Detection of change of tilt angle exceeding a threshold.

Use Case 1: Detection of linear acceleration exceeding a threshold

An accelerometer sensor measures the difference between linear acceleration and the local gravitational field as measured in its frame of reference. Since the accelerometer may be mounted at an arbitrary orientation in the device or object being monitored, it is convenient to introduce a rotation matrix $\boldsymbol{R}_0$ defining the accelerometer mounting orientation relative to the fixed earth frame of reference. The accelerometer reading ***G*** is then:

$$\boldsymbol{G} = \boldsymbol{R}_0(\boldsymbol{a} - \boldsymbol{g}) \qquad \textit{Eqn. 6}$$

where ***g*** is the earth's gravitational field vector, which has constant magnitude and constant downwards direction in the earth's reference frame, and ***a*** is the device or object's acceleration in the earth's reference frame.

When the device or object is stationary, there can be no linear acceleration and the accelerometer reading $\boldsymbol{G}_{ref}$ will be:

$$\boldsymbol{G}_{ref} = -\boldsymbol{R}_0\boldsymbol{g} \quad \textit{Eqn. 7}$$

If the reading $\boldsymbol{G}_{ref}$ is recorded and stored as the reference accelerometer reading, the vector magnitude threshold test is:

$$\left|\boldsymbol{G} - \boldsymbol{G}_{ref}\right| = \left|\boldsymbol{R}_0(\boldsymbol{a} - \boldsymbol{g}) + \boldsymbol{R}_0\boldsymbol{g}\right| = \left|\boldsymbol{R}_0\boldsymbol{a}\right| = \left|\boldsymbol{a}\right| > r \quad \textit{Eqn. 8}$$

The mounting orientation rotation matrix $\boldsymbol{R}_0$ cancels from Equation 8 since the magnitude of the acceleration vector $\boldsymbol{a}$ is independent of any rotation of the sensor. The threshold test is therefore completely independent of mounting orientation and is triggered whenever the acceleration magnitude exceeds the user-defined threshold r which is stored in the *A_VECM_THS* register.

Use Case 2: Detection of change of tilt angle exceeding a threshold

Accelerometers are widely used for determining the orientation angles of a device held in the earth's gravitational field. If there is negligible linear acceleration, the accelerometer reading $\boldsymbol{G}_{ref}$ with the device in its default orientation, represented by rotation matrix $\boldsymbol{R}_0$, will again be:

$$\boldsymbol{G}_{ref} = -\boldsymbol{R}_0\boldsymbol{g} \quad \textit{Eqn. 9}$$

If the device orientation changes, the accelerometer reading due to gravity alone will be:

$$\boldsymbol{G} = -\boldsymbol{R}\boldsymbol{g} \quad \textit{Eqn. 10}$$

where $\boldsymbol{R}$ is the rotation matrix representing the new orientation.

If $\boldsymbol{G}_{ref}$ is stored as the reference accelerometer reading for the modulus vector difference calculation, the threshold test becomes:

$$\left|\boldsymbol{G} - \boldsymbol{G}_{ref}\right| = \left|-\boldsymbol{R}\boldsymbol{g} + \boldsymbol{R}_0\boldsymbol{g}\right| > r \quad \textit{Eqn. 11}$$

Substituting Equation 5, which is applicable because the rotations have no effect on the magnitude of the measured earth's gravitational field, gives:

$$\left|\boldsymbol{G} - \boldsymbol{G}_{ref}\right| = \left|2g\sin\left(\frac{\alpha}{2}\right)\right| < r \quad \textit{Eqn. 12}$$

where α is the change in tilt angle between the reference measurement $\boldsymbol{G}_{ref}$ and the current measurement $\boldsymbol{G}$.

For a given threshold r, the event will trigger for a change in tilt angle α given by:

$$|\alpha| > 2\sin^{-1}\left(\frac{r}{2g}\right) \quad \textit{Eqn. 13}$$

If r is set to 0.3g, as shown in the code Example 1 below, the event will trigger for a change in tilt angle α exceeding 17.25°.

