Spring 2015 Lab Assignment #7
Stopwatch

1 Introduction

You will use your Arduino UNO board and the LCD shield to write C code to implement the functionality of a stopwatch. You may utilize code from other labs or in class demos as modules/functions for this project.

2 What you will learn

This lab will help you integrate the embedded coding concepts and different I/O functionality including timers, ADC, and digital I/O in a single project.

3 Background Information and Notes

3.1 Stopwatch Application

This project will implement a stopwatch application that counts upwards in increments of tenths of seconds from 00.0 to 59.9 seconds. It will provide the ability to start, stop, and reset (back to 00.0) the stopwatch. It should also implement a “lap” feature which freezes the displayed time at the instant the “lap” button is pressed while still keeping the internal watch time incrementing. When “lap” is toggled again (or “start” is pressed again) the internal watch time (which has been running) should be re-displayed and then continue as normal.

3.2 Other I/O

As input to your stopwatch you will use two push-buttons (the Up/Top and Down/Bottom) on your LCD shield. The Up/Top button will serve as your “Start_Stop” button. When pressed it shall simply toggle the state unless in lap mode (more to come). The Bottom/Down button will serve as the “Lap_Reset” button. When the stopwatch is started/running, a press of “Lap_Reset” will serve as the ‘lap’ feature. Any subsequent press of either “Start_Stop” or “Lap_Reset” will toggle the lap feature to return to running display of the current time. When the stopwatch is stopped, a press of “Lap_Reset” will reset the time to 00.0.

3.3 Suggested Approach

It will likely be beneficial to keep a state or mode variable tracking what mode your stopwatch is in. Presses of the button will cause updates/transitions between modes and states as shown in Fig. 1. What should be done in each state (whether the time should be incremented and what should be displayed should be fairly intuitive). You may also find it beneficial to keep an internal representation of the time (separate variables for each digit) and then only update the display I/O every tenth of a second (using some kind of timer interrupt).

In terms of overall structure, there are many ways to organize your program. One suggested approach is to sit in a loop to poll the ADC for button presses updating the state as needed. Then keep your timer

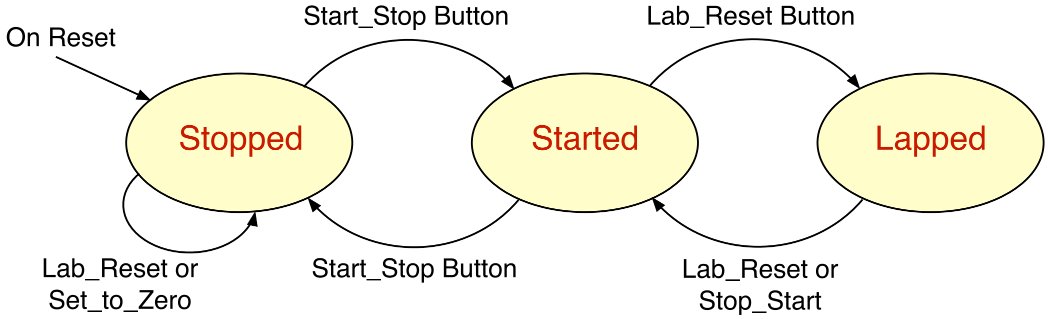

Figure 1: State transition diagram for the stopwatch

generating interrupts so that you can put your display and time update logic in the timer ISR. That code can examine the state variable to determine if an update of the internal and/or displayed time is necessary.

4 Prelab

Consider what prescalar and modulus count you can use that will yield a 0.1 second timer interrupt interval. Refer to the slides shown in class for the register definitions of the 16-bit timer (TIMER1).

5 Procedure

1. Create a new folder and copy a `Makefile` and old `.c` file from some other exercise lab and name it `lab7.c`. Make sure the program comments at the top include your name and class time. Modify the `Makefile` appropriately to compile the file named `lab7.c`.

2. Perform initialization code for the necessary PORT registers, ADC, LCD, and timer. You MUST make separate functions for each of these I/O devices and consider what appropriate arguments would be to each initialization function that would allow you to reuse these functions in a future project.

3. Use/alter code from your previous labs/exercises as a basis for completing the rest of the lab. You will need to integrate code from the timer and ADC examples to use a timer that generates an interrupt every 0.1s, code to poll the button inputs, code to output to the LCD, and code for the main routine.

4. Define a global variable, '`state`', and character array to hold the time, '`time[3]`'. Use these to store appropriate state data.

5. Your program should meet the following requirements:

   (a) Initialize the count to 00:0 on startup.

   (b) Correctly display all times on the LCD.

   (c) Starting counting in tenths of seconds when "Start_Stop" is pressed and the timer is stopped.

   (d) Stop counting in tenths of seconds when "Start_Stop" is pressed and the timer is started.

   (e) Hold the displayed time while continuing internal time updates when "Lap_Reset" is pressed and the timer is started.

   (f) Update the display with the current internal time and continue counting when "Start_Stop" or "Lap_Reset" is pressed and the display time is being held (i.e. LAP state).

   (g) Reset the time to 00:0 when "Lap_Reset" is pressed and the timer is stopped.

6. Comment your code with enough information to convey your approach and intentions. Try to organize your code in a coherent fashion.

7. Once you have the assignment working demonstrate it to one of the instructors and get their initials on your rubric.
8. Turn in a copy of your source code through the link on the web site. If you have multiple source code files zip them all into a single file and submit the zip file.

6 EE109 Lab 7 Grading Rubric

Name: ____

TA/Instructor Successful Demo initials: ______________

Item	Outcome	Score	Max.
Initialization			
• Provided code to initialize appropriate Port registers	Yes/No		1
• Provided an appropriate Timer init. function	Yes/No		1
• Chose an appropriate Timer prescalar and modulus count	Yes/No		1
• Provided an appropriate ADC init. function	Yes/No		1
• Provided an appropriate LCD init. function	Yes/No		1
Main Program			
• Used state variable to control display	Yes/No		2
• Time increment code is correct	Yes/No		2
Requirements			
• Initialize count to 00.0 on startup	Yes/No		1
• Correct display of time on the LCD	Yes/No		2
• Time starts counting when “Start_StoP” pressed	Yes/No		1
• Time stops counting when “Start_Stop” pressed	Yes/No		1
• Display time holds on “Lap_Reset”	Yes/No		1
• Time continue to increment internally and is then displayed on next “Lap_Reset” or “Start_Stop”	Yes/No		2
• Reset the time to 00.0 when “Lap_Reset” is pressed and time is stopped	Yes/No		1
Code Organization			
• Well-commented code	Yes/No		2
Total			20
Open ended comments:			