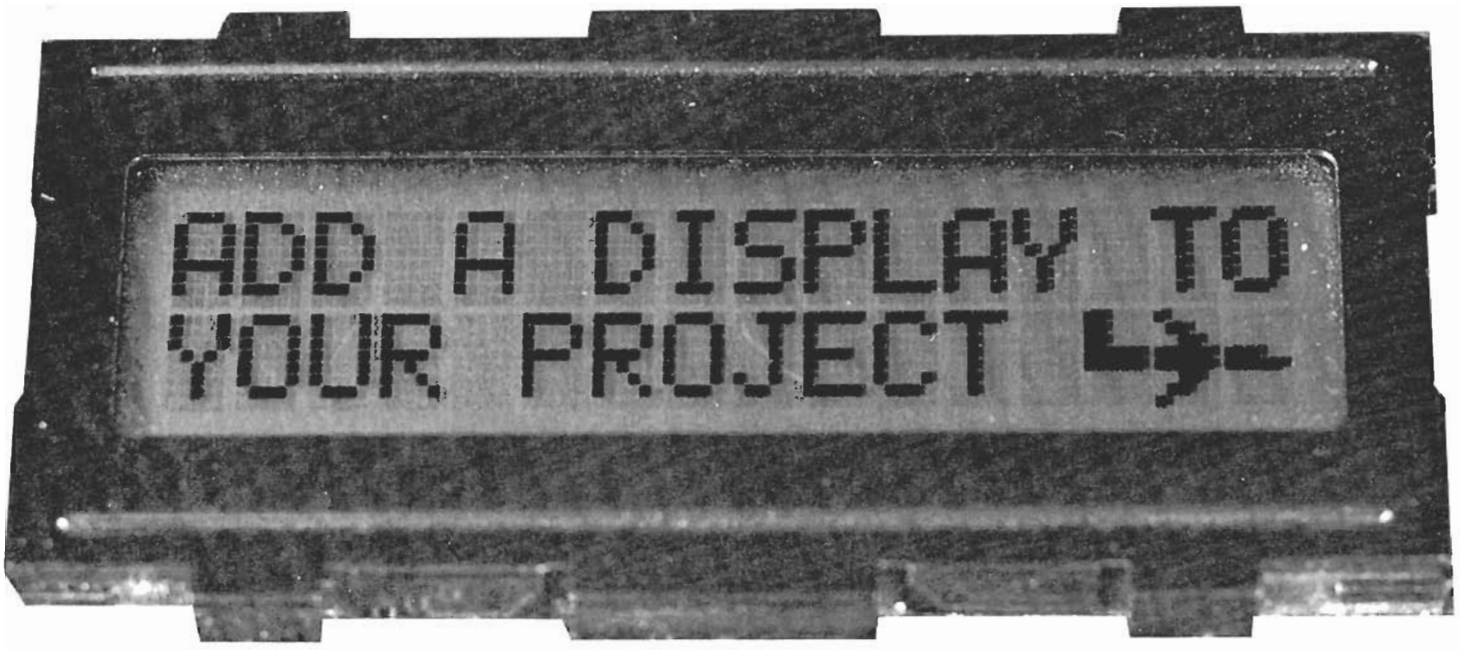

The days of LED indicators and segmented displays are numbered. Now you can add an alpha-numeric LCD to your home project easily and inexpensively.

STEVEN AVRITCH

HAVE YOU EVER AVOIDED A PROJECT BE-cause it required a display that could handle numbers, letters, and symbols? Have you ever given up on a project because the display had to be at least 10, 20, maybe even 40 characters long?

You can solve all of those problems by using a simple and inexpensive alpha-numeric LCD module which contains a controller chip that does most of the work for you! This article will show you how to use LCD's with a simple microcontroller- or microprocessor-based design. Note that most small LCD modules use the Hitachi HD44780 LCD controller chip (see block diagram in Fig. 1). This article will therefore be limited to a discussion of LCD modules that use, or are compatible with, the HD44780 controller format. Common LCD modules include those manufactured by Optrex, Epson, Hitachi, Amperex, and Densitron.

Multi-character readouts are usually constructed using individually wired, multiplexed display segments. The host microprocessor sequentially flashes the desired character on each digit of the display, one at a time. The microprocessor is fast enough so that the naked eye sees the display as it should appear. That method of multiplexing the digits of a display is often used because it reduces the amount of external hardware required compared to non-multiplexed systems. However, multiplexing requires the microprocessor to continually update the display, and the amount of external wiring must be increased as additional digits are added (see Fig. 2).

