Assignment 3 report

Arya Javani 2402679

Introduction

In this report I will explain the steps taken to do the third assignment along with figures.

Circuit setup

For the first step which was setting up the circuit, I followed the shematic in instruction and using thinkercad environment, the final setup looked like this:

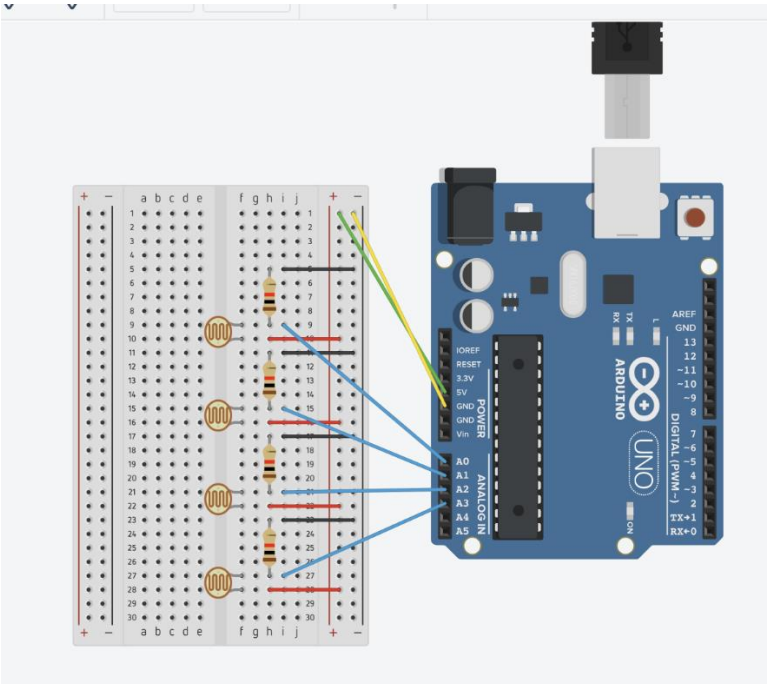


Figure 1: The circuit consist of an arduino and three light diods to its inputs

Reading sensor data

For this part using the link in the instructions, I just extended it with an array of inputs and used a loop to read the inputs.

```
int sensorPin[4] = {A0,A1,A2,A3}; // select the input

int sensorValue = 0; // variable to store the value co
void setup() {
Serial.begin(9600); //sets serial port for communicati
}
void loop() {
  for(int i=0;i<4;i++){
    sensorValue = analogRead(sensorPin[i]); // read th
    Serial.println(sensorValue); //prints the values
  }


delay(100);

}
```

Figure 2: the code for reading multiple diods

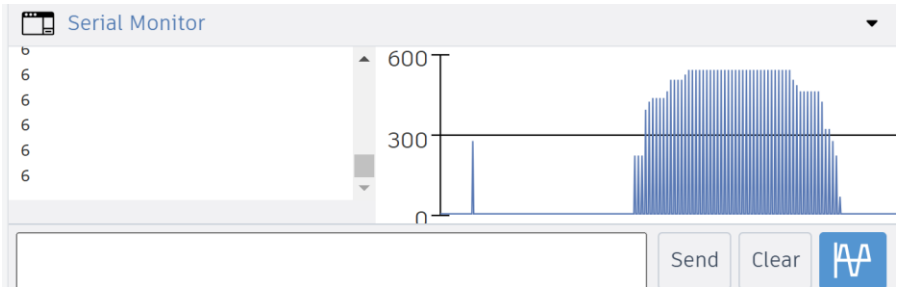


Figure 3: demonstrating input on serial port

Calculating average and variance

For this part I just simply defined two functions, one for average and another one for the variance and called them.

```
4  int Avg(int arr[],int size){
5
6
7    int sum=0;
8    for(int i=0;i<size;i++){
9
10     sum+=arr[i];
11   }
12   return sum/size;
13 }
14 int Variance(int arr[],int mean,int size){
15   int sum=0;
16   for(int i=0;i<size;i++){
17     sum+=(arr[i]-mean)*(arr[i]-mean);
18   }
19   return sum/(size-1);
20 }
```

Figure 4: Average and Variance functions

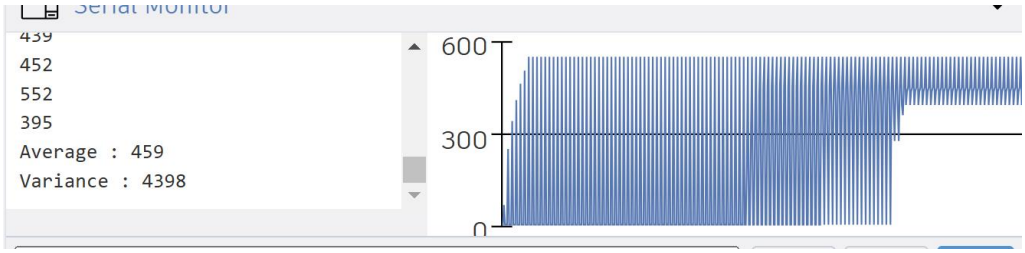


Figure 5: Output of the program for average and variance

Average and Variance for each sensor

For this part I used the previous functions. For buffers I used a 2d array for all the sensors and gave each sensor a buffer of size 100. The buffer pointers were kept in another array. Each time a data entry is read, we use the buffer pointer to store it in the correct index. To implement the sliding window, I used modulo (%) operator to in case that buffer pointer exceed 100 (or its coefficients) we will go back to first index, in this way the oldest input will be replaced with the newest without the need to actually shift the array elements.

