```
'*****************************************************************
'*  Name    : Compass.BAS                                        *
'*  Author  : Ed Cannady edcannady@earthlink.net                 *
'*  Notice  : Copyright (c) 2003                                 *
'*                                   All Rights Reserved         *
'*  Date    : 9/29/2003                                          *
'*  Version : 1.0                                                *
'*  Notes   : For a 16F628 / LTC 1258                            *
'*          :                                                    *
'*****************************************************************
' Define LOADER_USED to allow use of the boot loader.
' This will not affect normal program operation.

Define      LOADER_USED 1

Include "Modedefs.bas"

' ** Set Xtal Value in MHz **

Define      OSC   20    ' Set the Crystal Frequency to 20 MHz

' ** Declared Pins Used **

CK    VAR         PortA.0     ' Clock from Pin 17 to Pin 7 of the 1298
DO    VAR         PortA.1     ' Data Out from Pin 18 to Pin 6 of the 1298
DI    VAR         PortB.6     ' Data In from Pin 12 to Pin 5 of the 1298
CS    VAR         PortB.7     ' Chip Select From Pin 13 to Pin 1 of the 1298

' Not Used        PortA.3     Pin 2
' Not Used        PortA.4     Pin 3
' Not Used        PortB.0     Pin 6
' Not Used        PortA.2     Pin 1

'                 PortB.1     Pin 7 Debug RX
'                 PortB.2     Pin 8 Debug TX
'                 PortB.3     Pin 9 Display Clk
'                 PortB.4     Pin 10 Display Dta
'                 PortB.5     Pin 11 Display Load

' ** Allocate variables **

' The highest number we can use is in a PIC is 65,535. Therefore, we
' Must use small numbers for our math.

' Upper cross point is 2.8814 volts or A/D output of 2361
' Lower cross point is 2.10 volts or A/D output of 1721

' 2.88 - 2.10 is .78 which is the voltage change over 90 degrees
' 90 degrees divided by .78 is 115.38462 which is our scale factor

' 5.000 volts divided by 4096 (the 12-bit reading of 5.000 volts
' Tells us that .0012207 volts is equal to 1 bit from the A/D

' 115.38462 times .0012207 gives .14085 to be used to multiply time the
' formula results to get the degrees! Times 10000 = 14085. Divided by 4
' so we have a small number to use or 3521.25 divided by 1000 or 35.
```

```
addr        VAR   BYTE        ' Channel address / mode
Result      VAR   WORD        ' Reading from the A/D
B_Curve     VAR   WORD        ' The B curve reading from the A/D
A_Curve     VAR   WORD        ' The A curve reading from the A/D
CompDisp    VAR   WORD        ' Our final value to display
RawCompass  VAR   WORD        ' Factored A/D results
Compass     VAR   WORD        ' Variable to reduce the calculations
CrossZero   VAR   WORD        ' We are Crossing 360 degrees
ZeroCross   VAR   WORD        ' We have Crossed 360 degrees
UpLimit     VAR   WORD        ' Upper crossover point
LowLimit    VAR   WORD        ' Lower crossover point
Scale       VAR   WORD        ' Spread times 10 divides by 4
Spread      VAR   WORD        ' 90 degrees divided by the Upper - Lower Limits


UpLimit  = 2361               ' A starting value before auto calibration
LowLimit = 1721               ' A starting value before auto calibration
Scale    = 35                 ' A starting value before auto calibration

cmcon=7                       ' Set all the PIC inputs to be digital

High CS                       ' A/D Chip select inactive

' ** Pins Used to Display Results**

Clk         Var   PortB.3     ' Data is clocked on rising edge of this pin
Dta         Var   PortB.4     ' Bits are shifted out of this pin
Load        Var   PortB.5     ' Transfers data to LEDs when Pulsed

' ** Declare Constants **
Decode_Reg  Con   9     ' Decode register, a 1 turns on BCD decoding for each digit.
