MEMO

Drexel University

To: Dr. Christopher Peters, ECE 303 Fall 2020

From: Dinh Nguyen

Date: Oct 20th, 2020

Re: Lab 4 Serial Communication

Purpose

The goal of this lab is to continue the study of lab 3 on the photoresistor and the relationship between the photoresistor and the LED. The lab also covers basics about serial communication, how Arduino communicates through serial terminal to another local scripts (Python or MATLAB).

Discussion

Hardware: Figure 1 shows how the photoresistor and the LED are connected in series with a 1k Ohm resistor to the Arduino board. Photoresistor will be connected to pin A0 to read its voltage in bits. LED will be connected to pin 5 to be controlled by analogWrite and duty cycle to adjust its brightness.

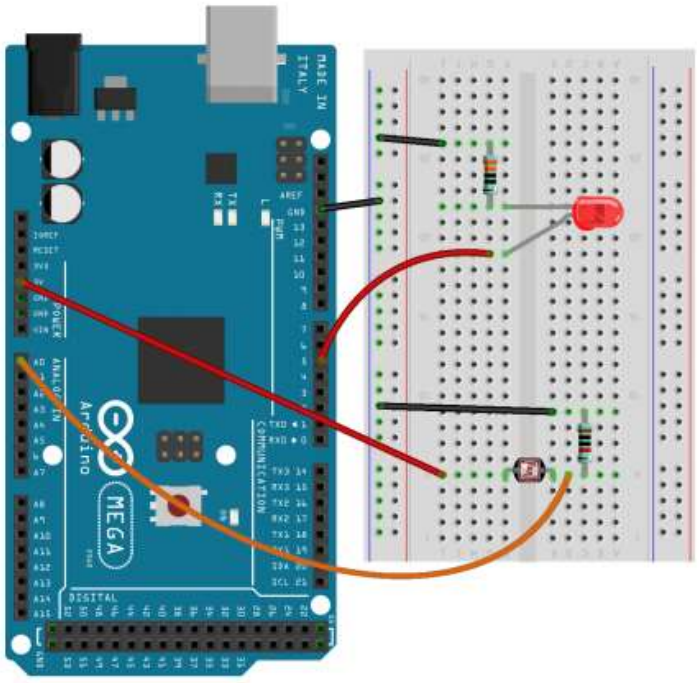


Figure 1. Photoresistor and LED circuit

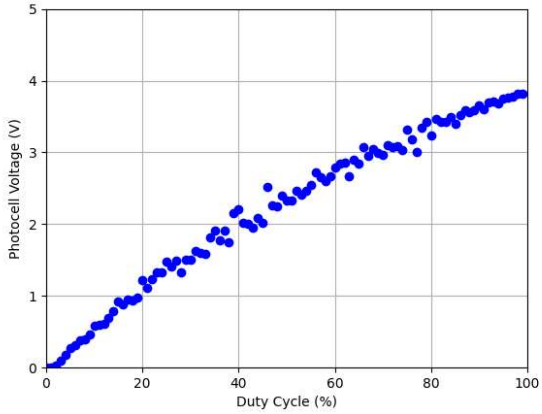
Software: In order to change the duty cycle over time, I uses analogWrite and the duty cycle will be incremented by 1% which is 2.55 bits at a time. There will be 5 reading at each iteration so retry variable will keep track of that. We will have 100 iteration in total matching 100% duty cycle. The duty cycle will

