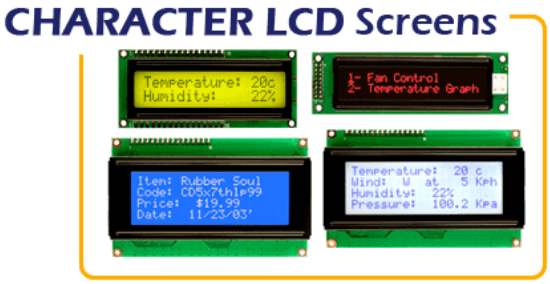
Alphanumeric LCD Displays

55:036

Embedded Systems and Systems Software

Alphaneumeric LCDs



The HD44780 LCD Controller

- Most low cost Character-based LCD modules use the Hitachi HD44780 controller chip
  - Typically 8, 16, 20, 24 or 40 characters/line
  - 1, 2, or 4 lines
  - Handles up to 2^7 = 128 total characters/display
- Standard 14-pin interface

LCD Pinouts

Pin number	Symbol	Level	I/O	Function
1	Vss	-	-	Power supply (GND)
2	Vcc	-	-	Power supply (+5V)
3	Vee	-	-	Contrast adjust
4	RS	0/1	I	0 = Instruction input 1 = Data input
5	R/W	0/1	I	0 = Write to LCD module 1 = Read from LCD module
6	E	1, 1->0	I	Enable signal
7	DB0	0/1	I/O	Data bus line 0 (LSB)
8	DB1	0/1	I/O	Data bus line 1
9	DB2	0/1	I/O	Data bus line 2
10	DB3	0/1	I/O	Data bus line 3
11	DB4	0/1	I/O	Data bus line 4
12	DB5	0/1	I/O	Data bus line 5
13	DB6	0/1	I/O	Data bus line 6
14	DB7	0/1	I/O	Data bus line 7 (MSB)

LCD Interface Modes

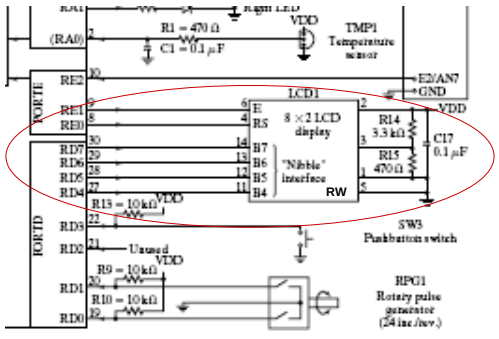
- 8 bit mode
  - Uses all 8 data lines DB0-DB7
  - Data transferred to LCD in byte units
  - Interface requires 10 (sometimes 11) I/O pins of microcontroller (DB0-DB7, RS, E) (sometimes R/W)
- 4-bit mode
  - 4-bit (nibble) data transfer
  - Doesn't use DB0-DB3
  - Each byte transfer is done in two steps: high order nibble, then low order nibble
  - Interface requires only 6 (sometimes 7) I/O pins of microcontroller (DD4-DB7, RS, E) (sometimes R/W)

LCD Interface Modes

- 8 bit mode
  - Uses all 8 data lines DB0-DB7
  - Data transferred to LCD in byte units
  - Interface requires 10 (sometimes 11) I/O pins of microcontroller (DB0-DB7, RS, E) (sometimes R/W)
- 4-bit mode
  - 4-bit (nibble) data transfer
  - Doesn't use DB0-DB3
  - Each byte transfer is done in two steps: high order nibble, then low order nibble
  - Interface requires only 6 I/O (sometimes 7) pins of microcontroller (DD4-DB7, RS, E) (sometimes R/W)

QwikFlash uses 4-bit interface mode

QwikFlash LCD Configuration

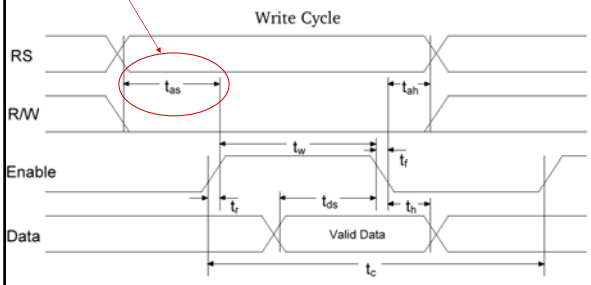


LCD Control: RS, E, R/W

- RS (Register Select)
  - When low: data transferred to (from) device is treated as commands (status)
  - When high: data transferred to/from device is characters.
- R/W (Read/Write)
  - Controls data transfer direction
    - low to write to LCD
    - high to read frrom LCD
    - On the QwikFlash, this pin is wired to ground—i.e. can't read from LCD
- E (Enable) Input
  - Initiates data transfer
  - For write, data transferred to LCD on high to low transition
  - For read, data available following low to high transition