Lum_Reg     Con   10    ' Intensity register.
Scan_Reg    Con   11    ' Scan-limit register.
Switch_Reg  Con   12    ' On/Off Register.
Test_Reg    Con   15    ' Test mode register (all digits on, 100% bright)

' ** Declare Variables **
Max_Disp    Var   Word  ' 16-bit value to be displayed by the MAX7219
Max_Dp      Var   Byte  ' Digit number to place Decimal point (0-4)
Register    Var   Byte  ' Pointer to the Internal Registers of the MAX7219
R_Val       Var   Byte  ' Data placed in Each Register
Digit       Var   Byte  ' Position of individual numbers within MAX_Disp (0-3)
Position    Var   Byte  ' Position of each LED display (1-4)


' ** INITIALIZE THE MAX7219 **
' Each register address is sent along with its setting data.
' Because the MAX7219 expects to see a packet of 16 bits, then the LOAD pin is pulsed
' Set the scan limit to 3 (4 digits, numbered 0-3)
' Set the Brightness to 5
' BCD decoding to the lower 4 digits
' Switch the display on.
' Turn Off test mode

Register=Scan_Reg ' Point to the Scan Register
```

```
R_Val=3                 ' send 3, (Four LED Displays 0-3)
Gosub Transfer          ' Transfer this 16-bit Word to the MAX7219

Register=Lum_Reg  ' Point to the Luminance Register
R_Val=5                 ' Send 5, (Value for Brightness)
Gosub Transfer          ' Transfer this 16-bit Word to the MAX7219

Register=Decode_Reg     ' Point to BCD Decode Register
R_Val=%00011111         ' Decode the first 5 digits
Gosub Transfer          ' Transfer this 16-bit Word to the MAX7219

Register=Switch_Reg     ' Point to the Switch Register
R_Val=1                 ' Set to One, (switches the display ON)
Gosub Transfer          ' Transfer this 16-bit Word to the MAX7219

Register=Test_Reg ' Point to the Test Register
R_Val=0                 ' Reset to Zero, (turns off Test mode)
Gosub Transfer          ' Transfer this 16-bit Word to the MAX7219

' ***** MAIN PROGRAM *****

Goto  mainloop          ' Skip subroutines


' ** Subroutine to read A/D converter

getad:

CS = 0                                  ' A/D Chip select active

' ** Send address / mode - Start bit, 3 bit addr, null bit].**

Shiftout DI, CK, MSBFIRST, [1\1, addr\3, 0\1]

Shiftin DO, CK, MSBPRE, [result\12]       ' Get 12-bit result

CS = 1                                  ' A/D Chip select inactive

Return

' ** Subroutine to get B_Curve value (channel 0)
getB_Curve:
      addr = $05                    ' Single ended, channel 0, MSBF high
      Gosub getad
      B_Curve = result              ' Our reading for the B lead
      Return


' Subroutine to get A_Curve value (channel 1)
getA_Curve:
      addr = $07              ' Single ended, channel 1, MSBF high
      Gosub getad
      A_Curve = result        ' Our reading for the A lead
      Return


Mainloop:
```

```
RawCompass = 0          ' Reset our values
Compass = 0             ' Reset our values
CompDisp = 0            ' Reset our values

Gosub       getB_Curve  ' Get the B_Curve value
Gosub       getA_Curve  ' Get the A_Curve Value

IF A_Curve = B_Curve then GOSUB Calibrate

If A_Curve >= UpLimit  THEN GOTO SectionB  'Reading is in the second section
If B_Curve >= UpLimit  THEN GOTO SectionA  'Reading is in the first  section
If A_Curve <= LowLimit THEN GOTO SectionC  'Reading is in the third  section
If B_Curve <= LowLimit THEN GOTO SectionD  'Reading is in the fourth section


SectionA: ' 0-90 degree reading

RawCompass = (UpLimit - A_Curve) * Scale
Compass = ((RawCompass/10) * 4) /10
