Freescale Semiconductor
Application Note

Document Number: AN4446
Rev. 0, 08/2012

Using the MMA9550 Intelligent Accelerometer to Detect the High Point of a Vertical Trajectory

by Fengyi Li, Applications Engineer

1 Introduction

One of the classic applications of an accelerometer sensor is the detection of the freefall condition when the sensor is no longer supported and is following a ballistic trajectory under gravity. If a notebook PC with an integrated accelerometer sensor is held and then dropped without spinning, the accelerometer reading falls from 1g to 0g at the moment the notebook leaves the owner's hands and remains zero until impact with the ground. Similarly, if the notebook is thrown into the air without spinning, the accelerometer reading again drops to zero once the notebook leaves the owner's hands and remains zero until impact. In the presence of spinning, a centripetal acceleration will be present which complicates matters and prevents the accelerometer reading reaching zero.

This application note addresses the related problem of how to detect the high point (or apogee) of a vertical trajectory in the absence of spinning. The instantaneous accelerometer reading cannot be used because the reading is zero from the moment it leaves the thrower's hands until the point of impact with the ground. A more advanced approach is required which integrates the accelerometer reading to predict the high point of the trajectory where the vertical

Contents

freescale Xtrinsic

velocity is, by definition, zero. The Freescale MMA9550 smart accelerometer sensor, which contains a processor and memory, is ideally suited for problems such as this. Reference C code to implement the algorithm on the MMA9550 is therefore included with this application note (See Example 1).

The following Freescale Application Notes describe related topics that may be useful to the reader:

- *Tilt Sensing Using Linear Accelerometers* (document number AN3461), describes the use of accelerometers for tilt estimation and portrait or landscape selection.
- *High Precision Calibration of a Three Axis Accelerometer* (document number AN4399), describes how a product containing a consumer grade accelerometer can be re-calibrated after manufacture to achieve a high level of accuracy.

2 Summary

In the absence of spinning, an accelerometer sensor which is dropped or thrown will report near zero g acceleration from the moment it is released from the thrower's hands to the point it impacts the ground. The instantaneous accelerometer reading cannot therefore be used to determine the high point of the trajectory.

Integration of the accelerometer reading minus 1g during a vertical throw provides an estimate of the accelerometer vertical velocity at the point of release. The moment of release is easily determined by the appearance of near zero accelerometer readings.

The high point of a vertical trajectory has zero vertical velocity. The known 1g deceleration under gravity allows the calculation of the time interval from release to the high point of the trajectory from the ratio of the release velocity to 1g.

The Freescale MMA9550 smart accelerometer sensor is an ideal single device solution to perform the accelerometer integration. Example 1 includes reference C code for an MMA9550 implementation which flashes an LED at the top of the vertical trajectory.

3 Mathematical Model

Application Note AN3461 derives the reading $\boldsymbol{G}_p$ of an accelerometer sensor oriented at an arbitrary angle in the earth's gravitational field $\boldsymbol{g}$ and undergoing linear acceleration $\boldsymbol{a}_r(t)$ (as measured in the earth's reference frame *r*) as:

$$\boldsymbol{G}_p = \begin{pmatrix} G_{px} \\ G_{py} \\ G_{pz} \end{pmatrix} = \boldsymbol{R}(t)\{\boldsymbol{g} - \boldsymbol{a}_r(t)\} \qquad \textbf{Eqn. 1}$$

Where $\boldsymbol{R}(t)$ is the rotation matrix describing the instantaneous orientation of the sensor relative to the earth's reference frame.

At the moment that the accelerometer is released from the hand, it follows a ballistic freefall trajectory where $\boldsymbol{a}_r(t) = \boldsymbol{g}$. Equation 1 then simplifies to:

$$\boldsymbol{G}_p = \boldsymbol{0} \quad \textit{in freefall} \qquad \textbf{Eqn. 2}$$

The detection of a zero, or near zero, accelerometer reading is therefore indicative that the accelerometer has left the user's hands.