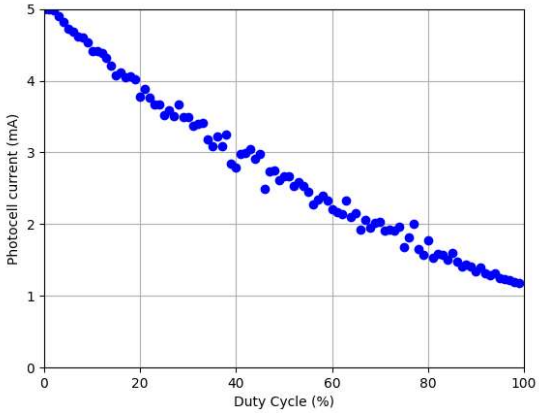
be reset when reaching 100%. Using analogRead, photocell reading will be recorded in bits and send to the Python script and from that, the Python script will read data from the serial, check if the data is valid (within +- 2 standard deviation) and then calculate the average voltage from 5 points of each iteration. Using the results, we can calculate the voltage across the photocell by dividing the reading by 255 bits and multiply by 5. Since the photocell is in series with the resistor, the voltage across the resistor in photocell circuit is 5 – voltage of photocell. On the other hands, the voltage across the LED is equivalent to 5 (the input voltage) multiplied by the duty cycle. Thus, the voltage across the resistor of the LED circuit is 5-voltage of LED. We will write the data collected to a log file and use matplotlib to develop some plots to study the relationship between LED and photocell.

To run, upload the Arduino sketch to the board and run python script by command 'python3 lab4.py'.

