ECE 3710 LAB IV

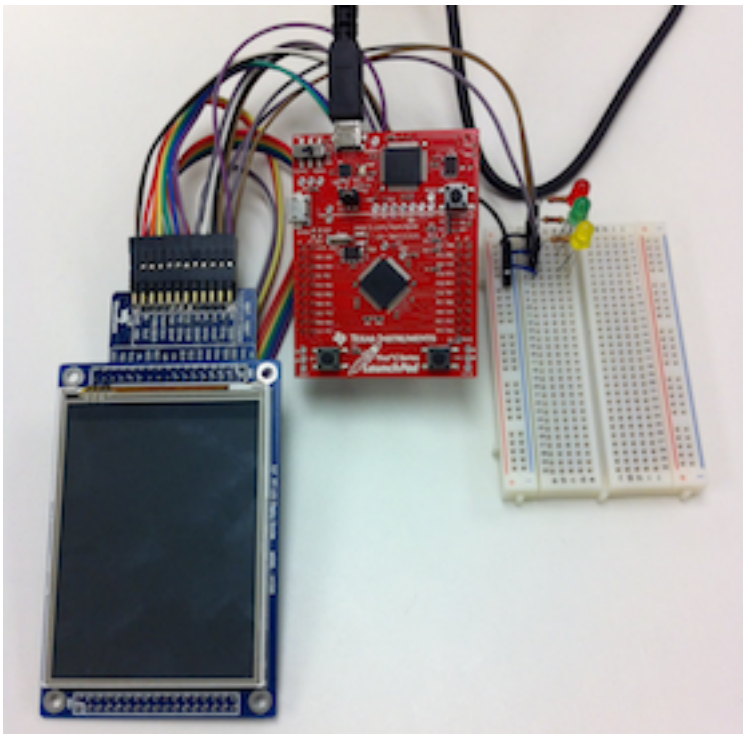
Due Date: Week of November 4th 2013 (20 Points)

Objectives

The student should be able to interface with the LCD touch panel for input and output in the C programming language.

Overview

In this lab you will practice programing the microcontroller using C. You will be using the LCD touch panel as a 3-button array to turn on or off corresponding LEDs on your breadboard.



Preparation: week one

- Look over the SSD1289 datasheet. This is the controller for the LCD display. Here you will find descriptions of the display controller, its command indexes, the display RAM addresses, display on/off sequences, etc. Particularly relevant chapters are seven, thirteen, and fifteen.
- Review the lecture notes from lectures 19 and 20.
- Download the LCD project code (LCD.zip) from the wiki.

Hardware

- Tiva™ C Series TM4C123G LaunchPad Evaluation Board
- SSD1289 / XPT2046 LCD touch screen
- 3 LEDs (1 red, 1 green, 1 yellow)
- Three 220 ohm resistors or 220 ohm resistor bus
- Jumper wires

Requirements

1. Three buttons must be drawn on the LCD. One red, one green, and one yellow. The buttons will be blank colored outlines initially (like the red and yellow button in figure 1).
2. When a button is touched the button will be filled (like the green button in figure 1). When that button is touched again the button should be return to a blank outline.
3. The red, green, and yellow LEDs should be wired to three available ports for output. The LEDs must be lit up with the corresponding LCD button. If the LCD button is solid, the LED should be on and vice versa.

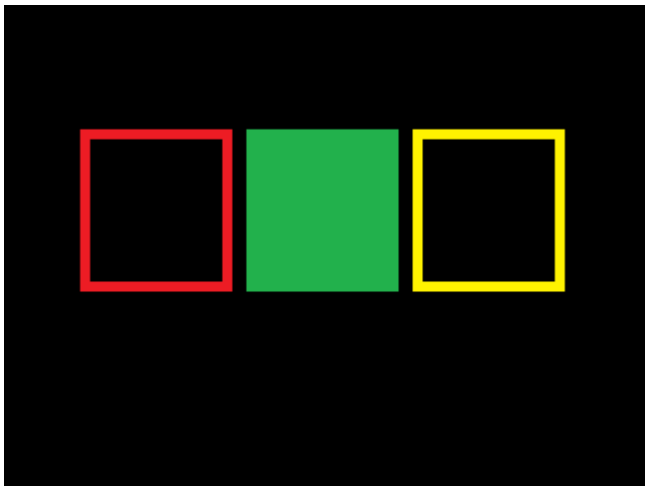


Figure 1: LCD Button Example

Verify Requirements

Demonstrate to your TA that all buttons work correctly as described in the requirements section. Touching outside the buttons must have no effect on the buttons or LEDs. Debouncing must be properly implemented. Touching and holding a button must not toggle the button state until the button is released. You must discuss in your report how you handled various touch scenarios including touches outside the button boundaries, partly-in-partly-out touches, touch and holds, debouncing, etc.

