

Student Name
Eliza Gamal

Student Number
2331425

Portfolio Introduction

Workshop Activities 50% Weighting

Mini Project 50% Weighting

This completed portfolio will need submitting to Canvas by the due date.

Questions please email

Dr Sarah Slater

s.i.slater@wlv.ac.uk

Portfolio

Contents

Workbook 1.....	4
Activity 1.1: Actual voltage across 5V breadboard pins.....	4
Activity 1.2: Actual voltage across 3.3V breadboard pins.	5
Activity 1.3: Potential Divider Calculations	7
Activity 1.4: 3V Calculations from either the 5V supply or 3.3V supply	8
Activity 1.5: Voltage Divider circuit readings from Breadboard circuit.	9
Activity 1.6: LED Circuits.....	10
Activity 1.7: Current Measurement	12
Activity 1.8: Fritzing for 4 switches & LEDs.....	14
Activity 1.9: Fritzing for Number 0-7.....	15
Workbook 2.....	16
Activity 2.1: LED Flashing to show decimal number 63 as binary.....	16
Activity 2.2: 4 LED's for counting up in binary from 0 to 15.	19
Activity 2.3: Traffic Lights.....	22
Workbook 3.....	25
Activity 3.1: Circuit Diagram of Button & LED.....	25
Activity 3.2: 3 Switches & Led	26
Activity 3.3: 8 Buttons & LEDs (SWITCH STATEMENTS).....	29
Workbook 4.....	32
Activity 4.1: Serial Port.....	32
Activity 4.2: Serial Port binary to decimal.....	35
Activity 4.3: Calibrating Analogue Information.....	39
Activity 4.4: Temperature Sensor & Serial Port	43
Workbook 5.....	45
Activity 5.1: RGB Led and switches	45
Activity 5.2: Distance Sensor.....	47
Activity 5.3: 1602 LCD Display.....	49
Workbook 6.....	52
Activity 6.1: PWM	52

Workbook 7.....	54
Activity 7.1: 2 Arduinos – using Digital Pins	54
Activity 7.2: 2 Arduinos – using Serial I/O.....	55
Workbook 8.....	57
Activity 8.1: Stepper Motor Circuit Diagram.....	57
Activity 8.2: 2 Stepper Motors	59
Workbook 9.....	61
Activity 9.1: Windscreen Wiper Code using Servos & Temperature Sensor	61
Individual Project (50%)	62
Rationale	62
Timescales	62
Equipment.....	62
The Project	62
Step 1 produce a detailed description of your project.	62
Step 2 Circuit Diagram & Fritzing Schematic	62
Step 3 A Program	63
Step 4 Testing.....	63
Step 5 Conclusions	63
Layout.....	63
Marking	63
All sections carry equal marks.	63

If you prefer, you may use Tinkercad to show a component layout, rather than a circuit Diagram in Fritzing or other circuit design software, though a circuit diagram is more useful as this is what you would most likely see if you were working on embedded systems.

Workbook 1

Activity 1.1: Actual voltage across 5V breadboard pins.

In tinkercad,

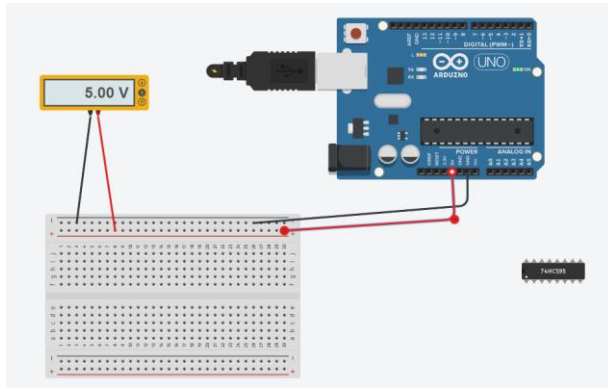


Figure 1: Voltage measuring

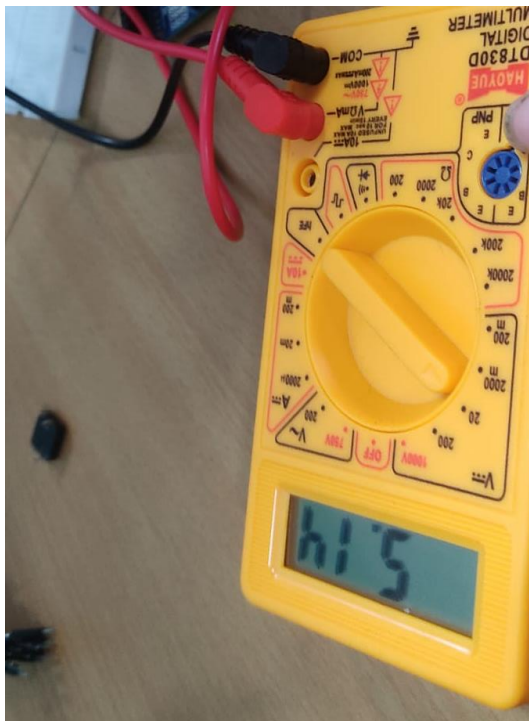


Figure 2: Voltage measuring

5.63

Activity 1.2: Actual voltage across 3.3V breadboard pins.

In tinkercad,

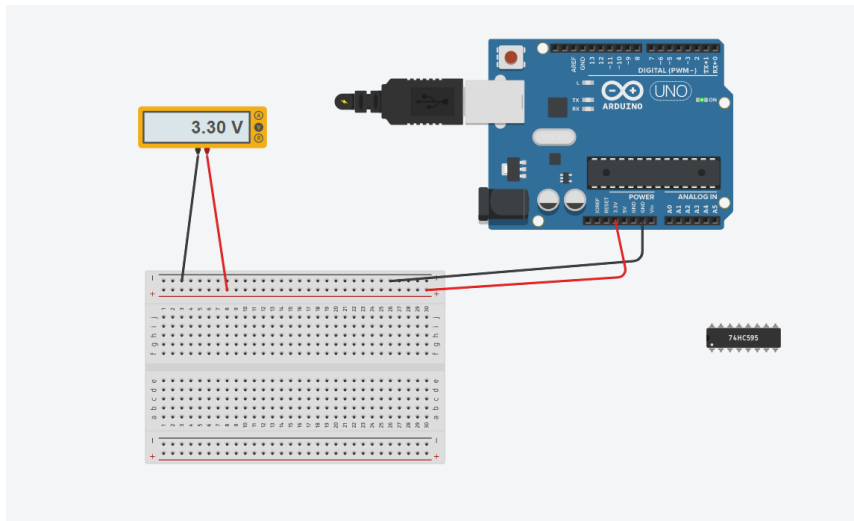


Figure 3: Multimeter measuring voltage



Figure 4: Measuring voltage

3.31

Explain in around 100 words why you think the value read by a multi meter on a circuit, may be different to a simulator value such as Tinker-Cad.

Due to the real-world parameters that a Multimeter takes into consideration, the value that it reads may differ from a simulator like TinkerCad. Electrical engineers and professionals utilize two key tools: TinkerCad and the Multimeter. While the TinkerCad computer-aided design program provides a virtual platform for circuit modeling, the Multimeter is a real tool used to measure the voltage, current, and resistance of a circuit. The real readings from a multimeter, which account for elements like wire resistance, voltage drop between components, and interference, may differ from the theoretical numbers produced using TinkerCad. For the best circuit design and analysis, it is crucial to understand the benefits and limits of both tools and employ them effectively.

If the read value is 4.84V on a 5V supply, what would be a sensible tolerance to quote, explain your answer.

Depending on how accurate the measurement must be, the proper tolerance for a certain application must be chosen. The variance is 3.2%, for instance, if the target value is 5V and the measured value is 4.84V. For some applications, a tolerance of $\pm 5\%$ would be enough, and a variance of up to 0.25V would be acceptable. Nevertheless, a tolerance of merely $\pm 1\%$ may be required in circumstances requiring great accuracy, such as with medical equipment or scientific instruments, allowing for a fluctuation of up to 0.05V. It's critical to take into account the possible repercussions of even little departures from the target value. When choosing a tolerance, it's important to consider the system's unpredictability, the desired accuracy, and the expense of reaching tighter tolerances. Ultimately, it's a balance between accuracy and feasibility.

Activity 1.3: Potential Divider Calculations

Show the working on how you achieved 2.5V

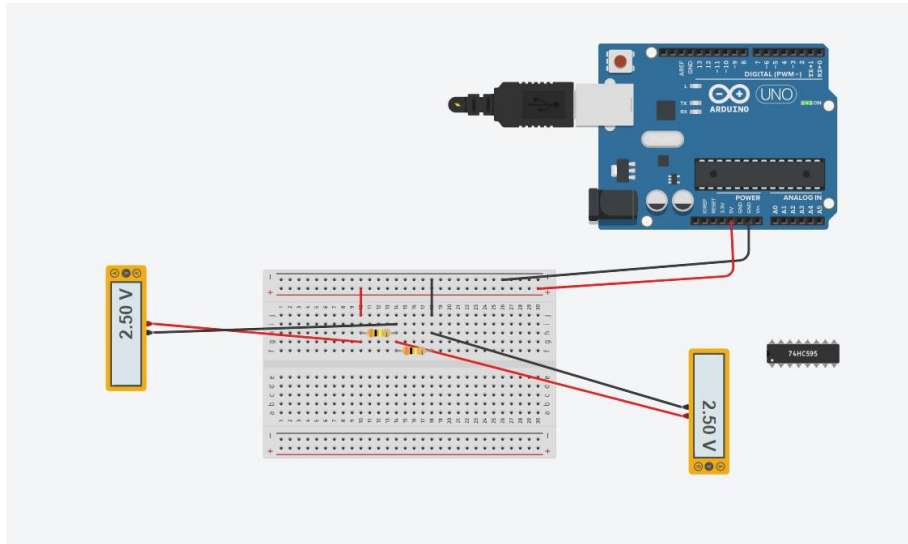


Figure 5 :Voltage divided and shown in two multimeter

By using two resistors of 2200Ohms in series we can drop the potential difference across it by 2.5 in 5V Supply

$$V * R_{\text{total}} = V_{\text{supply}} * R$$

Where,

V = Voltage to be found

R total = Total Resistance voltage across the circuit

V supply = Voltage being divided

R = Individual Resistance

$$\text{Thus, } V = V_{\text{supply}} * R / R_{\text{total}}$$

$$\text{Or, } V = 5 * 1000 / 2000$$

$$V = 2.5 \text{ Volts}$$

Thus, 2.5 volts can be obtained using two resistors of 1000 ohms each connected in series.

