

Controlling a LCD with a PIC

Interfacing a Matrix Orbital parallel LCD with a Microchip PIC

Keywords: LCD, PIC, Matrix Orbital, Microchip, Parallel, Interface

Author: Adam Young

Date: 2007-03-29

Executive Summary

Many times a Liquid Crystal Display (LCD) interface is needed to display information. There are multiple common protocols that are used for LCDs. This application note will show how to interface a typical parallel protocol LCD and a Programmable Integrated Circuit (PIC).

Objective

This application note will explain how to control the Matrix Orbital MOP-AL082B-BYFY LCD with a Microchip dsPIC30F4012 chip.

Introduction

Matrix Orbital is a provider of inexpensive Liquid Crystal Displays (LCD) and Microchip is a provider of inexpensive Programmable Integrated Circuits (PIC). Together, the devices can be used to create a simple and inexpensive information display for a wide range of projects. This application note will describe how to interface the Matrix Orbital MOP-AL082B-BYFY with a Microchip dsPIC30F4012 chip by using the free MPLAB IDE and free dsPIC Peripheral Library provided by Microchip.

Hardware

The Matrix Orbital MOP-AL082B-BYFY LCD is an 8 character, 2-line display. This display utilizes a parallel interface for communications. The parallel interface has 8 data lines and also separate lines for Register Select (RS), Read/Write (R/W), and Enable (E). This means that 11 connections will have to be made between the LCD and the PIC. For this interface to function, all the bits are set correctly except the enable bit, which is kept high. When the other bits are set, the enable bit is toggled low so that the LCD knows it is time to read the bits.

The Microchip dsPIC30F4012 has 20 I/O pins. This makes it sufficient for communications with the LCD still allows for 9 more connections to be made with other possible devices in a project.

Figure 1 outlines the connections that will need to be made between the LCD and the PIC.

LCD Pin	LCD Symbol	PIC Pin	PIC Symbol
4	RS	22	RE4
5	R/W	12	RC14
6	E	21	RE5
7	DB0	14	RD1
8	DB1	3	RB1
9	DB2	4	RB2
10	DB3	5	RB3
11	DB4	6	RB4
12	DB5	7	RB5
13	DB6	23	RE3
14	DB7	11	RC13

Figure 1: LCD and PIC interconnections

Software

Microchip offers a peripheral library for download (see reference #2 for download location) that contains many helpful tools to successfully interface the LCD. One of the peripheral modules that is included is the XLCD module. This is a module meant for interfacing a different LCD, the P-tec PCOG1602B. For this reason, the XLCD module will not operate the Matrix Orbital LCD defined in this application note.

Setup MPLAB Project

Create a new project in MPLAB and add to the project the following files:

- Peripheral Library/src/peripheral/include/xlcd.h
- Peripheral Library/src/peripheral/src/pmc/xlcd/*

Change PIC Requirements

Open each of the files that were added to the project in step 1 and change the following lines from:

```
#if defined(__dsPIC30F5011__) || defined(__dsPIC30F5013__) || defined(__dsPIC30F6010__) || \
defined(__dsPIC30F6011__) || defined(__dsPIC30F6012__) || defined(__dsPIC30F6013__) || \
defined(__dsPIC30F6014__) || defined(__dsPIC30F6010A__) || defined(__dsPIC30F6011A__) || \
defined(__dsPIC30F6012A__) || defined(__dsPIC30F6013A__) || defined(__dsPIC30F6014A__) || \
defined(__dsPIC30F5016__) || defined(__dsPIC30F6015__)
```

To:

```
#if defined(__dsPIC30F4012__)
```

This will tell the compiler that the code between the #if and #endif tags should only be used for the dsPIC30F4012. This is also a good time to edit the headers of these files to reflect the Matrix Orbital LCD since these files will no longer work for the P-tec PCOG1602B LCD.