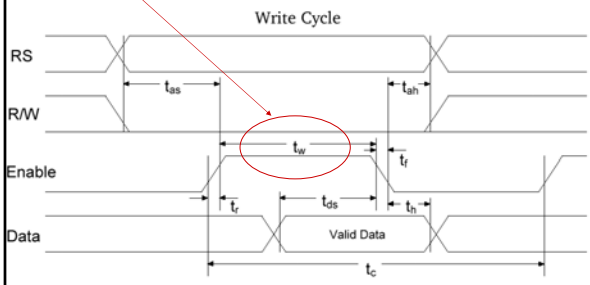
LCD Timing

At least 40 nsec



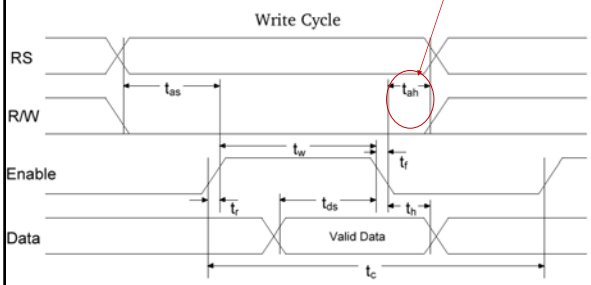
LCD Timing

At least 230 nsec

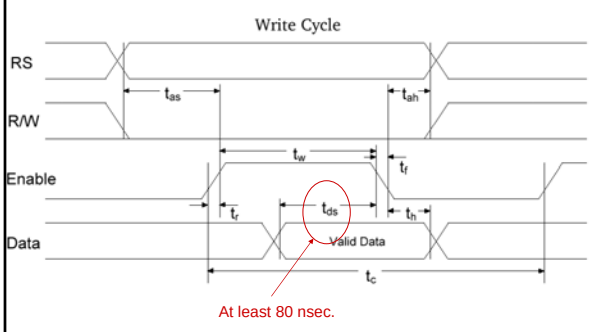


LCD Timing

At least 10 nsec



LCD Timing



LCD Timing Parameters

Write-Cycle	V_{DD}	2.7 - 4.5 V [2]	4.5 - 5.5 V [2]		2.7 - 4.5 V [2]	4.5 - 5.5 V [2]	
Parameter	**Symbol**	**Min[1]**		**Typ[1]**	**Max[1]**		**Unit**
Enable Cycle Time	t_c	1000	500	-	-	-	ns
Enable Pulse Width (High)	t_w	450	230	-	-	-	ns
Enable Rise/Fall Time	t_r, t_f	-	-	-	25	20	ns
Address Setup Time	t_{as}	60	40	-	-	-	ns
Address Hold Time	t_{ah}	20	10	-	-	-	ns
Data Setup Time	t_{ds}	195	80	-	-	-	ns
Data Hold Time	t_h	10	10	-	-	-	ns

LCD Commands

Command	Binary								Hex
	D7	D6	D5	D4	D3	D2	D1	D0	
Clear Display	0	0	0	0	0	0	0	1	01
Display & Cursor Home	0	0	0	0	0	0	1	x	02 or 03
Character Entry Mode	0	0	0	0	0	1	I/D	S	04 to 07
Display On/Off & Cursor	0	0	0	0	1	D	U	B	08 to 0F
Display/Cursor Shift	0	0	0	1	D/C	R/L	x	x	10 to 1F
Function Set	0	0	1	8/4	2/1	10/7	x	x	20 to 3F
Set CGRAM Address	0	1	A	A	A	A	A	A	40 to 7F
Set Display Address	1	A	A	A	A	A	A	A	80 to FF

I/D: 1=Increment*, 0=Decrement
R/L: 1=Right shift, 0=Left shift
S: 1=Display shift on, 0=Display shift off*
8/4: 1=8 bit interface*, 0=4 bit interface
D: 1=Display On, 0=Display Off*
2/1: 1=2 line mode, 0=1 line mode*
U: 1=Cursor underline on, 0=Underline off*
10/7: 1=5x10 dot format, 0=5x7 dot format*
B: 1=Cursor blink on, 0=Cursor blink off*
D/C: 1=Display shift, 0=Cursor move
x = Don't care
* = Initialisation settings