CompDisp = Compass            ' The Compass display is the value of compass
MAX_DISP = CompDisp
GOSUB Display                 ' Display the reading
Pause 100
GoTO Mainloop                 ' Do it indefinitely


SectionB: ' 270-360 degreee reading

RawCompass = (B_Curve - LowLimit) * Scale
Compass = ((RawCompass/10) * 4) /10
CrossZero = Compass + 2700
if CrossZero >= 3600 then SectionBA ' We have really crossed Zero Degrees
MAX_DISP = CrossZero
GOSUB Display                 ' Display the reading
Pause 100
GoTO Mainloop                 ' Do it indefinitely


SectionBA:

ZeroCross = CrossZero -3600
MAX_DISP = ZeroCross
GOSUB Display                 ' Display the reading
Pause 100
GoTO Mainloop                 ' Do it indefinitely


SectionC: ' 90-180 degree reading

RawCompass = 0
RawCompass = (UpLimit - B_Curve) * Scale
Compass = ((RawCompass/10) * 4) /10
CompDisp = Compass + 900
MAX_DISP = CompDisp
GOSUB Display                 ' Display the reading
Pause 100
```

```
GoTO Mainloop                 ' Do it indefinitely


SectionD: ' 180-270 degreee reading

RawCompass = (A_Curve - LowLimit) * Scale
Compass = ((RawCompass/10) * 4) /10
CompDisp = Compass + 1800
MAX_DISP = CompDisp
GOSUB Display                 ' Display the reading
Pause 100
GoTO Mainloop                 ' Do it indefinitely

Calibrate:

IF A_Curve > 2048 THEN UpLimit  = A_Curve ' Establish the actual upper limit
If A_Curve < 2048 THEN LowLimit = A_Curve ' Establish the actual lower limit
Spread = 9000 / (UpLimit - LowLimit)            ' 90.00 degrees divided by the spread
Scale = (Spread * 10) /4                  ' We now have our scaling factor
Return


' ** Subroutines **
' Display the Value held in the Variable "MAX_DISP" on the four LED's
' The value held in "MAX_DP" places the decimal point on that LED (0-3)
' Sending the value 15 blanks the display, this allows Zero suppression
' By setting bit-7 of the value sent to the individual LED displays, the decimal
point
' Is illuminated
Display:
      Digit=0                             ' Start at Digit 0 of Max_Disp Variable
      For Position=4 to 1 step -1         ' Start at Farthest Right of Display

      Register=Position                   ' Place Position into Register
      R_Val=Max_Disp Dig Digit            ' Extract the individual numbers from
Max_Disp
If Max_Disp<10 and Position=3 then R_Val=15          ' Zero Suppression for the
First digit
If Max_Disp<100 and Position=2 then R_Val=15    ' Zero Suppression for the second
digit
      If Max_Disp<1000 and Position=1 then R_Val=15   ' Zero Suppression for the
Third digit
      If Max_Disp<10000 and Position=0 then R_Val=15  ' Zero Suppression for the
Forth digit
If Digit=Max_Dp then R_Val.7=1      ' Place the decimal point, held in Max_DP
      Max_Dp=1
Gosub Transfer                      ' Transfer the 16-bit Word to the MAX7219
      If Digit>=3 then Digit=0            ' We only need the first four digits
      Digit=Digit+1                       ' Point to next Digit within Max_Disp
      Next Position                       ' Close the Loop
      Return                              ' Exit from subroutine

' Send a 16-bit word to the MAX7219
Transfer:
Shiftout Dta,Clk,msbfirst,[Register,R_Val] ' Shift Out the Register first, then the
data
High Load                     ' The data is now acted upon
```

```
@     Nop
@     Nop                    ' A small delay to ensure correct clocking times
Low Load                     ' Disable the MAX7219
Return                       ' Exit from Subroutine
```