



Column #31, September 1997 by Jon Williams:

Demystifying Character Based LCDs

There is no doubt that Scott's LCD Backpack has saved countless Stamp projects from oblivion and, in the process, has become one of the most popular products in his catalog.

There are times, however, that I do very simple Stamp projects that have adequate code space and I/O resources, so I cannot justify the expense of using an LCD Backpack. Direct control of character-based LCDs is not difficult, yet, it requires some specific steps to succeed. The gory details like initialization code and Register Select control -- things that the LCD Backpack hides from us -- must be handled by your Stamp code. Like I said, it's not that difficult once you get the hang of it.

How They Work

The operation of character-based LCDs is analogous to your PC's video card and monitor. In your PC, information is sent by the program to the video card via the PC bus. The video card is responsible for the actual task of creating, updating, and refreshing the display. Information in your display is maintained in the video card memory. Our LCDs behave in the same manner: they accept data and commands via a bus, maintain character information in memory, and manage the built-in display. The brains behind the

Column #31: Demystifying Character Based LCDs

LCD is the Hitachi HD44780. In addition to managing the display (i.e., writing a character, clearing the display, moving the cursor, etc.), it contains three areas of memory: CGROM, DDRAM, and CGRAM.

The CGROM (Character Generator ROM) contains the dot patterns for the characters that can be displayed. This ROM contains most of the US ASCII character set and several Japanese kana characters and symbols.

The DDRAM (Data Display RAM) is the memory where the display contents are stored. The actual content of this memory are the character codes to be displayed. The HD44780 takes care of retrieving the dot patterns from the CGROM and putting them into the display. The DDRAM and its manipulation can be a bit confusing at first, since it usually contains many more characters than can be physically displayed. You can find LCDs as small as 2x8 and as large as 4x20. Keep in mind that the physical display is simply a window into the DDRAM.

In practice, this can be quite useful. You could, for example, write several messages to the LCD as part of the initialization sequence. During program operation, you can then point to a message by setting the display to the correct DDRAM address.

The CGRAM (Character Generator RAM) is memory that we can manipulate to create our own characters. In a pinch, it can also be used as external, general-purpose memory. Use the CGRAM as external memory requires another control line (R/W, LCD pin 5) that we don't normally use in Stamp projects. We will, however, be covering the detailed use of CGRAM to create our own characters next month.

Making the Connection

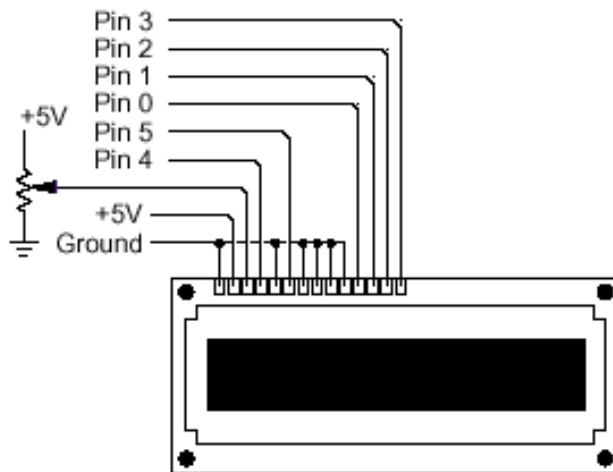
Connection to the LCD is through a 14-pin interface, physically arranged 1x14 or 2x7. Don't panic, Stamp 1 users. We only need to use six lines to write to the display. And since four of these lines are tri-stated when not in use, they can be shared by other hardware (refer to Parallax Stamp Application Note #1). Table 31.1 shows the LCD pin connections and how we'll interface them to the Stamp (the connections are the same for the Stamp 1 and Stamp 2).

Notice that the data bus is eight bits wide, but we're only using four bits. The ability to connect to the LCD through a four-bit interface makes it possible to use the LCD with the Stamp 1. If you're using the Stamp 2, you may elect to use the full eight-bit data bus.