Edit PIC/LCD Pin Layout

Open the file xlcd.h and edit the PORT and TRIS definition lines to match the pin layout that is outlined in Figure 1. The top portion of the file, which includes all of the changes, can be seen in Figure 2. Changed values are marked in red.

```
/*
Header for XLCD module library functions for
Matrix Orbital MOP-AL082B-BYFY LCD controller
*/

#ifndef _XLCD_H
#define _XLCD_H

/*External LCD functions are only defined for the following devices */
#if defined(__dsPIC30F4012__)

/* uncomment the following line if 8 bit interface is being used */
#define EIGHT_BIT_INTERFACE

/* #defines of the data pins and the corresponding tris pins */
#define DATA_PIN_7 PORTCbits.RC13
#define DATA_PIN_6 PORTEbits.RE3
#define DATA_PIN_5 PORTBbits.RB5
#define DATA_PIN_4 PORTBbits.RB4

#ifdef EIGHT_BIT_INTERFACE
#define DATA_PIN_3 PORTBbits.RB3
#define DATA_PIN_2 PORTBbits.RB2
#define DATA_PIN_1 PORTBbits.RB1
#define DATA_PIN_0 PORTDbits.RD1
#endif

#define TRIS_DATA_PIN_7 TRISCbits.TRISC13
#define TRIS_DATA_PIN_6 TRISEbits.TRISE3
#define TRIS_DATA_PIN_5 TRISBbits.TRISB5
#define TRIS_DATA_PIN_4 TRISBbits.TRISB4
```

```

#ifdef EIGHT_BIT_INTERFACE
#define TRIS_DATA_PIN_3    TRISBbits.TRISB3
#define TRIS_DATA_PIN_2    TRISBbits.TRISB2
#define TRIS_DATA_PIN_1    TRISBbits.TRISB1
#define TRIS_DATA_PIN_0    TRISBbits.TRISD1
#endif

/* #defines of the control pins and the corresponding tris pins */
#define E_PIN              PORTEbits.RE5      /* PORT for E */
#define RW_PIN             PORTCbits.RC14     /* PORT for RW */
#define RS_PIN             PORTEbits.RE4     /* PORT for RS */

#define TRIS_E             TRISEbits.TRISE5   /* TRIS for E */
#define TRIS_RW           TRISCbits.TRISC14  /* TRIS for RW */
#define TRIS_RS           TRISEbits.TRISE4   /* TRIS for RS */

```

Figure 2: Altered portion of `xlcd.h` with changes marked in red.

Edit Initialization Commands

The Matrix Orbital LCD and the P-tec LCD use the same type of parallel interface to send and receive data but the initialization sequences the LCDs use are completely different. Figure 3 shows the initialization sequence that will need to be implemented for the Matrix Orbital LCD.

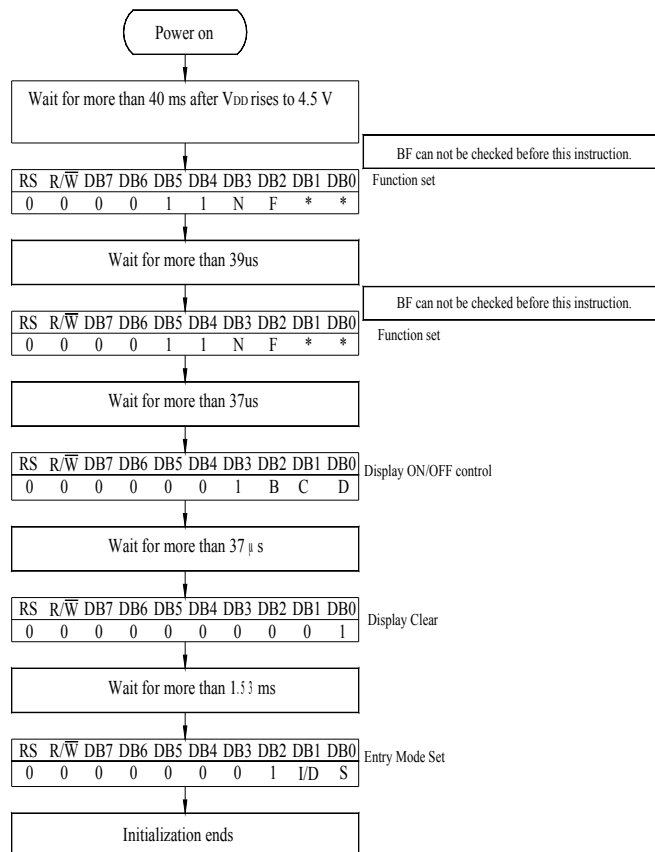


Figure 3: LCD initialization sequence¹. The variables *B*, *C*, *D*, *N*, *L*, *I/D*, and *S* are defined in the LCD data sheet and briefly described in Figure 4.

Open the file OpenXLCD.c and look at lines 56 - 117. These lines implement the P-tec initialization sequence and need to be replaced with the correct sequence. To do this, an 8-bit sequence for each of the steps outline in Figure 3 will be written using the function WriteCmdXLCD(). Before the first write and after each additional write command, a delay is needed to wait for the LCD to process the command.

