Lab 1 (Team Lab): Stopwatch

Start: After demo Lab 0
Demo Due: October 3, 2014
Points: 50
After your demo, ***submit your C file*** on D2L Dropbox by ***11.59 PM on Oct 3.*** Otherwise, 20% will be deducted from your lab1 points.

Lab Overview: In this lab,

- You will begin with learning the basics of interfacing external components (switches, LEDs, LCD) with the PIC microcontroller, utilizing your soldering and wire wrap skills to integrate additional components with you PIC microcontroller, and using timers to make precise timing measurements.
- Then, you will build a stopwatch capable of timing minutes, seconds, and 1/100th of seconds with basic start, stop, and reset controls. This assignment will interface with a 8x2 character LCD display and require the use of interrupts for precise timing, and bit masking operations to access 4-bit simultaneously from a single port (PORT B).

  NOTE: As you will be incorporating additional components with your PIC microcontroller, those components can either be directly integrated in the prototyping area of the 16-bit 28-pin Starter Board or using a separate prototype board. Each team is provided with a prototype board within their Parts Kits.

  If your team decides to integrate the components in the prototype area of the Starter Board, one of the team member must be willing to use his/her board for this purpose.

Datasheets and References (also on D2L)
LCD Display Datasheet
Microchip 16-bit 28-pin Starter Board User's Guide
PIC24FJ64GA002 Datasheet
PIC24F Family Reference Manuals
PICkit 3 User's Guide

Parts Needed for Lab (all parts are provided in your team's Parts Kit):
2 - LEDs
2 - 330 Ohm resistors
1 - 8x2 LCD Screen (1.50” x 1”)
2 - momentary tactile switch
Male Headers
Optional: 4x4 Vector Prototype Board
Optional: 2 - 1K Resistors
Optional: 2 - 10K Resistors
Optional: 2 - .1uF Capacitors

Wire Wrapping and Color Coding Requirements:

The provided jumper wires are free to be used for developing and testing your system, however, when it comes to the ***final demo, everything should be*** **wire wrapped** ***cleanly with "reasonable" color coding*** (using black wires all the time is obviously a BAD color coding), otherwise 5% of the points will be deducted.

Stand-Alone Program Demo and Code Explanation (all team members must be presented at the demo time):

The TAs and/or instructor will be available on the lab and *open* lab sections for you to demonstrate your Lab 1 assignment (Part 3). During this demo, your team will need to both provide a demonstration of your working stand-alone programmed board and provide an explanation of the required modifications made to the provided software code.

C Code Submission on D2L: All C files for each lab assignment and final project should be submitted on D2L in the Dropbox. For scheduled lab assignments, the C files must be submitted by 11:59PM on the Friday of the week during which the lab assignment is due.

Provided Software Code:

Source Code for Lab 1 Assignment: Available on class D2L

Part 1: lab1p1.c

Part 2: lab1p2.c, lcd.h, lcd.c

There are several // TO DO statements throughout the given codes to help you complete the lab.

When writing your code, follow C Coding and Commenting Style Guidelines (see page 5)

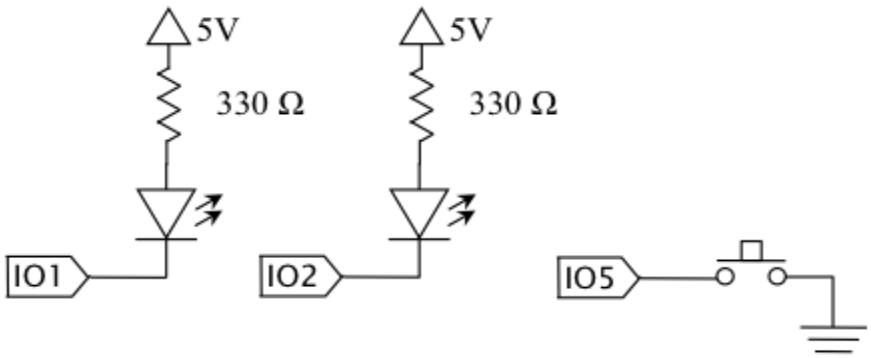
Lab Procedure and Demo:

Part 1: Dueling LEDs

Hardware Connection:

1. Connect the 2 LEDs to IO1 and IO2 of the 16-bit 28-pin Starter Board. The anodes *(long wire)* of the LEDs should be connected to 5V through a 330 Ohm resistor. The cathodes *(short wire)* should be connected to the IO1 and IO2 inputs.
2. Connect the momentary tactile switch to IO5 of the 16-bit 28-pin Starter Board.