Activity 1.4: 3V Calculations from either the 5V supply or 3.3V supply

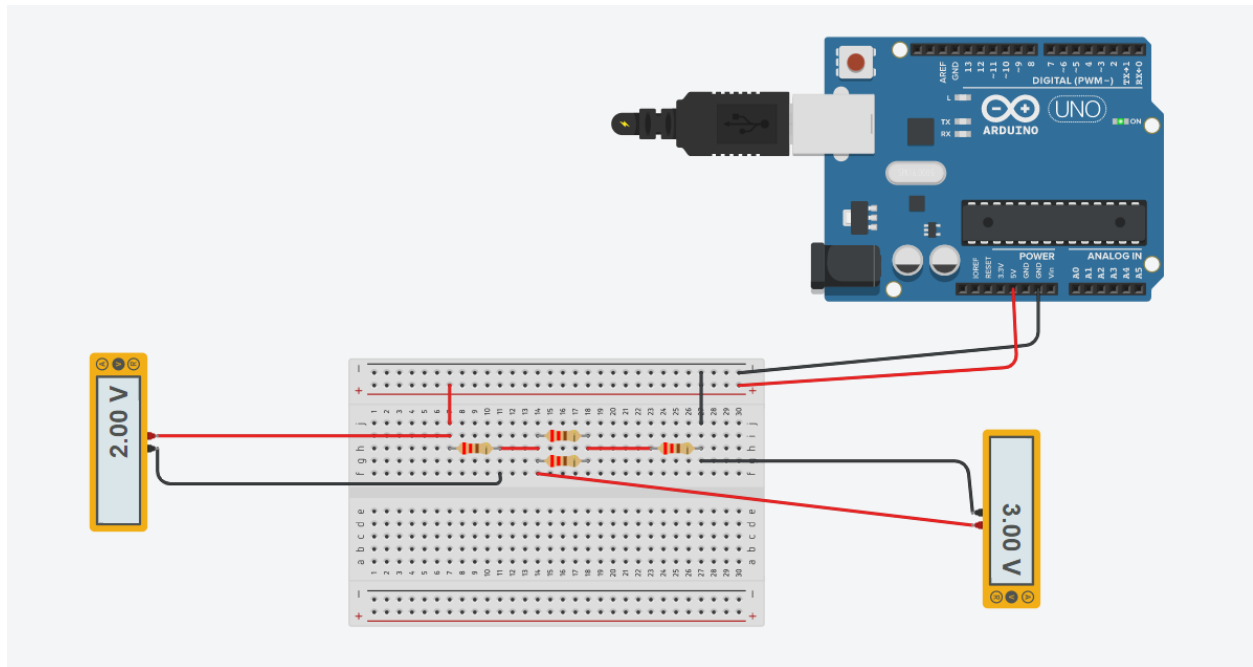


Figure 6: Calculating voltage supply of 5v or 3v

To obtain a voltage of 3V from a 5V supply with a resistor of 330 ohms, the following calculations can be made: Assuming only a 220 ohms resistor is available, the voltage obtained will be 2V using the formula:

$$\text{Voltage Out} = V_{in} \times R_1 / (R_1 + 220)$$

To obtain the remaining 1V, a resistor of 110 ohms is needed, which can be achieved by placing two 220 ohms resistors in parallel:

$$1/R_p = 1/R_1 + 1/R_2 = 2/220$$

$$R_p = 110 \text{ ohms}$$

The required total resistance (R_t) can then be obtained by combining the 220 ohms resistor and the 110 ohms resistor in series:

$$R_t = R_1 + R_p = 220 + 110 = 330 \text{ ohms}$$

Therefore, to obtain a voltage of 3V from a 5V supply with only a 220 ohms resistor available, a resistor with a value of 110 ohms and the existing 220 ohms resistor can be connected in series to achieve the required total resistance of 330 ohms.

Activity 1.5: Voltage Divider circuit readings from Breadboard circuit.

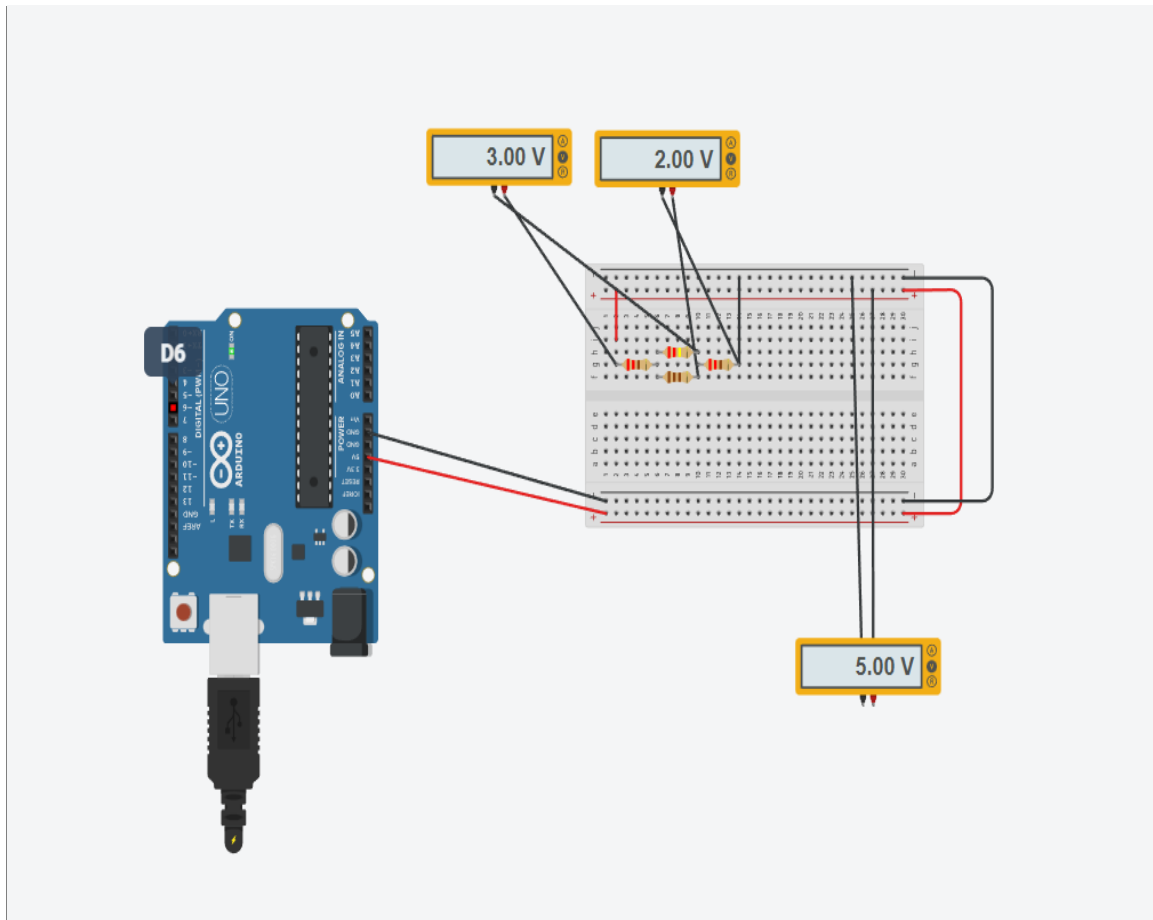


Figure 7: Voltage divider

We may create three distinct resistances using a combination of resistors, especially a series connection of a 330-ohm resistor and a 220-ohm resistor, as well as a parallel connection of two 220-ohm resistors. Hence, various voltages may be produced using these resistances. A voltage of 3.6 V is produced by the 330-ohm resistor, whereas a voltage of 2 V is produced by the 220-ohm resistor. By mixing different resistances in series and parallel arrangements, this design offers extra flexibility in attaining desired voltages.

Activity 1.6: LED Circuits

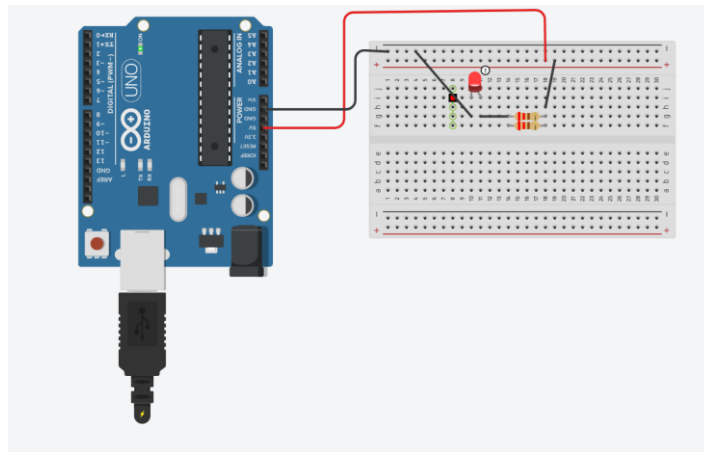


Figure 8: Circuit led

Each resistor Value

220 Ohm

220 Ohm

Total resistance Calculation

The total resistance (R_t) can be calculated by using the formula:

$1/R_t = 1/R_1 + 1/R_2$, where R_1 and R_2 are the two resistor values.

Substituting the values of R_1 and R_2 as 220 ohms, we get:

$$1/R_t = 1/220 + 1/220$$

Simplifying this equation, we get:

$$1/R_t = 2/220$$

$$1/R_t = 1/110$$

Therefore, the total resistance (R_t) is equal to 110 ohms.

Measured Resistance

211-ohm

If measured resistance is not the same, why not? If you simulated this, why might the real value be different.

Due to variations in the wire's length and cross-sectional area, the actual resistance of a resistor may vary. Certain elements could have an impact on the resistance, leading it to vary from its present value.