LCD Command Execution Times

Instruction	Time (Max)
Clear Display	82μs to 1·64ms
Display & Cursor Home	40μs to 1·64ms
Character Entry Mode	40μs
Display On/Off & Cursor	40μs
Display/Cursor Shift	40μs
Function Set	40μs
Set CGRAM Address	40μs
Set Display Address	40μs
Write Data	40μs
Read Data	40μs
Read Status	1μs

Command Execution Times--Continued

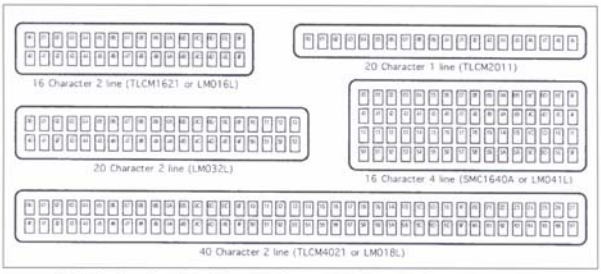
- Most HD44780 commands take 40 **microseconds** to execute
- **Clear Display** and **Cursor Home** commands can take much longer (as much at several milliseconds)
- Can't issue another command until previous one has finished
- Two options
  - Busy-wait:
    - After issuing a command, continuously monitor HD44780 status until device is not **busy**
    - Can't do this with QwikFlash, since we can't read from the HD44780
  - Insert a 40 microsecond (or, in some cases, much longer) delay between commands

Command Execution Times--Continued

- Most HD44780 commands take 40 **microseconds** to execute
- **Clear Display** and **Cursor Home** commands can take much longer (as much at several milliseconds)
- Can't issue another command until previous one has finished
- Two options
  - Busy-wait:
    - After issuing a command, continuously monitor HD44780 status until device is not **busy**
    - Can't do this with QwikFlash, since we can't read from the HD44780
  - Insert a 40 microsecond (or, in some cases, much longer) delay between commands

This is our only option with QwikFlash, since R/W is hardwired to zero.

LCD Cursor Position Addresses



Note: The HD44780 always maintains an internal buffer of 128 character positions. For a given LCD, not all of these are displayable. You can still write characters to these positions, but they won't appear on the screen

More about Timing

- Timing for writing to the LCD is not critical (as long as setup and hold times are observed:
  - Drive E high (pin RE1)
  - Send upper "nibble" of data to Port D (RD7-RD4)
  - Drive E low (pin RE1)
  - Drive E high (pin RE1)
  - Send lower nibble of data to Port D (RD7-RD4)
  - Drive E low (pin RE1)
- Note: Throughout this process RS must be properly set (low for commands; high for characters)

More about Timing

- Timing for writing to the LCD is not critical (as long as setup and hold times are observed:
  - Drive E high (pin RE1)
  - Send upper "nibble" of data to Port D (RD7-RD4)
  - Drive E low (pin RE1)
  - Drive E high (pin RE1)
  - Send lower nibble of data to Port D (RD7-RD4)
  - Drive E low (pin RE1)
- Note: Throughout this process RS must be properly set (low for commands; high for characters)

For very fast instruction clock, it may be necessary to insert a nop to satisfy minimum t_w and t_c

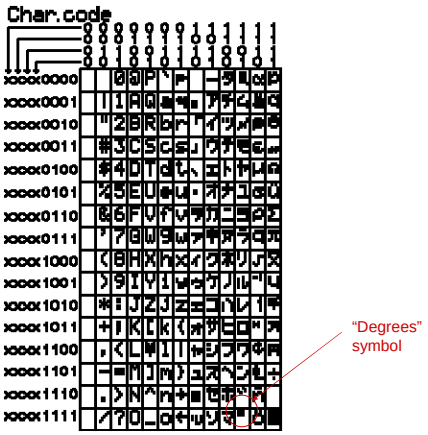
Writing to the LCD--Example

```
;This example code writes the byte stored in SFR TABLAT to the LCD
bsf  PORTE,RE1          ;Drive E pin high
movff  TABLAT,PORTD     ;Send upper nibble
bcf  PORTE,RE1          ;Drive E pin low so LCD will
                         ;accept nibble
bsf  PORTE,RE1          ;Drive E pin high again
swapf  TABLAT,W         ;Swap nibbles
movwf  PORTD            ;Write lower nibble
bcf  PORTE,RE1          ;Drive E pin low so LCD will
                        ;process byte
```

The Cursor Position Map for the QwikFlash LCD

0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7
0x40	0x41	0x42	0x43	0x44	0x45	0x46	0x47

HD44780 Character Codes



To “write” a string of characters to the LCD

- Drive RS low (command mode)
- Send a **Set Display Address** command to the LCD to establish initial display position
- Drive RS high (character mode)
- Send first character to LCD
- Send second character to LCD
- etc.

