CPE 490 Lab 8

## Goals

1. Write code to improve the performance of the LCD drivers
2. To think about how to improve embedded system performance by elimination of delay loops
3. Read timing diagram and write software in accordance to timing requirements

## Equipment

You will be using the explorer 16 board with the MPLAB IDE and the MPLAB C30 compiler

## Overview

This is a continuation of last week’s lab. Use this lab period to become very familiar with how to use the LCD, so it is not issue in future labs when you will need a display.

You will be modifying LCD.c It is a good idea to rename this file and the associated header file so you can make sure that multiple files with same name but different versions of code exists. If you don’t it can be very confusing, and you can end up using an old version of the file in future projects.

It is important to understand that there is a one to one correspondence to the Display Data RAM (DDR) and the actual display. The address space of the RAM starts at 0 and goes to 0x4F. The top line most left character is at address 0. The last visible character on the top line has the address of 0x0F. The bottom line most left character is 0x40, and the most right on the bottom is 0x4F.

## Tips and Lessons Learned

1. The code given last week, LCD.C ,has a code line: #include "p33FJ256GP710.h". This should be changed to: #include "p33FJ256GP710A.h"
2. The LCD data lines are on the lower byte of port E. Remember GPIO lines can share analog input function and that function is enabled by default. Therefore we need to make sure to turn off the analog function for our interface lines (control and data) going to the LCD. We can go ahead and place the code to do this in Init\_LCD function call. From lcd.c the following GPIO are used to make the LCD interface
   1. RW control signal – PORTD.5 – No analog function
   2. RS control signal – PORTB15 – shares analog channel AN15
   3. E control signal – PORTD4 – No analog function
   4. Data 8-bit data bus – PORTE.0 – PORTE.7 –shares analog channels AN24-31

To turn off the analog channels we set bits true in the ADXPCFGX registers. The following code can be included to turn off the analog channels:

AD1PCFGH = 0xFF00; //turns off channels AN24-31

AD1PCFGL = 0x8000; //turns off channel AN15 for ADC1

AD2PCFGL = 0x8000; //turns off channel AN15 for ADC2

## Design

Open the source file lcd.c. Check out the functions lcd\_cmd(), and lcd\_data(). These two functions are used to write all the commands and all the data to DDR. At the end of the lcd\_cmd() and lcd\_data() functions there are calls for a 5ms delay, and a 400 us delay respectively. These delays are to make sure the LCD module executes the last write of a command or data before a new write is done. They are very wasteful use of the microprocessor especially if some other task could be running.

There is a better way to do this. The LCD controller supports reading a “Busy Flag / Address Counter Read” see the instruction set table that was included in the last lab. The busy flag bit can be read to see if the LCD is ready for a new write.

The read cycle is described in the data sheet. The read cycle looks like this:



The important times (see data sheet for all values) are:

Tas: RS,R/W Setup Time = 60 ns min

trd: Read Data Output Delay = 190 ns max

When writing code to achieve these times short delay can be inserted with a nop instruction. This assembly instruction can be called by using the following function call Nop(); This function is defined in p33FJ256GP710A.h file (it actually calls a built in function of the compiler). The nop assembly instruction takes 1 instruction cycle. Using an oscillator frequency of 32 MHz, yields an instruction frequency of 16 MHz or a time of 62.5 ns.

You are to design a function checkbusy() that will check if the LCD module is ready to take a new write or that it is busy. This function will return a char that will have some non-zero value if busy, or zero if not busy. The function will require the PORTE port to (RE7:RE0) pins to become inputs to read the LCD. Make sure that you set these pins back to being outputs before leaving the function.

Verify the function works by modifying lcd\_data() and lcd\_cmd() to remove their calls to the delay function and instead use the new function to ensure that LCD is not busy before writing to it. This call to the new function should be done in lcd\_data and lcd\_cmd functions and should be done prior to trying to write to the LCD.

**Question: Why is it much better to call checkbusy at the beginning of the lcd\_data and lcd\_cmd functions than at the end of the function (that is enter a loop after writing that checks for busy and when it is not busy return from the function).**

To try out your new code use your new routines in the LAB 7 program you wrote. Demonstrate to the instructor \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.