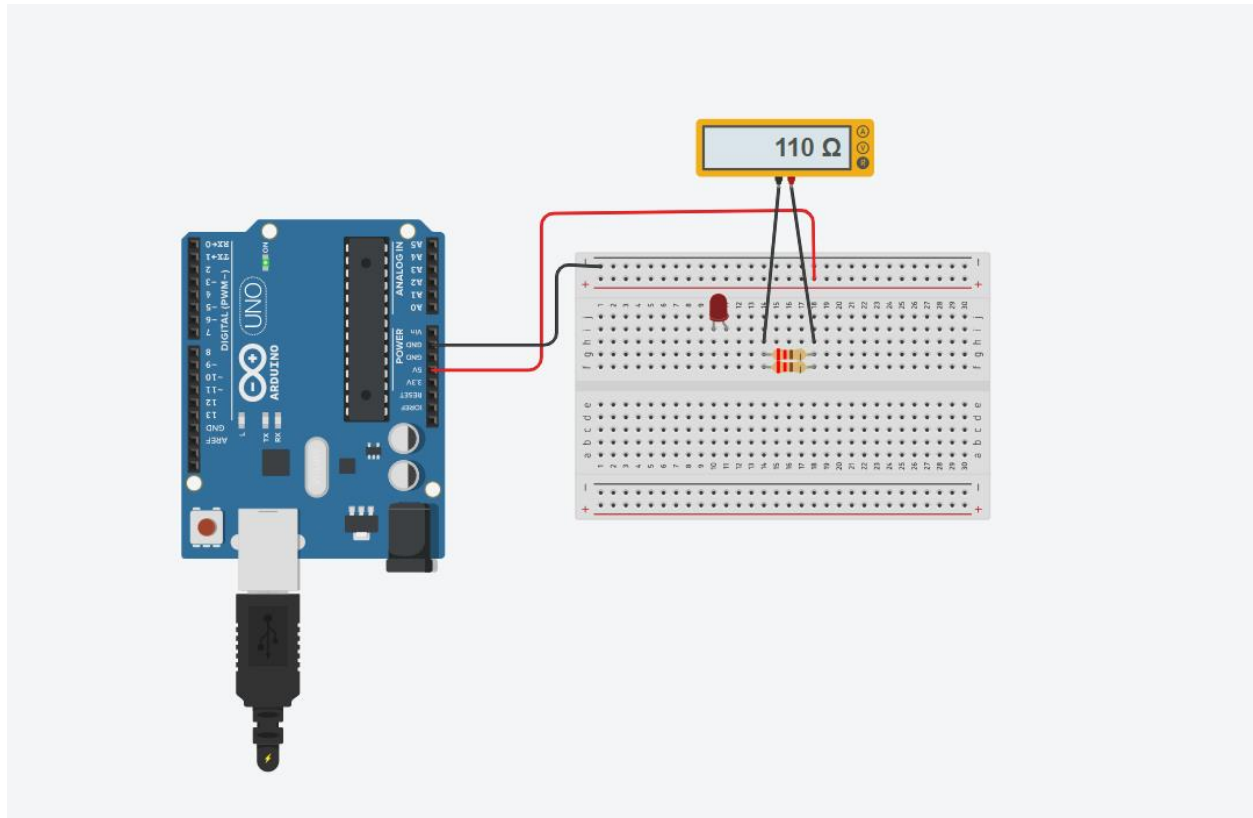


Figure 9: Led circuit

Activity 1.7: Current Measurement

Calculation of current flowing into LED

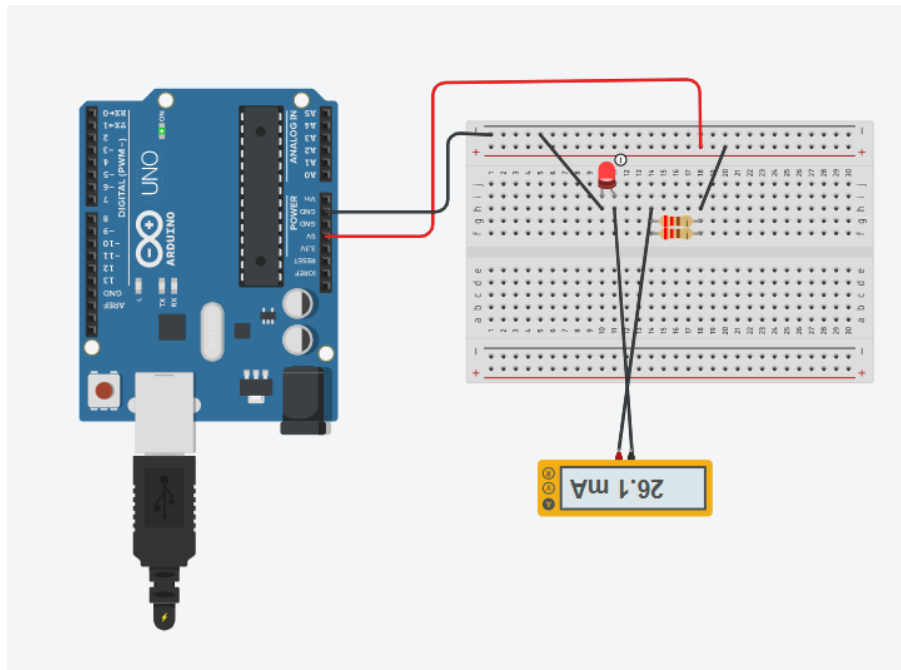


Figure 10: Current measurement

$$V=5V$$

$$R=110\Omega$$

$$I=?$$

Now,

$$I=VR$$

$$=5/110$$

$$=0.045\text{ A}$$

Actual measured value of current

$$0.0451\text{ A}$$

Why might they be different?

45.45 mA is current flowing through the circuit

26.1 mA is the current flowing through LED

The recorded result may differ from the real measurement because of the added resistance from several sources, such the breadboard, cables, and even the multimeter itself, which might affect the current flow.

Activity 1.8: Fritzing for 4 switches & LEDs

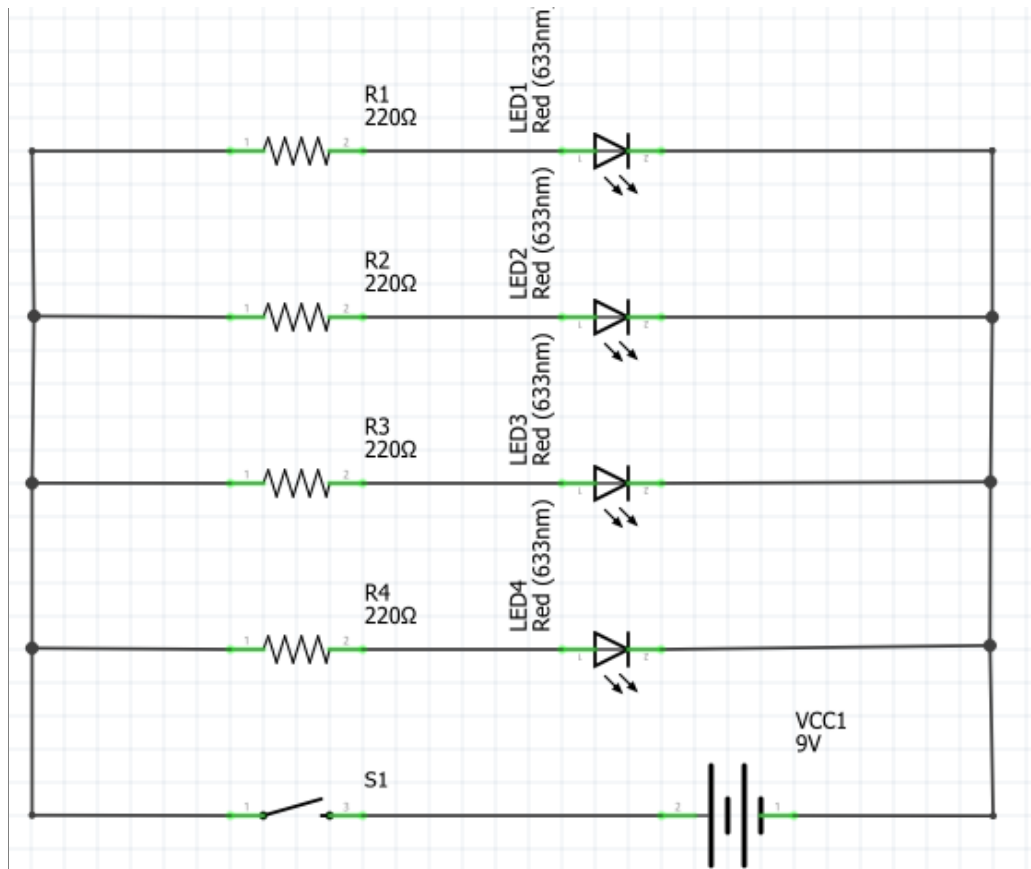


Figure 11: Fritzing diagram for 4 switches and leds

Activity 1.9: Fritzing for Number 0-7

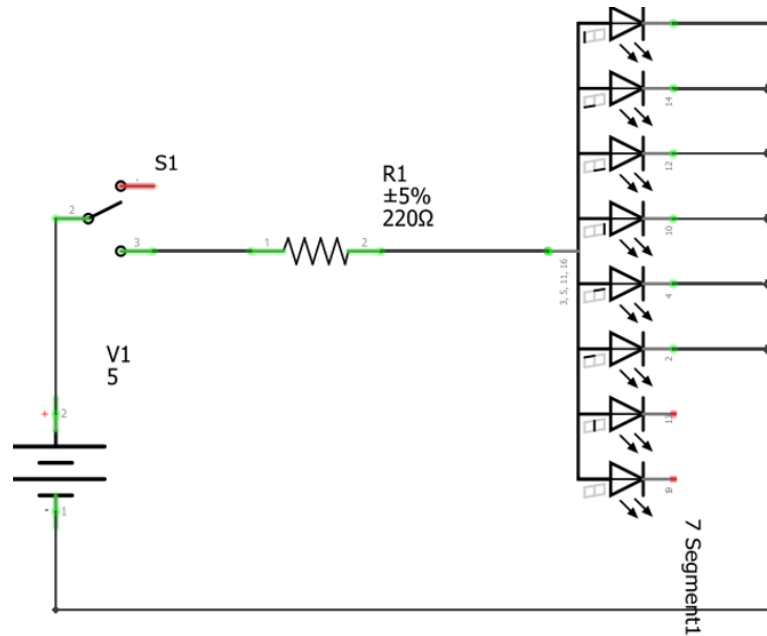


Figure 12: Fritzing for number 0-7

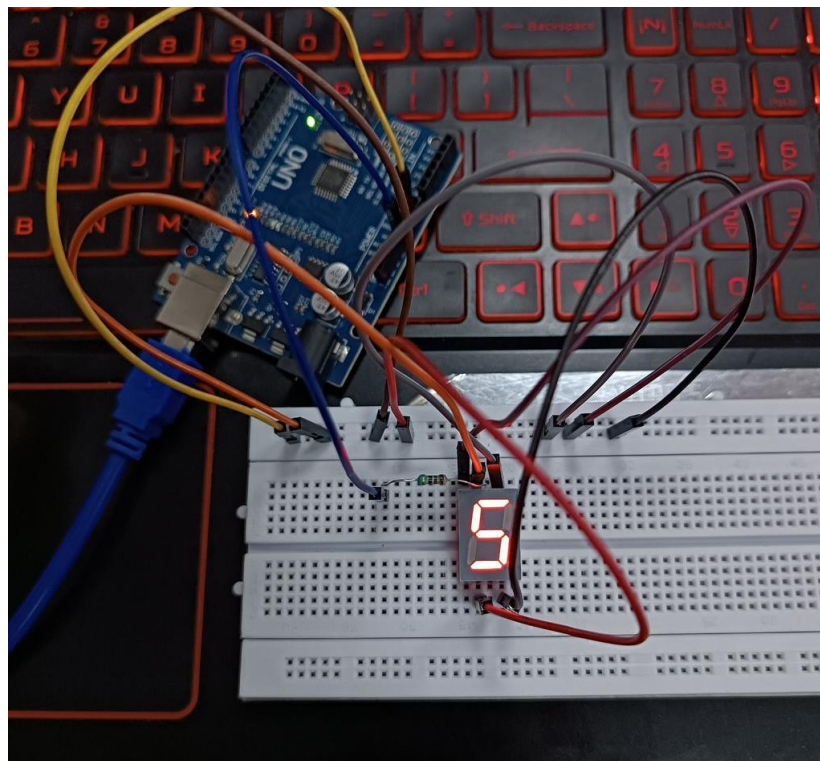


Figure 13: Fritzing for number 5

Workbook 2

Activity 2.1: LED Flashing to show decimal number 63 as binary.
63 as binary, including working

