ECE2049: Embedded Computing and Engineering Design
Lab Exercise #2 -- A Term 2018

MSP430 Hero!
Digital IO Meets Timers & Interrupts

The Guitar Hero series of music video games are very popular and a lot of fun. They are also, like many popular games, “conceptually simple”. The game plays a familiar rock song while flashing lights indicating the notes that the player should press on their peripheral I/O device (i.e. plastic guitar!). In this laboratory you will use MSP430 to implement a simple version of this game, MSP430 Hero. Your game will play a tune using a buzzer and flash the 4 multi-colored LEDs with the notes. The player will play by entering the notes flashed using the buttons. If the player can't keep up then he or she looses the game. There are several objectives in this lab. They include gaining further experience with digital I/O, understanding the operation of the MSP430's timers and gaining experience writing software which depends critically on the passage of time. You have 2 weeks to complete this lab but you will need to start right away. There are a large number of functions that must be implemented in order to complete the whole project.

********** ***Pre-lab (to be signed off in lab 9/12/18)*** **********

Remember that BOTH partners are required to complete do the pre-lab individually (submit on paper) and to get the pre-lab signed off during lab. Submit the signed pre-lab assignments from both partners as an appendix to your report!!! As always, code should be typed.

1) *READ THE ENTIRE LAB ASSIGNMENT!!*

2) Write a function to configure the 4 lab board buttons., S1 through S4 See the Lab Board schematics, Homework 2 and Lectures 6-8 for information on digital IO and the buttons.

3) Write a function that returns the state of the lab board buttons with 1=pressed and 0=not pressed. For example if S1 alone is pressed then the function should return 00000001b = 0x01. If only S3 is pressed then the function should return 00000100b = 0x04. If both S1 and S3 are pressed then the function should return 00000101b = 0x05, etc. Remember that the buttons are not on contiguous I/O port pins. You will have to check button each individually and combine their states before returning from your function.

5) Write a complete C function to configure and light the 2 user LEDs on the MSP430F5529 Launchpad board based on the char argument passed. If BIT0 of the argument = 1, LED1 is lit and if BIT0=0 then LED1 is off. Similarly, if BIT1 of the argument = 1, LED2 is lit and if BIT1=0 then LED2 is off. Again, see the MSP430F5529 Launchpad User's Guide (Useful Links), HW#2 and Lecture 8 for

information on the user LEDs and functions configuring and the using other LEDs.

```
void configUserLED(char inbits);      // example prototype
```

4) In order to play a song, you will need to find a way to give each note of your song both pitch and duration. You will also need to find a way to map notes to LEDs such that the same note always light the same LED. With only 4 multi-colored LEDs the same LED will need to correspond to more than one note but that is the case in Guitar Hero, too. What data structure(s) will you use to store pitch, duration and the corresponding LED? What length songs will you eventually want to play? Given how you choose to save your notes, etc., how much memory will that require?

****

Requirements:

In implementing your game you are required to complete each of the following tasks. You do not have to complete the tasks in the order listed. This is a 2-week lab but start right away.

1) At start up, the LCD should display a MSP430 Hero welcome message which instructs the user to press the '*' to begin. On system power up or game reset (caused by losing or pressing '#'), the game should be in this mode.
2) After the '*' key is pressed, the game should give a 3 second count down before it begins playing its song. The LCD should display the time 3, then 2, then 1, then GO. The Launchpad user LEDs should also flash on the count down something like LED 1 (on 3) then LED 2 (on 2) then LED1 (on 1) then both together (on GO). *In the final version, the count down must be measured by Timer A2 and NOT implemented using software delays. Explain the difference between event (or interrupt) driven code and polling. Is your final code strictly event driven or will you use a mix of interrupts and polling? Explain in your report.*
3) To produce sound you will use the simple ceramic buzzer on the lab board. The buzzer sounds when a periodic waveform is applied to it. In our template lab project this is done by generating a pulse-width modulation (PWM) wave form using Timer B0.5. See the `buzzerOn()` and `buzzerOff()` functions given in `peripherals.c`.
4) In order to play different notes you will need to create a new buzzer function that accepts an input variable for pitch. The Timer B CCR0 register controls the *period* of the PWM waveform. Changing the period will change the pitch of the sound. Remember that frequency of a sound is *1/period* and that the period here is specified as a number of ACLK tics. Below is a table of the frequencies of an octave of musical notes in Hz. *You will need to do some math to convert these to number of ACLK tics. Discuss your conversion of frequency in Hz to Timer B CCR0 settings in your report*

5) Your song should be ***at least 28 notes long*** with at least **8** different pitches. With each note, one on the four multi-colored LEDs should flash. As noted in the pre-lab, the same note should always flash the same LED but with only 4 LEDs, multiple notes may be mapped to a single LED. *How will you control the duration of your notes? Will you do this within the buzzer function or within the main game loop? Remember you will need to be checking buttons during the notes. Explain your choice in your report.*

6) Initially you may use software delays just to get your song playing but eventually you will need to be checking for button presses while your song plays. *Explain in you report why software delay would then no longer work and why you must implement note duration using the timer interrupts*.