For example, a 10-digit numeric display requires approximately 100 wires and over 20 components. (A 10-digit alpha-numeric display requires even more wires.) The equivalent display (including alpha-numerics) implemented with an LCD module would require only 10 wires and 2 components: the LCD module and a potentiometer for contrast control. Using an LCD module, a designer can add a display containing up to 80 characters with as little as 10 wires, 7 of which connect the display module to the host microcontroller/processor, plus 1 power, 1 ground, and 1 LCD drive wire for contrast control. That's all!

The software interface between the host and the display module is just as simple as the wiring. The display modules automatically handle all refresh and multiplexing functions. The host needs only to write the data to be displayed and a few control codes (such as display on, display off, scroll left, scroll right, etc.) to the module; the on-board LCD controller chip does the rest.

LCD modules have not been used heavily in the past because of their high costs. However, the cost of the modules has since dropped considerably, and they are now commonly found in many of the popular electronics supply houses. For example, a 32-character display (2 lines, 16 characters per line, 16×2) is available from Digi-Key for approximately $23. Similar displays can be obtained through surplus houses for approximately $8–$10.

Most of the small, inexpensive LCD modules contain a Hitachi HD44780 LCD Controller chip. That means that most of LCD modules follow the same standard format, have the same 14-pin interface, and are therefore compatible and interchangeable. The HD44780 is capable of controlling any size display up to 2 lines long and 40 characters wide with the same hardware interface. Commonly available display sizes include 16×1, 16×2, 20×2, 24×2, and 40×2 formats. That means that you

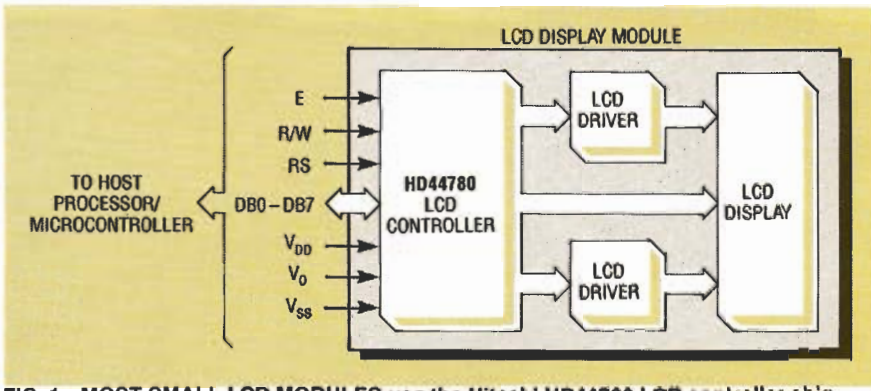

FIG. 1—MOST SMALL LCD MODULES use the Hitachi HD44780 LCD controller chip.

