ECE2049: Foundations of Embedded Systems
Lab Exercise #1 – A Term 2022

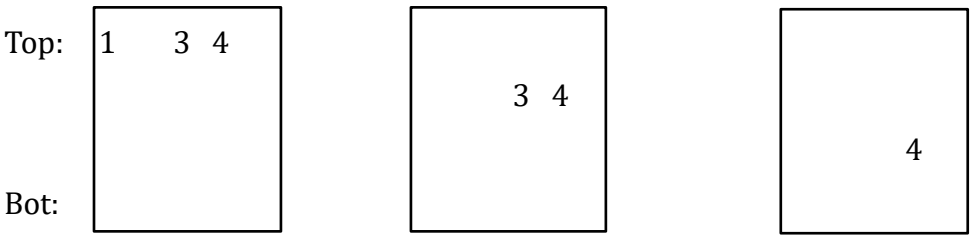
Implementing a "Space Invaders" game

Created in the late 1970's, one of the earliest and most popular video arcade games was Space Invaders. The objective of Space Invaders was simple, shoot the descending aliens before they land (or they shoot you). The original game displayed rows of aliens who "fell" down the screen unless the player shot them. As soon as the player cleared all the threatening aliens, more would appear, and these ones would be descended even more quickly!! In this lab you will implement a simplified version of Space Invaders using the MSP-EXP430F5529 lab board.

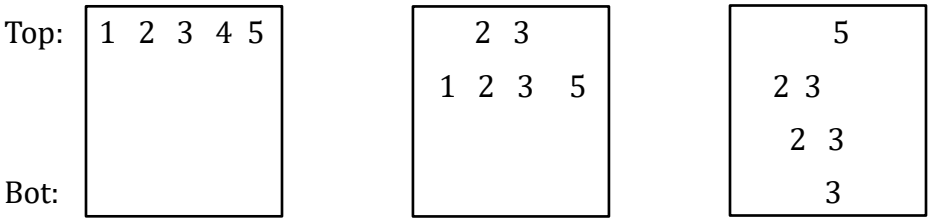
Assignment:

Staring with the I/O functions from the Lab 0 Demo project, implement a Space Invaders-like game where 1 to 5 "alien" icons appear across the top of the screen. The player "shoots" the alien by pressing the correspondingly numbered key on the keypad. If the player does not shoot the alien promptly the alien moves down the screen. After a player defeats the invasion (wins the level), more aliens appear at the top and fall more quickly. The game is lost when an alien lands on the bottom of the screen. The game should flash the LEDs, sound the buzzer and write a message to the screen when the player loses.

Here's an example of a level 1 game sequence:



This could be a more advanced level



Here alien '3' has landed. The player has lost. This would prompt a 'GAME OVER' or a 'RESISTANCE IS FUTILE!' or something accompanied by annoying flashing LEDs, buzzer sounds and such.

MANDATORY PRE-LAB Assignment: Using the main( ) of demo project as a guide write the MAIN LOOP of your Space Invaders game. Pseudo code is fine. You do not have to implement all the function calls or compile the loop. However, *each student* (as in ***both*** lab partners) MUST show the TA (and get a sign-off) a draft of your Space Invaders 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, Draw Aliens, Check Keypad, Update Aliens, 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: // Draw Aliens
    . . . .
    break;
case 2:   // Check Keypad
    . . . .
    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.

System Requirements: Read ALL requirements before starting implementation!

1) When the game is not being played the LCD should display SPACE INVADERS. The game should revert to this welcome screen after the player loses.
2) A new game starts when the "*" key is pressed. The game should give a 3-2-1 countdown on the LCD before starting.
3) At least one "alien" icon (letter or number of your choosing) should appear and begin "falling" down the screen. The illusion of falling will be created by displaying the icon for a period, then clearing it and drawing the icon in its new,

lower position. *You should not use a software delay for this!* Instead, you should measure the passage of time by counting how many times you have executed the main loop. For example, try having the alien descend one position after ~40000 loops (This implies that your loop counter variable is a long int, why?). It will take some trial and error in order to determine a suitable fall rate. ***BONUS***: Display the geometric shapes (triangles, squares) or other unique graphical alien shapes instead of an alpha numeric character for each column.

4) Once the alien(s) appear, the game should check the keypad to determine if the player has pressed the correct key. If the correct key has been pressed, then the lowest alien in the corresponding column should disappear.

5) When the last alien on the screen has been eliminated, the game should begin the next level. More aliens should appear, and they should fall faster. It is up the game designer how to implement this.

6) Although aliens should always start at the top of the screen, the column(s) in which new alien(s) appear in should be random. 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) Realize that you'll need to have some sort of array or other data structure that keeps track of the current positions for all of the aliens on the screen. There will be a fixed number of vertical positions (rows) and 5 columns in which an alien can appear. You will need to know the range of screen X-Y coordinates. You can determine this by trial and error. Figure out where (0,0) is and then what the coordinates of the opposite corner are. *Include this information in your report. It is part of your design considerations*. You will need to update each creature's Y coordinate every time the aliens descend.

8) ***BONUS***: Add a soundtrack to your game. Have the buzzer make different sounds for different events like shooting aliens or aliens descending. Just buzzing on game over is not enough. You must buzz at least 5 different pitches during the game.

9) Any new functions you write should use the port register and bit 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) Submit a zipped version of your CCS Project electronically. Instructions are given below.

IMPORTANT:

********Remember to make your project names meaningful and unique. Use revision numbers to preserve hard-won functionality <u>before</u> going on to next step! Save your work to your network 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 MORE 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. 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, 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. SIGNED-OFF PRE-LAB assignments from BOTH students should be included in appendices.

SUBMIT YOUR CODE

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: *ece2049a19_lab1_username1_username2*, where username1 and username2 are the usernames 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 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 ***Canvas*** website. Click **Lab 1 Submission**. Attach the archive file of your project that you just created and hit the Submit button. Only one submission per team.

ECE2049 A-2019 Lab 1
Sign-off Sheet

Early (Bonus) Sign-off: **Friday 9/9/22** ***Report due***: **Wednesday 9/14/22**

Student 1: _____ ECE mailbox: _____

Student 2: _____ ECE mailbox: _____

Task Max points	***Max points***	***TA's assessment***
PRE-LAB Complete at start of lab (Students graded individually) BOTH PRE-LABS MUST BE ATTACHED	10	Student 1 Student 2
SPACE INVADERS on LCD	5	
Game start with count down (on * key)	10	
Display descending aliens	10	
Check keypad and remove dead aliens	10	
Have graphical icons display "aliens" instead of alpha-numeric characters	***5***	
Play complete game (i.e. display aliens, check keypad, update aliens positions, increase speed and number of aliens)	10	
Proper player humiliation on error & reset to welcome screen	5	
Answer to TA Questions (up to 5 points per student and up to 10 points per team)	10	Student 1 Student 2
Adding buzzer "sound track" with at least 5 pitches that plat on game events. (Buzzer on Game Over is not enough!)	***5***	
Report	30	
Total points (with bonuses)	100 ***110***	

TA's signature: _____ Date: _____

***** Both Students MUST be present at Sign-Off and answer TA questions***

ECE2049 Lab Report Grading Rubric

Format -- 5 pts

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

Is your code formatted, properly commented, etc.?

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

Introduction – 3 pts

Discussion – 15 pts

Conclusion – 3 pts

Appendices – 1 pt

Professionalism – 3 pts

Spelling, grammar, neatness, presentation, etc.