Three Bit Counter

Learning Goals:

- Learn how to create a circuit that sends input from a connected component to the Arduino.
- Get practice with slightly more complicated Arduino programming.
- Refresh your memory on binary conversion.

Prelab: (These are things to try out to help you be successful in the lab.)

- Set up a push button and LED with the Arduino.
- When the push button is pressed the Arduino should read in the value from the push button, and then toggle the LED (if the LED is on, it goes off; if the LED is off, it goes on).
- A very simple push button example is here: https://www.arduino.cc/en/Tutorial/Button
- Switch Bounce is a potential hazard when using a push button or similar devices. A helpful resource to remove Switch Bounce is the "Brooklyn Debounce" code in Arduino Lesson 5 at: http://www.ladyada.net/learn/arduino/lesson5.html
- Using the serial monitor to read print statements from your Arduino can be very helpful here. The serial monitor can be accessed from the Arduino IDE, under Tools -> Serial Monitor. The functions to interact with the serial monitor can be found here: https://www.arduino.cc/reference/en/language/functions/communication/serial/
- For debugging you will want to use the print functions.

Lab 2 - Due Week 5: Monday 9/18 or Tuesday 9/19 – depending on your scheduled time

Materials

- 1x Arduino
- 1x Breadboard
- 2x Pushbutton
- 2x 10k Ohm resistor
- 3x LED (any color)
- 3x 220 Ohm resistor
- Wires

In this lab, you will create a 3-bit binary counter, using 3 LEDs to represent the bits (On = 1, Off = 0). The 3-bit counter will maintain values from 0 to 7. The push buttons will change an internal "3-bit count". The LED's will display the value of this internal "3-bit count".

- One push button will increment the 3-bit count. This is the "up button".
- The other push button should decrement the 3-bit count. This is the "down button".
- The 3-bit count value will be displayed on the three LEDs in binary
    - one LED per binary digit
    - a binary digit of 1 is indicated when the LED is on

You should write code to keep track of the number of button presses and algorithmically convert that number to a binary output. Your code should NOT just be implementing a giant lookup table

(which is often written as either multiple if statements or a big switch/case statement) to translate to binary, and solutions which do so will not receive credit. You should be using the binary masking operations in your "algorithmic conversion".

Your lights should increment (or decrement) by only one each time a push button is pressed. If we continually depress the push button or if Switch Bounce occurs, the lights should not run up (or down) towards 7 (or 0) (i.e. it should only change the value once per button press).

Also, when you reach 7 and press the "up button", the value should remain at 7. Also, when reach 0 and press the "down button", the values should remain at 0.

Also, you code must account for "switch bounce". Use of the Brooklyn Debounce code is expected. See the discussion for this in Arduino Lesson 5 listed above.

Due by Lab Time Week 5: To be considered completed "on time", Lab needs to be demonstrated before the end of Lab on Monday 9/18/23 or Tuesday 9/19/2023, **AND** the code must be submitted to Gradescope, before you demonstrate your lab.

Import notes concerning the submission of your Arduino Code for the Lab:

1. Your code must be written as an .ino code file using C/C++ as the base language. Other programming languages are not accepted for CS 362.

2. If you fail to submit your code to Gradescope before you demonstrate your project to the TA, the TA will not have an entry in Gradescope to record your results. Thus, your demo would be pointless and would require you to demonstrate your lab again.

3. Updating your Gradescope submission after you demonstrate your project to TA will cause another entry in the Gradescope system which will cause any points from your demonstration to be lost. Thus requiring you to demonstrate your lab again.

4. Failure to follow these points will result in a large deduction of the points for your lab.

5. If possible, just prior to your demonstration add a comment at the end of your Header Comments with the Date, Time and Name of the TA to whom you are demonstrating. If a question comes up regarding the grading or demonstration of your lab, having that information can mean the difference between successfully resolving the question or not getting it resolved. Even if you are not able to make this change PRIOR to submitting your code to Gradescope, you should at least add this information to your own copy of your code.

What should I include with my .ino Code File?

As with any code file, it should be written in Good Coding Style: in a manner that will help other people read and understand the intent, purpose, operation of the code. So your code must include:

- Name the .ino file with your NetId and Lab Number – 3 points

    - I.E. something like: ptroy4Lab2.ino
    - where "ptroy4" is replaced with your own NetID
    - your NetID is the part of your UIC email address before the "@uic.edu"
- Header Comments (including the following) - 12 points
    - // FirstName LastName, UIN and NetID
    - // Lab x - Title
    - // Description - what is this code supposed to do?
    - // Include any assumptions you may have made, what do you expect from the hardware, pinouts, particular arduino versions, etc.
    - // References - where did you find code snippets, ideas, inspirations? if no references used say: "no references used"
- Code is well documented/formatted with comments, indentations, and descriptive variable names - 10 points
- Actual code - the functions in the cpp/ino file - 25 points

Total: 50 points per lab based on the code file submitted.

So, if you just submit working code with out doing a demo, you will only get half of the total allocated points.

Academic Integrity Guidelines:

This is an individual assignment, and should be completed on your own. You may use any resources linked from this lab, or posted by the professor or TAs on piazza/class web page/etc. You should not look at any other internet resources for this. You should not show anyone your code, or look at anyone else's code. You are responsible for writing the code on your own that converts the input to binary which lights up the proper LEDs.

Late Policy

- Late Submission 1: within 1 week late
    - Demonstrated after the end of Tuesday Lab Time (11:50AM) on 9/19, but before the end of Tuesday Lab time on 9/26
    - NOTE: This cut-off is at NOON, NOT at Midnight!
    - 25% Penalty
- Late submission 2: 1 week to 2 weeks late
    - Demonstrated after the end of Tuesday Lab Time (11:50AM) on 9/26, but before the end of Tuesday Lab time on 10/3
    - 50% Penalty