In Example 1, the Python code snippet sets up the device for detection of a change in tilt angle exceeding 17.25° from the horizontal plane. The device is set up so that once an event is triggered, an interrupt will be generated on the INT1 pin. The user application will read register INT_SOURCE (0x0C) in an interrupt service routine, thereby clearing the interrupt. This example uses polling for checking the state of the INT1 pin by reading the INT_SOURCE interrupt status register.

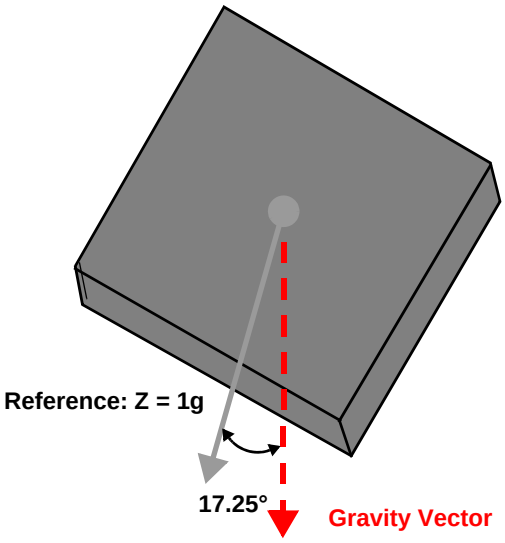

Figure 4. Calculations reveal that tilting the PCB about ±17° with respect to the horizontal plane will trigger a vector-magnitude event

Example 1. Python code snippet for accelerometer vector-magnitude function

```
# Issue a soft reset

write_byte( [0x2B, 0x40] )
time.sleep(0.1)

# 0.3G
# A_VECM_THS_MSB, LSB  = 0x84CC => 1228 *0.224mg = ~300mg
# 0x8000 designates the debounce behavior

write_byte( [0x60, 0x84] ) # write 0x84 to reg 0x60
write_byte( [0x61, 0xCC] )

# A_VECM_CNT = 1 * 20ms  = 20ms
# ! - steps double in hybrid mode

write_byte( [0x62, 0x01] )

# A_VECM_INITX/Y/Z
# Set Z only as 1g

write_byte( [0x63, 0x00] )
write_byte( [0x64, 0x00] )

write_byte( [0x65, 0x00] )
```

```
write_byte( [0x66, 0x00] )

write_byte( [0x67, 0x10] )
write_byte( [0x68, 0x00] )

# A_VECM_CFG
# a_vecm_ele = 1 => event latching enabled
# a_vecm_initm = 1 => use A_VECM_INITX/Y/Z as initial reference
# a_vecm_updm = 1 => do not update initial reference
# a_vecm_en = 1 => enable acceleration vector magnitude detection feature

write_byte( [0x5F, 0x78] )

# enable interrupts for the feature using CTRL_REG4

write_byte( [0x2D, 0x02] )

# route interrupts to INT1 pin using CTRL_REG5

write_byte( [0x2E, 0x02] )

# Setup device for hybrid mode, enable hybrid mode auto-increment,
# ODR = 50Hz, OSR=32, go to ACTIVE mode using M_CTRL_REG1, M_CTRL_REG2 and
# CTRL_REG1

write_byte( [0x5B, 0x1F] )
write_byte( [0x5C, 0x20] )
write_byte( [0x2A, 0x19] )

# Wait for INT1 to assert and clear interrupt by reading register INT_SOURCE (0x0C)

while( True ):

   transition = aa_gpio_change( handle, 100 )

   if (transition & INT1_PIN ) == INT1_PIN:
      print "No interrupt..."
      continue

   print "Interrupt occured!!!"
   (count, dataIn) = read_dev( 0x0C, 1 )
   print "0x0C =0x%X" % dataIn[0]
```

2.2 Magnetometer vector-magnitude function

The magnetometer vector-magnitude function use case below is discussed:

