599 Menlo Drive, Suite 100
Rocklin, California 95765, USA
Office: (916) 624-8333
Fax: (916) 624-8003

General: info@parallaxinc.com
Technical: support@parallaxinc.com
Web Site: www.parallaxinc.com
Educational: www.stampsinclass.com

Memsic 2125 Accelerometer Demo Kit (#28017)

Tilt and Rotation Measurement

Introduction

The Memsic 2125 is a low cost, dual-axis thermal accelerometer capable of measuring dynamic acceleration (vibration) and static acceleration (gravity) with a range of ±2 g. For integration into existing applications, the Memsic 2125 is electrically compatible with other popular accelerometers.

What kind of things can be done with the Memsic 2125? While there are many possibilities, here's a small list of ideas that can be realized with a Memsic 2125 and the Parallax BASIC Stamp:

- Dual-axis tilt sensing for autonomous robotics applications (BOE-Bot, Toddler, SumoBot)
- Single-axis rotational position sensing
- Movement/Lack-of-movement sensing for alarm systems

Packing List

Verify that your Memsic 2125 Demo Kit is complete in accordance with the list below:

- Parallax Memsic 2125 Demo PCB (uses Memsic MXD2125GL)
- Documentation

Note: Demonstration software files may be downloaded from www.parallax.com.

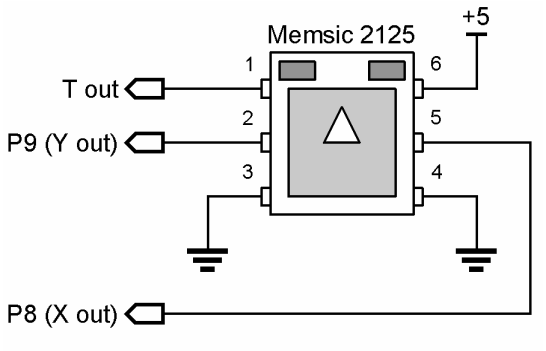
Features

- Measure 0 to ±2 g on either axis; less than 1 mg resolution
- Fully temperature compensated over 0° to 70° C range
- Simple, pulse output of g-force for X and Y axis – direct connection to BASIC Stamp
- Analog output of temperature (TOut pin)
- Low current operation: less than 4 mA at 5 vdc

Connections

Connecting the Memsic 2125 to the BASIC Stamp is a straightforward operation, requiring just two IO pins. If single-axis tilt of less than 60 degrees is your requirement, only one output from the Memsic 2125 need be connected. See Figure 1 for connection details.

Figure 1. Essential Memsic 2125 Connections

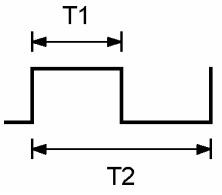


How It Works

Internally, the Memsic 2125 contains a small heater. This heater warms a "bubble" of air within the device. When gravitational forces act on this bubble it moves. This movement is detected by very sensitive thermopiles (temperature sensors) and the onboard electronics convert the bubble position [relative to g-forces] into pulse outputs for the X and Y axis.

The pulse outputs from the Memsic 2125 are set to a 50% duty cycle at 0 g. The duty cycle changes in proportion to acceleration and can be directly measured by the BASIC Stamp. Figure 2 shows the duty cycle output from the Memsic 2125 and the formula for calculating g force.

Figure 2. Memsic 2125 Pulse Output



A(g) = ((T1 / T2) – 0.5) / 12.5%

The T2 duration is calibrated to 10 milliseconds at 25° C (room temperature). Knowing this, we can convert the formula to the following BASIC Stamp routine:

```
Read_X_Force:
  PULSIN Xin, HiPulse, xRaw
  xGForce = ((xRaw / 5) - 500) * 8
  RETURN
```

The T1 duration (Memsic output) is placed in xRaw, then divided by five because the BASIC Stamp PULSOUT function returns two-microsecond units. Since the result of this division is in microseconds, we multiply 0.5 by 1000 to get 500. Finally, one divided 0.125 (12.5%) is eight, hence the final multiplication. The result is a signed value representing g-force in milli-g's (1/1000th g).

Experiments

Experiment 1: Dual-Axis Tilt Measurement

This experiment reads both axis values and displays the results in the DEBUG window. Calculations for g-force measurement and conversion to tilt were taken directly from Memsic documentation. Since the BASIC Stamp does not have an Arcsine function, it must be derived. Code for Arccosine and Arcsine are provided courtesy Tracy Allen, Ph.D.

