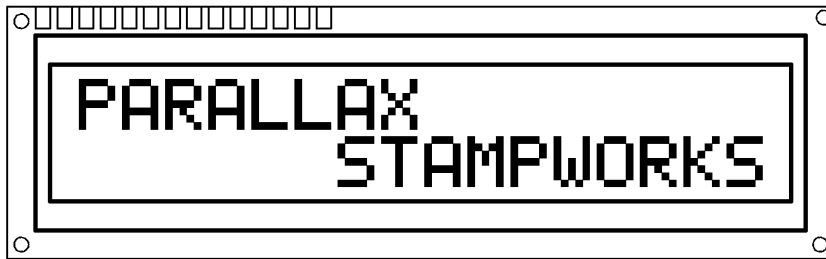


## StampWorks Using Character LCDs

While LEDs and seven-segment displays make great output devices, there will be projects that require providing more complex information to the user. Of course, nothing beats the PC video display, but these are large, expensive and almost always impractical for microcontroller projects. Character LCD modules, on the other hand, fit the bill well. These inexpensive modules allow both text and numeric output, use very few I/O lines and require little effort from the BASIC Stamp.

Character LCD modules are available in a wide variety of configurations: one-line, two-line and four-line are very common. Screen width is also variable, but is usually 16 or 20 characters for each line.



The StampWorks LCD module (2 lines x 16 characters).  
Datasheet is available for download from [www.parallaxinc.com](http://www.parallaxinc.com).

The StampWorks LCD module connects to the lab board by a 14-pin IDC header. The header is keyed, preventing the header from being inserted upside-down.

## Using Character LCDs

---

### Initialization

The character LCD must be initialized before sending information to it. The projects in this document initialize the LCD in accordance with the specification for the Hitachi HD44780 controller. The Hitachi controller is the most popular available and many controllers are compatible with it.

### Modes Of Operation

There are two essential modes of operation with character LCDs: sending a character and sending a command. When sending a character, the RS line is high and the data sent is interpreted as a character to be displayed at the current cursor position. The code sent is usually the ASCII code FOR the character. Several non-ASCII characters also are available in the LCD, as well as up to eight user-programmable custom characters.

Commands are sent to the LCD by taking the RS line low before sending the data. Several standard commands are available to manage and manipulate the LCD display.

Clear	\$01	Clears the LCD and moves cursor to first position of first line
Home	\$02	Moves cursor to first position of first line
Cursor Left	\$10	Moves cursor to the left
Cursor Right	\$14	Moves cursor to the right
Display Left	\$18	Shifts entire display to the left
Display Right	\$1C	Shifts entire display to the right

### Connecting The LCD

The StampWorks LCD has a 14-pin IDC connector at the end of its cable. The connector is "keyed" so that it is always inserted correctly into the StampWorks lab. Simply align the connector key (small bump) with the slot in the LCD socket and press the connector into the socket until it is firmly seated.



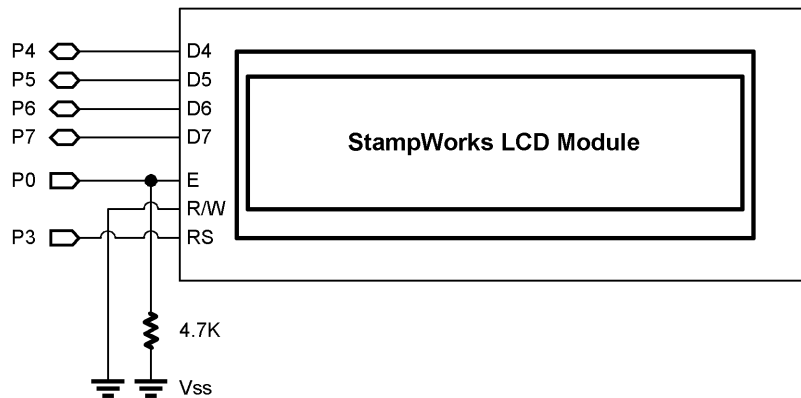
## Experiment #11: A Basic LCD Demonstration

This program demonstrates character LCD fundamentals by putting the StampWorks LCD module through its paces.

### New PBASIC elements/commands to know:

- PULSOUT
- HighNib, LowNib
- ^ (Exclusive OR operator)

### Building The Circuit



```
=====
|
|
| File..... Ex11 - LCD Demo.BS2
| Purpose... Essential LCD control
| Author... Parallax
| E-mail... stamptech@parallaxinc.com
| Started...
| Updated... 01 MAY 2002
|
|
| {$STAMP BS2}
|
|=====
```

## Experiment #11: A Basic LCD Demonstration

---

```
' -----
' Program Description
' -----

' This program demonstrates essential character LCD control.
'
' The connections for this program conform to the BS2p LCDIN and LCDOUT
' commands. Use this program for the BS2, BS2e or BS2sx. There is a separate
' program for the BS2p.

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

E             CON      0             ' LCD Enable pin (1 = enabled)
RS           CON      3             ' Register Select (1 = char)
LCDbus       VAR      OutB         ' 4-bit LCD data bus

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

ClrLCD       CON      $01          ' clear the LCD
CrsrHm       CON      $02          ' move cursor to home position
CrsrLf       CON      $10          ' move cursor left
CrsrRt       CON      $14          ' move cursor right
DispLf       CON      $18          ' shift displayed chars left
DispRt       CON      $1C          ' shift displayed chars right
DDRam        CON      $80          ' Display Data RAM control

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

char         VAR      Byte         ' character sent to LCD
index        VAR      Byte         ' loop counter

' -----
' EEPROM Data
' -----
```