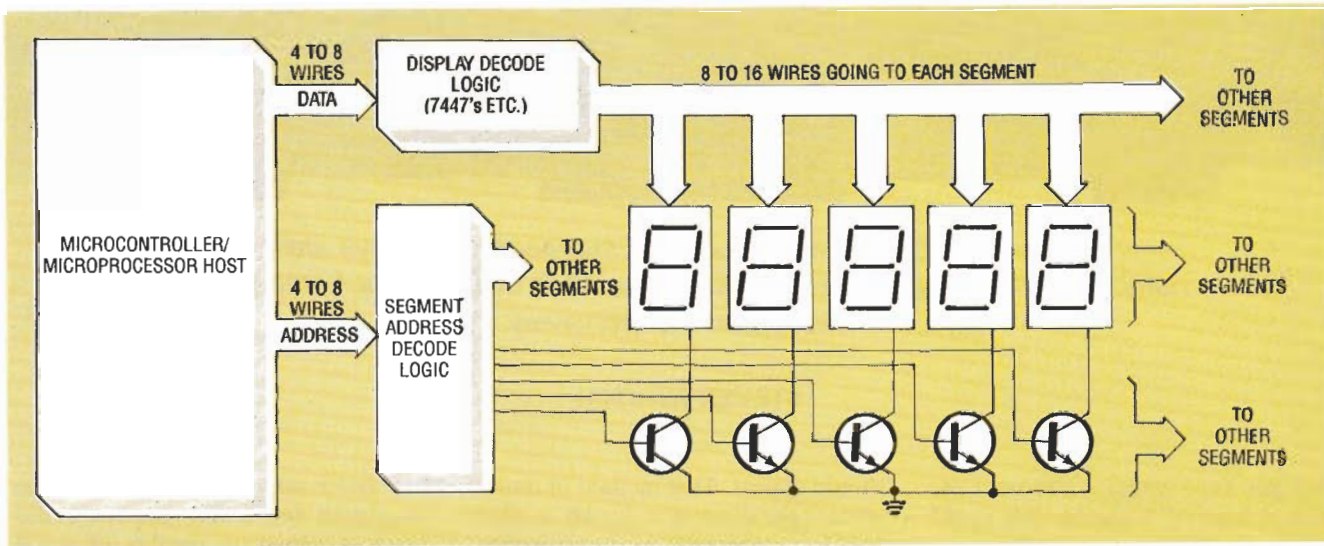

FIG. 2—MULTIPLEXING REQUIRES the microprocessor to continually update the display, and the amount of external wiring must be increased as additional digits are added.

can change the size of your display by simply plugging in a larger module. No other hardware modifications are required; only the software drivers specific to the application would need to change.

The LCD modules recognize standard ASCII code for letters (upper and lower case) and numbers in addition to a variety of symbols including ?, !, $AK, %, and ', just to name a few. In all, the LCD module supports 192 alpha-numeric characters and 32 special symbols. The modules also allow you to customize up to 8 user-defined characters of your own. On one home project the author customized three characters that, when displayed together, formed an airplane as can be seen in the photo.

The LCD modules are dot-matrix type displays with each character being formed from a 5-dot-wide by 7-dot-high block (5 × 7 font) or a 5-dot-wide by 10-dot-high block (5 × 10 font). The font is selected by issuing a control command as discussed later in this article.

There is also a cursor line under each character. The 5 × 10 font is better suited for certain lower-case letters such as g, y, and p (i.e. letters with descenders that go below the line that they're written on). Figure 3 shows examples of letters formed using the 5 × 7 and 5 × 10 dot-matrix formats for comparison. It should be noted that the 5 × 10 matrix font limits the display to one line regardless of whether the LCD module is a one-line or two-line display.

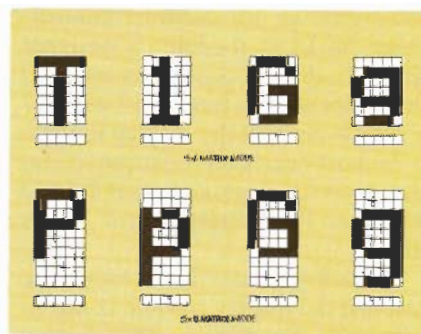

FIG. 3—HERE ARE SOME EXAMPLES of letters formed using the 5 × 7 and 5 × 10 dot-matrix formats.

Features of LCD modules

The LCD modules support a variety a display features that can accommodate just about any application. The following is a brief description of their features:

- Display on/off—allows the user to turn the display on and off from the host processor.
- Cursor on/off—user may select to display the cursor or suppress it.
- Cursor blink—the user may select a steady cursor or a blinking cursor. The character above the cursor also blinks.
- Scroll left/right—scrolls the data on the display.
- Return home—returns the cursor to the home position (address 0) and returns the display to the original position (if it had been previously scrolled)

Software interface

The software interface between the LCD module and a processor or microcontroller is relatively simple. There are two basic types of software operations: control operations (i.e. display on/off, cursor blink/noblink, etc.) and data operations. The control operations set up the features of the display, while the data operations write the actual data to be displayed to the LCD module.

The LCD module's on-board HD44780 controller chip contains 80 bytes of display RAM and is capable of supporting up to a 40 × 2 display (each byte of display RAM corresponds to a digit of the display). Smaller LCD modules simply do not

display the full 80 bytes of RAM. The display RAM is organized in the following format:
LINE 1:
Character position: 1 2 3 4 5 6 7 8 9...40
RAM address 0 1 2 3 4 5 6 7 8...27(hex)
LINE 2:
Character position: 1 2 3 4 5 6 7 8 9...40
RAM address 40 41 42 43 44 45 46 47 48...67(hex)

Smaller modules simply do not display the upper character positions associated with the upper addresses. For example, a 16 × 2 display uses addresses 00–0F (hex) for line 1 and 40–4F (hex) for line 2.