Note: This program is written using PBASIC 2.5 syntax and requires the Version 2.0 compiler or later to run.

```
' =====================================================================

'
'   File...... MEMSIC2125-Dual.BS2

'   Purpose... Memsic 2125 Accelerometer Dual-Axis Demo
'   Author.... Parallax
'   E-mail.... support@parallax.com
'   Started...
'   Updated... 15 JAN 2003
'
'   {$STAMP BS2}
'   {$PBASIC 2.5}
'
' =====================================================================


' ---------------------------------------------------------------------
' Program Description
' ---------------------------------------------------------------------
'
' Read the pulse outputs from a Memsic 2125 accelerometer and converts to
' G-force and tilt angle.
'
' g = ((t1 / 10 ms) - 0.5) / 12.5%
'
' Tilt = ARCSIN(g)
'
' Refer to Memsic documentation (AN-00MX-007.PDF) for details on g-to-tilt
' conversion and considerations.
'
' www.memsic.com


' ---------------------------------------------------------------------
' I/O Definitions
' ---------------------------------------------------------------------

Xin             PIN     8                       ' X input from Memsic 2125
Yin             PIN     9                       ' Y input from Memsic 2125


' ---------------------------------------------------------------------
' Constants
' ---------------------------------------------------------------------
```

```
HiPulse         CON     1                       ' measure high-going pulse
LoPulse         CON     0

DegSym          CON     176                     ' degrees symbol


' -----------------------------------------------------------------------------
' Variables
' -----------------------------------------------------------------------------

xRaw            VAR     Word                    ' pulse from Memsic 2125
xmG             VAR     Word                    ' g force (1000ths)
xTilt           VAR     Word                    ' tilt angle

yRaw            VAR     Word
ymG             VAR     Word
yTilt           VAR     Word

disp            VAR     Byte                    ' displacement (0.0 - 0.99)
angle           VAR     Byte                    ' tilt angle


' -----------------------------------------------------------------------------
' Initialization
' -----------------------------------------------------------------------------

Setup:
  PAUSE 250                                     ' let DEBUG window open
  DEBUG "Memsic 2125 Accelerometer", CR
  DEBUG "--------------------------"


' -----------------------------------------------------------------------------
' Program Code
' -----------------------------------------------------------------------------

Main:
  DO
    GOSUB Read_Tilt                             ' reads G-force and Tilt

    ' display results

    DEBUG CRSRXY, 0, 3
    DEBUG "X Input...  ",
          DEC (xRaw / 500), ".", DEC3 xRaw, " ms",
          CLREOL, CR,
          "G Force... ", (xmG.Bit15 * 13 + " "),
          DEC (ABS xmG / 1000), ".", DEC3 (ABS xmG), " g",
          CLREOL, CR,
          "X Tilt.... ", (xTilt.Bit15 * 13 + " "),
          DEC ABS xTilt, DegSym, CLREOL

    DEBUG CRSRXY, 0, 7
    DEBUG "Y Input...  ",
          DEC (yRaw / 500), ".", DEC3 yRaw, " ms",
          CLREOL, CR,
          "G Force... ", (ymG.Bit15 * 13 + " "),
          DEC (ABS ymG / 1000), ".", DEC3 (ABS ymG), " g",
          CLREOL, CR,
          "Y Tilt.... ", (yTilt.Bit15 * 13 + " "),
```

```
          DEC ABS yTilt, DegSym, CLREOL

    PAUSE 200                                   ' update about 5x/second
  LOOP
  END


' -----------------------------------------------------------------------------
' Subroutines
' -----------------------------------------------------------------------------

Read_G_Force:
  PULSIN Xin, HiPulse, xRaw                     ' read pulse output
  xmG = ((xRaw / 5) - 500) * 8                  ' convert to 1/1000 g
  PULSIN Yin, HiPulse, yRaw
  ymG = ((yRaw / 5) - 500) * 8
  RETURN


Read_Tilt:
  GOSUB Read_G_Force

  ' restrict displacement to unit circle (0.0 - 1.0)

  disp = ABS xmG / 10 MAX 100                   ' x displacement
  GOSUB Arcsine
  xTilt = angle * (-2 * xmG.bit15 + 1)          ' fix sign
  disp = ABS ymG / 10 MAX 100                   ' y displacement
  GOSUB Arcsine
  yTilt = angle * (-2 * ymG.bit15 + 1)          ' fix sign
  RETURN


' Trig routines courtesy Tracy Allen, PhD. (www.emesystems.com)

Arccosine:
  disp = disp */ 983 / 3                        ' normalize input to 127
  angle = 63 - (disp / 2)                       ' approximate angle
  DO                                            ' find angle
    IF (COS angle <= disp) THEN EXIT
    angle = angle + 1
  LOOP
  angle = angle */ 360                          ' convert brads to degrees
  RETURN


Arcsine:
  GOSUB Arccosine
  angle = 90 - angle
  RETURN
```

