ECE2049: Foundations of Embedded Systems
Lab Exercise #1 –A Term 2018

Implementing a “Simon Says” game

In the late 1970’s and early 1980's, one of the first and most popular electronic games was Simon by Milton Bradly. The objective of Simon was simple. The game displayed a sequence using 4 lighted buttons each with a unique musical note. The player’s task was to repeat from memory the sequence that game had just played. When the player correctly repeated a sequence a longer sequence was played – at a faster rate. In this lab you will implement a simplified version of the Simon game using our MSP430F5529 lab board.

Assignment:

Starting with the I/O functions from the demo project, implement a Simon-like game where the 4 colored LEDs on the IO board are lit one at a time and the buzzer sounds in a pseudo-random sequence (play the game at the link above). When the player correctly repeats the sequence by pressing the corresponding keys (1-4) on the keypad then a longer sequence is played. If the player makes an error then the game should indicate it by flashing LEDs, sounding the buzzer and writing to the LCD.

An example of a game would be

Flash LED 2	Answer 2
Flash LED 2 1	Answer 2 1
Flash LED 2 1 1	Answer 2 1 1
Flash LED 2 1 1 4	Answer 2 1 1 4
Flash LED (faster) 2 1 1 4 3	Answer 2 1 1 4 2

'SIMON SAY ERROR' or something … accompanied by annoying flashing LEDs and buzzer sounds and such

MANDATORY PRE-LAB Assignment: ***READ THE ENTIRE LAB ASSIGNMENT!*** Using the main( ) function of the demo project from Lab 0 as a guide write the MAIN LOOP of your Simon game. You do not have to implement all the function calls or compile the loop. However, you MUST show the TA (and sign-off) a draft of your Simon main loop ***at the beginning of your lab session.*** The idea is to actually THINK about the HOW to tackle the problem BEFORE you get to lab!

It is recommended that you implement your game as a state machine. There are specific tasks or states that your code will move through within the main loop like Display the Welcome message, Play Sequence, Check Player Input, Update Sequence/Difficulty, etc. You can assign these states numbers (or define an enumerated type) and use a switch – case construct to implement your game inside a while (1) loop something like

```
switch (state) {
   case 0:  // Display welcome screen
      . . . .
      break;
   case 1: // Play Sequence
      . . . .
      break;
    case 2:   // Check Player Input
      . . . .
      break;
    etc...
```

This is just a notional example you will need to determine what states your game requires and what actions take place while in each state.

Note: The labs for ECE2049 are not tutorial in nature. Rather, you will apply what you have learned in lecture to expand your code base (i.e. the demo project) to complete mini-projects. How you complete this week's project is up to you but you need to meet ALL the requirements listed below. You do not have to complete the steps in the order given.

System Requirements:

1) When the game is not being played the LCD should display “ECE2049 SIMON” and “Press * to start”. The game should revert to this welcome screen after the player loses.
2) A new game starts whenever the '*' key is pressed. The game should then give a 3-2-1 countdown on the LCD before starting.
3) The Game should flash a sequence on the 4 multi-colored LEDs, one LED at a time. The buzzer should sound with each LED flash. The sequence should get longer with each correct answer. For the

purposes of the game consider left-most LED (D1) to be “1” and the right-most to be “4”.

4) When the player presses a key, the key number (1-4) should be displayed on the LCD with a spatial alignment that correspond to the key value. That is, if the player presses button 1 a 1 should appear on the left side of the screen. If button 2 is pressed a 2 should appear more toward the middle of the screen and if 4 is pressed then a 4 should appear on the right hand side of the screen. Pressing a numeric key other than 1-4 should be treated as a Game Over error.

5) *BONUS*: The sequence should playfaster as it gets longer. It is up the game designer how to implement this but the change must be noticable an add challenge to the game. (Hint: Start with a single, fairly slow speed and add the variable speed last!). Explain how you implemented the change of speed in your report.

6) There is a random number generator function `rand()` in the C Standard Library (#include <stdlib.h>). See CCS help for details on its use. *Note*: It is NOT required that the game generate a new sequence of random numbers each time you start the game as this would require randomizing the seed for the random number generator. This would be a challenging task at present.

7) Also realize that you'll need to save the sequence played in order to check it. Your game should be able to play sequences of up to length 32, at least. Why (in terms of your code design) do you need to select a maximum length for the sequence? What variable type will you use to store your sequence numbers and why?

8) *BONUS*: The buzzer buzzes because a pulse width modulated (PWM) signal is applied to a tiny sliver of ceramic. Modify the buzzerOn() function to play a different pitch for each LED. Explain fully in your report how you achieved this.

9) Any modifications of functions or new functions you write should use the port register names defined in msp430F5529.h. No magic numbers!

10) Write a high quality lab report. Your report should include a flow chart of your game (or several flowcharts if that shows functionality better). All flow charts should be computer generated.

11) Remember to submit your code on-line.

