CSE325 Project 2 – Press Your Luck!

Instructions

In this lab you can work in pairs with a partner if you wish, or you may work alone. If you work with a partner, submit only one file with both of your names in it. You will each earn the same number of points.

The Game

In this project you will develop a simple game using a button input, dip-switch input, the random number generator, and the 4 LEDs on the TWR-MCF52259 board.

In this game, you will play against the computer to test your reflexes. The computer will choose a random LED, display its choice to the player, then light up LED's in either a random or linear pattern, depending on the dip-switch setting. When the chosen LED is lit, the player must be fast enough to press the button to win that level. If the wrong LED is lit when the player presses the button, a losing animation is displayed on the LEDs, and the game resets. If the player pressed the button when the correct LED is lit, a winning animation is displayed, and the player progresses to the next level, where LED's are displayed faster. There is no "last level," the game just gets harder and harder until the player loses.

Requirements

1. Develop a library of functions for initializing and setting the state of the 4 user LEDs.
2. Develop a library of functions for initializing and reading the state of the DIP Switches (SW2 on the TWR-MCF52259 schematic)
3. Develop a library of functions for initializing and reading the state of the Pushbutton Switches (SW1 and SW3 on the TWR-MCF52259 schematic).
4. Develop a library of functions for initializing and reading the on-board random number generator (Chapter 6 in the IMRM. Hint: Chapter 6.4 gives an overview of how to use it).
5. Develop a library of functions for initializing and using DTIM0 as a timed-delay.
6. Implement the Game.
    a. SW2-1 (Dip switch 1) determines the mode of the game (Random or Linear Play)
        i. ON – If DIP switch #1 is on, the computer lights a random LED.
        ii. OFF – If DIP switch #1 is off, the computer lights the LEDs in sequence (e.g. LED1 lights, pauses, turns off, LED2 lights, pauses, turns off, LED3 lights, etc.)
    b. Pushbutton SW1 is the player input switch.
    c. The losing animation should light all of the LEDs at once, pause, then turn them off one at a time (e.g. LED4 turns off, then LED3 turns off, etc.)
    d. The winning animation should flash all of the LEDs together several times.

HINT: For each of the above steps, write a program that tests and verifies the behavior of what you implemented before you move onto the next step. This will make sure that when you put it all together it works the way you expect it to, and you're only debugging what you added each time, not the whole thing at once.

Watch this video for an example of the program behavior and play experience:
http://youtu.be/HftxIxua_hM

Questions

1. See the lecture notes discussing the configuration of DTIM0 for a 1.7 ms delay. If DTIM0 is initialized as described in the notes, but DTMR[PS] is set to 80 (decimal) rather than 79, then instead of delaying for ***n*** ms, the actual delay would be ***how many*** ms? (Assuming DTRR0[REF] = 0x6A3). What would the error between ***n*** and the actual delay, expressed as a percentage? HINT: Read IMRM Ch. 26.4.2
2. The microcontroller board has three tactile push button switches labeled SW1, SW3, and SW4 (SW2 is the DIP switch). SW4 is shown on p. 3 of the TWR-MCF5225X schematic, and switches SW1 and SW4 are shown on p. 5. Locate SW4 on page 3.
    a. What is the name assigned to the signal coming out of SW4 on pin 4.
    b. You will notice that some signals on the schematic are labeled with _b on the end of the signal name. What does _b mean?
    c. To what pin of the MCF52259 is this signal (from question a) connected and what is the name of this pin?
    d. Locate the description of this signal in Ch. 2 of the IMRM. What is the function of this signal. Hint: I am looking for you to quote directly from the IMRM.
3. Locate SW1 and SW3 on the board and on p. 5 of the TWR-MCF5225X schematic.
    a. What is the name of the signal coming out of pin 4 of SW1?
    b. Locate that signal on p. 3 of the schematic. To what pin of the MCF52259 is the signal connected and what is the name of this pin?
    c. Go to Table 2-1 in Ch. 2 of the IMRM. This generalpurpose I/O (GPIO) pin can function in one of three ways. What are the primary, tertiary, and quaternary functions for this pin?
    d. The quaternary function column tells you what GPIO port this pin is connected to when the pin is configured for the GPIO function. What GPIO port is this pin connected to?
    e. Go to Ch. 15 of the IMRM. Look at Fig. 15-1 and locate the port from question d. What pin of this port is SW1 connected to?
    f. Suppose we wish to program this pin of this port so we can detect when SW1 is pushed or not pushed. We first have to program what register of the GPIO module to set the function of this pin to be GPIO?
    g. Write the C statement that would write the appropriate bits to this register to configure this pin for GPIO function—using the name of this register from the MCF_GPIO.h header file. Note: When writing the bits to the proper field of the register, the bits in the other fields shall not be affected.

h. We next have to program one of the GPIO registers to configure this pin for input mode. What is the name of the GPIO register?
i. Write the C statement that would write the appropriate bits to this register to configure this pin for input mode—using the name of this register from the MCF_GPIO.h header file. Note: When writing the bits to the proper field of the register, the bits in the other field shall not be affected.
j. How would we detect whenSW1 is pushed? That is, what GPIO register would we access? Write the C statement that would set an int variable named sw1 to 1 if the button is pushed and 0 if it is not pushed.

Grading

You will be graded on the completeness and program structure of your project. Your score is out of 25 points.

Completeness: 5 points total.

All required game features are implemented and function correctly. No errors occur during compilation or execution. All questions are answered correctly.

Program Structure: 20 points total

All required libraries exist in their own source files, with their own header files. **(10 points)**

All function and variable names are clear and sensible, all functions are well commented. **(5 points)**

Program is modular and well organized. Formatting is consistent and readable. Functions are used where appropriate. **(5 points)**

Submission

Type your solution to the short answer questions using a word processor or text editor.

At the top of your document type your name and your partner's name if you worked with someone else.

Convert this document to Adobe PDF format for submission.

For the programming part of the project, make sure all source files and header files you've written contain the following information in a header comment block:

1. The name of the source code file.
2. The Lab project number and title.
3. Your name (and your partner's name)
4. Your email address (and your partner's email address)
5. The course number and name (CSE325 Embedded Microprocessor Systems)
6. The semester (Spring 2015)

Submit only the source files of your project by doing the following:

1. In CodeWarrior, with your project open, click **File->Export**.
2. In the **Export Dialog**, expand the **General category**. Click on **File System**. Click **Next**.
3. In the next dialog, click **Select All**. Enter a destination directory (like a temporary folder on your desktop).
4. Click **Create Directory Structure for Files**. Click **Finish**. The entire project will be exported to the directory you chose above in a "Project 2" directory, or whatever you named the project when you created it.

Copy your question solution PDF to the Project 2 directory, then zip this entire directory and submit it to blackboard using the project submission link.