To “write” a string of characters to the LCD

- Drive RS low (command mode)
- Send a **Set Display Address** command to the LCD to establish initial display position
- Drive RS high (character mode)
- Send first character to LCD
- Send second character to LCD
- etc.

The display position will automatically increment (or decrement) depending upon how you configured the LCD with the **Character Entry** Command

To “write” a string of characters to the LCD

- Drive RS low (command mode)
- Send a **Set Display Address** command to the LCD to establish initial display position
- Drive RS high (character mode)
- Send first character to LCD
- Send second character to LCD
- etc.

Note: Need to wait 40 microseconds between each character

The display position will automatically increment (or decrement) depending upon how you configured the LCD with the **Character Entry** Command

Sending Characters to the LCD

```
bcf     PORTE,RE0     ;Drive RS pin low for cursor positioning command
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x80,PORTD    ;Send upper nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x00, PORTD   ; Send lower nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE0     ;Drive RS pin high for displayable characters
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x40,PORTD    ;Send upper nibble of Character ‘H’
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x80, PORTD   ; Send lower nibble of Character ‘H’
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x60,PORTD    ;Send upper nibble of character ‘i’
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x90, PORTD   ; Send lower nibble of character ‘i’
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
```

Sending Characters to the LCD

```
bcf     PORTE,RE0     ;Drive RS pin low for cursor positioning command
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x80,PORTD    ;Send upper nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x00, PORTD   ; Send lower nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE0     ;Drive RS pin high for displayable characters
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x40,PORTD    ;Send upper nibble of Character ‘H’
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x80, PORTD   ; Send lower nibble of Character ‘H’
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x60,PORTD    ;Send upper nibble of character ‘i’
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,1       ;Drive E pin high again
MOVLF   0x90, PORTD   ; Send lower nibble of character ‘i’
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
```

set command mode

Sending Characters to the LCD

```
bcf     PORTE,RE0     ;Drive RS pin low for cursor positioning command
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x80,PORTD    ;Send upper nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x00, PORTD   ; Send lower nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE0     ;Drive RS pin high for displayable characters
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x40,PORTD    ;Send upper nibble of Character 'H'
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x80, PORTD   ; Send lower nibble of Character 'H'
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x60,PORTD    ;Send upper nibble of character 'i'
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x90, PORTD   ; Send lower nibble of character 'i'
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
```

Send set address command to LCD B'10000000'

Set display address

to 0

Sending Characters to the LCD

```
bcf     PORTE,RE0     ;Drive RS pin low for cursor positioning command
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x80,PORTD    ;Send upper nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x00, PORTD   ; Send lower nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE0     ;Drive RS pin high for displayable characters
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x40,PORTD    ;Send upper nibble of Character 'H'
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x80, PORTD   ; Send lower nibble of Character 'H'
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x60,PORTD    ;Send upper nibble of character 'i'
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x90, PORTD   ; Send lower nibble of character 'i'
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
```

Sending Characters to the LCD

```
bcf     PORTE,RE0     ;Drive RS pin low for cursor positioning command
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x80,PORTD    ;Send upper nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x00, PORTD   ; Send lower nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE0     ;Drive RS pin high for displayable characters
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x40,PORTD    ;Send upper nibble of Character 'H'
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x80, PORTD   ; Send lower nibble of Character 'H'
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x60,PORTD    ;Send upper nibble of character 'i'
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x90, PORTD   ; Send lower nibble of character 'i'
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
```

switch to character mode

Sending Characters to the LCD

```
bcf     PORTE,RE0     ;Drive RS pin low for cursor positioning command
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x80,PORTD    ;Send upper nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x00, PORTD   ; Send lower nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE0     ;Drive RS pin high for displayable characters
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x40,PORTD    ;Send upper nibble of Character 'H'
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x80, PORTD   ; Send lower nibble of Character 'H'
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x60,PORTD    ;Send upper nibble of character 'i'
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x90, PORTD   ; Send lower nibble of character 'i'
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
```