********Remember to make your project names meaningful and unique. Use revision numbers to preserve hard-won functionality before going on to next step! Save your work to your R drive directory not on the local lab machine!***

SAVE OFTEN! ***SAVE MORE OFTEN!***
NOW IS A GOOD TIME TO SAVE!

IMPORTANT REMINDERS!!

1) YOU and you LAB PARTNER are responsible for the lab board that was signed out to you. You are required to return the lab board in working order at the end of the term. You will not receive a grade until you lab board is returned.

2) IF your board is not functional at the end of term, you will receive a grade penalty. See the syllabus or class web page for more information.

3) Please properly STORE THE BOARD IN ITS PROTECTIVE BOX when not in use. Also, don't destroy the box!

4) DO NOT PLACE THE BOARD ON A METAL SURFACE (like the computer cases in AK-113!) WHILE PLUGGED IN. It can short the board. Be sure the rubber feet are attached to the back of the board to further guard against this.

Writing a High Quality Lab Report:

Your lab should be written in a professional style. It should be an electronically prepared technical document similar to what you would submit to a co-worker or your boss. The report should include

Introduction = 1-3 paragraphs (1/2 page tops) succinctly stating the objectives of the lab and giving an overview of what you accomplished.

Discussion and Results = As many pages as it takes (without padding!). ***In this section you should thoroughly discuss what you did to meet the stated requirements for each part of the lab and fully answer ALL questions highlighted in YELLOW!***. You should describe the approach you took to solving any problems. Include pseudo code descriptions and/or flow charts for any code you developed. Results should also be thoroughly discussed. Any measurements should be tabulated, questions should be answered completely (in complete sentences) such that *the grader can find the answers*, figures should not be hand drawn, snippets of code may be included where useful but do not print out or attach all the code as you will be submitting your code on-line.

Summary and Conclusion = 1-3 paragraphs. Wrap-up and summarize what you accomplished in the lab. This should be a "bookend" to the introduction.

Appendices = *SIGNED* PRE-LAB assignments from BOTH students should be included. Also include any relevant raw data sheets. Do not include code listing. Code is to be submitted on-line.

The original copy of the lab sign-off sheet must be attached to you report!

BOTH PARTNERS MUST BE PRESENT FOR SIGN-OFF OF ANY OR ALL PARTS OF THE LAB!

To submit your code for grading, you will need to create a zip file of your CCS project so that the TAs can build it.

You can also use this method to create a complete backup copy of your project (perhaps to archive or send to your partner) for later. To do this:

1. Right click on your project and select "Rename..."

2. If you are submitting your project, enter a name in the following format: *ece2049da18_lab1_username1_username2*, where username1 and username2 are the user names of you and your partner. (NOTE: Failure to follow this step will result in points deducted from your lab grade!)

3. Click OK and wait for CCS to rename your project.

4. Right click on your project again and select "Export..." then select “General” and "Archive file" from the list and click Next.

5. In the next window, you should see the project you want to export selected in the left pane and all of the files in your project selected in the right pane. Select all. You should not need to change which files are selected.

6. Click the "Browse" button, find a location to save the archive (like your R drive) and type in a file name using the EXACT SAME NAME used in Step (2).

7. Click "Finish". CCS should now create a zip file in the directory you specified.

8. Go to the Assignments page on the class ***Canvas*** website. Click **Lab 1 Code**. Attach the archive file of your project that you just created and hit the Submit button. Only one code submission per team.

ECE2049-A-2018
Lab 1 Sign-off Sheet

Early (Bonus) Sign-off: **Fri 9/7/18** ***Report due***: **Tue 9/11/18**

Student 1: _____ ECE mailbox: _____ BRD #____

Student 2: _____ ECE mailbox: _____

Task Max points	***Max points***	***TA's assess-ment***
PRE-LAB Complete at start of lab (students graded individually) BOTH PRE-LABS MUST BE ATTACTED TO REPORT	10	Student 1 Student 2
SIMON on LCD	5	
Game Start/ Restart (on *)	10	
Display random sequence on 4 LEDs (length of at least 32)	10	
Display player's button press on LCD, spatially aligned with button	10	
Play complete game (i.e. play se-quence and check sequence correctly) BONUS: Increase speed as you go	15 (+5)	
BONUS: Have buzzer play different pitch for each LED	(+5)	
Proper player humiliation on error & re-set to welcome screen	5	
Answer to TA Questions (5 pts per student, graded individually)	5	Student 1 Student 2
Report	30	
Total points (with bonuses)	100 ***110***	

TA's signature: _____ Date: _____

***** Both Students MUST be present at Sign-off for any and all parts!!***

ECE2049 Lab Report Grading Rubric

Format -- 5 pts

Did you follow instructions given above as to the format of your report?

Is you code formatted, properly commented, etc.?

What is expected for the following parts was already described above.

Introduction – 3 pts

Discussion/Questions – 15 pts

Conclusion – 3 pts

Appendices – 1 pt

Professionalism – 3 pts

Spelling, grammar, neatness, presentation, etc.