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

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

Adding an LCD to the SumoBot® Robot

Introduction

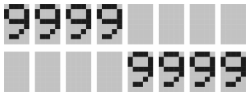
As mini sumo robotics is a competitive endeavor it is helpful to fine-tune competition code for the environment where the contest is held. What is not always possible, however, is to keep the SumoBot tethered to a PC to display sensor values – a local display would be useful.

One can add a local display to the SumoBot by using the Parallax LCD AppMod. It plugs into the SumoBot PCB AppMod header and will not interfere with sensors when testing. When sensor testing and program fine-tuning is complete, the LCD AppMod can be removed so as not to interfere with the weight or center-of-gravity of the SumoBot.

The LCD code used in the following programs is that same as used in many BASIC Stamp® 2 applications – with one very small change: the LCD data bus is disconnected (bus pins are made inputs) at the end of any LCD write to prevent activating or interfering with sensors that share pins with the LCD bus.

Program Code: Reading QTI Line Sensors

This program reads and displays the QTI line sensor values on the LCD. As the LCD is only eight characters wide and the QTI values may require up four characters, the display of QTI values is configured as shown below:




```
' =============================================================================
'
'   File..... SumoBot_3.1_Line_Sensor_LCD.BS2
'   Purpose... Line Sensor Test
'   Author.... Parallax, Inc.
'              (Copyright (c) 2002 - 2004, All Rights Reserved)
'   E-mail.... support@parallax.com
'   Started...
'   Updated... 03 JUN 2004
'
'   {$STAMP BS2}
'   {$PBASIC 2.5}
'
' =============================================================================
```

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

LLinePwr        PIN     10                      ' left line sensor power
LLineIn         PIN     9                       ' left line sensor input
RLinePwr        PIN     7                       ' right line sensor power
RLineIn         PIN     8                       ' right line sensor input

E               PIN     1                       ' LCD Enable (1 = enabled)
RW              PIN     2                       ' Read/Write\
RS              PIN     3                       ' Reg Select (1 = char)
LcdDirs         VAR     DIRB                    ' dirs for I/O redirection
LcdBusOut       VAR     OUTB


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

LcdCls          CON     $01                     ' clear the LCD
LcdHome         CON     $02                     ' move cursor home
LcdCrsrL        CON     $10                     ' move cursor left
LcdCrsrR        CON     $14                     ' move cursor right
LcdDispL        CON     $18                     ' shift chars left
LcdDispR        CON     $1C                     ' shift chars right

LcdDDRam        CON     $80                     ' Display Data RAM control
LcdCGRam        CON     $40                     ' Character Generator RAM
LcdLine1        CON     $80                     ' DDRAM address of line 1
LcdLine2        CON     $C0                     ' DDRAM address of line 2


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

lLine           VAR     Word                    ' left sensor raw reading
rLine           VAR     Word                    ' right sensor raw reading

idx             VAR     Nib                     ' loop counter
char            VAR     Byte                    ' character for LCD
value           VAR     Word                    ' value for LCD printing


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

Initialize:
  NAP 5                                         ' let LCD self-initialize
  DIRL = %11111110                              ' setup pins for LCD

LCD_Init:
  LcdBusOut = %0011                             ' 8-bit mode
  PULSOUT E, 3 : PAUSE 5
  PULSOUT E, 3 : PAUSE 0
  PULSOUT E, 3 : PAUSE 0
  LcdBusOut = %0010                             ' 4-bit mode
  PULSOUT E, 3
  char = %00101000                              ' 2-line mode
  GOSUB LCD_Command
  char = %00001100                              ' on, no crsr, no blink
  GOSUB LCD_Command
  char = %00000110                              ' inc crsr, no disp shift
```

```
  GOSUB LCD_Command
  char = LcdCls
  GOSUB LCD_Command


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

Main:
  DO
    GOSUB Read_Left                             ' read sensors
    GOSUB Read_Right

    char = LcdLine1                             ' left on left, line 1
    GOSUB LCD_Command
    value = lLine
    GOSUB LCD_DEC4

    char = LcdLine2 + 4                         ' right on right, line 2
    GOSUB LCD_Command
    value = rLine
    GOSUB LCD_DEC4

    PAUSE 100
  LOOP

  END


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

Read_Left:
  HIGH LLinePwr                                 ' activate sensor
  HIGH LLineIn                                  ' discharge QTI cap
  PAUSE 1
  RCTIME LLineIn, 1, lLine                      ' read sensor value
  LOW LLinePwr                                  ' deactivate sensor
  RETURN


Read_Right:
  HIGH RLinePwr                                 ' activate sensor
  HIGH RLineIn                                  ' discharge QTI cap
  PAUSE 1
  RCTIME RLineIn, 1, rLine                      ' read sensor value
  LOW RLinePwr                                  ' deactivate sensor
  RETURN


' Writes value in DEC4 format to LCD at cursor
' -- position LCD cursor
' -- put number to write in 'value'