Table 31.1: LCD pin connections and BASIC Stamp interface

LCD Pin	Symbol	Level	Function	Stamp Pin
1	Vss	0V	Ground	Ground
2	Vdd	+5V	Supply voltage (2 mA max)	+5 or pin X
3	Vo	0 – 5V	Contrast control	
4	RS	L/H	L = Instruction, H = Data	4
5	R/W	L/H	L = Write data, H = Read data	Ground
6	E	H → L	Enable signal	5
7	DB0	L/H	Data bus, bit 0	Ground
8	DB1	L/H	Data bus, bit 1	Ground
9	DB2	L/H	Data bus, bit 2	Ground
10	DB3	L/H	Data bus, bit 3	Ground
11	DB4	L/H	Data bus, bit 4	0
12	DB5	L/H	Data bus, bit 5	1
13	DB6	L/H	Data bus, bit 6	2
14	DB7	L/H	Data bus, bit 7	3

Figure 31.1: BASIC Stamp connections to HD44780-compatible LCD



In my projects, I always use the four-bit connection. In my opinion, the four pins this mode saves is worth the few extra lines of code necessary to use the four-bit bus. Another nice aspect of these LCDs is that their power requirement is very low (2 mA max); so much so that you may use a Stamp pin for power control.

Display Basics

Use of the LCD is pretty straightforward. After power-up, wait a half second or so to let the LCD run its own initialization. Since the default mode is eight bits, we'll have to re-initialize it to accept our data via the four-bit bus. When the four-bit initialization is complete, we can send our characters or commands. The difference is determined by the state of the RS (Register Select) line at the time we write a code. The RS line is set high for characters, low for LCD commands.

Enough chat, let's get to the code. Listings 31.1 (Stamp 1) and 31.2 (Stamp 2) demonstrate the basic LCD features. This code is designed for a one-line, 16-character display, which is readily available from surplus suppliers like Timeline and B.G. Micro. Timeline supplies the Hitachi driver manual with the purchase of LCDs. I recommend that you get a copy. It explains the details of initialization, writing to and reading from the LCD, and different schemes for display contrast control. Don't worry if you're not able to get a copy right away, the listings are well documented and, with a little study, you should have no trouble.

This initialization code is required to allow the LCD to operate in four-bit mode. After setting the four-bit interface, this section of code turns the display on, turns off the underline cursor, and causes the cursor to increment after each character is written. Just to ensure that there is no garbage left from any previous operations, the Display Clear command is sent to the LCD.

Writing a character or command is done in these steps:

1. Set the RS line (HIGH for character, LOW for command).
2. Place the high nibble of the character/command byte on the bus.
3. Strobe the Enable line (cause a HIGH-to-LOW transition).
4. Place the low nibble on the bus.
5. Strobe the Enable line one more time.

All the work is done by the sub-routine LCDwr. The entry at label LCDcmd sets the RS line low for LCD commands before writing the code with LCDwr.

Multiple Lines

Multi-line control is an aspect of character-based LCDs that often gives new users trouble. As stated earlier, this has to do with the physical display being a window into the

DDRAM. Once the layout of the DDRAM is understood, there is rarely a problem. Be aware that you can obtain 2x8 displays that are physically configured as a 1x16.

Honestly, they're a pain to deal with, so I try to avoid them. The DDRAM address range can be extended with the use of additional drivers (HD44100s that are built into the LCD module), and the addresses are not contiguous. Table 31.2 shows how they're laid out in one-, two-, and four-line displays.

Do you see how the addresses are interleaved? Keep this in mind when shifting multi-line displays. Please refer to the HD44780 manual for additional information. The ability to write beyond the first line requires a small change in the initialization sequence. Once this is done, it is a simple matter of setting the DDRAM address before writing your characters. See Listing 31.3 for details.

Next Time

Next time, we'll cover the use of CGRAM and how to create and display custom characters, including animation techniques. Until then, run the sample code and play with it. If you have any questions, please send a letter or E-Mail. And don't hesitate to pass on any new tricks that you create so we can share them with others.

