EE 433/533
Embedded and Portable Computing Systems

Alphanumeric LCDs

Alphanumeric vs. Graphical

- An *alphanumeric LCD* has fixed character cells that are written to using an ASCII code.
- A *graphical LCD* allows individual pixels to be turned on or off.
- In this course only alphanumeric LCDs will be examined.

Hitachi HD44780

- The HD44780 is an LCD controller chip which is usually purchased already attached to the LCD display.
- Almost all alphanumeric LCD displays use the HD44780 interface signals, whether they actually use an HD44780 or not.
- The interface comes in 8-bit and 4-bit varieties. The 4-bit is used on the QwikFlash board.

HD44780 Timing

- Most of the timing requirements of the HD44780 are hard to violate with the PIC18F452 running at maximum clock.
- The main exception is:

  There needs to be a 40 microsecond delay between characters sent to the LCD display.

LCD Signals

- Clock: *E* (values sampled on falling edge)
- Register select: *RS* (0 control, 1 data)
- Data bits:

  4-bit "nibble" interface: *DB7-DB4*

  8-bit interface: *DB7-DB0*

  (highest numbered is most significant)

Reset

- Both the PIC18F452 and the HD44780 will come out of reset when power is turned on.
- If the PIC18F452 is slightly faster at coming out of reset than the HD44780, the later may miss instructions sent to it by the PIC.
- A delay of at least 0.1 seconds after PIC reset is usually enough to guarantee that the HD is also running.

HD44780 Initialization

- All initialization data is sent to the control register, so RS should be left at 0 throughout.
- Every time a new initialization nibble is sent, the following pattern is used:
  1) Make E = 1
  2) Output nibble
  3) Make E = 0
  4) Wait at least 40 ms

4-bit Interface Setup

- The same HD44780 chip is used for 4-bit and 8-bit interfaces. To tell the HD44780 to use the 4-bit interface, it needs to see the following pattern on the nibble interface:
  1) 0x3
  2) 0x3
  3) 0x3
  4) 0x2

Number of Lines Setup

- The same HD44780 chip is used for displays with differing numbers of lines. For a 2-line display, the nibble interface should see:
  1) 0x2
  2) 0x8

Clearing the Display

- The data memory of the HD44780 powers up with random data. To make all data characters equal to an ASCII “space” character:
  1) 0x0
  2) 0x1

Turning on Display

- The HD44780 powers up with the display turned off (nothing displayed, no matter what is placed in data memory). To turn the display on with the option of not showing a cursor:
  1) 0x0
  2) 0xC

Position Auto-increment

- The HD44780 can either auto-increment the character insertion point one to the right or push all existing characters to the left (stock ticker type display). To choose auto-increment:
  1) 0x0
  2) 0x6

Character Entry

- Now that the display is initialized, the RS signal is used as follows:

  RS = 0: Each pair of nibbles forms a *cursor position code* byte.

  RS = 1: Each pair of nibbles forms an ASCII code byte.

QwikFlash Position Codes

- An 8X2 LCD controlled by an HD44780 (like the QwikFlash LCD) uses the following positioning codes:

  Line 1: 0x80 – 0x87 (left to right)

  Line 2: 0xC0 – 0xC7 (left to right)

Define Byte Directive

- Entering a table into program memory can be done with the *db* directive:

```
LCDstr   db   0x33, 0x32, 0x28, 0x01, 0x0C, 0x06, 0x00
```

  The label *LCDstr* is assigned the program memory address of the first byte.

TBLPTR Setup Macro

- To setup TBLPTR to point to a string define a macro called POINT:

```
POINT   macro    strname
        MOVLF    high strname, TBLPTRH
        MOVLF    low strname, TBLPTRL
        endm
```

  Note that MOVLF is itself a macro.

TBLPTR Setup Macro

- To invoke the macro use:

```
POINT    LCDstr
```

  If LCDstr happened to be at program address 0x1234, then the values in the TBLPTR registers will now be:

  TBLPTRH = 0x12

  TBLPTRL = 0x34

QwikFlash LCD Ports

- The following electrical connections exist on the QwikFlash board:
  1) PIC pin RE0 to LCD pin RS
  2) PIC pin RE1 to LCD pin E
  3) PIC pins RD4-RD7 to LCD pins DB4-DB7 (in that order)

LCD Initialization Code

<wait at least 0.1 second>

```
bcf     PORTE, 0 ; make RS=0 (RE0 = 0)
POINT   LCDstr   ; set up TBLPTR
tblrd*           ; get first byte (0x33)
```

<output nibbles from table up to, but not including, the 0x00 end-of-table marker byte>

Body of Table Loop

```
; Note: assume that RD3-RD0 are inputs
bsf    PORTE, 1 ; make E=1 (RE1 = 1)
movff TABLAT, PORTD ; output nibble
bcf    PORTE, 1 ; make E=0 (RE1 = 0)
```

<wait at least 40 microseconds>

```
swapf    TABLAT, W    ; swap nibbles
```

Body of Table Loop

```
bsf    PORTE, 1 ; make E=1 (RE1 = 1)
movwf    PORTD    ; output other nibble
bcf    PORTE, 1 ; make E=0 (RE1 = 0)
```

<wait at least 40 microseconds>

```
tblrd+*    ; increment and get next table byte
```

<if byte is 0x00 stop, else go to loop top>

40 Microsecond Wait

- Since the only requirement for the delay is that it is *at least* 40 microseconds, a 10 millisecond is more than enough and the extra delay is not noticeable to the user.
- If the "LoopTime" subroutine has already been written for 10 millisecond delays in the main loop and Timer 0 has already been initialized, then the 40 μs delay can simply be:

```
call LoopTime
```

Using "db" with Characters

- The *db* directive can also be supplied characters as well as hex values:

```
DispStr  db    "ABC"
```

This places 0x41, 0x42, 0x43 (the ASCII codes for A, B, and C in program memory)

Hex Code Escapes in "db"

- Hex code can be placed within a *db* character string using the \ escape character:

```
DispStr  db    "\x82ABC\x00"
```

This places 0x82, 0x41, 0x42, 0x43, 0x00 in program memory at address DispStr.