Plots

RED LED





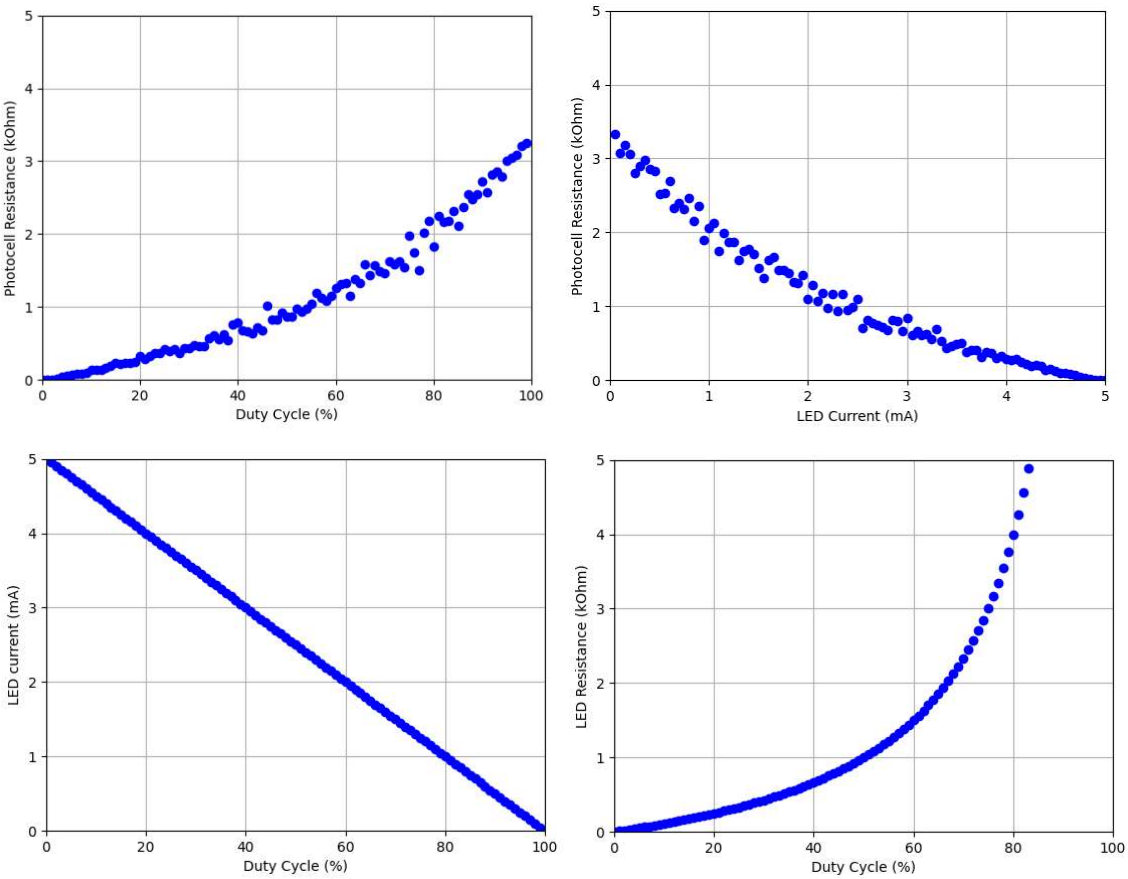


Figure 2. Analysis plots for RED LED (LED circuit and Photocell circuit)