The HD44780 also contains 64 bytes of character-generator RAM. That is used to store the character patterns of the 8 user-defined characters (8 bytes per character). Once a user-defined character is set up in character-generator RAM, it may be accessed just as any other regular character. NOTE: in the 5 × 10 matrix mode, only four user-defined characters are supported, with each character requiring 11 bytes of character-generator RAM.

Software drivers

The host must contain two basic software drivers to support the LCD modules, the Control Write and Data Write drivers. The minimum functions that the software drivers must perform are:
Control Write:
- Sets up DB0–DB7 with the desired control code
- Sets the R/W line to logic zero
- Sets the RS line to logic zero
- Strobes the ENABLE line

Data Write:
- Sets up DB0–DB7 with the desired character
- Sets R/W line to logic zero
- Sets the RS line to logic one
- Strobes the ENABLE line

The user may also read data and control signals from the HD44780. Control Read and Data Read drivers are similar to the write drivers except that the R/W line is set to a logic one. Refer to Table 1 for a complete listing of the control codes and status flags available with the HD44780 LCD controller chip.

Subroutines for the MC68705

The following subroutines show the software drivers for data and control writes. The examples shown here are written in Motorola 6800-series assembler code and are targeted for the MC68705 microcontroller. These short routines can be easily translated into other assembly languages that can be used with other microcontrollers/microprocessors.

The Data Write subroutine (Listing 1) displays letters and symbols. The ASCII code of the letter/symbol to be displayed must be loaded into the Accumulator before calling the Data Write subroutine. Before the Control Write subroutine (Listing 2) can be called, the code of the control operation to be performed (from Table 1) must be loaded into the Accumulator.

Display initialization

The first operation that the software

LISTING 1

```
DISLET    STX    TEMPX        SAVE INDEX REGISTER
          STA    PORTA        PUT CHARACTER ON BUS
          BCLR   1,PORTB      SET R/W TO WRITE
          BSET   2,PORTB      SET RS TO DATA
          BSET   0,PORTB      TURN ON ENABLE
          BCLR   0,PORTB      TURN OFF ENABLE
          BCLR   2,PORTB      SET RS TO CONTROL
          LDX    #$20         \
DELAY1    DECX                 \  DELAY 120 us
          BNE    DELAY1        /  ASSUMING 1 us
                               /   CLOCK
          LDX    TEMPX        RESTORE INDEX REGISTER
          RTS                 RETURN FROM SUBROUTINE
```

LISTING 2

```
CONTROL   STX    TEMPX    SAVE INDEX REGISTER
          STA    PORTA    PUT CONTROL CODE ON BUS
          BCLR   1,PORTB  SET R/W TO WRITE
          BCLR   2,PORTB  SET RS TO CONTROL
          BSET   0,PORTB  TURN ENABLE ON
          BCLR   0,PORTB  TURN ENABLE OFF
          LDX    #$FF     \
DELAY2    DECX             \
          BNE    DELAY2    \
          CMP    #$02       \ DELAY FOR CONTROL
          BHI    DELAY4     /
DELAY3    JSR    ENDCNTL   /
          DECX             /
DELAY4    LDX    TEMPX      RESTORE INDEX REGISTER
ENDCNTL   RTS                RETURN
```

LISTING 3

```
INIT      LDA    #$01       CLEAR DISPLAY
          JSR    CONTROL
          LDA    #$02       RETURN DISPLAY TO HOME POSITION
          JSR    CONTROL
          LDA    #$38       SET UP FOR 2 LINES, 8 BIT INTERFACE,
                                 AND 5X7 MATRIX FORMAT
          JSR    CONTROL
          LDA    #$06       SET UP FOR CURSOR SHIFT WITH DATA WRITE
          JSR    CONTROL
          LDA    #$0C       SET UP FOR DISPLAY ON, CURSOR OFF,
                                 AND STEADY CURSOR (NO BLINK)
          JSR    CONTROL
          RTS               RETURN FROM SUBROUTINE
```

