```
'--------Title--------

' File......Braitenberg1.pbp
' Started....2/28/08
' Microcontroller used:  Microchip Technology PIC16F88
'                        microchip.com
' PicBasic Pro Code: micro-Engineering Labs, Inc.
'                    melabs.com

'--------Program Desciption--------

' The program uses two of the analog-to-digital
' converters,(AN0 & AN1), to create a
' photophilic, light seeking, robot.
' Photo- indicates light and -philic indicates
' attraction to.

'------------Comments--------------

' Program named after Valentino Braitenberg
' who wrote a book, titled Vehicles, in 1984
' which describes simple robot behaviors
' that appear life-like.

'---------PIC Connections---------

'       16F88 Pin              Wiring
'       ---------           ----------
'         RA0               Center lead CdS voltage divider 1
'         RA1               Center lead CdS voltage divider 2
'         RA4               LCD Register Select(RS)
'         RB0               To base of NPN controlling Motor 1
'         RB1               To base of NPN controlling Motor 2
'         RB3               LCD Enable(E)
'         RB4               LCD (DB4)
'         RB5               LCD (DB5)
'         RB6               LCD (DB6)
'         RB7               LCD (DB7)
'       See schematic for the other usual PIC connections

'---------LCD Connections---------

'        LCD Pin              Wiring
'       ---------           ----------
'          1               Ground(Vss)
'          2               + 5v(Vdd)
'          3               Center of 20K Pot(Contrast)
'          4               RA4(Register Select,RS)
'          5               Ground(Read/Write,R/W)
'          6               RB3(Enable)
'          7               No Connection(DB0)
'          8               No Connection(DB1)
'          9               No Connection(DB2)
'         10               No Connection(DB3)
'         11               RB4(DB4)
```

```
'         12                RB5(DB5)
'         13                RB6(DB6)
'         14                RB7(DB7)

'---------Revision History--------

'---------Constants/Defines-------

' To free up AN0 and AN1 (Pins RA0 and RA1) for
' an analog input, the default LCD data lines, DB4-DB7,
' function must be removed from RA0 - RA3.
' They are relocated to PORTB.4 - PORTB.7 (RB4-RB7)
' using the LCD DEFINE statements below.  All other
' default LCD pins and functions are left unchanged.

    DEFINE LCD_DREG  PORTB       'Sets PORTB as LCD data port
    DEFINE LCD_DBIT  4           'Start data connections to bit 4

    DEFINE ADC_BITS  10          'Sets the number of bits in
                                 'the result to 10

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

     left_cds     VAR    WORD       'Word for voltage divider 1 value
     right_cds    VAR    WORD       'Word for voltage divider 2 value

' As more light enters the CdS photoresistor, the values of left_cds
' and right_cds reduce, an inverse relationship.

     left_motor   VAR    PORTB.0    'Defines PORTB.0 name as left_motor
     right_motor  VAR    PORTB.1    'Defines PORTB.1 name as right_motor

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

    ANSEL = %00000011  'Leaves AN0 & AN1 in analog mode, but

                       'changes other analog bits to digital.
                       'See table below.

'    Analog Bit    Analog or Digital      PIC16F88 Pin
'   ------------  ------------------     --------------
'       AN0            Analog                 RA0
'       AN1            Analog                 RA1
'       AN2            Digital                RA2
'       AN3            Digital                RA3
'       AN4            Digital                RA4
'       AN5            Digital                RB6
'       AN6            Digital                RB7

    ADCON1 = %10000000           'Right justifies 10-bit value of x
                                 'in 16-bit WORD.  Adds "0" in the
                                 '6 Most Significant bits of the Word,
                                 'shifting the 10-bit value of x to
                                 'the right.
```

```
    OSCCON = $60                'Sets the internal oscillator in the
                                '16F88 to 4 MHz

'------------Main Code------------

    PAUSE 1000                  'Pause to allow LCD to setup

start:

    ADCIN 0, left_cds           'Read analog voltage on AN0 and
                                'convert to 10-bit digital value
                                'and store as left_cds.

    ADCIN 1, right_cds          'Read analog voltage on AN0 and
                                'convert to 10-bit digital value
                                'and store as right_cds.

    LCDOUT $FE,1,"Left CdS = ", DEC left_cds
                                'Clears LCD screen, displays
                                '"Left CdS = " and the 10-bit
                                'value of left_cds
    LCDOUT $FE,$C0,"Right CdS = ", DEC right_cds
                                'LCD jumps to beginning of second
                                'line and displays "Right CdS = "
                                'and the 10-bit value of right_cds


    IF (left_cds < 400) OR (right_cds < 400) THEN GOTO shutdown
                                'When robot approaches light, it will
                                'stop when either left or right CdS
                                'value is less than 400.  This is
                                'the "anti-moth" feature.

    IF left_cds >= right_cds THEN
                                'If the left_cds value is greater than or
                                'equal to the right_cds value(the light
                                'is brighter on right CdS), the robot
                                'turns to the right.
    GOSUB turnright             'Program jumps to subroutine turnright

    ELSE                        'If the left_cds value is less than
                                'the right_cds value(the light
                                'is brighter on the left CdS), the robot
                                'turns to the left.
    GOSUB turnleft              'Program jumps to subroutine turnleft

    ENDIF

    GOTO start                   'Go to loop label

shutdown:
    LOW left_motor              'Turn off left motor (RB0)
    LOW right_motor             'Turn off right motor (RB1)
    PAUSE 10
    END
```

```
turnright:

    HIGH left_motor             'Turn on left motor (RB0)
    LOW right_motor             'Turn off right motor (RB1)
    PAUSE 100                   'Pause 100 mS
    RETURN

turnleft:

    LOW left_motor              'Turn off left motor (RB0)
    HIGH right_motor            'Turn on right motor (RB1)
    PAUSE 100                   'Pause 100 mS
    RETURN
```