As a final note, I'd like to publicly thank Scott Edwards for the fantastic work he has done with the Stamp Applications column and for having the faith in me to carry the torch. I'm sure you'll join me in wishing Scott well in his ever-expanding business. And, rest assured, we haven't heard the last of Scott when it comes to Stamps. I'm sure he has something big brewing.

```
' Program Listing 31.1
' Nuts & Volts: Stamp Applications, September 1997

' ---- [ Title ]-----
'
' File..... LCDDEMO1.BAS
' Purpose... Stamp 1 -> LCD (4-bit interface)
' Author.... Jon Williams
' E-mail.... jonwms@aol.com
' Started... 16 JUL 1994
' Updated... 25 JUL 1997

' ---- [ Program Description ]-----
'
' This program demonstrates the various standard features of a 1x16 LCD
' display that uses the Hitachi HD44780 controller.
```

Column #31: Demystifying Character Based LCDs

```
'
' LCD Connections:
'
' LCD          (Function)          Stamp
' -----
' pin 1        Vss                  Gnd
' pin 2        Vdd                  +5
' pin 3        Vo                    Gnd (or wiper of 10K pot)
' pin 4        RS                    Pin 4
' pin 5        R/W                   Gnd
' pin 6        E                      Pin 5
' pin 7        DB0                   Gnd
' pin 8        DB1                   Gnd
' pin 9        DB2                   Gnd
' pin 10       DB3                   Gnd
' pin 11       DB4                   Pin 0
' pin 12       DB5                   Pin 1
' pin 13       DB6                   Pin 2
' pin 14       DB7                   Pin 4

' ---- [ Revision History ]-----
'
' 16 JUL 94 : Version 1.0 - compilation of code from last 3 months
' 08 AUG 96 : Trimmed code to save space -- no performance changes!
' 25 JUL 97 : Updated for Nuts & Volts

' ---- [ Constants ]-----
'
SYMBOL  RS      = 4                    ' Register Select (1 = char)
SYMBOL  E       = 5                    ' LCD enable pin (1 = enabled)

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

' ---- [ Variables ]-----
'
SYMBOL  char     = B1                  ' character sent to LCD
SYMBOL  index    = B2                  ' loop counter

' ---- [ EEPROM Data ]-----
```

Column #31: Demystifying Character Based LCDs

```
'
    EEPROM ("THE BASIC STAMP!")      ' preload EEPROM with message

' ----[ Initialization ]-----
'
Init:  Dirs = %00111111             ' set 0-5 as outputs
      Pins = %00000000             ' clear the pins

' Initialize the LCD (Hitachi HD44780 controller)
'
LCDini: PAUSE 500                   ' let the LCD settle
      Pins = %0011                 ' 8-bit mode
      PULSOUT E, 1
      PAUSE 5
      PULSOUT E, 1
      PULSOUT E, 1
      Pins = %0010                 ' 4-bit mode
      PULSOUT E, 1
      char = %00001100             ' disp on, crsr off, blink off
      GOSUB LCDcmd
      char = %00000110             ' inc crsr, no disp shift
      GOSUB LCDcmd
      char = ClrLCD
      GOSUB LCDcmd

' ----[ Main Code ]-----
'
Start:  FOR index = 0 TO 15
      READ index, char             ' get character from EEPROM
      GOSUB LCDwr                  ' write it
      PAUSE 50                     ' delay between chars
    NEXT
    PAUSE 2000                     ' wait 2 seconds

    char = CrsrHm                  ' move the cursor home
    GOSUB LCDcmd

    char = %00001110               ' turn the cursor on
    GOSUB LCDcmd
    PAUSE 500

    char = CrsrRt
    FOR index = 1 TO 15            ' move the cursor accross display
      GOSUB LCDcmd
      PAUSE 100
    NEXT

    FOR index = 14 TO 0 STEP -1    ' go backward by moving to
```

Column #31: Demystifying Character Based LCDs

```
char = DDRam + index      ' a specific address
GOSUB LCDcmd
PAUSE 100
NEXT

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

char = %00001100         ' blink off
GOSUB LCDcmd

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

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

FOR index = 1 TO 16      ' shift display back
char = DispLf
GOSUB LCDcmd
PAUSE 100
NEXT
PAUSE 1000

char = ClrLCD            ' clear the LCD
GOSUB LCDcmd
PAUSE 500

GOTO Start              ' do it all over

' ----[ Subroutines ]-----
'
' Send command to the LCD
'
' Load char with command value, then call
'
' Clear the LCD..... $01, %00000001
' Home the cursor..... $02, %00000010
' Display control..... (see below)
' Entry mode..... (see below)
```