- Detection of magnetic jamming

Use Case 3: Detection of magnetic jamming

The magnetometer reading $\boldsymbol{B}$ is the vector sum of the:

- Geomagnetic field $\boldsymbol{b}$, rotated by the current orientation matrix $\boldsymbol{R}$
- The calibration hard-iron offset $\boldsymbol{V}$ for the PCB and sensor
- Any extraneous magnetic jamming field $\boldsymbol{J}$.

$$\boldsymbol{B} = \boldsymbol{Rb} + \boldsymbol{V} + \boldsymbol{J} \qquad \textit{Eqn. 14}$$

If the calibration hard-iron offset $\boldsymbol{V}$ is loaded as the reference reading $\boldsymbol{B}_{ref}$, the alert will be triggered when:

$$\left|\boldsymbol{B} - \boldsymbol{B}_{\text{ref}}\right| = \left|\boldsymbol{B} - \boldsymbol{V}\right| = \left|\boldsymbol{Rb} + \boldsymbol{J}\right| > r \qquad \textit{Eqn. 15}$$

Since $\boldsymbol{Rb}$ has magnitude b, setting the threshold $r = 10b$ will result in the detection of a jamming field $\boldsymbol{J}$ exceeding the range $9b$ to $11b$ depending on whether it is aligned or anti-aligned with the rotated geomagnetic field $\boldsymbol{Rb}$.

The magnetic jamming alert can be used to flag that the compass heading information will be inaccurate or to detect the insertion of a smartphone into a magnetic holster or a smart cover.

$$\boldsymbol{B}_{ref} = \boldsymbol{V} = \begin{pmatrix} V_x \\ V_y \\ V_z \end{pmatrix} = \frac{1}{2}\begin{pmatrix} m\_x\_out_{max} + m\_x\_out_{min} \\ m\_y\_out_{max} + m\_y\_out_{min} \\ m\_z\_out_{max} + m\_z\_out_{min} \end{pmatrix} \qquad \textit{Eqn. 16}$$

The code below demonstrates how to utilize Use Case 3. With every data acquisition hard-iron offset $\boldsymbol{V}$ is estimated using Equation 16. The estimated hard-iron vector is then used as the reference vector $\boldsymbol{B}_{ref}$ as specified in M_VECM_INITX/Y/Z registers for the next acquisition. Picking threshold multiplier $r = 2$, and $\boldsymbol{b}$ = 50 µT (in Phoenix, AZ) makes the threshold 2(50 µT) = 100 µT = 1000 counts; this is set via M_VECM_THS register.

With the code running the user can use a rare-earth magnet to mimic an external magnetic interference depicted as $\boldsymbol{J}$ in Equation 14 and Equation 15.

NOTE

Bringing a magnet in close proximity to the device may result in permanently damaging the sensor. The keep-out radius depends on the strength of the magnet used. A rule-of-thumb is to not let the magnetic readings rail.

With these settings, an interrupt generated by the vector-magnitude function will indicate magnetic jamming. Applications can use this to notify the end-user that the compass readings may not be reliable

due to magnetic interference, or that a new calibration is required to compensate for the presence of the magnet.

The hard-iron estimation in Equation 16 is not very robust; it is used in this example for simplicity. Ideally, hard- and soft-iron compensation algorithms should incorporate the methods described in AN4246 and AN4248 (involving ellipsoid fits to constellations from recent data). This way hard- and soft-iron estimations take into account the changing magnetic environmental conditions and are less affected by spurious data.

In this example, the hard-iron estimation is updated with every acquisition. This results in part of the hard-iron interference introduced by the magnet's proximity being included in the hard-iron estimate. In general, how quickly the interference is detected will depend on the specifics of the estimation/compensation engine.