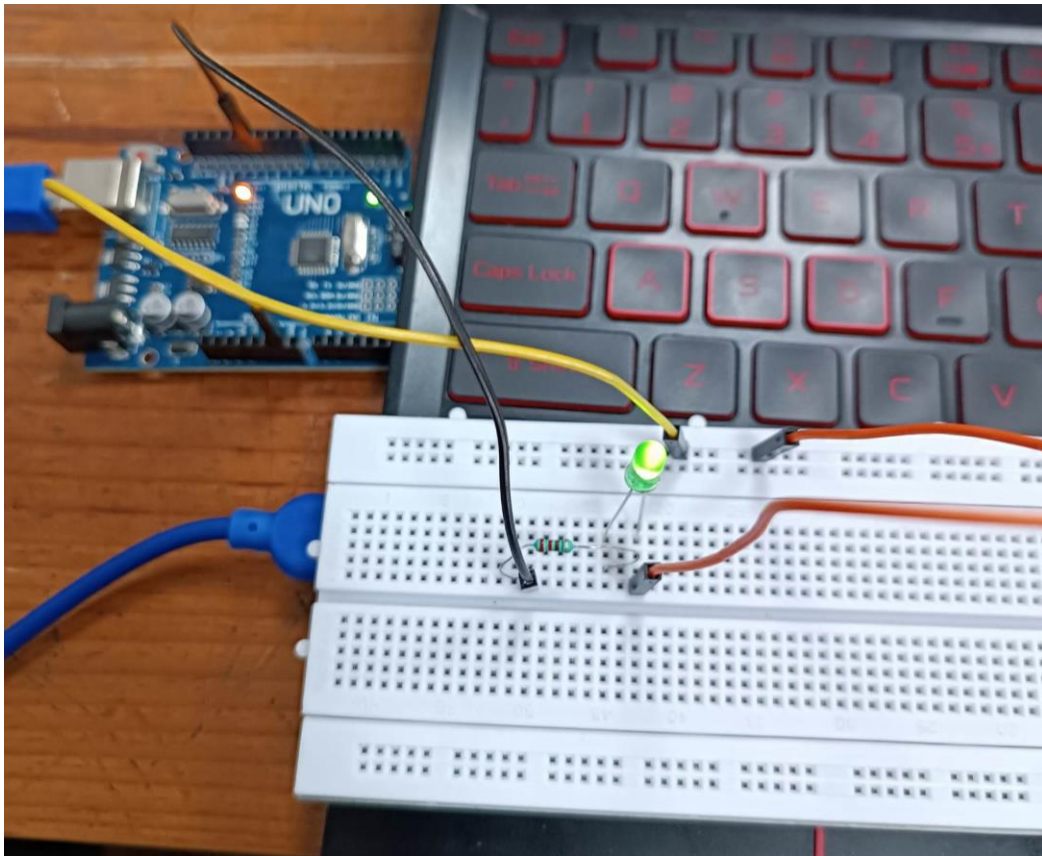


Figure 14: Led flashing to show decimal number 63 as binary

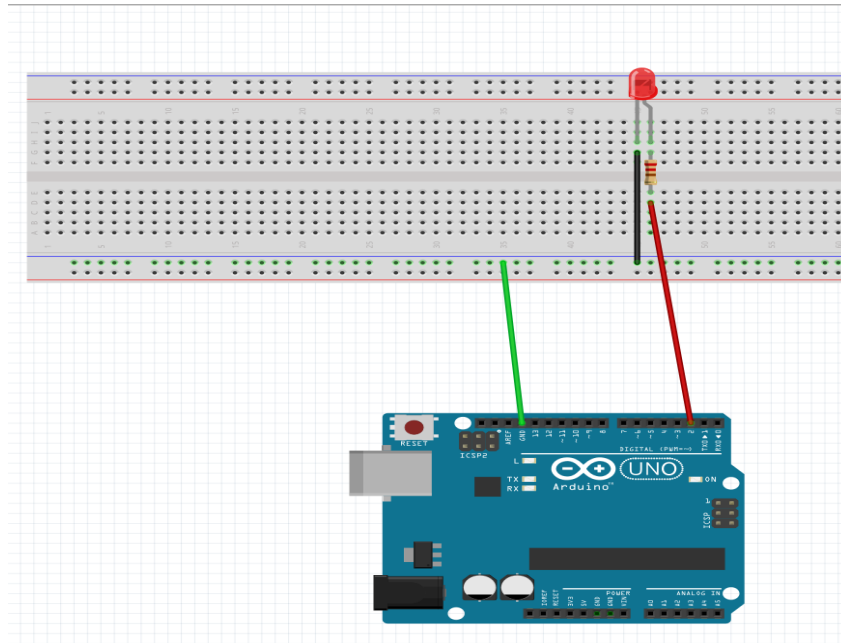


Figure 15: Fritzing diagram to display 63 as binary

$(111111)^2$ is the binary representation of the number 63. The first pair occupies rank 63 in this binary value. The cathode is linked to 0V GND to turn on the LED, while the anode is connected to 13V through a resistor to turn it on in state 1. Up until the LED is switched on, the frame will stay on. The LED is repeatedly lit in the example below, with a 5-second internal delay between each occurrence.

Copy & Post your code with a suitable comment at the top of code with your name & student number ☺

```
// This code display 63 decimal Numbers in binary

// Student Id: 2331425
//Student Name: Eliza Gamal

// The setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(11, OUTPUT);
}

// the loop function turns the LED on and off in a pattern that represents the
binary digits of number 63

void loop() {
  for(int a=0;a<7;a++){
    digitalWrite(11, HIGH); // turn LED on (HIGH is the voltage level)
    delay(1000);             // wait for one second
    digitalWrite(11, LOW);  // turn LED off (making the voltage LOW)
    delay(1000);           } // wait for one second
  }
```

Activity 2.2: 4 LED's for counting up in binary from 0 to 15.

Fritzing Circuit diagram for Step 4 i.e. 4 LEDs

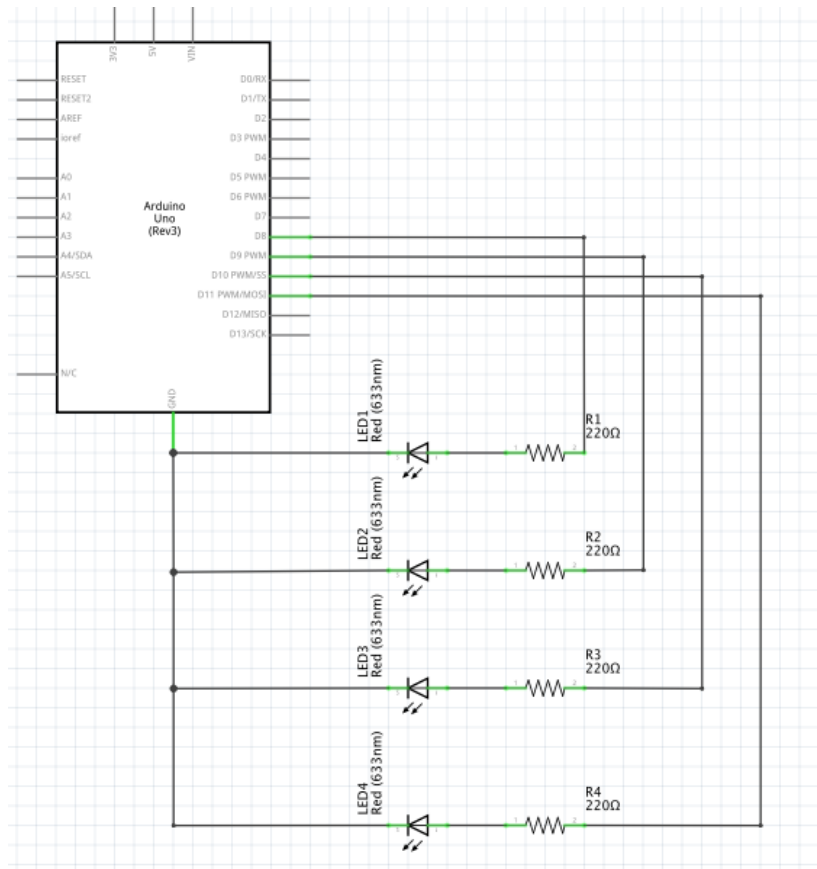


Figure 16: Fritzing Circuit diagram for counting up in binary from 0 to 15.

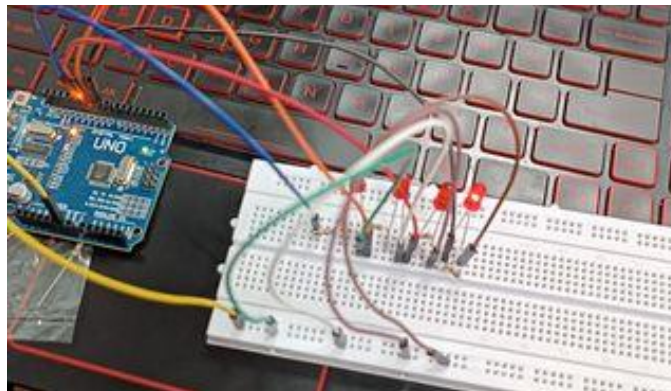


Figure 17: Led display for counting up in binary from 0 to 15.

Arduino Program for Step 4 i.e. 4 LEDs

```
//name:Eliza Gamal
//Student id: 2331425

int ledone = 12; //initializing led 1 in 12
int ledtwo = 11; //initializing led 1 in 11
int ledthree=10; //initializing led 1 in 10
int ledfour=9;  //initializing led 1 in 9

void setup() {
    // put your setup code here, to run once:
    pinMode(ledone, OUTPUT); // setup pin 1 as led one
    pinMode(ledtwo, OUTPUT); // setup pin 2 as led two
    pinMode(ledthree, OUTPUT); // setup pin 1 as led three
    pinMode(ledfour, OUTPUT); // setup pin 1 as led four
}

void loop() {
    // put your main code here, to run repeatedly:
    binary(0,0,0,0); //all the led will be off
    binary(0,0,0,1); // all the led except last one will be off
    binary(0,0,1,0); //all the led except third one will be off
    binary(0,1,0,0); //all the led except second one will be off
    binary(0,1,0,1); // the led will be off and on and repeat the same
    binary(0,1,1,0); //the led will be on except first and last
    binary(0,1,1,1); //all the led will be on except first
    binary(1,0,0,0); //all the led will be off except first one
    binary(1,0,0,1); //all led will be off except first and last
    binary(1,0,1,0); //the led will be on and off and repeat the same
```

```
binary(1,0,1,1); //all led will be on except second one  
binary(1,1,0,0); //the first two led will be on  
binary(1,1,0,1); //all led except third one will be on  
binary(1,1,1,0); //all led except last one will be on  
binary(1,1,1,1); //all led will be on  
}
```

```
void binary(int a, int b, int c , int d)  
{  
    digitalWrite(ledone,a);  
    digitalWrite(ledtwo,b);  
    digitalWrite(ledthree,c);  
    digitalWrite(ledfour,d);  
    delay(1000); //wait for 1000 milliseconds  
}
```