## Experiment #11: A Basic LCD Demonstration

```
Msg          DATA    "THE BASIC STAMP!", 0    ' preload EEPROM with message

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

Initialize:
  DirL = %11111101    ' setup pins for LCD

LCD_Init:
  PAUSE 500           ' let the LCD settle
  LCDbus = %0011      ' 8-bit mode
  PULSOUT E, 1
  PAUSE 5
  PULSOUT E, 1
  PULSOUT E, 1
  LCDbus = %0010      ' 4-bit mode
  PULSOUT E, 1
  char = %00001100    ' disp on, crsr off, blink off
  GOSUB LCD_Command
  char = %00000110    ' inc crsr, no disp shift
  GOSUB LCD_Command

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

Main:
  char = ClrLCD       ' clear the LCD
  GOSUB LCD_Command
  PAUSE 500
  index = Msg         ' get EE address of message

Read_Char:
  READ index, char    ' get character from EEPROM
  IF (char = 0) THEN Msg_Done ' if 0, message is complete
  GOSUB LCD_Write     ' write the character
  index = index + 1   ' point to next character
  GOTO Read_Char      ' go get it

Msg_Done:
  PAUSE 2000          ' the message is complete
  char = CrsrHm       ' wait 2 seconds
  GOSUB LCD_Command   ' move the cursor home
  char = %00001110    ' turn the cursor on
```

## Experiment #11: A Basic LCD Demonstration

---

```
GOSUB LCD_Command
PAUSE 500

char = CrsrRt
FOR index = 1 TO 15           ' move the cursor accross display
  GOSUB LCD_Command
  PAUSE 150
NEXT

FOR index = 14 TO 0         ' go backward by moving cursor
  char = DDRam + index     ' to a specific address
  GOSUB LCD_Command
  PAUSE 150
NEXT

char = %00001101           ' cursor off, blink on
GOSUB LCD_Command
PAUSE 2000

char = %00001100           ' blink off
GOSUB LCD_Command

FOR index = 1 TO 10        ' flash display
  char = char ^ %00000100  ' toggle display bit
  GOSUB LCD_Command
  PAUSE 250
NEXT
PAUSE 1000

FOR index = 1 TO 16        ' shift display
  char = DispRt
  GOSUB LCD_Command
  PAUSE 100
NEXT
PAUSE 1000

FOR index = 1 TO 16        ' shift display back
  char = DispLf
  GOSUB LCD_Command
  PAUSE 100
NEXT
PAUSE 1000
GOTO Main                 ' do it all over

END
```

## Experiment #11: A Basic LCD Demonstration

---

```
'-----  
' Subroutines  
'-----  
  
LCD_Command:  
  LOW RS                                ' enter command mode  
  
LCD_Write:  
  LCDbus = char.HighNib                 ' output high nibble  
  PULSOUT E, 1                          ' strobe the Enable line  
  LCDbus = char.LowNib                  ' output low nibble  
  PULSOUT E, 1  
  HIGH RS                                ' return to character mode  
  RETURN
```

### Behind The Scenes

This is a very simple program, which demonstrates the basic functions of a character LCD. The LCD is initialized using four-bit mode in accordance with the Hitachi HD44780 controller specifications. This mode is used to minimize the number of BASIC Stamp I/O lines needed to control the LCD. While it is possible to connect to and control the LCD with eight data lines, this will not cause a noticeable improvement in program performance and will use four more I/O lines.

## Experiment #11: A Basic LCD Demonstration

---

The basics of the initialization are appropriate for most applications:

- The display is on
- The cursor is off
- Display blinking is disabled
- The cursor is automatically incremented after each write
- The display does not shift

With the use of four data bits, two write cycles are necessary to send a byte to the LCD. The BASIC Stamps' `HighNib` and `LowNib` variable modifiers make this process exceedingly easy. Each nibble is latched into the LCD by blipping the E (enable) line with `PULSOVT`.

The demo starts by clearing the LCD and displaying a message that has been stored in a `DATA` statement. This technique of storing messages in EEPROM is very useful and makes programs easier to update. In this program, characters are written until a zero is encountered. This method lets us change the length of the string without worry about `FOR-NEXT` control settings. With the message displayed, the cursor position is returned home (first position of first line) and turned on (an underline cursor appears).

The cursor is sent back and forth across the LCD using two techniques. The first uses the cursor-right command. Moving the cursor back is accomplished by manually positioning the cursor. Manual cursor positioning is required by many LCD programs for tidy formatting of the information in the display.

With the cursor back home, it is turned off and the blink attribute is enabled. `Blink` causes the current cursor position to alternate between the character and a solid black box. This can be useful as an attention getter. Another attention-getting technique is to flash the entire display. This is accomplished by toggling the display enable bit. The Exclusive OR operator (^) simplifies bit toggling, as any bit XOR'd with a "1" will invert (1 XOR 1 = 0, 0 XOR 1 = 1).

Using the display shift commands, the entire display is shifted off-screen to the right, then back. What this demonstrates is that the display is actually a window into the LCD's memory. One method of using the additional memory is to write messages off-screen and shift to them.