a

# TECHNICAL NOTE

**ONE TECHNOLOGY WAY • P.O. BOX 9106 • NORWOOD, MASSACHUSETTS 02062-9106 • 781/329-4700**

## Using The ADXL202 Duty Cycle Output

**by Harvey Weinberg**

### Introduction

The ADXL202 was designed to be easy, and inexpensive, to interface with a microcontroller. The outputs are Duty Cycle Modulated signals whose duty cycles (ratio of pulse width to period) are proportional to the acceleration in each of the 2 sensitive axes. These outputs may be measured directly with a microcontroller counter. No A/D converter or glue logic is required. The Duty Cycle Modulator (DCM) has a resolution of around 14 bits. Better resolution than the accelerometer itself.

### Calculation of the Acceleration Output

Acceleration experienced by the ADXL202 may be calculated by the following formula:

$$\text{Acceleration (in g)} = \frac{\text{Duty Cycle - Duty Cycle at Zero g}}{\text{Duty Cycle per g}}$$

As outlined in the data sheet, the nominal duty cycle output of the ADXL202 is 50% at zero g and 12.5% duty cycle change per g. Therefore to calculate acceleration from the duty cycle:

$$\text{Acceleration (in g)} = \frac{(T1/T2) - 50\%}{12.5\%}$$

If the zero g duty cycle output of the ADXL202 is other than 50%, and/or the duty cycle changes more or less than 12.5% per g, the acceleration calculation will be inaccurate. In practice the zero g output and the sensitivity of the ADXL202 vary somewhat from device to device (see the data sheet for details). So this formula can only be used for low accuracy measurements. For higher accuracy measurements you must substitute the actual offset and scale values.

In addition, the result of this equation would be a number in the range of ±2 for an ADXL202. In general, it is inconvenient to use real numbers in calculations with a small microcontroller since floating point mathematics would be required to get meaningful results. A preferred method for calculation of acceleration with a small microcontroller using fixed point math is shown below, along with two simple methods of calibration to find the actual offset and scale values.

This application note outlines methods to decode the duty cycle output, conversion from duty cycle to acceleration (or tilt angle), and calibration of the ADXL202. These methods are geared towards use with 8 bit microcontrollers having limited computational capability.

### Decoding the Output

The most direct way to decode the duty cycle output is shown in figure 1. A counter is started at the rising edge of the X output (Ta = 0). The count at the falling edge (Tb) is recorded, and the timer is stopped at the next rising edge of the X output (Tc). This process is then repeated for the Y output (Td, Te, and Tf).

While this technique is very easy to understand, you can only acquire one sample of acceleration from both axis every three cycles (i.e. 3 * T2) since you must wait for the next rising edge of Xout after Tf.

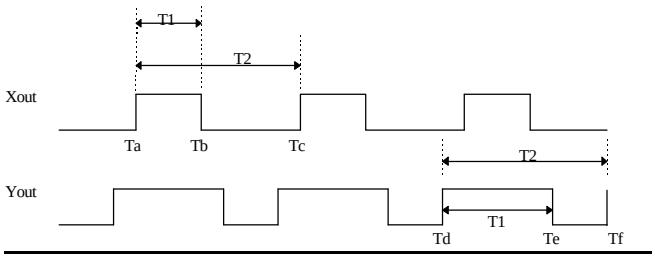


*Figure 1. Basic decode technique for the ADXL202*

Note that while T2 is nominally constant, it does change over temperature and contains some jitter. For systems that do not require resolutions of better than 100 mg, T2 may be measured only once. For more accurate measurements, several T2 measurements should be made and averaged. The average should be updated periodically to account for T2 drift over temperature.

**An Improved PWM Decode Scheme**

Since the Duty Cycle Modulator (DCM) uses the same triangle wave reference for the X and Y channels, the mid-points of the T1 of each period must be coincident. This is illustrated in figure 2.

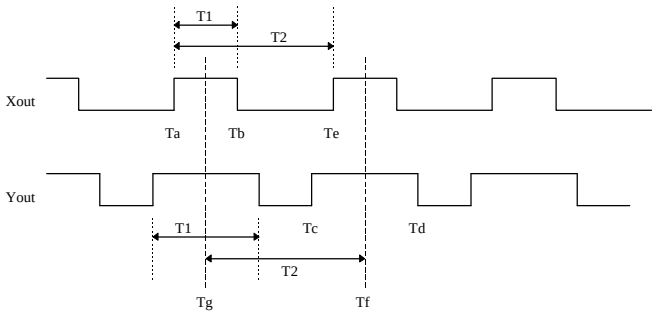


*Figure 2. High speed decoding technique for the ADXL202*

Here you can see that even though the X and Y duty cycle outputs are different, the mid points of T1 are synchronized. Therefore we can use an improved PWM decode technique to speed up the data acquisition time. Figure 2 shows the sequence of events. A counter is started at the rising edge of the X output (Ta = 0). The count at the falling edge of the X output (Tb) is recorded. Then the count at the rising and falling edges of the Y output (Tc and Td) are recorded. A flow chart outlining this method of calibration is shown in figure 3.