Fritzing Circuit diagram for Step 4 i.e. 4 LEDs

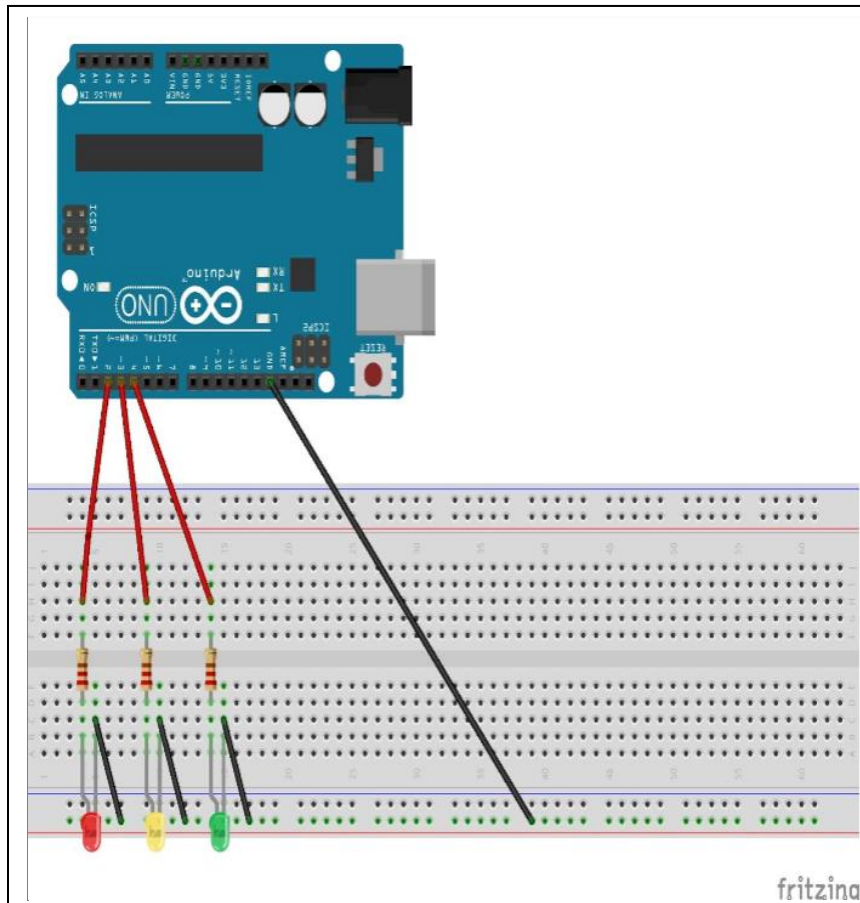


Figure 18: Fritzing diagram for traffic lights

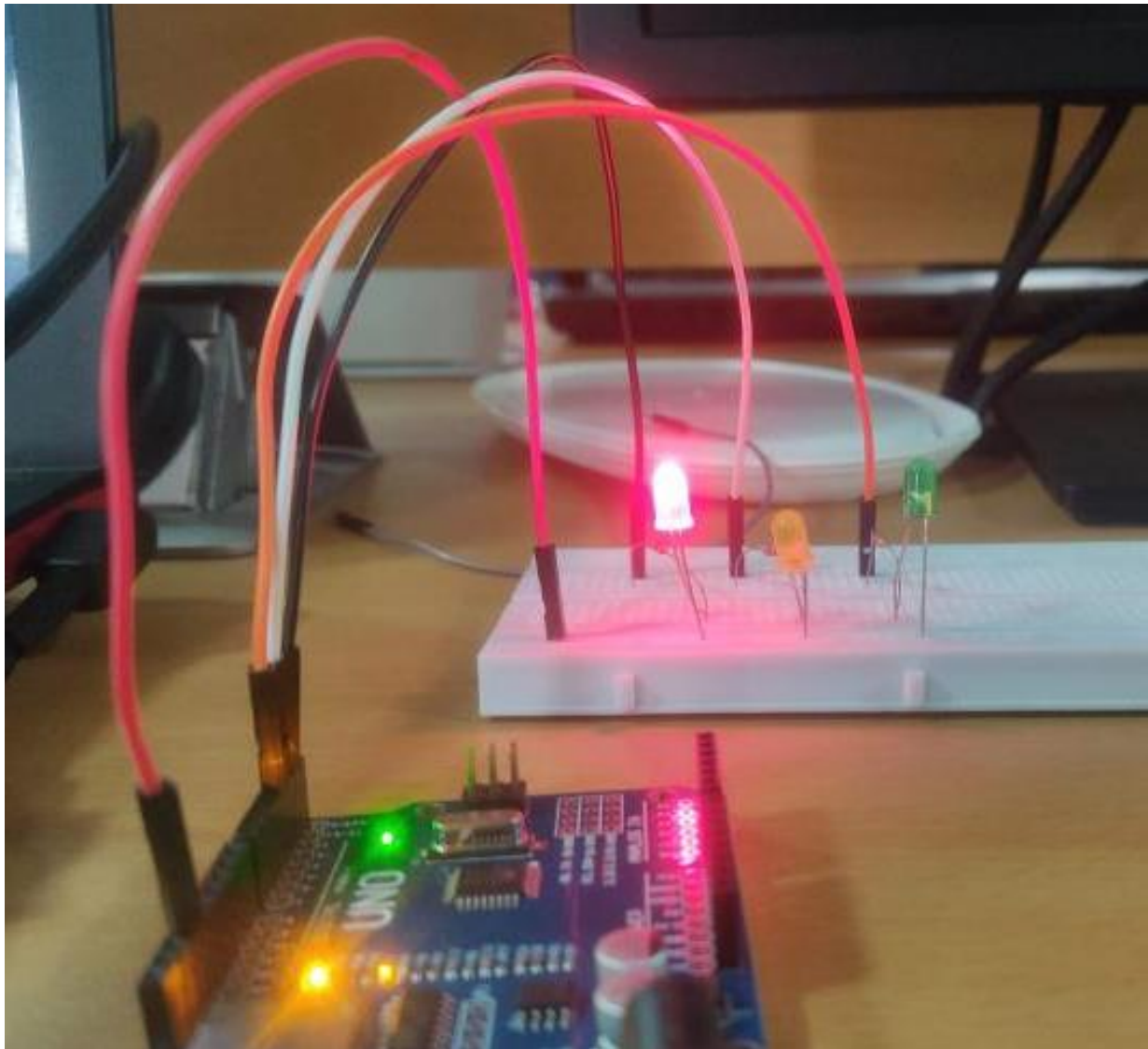


Figure 19: Led display for traffic lights

Arduino Program for Step 4 i.e. 4 LEDs

```
// Student Name: Eliza Gamal
// Student Id: 2331425

// It shows the traffic light
void setup() {
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
}
void loop() {
  int i = 0;
  do {
    digitalWrite(2, HIGH); // Turn Red on while,
    digitalWrite(3, LOW); // Yellow and,
    digitalWrite(4, LOW); // Green are off
    delay(8000); // Wait for 8 seconds
    digitalWrite(2, HIGH); // Red
    digitalWrite(3, HIGH); // Yellow
    digitalWrite(4, LOW); // Green
    delay(1000); // Wait for 1 second
    digitalWrite(2, LOW); // Red
    digitalWrite(3, LOW); // Yellow
    digitalWrite(4, HIGH); // Green
    delay(3000); // Wait for 3 seconds
    digitalWrite(2, LOW); // Red
    digitalWrite(3, HIGH); // Yellow
    digitalWrite(4, LOW); // Green
    delay(1000); // Wait for 1 second
    i++;
  }
  while (i < 5);
}
```


Workbook 3

Activity 3.1: Circuit Diagram of Button & LED

Fritzing

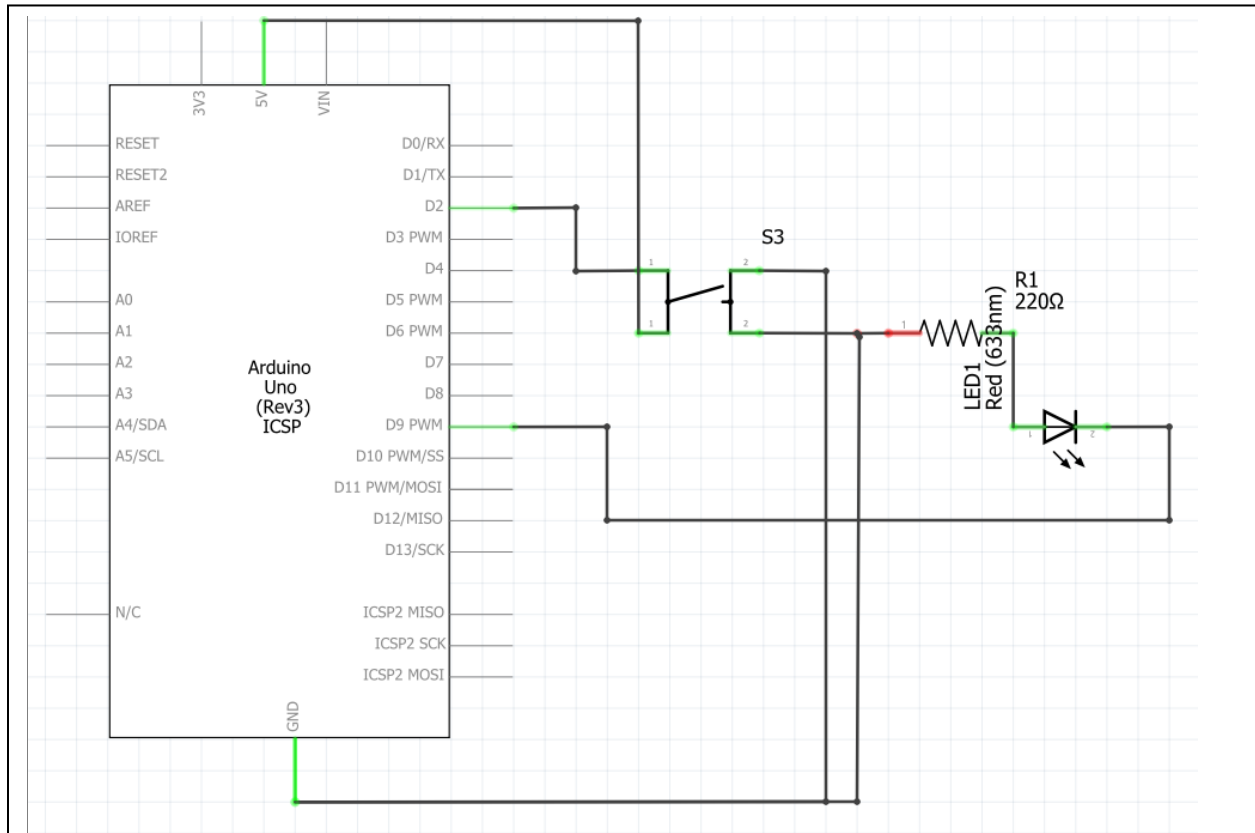


Figure 20: fritzing diagram of button and led

Arduino Program

```
// Name: Eliza Gamal
// Student Id: 2331425

int buttonState = 0;
void setup()
{
  pinMode(5, OUTPUT);
  pinMode(6, INPUT);
}
void loop()
{
  buttonState=0;
  buttonState=digitalRead(6);
  if (buttonState==1){
    digitalWrite(5, HIGH);
    delay(3000); // Wait for 5000 millisecond(s)
  } else {
    digitalWrite(5, LOW);
  }
}
```