Column #31: Demystifying Character Based LCDs

```
' Cursor left..... $10, %00010000
' Cursor right..... $14, %00010100
' Scroll display left..... $18, %00011000
' Scroll display right..... $1C, %00011100
' Set CG RAM address..... %01aaaaaa (Character Generator)
' Set DD RAM address..... %1aaaaaaa (Display Data)
'
' Display control byte:
'
'   % 0 0 0 0 1 D C B
'       | | +-- blink character under cursor (1=blink)
'       | +---- cursor on/off (1=on)
'       +----- display on/off (1=on)
'
' Entry mode byte:
'
'   % 0 0 0 0 0 1 X S
'       | +-- shift display (S=1), left (X=1), right (X=0)
'       +---- cursor move: right (X=1), left (X=0)
'
'
LCDcmd: LOW RS                               ' enter command mode
                                              ' then write the character
'
' Write ASCII char to LCD
'
LCDwr:  Pins = Pins & %11010000             ' save 7, 6 and RS; clear bus
        Pins = char / 16 | Pins             ' output high nibble
        PULSOUT E, 1                       ' strobe the Enable line
        Pins = Pins & %11010000
        Pins = char & $0F | Pins            ' output low nibble
        PULSOUT E, 1
        HIGH RS                             ' return to character mode
        RETURN
```

Column #31: Demystifying Character Based LCDs

```
' Program Listing 31.2
' Nuts & Volts: Stamp Applications, September 1997

' ----[ Title ]-----
'
' File..... LCDDEMO1.BS2
' Purpose... Stamp 2 -> LCD (4-bit interface)
' Author.... Jon Williams
' E-mail.... jonwms@aol.com
' Started... 16 JUL 1994
' Updated... 25 JUL 1997

' ----[ Program Description ]-----
'
' This program demonstrates the various standard features of a 1x16 LCD
' display that uses the Hitachi HD44780 controller.
'
' LCD Connections:
'
' LCD          (Function)          Stamp
' -----
' pin 1        Vss                 Gnd
' pin 2        Vdd                 +5
' pin 3        Vo                  Gnd (or wiper of 10K pot)
' pin 4        RS                  Pin 4
' pin 5        R/W                 Gnd
' pin 6        E                   Pin 5
' pin 7        DB0                 Gnd
' pin 8        DB1                 Gnd
' pin 9        DB2                 Gnd
' pin 10       DB3                 Gnd
' pin 11       DB4                 Pin 0
' pin 12       DB5                 Pin 1
' pin 13       DB6                 Pin 2
' pin 14       DB7                 Pin 4

' ----[ Revision History ]-----
'
' 16 JUL 94 : Version 1.0 - compilation of code from last 3 months
' 08 AUG 96 : Trimmed code to save space -- no performance changes!
' 25 JUL 97 : Updated for Nuts & Volts

' ----[ Constants ]-----
'
RS      CON      4          ' Register Select (1 = char)
E       CON      5          ' LCD Enable pin (1 = enabled)

' LCD control characters
```

Column #31: Demystifying Character Based LCDs

```

'
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 ]-----
'
Msg      DATA  "THE BASIC STAMP!"  ' preload EEPROM with message

' ---- [ Initialization ]-----
'
Init:    DirL = %00111111          ' set pins 0-5 as outputs
         Outs = $0000              ' clear the pins

' Initialize the LCD (Hitachi HD44780 controller)
'
LCDini:  PAUSE 500                 ' let the LCD settle
         OutA = %0011              ' 8-bit mode
         PULSOUT E, 1
         PAUSE 5
         PULSOUT E, 1
         PULSOUT E, 1
         OutA = %0010              ' 4-bit mode
         PULSOUT E, 1
         char = %00001100          ' disp on, crsr off, blink off
         GOSUB LCDcmd
         char = %00000110          ' inc crsr, no disp shift
         GOSUB LCDcmd
         char = ClrLCD
         GOSUB LCDcmd

' ---- [ Main Code ]-----
'
Start:   FOR index = 0 TO 15
         READ Msg + index, char    ' get character from EEPROM
         GOSUB LCDwrt              ' write it
         PAUSE 50                  ' delay between chars
NEXT