Experiment 2: Rotational Position Sensing

If the Memsic 2125 is tilted up on its edge (X axis), the X and Y outputs can be combined to measure rotational position through 360 degrees. Output from this program is in both Brads (binary radians, 0 to 255, the BASIC Stamp's unit of angular measurement) and degrees (0 to 359).

For this code to work, the Memsic 2125 PCB must be positioned such that the sensor is perpendicular to the ground.

Note: This program is written using PBASIC 2.5 syntax and requires the Version 2.0 compiler or later to run.

```
' =============================================================================
'
'   File..... MEMSIC2125-Rotation.BS2
'   Purpose... Memsic 2125 Accelerometer Rotational Angle Measurement
'   Author.... Parallax
'   E-mail.... support@parallax.com
'   Started...
'   Updated... 15 JAN 2003
'
'   {$STAMP BS2}
'   {$PBASIC 2.5}
'
' =============================================================================


' -----------------------------------------------------------------------------
' Program Description
' -----------------------------------------------------------------------------
'
' Read the pulse outputs from a Memsic 2125 accelerometer and combine to
' calculation rotational position.
'
' Refer to Memsic documentation (AN-00MX-007.PDF) for details on angle
' conversion and considerations.
'
' www.memsic.com


' -----------------------------------------------------------------------------
' I/O Definitions
' -----------------------------------------------------------------------------

Xin             PIN     8                       ' X input from Memsic 2125
Yin             PIN     9                       ' Y input from Memsic 2125


' -----------------------------------------------------------------------------
' Constants
' -----------------------------------------------------------------------------

HiPulse         CON     1                       ' measure high-going pulse
LoPulse         CON     0

DegSym          CON     176                     ' degrees symbol
```

```
' -----------------------------------------------------------------------------
' Variables
' -----------------------------------------------------------------------------

pulse           VAR     Word                    ' pulse input
xmG             VAR     Word                    ' g force (1000ths)
ymG             VAR     Word
brads           VAR     Word                    ' binary radians
degrees         VAR     Word


' -----------------------------------------------------------------------------
' Initialization
' -----------------------------------------------------------------------------

Setup:
  PAUSE 250                                     ' let DEBUG window open
  DEBUG "Memsic 2125 Rotation", CR
  DEBUG "--------------------"


' -----------------------------------------------------------------------------
' Program Code
' -----------------------------------------------------------------------------

Main:
  DO
    GOSUB Read_G_Force                          ' read X and Y

    brads = (xmG / 8) ATN (ymG / 8)             ' calculate angle
    degrees = brads */ 360                      ' convert to degrees

    DEBUG CRSRXY, 0, 3
    DEBUG "Axis   A(g)", CR,
          "X     ", (xmG.Bit15 * 13 + " "),
          DEC (ABS xmG / 1000), ".", DEC3 (ABS xmG), " g", CR,
          "Y     ", (ymG.Bit15 * 13 + " "),
          DEC (ABS ymG / 1000), ".", DEC3 (ABS ymG), " g", CR, CR,
          "Tilt = ", DEC3 brads, " Brads", CR,
          "       ", DEC3 degrees, " Degrees"

    PAUSE 200                                   ' update about 5x/second
  LOOP
  END


' -----------------------------------------------------------------------------
' Subroutines
' -----------------------------------------------------------------------------

Read_G_Force:
  PULSIN Xin, HiPulse, pulse                    ' read pulse output
  xmG = ((pulse / 5) - 500) * 8                 ' convert to 1/1000 g
  PULSIN Yin, HiPulse, pulse
  ymG = ((pulse / 5) - 500) * 8
  RETURN
```

Experiment 3: Motion Detector

This experiment uses the Memsic 2125 as a movement or vibration detector. The program starts by reading the initial state of the sensor and storing these readings as calibration values. By doing this, the starting position of the sensor is nullified. The main loop of the program reads the sensor and compares the current outputs to the calibration values. If the output from either axis is greater than its calibration value the motion timer is incremented. If both fall below the thresholds motion timer is cleared. If the motion timer exceeds its threshold, the alarm will be turned on and will stay on until the BASIC Stamp is reset.