   The momentary switch has 2 poles but 4 pins. Two pins on each side of the switch are connected together. Using the continuity measurement of your multimeter, determine which pins correspond to the two poles of the switch. Connect one side of the switch to ground and the other (not connected to the first side which is already connected to ground) side to IO5.
3. A schematic for the required circuit is provided below:




```
Note: Configure the CNPU register
to enable internal pull-up resistor
for this switch input.
```

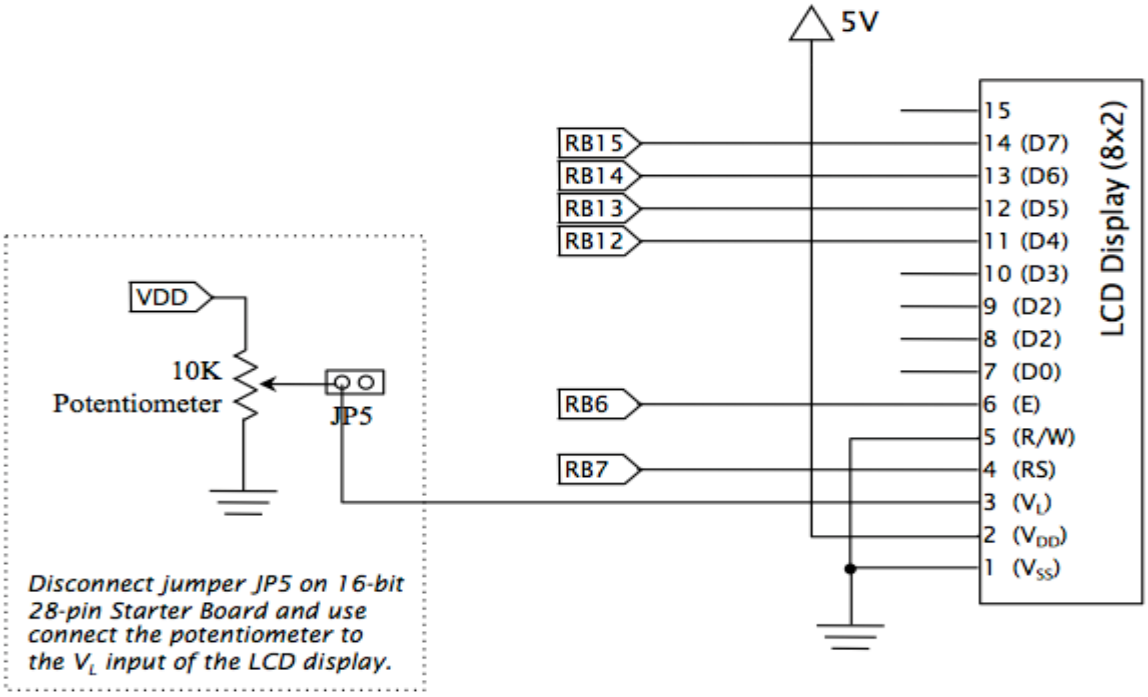
Software: Using the provided C code template (**lab1p1.c**) for Part 1 of this lab, write C programming code with the following requirements:

- The LEDs should be organized in such a way that they can be differentiated as a *Run* LED and *Stop* LED.
- Initially, the *Run* LED should be illuminated.
- For every distinct press of the switch, the LED being illuminated should be alternated. For example, if the *Run* LED is currently on, the *Run* LED will be turned off and the *Stop* LED will be turned on.
- Both the button press and button release should be debounced in software using precise 5 ms delay. This delay should be controlled using Timer1 of the PIC microcontroller.

Part 2: LCD Interface and Software Driver

Hardware Connection: Connect the 8x2 LCD Display to the PIC24F using the provided schematic:
Note: The V_L input of the LCD Display is used to control the contrast of the display. The 16-bit 28-pin Starter Board has a 10K potentiometer that can be used to adjust this contrast. To connect the potentiometer, you can connect to jumper JP5.

Test your LCD by connecting the first 3 pints correctly and adjusting the potentiometer used on the board until you see black boxes on the first line of the LCD.



Software: Using the provided C code (**lab1p2.c, lcd.h, lcd.c**) complete the software driver code required to interface with the LCD. This C code includes a simple demonstration program that will utilize the LCD software driver to print sample messages on the LCD.