Example 2. Python code snippet for the magnetometer vector-magnitude function

```
# Issue soft reset

write_byte( [0x2B, 0x40] )
time.sleep(0.1)

# Initialize reference vector

refMagX = refMagXHi = refMagXLo = 0
refMagY = refMagYHi = refMagYLo = 0
refMagZ = refMagZHi = refMagZLo = 0

# Initialize min/max for hard iron estimate

magXMax = magYMax = magZMax = int16( 0x8000 )
magXMin = magYMin = magZMin = int16( 0x7FFF )

# Set threshold. 1000 counts = 100.0uT

magThreshold = 1000 # counts
magThresholdHi = (magThreshold & 0xFF00) >> 8
magThresholdLo = magThreshold & 0xFF

write_byte( [0x6A, 0x80 | magThresholdHi] )
write_byte( [0x6B, magThresholdLo] )

# M_VECM_CNT = 1 * 20ms  = 20ms
# ! - steps double in hybrid mode

write_byte( [0x6C, 0x01] )

# M_VECM_INIT_X_MSB...Z_LSB = refMagX/Y/Z

write_byte( [0x6D, refMagXHi] )
write_byte( [0x6E, refMagXLo] )

write_byte( [0x6F, refMagYHi] )
```

```
write_byte( [0x70, refMagYLo] )

write_byte( [0x71, refMagZHi] )
write_byte( [0x72, refMagZLo] )

# M_VECM_CFG
# m_vecm_ele = 1 => event latching enabled
# m_vecm_initm = 1 => use M_VECM_INITX/Y/Z as initial reference
# m_vecm_updm = 1 => do not update initial reference
# m_vecm_en = 1 => enable magnetometer vector magnitude detection feature

write_byte( [0x69, 0x7B] )

# enable interrupts for DRDY using CTRL_REG4

write_byte( [0x2D, 0x01] )

# route interrupts to INT1 pin using CTRL_REG5

write_byte( [0x2E, 0x01] )

# Setup device for hybrid mode, enable hybrid mode auto-increment,
# ODR = 50Hz, OSR=32, go to ACTIVE mode using M_CTRL_REG1, M_CTRL_REG2 and
# CTRL_REG1

write_byte( [0x5B, 0x1F] )
write_byte( [0x5C, 0x20] )
write_byte( [0x2A, 0x19] )

# Wait for INT1 to assert and clear interrupt by reading register INT_SOURCE (0x0C)
# or reading sensor data (0x01...0x06, 0x33...0x38) depending on the function that
# generated the interrupt

while( True ):

  transition = aa_gpio_change( handle, 100 )
  if (transition & INT1_PIN ) == INT1_PIN:
    #print "No interrupt..."
    continue

  print "\nODR cycle ==============>\n"

  (count, dataIn) = read_dev( 0x5E, 1 )

  if dataIn[0] & 0x02 == 0x02:

    print "Interrupt due to Magnetometer Vector Magnitude feature"
    print "Magnetic Jamming detected, resetting hard iron estimate..."

    # Reset hard iron estimate

    magXMax = magYMax = magZMax = int16( 0x8000 )
    magXMin = magYMin = magZMin = int16( 0x7FFF )
```

```
(count, dataIn) = read_dev( 0x0C, 1 )

if dataIn[0] & 0x01 == 0x01:

   print "Interrupt due to data ready"

   (magX, magY, magZ, accX, accY, accZ) = dev_display_sens_data()

   # Calculate VECM

   magnitude = math.sqrt( math.pow( (magX - refMagX), 2) +
      math.pow( (magY - refMagY), 2) + math.pow( (magZ - refMagZ), 2) )

   print "Magnetometer Vector Magnitude = %f uT" % (magnitude / 10)

   if magnitude > magThreshold:

      print "Magnetometer Vector Magnitude interrupt should have “,
      print “been triggered"

   # Finally calculate new hard iron estimate and put in reference
   # registers

   if magX < magXMin:

      magXMin = magX

   if magY < magYMin:

      magYMin = magY

   if magZ < magZMin:

      magZMin = magZ

   if magX > magXMax:

      magXMax = magX

   if magY > magYMax:

      magYMax = magY

   if magZ > magZMax:

      magZMax = magZ

   refMagX = (magXMax + magXMin) / 2
   refMagY = (magYMax + magYMin) / 2
   refMagZ = (magZMax + magZMin) / 2

   print "refMagX = %d, refMagY = %d, refMagZ = %d" % (refMagX, refMagY, refMagZ)
```

```
    refMagXHi = (refMagX & 0xFF00) >> 8
    refMagXLo = (refMagX & 0x00FF)

    refMagYHi = (refMagY & 0xFF00) >> 8
    refMagYLo = (refMagY & 0x00FF)

    refMagZHi = (refMagZ & 0xFF00) >> 8
    refMagZLo = (refMagZ & 0x00FF)

    # M_VECM_INIT_X_MSB...Z_LSB = refMagX/Y/Z

    write_byte( [0x6D, refMagXHi] )
    write_byte( [0x6E, refMagXLo] )

    write_byte( [0x6F, refMagYHi] )
    write_byte( [0x70, refMagYLo] )

    write_byte( [0x71, refMagZHi] )
    write_byte( [0x72, refMagZLo] )

  else:

    print "unexpected interrupt"
```

```
ODR cycle ==============>

Interrupt due to data ready
X MAG: 756, Y MAG: 361, Z MAG: -5051, X ACC: 110, Y ACC: 64, Z ACC: 4147
Magnetometer Vector Magnitude = 46.986807 uT
refMagX = 860, refMagY = 375, refMagZ = -4593

ODR cycle ==============>

Interrupt due to data ready
X MAG: 764, Y MAG: 357, Z MAG: -5040, X ACC: 118, Y ACC: 34, Z ACC: 4149
Magnetometer Vector Magnitude = 45.754672 uT
refMagX = 860, refMagY = 375, refMagZ = -4593

ODR cycle ==============>

Interrupt due to data ready
X MAG: 763, Y MAG: 362, Z MAG: -5049, X ACC: 116, Y ACC: 42, Z ACC: 4152
Magnetometer Vector Magnitude = 46.638396 uT
refMagX = 860, refMagY = 375, refMagZ = -4593
```