You can adjust the sensitivity (to motion/vibration) of the program by changing the **XLimit** and **YLimit** constants, as well as the **SampleDelay** constant (should be 100 ms or greater). The **AlarmLevel** constant determines how long motion/vibration must be present before triggering the alarm.

Note: This program is written using PBASIC 2.5 syntax and requires the Version 2.0 compiler or later to run.

```
' =============================================================================
'
'   File...... MEMSIC2125-Motion.BS2
'   Purpose... Detects continuous motion for given period
'   Author.... Parallax (based on code by A. Chaturvedi of Memsic)
'   E-mail.... support@parallax.com
'   Started...
'   Updated... 15 JAN 2003
'
'   {$STAMP BS2}
'   {$PBASIC 2.5}
'
' =============================================================================


' -----[ Program Description ]---------------------------------------------
'
' Monitors X and Y inputs from Memsic 2125 and will trigger alarm if
' continuous motion is detected beyond the threshold period.


' -----[ I/O Definitions ]-------------------------------------------------

Xin             PIN     8                       ' X pulse input
Yin             PIN     9                       ' Y pulse input
ResetLED        PIN     10                      ' reset LED
AlarmLED        PIN     11                      ' alarm LED


' -----[ Constants ]------------------------------------------------------

HiPulse         CON     1                       ' measure high-going pulse
LoPulse         CON     0

SampleDelay     CON     500                     ' 0.5 sec
AlarmLevel      CON     5                       ' 5 x SampleDelay

XLimit          CON     5                       ' x motion max
YLimit          CON     5                       ' y motion max
```

```
' -----[ Variables ]------------------------------------------------------

xCal            VAR     Word                    ' x calibration value
yCal            VAR     Word                    ' y calibration value
xMove           VAR     Word                    ' x sample
yMove           VAR     Word                    ' y sample
xDiff           VAR     Word                    ' x axis difference
yDiff           VAR     Word                    ' y axis difference

moTimer         VAR     Word                    ' motion timer


' -----[ Initialization ]--------------------------------------------------

Initialize:
  LOW AlarmLED                                  ' alarm off
  moTimer = 0                                   ' clear motion timer

Read_Cal_Values:
  PULSIN Xin, HiPulse, xCal                     ' read calibration values
  PULSIN Yin, HiPulse, yCal
  xCal = xCal / 10                              ' filter for noise & temp
  yCal = yCal / 10

  HIGH ResetLED                                 ' show reset complete
  PAUSE 1000
  LOW ResetLED


' -----[ Program Code ]-----------------------------------------------------

Main:
  DO
    GOSUB Get_Data                              ' read inputs
    xDiff = ABS (xMove - xCal)                  ' check for motion
    yDiff = ABS (yMove - yCal)

    IF (xDiff > XLimit) OR (yDiff > YLimit) THEN
      moTimer = moTimer + 1                     ' update motion timer
      IF (moTimer > AlarmLevel) THEN Alarm_On
    ELSE
      moTimer = 0                               ' clear motion timer
    ENDIF
  LOOP
  END


' -----[ Subroutines ]------------------------------------------------------

' Sample and filter inputs

Get_Data:
  PULSIN Xin, HiPulse, xMove                    ' take first reading
  PULSIN Yin, HiPulse, yMove
  xMove = xMove / 10                            ' filter for noise & temp
  yMove = yMove / 10
  PAUSE SampleDelay
  RETURN
```

```
' Blink Alarm LED
' -- will run until BASIC Stamp is reset

Alarm_On:
  DO
    TOGGLE AlarmLED                              ' blink alarm LED
    PAUSE 250
  LOOP                                           ' loop until reset
```

Application Idea

Using the tilt code from Experiment 1, you can create a 3D joystick by mounting the Memsic 2125 and a pushbutton in a small, spherical enclosure (like a tennis ball). With just three pins you can measure tilt of each axis and the status of the switch. This would make an interesting, intelligent "leash" for a Parallax BOE-Bot.

Using TOut

Since the Memsic 2125 is a thermal device, the temperature is available from the TOut pin and can be measured using an external analog to digital converter (i.e., LTC1298).

Details:

- Output calibrated to 1.25 volts @ 25.0° C
- Output change: 5 millivolts per degree C