By definition;

T1x = Tb - Ta = Tb (if the counter is zero at Ta)

T1y = Td - Tc

T2x = T2y = Te - Ta = Tg - Tf

Since the mid points of the high states of the X and Y Duty Cycle signals are coincident;

T2 = [Td - ((Td - Tc)/2)] - [(Tb - Ta)/2]

T2 = [Td - ((Td - Tc)/2)] - [Tb/2] (if the counter is zero at Ta)

The advantages of this system of decoding are:

1. You can acquire one sample of acceleration from both axis every two T2 cycles.
2. T2 is only calculated once for both the X and Y signals.

**Calibration of the ADXL202**

The easiest way to calibrate the ADXL202 is by using the Earth's gravity as a reference input. If your application does not require high accuracy, you may use a "quick calibration" technique. Here we assume that the sensitivity of the accelerometer is 12.5% duty cycle per g (the typical specification). One digital input may be used to tell the microcontroller that it is in calibrate mode. When in calibrate mode the accelerometer must be level, with the X and Y axis horizontal to the earth so both axis experience 0 g.

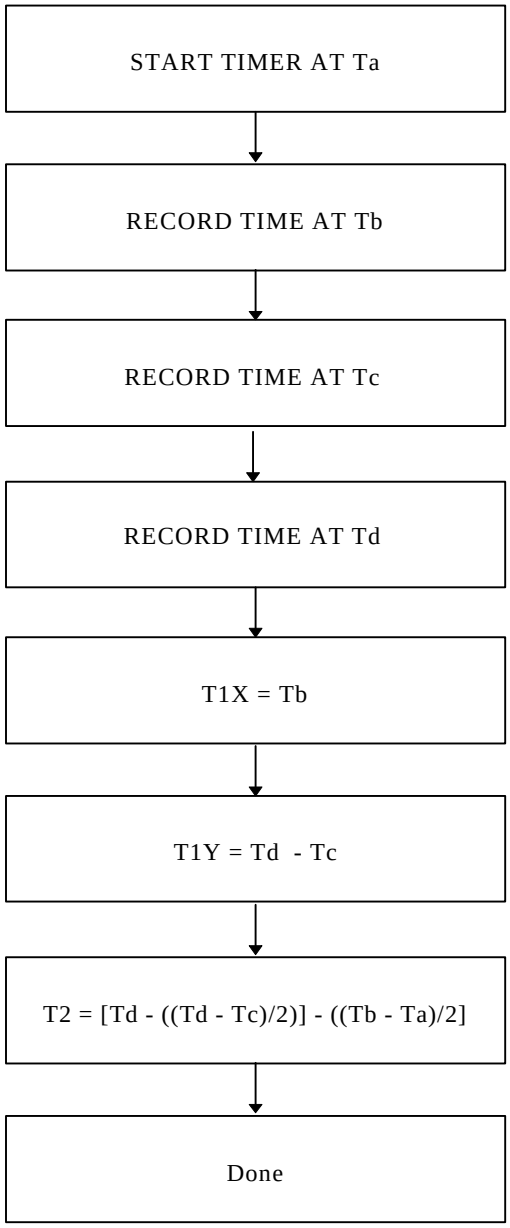


*Figure 3. Flowchart for high speed decoding technique for the ADXL202*

When the microcontroller is told to calibrate, the microcontroller reads the duty cycle output (T1) and period (T2) of the accelerometer from each axis. Several readings of T1 and T2 may be averaged to improve accuracy. These values are stored as calibration constants, and retained for use in calculating the acceleration after calibration. A scale

factor (K) used to scale the acceleration output into an n-bit word should also be calculated. The calibration constants for each axis consists of the following:

1. T2cal = The value (from the microcontrollers counter) of T2 during calibration. T2cal must be stored as the value of T2 does drift over temperature and has jitter.

2. Zcal = The value (from the microcontrollers counter) of T1 during calibration.

3. Bit scale factor. You choose the bit scale factor to determine the resolution (in bits) of the acceleration calculation.

4. K = [4 * (T2cal * bit scale factor) / T2cal] Note that K need only be calculated once for the two axis if you choose the same bit scale factor for both axis.

The bit scale factor will be determined by the size (in bits) of the result you are looking for. For example, if you want a result of ±1 g = ±128 counts (to result in an 8 bit number) the scale factor would be 256. For a result of ±1 g = ± 90 counts (to approximate 1° per count) the scale factor would be 180.