Send 'H' (0x48) to the LCD

Sending Characters to the LCD

```
bcf     PORTE,RE0     ;Drive RS pin low for cursor positioning command
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x80,PORTD    ;Send upper nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x00, PORTD   ; Send lower nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE0     ;Drive RS pin high for displayable characters
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x40,PORTD    ;Send upper nibble of Character 'H'
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x80, PORTD   ; Send lower nibble of Character 'H'
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x60,PORTD    ;Send upper nibble of character 'i'
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x90, PORTD   ; Send lower nibble of character 'i'
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
```

Sending Characters to the LCD

```
bcf     PORTE,RE0     ;Drive RS pin low for cursor positioning command
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x80,PORTD    ;Send upper nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x00, PORTD   ; Send lower nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE0     ;Drive RS pin high for displayable characters
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x40,PORTD    ;Send upper nibble of Character 'H'
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x80, PORTD   ; Send lower nibble of Character 'H'
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x60,PORTD    ;Send upper nibble of character 'i'
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x90, PORTD   ; Send lower nibble of character 'i'
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
```

Send 'i' (0x69) to the LCD

Sending Characters to the LCD

```
bcf     PORTE,RE0     ;Drive RS pin low for cursor positioning command
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x80,PORTD    ;Send upper nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x00, PORTD   ; Send lower nibble of set address command
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE0     ;Drive RS pin high for displayable characters
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x40,PORTD    ;Send upper nibble of Character 'H'
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x80, PORTD   ; Send lower nibble of Character 'H'
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
bsf     PORTE,RE1     ;Drive E pin high
MOVLF   0x60,PORTD    ;Send upper nibble of character 'i'
bcf     PORTE,RE1     ;Drive E pin low so LCD will accept nibble
bsf     PORTE,RE1     ;Drive E pin high again
MOVLF   0x90, PORTD   ; Send lower nibble of character 'i'
bcf     PORTE,RE1     ;Drive E pin low so LCD will process byte
rcall   T40           ;Wait 40 usec
```

The DisplayC Subroutine

- Displays a constant character string (stored in **program memory**)
  - First byte of string contains the cursor-positioning command
  - Last byte of string is 0x00
  - Intervening bytes contain codes for characters to be displayed
- E.g.:

  MYSTR db "\x80Hello\x00"

  or, equivalently

  MYSTR db 0x80,0x48, 0x65,0x6c,0x6c,0x6f,0x00

Using TBLPTR to Access a Constant String in Program Memory

- Must make TBLPTRH:TBLPTRL "point at" the string:

```
MOVLF high MYSTR, TBLPTRH
MOVLF low MYSTR, TBLPTRL
```

Now can read bytes from the string:

```
tblrd*    ; reads byte pointed to by TBLPTR
          ; into TABLAT
or
tblrd+*   ; increments TBLPTR and reads byte pointed to by
          ; into TABLAT
```

Note: can also do: tblptr*+, tblptr*-

The DisplayC Subroutine

```
DisplayC
    bcf  PORTE,RE0      ;Drive RS pin low for cursor-positioning command
    tblrd*              ;Get byte from string into TABLAT
    movf  TABLAT,F      ;Check for leading zero byte
    IF_  .Z.
      tblrd+*           ;If zero, get next byte
    ENDIF_
    REPEAT_
      bsf  PORTE,RE1        ;Drive E pin high
      movff  TABLAT,PORTD   ;Send upper nibble
      bcf  PORTE,RE1        ;Drive E pin low so LCD will accept nibble
      bsf  PORTE,RE1        ;Drive E pin high again
      swapf  TABLAT,W       ;Swap nibbles
      movwf  PORTD          ;Write lower nibble
      bcf  PORTE,RE1        ;Drive E pin low so LCD will process byte
      rcall  T40            ;Wait 40 usec
      bsf  PORTE,RE0        ;Drive RS pin high for displayable characters
      tblrd+*               ;Increment pointer, then get next byte
      movf  TABLAT,F        ;Is it zero?
    UNTIL_  .Z.
  return
```

Before subroutine is called, TBLPTRH:TBLPTRL must contain the address of the first byte of the string

```
DisplayC
    bcf  PORTE,RE0      ;Drive RS pin low for cursor-positioning command
    tblrd*              ;Get byte from string into TABLAT
    movf  TABLAT,F      ;Check for leading zero byte
    IF_  .Z.
      tblrd+*           ;If zero, get next byte
    ENDIF_
    REPEAT_
      bsf  PORTE,RE1        ;Drive E pin high
      movff  TABLAT,PORTD   ;Send upper nibble
      bcf  PORTE,RE1        ;Drive E pin low so LCD will accept nibble
      bsf  PORTE,RE1        ;Drive E pin high again
      swapf  TABLAT,W       ;Swap nibbles
      movwf  PORTD          ;Write lower nibble
      bcf  PORTE,RE1        ;Drive E pin low so LCD will process byte
      rcall  T40            ;Wait 40 usec
      bsf  PORTE,RE0        ;Drive RS pin high for displayable characters
      tblrd+*               ;Increment pointer, then get next byte
      movf  TABLAT,F        ;Is it zero?
    UNTIL_  .Z.
  return
```

