EECE.3170: Microprocessor Systems Design I

Summer 2016

Homework 6

Due **1:00 PM, Thursday, 6/23/16**

Notes:

- You can earn an extra 10% for submitting everything by **11:59 PM on Wednesday, 6/22.**
- You must return your PICkit by the end of the exam on Thursday, 6/23.
- Only typed solutions will be accepted for this assignment, as you must submit an assembly or C file for each part. Electronic submissions of code are preferred.
- As noted in class, you may work in a group on this assignment. **Please be sure to list the names of ALL group members on your final submission(s).**
- This assignment is worth a total of 100 points.

For each part of this assignment, you will write a program to be executed on the PIC16F1829 microcontroller included with your PICkit3 Starter Kit. Each problem will be graded based on the following:

- Your source code for the exercise, written either in C or PIC assembly language. **You must submit the actual .c or .asm files—do not simply copy and paste your code into another document.**
- A demonstration of your code working correctly.
  - The key points to be demonstrated are described with each exercise.
  - For each demonstration, you may either show your working project to an instructor or provide a video of the project working correctly. Any video submissions must demonstrate all of the key points.

1. (30 points) This program should repeatedly go through the sequence below, showing each set of values on the LEDs for approximately one second before switching. A value of '1' indicates the LED is on; a value of '0' indicates the LED is off.

   0111 → 0101 → 0001 → 1001 → 1000 → 1010 → 1110 → 0110 → 0111 (restart)

   Note that:
   - After returning to the first state (0111), the sequence should restart with the 0111 → 0101 step, and then follow the remaining steps given. You should not display the 0111 state twice in a row when restarting.
   - The leftmost LED (DS1) should show the most significant bit of each step in the sequence, while the rightmost LED (DS4) should show the least significant bit. These LEDs are connected to Port C, but in the reverse order (DS4 holds the most significant bit within that port).

Demonstration (10 points): You must demonstrate at least two complete cycles of the sequence above, showing all eight steps in the sequence.

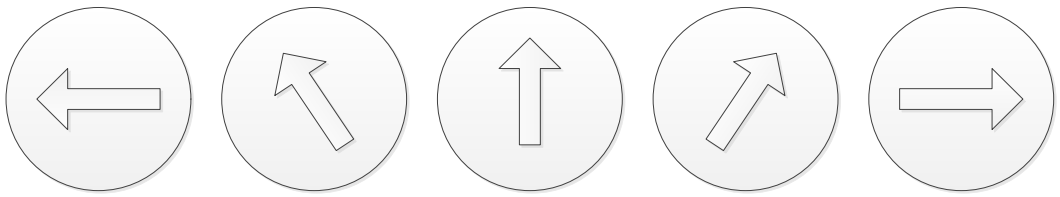
2. (30 points) This program should count the number of times the pushbutton on the board has been pressed and display that count in binary using the LEDs, with the leftmost LED (DS1) displaying the most significant bit of the count. (See the note above regarding how the LEDs are actually connected to Port C—you can't simply write the current count to the LEDs.)

   Note that, since there are only four LEDs, the maximum count value you can represent is 15. Once the count reaches 16, the LEDs should all reset to 0 and restart the count at that point. Also, please note that your program should count button presses (1 → 0 transitions), not the amount of time the button is pressed (while the input pin is set to 0).

Demonstration (10 points): You must show your project correctly handling at least 20 button presses and displaying the appropriate count on the LEDs. Also, show that when you hold the button down for several seconds, your project only counts that action as one button press.

3. (40 points) This program should rotate the LEDs using both variable delay and direction. The delay and direction should be controlled by the on-board potentiometer, which is read by the analog to digital converter (ADC) on the microcontroller.

   When the arrow on the potentiometer is pointing straight up (toward the edge of the board where the PICkit3 programmer connects), the LEDs should not change at all. (Note that it may be difficult to determine this exact position.) If the arrow points to the left of that position, the LEDs should rotate left; if the arrow points to the right of that position, the LEDs should rotate right. The speed of rotation is dependent on how far from the center the arrow is, as shown below. Please note that "fast rotation" should still be slow enough that you can see the LEDs rotating—they shouldn't change so fast that all lights appear to be on.



   In addition to using the potentiometer to control direction and speed of rotation, this exercise should use the switch to enable and disable rotation. If the switch is pressed while the LEDs are rotating, the system should pause in its current position, regardless of whether the ADC input value changes. If the switch is pressed while everything is paused, the system should resume rotating in the appropriate direction based on the ADC input value.

   Demonstration (15 points): You must show that the potentiometer controls the direction and speed of rotation in the manner described—both slow and fast rotation in either direction, as well as pausing when the potentiometer is centered. You must also show that the button toggles the mode of operation—press the button to pause the system, and move the potentiometer to show that it does not affect the LEDs when paused. Press the button again to show that rotation resumes at that point.

Important notes and hints:

- The schedule page contains links to sites where you can download the MPLAB software and the XC8 C compiler onto your own machines.
- Starting an MPLAB project from scratch can be somewhat difficult. I therefore recommend you use one of the existing demo projects from the zip file on the web page and simply reconfigure it to include your new source file(s):
    - Open the appropriate sample project (I typically use the "Hello World" one) depending on whether you're writing assembly or C.
    - Right click the source file that's part of the project and choose "Remove from Project."
    - Right click the "Source Files" folder and choose "New --> C Source File" or "New --> pic_8b_general.asm".

  If you've written code separately and just want to import it into the existing project, in the third step above, right click the "Source Files" folder and choose "Add Existing Item ... " Navigate to where your code is and add it to the file.
- Make sure you set the configuration bits for the microcontroller at the start of each program—if you fail to do so, your program won't function correctly.
- Past students have occasionally reported problems with MPLAB detecting the microcontroller. The first thing to check is that the PICkit is configured to power the board. To configure the project properly:
    - Open the Project Properties window, either by right-clicking the project and choosing "Properties," or by choosing "Project Properties" from the File menu.
    - Choose the PICkit 3 from the left side of the Project Properties window.
    - Use the "Option categories" drop-down to select "Power."
    - Make sure that the box next to "Power target circuit from PICkit3" is checked.
- I strongly encourage you to make use of the presented example code. While you do have to put some thought into these projects, they're all partially based on the examples we covered in class. Don't try to reinvent the wheel when reusing some of the same code will help you.