There are delay functions pre-defined in delay.c. These are the delays:

- Delay18Tcy() - Delay for 18 time cycles
- Delay100Tcy() - Delay for 100 time cycles
- DelayPORXLCD() - Delay for a minimum of 15ms
- DelayXLCD() - Delay for a minimum of 5ms

These delays do not correspond exactly with the required delays that are outlined in Figure 3. Since those numbers are minimum wait times, any delay that is longer than the minimum can be still be used.

To begin the correct initialization, remove lines 56 - 117 of OpenXLCD.h. In place of these lines, enter the first necessary delay. This should be any combination of the delay functions listed above which sum to greater than a 40ms delay. The easiest sequence is to call DelayPORXLCD() three consecutive times. A write command is used next with a bit sequence of 001111** (where * can be either 0 or 1). This can be expressed as: WriteCmdXLCD(0b00111100); Continue adding delays and write sequences until the entire initialization process is complete. Figure 4 shows a correct initialization sequence.

```
/* Allow a delay for POR.(minimum of 45ms) (requires 40ms) */
DelayXLCD();
DelayXLCD();
DelayXLCD();

/* Set up the interface to the lcd.
   0 0 1 1 N F * *
   N = number of display lines N:2-line/1-line
   F = display font type F:5x11 dots/5x8 dots
*/
WriteCmdXLCD(0b00111100);

/* Allow a delay of at least 100us (requires 39us) */
Delay100XLCD();

/* Set up the interface to the lcd.
   0 0 0 0 1 D C B
   D = display ON/OFF
   C = cursor
   B = cursor blink
*/
WriteCmdXLCD(0b00001111);

/* Allow a delay of atleast 100 micro secs (required: 37us)*/
```

```

Delay100XLCD());

/* Display Clear */
WriteCmdXLCD(0b00000001);

/* wait for atleast 5ms (required: 1.53ms) */
DelayXLCD());

/* Entry Mode Set
   0 0 0 0 0 1 I/D S
   I/D = cursor moving direction
   S = shift of entire display
*/
WriteCmdMOLCD(0b00000110);

```

Figure 4: Correct initialization code

Using the LCD module

To use the LCD module in a project, the `xlcd.h` file must be included. Before commands can be sent to the LCD module, the `OpenXLCD()` function that was edited above must be called first. Any other LCD commands can be called after it. Single characters can be sent to the LCD with `WriteCmdXLCD()` and strings can be sent with `PutsXLCD()`. To move between the top and bottom lines, the cursor needs to be moved. For this, a command is sent. The bit sequence `10000000` will put the cursor at the start of the top line. The bit sequence `11000000` switches to the start of the bottom line. Figure 5 shows a functional example application that will output “Hello” on the top line of the LCD and “World” on the bottom line of the LCD.

```

#include <p30f4012.h>
#include <xlcd.h>

void display_line1(char*);
void display_line2(char*);

int main(void) {
    OpenXLCD();
    display_line1("Hello");
    display_line2("World");
    return 0;
}

void display_line1(char* str) {
    WriteCmdXLCD(0b10000000); // go to line 1 start
    PutsXLCD(" "); // blank line 1 (in case it wasn't)
    WriteCmdXLCD(0b10000000); // go back to line 1 start
    PutsXLCD(str);
}

void display_line2(char* str) {
    WriteCmdXLCD(0b11000000); // go to line 2 start
    PutsXLCD(" "); // blank line 2 (in case it wasn't)
    WriteCmdXLCD(0b11000000); // go back to line 2 start
    PutsXLCD(str);
}

```

Figure 5: Sample application displays “Hello World”

Recommendations

Although this code was built to work on a certain model of LCD, the Matrix Orbital MOP-AL082B-BYFY, and use the Microchip dsPIC30F4012, the techniques can be adapted to any parallel LCD or PIC that has similar characteristics. The pins chosen in Figure 1 were recommended pins and are not in any particular order. Any of the 20 I/O pins that the PIC provides are sufficient to control the LCD. For a particular project, use the pins that make the most sense for board design and implementation.

Conclusion

This application note demonstrates how to control a parallel LCD with a PIC. Using the same techniques, other parallel interfaces can be controlled via a PIC. Furthermore, this application note demonstrated how to take existing libraries and manipulate them into operating different hardware and running on different hardware than the library was intended. By using this technique, the developer can save time and reduce bugs by using software that has already been used by other people.

References

1. Matrix Orbital. (n.d.). MOP-AL082B-xxxx-xxE-xxx. *In Specifications for LCD Module*.
2. Microchip Technology, Inc. (2006). *16-bit Development Boards, Tools and Libraries*. Retrieved March 29, 2007, from http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=2035
3. Microchip Technology, Inc. (2007). *In DsPIC30F4011/4012 Data Sheet*.