For the particular case where the accelerometer sensor is held at an arbitrary, but fixed, orientation and accelerated in the vertical axis z only, Equation 1 simplifies to:

$$\boldsymbol{G}_p = \begin{pmatrix} G_{px} \\ G_{py} \\ G_{pz} \end{pmatrix} = \boldsymbol{R} \begin{pmatrix} 0 \\ 0 \\ g - a_r(t) \end{pmatrix} \qquad \textbf{Eqn. 3}$$

Where g is the magnitude of the earth's gravitation field or approximately 9.81 ms^{-2}. The term $a_r(t)$ is now the acceleration in the vertical z axis with the sign convention that an acceleration downwards parallel to gravity has a positive sign.

The orientation matrix $\boldsymbol{R}$ can be eliminated from Equation 3 by taking the modulus of both sides to give:

$$|\boldsymbol{G}_p| = \sqrt{G_{px}^2 + G_{py}^2 + G_{pz}^2} = |g - a_r(t)| \qquad \textbf{Eqn. 4}$$

The application we are considering is the vertical throw of the accelerometer and the detection of the high point of the trajectory. The sequence of events will be:

1. The accelerometer will first be held steady with $\boldsymbol{a}_r(t) \approx 0$.
   The magnitude of the accelerometer reading $|\boldsymbol{G}_p|$ will therefore be approximately 1g.
2. When the hand holding of the accelerometer is lowered prior to the upwards throw, the linear acceleration $\boldsymbol{a}_r(t)$ will spike in the downwards (positive) direction, reduce towards zero, and then spike in the upwards (negative) direction when the hand stops in the lowered position.
   The magnitude of the accelerometer reading $|\boldsymbol{G}_p|$ in Equation 4 will therefore spike below 1g, return to 1g and then spike above 1g.
3. The vertical throw will start with a strong acceleration $\boldsymbol{a}_r(t)$ in the upwards (negative) direction.
   The magnitude of the accelerometer reading $|\boldsymbol{G}_p|$ in Equation 4 will therefore increase significantly above 1g.
4. From the moment of release until impact with the ground, the accelerometer is in a freefall ballistic trajectory and $\boldsymbol{a}_r(t) = \boldsymbol{g}$.
   The magnitude of the accelerometer reading $|\boldsymbol{G}_p|$ in Equation 4 is then zero until impact.

Figure 1 illustrates this sequence of events.