LISTING 4

```
CGINIT    LDA    #$40       SET UP FOR WRITES TO CG RAM
          JSR    CONTROL
          CLR    DATCNT     CLEAR BYTE COUNTER
NEXT      LDX    DATCNT     LOAD BYTE COUNTER
          LDA    PLANE,X    LOAD CG RAM DATA INTO ACCUMULATOR
          JSR    DISLET     WRITE BYTE TO CG RAM
          INC    DATCNT     INCREMENT COUNTER FOR NEXT BYTE
          LDA    DATCNT     \
          CMP    #24        / 24 BYTES WRITTEN ?
          BNE    NEXT
          LDA    #$80       SET UP FOR WRITES TO DD RAM
          JSR    CONTROL
          LDA    #$02       INITIALIZE DISPLAY TO HOME POSITION
          JSR    CONTROL
          RTS               RETURN FROM SUBROUTINE
```

TABLE 1—CONTROL OPERATIONS

Instruction	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Description and execution time
Clear	0	0	0	0	0	0	0	0	0	1	Clears Display Returns cursor to home position (1.64 ms)
Home	0	0	0	0	0	0	0	0	1	X	Return cursor to home position Return shifted display to home position (40 µs)
Mode	0	0	0	0	0	0	0	1	I/D	S	Control automatic RAM address INC/DEC and whether display shifts on writes (40 µs)
Display ON/OFF	0	0	0	0	0	0	1	D	C	B	Controls display ON/OFF Controls cursor ON/OFF Cursor blink ON/OFF (40 µs)
Cursor or display shift	0	0	0	0	0	1	S/C	R/L	X	X	Sifts cursor and/or display without changing display RAM (40 µs)
Function set	0	0	0	0	1	D/L	N	F	X	X	Set interface to 4 or 8 bits Set number of display lines Sets character font (40 µs)
Set CG RAM address	0	0	0	1	Address (CG)						Set address for subsequent writes to character generator. (CG) RAM. (40 µs)
Set DD RAM address	0	0	1	Address (DD)							Set address for subsequent writes to display (DD) RAM (40 µs)
Read busy flag & address	0	1	BF	Address Counter							Read status of busy flag (BF) and present address counter value (1 µs)
Write data to CG or DD RAM	1	0	Data to be written to DD or CG RAM. DD/CG destination based on last "set DD/CG RAM address" control command								Writes data to HD44780 (40µs)
Read data from CG or DD RAM	1	1	Data read from DD or CG RAM. Source of data (CG or DD) based on last "set DD/CG RAM address" control command								Reads data from HD44780 (40 µs)

I/D = 1: Increment address pointer on each subsequent read/write
0: Decrement address pointer on each subsequent read/write

2 = 1: Shift cursor with display
0: Hold cursor fixed

S/C = 1: Shift display (without changing DD RAM)
0: Shift cursor only

R/L = 1:Shift operations occur to the right
0: Shift operations occur to the left

DL = 1: 8-bit data-bus interface
0: 4-bit data-bus interface

N = 1: 2-line display
0: 1-line display

F = 1: 5×10 dot character font
0: 5×7 dot character font

BF = 1: Busy
0: Not busy

must perform is the initialization of the display. Initialization includes clearing the display and issuing the appropriate control commands that set the display up with the desired features. The INIT subroutine (Listing 3) is a sample initialization routine for a 16×2 display. The INIT routine sets the display up for 2 line, 5×7-font format, 8-bit interface mode, and suppressed cursor. Also, the INIT routine sets up the display to shift the cursor one position to the right on every data write.

Note: The display module requires 10 milliseconds to initialize after power is applied. The host, therefore, must wait at least 10 milliseconds before writing to the display following the power-up of the module.

CG RAM initialization

The CGINIT routine (Listing 4) illustrates the operations required to set up 3 of the 8 user-defined characters. The characters defined in the routine form an airplane when displayed together. Table 2 illustrates how each of the three user-definable characters are generated. Listing 5

continued on page 80

LISTING 5

```
PLANE   FCB   $00
        FCB   $00
        FCB   $00
        FCB   $1C
        FCB   $1F
        FCB   $00
        FCB   $00
        FCB   $00
        FCB   $10
        FCB   $0C
        FCB   $06
        FCB   $1F
        FCB   $1F
        FCB   $06
        FCB   $0C
        FCB   $10
        FCB   $18
        FCB   $18
        FCB   $18
        FCB   $1F
        FCB   $1F
        FCB   $00
        FCB   $00
        FCB   $00
```