Set command mode

```
DisplayC
    bcf  PORTE,RE0      ;Drive RS pin low for cursor-positioning command
    tblrd*              ;Get byte from string into TABLAT
    movf  TABLAT,F      ;Check for leading zero byte
    IF_  .Z.
      tblrd+*           ;If zero, get next byte
    ENDIF_
    REPEAT_
      bsf  PORTE,RE1        ;Drive E pin high
      movff  TABLAT,PORTD   ;Send upper nibble
      bcf  PORTE,RE1        ;Drive E pin low so LCD will accept nibble
      bsf  PORTE,RE1        ;Drive E pin high again
      swapf  TABLAT,W       ;Swap nibbles
      movwf  PORTD          ;Write lower nibble
      bcf  PORTE,RE1        ;Drive E pin low so LCD will process byte
      rcall  T40            ;Wait 40 usec
      bsf  PORTE,RE0        ;Drive RS pin high for displayable characters
      tblrd+*               ;Increment pointer, then get next byte
      movf  TABLAT,F        ;Is it zero?
    UNTIL_  .Z.
  return
```

Loop until 0x00 byte is encountered

```
DisplayC
      bcf  PORTE,RE0       ;Drive RS pin low for cursor-positioning command
      tblrd*               ;Get byte from string into TABLAT
      movf  TABLAT,F       ;Check for leading zero byte
      IF_  .Z.
        tblrd+*            ;If zero, get next byte
      ENDIF_
      REPEAT_
        bsf  PORTE,RE1          ;Drive E pin high
        movff  TABLAT,PORTD     ;Send upper nibble
        bcf  PORTE,RE1          ;Drive E pin low so LCD will accept nibble
        bsf  PORTE,RE1          ;Drive E pin high again
        swapf  TABLAT,W         ;Swap nibbles
        movwf  PORTD            ;Write lower nibble
        bcf  PORTE,RE1          ;Drive E pin low so LCD will process byte
        rcall  T40              ;Wait 40 usec
        bsf  PORTE,RE0          ;Drive RS pin high for displayable characters
        tblrd+*                 ;Increment pointer, then get next byte
        movf  TABLAT,F          ;Is it zero?
      UNTIL_  .Z.
      return
```

Write next byte to LCD

```
DisplayC
      bcf  PORTE,RE0       ;Drive RS pin low for cursor-positioning command
      tblrd*               ;Get byte from string into TABLAT
      movf  TABLAT,F       ;Check for leading zero byte
      IF_  .Z.
        tblrd+*            ;If zero, get next byte
      ENDIF_
      REPEAT_
        bsf  PORTE,RE1          ;Drive E pin high
        movff  TABLAT,PORTD     ;Send upper nibble
        bcf  PORTE,RE1          ;Drive E pin low so LCD will accept nibble
        bsf  PORTE,RE1          ;Drive E pin high again
        swapf  TABLAT,W         ;Swap nibbles
        movwf  PORTD            ;Write lower nibble
        bcf  PORTE,RE1          ;Drive E pin low so LCD will process byte
        rcall  T40              ;Wait 40 usec
        bsf  PORTE,RE0          ;Drive RS pin high for displayable characters
        tblrd+*                 ;Increment pointer, then get next byte
        movf  TABLAT,F          ;Is it zero?
      UNTIL_  .Z.
      return
```

After first byte, switch to character mode

```
DisplayC
      bcf  PORTE,RE0       ;Drive RS pin low for cursor-positioning command
      tblrd*               ;Get byte from string into TABLAT
      movf  TABLAT,F       ;Check for leading zero byte
      IF_  .Z.
        tblrd+*            ;If zero, get next byte
      ENDIF_
      REPEAT_
        bsf  PORTE,RE1          ;Drive E pin high
        movff  TABLAT,PORTD     ;Send upper nibble
        bcf  PORTE,RE1          ;Drive E pin low so LCD will accept nibble
        bsf  PORTE,RE1          ;Drive E pin high again
        swapf  TABLAT,W         ;Swap nibbles
        movwf  PORTD            ;Write lower nibble
        bcf  PORTE,RE1          ;Drive E pin low so LCD will process byte
        rcall  T40              ;Wait 40 usec
        bsf  PORTE,RE0          ;Drive RS pin high for displayable characters
        tblrd+*                 ;Increment pointer, then get next byte
        movf  TABLAT,F          ;Is it zero?
      UNTIL_  .Z.
      return
```