7) Implement the game requiring the player to press the lab board button (i.e. one of the 4 buttons) corresponding to the LED flashed ***within the assigned duration of the note***. You MUST use Timer A2 to measure this interval. ***The time resolution of Timer A2 should be at least 0.005 sec***. *Explain in your report how you setup Timer A2 and why Timer A2's resolution should be several times smaller than the duration of a note.* You will need to poll the lab board buttons to see if they have been pressed and then determine if the press was in time. You can use the RED and GREEN user LED1 and user LED2 on the Launchpad board (in addition to the LCD screen) to provide user feedback as to whether the proper note was hit. ***It is recommended that Timer A2 be started once at the beginning of the song*** and that you compare the duration of the note against the global count of interrupts (e.g. If a note of duration 700 starts at a timer count of 32400 then it should end when the timer count is 33100). This will be much simpler then starting and stopping the timer with each note.

8) In the Guitar Hero games, when the player falls behind, the game plays bad notes and eventually boos them off the stage. How will your players lose? Implement proper player humiliation for losing. *Explain your rules for scoring and losing and how you implemented them in your report.*

9) Also, implement proper player congratulations for winning.

10) After losing or winning or when the '#'' is pressed, reset the game to the welcome (1).

11) Write a high quality lab report answering all the questions in the requirements section!

BONUS: Present your complete game to a member of the Course Staff to be nominated for Best Game awards. The course staff will select ~5 games that go above and beyond we'll have a run-off in class to award the 15 pt BEST GAME AWARD! (Runners up will receive 10 pts and 5 pts)

What will make *your* game better than the rest? Multiple songs? Lighting effects? Improved buzzer routines? LCD displays? Different player levels? Remember, as always, you only have 8 KiB of RAM and 128 KiB of FLASH.

NASA went to the moon with less on-board computing then what is available on our lab board!! What can you do?

Note	Freq. (Hz)
A	440
B flat	466
B	494
C	523
C sharp	554
D	587
E flat (Eb)	622
E	659
F	698
F sharp	740
G	784
A flat	831
A	880

Beginning of “Three Blind Mice” = E D C, E D C, F E D, F E D ...
Intro to “Smoke on the Water” = C D Eb, C D F Eb, C D Eb D C, repeats.

You may use one of these song or one of your own.

IMPORTANT:

*****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!***

*****You can create a new Code Composer workspace for each lab***

*****Or, clone the template demo project from Lab 0 by selecting the that project followed by Edit>Copy, Edit>Paste, and entering new name***

/-/

SAVE OFTEN! ***SAVE <u>MORE</u> OFTEN!*** **NOW IS A GOOD TIME TO SAVE!**

/-/

Writing a High Quality Lab Report:

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

1) *Cover sheet* = An appropriate, professional-looking cover sheet with the title of the assignment and the names and ECE box numbers of each student.

2) ***YOUR SIGN-OFF SHEET!!!*** Most of the points for the lab are listed on the sign-off sheet. If you do not submit it then you will not receive the points.

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

4) *Discussion and Results* = As many pages as it takes (without padding!). In this section you should thoroughly discuss what you did in each part of the lab. You should describe the approach you took to solving any problems. ANSWER ALL QUESTIONS highlighted in yellow in complete sentences. Make sure the grader can FIND your ANSWERS! Include pseudo code descriptions and/or flow charts for any code developed. Results should also be thoroughly discussed. Any measurements should be tabulated, questions should be answered completely, figures should not be hand drawn, snippets of code may be included where useful but a full code listing is not required or desired. Instead, code should be submitted electronically (see below).

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

6) *Appendices* = Include any relevant raw data sheets or reference material, etc. The SIGNED-OFF PRE-LAB assignments (on-paper) from BOTH students should be included in appendices. Do not attach a code listing. Code is to be submitted on-line.

Submit Code on-line:

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: *ece2049a18_lab2_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 "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. You should not need to change which files are selected.

6. Click the "Browse" button, find a location to save the archive (like your M 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 ***myWPI*** website. Click **Lab 2 Code Submission**. Attach the archive file of your project that you just created and hit the Submit button. Only one submission per team.

ECE2049 A-2018 Lab 2
Sign-off Sheet

Bonus Sign-off: **Friday 9/21/18** ***Report due***: **Tuesday 9/25/18**

Student 1: _____ ECE mailbox: _____ Lab brd #: _____

Student 2: _____ ECE mailbox: _____

YOU ARE RESPONSIBLE FOR *ALL* THE REQUIREMENTS LISTED IN THE REQUIREMENTS SECTION OF THIS ASSIGNMENT!

Pre-lab (students receive points individually) Signed paper submissions to be attached to lab report	10 10	Student 1 Student 2
Power Up and Reset to Welcome mode (Display welcome message)	5	
Timer A2 properly configured with resolution set to 5 ms or less	10	
Count down to start of game w/ Launchpad LEDs flashing using Timer A2	5	
Buzzer routines with pitch and duration (min of 8 pitches)	10	
Playback of song at least 28 notes long using Timer A2 to measure note duration	20	
Playing game with timing of button (4 buttons) presses using Timer A2	20	
Proper player humiliation on losing	10	
Proper celebration for winning	5	
Answer to TA Questions at Sign-off (students receive points individually)	5 pts 5 pts	Student 1 Student 2
Report (answering ***all*** questions from the requirements section)	50	
Total points	150	

BOTH partners MUST be present for ALL sign-offs!!

TA's signature: _____ Date: _____

ECE2049 Lab Report Grading Rubric

Format -- 5 pts

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

Is the code you submitted formatted, properly commented, etc.?

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

Introduction – 3 pts

Discussion/questions – 35 pts

Conclusion – 3 pts

Appendices – 1 pt

Professionalism – 3 pts

Spelling, grammar, neatness, presentation, etc.