```

Column #31: Demystifying Character Based LCDs

```
PAUSE 2000                                ' wait 2 seconds

char = CrsrHm                              ' move the cursor home
GOSUB LCDcmd

char = %00001110                          ' turn the cursor on
GOSUB LCDcmd
PAUSE 500

char = CrsrRt
FOR index = 1 TO 15                        ' move the cursor accross display
  GOSUB LCDcmd
  PAUSE 100
NEXT

FOR index = 14 TO 0                       ' go backward by moving to
  char = DDRam + index                    ' a specific address
  GOSUB LCDcmd
  PAUSE 100
NEXT

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

char = %00001100                          ' blink off
GOSUB LCDcmd

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

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

FOR index = 1 TO 16                       ' shift display back
  char = DispLf
  GOSUB LCDcmd
  PAUSE 100
NEXT
PAUSE 1000

char = ClrLCD                              ' clear the LCD
GOSUB LCDcmd
```

Column #31: Demystifying Character Based LCDs

```

        PAUSE 500

        GOTO Start                ' do it all over

' ----[ Subroutines ]-----
'
' Send command to the LCD
'
' Load char with command value, then call
'
'   Clear the LCD..... $01, %00000001
'   Home the cursor..... $02, %00000010
'   Display control..... (see below)
'   Entry mode..... (see below)
'   Cursor left..... $10, %00010000
'   Cursor right..... $14, %00010100
'   Scroll display left..... $18, %00011000
'   Scroll display right..... $1C, %00011100
'   Set CG RAM address..... %01aaaaaa (Character Generator)
'   Set DD RAM address..... %1aaaaaaa (Display Data)
'
' Display control byte:
'
'   % 0 0 0 0 1 D C B
'           | | +-- blink character under cursor (1=blink)
'           | +---- cursor on/off (1=on)
'           +----- display on/off (1=on)
'
' Entry mode byte:
'
'   % 0 0 0 0 0 1 X S
'           | +-- shift display (S=1), left (X=1), right (X=0)
'           +---- cursor move: right (X=1), left (X=0)
'
LCDcmd: LOW RS                    ' enter command mode
                                   ' then write the character

' Write ASCII char to LCD
'
LCDwr:  OutA = char.HIGHNIB        ' output high nibble
        PULSOUT E, 1              ' strobe the Enable line
        OutA = char.LOWNIB        ' output low nibble
        PULSOUT E, 1
        HIGH RS                    ' return to character mode
        RETURN

```

Column #31: Demystifying Character Based LCDs

```
' Program Listing 31.3
' Nuts & Volts: Stamp Applications, September 1997

' ----[ Title ]-----
'
' File..... LCDDEMO2.BAS
' Purpose... Stamp 1 -> Multi-line LCD (4-bit interface)
' Author.... Jon Williams
' E-mail.... jonwms@aol.com
' Started... 16 JUL 1994
' Updated... 25 JUL 1997

' ----[ Program Description ]-----
'
' This program demonstrates the various standard features of a 1x16 LCD
' display that uses the Hitachi HD44780 controller.
'
' LCD Connections:
'
' LCD          (Function)          Stamp
' -----
' pin 1        Vss                 Gnd
' pin 2        Vdd                 +5
' pin 3        Vo                  Gnd (or wiper of 10K pot)
' pin 4        RS                  Pin 4
' pin 5        R/W                 Gnd
' pin 6        E                   Pin 5
' pin 7        DB0                 Gnd
' pin 8        DB1                 Gnd
' pin 9        DB2                 Gnd
' pin 10       DB3                 Gnd
' pin 11       DB4                 Pin 0
' pin 12       DB5                 Pin 1
' pin 13       DB6                 Pin 2
' pin 14       DB7                 Pin 4

' ----[ Revision History ]-----
'
' 16 JUL 94 : Version 1.0 - compilation of code from last 3 months
' 08 AUG 96 : Trimmed code to save space -- no performance changes!
' 25 JUL 97 : Updated for Nuts & Volts

' ----[ Constants ]-----
'
SYMBOL  RS      = 4                ' Register Select (1 = char)
SYMBOL  E       = 5                ' LCD enable pin (1 = enabled)

' LCD control characters
```

