Phase II, Lab 1 – Arduino Basics

INSTRUCTIONS

Complete the exercises below and upload them to Canvas as a single MATLAB script file using the naming convention "ENGR131_21S_Lab##_abc123.m", replacing abc123 with your Case ID, and ## with the two-digit lab number.

For example, if Dr. Williams were submitting Lab 4 it would be ENGR131_21S_Lab04_mrw8.m

For your script, please perform the following:

1. Separate each question into separate, runnable sections using the "%%" comment notation.
2. You may use the code and notes from class, the textbook, MATLAB's documentation, and anything you find using Google to solve these problems.
3. Use comments as appropriate to indicate your thoughts and how your code works (or is supposed to work). This is 5 points (10%) of your grade.

OBJECTIVE

The Aim of this lab exercise is to introduce the Arduino microcontroller and to demonstrate several key capabilities of the device including:

- Programming basics and code structure
- Digital input and output (I/O)
  - Read the state of a switch
  - Turn on an LED
- Analog I/O
  - Read the voltage through a photoresistor.
  - Output a pseudo-analog voltage to make variable brightness LED
- Writing to the Arduino Development Environment (ADE) Serial Monitor, Plotter, and Matlab.

The Arduino open-source microcontroller in its many variations has become a mainstay of hobby users and allows for rapid prototyping of logic-based devices. While it is not the "heavy lifter" of the Raspberry Pi, it is well-suited for robotics and small-scale data collection, transmission, actuator (motors and servos) control, and endpoint IoT control.

A programming concept we have not explored much Matlab is the use of a variable to hold on to the state (on or off) of the condition of something from one iteration of a loop to the next. These are often referred to as "flags". By convention, when the condition becomes true, the flag is set to 1 (the event or condition occurred) and when the opposite happens, the flag is set to 0 (the condition or event went back to a not occurred state). In this exercise, when a button is pressed, the flag will be set to 1, when it is released, it will be set to 0.

One thing to keep in mind as you work with microcontrollers is that they are limited in terms of how much current they can source (provide) or sink (take in). For this reason (and to preserve the LED) a resistor is used in series with the LEDs in this lab. Similarly, the pushbuttons are also connected through a resistor to limit the input current to the digital I/O pins. Since a button is essentially just an intentional "short", connecting your 5V pin directly to the input pin would have a harmful effect on your device.

Resistors have a resistance (listed in ohms, Ω) that ranges from 1 Ω to many megaohms (MΩ). The resistance of each resistor is encoded in colored stripes on the body of the resistor. Please refer to **ENGR 131 ZZZ-RF-030-21-A (Resistors)** for more information on resistor color codes. As these can be hard to read on very small resistors, the resistance is also shown on the paper strip they come packaged in.

DELIVERABLES

- Demonstrate the circuit to your TA, making sure to show the following:
  - Question 1
    - The green LED light up when the button is pressed
    - The button state (on or off) and the number of button presses is shown in the Serial Monitor
  - Question 2
    - The blue LED dims and brightens in response to the light on the photoresistor
    - The data from the photoresistor is shown on the Serial Plotter
- Submit on Canvas a zip file containing:
  - Your Matlab script
  - Your saved plot showing the data from the photoresistor

EQUIPMENT & MATERIALS

- 9V Battery
- Resistors
  - R_1 = 330 Ω (2)
  - R_2 = 1 kΩ (3)
  - R_3 = 10 kΩ (2)
  - R_4 = Photoresistor
- Arduino UNO
- Proto board
- Battery (9V)
- Battery Connector
- Pushbuttons (2)
- LEDs (1 green, 1 blue)
- Jumper Wires
- Documentation
  - ENGR 131 21S-LC-001-20-A (Microcontrollers)
  - ENGR 131 ZZZ-VD-030-21-A (Operation Demo)
  - EBME 360 21S-IN-030-21-A (Arduinos) – this document

QUESTIONS

There are 3 questions for this lab.

1. Digital Input/Output & Writing to the Serial Monitor (30 pts)

For this question, you will program the Arduino to the following:

1) Turn on an LED when a button is pressed
2) Count the number of button presses
3) Report the state of the button and the number of button presses back to the Serial Monitor in the ADE.

To do this, do the following:

A. Build the physical circuit shown in Figure 1. A photograph of the complete lab exercise can be seen in Figure 3.

B. Define constants (int type) for the button (D2) and LED (D4) pins.

C. Define the variables (int type) for the state of the button, a "flag" to determine when to increment the counter, and the counter variable itself.