Activity 3.2: 3 Switches & Led

Fritzing

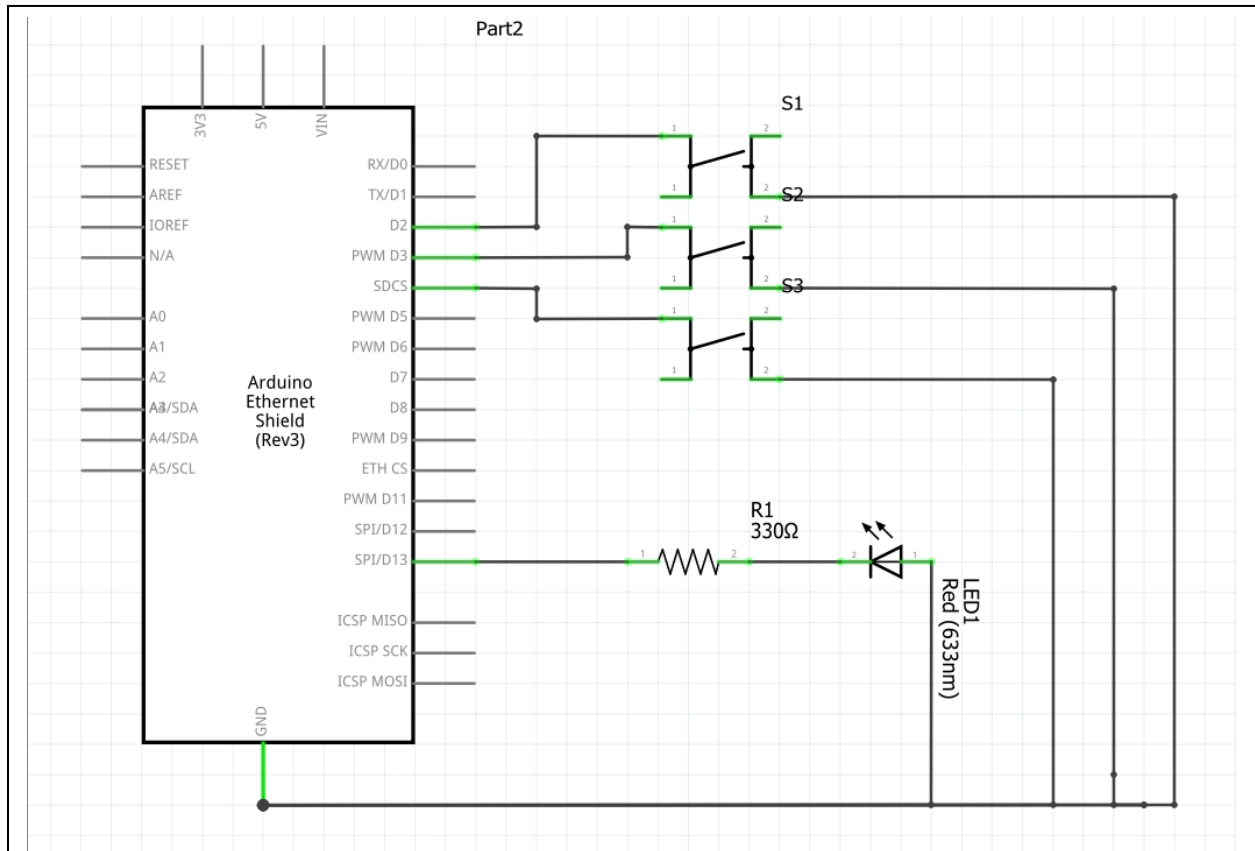


Figure 21: Fritzing diagram of switches and led

Circuit Diagram

Arduino Program

```
//Student Name: Eliza Gamal
// Student Id: 2329563

int buttonState = 0;
void setup()
{
  pinMode(3, OUTPUT);
  pinMode(2, INPUT);
  pinMode(5, OUTPUT);
  pinMode(4, INPUT);
}
void loop()
{
  buttonState=0;
  buttonState=digitalRead(2);
  if (buttonState==1){
    digitalWrite(3, HIGH);
    delay(3000); // Wait for 5000 millisecond(s)
  } else {
    digitalWrite(3, LOW);
  }

  buttonState=0;
  buttonState=digitalRead(4);
  if (buttonState==1){
    digitalWrite(5, HIGH);
    delay(4000); // Wait for 5000 millisecond(s)
  } else {
    digitalWrite(5, LOW);
  }
}
```

Activity 3.3: 8 Buttons & LEDs (SWITCH STATEMENTS)

Fritzing

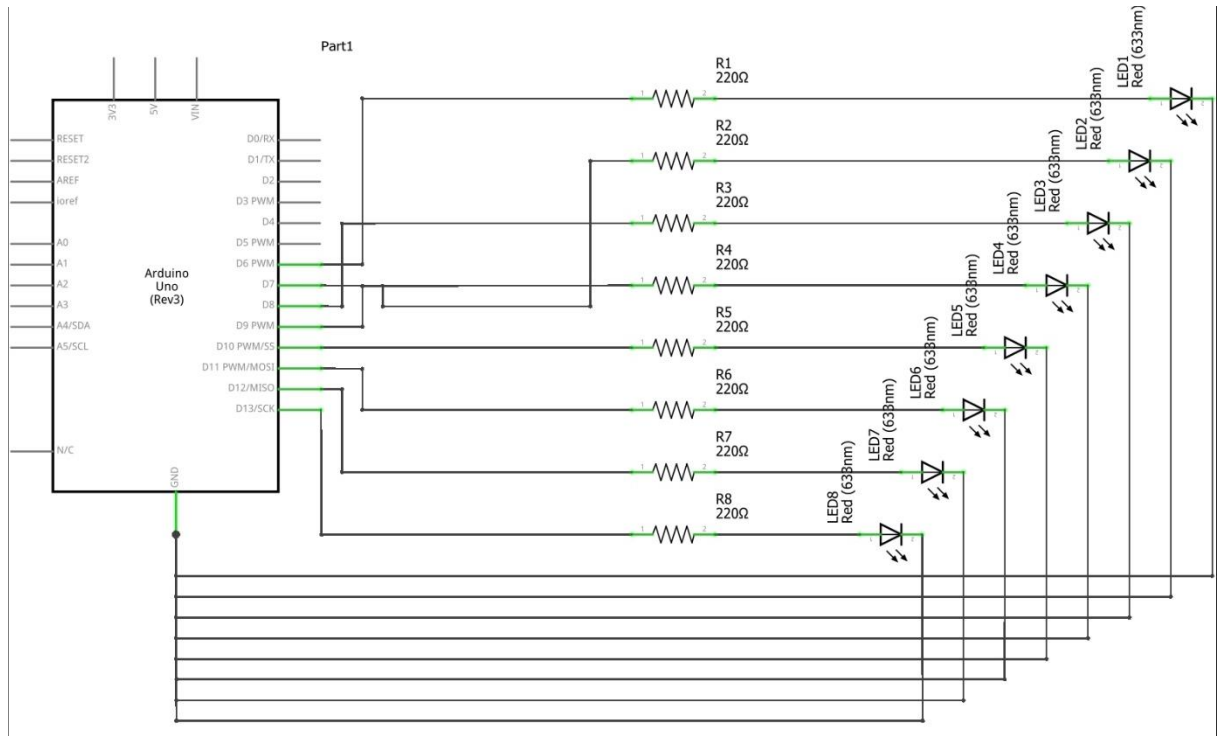


Figure 22: Fritzing diagram of buttons and led(switch statement)

Arduino Program

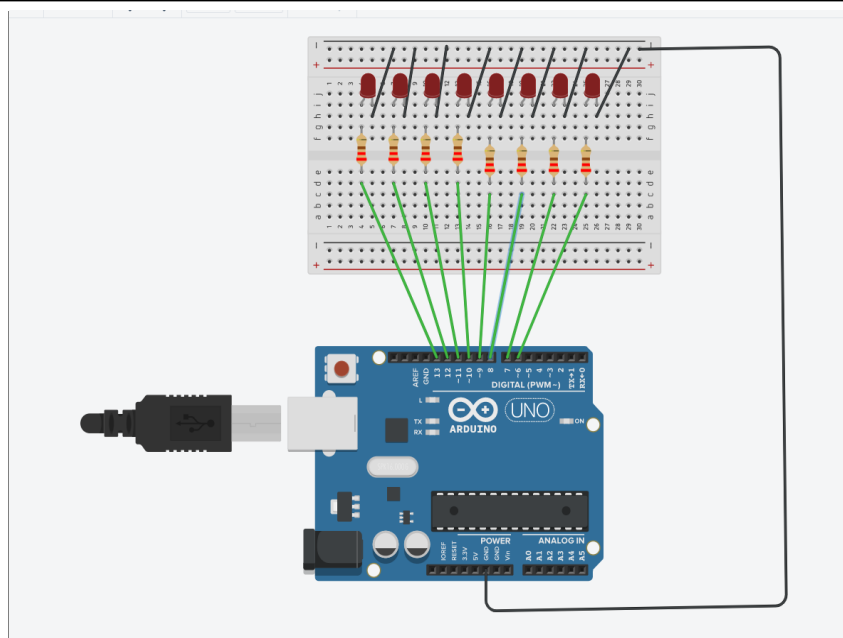


Figure 23: Tinkercad diagram of buttons and led

```
//Student-name: Eliza Gamal
```

```
//Student-id: 2331425
```

```
int led[8] = {6,7,8,9,10,11,12,13};
```

```
int value = 254;
```

```
void setup() {
```

```
    int i;
```

```
    for (i = 0; i < 8; i++)
```

```
    {
```

```
        pinMode(led[i], OUTPUT); //pin iterated as led0 to led7
```

```
    }
```

```
}
```

```
void loop() //logic for converting binary to decimal
```

```
{  
  displayBinaryforDecimal(value);  
}
```

```
void displayBinaryforDecimal(int val)  
{  
  int i;  
  for (i=0;i<8;i++)  
  {  
    if (val%2==0)  
    {  
      digitalWrite(led[i],LOW); // LED off for a 0  
    }  
    else  
    {  
      digitalWrite(led[i],HIGH); // LED on for a 1  
    }  
    val=val/2;  
  }  
}
```

Workbook 4

Activity 4.1: Serial Port

Fritzing

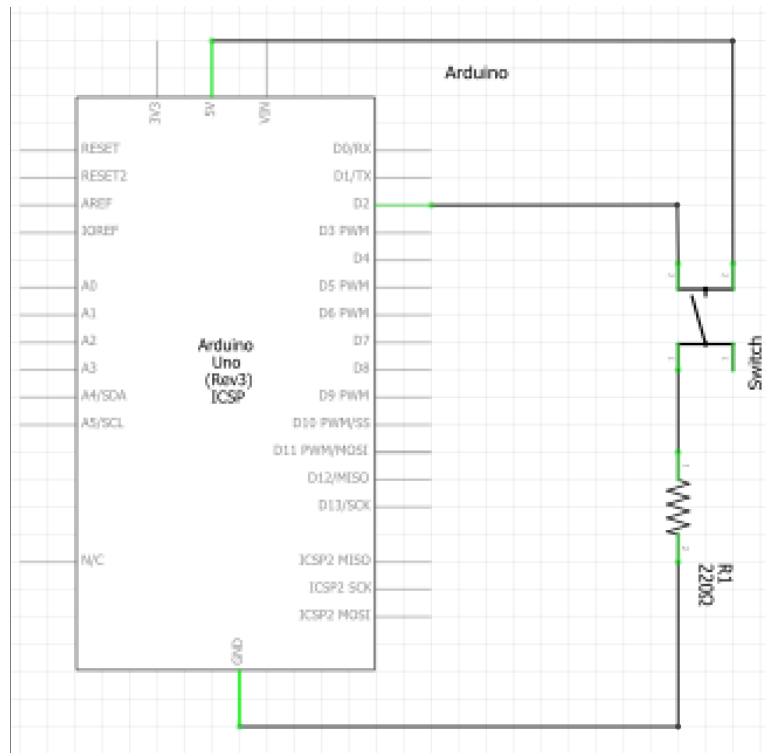


Figure 24: Fritzing diagram of serial port

Arduino Program

```
//Student Name: Eliza Gamal
//Student ID: 2331425

//activity 4.1 Serial Port

int input=2;
void setup() {

    // put your setup code here, to run once:
    Serial.begin(9600);
    pinMode(input,INPUT_PULLUP);
}

void loop() {
    // put your main code here, to run repeatedly:

    if (digitalRead(input)==1){
        Serial.println("Eliza Gamal 2331425");
        delay(1000);
    }
}
```