LCD_DEC4:
  FOR idx = 3 TO 0                              ' loop through digits
    char = (value DIG idx) + "0"                ' convert to ASCII
    GOSUB LCD_Write_Char                        ' write on LCD
  NEXT
```

```
  RETURN


' Send command to LCD
' -- put command byte in 'char'

LCD_Command:                                    ' write command to LCD
  LOW RS
  GOTO LCD_Write_Char

' Write character to current cursor position
' -- but byte to write in 'char'

LCD_Write_Char:                                 ' write character to LCD
  LcdDirs = %1111                               ' enable LCD bus outputs
  LcdBusOut = char.HIGHNIB                      ' output high nibble
  PULSOUT E, 3                                  ' strobe the Enable line
  LcdBusOut = char.LOWNIB                       ' output low nibble
  PULSOUT E, 3
  HIGH RS                                       ' return to character mode
  LcdDirs = %0000                               ' release LCD bus
  RETURN
```

Program Code: Reading IR Object Detection Sensors

This program reads and displays the IR object detection sensor values on the LCD. The graphic below shows what the LCD will would display when an object is detected by the right sensor, but not with the left.


```
' =============================================================================
'
'   File..... SumoBot_4.1_IR_Sensor_LCD.BS2
'   Purpose... Tests IR sensor circuits
'   Author.... Parallax, Inc.
'              (Copyright (c) 2002 - 2004, All Rights Reserved)
'   E-mail.... support@parallax.com
'   Started...
'   Updated... 03 JUN 2004
'
'   {$STAMP BS2}
'   {$PBASIC 2.5}
'
' =============================================================================


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

LfIrOut         PIN     4                       ' left IR LED output
```

```
LfIrIn          PIN     11                      ' left IR sensor input
RtIrOut         PIN     15                      ' right IR LED output
RtIrIn          PIN     14                      ' right IR sensor input

E               PIN     1                       ' LCD Enable (1 = enabled)
RW              PIN     2                       ' Read/Write\
RS              PIN     3                       ' Reg Select (1 = char)
LcdDirs         VAR     DIRB                    ' dirs for I/O redirection
LcdBusOut       VAR     OUTB


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

LcdCls          CON     $01                     ' clear the LCD
LcdHome         CON     $02                     ' move cursor home
LcdCrsrL        CON     $10                     ' move cursor left
LcdCrsrR        CON     $14                     ' move cursor right
LcdDispL        CON     $18                     ' shift chars left
LcdDispR        CON     $1C                     ' shift chars right

LcdDDRam        CON     $80                     ' Display Data RAM control
LcdCGRam        CON     $40                     ' Character Generator RAM
LcdLine1        CON     $80                     ' DDRAM address of line 1
LcdLine2        CON     $C0                     ' DDRAM address of line 2


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

irBits          VAR     Nib                     ' storage for IR target data
irLeft          VAR     irBits.BIT1
irRight         VAR     irBits.BIT0

char            VAR     Byte                    ' character for LCD


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

Initialize:
  NAP 5                                         ' let LCD self-initialize
  DIRL = %11111110                              ' setup pins for LCD

LCD_Init:
  LcdBusOut = %0011                             ' 8-bit mode
  PULSOUT E, 3 : PAUSE 5
  PULSOUT E, 3 : PAUSE 0
  PULSOUT E, 3 : PAUSE 0
  LcdBusOut = %0010                             ' 4-bit mode
  PULSOUT E, 3
  char = %00101000                              ' 2-line mode
  GOSUB LCD_Command
  char = %00001100                              ' on, no crsr, no blink
  GOSUB LCD_Command
  char = %00000110                              ' inc crsr, no disp shift
  GOSUB LCD_Command
  char = LcdCls
  GOSUB LCD_Command
```

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

Main:
  DO
    FREQOUT LfIrOut, 1, 38500                   ' modulate left IR LED
    irLeft = ~LfIrIn                            ' read input (1 = target)

    FREQOUT RtIrOut, 1, 38500                   ' modulate right IR LED
    irRight = ~RtIrIn                           ' read input (1 = target)

    char = LcdLine1 + 1                         ' show left IR
    GOSUB LCD_Command
    char = irLeft + "0"                         ' convert bit to ASCII
    GOSUB LCD_Write_Char                        ' write it

    char = LcdLine1 + 6                         ' show right IR
    GOSUB LCD_Command
    char = irRight + "0"
    GOSUB LCD_Write_Char

    PAUSE 20
  LOOP

  END


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

' Send command to LCD
' -- put command byte in 'char'

LCD_Command:                                    ' write command to LCD
  LOW RS
  GOTO LCD_Write_Char

' Write character to current cursor position
' -- but byte to write in 'char'

LCD_Write_Char:                                 ' write character to LCD
  LcdDirs = %1111                               ' enable LCD bus outputs
  LcdBusOut = char.HIGHNIB                      ' output high nibble
  PULSOUT E, 3                                  ' strobe the Enable line
  LcdBusOut = char.LOWNIB                       ' output low nibble
  PULSOUT E, 3
  HIGH RS                                       ' return to character mode
  LcdDirs = %0000                               ' release LCD bus
  RETURN
```