Column #31: Demystifying Character Based LCDs

```

'
SYMBOL ClrLCD = $01           ' clear the LCD
SYMBOL CrsrHm = $02         ' move cursor to home position
SYMBOL CrsrLf = $10        ' move cursor left
SYMBOL CrsrRt = $14        ' move cursor right
SYMBOL DispLf = $18        ' shift displayed chars left
SYMBOL DispRt = $1C        ' shift displayed chars right
SYMBOL DDRam = $80         ' Display Data RAM control
SYMBOL Line1 = $00         ' starting address of line 1
SYMBOL Line2 = $40         ' starting address of line 2

' ---- [ Variables ] -----
'
SYMBOL char = B1           ' character sent to LCD
SYMBOL index = B2         ' loop counter

' ---- [ EEPROM Data ] -----
'
      EEPROM ("This is Line 1")   ' preload EEPROM with messages
      EEPROM ("This is Line 2")

' ---- [ Initialization ] -----
'
Init:  Dirs = %00111111         ' set 0-5 as outputs
       Pins = %00000000         ' clear the pins

' Initialize the LCD (Hitachi HD44780 controller)
'
LCDini: PAUSE 500               ' let the LCD settle
        Pins = %0011           ' 8-bit mode
        PULSOUT E, 1
        PAUSE 5
        PULSOUT E, 1
        PULSOUT E, 1
        Pins = %0010           ' 4-bit mode
        PULSOUT E, 1
        char = %00101000       ' 2-line mode
        GOSUB LCDcmd
        char = %00001100       ' disp on, crsr off, blink off
        GOSUB LCDcmd
        char = %00000110       ' inc crsr, no disp shift
        GOSUB LCDcmd

' ---- [ Main Code ] -----
'
Start: char = ClrLCD           ' clear LCD, home cursor
       GOSUB LCDcmd

```

Column #31: Demystifying Character Based LCDs

```
        PAUSE 1000

L1:     FOR index = 0 TO 13
        READ index, char                ' get character from EEPROM
        GOSUB LCDwr                    ' write it
        PAUSE 50                       ' delay between chars
    NEXT
    PAUSE 2000                          ' wait 2 seconds

    char = DDRam + Line2                ' move to line 2
    GOSUB LCDcmd

L2:     FOR index = 14 TO 27
        READ index, char
        GOSUB LCDwr
        PAUSE 50
    NEXT
    PAUSE 2000

    GOTO Start

' ----[ Subroutines ]-----
'
' Send command to the LCD
'
' Load char with command value, then call
'
' Clear the LCD..... $01, %00000001
' Home the cursor..... $02, %00000010
' Display control..... (see below)
' Entry mode..... (see below)
' Cursor left..... $10, %00010000
' Cursor right..... $14, %00010100
' Scroll display left..... $18, %00011000
' Scroll display right..... $1C, %00011100
' Set CG RAM address..... %01aaaaaa (Character Generator)
' Set DD RAM address..... %1aaaaaaa (Display Data)
'
' Display control byte:
'
'   % 0 0 0 0 1 D C B
'       | | +-- blink character under cursor (1=blink)
'       | +---- cursor on/off (1=on)
'       +----- display on/off (1=on)
'
' Entry mode byte:
'
'   % 0 0 0 0 0 1 X S
'       | +-- shift display (S=1), left (X=1), right (X=0)
```

Column #31: Demystifying Character Based LCDs

```
'          +---- cursor move: right (X=1), left (X=0)
'
'
LCDcmd: LOW RS          ' enter command mode
                        ' then write the character

' Write ASCII char to LCD
'
LCDwr: Pins = Pins & %11010000 ' save 7, 6 and RS; clear bus
      Pins = char / 16 | Pins   ' output high nibble
      PULSOUT E, 1             ' strobe the Enable line
      Pins = Pins & %11010000
      Pins = char & $0F | Pins  ' output low nibble
      PULSOUT E, 1
      HIGH RS                  ' return to character mode
      RETURN
```