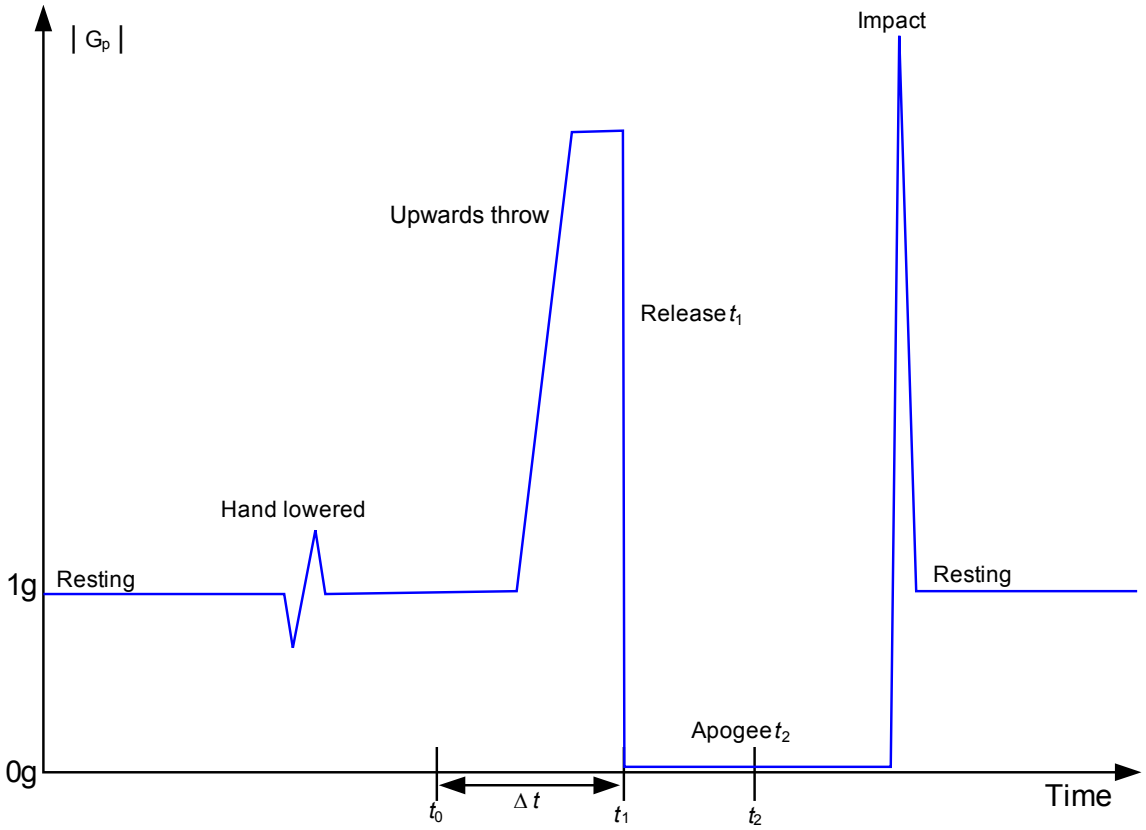

Figure 1. Magnitude of accelerometer output during throw

It is now necessary to impose one more constraint on the motion of the accelerometer. Because the release of the accelerometer from the hand will be detected by the zero *g* freefall condition, it is necessary to require that the downwards linear acceleration of the hand prior to the throw is always less than 1g so that the zero *g* freefall condition in Equation 4 is never reached accidentally prior to the throw. In practice this is easy to achieve using a smooth lowering of the hand rather than a jerky motion.

With this assumption, the modulus sign in Equation 4 can be removed giving:

$$|\boldsymbol{G}_p| = \sqrt{G_{px}^{\ 2} + G_{py}^{\ 2} + G_{pz}^{\ 2}} = g - a_r(\mathrm{t}) \qquad \textbf{Eqn. 5}$$

Equation 5 can now be integrated to give the vertical velocity $\boldsymbol{v}(t_1)$ at the moment of t_1 when the accelerometer is released from the hand at the end of throw. Where $\boldsymbol{v}(t_1)$ is positive in the downwards direction because of the sign convention that the positive z axis is downwards.

$$\boldsymbol{v}(t_1) = \int_{t_0}^{t_1} a_r(t)dt + \boldsymbol{v}(t_0) = \int_{t_0}^{t_1} (g - |\boldsymbol{G}_p|)dt + \boldsymbol{v}(t_0) = \int_{t_0}^{t_1} (g - \sqrt{G_{px}^{\ 2} + G_{py}^{\ 2} + G_{pz}^{\ 2}})dt + \boldsymbol{v}(t_0) \qquad \textbf{Eqn. 6}$$

If we identify the time t_0 (the start of the velocity integration) as being any time when hand holding the accelerometer is in the lowered position then, by definition, $v(t_0) = 0$. This vague definition of t_0 is acceptable because the modulus of the accelerometer reading equals 1g throughout the period when the

hand is lowered and the integrand is zero throughout this period. The integrand only becomes non-zero when the hand starts to accelerate upwards at the start of the throw.

The integration interval $t_1 - t_0$ will be set to a fixed value Δt which is long enough to capture the acceleration of the throw however not so long that it exceeds the period when the hand is held steady prior to the throw. A value of approximately 0.5s has been found suitable. Equation 6 can then be written as:

$$\boldsymbol{v}(t_1) = \int_{t_1 - \Delta t}^{t_1} (g - \sqrt{G_{px}^2 + G_{py}^2 + G_{pz}^2})dt \qquad \textbf{Eqn. 7}$$

The time t_1 when the accelerometer is released from the hand is easily determined by the freefall condition $|\boldsymbol{G}_p| \approx 0$. It is now simple to integrate from the release time t_1 to the time t_2 at the apogee or high point of the trajectory when the vertical velocity is zero.

$$v(t_2) = 0 = \int_{t_1}^{t_2} g dt + v(t_1) = g(t_2 - t_1) + v(t_1) \qquad \textbf{Eqn. 8}$$

$$\Rightarrow t_2 - t_1 = \frac{-v(t_1)}{g} = \int_{t_1 - \Delta t}^{t_1} \left(\frac{1}{g}\right)(\sqrt{G_{px}^2 + G_{py}^2 + G_{pz}^2} - g)dt \qquad \textbf{Eqn. 9}$$

Equation 9 is the physically intuitive result that the time interval $t_2 - t_1$ from the release of the accelerometer from the hand to reaching the high point of the trajectory is simply the upwards vertical velocity divided by the acceleration due to gravity.