ADD A DISPLAY

continued from page 62

TABLE 2—USER DEFINABLE CHARACTERS

CHARACTER CODES (DD RAM DATA) 7 6 5 4 3 2 1 0	CG RAM ADDRESS 5 4 3 2 1 0	CHARACTER PATTERN (CG RAM DATA) 7 6 5 4 3 2 1 0
0 0 0 0 0 0 0 0 (00 HEX) ACCESSED BY ASCII CODE 00 (HEX)	0 0 0 0 0 0	X X X 0 0 0 0 0
	0 0 0 0 0 1	X X X 0 0 0 0 0
	0 0 0 0 1 0	X X X 0 0 0 0 0
	0 0 0 0 1 1	X X X 1 1 1 0 0
	0 0 0 1 0 0	X X X 1 1 1 1 1
	0 0 0 1 0 1	X X X 0 0 0 0 0
	0 0 0 1 1 0	X X X 0 0 0 0 0
	0 0 0 1 1 1	X X X 0 0 0 0 0

NOSE SECTION

CHARACTER CODES (DD RAM DATA) 7 6 5 4 3 2 1 0	CG RAM ADDRESS 5 4 3 2 1 0	CHARACTER PATTERN (CG RAM DATA) 7 6 5 4 3 2 1 0
0 0 0 0 0 0 0 1 (01 HEX) ACCESSED BY ASCII CODE 01 (HEX)	0 0 1 0 0 0	X X X 1 0 0 0 0
	0 0 1 0 0 1	X X X 0 1 1 0 0
	0 0 1 0 1 0	X X X 0 0 1 1 0
	0 0 1 0 1 1	X X X 1 1 1 1 1
	0 0 1 1 0 0	X X X 1 1 1 1 1
	0 0 1 1 0 1	X X X 0 0 1 1 0
	0 0 1 1 1 0	X X X 0 1 1 0 0
	0 0 1 1 1 1	X X X 1 0 0 0 0

BODY SECTION

CHARACTER CODES (DD RAM DATA) 7 6 5 4 3 2 1 0	CG RAM ADDRESS 5 4 3 2 1 0	CHARACTER PATTERN (CG RAM DATA) 7 6 5 4 3 2 1 0
0 0 0 0 0 0 1 0 (02 HEX) ACCESSED BY ASCII CODE 02 (HEX)	0 1 0 0 0 0	X X X 1 1 0 0 0
	0 1 0 0 0 1	X X X 1 1 0 0 0
	0 1 0 0 1 0	X X X 1 1 0 0 0
	0 1 0 0 1 1	X X X 1 1 1 1 1
	0 1 0 1 0 0	X X X 1 1 1 1 1
	0 1 0 1 0 1	X X X 0 0 0 0 0
	0 1 0 1 1 0	X X X 0 0 0 0 0
	0 1 0 1 1 1	X X X 0 0 0 0 0

TAIL SECTION

COMPLETE AIRPLANE

ORDERING INFORMATION

The following items are available from Simple Design Implementations (SDI), P.O. Box 9303, Forestville, CT 06010 (203) 582-8526: Experimenter's kit (contains 16×1 OPTREX LCD module, programmed MC68705P3, contrast-control potentiometer, PC board, IC socket, software listings, schematic, and instructions), $29.95 + $3 S/H; Same experimenter's kit with 40×1 display, $39.95 + $3 S/H; Programmed MC68705P3 and instructions, $15.95 + $2.50 S/H.

shows the 24 data bytes that must be written to CG RAM to form the three user-defined characters that form the airplane.

Displaying actual data

Once the display has been properly initialized, displaying data is as simple as writing out the proper ASCII codes with a series of Data Write operations. Remember, the "SET DD RAM ADDRESS" command must precede the data operations to ensure that the data goes to DD RAM and not CG RAM. Similarly, data writes to CG RAM must be preceded by a "SET CG RAM ADDRESS" command. For example, the routine in Listing 6 will display the message "PLANE" (assuming that the user-defined Character Generator RAM is set up as defined in Listing 5).

Next month

Due to space limitations, that's all we have room for this month. Next month we will continue our discussion on LCD modules, and cover some of the different kinds of interfaces, including hardware, microcontroller, and microprocessor. **R-E**