YELLOW LED

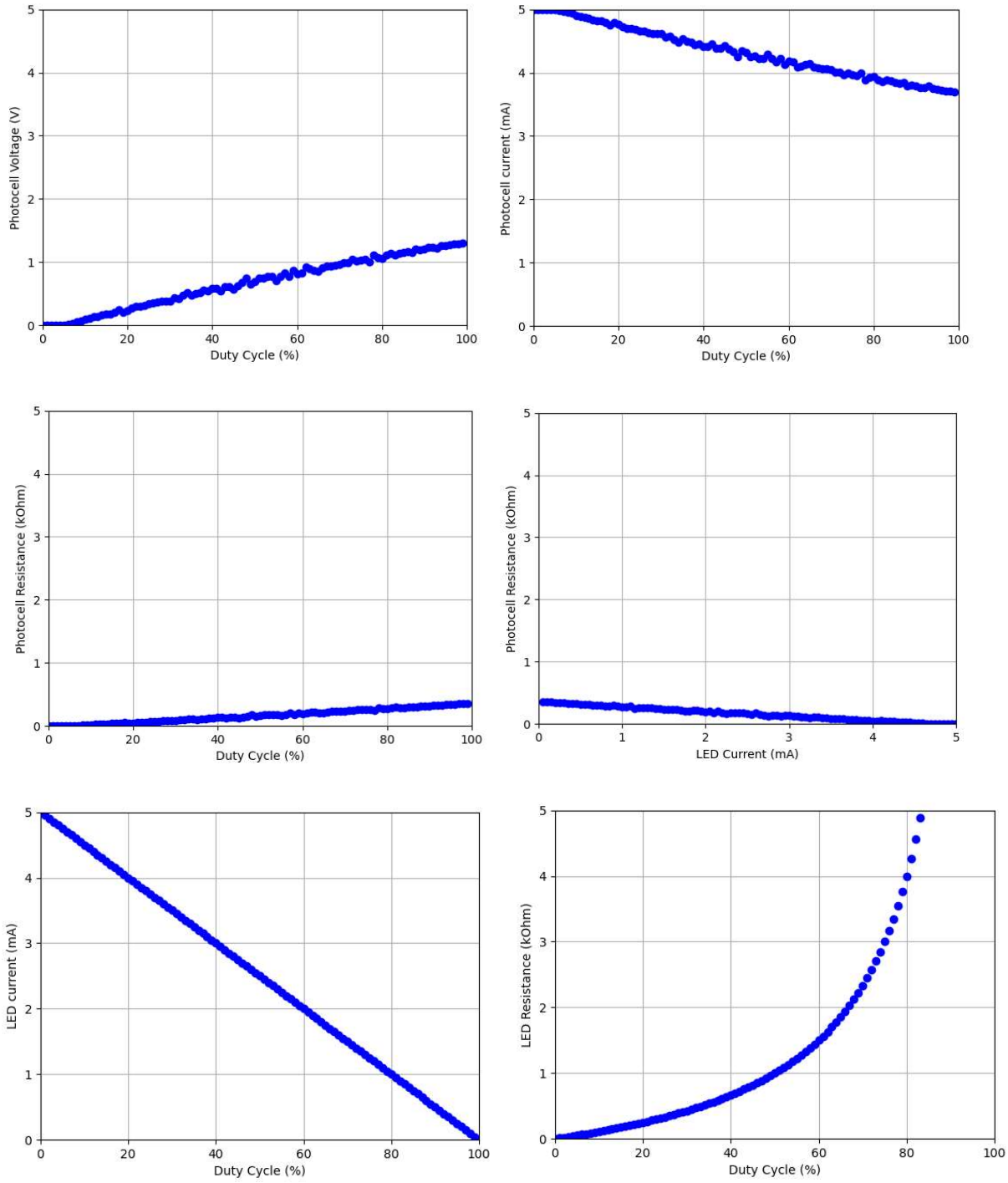


Figure 3. Analysis plots for YELLOW LED (LED circuit and Photocell circuit)

