Lab 5 – Multiple Inputs and Outputs

Learning Goal:

- Be able to have your Arduino do two unconnected things at the same time, using multiple inputs and outputs.
- Be able to read multiple analog inputs.

Lab 5: Due Monday 2/19/24 on your registered lab time

Materials

1x Arduino
1x Breadboard
4x LEDs (all same color if you can)
1x Photoresistor
1x Potentiometer
1x Passive buzzer (Piezo buzzer)
4x 220 Ohm resistors
1x 10k Ohm resistor
1x 100 Ohm resistor
Wires

Optional Materials

1x Stepper or Servo Motor
1x Control Unit for the type of motor being used

Lab

This lab we will be experimenting with multiple inputs and outputs. There will be two parts that are to be set up on the same Arduino and running at the same time. One of the parts will have the potential for some time to wait between operations. The waiting for this part must NOT have any impact on the functionality of the other parts. Using delay( ) would impact the other part. So, delay( ) is not allowed for this lab.

1) You will be using 4 LEDS and a photoresistor. Put all 4 LEDs next to each (preferably the same color). As the amount of light seen by the photoresistor decreases, increase the number of LEDs that are lit. For example, if your circuit receives no light then all of the LEDs should light up. As you continue to increase the amount of light, the number of LEDs that light up will decrease. So when the photoresistor receives half the amount of light, 2 LEDS should be lit up, and when there is no light, all 4 LEDs should be lit up. Note that there are 5 levels of light being used, which matches the number of levels from Lab 4.

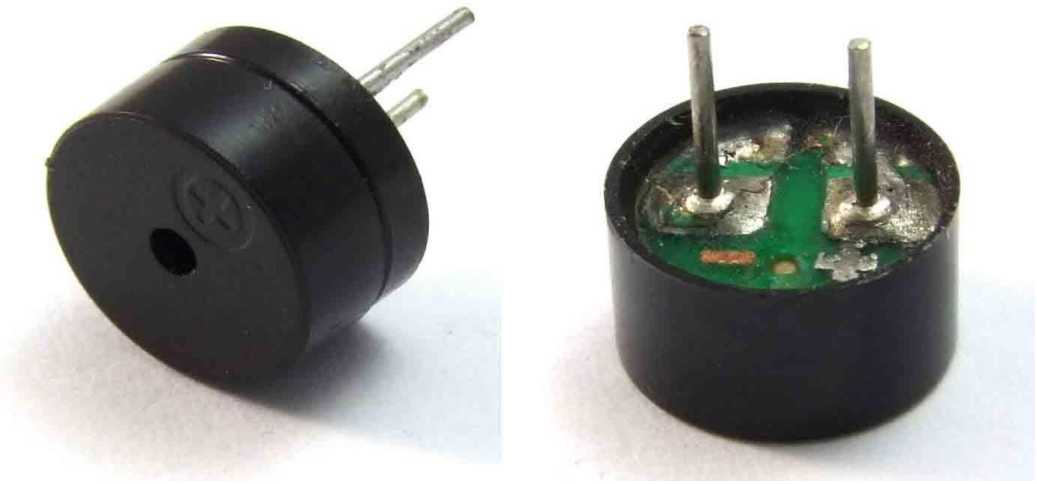
2) In parallel with part 1, you will be using the passive buzzer and the potentiometer. Use the potentiometer as an analog input and have the output that controls the buzzer come from another pin. When the buzzer creates a tone, the tone is to last for 1 second. You can pick whatever frequency for the tone you wish the buzzer to create as long as it is audible and associated with a musical note (see the tune melody examples). The value from the potentiometer is to specify the time to wait between tones produced by the buzzer. You are to determine the range of input values from the potentiometer and divide this range into 7 equal parts. The values read in are supposed to range from 0 to 1023; however, you should verify/validate the actual range for your potentiometer. Once you determine this the buzzer is controlled as follows:

a. If the value read from the potentiometer is in its lowest range, the buzzer is not to produce any tones (the buzzer is turned off).
b. If the value read from the potentiometer is in the second lowest range, the time between the tones produced by the buzzer is to be 8 seconds.
c. If the value read from the potentiometer is in the third lowest range, the time between the tones produced by the buzzer is to be 4 seconds.
d. If the value read from the potentiometer is in the fourth lowest range, the time between the tones produced by the buzzer is to be 2 seconds.
e. If the value read from the potentiometer is in the fifth lowest range, the time between the tones produced by the buzzer is to be 1 second.
f. If the value read from the potentiometer is in the sixth lowest range, the time between the tones produced by the buzzer is to be 1/2 of a second.
g. If the value read from the potentiometer is at its highest range, the time between tones produced by the buzzer should be zero seconds (the buzzer should be constantly on).

According to Wikipedia, Human hearing can range from 20Hz to 20,000Hz. Further note that the Arduino timing hardware does not allow it to produce tones lower than 31Hz.

Note that the original idea for this lab was inspired by how an intermittent windshield wiper control might work for a car. So instead of using the buzzer, the original idea was to use a stepper motor or a servo motor and control it with your potentiometer. **If you wish to use a stepper motor or a servo motor instead of a buzzer, you may.** The idea would be to have the motor turn on for a second, then have the time while the motor is off match the timing as specified above. The issue with using the motors is that motors often need a separate power supply and need additional control units to run. Also, there are many types of motors that exist and how each is controlled varied. However, if you are thinking about using a motor in your final project, you might want to consider using a motor for this lab instead of a buzzer.

This is what the passive buzzer looks like from various angles:



No specific "Pre-Lab" is given for this lab. However reading and testing out ideas from the following useful links is highly encouraged:

- https://www.arduino.cc/en/Reference/AnalogWrite
- https://www.arduino.cc/en/Reference/Map
- https://www.arduino.cc/en/Reference/AnalogRead
- https://create.arduino.cc/projecthub/SURYATEJA/use-a-buzzer-module-piezo-speaker-using-arduino-uno-89df45
- https://www.arduino.cc/en/Tutorial/toneMelody
- https://www.arduino.cc/en/Tutorial/BuiltInExamples/toneMelody
- https://www.arduino.cc/en/tutorial/potentiometer
- https://www.arduino.cc/reference/en/language/functions/advanced-io/tone/

Notes

Both parts of the lab should function at the same time. The buzzer should be able to make noise while the LEDs are showing the amount of light seen by the photoresistor.

The photoresistor should not be wired to the LEDs, and the potentiometer should not be wired to the buzzer. The Arduino should receive input from the photoresistor and potentiometer through the analog pins (A0 - A5), which the Arduino should then interpret and send the correct commands to the LEDs and the buzzer.

To be considered completed "on time", this Lab needs to be demonstrated by 4:50pm on Monday, 2/19/24. Your code must be submitted to Gradescope BEFORE you demo your lab!

Late Policy

- Lateness is determined by the time the lab is demonstrated, not when the .ino file is submitted.
- Labs that are not demonstrated get a score of 0.
  - -50% for no demonstration
  - -50% for being late
- Late Submission 1
  - Demonstrated before 11:59pm Thursday after Lab Due Date
  - 25% Penalty
- Late submission 2
  - Demonstrated between Friday and Monday after Lab Due Date
  - 50% Penalty
-

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
  - I.E. something like: ptroy4Lab2.ino
- Header Comments (including the following)
  - // 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
- Actual code - the functions in the cpp/ino file

Academic Integrity Guidelines:
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. This is an individual assignment, and should be completed on your own. You should not show anyone your code, or look at anyone else's code.