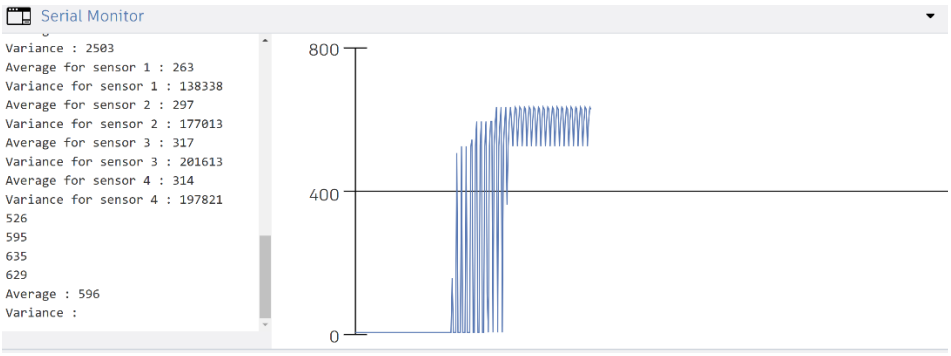


Figure 6: Calculating and showing the average and variance for each sensor

Finding faulty sensor

For this part, I used mean and average of all sensors. The idea is to use a threshold for comparing the mean and a coeff for comaring the variances. If the difference between mean of all sensors is more than the threshold of if the variance is larger or smaller than the average variance by magnitude of the coefficient, then the sensor is faulty. Keep in mind that this is very naïve and also since there are only four sensors there's not much robustness since the average of one fauly sensor can outweight the other three sensors.

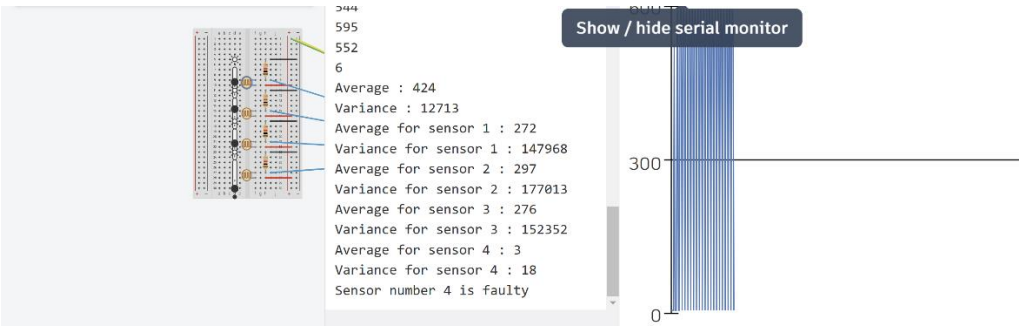


Figure 7: Example of a faulty sensor in which the fourth sensor is put to zero

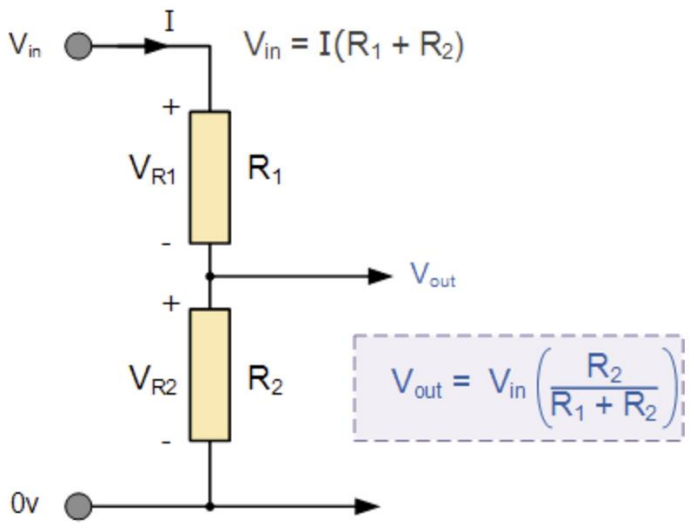
Minimum and maximum calculation

The minimum value in the input was 6 and the maximum value was 679. In Arduino documentation it is mentioned that it devides 1024 for the voltage range which here is from 0 to 5 volt. Thus :

$$\text{Minimum voltage} = \frac{5}{1024} \times 6 = 0.02 \text{ v}$$

$$\text{Maximum voltage} = \frac{5}{1024} \times 679 = 3.31 \text{ v}$$

Now that we have the voltages, using the schematic below which is similar to our circuit we can find the maximum and minimum resistance of the photo resistors.



$$V_{out} = V_{in}\left(\frac{R_2}{R_1 + R_2}\right) \rightarrow \frac{R_1}{R_2} + 1 = \frac{V_{in}}{V_{out}} \rightarrow$$

$$R_2 = \frac{R_1}{\frac{V_{in}}{V_{out}} - 1} \rightarrow R_2 = \frac{1000}{\frac{V_{in}}{V_{out}} - 1} \rightarrow$$

$$R_{max} = \frac{1000}{\frac{5}{3.31} - 1} = 1960\Omega$$

$$R_{min} = \frac{1000}{\frac{5}{0.02} - 1} = 5\Omega$$

Final thoughts

This assignment was very instructive, and all the steps were clearly explained. Mentioning the integer range in Arduino really helped with the overflowing problem for variance had it not been mentioned, it would've been a headache to discover such a problem. Another terrible bug that I faced, was stack overflow which wasn't mentioned in error log and board was just running but showing gibberish in serial output. The problem was that I used long integer for buffer (it was a 2d array with overall size of 400).