Figure 5. Sample output after rotation of PCB for initial hard-iron calibration

After the execution starts, it is necessary to rotate the PCB around in order to obtain an initial estimation of the ambient hard-iron interference (seen in *refMagX/Y/Z* in Figure 5). The vector magnitude at this point is 46.6 µT which is expected in this geographical locale (Tempe, AZ).

```
ODR cycle ==============>

Interrupt due to data ready
X MAG: 257, Y MAG: -873, Z MAG: -4619, X ACC: 116, Y ACC: 54, Z ACC: 4140
Magnetometer Vector Magnitude = 96.621995 uT
refMagX = 761, refMagY = -54, refMagZ = -4593

ODR cycle ==============>

Interrupt due to data ready
X MAG: 230, Y MAG: -882, Z MAG: -4697, X ACC: 110, Y ACC: 48, Z ACC: 4141
Magnetometer Vector Magnitude = 98.912133 uT
refMagX = 748, refMagY = -55, refMagZ = -4593

ODR cycle ==============>

Interrupt due to Magnetometer Vector Magnitude feature
Magnetic Jamming detected, resetting hard iron estimate...
Interrupt due to data ready
X MAG: 211, Y MAG: -901, Z MAG: -4712, X ACC: 114, Y ACC: 38, Z ACC: 4160
Magnetometer Vector Magnitude = 100.908176 uT
Magnetometer Vector Magnitude interrupt should have been triggered
refMagX = 211, refMagY = -901, refMagZ = -4712
```

Figure 6. Sample output with increasing hard-iron interference

As the magnet gets closer to the device, the hard-iron estimate is affected as displayed in Figure 6 and due to the specifics of this example, magnetic jamming is still detected once the vector magnitude goes beyond 100 µT.

Once jamming has been detected, it is up to the system designer to handle this condition appropriately. One possible scenario is to notify the end-user about a possibly inaccurate heading, resetting the hard-iron estimate, and requesting a recalibration in order to absorb the recently introduced interference into the hard-iron estimation.

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

AN4458
Rev. 0
06/2012