BLUE LED

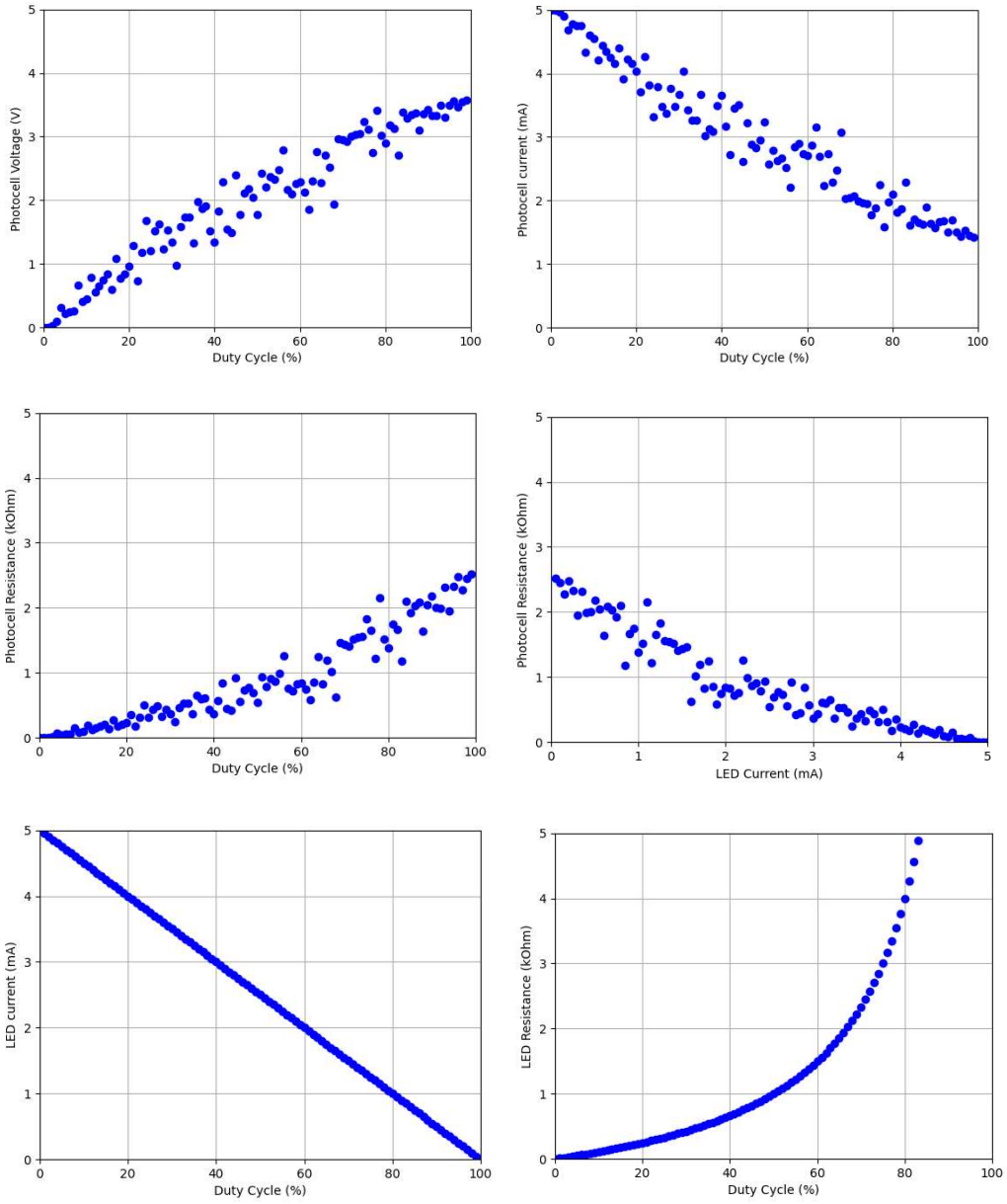


Figure 4. Analysis plots for BLUE LED (LED circuit and Photocell circuit)