Screen Shot of Serial Port

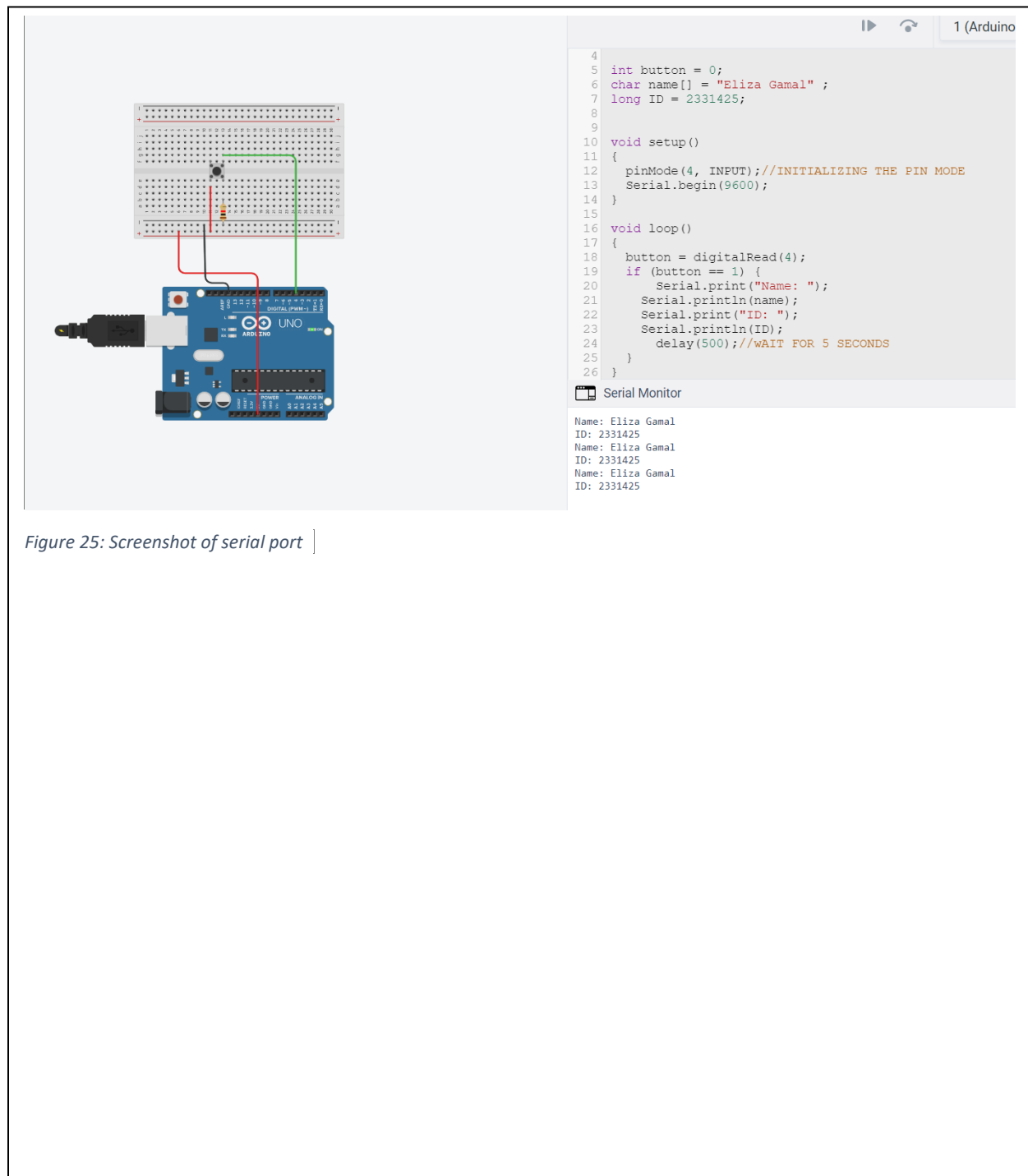


Figure 25: Screenshot of serial port

Activity 4.2: Serial Port binary to decimal

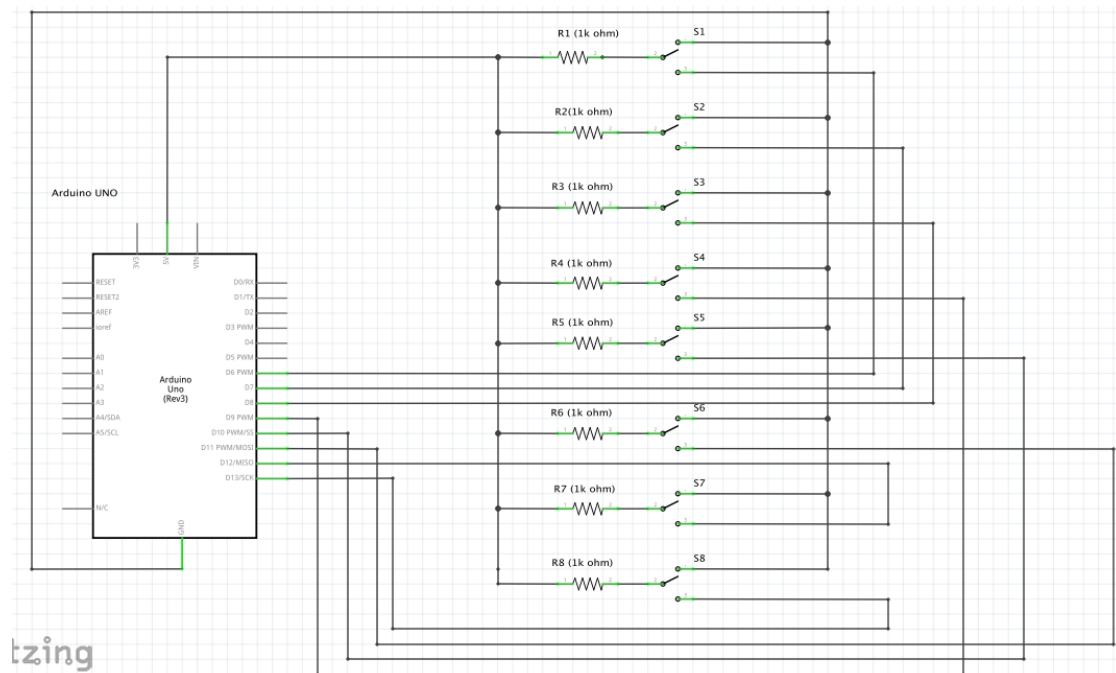


Figure 26: Fritzing diagram of serial port binary to decimal

Code

```
// Name:Eliza Gamal
//ID: 2331425
//Activity 4.2: Serial Port binary to decimal

const int switchPin[] = { 6, 7, 8, 9,10 , 11 , 12 ,13 };
int i;

void setup() {
    Serial.begin(9600);

    for (i = 0; i < 8; i++) {
        pinMode(switchPin[i], INPUT_PULLUP);
    }
}

void loop() {
    int binary[8];

    for (i = 0; i < 8; i++) {
        binary[i] = digitalRead(switchPin[i]);
    }

    int decimal = 0;

    for (int j = 0; j < 8; j++) {
        decimal += binary[j] * pow(2, 7 - j);
    }

    Serial.print("The decimal representation of ");
    Serial.print(decimal, BIN);
    Serial.print(" is: ");
    Serial.print(decimal);
    Serial.print("\n");

    delay(3000);
}
```