D. In the setup() function:
   1. Set the **pinMode** of the button pin to input and the LED pin to output
   2. Initialize the serial port at 9600 baud

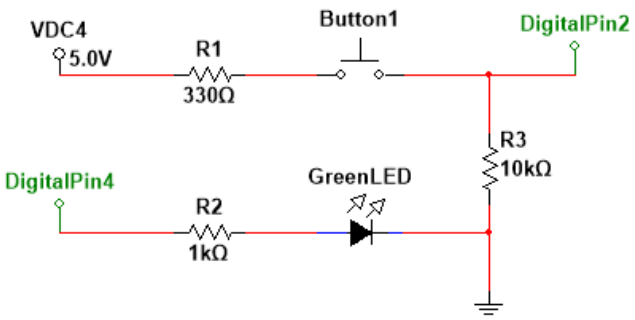

Figure 1. Schematic set up showing components and Ardunio UNO header connections.

E. In the main loop() function
   1. Use **digitalRead** to read the state of the button pin
   2. Create an **if-else** selection statement structure:
      a. Check if the button is pressed, if so, use **digitalWrite** to set the LED pin HIGH (on)
      b. Next, check that the button has transitioned from not pressed to pressed using an **if**-statement. If the flag is currently less than 1 (indicating that the button was NOT pressed in the previous loop iteration) increment the counter and set the flag to 1 (logging that the button has been pressed in the current loop iteration).
      c. For the else condition (the button is not pressed):
         i. Use **digitalWrite** to set the LED pin to LOW
         ii. Set the button flag to 0 (logging that the button was not pressed in the current loop iteration).
   3. Use **Serial.print** and **Serial.println** to report the state of the button and the value of the counter.
   4. Use a **delay** of 100 ms to give the Arduino time to do everything.

F. Download the program and open the Serial Monitor. Press the button and demonstrate that you can read the button state and count the number of presses.

G. DO NOT disconnect these components as you will need to show that this is working to the TA.

2. Analog Input/Output & Writing to the Serial Plotter (30 pts)

For this question, you will program the Arduino to the following:

1) Read the light level using photoresistor (a device that changes its resistance based on the intensity of light shining on it).
2) Turn on an LED with a brightness that is proportional to the light on the photoresistor.
3) Display the relative light level back to the Serial Plotter in the ADE.

To do this, do the following:

A. Add the components in shows in Figure 2 to your current circuit. Use the blue LED for this part of the lab. A photograph of the complete lab exercise can be seen in Figure 3.
B. Create a new program for this question.
C. Define constants for the photoresistor (A0) and LED (D3) pins
D. Define variables for the signal read in from the photoresistor and the computed brightness of the LED.
E. Similar to Q1, in the setup() function:
    1. Initialize the **pinMode** for each pin.
    2. Initialize the serial output.

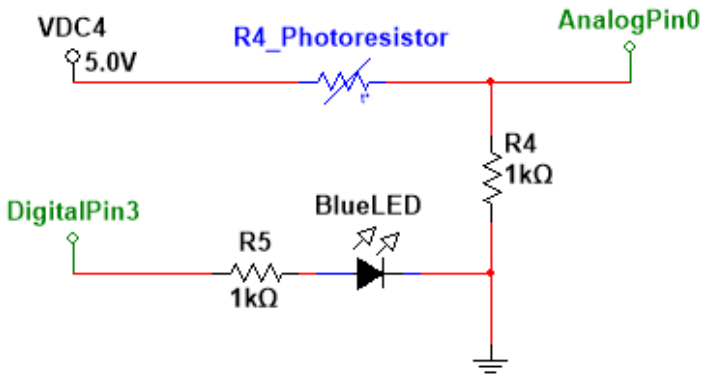

Figure 2. Schematic set up showing components and Ardunio UNO header connections.

F. In the main loop() function:
    1. Use the **analogRead** function to read the voltage on the photoresistor pin.
    2. Use the **map** function to convert the A0 voltage into a value from 0 to 255 (min and max of the analog output).
    3. Use the **analogWrite** function to turn on the LED pin to the value you computed in the previous step.
    4. Use **Serial.pintln** to report the computed value back to the Serial Plotter
    5. Use a **delay** of 10 ms to give it all time to complete
G. Download the program and open the Serial Plotter. Wave you hand over the photoresistor – does the signal drop? Shine a bright light on it (like the flashlight on your phone), does it go up?
H. DO NOT disconnect these components as you will need to show that this is working to the TA.

3. Streaming Data to Matlab (40 pts)

While turning lights on and off based on sensor readings (yes, a button is a sensor) is neat, if we want to record and analyze data, we need a way to get it from the Arduino to the host computer. We will do this using Matlab to "listen" to the same serial stream you were sending to the Serial Plotter in Q2. To do this, follow these steps:

A. Start a new Matlab script.
B. Create a serial object using the **serialport** function. This function requires two input arguments the designation of the COM port to use and the baud rate.
    1. To find the COM port, in the ADE, look in the Tools menu under Port. The COM port your Arduino is connected to will be listed.
    2. Use 9600 baud for the rate (this is the default for both the Serial Monitor and Plotter).
C. Create a **while** loop that reads from the serial port for 15 second.
    1. Use the **tic** and **toc** commands to keep track of the time.
    2. At each iteration of the while loop, record the time to a variable.
    3. Use the **readline** command to read a line of data from the serial port. Store this value to a variable.
D. Run your script while waving your hand or a flashlight over the photoresistor.

