**420-N35-LA: IOT I (Fall 2020)**

Lab 3/4 – Creating a Project from A to Z

Date due: See Moodle assignment for due date **Description and Purpose**

This lab introduces you to Arduino programming and design, as well as source control. Basically, this is the full cycle you will use if you want to develop an Arduino solution in a professional way and even it over YouTube if you want.

You will use the Fritzing application to design your application before even starting. You will develop a small fritzing technical diagram as a base.

Then, you will use the standard “Arduino” application which gives you an integrated programming environment for the Arduino platform (also made by Arduino).

Finally, you will save your Arduino code to GIT, using classical command-line commands.

This lab will hold more weight than the other labs you have done. Basically, this lab counts for the value of two labs due to the amount of work needed. Also, you will have more time than usual to complete this lab. **Learning Objective(s)**

1. Use Fritzing to make a small diagram
2. Implement an interesting project using sensors and/or actuators

# Procedure

## PART 1 – Decide on what you will build

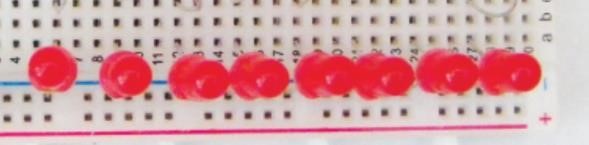
In class, we saw a way to read analog data from different devices with the Arduino, also how to light LED’s. Also, by the time you read this, you should see at least one way to interact with things like DIGITS (arrays of LED’s).

The application should read in values from a sensor (of your choice) in order to do something with this data (read on). The data being read should be ANALOG, meaning that it comes in as a value from 0 to 1023 (Arduino Uno).

Common analog choices are: Variable resistor, Photoresistor, Resistor(s), Thermistor, Sound sensor, Potentiometer

The application should then do something with the data involving something more complex than just 1 LED (RGB LEDs, arrays of LEDs, a digit, a LED 2D array).

As a minimum, here are some suggestions.

1. A series of LED’s which react to sounds. (eg: a sound level meter)
2. An RGB LED which changes colour depending on the level of light or sound. Perhaps switching to RED when the noise level is too high.
3. Anything else **OF YOUR CHOICE** that demonstrates analog INPUT and output consisting of MORE than just one LED.
4. Anything built which shows extreme creativity above and beyond – WILL gain 1-2 bonus points. This bonus allocation is at the discretion of the instructor and not arguable 😊. Creative projects generally are projects you can’t easily find as “tutorials” already on the web.

## PART 2 – Fritzing design

Now that you’ve decided on what you want to build, use Fritzing to illustrate it. If you’re not comfortable drawing the project before building it – you can build it first and debug it – then you can diagram it later – your choice.

Use Fritzing to build the closest representation you can of your project. Make it so that someone else can easily build the project from scratch with your prototype’s design. Try to illustrate it as much as you can and show notes and comments on it (your instructor needs to learn also how to make better diagrams). Paste your diagram in the area below and paste your own comments and notes about the diagram under it – anything you found interesting about it, or any special things I need to know that were “strange” in the build, etc. If you feel your project merits that “above and beyond” special quality, here’s where you mention that also.

Paste the diagram here.

|  |
| --- |
|  |

Your notes can be included here.

|  |
| --- |
| The aim of this build is to act as a motion sensor that alerts the user via a RGB LED and outputs the distance relative to an object on the LCD display.  This build is one that doesn’t need to be as involved as it is in the diagram. Really, the sensor w/the LED is enough to achieve what this build is for, a motion sensor. However, including the LCD w/an adjustable contrast makes it unique in my opinion.  The LED in the diagram is different from the one I ended up using, as such one must be wary of the pin setup for their own RGB LED (Red goes to pin 11, Green goes to pin 12 & Blue goes to pin 13).  Additionally, the LCD display may ONLY WORK if the wires are setup for it EXACTLY as it is shown in the diagram.  (GND/VSS goes to negative pin, 5V/VDD goes to a positive pin, VO goes to the middle pin of the potentiometer, RS goes to a pin 7, R/W goes to a negative pin, E goes to pin 6, DB4-7 go to pins 5-2 respectively, A goes to a 220ohm resistor and over to a positive pin & K goes to a negative pin)  I think this build is a great example of a more advanced setup for reading information from a sensor since we use one sensor and output the results to another display, removing the need for a serial monitor. |

## PART 3: Setting up the Arduino hardware project

Set up the Arduino project yourself and implement it (build it). I would like to see a video of it once it’s done – which should be uploaded in the ZIP file you create (see submission instructions).

## PART 4: Using GIT

Since the Arduino environment does not include a built-in GIT management system, you’ll have to manage it manually. We will use the command-line to control our GIT repositories. It sounds more complicated than it is, it’s a very simple process. Note: This works for Windows and Linux, but I cannot confirm it will work via the Mac terminal (but I can’t see why not).

## Step 1

Choose how you’d like to save your projects (the structure of your repository). Your repository structure is up to you.

Choose a folder on your computer where you store your code – which you probably already have.

## Step 2

In a windows CMD or Terminal: Find your code folder created above. The easiest way to do this, is via the Arduino IDE by selecting File-Preferences and observing the “Sketchbook Location” textbox value.