Don't worry about this. It's just here to make the subroutine work with strings that start with 0x00 (historical legacy)

The DisplayV Subroutine

- Displays a variable string stored in **data memory**
- String format is same as for DisplayC
  - First byte of string contains the cursor-positioning command
  - Last byte of string is 0x00
  - Intervening bytes contain codes for characters to be displayed

DisplayV Subroutine

```
;;;;;;; DisplayV subroutine ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; This subroutine is called with FSR0 containing the address of a variable
; display string.  It sends the bytes of the string to the LCD.  The first
; byte sets the cursor position.  The remaining bytes are displayed, beginning
; at that position.

DisplayV
    bcf  PORTE,RE0          ;Drive RS pin low for cursor positioning code
    REPEAT_
      bsf  PORTE,RE1        ;Drive E pin high
      movff  INDF0,PORTD    ;Send upper nibble
      bcf  PORTE,RE1        ;Drive E pin low so LCD will accept nibble
      bsf  PORTE,RE1        ;Drive E pin high again
      swapf  INDF0,W        ;Swap nibbles
      movwf  PORTD          ;Write lower nibble
      bcf  PORTE,RE1        ;Drive E pin low so LCD will process byte
      rcall  T40            ;Wait 40 usec
      bsf  PORTE,RE0        ;Drive RS pin high for displayable characters
      movf  PREINC0,W       ;Increment pointer, then get next byte
    UNTIL_  .Z.             ;Is it zero?
   return
```

DisplayV Subroutine

```
;;;;;;; DisplayV subroutine ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; This subroutine is called with FSR0 containing the address of a variable
; display string.  It sends the bytes of the string to the LCD.  The first
; byte sets the cursor position.  The remaining bytes are displayed, beginning
; at that position.

DisplayV
    bcf  PORTE,RE0          ;Drive RS pin low for cursor positioning code
    REPEAT_
      bsf  PORTE,RE1        ;Drive E pin high
      movff  INDF0,PORTD    ;Send upper nibble
      bcf  PORTE,RE1        ;Drive E pin low so LCD will accept nibble
      bsf  PORTE,RE1        ;Drive E pin high again
      swapf  INDF0,W        ;Swap nibbles
      movwf  PORTD          ;Write lower nibble
      bcf  PORTE,RE1        ;Drive E pin low so LCD will process byte
      rcall  T40            ;Wait 40 usec
      bsf  PORTE,RE0        ;Drive RS pin high for displayable characters
      movf  PREINC0,W       ;Increment pointer, then get next byte
    UNTIL_  .Z.             ;Is it zero?
   return
```

DisplayV Subroutine

```
;;;;;;; DisplayV subroutine ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; This subroutine is called with FSR0 containing the address of a variable
; display string.  It sends the bytes of the string to the LCD.  The first
; byte sets the cursor position.  The remaining bytes are displayed, beginning
; at that position.

DisplayV
    bcf  PORTE,RE0          ;Drive RS pin low for cursor positioning code
    REPEAT_
      bsf  PORTE,RE1        ;Drive E pin high
      movff  INDF0,PORTD    ;Send upper nibble
      bcf  PORTE,RE1        ;Drive E pin low so LCD will accept nibble
      bsf  PORTE,RE1        ;Drive E pin high again
      swapf  INDF0,W        ;Swap nibbles
      movwf  PORTD          ;Write lower nibble
      bcf  PORTE,RE1        ;Drive E pin low so LCD will process byte
      rcall  T40            ;Wait 40 usec
      bsf  PORTE,RE0        ;Drive RS pin high for displayable characters
      movf  PREINC0,W       ;Increment pointer, then get next byte
    UNTIL_  .Z.             ;Is it zero?
   return
```

