CS 362 Lab 4 Spring 2024

Analog Input: Photoresistor (LDR – Light Dependent Resistor)

Learning Goals:

- Be able to convert analog sensor input to digital information.
- Conducting experiments to determine valid ranges of values for specific equipment

We want to start incorporating more complicated devices into our circuits.

For Lab 4, your design will incorporate the light sensitive resistor photocell and display relevant information about the photocell value onto the 16x2 display.

Note that Lab 4 will also be used as the base for Homework #4 – "Lab Report". You should be aware of the requirements for Homework #4 – "Lab Report" before you disassemble your Arduino you demoing it.

Prelab:

Get your photoresistor wired correctly and have the "value" read from the photoresistor output to the serial monitor of the Arduino software. As shown here: https://www.circuitbasics.com/how-to-use-photoresistors-to-detect-light-on-an-arduino/

A slightly different schematic for wiring a photoresistor is at:
https://www.instructables.com/id/How-to-use-a-photoresistor-or-photocell-Arduino-Tu/

More information about photoresistors and their operation can be found at:
http://learn.adafruit.com/photocells

Lab 4: Due Lab Week 6: Monday 2/12 depending on your scheduled lab time

Create a circuit and program that will use a photoresistor and the 16x2 display.

The display should state the relative amount of light in the room as one of 5 predefined text values based on the value read from the photocell. The 5 predefined text values are:

- dark
- partially dark
- medium
- fully lit
- brightly lit

These values are to be displayed on the top line of the 16x2 LCD display.

On the bottom line of the LCD display, you are to display the number of milliseconds since that reset of the Arduino. This value must be continuously updating. Thus, if your code needs to wait when doing other operations, use of delay( ) will violate the above requirement. When your code needs to wait, you need to implement your wait using millis( ) as shown in the example BlinkWithoutDelay:

- https://docs.arduino.cc/built-in-examples/digital/BlinkWithoutDelay/

Examples showing the LCD display to update time values can be found at:

- https://www.arduino.cc/en/Tutorial/LibraryExamples/HelloWorld

Adding in wait code is not uncommon with a Light Dependent Resistor as they can be slow to react to changes in light. Also note that some of the use of a wait code shown in examples is to slow the reporting of values produced to make the output more readable.

Experiment with your Photoresistor:

The values received from two different photoresistors can vary greatly in the same lighting conditions. This is due to the nature of the physical differences that always occur during the manufacturing of photoresistors. Also, the various how-to pages use many different resistor values for the resistor between the photoresistor and GND. I have seen resistors values range from 220Ω to 10KΩ. That is a huge range!

So, you will need to determine the range of values the photoresistor provides under different lighting conditions. From this information, you will then need to make reasonable ranges for the values used in your code.

Thus, you will need to perform a series of tests using your photoresistor in multiple lighting conditions to see what are the actual values produced. These tests should also involve different resistor values for the resistor between the photoresistor and GND. After this testing has occurred, you should be able to determine the reasonable ranges needed for this lab.

Use of the Arduino Serial.print( ) library function can help you in setting up these tests.
https://www.arduino.cc/reference/en/language/functions/communication/serial/print/

As ideas on your tests, shining the cell phone flashlight at the photocell should give a reading of "brightly lit", while "fully lit" should be the result of being in the open of normal room lighting. The information on your experiments and how you determined your values become an important part of Homework #4 – "Lab Report", so keep track of what you are doing for this part!

Submission and Late Submission of your Lab to Gradescope

Your code must be submitted to Gradescope BEFORE you demo your lab!

Labs demoed after 4:50pm on Monday 2/12 are considered late.

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.