GREEN LED

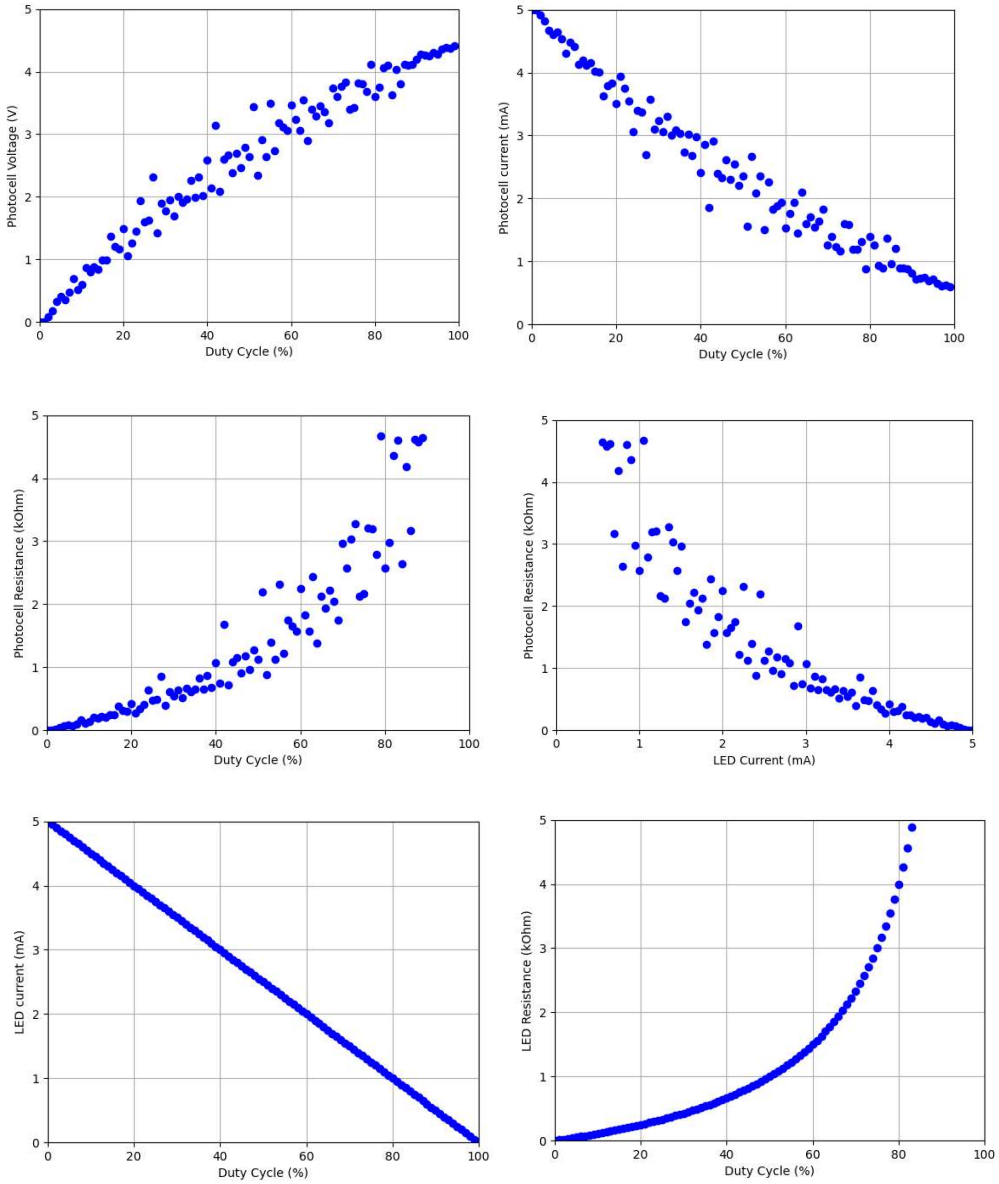


Figure 5. Analysis plots for GREEN LED (LED circuit and Photocell circuit)

Recommendation

From the plots, we can observe a lot of distortion in the data points which is due to the inconsistency of the testing environment and the performance of the photocell. A slight spot of light can easily alter the data and distort our graph. Additionally, the distance between the LED and the photocell also affect the results and it is difficult to keep the distance the same when switching between different LED. Those limitation will be responsible for the variety of the data collected. However, we can see an overall pattern among the 4 LEDs. The LED current always goes down linearly as the duty cycle go up, this is because the voltage across the LED increase, the voltage across the resistor will decrease while the resistor value remains unchanged. Same scenario will happen for the photocell circuit; however, the data is more distorted and we may not be able to observe the decrease in the current. The photocell voltage is also increasing overtime and getting more saturated when duty cycle is going to 100%. The photocell Resistor also increase with duty cycle and decrease with LED current. Moreover, the Red and Green LED seems to produce a more significant contact with the photocell compare to the other LED. Yellow LED seems to have less impact with the photocell may be due the nature of the yellow color or the limitations discussed above. Further experiments with better testing environment and more details data point (1% duty cycle margin, 10 points each iteration for example) needs to be carried out to improve the results and analysis.

Arduino Sketch

lab4

```
int photo_pin = A0;
int val1 = 0;
unsigned int val = 0;
float counter = 0.0;
unsigned int retry = 0;

void setup(){
  Serial.begin(9600);
  pinMode(5,OUTPUT);
}

void loop(){
  if (Serial.available() >0){
    val=Serial.parseInt();
    analogWrite(5,counter);
    //delay(100);
    val1 = analogRead(photo_pin);
    Serial.println(val1);
    if (retry > 4){
      counter +=2.55;
      retry = 0;
    }
    retry += 1;
  }
}
```