loop until end of string (zero byte) is detected

DisplayV Subroutine

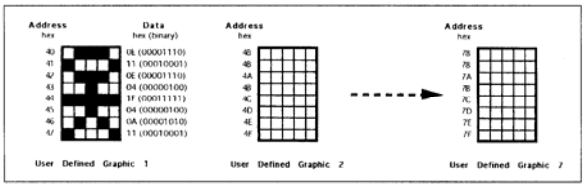
```
;;;;;;; DisplayV subroutine ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; This subroutine is called with FSR0 containing the address of a variable
; display string.  It sends the bytes of the string to the LCD.  The first
; byte sets the cursor position.  The remaining bytes are displayed, beginning
; at that position.

DisplayV
    bcf  PORTE,RE0          ;Drive RS pin low for cursor positioning code
    REPEAT_
      bsf  PORTE,RE1        ;Drive E pin high
      movff  INDF0,PORTD    ;Send upper nibble
      bcf  PORTE,RE1        ;Drive E pin low so LCD will accept nibble
      bsf  PORTE,RE1        ;Drive E pin high again
      swapf  INDF0,W        ;Swap nibbles
      movwf  PORTD          ;Write lower nibble
      bcf  PORTE,RE1        ;Drive E pin low so LCD will process byte
      rcall  T40            ;Wait 40 usec
      bsf  PORTE,RE0        ;Drive RS pin high for displayable characters
      movf  PREINC0,W       ;Increment pointer, then get next byte
    UNTIL_  .Z.             ;Is it zero?
   return
```

Note the use of indirect addressing: (FSR0 and INDF0)

User-defined characters

- HD44780 supports up to 8 user-defined characters
- Use character codes 0x01 – 0x00f
- Before using, must be defined by storing the pixel pattern in a character-generating RAM (CGRAM) on the controller chip

CGRAM Address Map



Writing to the CGRAM is done using the **Set CGRAM Address** command

User-defined Characters

- Can write a user-defined character to the CGRAM using the DisplayC subroutine
- See example 7.5 in the text

And finally, one last mystery

- The HD44780 has some initialization quirks.
- The recommended initialization sequence, following power-up is:
  - wait for 0.1 seconds
  - set the device to 8-bit mode three times
  - set device to 4-bit mode
  - complete additional device configuration

Initialization, Continued

- The QwikFlash LCD can be properly initialized by writing the following command string to the controller, **a nibble at a time** (Must be in Command mode (RS=0); also, must wait .01 seconds first):

LCDstr db 0x33,0x32,0x28,0x01,0x0c,0x06,0x00

Function set
8 bit mode

Initialization, Continued

- The QwikFlash LCD can be properly initialized by writing the following command string to the controller, **a nibble at a time** (Must be in Command mode (RS=0); also, must wait .01 seconds first):

LCDstr db 0x33,0x32,0x28,0x01,0x0c,0x06,0x00

Function set
8 bit mode

Initialization, Continued

- The QwikFlash LCD can be properly initialized by writing the following command string to the controller, **a nibble at a time** (Must be in Command mode (RS=0); also, must wait .01 seconds first):

LCDstr db 0x33,0x32,0x28,0x01,0x0c,0x06,0x00

Function set
8 bit mode

Initialization, Continued

- The QwikFlash LCD can be properly initialized by writing the following command string to the controller, **a nibble at a time** (Must be in Command mode (RS=0); also, must wait .01 seconds first):

LCDstr db 0x33,0x32,0x28,0x01,0x0c,0x06,0x00

Function set
4 bit mode

Initialization, Continued

- The QwikFlash LCD can be properly initialized by writing the following command string to the controller, **a nibble at a time** (Must be in Command mode (RS=0); also, must wait .01 seconds first):

```
LCDstr  db  0x33,0x32,0x28,0x01,0x0c,0x06,0x00
```

Function set
4 bit mode,
two rows,
5x7 characters

Initialization, Continued

- The QwikFlash LCD can be properly initialized by writing the following command string to the controller, **a nibble at a time** (Must be in Command mode (RS=0); also, must wait .01 seconds first):

```
LCDstr  db  0x33,0x32,0x28,0x01,0x0c,0x06,0x00
```

Clear display

Initialization, Continued

- The QwikFlash LCD can be properly initialized by writing the following command string to the controller, **a nibble at a time** (Must be in Command mode (RS=0); also, must wait .01 seconds first):

```
LCDstr  db  0x33,0x32,0x28,0x01,0x0c,0x06,0x00
```

Display on
cursor underline off
cursor blink off

Initialization, Continued

- The QwikFlash LCD can be properly initialized by writing the following command string to the controller, **a nibble at a time** (Must be in Command mode (RS=0); also, must wait .01 seconds first):

```
LCDstr  db  0x33,0x32,0x28,0x01,0x0c,0x06,0x00
```

Display shift off,
Address increment

This causes display address (cursor position) to be automatically incremented following each character write. Can also set controller to automatically *decrement* the address