E. **Plot** the data you recorded with time on the independent axis and brightness on the dependent axis. Include an appropriate figure and axis titles. Can you see when your hand or the light was over the sensor on the plot?

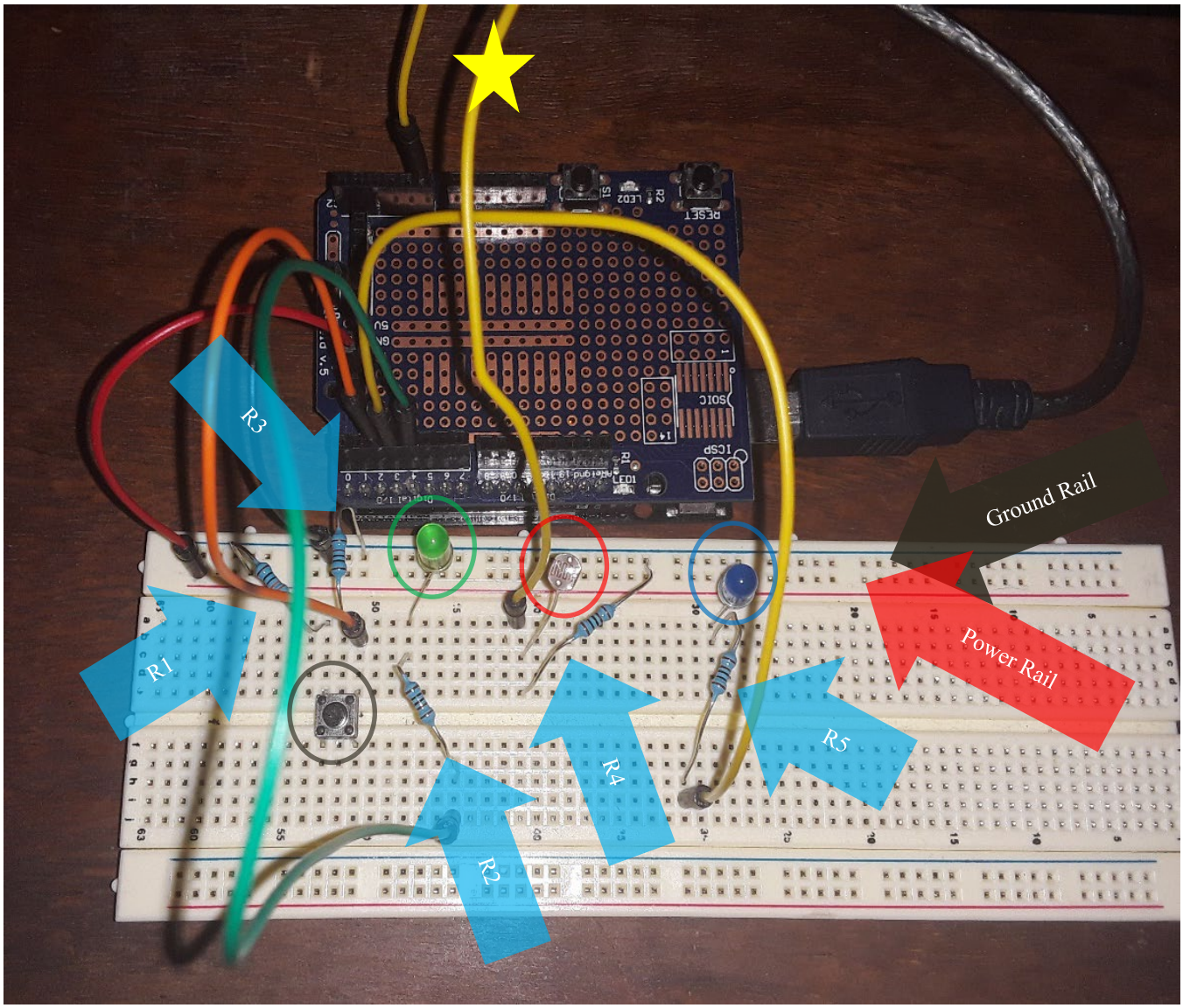

Figure 3. Photograph of complete system showing power, ground, and resistors as labeled. The button, green and blue LEDs, and photoresistor are indicated with black, green, blue, and red circles respectively. The power and ground rails are connected to the 5V and GND header using red and black wires respectively. The other connections are D2 (orange), D3 (yellow), D4 (green), and A0 (yellow, starred).

Revision	Description	Date
A	Original Document	3/10/2021
B	Specific deliverables added	3/16/2021