**Calculation of Acceleration From Duty Cycle**
Once the calibration constants are known, only two formulas are required for the calculation of acceleration. They are:

$$Zactual = \frac{Zcal * T2actual}{T2cal}$$

Where T2actual is the current measurement of T2. This formula corrects the zero g value for changes in T2 due to drift or jitter.

$$Acceleration = \frac{K * (T1 - Zactual)}{T2actual}$$

Note that all of the operations are fixed point, and therefore fairly easy and fast for any microcontroller to perform. See table 1 for some performance benchmarks with popular microcontrollers.
Care should be taken in the correct ordering of operations to preserve resolution. When designing your software keep an eye out for ways to order mathematical operations that will not drop significant bits. In general perform addition, multiplication, subtraction, and division in that order. Double and triple precision fixed point arithmetic (16 and 24 bit) will be required in most applications. Algorithms for multi-byte fixed point arithmetic are widely available in applications notes published by microcontroller manufacturers.

**High Accuracy Calibration Method**
Variance in sensitivity (i.e. duty cycle % per g ) from part to part is normal, and will result in a small error if not accounted for. If your application requires high accuracy measurements, the rotational calibration method presented below is preferred.

Simply place the ADXL202 such that the Z axis is horizontal and rotate the ADXL202 over at least 360°. The ADXL202 will then be exposed to ±1 g in both the X and Y axis. The rotation must be done slowly in order to minimize the effects of centrifugal acceleration.

The following pieces of calibration data must be retained for use in calculating the acceleration after calibration:

1. T2cal. That is the value of T2 during the calibration procedure.

2. Zcal. That is the Zero g value of T1 at the time of calibration. It is calculated as follows:

$$Zcal = \frac{T1max - T1min}{2}$$

3. Bit scale factor. You choose the bit scale factor to determine the resolution (in bits) of the acceleration calculation.

4. K. Where K is the scale factor. K may be calculated as follows:

$$K = \frac{T2cal * bit\ scale\ factor}{T1max - T1min}$$

The bit scale factor used here is the same as outlined in the quick calibration technique.

Variance in part to part sensitivity is corrected in the denominator by calculating the actual duty cycle output obtained from exposure to2 g of acceleration (±1 g from gravity).

A flow chart outlining a calibration algorithm is shown in figure 4. One digital input is used to tell the microcontroller that it is in “calibrate” mode.

Calculation of the acceleration is done as presented above.

That is:

$$Acceleration = \frac{K * (T1 - Zactual)}{T2actual}$$

Where**:**

$$Zactual = \frac{Zcal * T2actual}{T2cal}$$

For maximum accuracy several readings of T2 should be averaged to reduce error induced by T2 jitter. The flow chart below does not include this operation. It is up to the user to determine how many averaged samples of T2 are suitable in their application. Generally 4 to 8 averaged samples of T2 should be sufficient for systems with resolutions of 1 mg of greater.

## CALIBRATION ALGORITHM

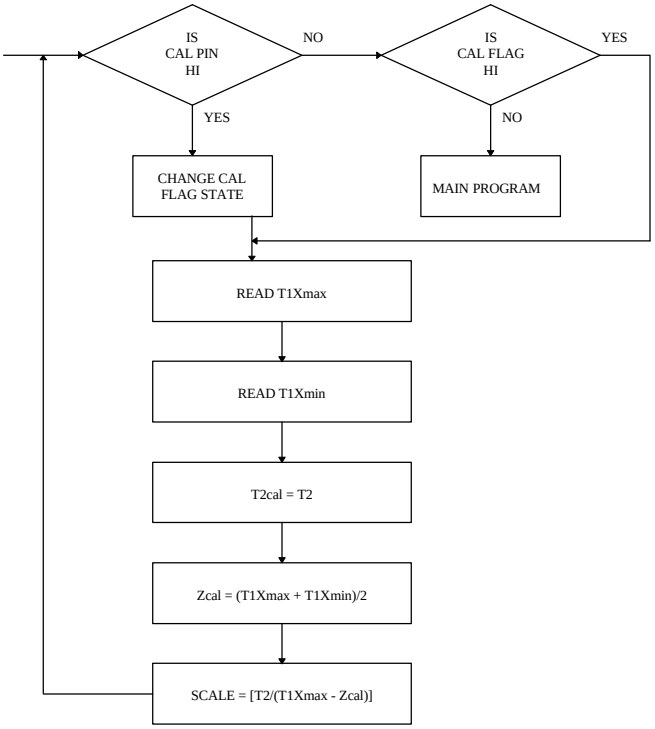


*Figure 4. Flow chart for the ADXL202 calibraiton algorithm*