Python Script

```
from serial import Serial
import matplotlib.pyplot as plt
import time
import numpy

def is_valid(val,buff):
    if len(buff) <= 3:
        return True
    if val <= ( numpy.mean(buff)+ 2* numpy.std(buff)) and val >= ( numpy.mean(buf
f)- 2* numpy.std(buff)):
        return True
    else:
        return False

with Serial('COM4',9600) as arduino:
    time.sleep(1)
```

```
    xdata=[]
    voltage=[]
    LED_I = []
    LED_R = []
    Photo_I = []
    Photo_R = []
    duty_cycle=[i for i in range(100)]

    with open('lab4logs.txt', 'w') as logFile:
        logFile.writelines("Measurements:\n")

        for k in range(100):
            # get 5 points for each iteration, add to buffer, take average
            buff = []

            for count in range(5):
                arduino.reset_output_buffer()
                arduino.reset_input_buffer()
                arduino.write(b'1')
                #time.sleep(0.1)
                a = int(arduino.readline().decode("utf-8"))
                v = float(a)*5.0/255.0          # voltage on photocell
                # Error checking, if voltage too large or too small, discard
                if is_valid(v,buff):
                    buff.append(v)
                    #logFile.writelines("Iteration: %d, Arduino: %f, Buff avg: %f
. Valid" % (k,a,numpy.mean(buff)))
                    print("Iteration: %d, Arduino: %f, Buff avg: %f. Valid" % (k,
a,numpy.mean(buff)))
                else:
                    #logFile.writelines("Iteration: %d, Arduino: %f, Buff avg: %f
. InValid" % (k,a,numpy.mean(buff)))
                    print("Iteration: %d, Arduino: %f, Buff avg: %f. InValid" % (
k,a,numpy.mean(buff)))
            voltage.append(numpy.mean(buff))
            logFile.writelines("Duty cycle: %d, Average Voltage Photocell: %f \n"
 %(k, numpy.mean(buff)))
            print ("Duty cycle: %d, Average Voltage Photocell: %f" %(k, numpy.mea
n(buff)))
            # Photocell circuit
            Photo_I.append(5.0 - numpy.mean(buff)) # mA
            Photo_R.append((numpy.mean(buff)/((5.0 - numpy.mean(buff))/1000.0))/1
000.0) # kOhm
            # LED circuit
            vol_R = 5.0-(k*5/100.0)
```

```
            LED_I.append(vol_R*1000.0/1000.0)       # mA
            LED_R.append(((k*5/100.0)/(vol_R/1000.0))/1000.0)       #kOhm
        # End iteration

        # Write logFile
        logFile.writelines("Calculations:\n")
        logFile.writelines("Photocell voltage: 5 * Arduino_reading/255\n")
        logFile.writelines("Photocell Current: (5 - Photocell_vol)/1000 Ohm\n")
        logFile.writelines("Photocell Resistance: Photocell_vol/Photocell_current
\n")
        logFile.writelines("LED Current: (5 - LED_voltage)/1000 Ohm\n")
        logFile.writelines("LED Resistance: LED_voltage/LED_current\n")
        for idx in range(100):
            logFile.writelines("Photocell Voltage: %f, Photocell current: %f, Pho
tocell Resistance: %f, LED Current: %f, LED Resistance: %f\n" % (voltage[idx], Ph
oto_I[idx], Photo_R[idx], LED_I[idx], LED_R[idx]))
        logFile.close()

    arduino.close()

plot1 = plt.figure(1)
plt.xlabel('Duty Cycle (%)')
plt.ylabel('Photocell Voltage (V)')
plt.grid(True)
plt.xlim(0,100)
plt.ylim(0,5)
plt.plot(duty_cycle,voltage,'bo')

plot2 = plt.figure(2)
plt.xlabel('Duty Cycle (%)')
plt.ylabel('Photocell current (mA)')
plt.grid(True)
plt.xlim(0,100)
plt.ylim(0,5)
plt.plot(duty_cycle,Photo_I,'bo')

plot3 = plt.figure(3)
plt.xlabel('Duty Cycle (%)')
plt.ylabel('Photocell Resistance (kOhm)')
plt.grid(True)
plt.xlim(0,100)
plt.ylim(0,5)
plt.plot(duty_cycle,Photo_R,'bo')

plot4 = plt.figure(4)
```

```
plt.xlabel('LED Current (mA)')
plt.ylabel('Photocell Resistance (kOhm)')
plt.grid(True)
plt.xlim(0,5)
plt.ylim(0,5)
plt.plot(LED_I,Photo_R,'bo')

plot5 = plt.figure(5)
plt.xlabel('Duty Cycle (%)')
plt.ylabel('LED current (mA)')
plt.grid(True)
plt.xlim(0,100)
plt.ylim(0,5)
plt.plot(duty_cycle,LED_I,'bo')

plot6 = plt.figure(6)
plt.xlabel('Duty Cycle (%)')
plt.ylabel('LED Resistance (kOhm)')
plt.grid(True)
plt.xlim(0,100)
plt.ylim(0,5)
plt.plot(duty_cycle,LED_R,'bo')

plt.show()
```