4 MMA9550 Implementation

The continuous time integral in Equation 9 must be replaced by a summation for a discrete time software implementation. If the accelerometer is sampled at frequency f_S, Equation 9 can be re-written for sample number n as:

$$N = \sum_{i=0}^{M-1} \left(\frac{1}{g}\right)(\sqrt{G_{px}[n-i]^2 + G_{py}[n-i]^2 + G_{pz}[n-i]^2} - g) \qquad \textbf{Eqn. 10}$$

Where M is now the number of sampling intervals from the detection of the low g condition at release to the accelerometer reaching the top of the trajectory. The number of terms M in the summation should be selected so that $Mf_s \approx \Delta t$.

The sampling frequency for the MMA9550 implementation was selected to be 122 Hz. The integration interval Δt was selected to be 64/122s = 0.52459s requiring a delay line length of 64 elements to store the components of the sum in Equation 10.

The key code kernel for a C code implementation on the MMA9550 is listed Example 1.

Example 1. Reference C code

```
// accelerometer scaling constants
#define ONE_G 4096              // 1g on MMA9550 equals 4096 bits
#define ZEROP3_G 1229           // 0.3g (low g threshold) equals 1229 bits
```

```
// interval constants for 122Hz sampling rate
#define ZEROP5246_S 64         // 0.5246s integration interval
#define ZEROPTWO_S 24          // 0.2s interval for LED flash

// acceleration delay line size
#define DLSIZE ZEROP5246_S     // 0.5246s or 64 elements delay line size

// local variables
int i;                         // loop counter
int apogee_counter;            // countdown counter for LED flash at apogee
int DL_counter;                // delay line index
int gx, gy, gz;                // accelerometer channel data (bits)
float accel_mag;               // accelerometer magnitude (bits)
float DLSum;                   // sum of delay line entries
float DL[DLSIZE];              // delay line of normalised acceleration magnitudes

// zero the delay line
for (i = 0; i < DLSIZE; i++)
  DL[i] = 0.0F;

// set apogee countdown timer to invalid value -1
apogee_counter = -1;

// zero the circular delay line index
DL_counter = 0;

// loop forever
while (TRUE)
{
  // idle till sampling interrupt
  idle();

  // read the accelerometer data scaled 1g=4096 bits
  gx = get_accel(XAXIS);
  gy = get_accel(YAXIS);
  gz = get_accel(ZAXIS);

  // calculate accelerometer magnitude in bits
  accel_mag = (float) sqrt(gx * gx + gy * gy + gz * gz);

  // store the normalised linear acceleration in the circular delay line
  DL[DL_counter++] = (accel_mag - (float) ONE_G) / (float) ONE_G;

  // wrap the delay line index back to zero if necessary
  if (DL_counter == DLSIZE)
    DL_counter = 0;

  // test for new release from hand using 0.3g as criterion
  if ((apogee_counter < 0) && (accel_mag < ZEROP3_G))
  {
```

```
      // sum the delay line to get sampling interval count to top of trajectory
      DLSum = 0.0F;
      for (i = 0; i < DLSIZE; i++)
         DLSum += DL[i];
      apogee_counter = (int) fabs(DLSum);
   }

   // if the apogee counter has reached zero, flash the LED for 0.2S
   if (apogee_counter == 0)
      led_pulse(LED_WHITE, ZEROPTWO_S);

   // decrement the apogee counter if non-negative
   if (apogee_counter >= 0)
      apogee_counter--;
}
```

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number: AN4446
Rev. 0
08/2012