The provided C code for the LCD software driver (**lcd.c**) provides specific instruction within the comments on what additional functionality must be incorporated. These required modifications are marked with ***TODO***. To complete the software driver, you will need to complete the following:

1. Develop a ***DelayUs( )*** function that will utilize ***Timer 2*** to create a user specified delay in micro seconds provided as function parameter.
2. The LCD software driver (lcd.c) utilizes a 4-bit interface to write 8-bit values as two consecutive 4-bit write operations. Complete the ***WriteLCD( )*** function to write the most significant 4-bits followed by the least significant 4-bits to the LCD_D output. Your code should utilize the #define **LCD_D** that is defined within the provided code to write data to the LCD. You will need to utilize bit masking and shifting operations to perform these writes.
   Hint: An example of how to utilize bit masking and shift operations can be found within the ***LCDInitialize( )*** function.
3. The initial steps needed to configure the LCD Display to utilize a 4-bit interface have been provided. Complete the ***LCDInitialize( )*** function to configure the operational mode of the LCD. Details of the initialization process and control instruction needed to configure the display are explained within the LCD Display Datasheet.
   Note: Pay close attention to the required delays needed for various control operations.
4. Complete the ***LCDClear( )*** function to clear the LCD display using the appropriate control instruction.
   Hint: This functionality should be the same as the clear display operation you are required to add the ***LCDInitialize( )*** function.
5. Complete the ***LCDMoveCursor( )*** function that will move the cursor to the user specified (x,y) coordinate. This operation must be performed using a single control instruction. As the move cursor control instruction depends on the format of the display, careful attention must be paid to utilize the correct control instruction for the 8x2 display format. This information can be found within the LCD Display Datasheet.
6. Complete the ***LCDPrintChar( )*** function for displaying a single ASCII character on the LCD Display.
   Hint: This functionality can be easily achieved using the various functions already implemented.

Part 3: Implement Stopwatch Functionality

Hardware: Using 2 LEDs, 2 momentary tactile switches and LCD

Besides the existing momentary tactile switch in "Part 1" of this lab, add an additional momentary tactile switch to any available input of the PIC24F that support change notifications. Ports supporting change notification are labeled on the PIC24F pin diagrams with *CN*.
Hint 1: You can utilize SW1 of the 16-bit 28-pin Starter Board as the additional switch.
Hint 2: To avoid requiring software methods to debounce your push-button (switch) input,

consider using hardware circuitry to debounce your buttons inputs. Refer to the circuit used for the connecting SW1 of the 16-bit 28-pin Starter Board.

Software:
Using the software driver you created to interface with the LCD Display in Part 2, and combining with the dueling LED functionality in Part 1, **develop a C program to implement a basic Stopwatch**, with the following requirements:

1) The Stopwatch will have 2 user inputs from the momentary tactile switches for *Start/Stop* and *Reset*.
   NOTE: The "Start/Stop" button has already been implemented in Part 1 of this lab, we will add more functions for it to control the LCD.
2) The *Start/Stop* input will be utilized to start and stop the execution of the Stopwatch. When stopped, the current time will be maintained, such that when the Stopwatch is started again, the timing will continue from the current time.
3) When the Stopwatch is started, the top line of the LCD Display should display "Running:", and the "Run" LED should be lightened up. When the Stopwatch is stopped, the top line of the LCD Display should display "Stopped:", and the "Stop" LED should be lightened up.
4) The *Reset* input will be utilized to reset the current time back to 0 when the stopwatch is stopped. If the stopwatch is started (i.e. running), the *Reset* input should be ignored.
5) A change notification **interrupt** ***must*** be utilized to detect user inputs for the *Start/Stop* and *Reset* inputs.
6) The Stopwatch must have an accuracy of 1/100th of a second (i.e. 10 milliseconds). The bottom line of the LCD Display will be utilized to display the current time using the format

   MM:SS:FF

   where MM is the current minute, SS is the current seconds, and FF is the current 1/10th ad 1/100th of a second.
7) An **interrupt** using Timer 1 ***must*** be utilized to control the timing of the Stopwatch.

C Coding and Commenting Style Guidelines

a. All files should have the names and date. If working in a group, then all group members should be listed.
b. All functions should have a descriptive note of what it does.
c. All variables should have a descriptive name and a comment on what it's used for.
d. Any PIC configuration setting should have a comment indicating what the configuration corresponds to.
e. In the later labs, if you can use an interrupt instead of a busy wait, then you should use the interrupt.

Note: Class examples don't always adhere to this standard, be more diligent then we were.