Make note of the location, because you’ll use it in the following steps.

On a typical Windows machine, your location might look something like:

C:\Users\{yourname}\Documents\Arduino.

If you want to change the location to somewhere else (like a D: drive) now is the time to do it. I personally keep everything on a drive separate from the operating system in case the OS is damaged.

Create a directory for your project and save your .ino file in that directory.

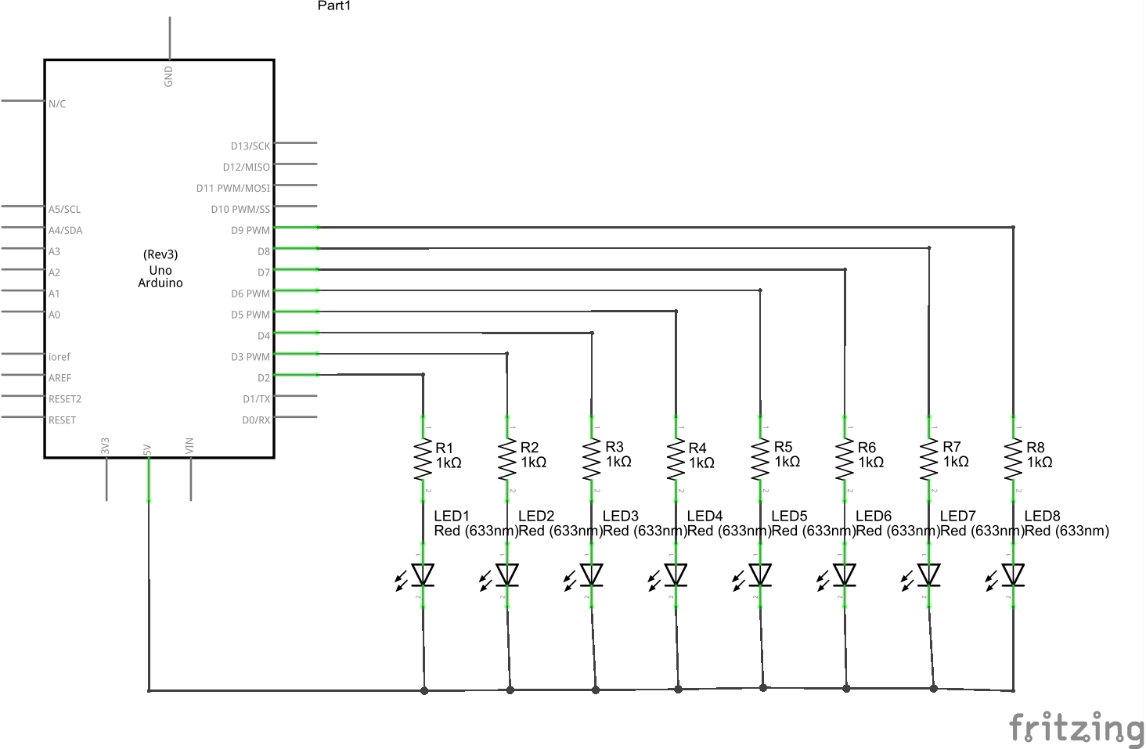
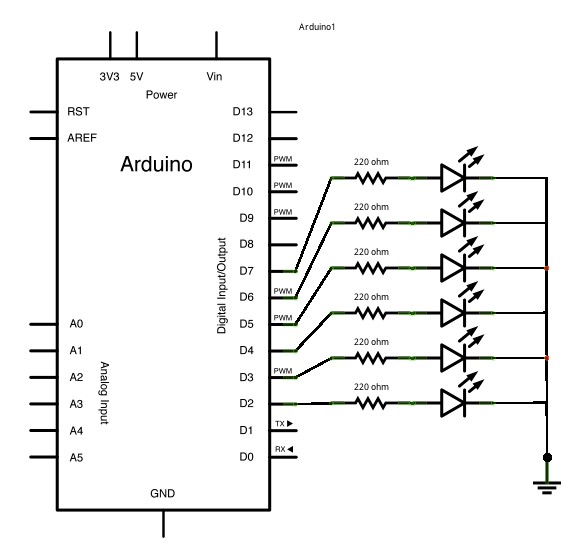
Add any files you might want to include there (it might be nice to include a fritzing file, a howto, or anything else you might think helps.

# Resources, Reference, Please See

**Arduino IDE** <https://www.arduino.cc/en/Main/Software>

**Multi-LED Setup**

How to connect a series of LED’s to Arduino (Schematics) – Experiment with your resistor values to get the best brightness desired, use the formula if necessary based on 5V.



# Submission Procedure

To submit your work, ZIP the following items, and upload the zipped file to Moodle before the deadline.

1. A screenshot (using snipping tool is fine) of your Fritzing project.
2. Your fritzing file.
3. Your Arduino Sketch
4. A SMALL video of your project working (remember there is a limit of 500MB for the zip file. It should show all the basic operation of your project (like a small movie trailer – to the point).

# Grading Scheme / Rubric

The project will be graded on 10.

Submitting everything, done as instructed guarantees you 6 points.

2 points are given for quality of coding.

2 points are given for the creativity of the project.