Week One

This is a two-week lab. At the end of the first week you should 1) have a complete interface (software and hardware) implemented for the LCD, and 2) be able to use said interface to draw the filled/unfilled

boxes. To make debugging your code easier, in the first week you should use the 16-bit parallel interface of the LCD; assume 65k color operation. In the second week you will use the 8-to-16-bit adapter to minimize the number of pins used by the LCD.

We have provided a skeleton project file that includes a number of function prototypes that we feel are useful; you are free to use our function definitions or write your own. The project file (LCD.zip) should work with your board without further modification.

Next week's phase of this lab will be discussed in detail after this week.

Notes and Tips: week one

Tip one:

The LCD controller on your LCD module from WaveShare has a few different interface configuration options; however, our LCD module is hardcoded for the **8080 16-bit parallel interface**. The pins used are D[15:0] (data), RD (read), WR (write), and D/C (data/command) and CS (chip select) where the read, write, command and chip select signals are asserted low (chapter seven and fifteen of datasheet). See the illustration below:

15.1.2 8080-series System Bus Interface

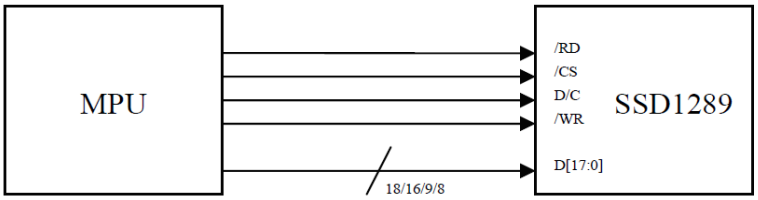


Table 15-3 – Interface Mode Selection

PS3	PS2	PS1	PS0	Interface Mode	Data bus	/WR	/RD	D/C	/CS	Operation
1	0	0	1	16-bit 8080 parallel interface	D[17:10], D[8:1]	1	0	0	0	Read 8-bit command
						1	0	1	0	Read 8-bit parameters or status*
						0	1	0	0	Write 8-bit command
						0	1	1	0	Write 16-bit display data

In order to write data to the LCD controller, the WR line must be brought low. The D/C pin determines whether you are writing a command (index register) or data. In order to make the LCD do something useful, you will be stringing together sequences of a command to select an index register (with D/C cleared) and then writing data (with D/C set) to that register. Your book, class notes, datasheets and the included example code will help you figure out the right sequence to complete this lab. Please see the table 8-1 (command table) in the LCD Controller Datasheet for more information about the registers, their addresses, and functions (chapter fifteen of datasheet).

In the future, it may be advantageous to read the data of a pixel or command register. A low D/C signal reads the contents of a command register whose address must be written prior to reading its contents. A high D/C signal reads the display data. Note that when reading display data, you must disregard the

first 8-bit word that is returned upon bringing the RD pin low. The next 8-bit word after bringing the read pin low once again is the first valid data chunk in a series of read cycles. This is illustrated below:

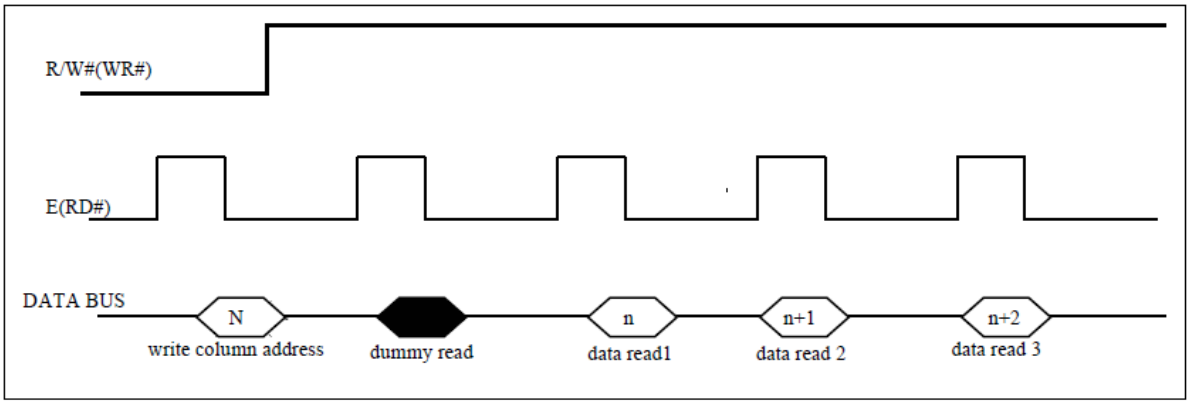


Figure 7-1 – Read Display Data

Note that the pins used in the 8080 16-bit interface (the one we will be using) are labeled in parentheses.

Tip two:

You will want to consult chapter thirteen of the LCD datasheet to ensure that your uC provides the correct timing. Without doing this, your code will not work. Chapter 13 also gives a graphical representation of the state of the pins during a write cycle that you may find useful.

Tip three:

Several drivers for the SSD1289 LCD touchscreen have been posted to the wiki. While you must write your own driver, you can look at existing code to figure out how to initialize the LCD or write pixel data; some of these are more useful than others.