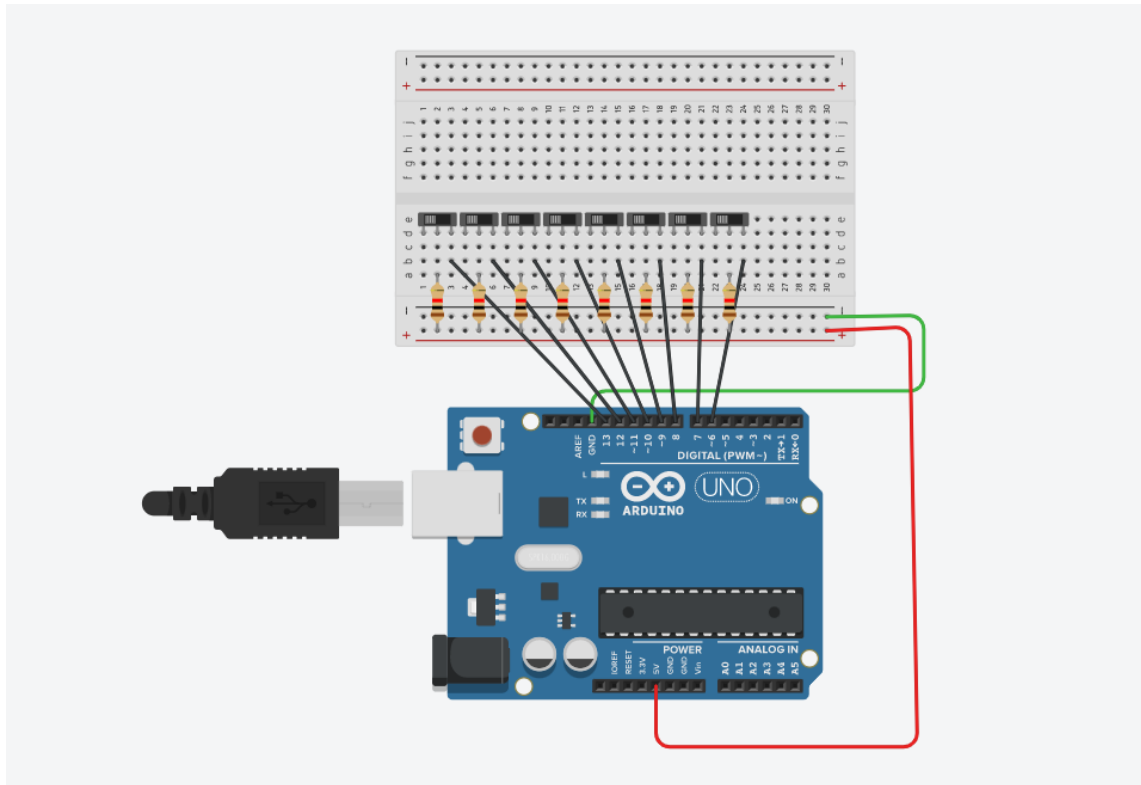


Figure 27: serial port binary to decimal in tinkercad

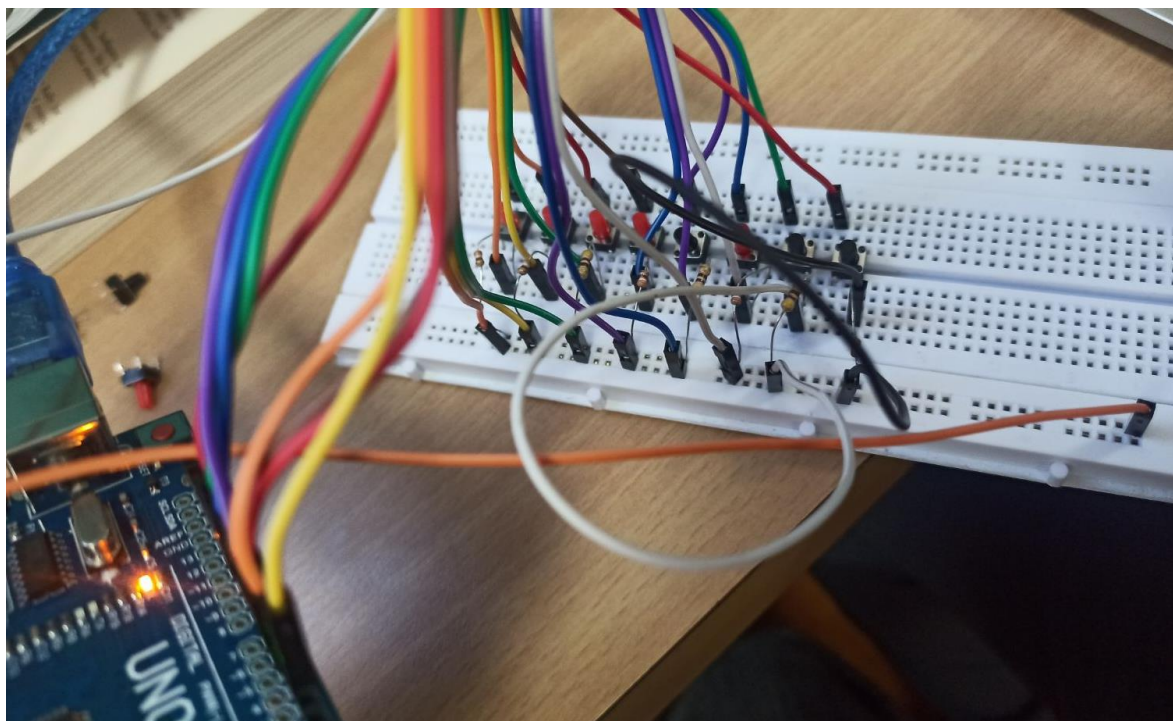
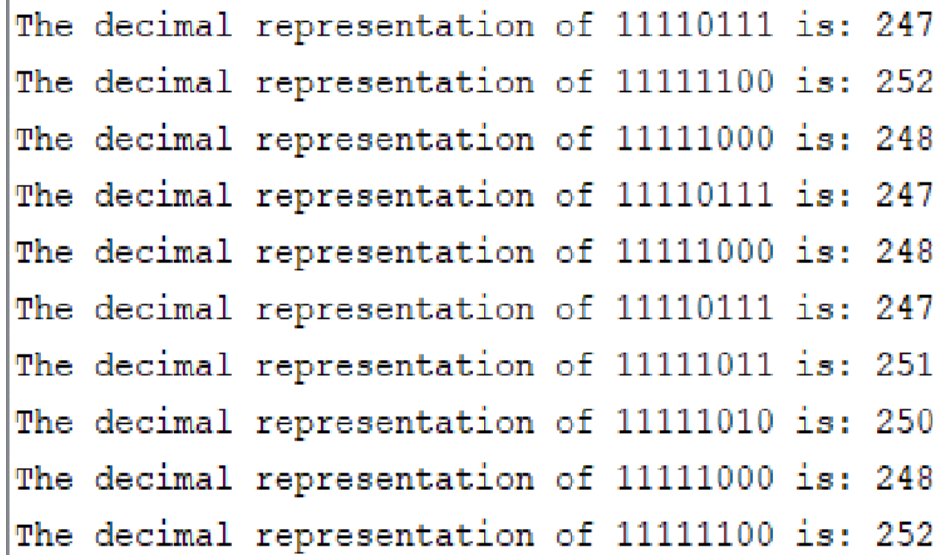


Figure 28: serial port binary to decimal

Screen Shot of Serial Port



The screenshot shows a serial port terminal window with a white background and a grey title bar. The text is displayed in a monospaced font. It contains ten lines of text, each showing a binary number followed by its decimal equivalent. The binary numbers are 11110111, 11111100, 11111000, 11110111, 11111000, 11110111, 11111011, 11111010, 11111000, and 11111100. The decimal values are 247, 252, 248, 247, 248, 247, 251, 250, 248, and 252 respectively.

```
The decimal representation of 11110111 is: 247
The decimal representation of 11111100 is: 252
The decimal representation of 11111000 is: 248
The decimal representation of 11110111 is: 247
The decimal representation of 11111000 is: 248
The decimal representation of 11110111 is: 247
The decimal representation of 11111011 is: 251
The decimal representation of 11111010 is: 250
The decimal representation of 11111000 is: 248
The decimal representation of 11111100 is: 252
```

Figure 29: screenshot of serial port binary to decimal

Activity 4.3: Calibrating Analogue Information

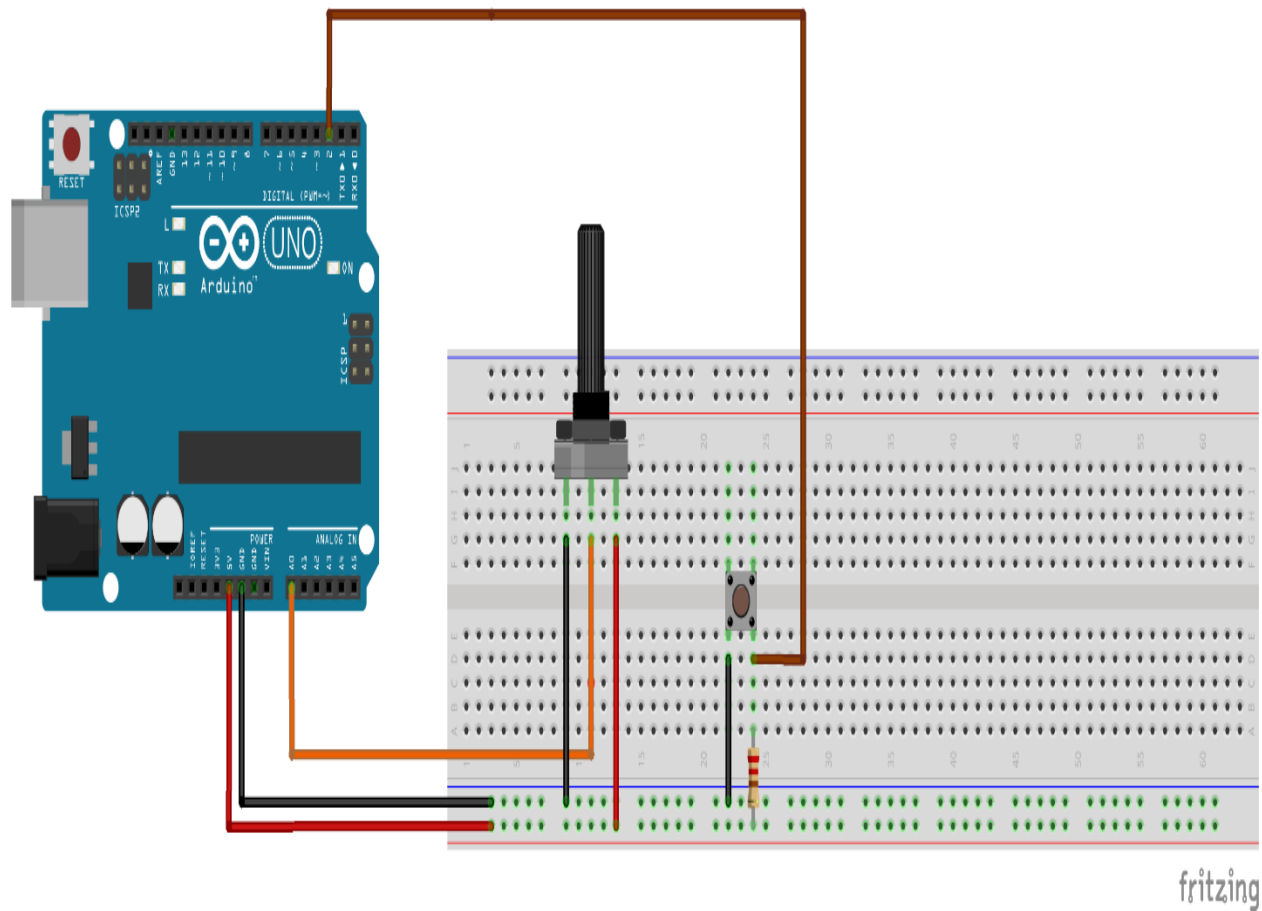


Figure 30: fritzing diagram of calibrating analogue information

Code

```
//Student Name: Eliza Gamal
// Student ID: 2331425

//Activity 4.3: Calibrating Analogue Information

const int buttonPin = 2;
void setup()
{
  pinMode(buttonPin, INPUT);
  Serial.begin(9600);
}
void loop()
{
  // Read the state of the button
  int buttonState = digitalRead(buttonPin);
  // If the button is pressed, measure the voltage and resistance
  if (buttonState == HIGH) {
    int sensorValue = analogRead(A0);
    float voltage = sensorValue * (5.0 / 1024.0);
    // Print the measured voltage
    Serial.print(&quot;The Voltage is: &quot;);
    Serial.print(voltage);
    Serial.println(&quot;V&quot;);
    delay(2000);
    // Calculate and print the resistance
    float resistance = (voltage * 220) / 5;
    Serial.print(&quot;The Resistance is: &quot;);
    Serial.print(resistance);
    Serial.println(&quot;KOhm&quot;);
    delay(2000);
  } else {
    // If the button is not pressed, print an error message
    Serial.println(&quot;Error&quot;);
    delay(2000);
  }
}
```


Pot Resistance Clockwise

61.30 KOhm

Pot Resistance Anti-clockwise

123.Kohm

Sample of Values

Pot Resistance against Voltage change

Pot Resitance	Voltage Measured
109.13 Kohm	2.48 V
61.2 Kohm	1.40 V
61.3 Kohm	1.38 V
62.98 Kohm	1.43 V
63.5 Kohm	1.38 V

Screen Shot of Meaningful Serial Port Output, not just numbers

```
The Voltage is:  0.72V
The Resistance is:  31.80KOhm
Error
Error
Error
The Voltage is:  0.72V
The Resistance is:  31.80KOhm
Error
Error
Error
Error
Error
The Voltage is:  0.73V
The Resistance is:  32.01KOhm
The Voltage is:  0.72V
The Resistance is:  31.80KOhm
Error
Error
The Voltage is:  1.83V
The Resistance is:  80.57KOhm
```

Figure 31: ss of meaningful serial port output |

Activity 4.4: Temperature Sensor & Serial Port

Code - Centigrade to Serial port, but when button Pressed Fahrenheit Displayed Instead

```
//Name: Eliza Gamal
// ID: 2331425
//Activity 4.4: Temperature Sensor & Serial Port

#include<LiquidCrystal.h>
const int rs = 7, en=6, d4=5, d5=4, d6=3,d7=2;
LiquidCrystal lcd (rs, en, d4, d5, d6, d7);
float celsius, fahrenheit;
int temperature= A0;
void setup(){
  pinMode (8, INPUT_PULLUP);
  pinMode (temperature, INPUT);
}
void loop(){
  int buttonState = digitalRead (8);
  int adcValue = analogRead (temperature);
  float voltage = adcValue * 5.0 / 1023;
  celsius = (voltage - 0.5) * 100;
  fahrenheit = celsius * (9.0/5.0) + 32.0;
  if (buttonState == 0){
    lcd.setCursor(0,1);
    lcd.print("Temp: ");
    lcd.print(celsius);
    lcd.print("C");
    delay(3000);
    lcd.clear();
  }else{
    lcd.setCursor(0,1);
    lcd.print("Temp");
    lcd.print(fahrenheit);
    lcd.print("F");
    delay(100);
    lcd.clear();
  }
}
```

Screen Shot of Serial Port

```
Temperature in Celsius is: 45.00 deg C  
Temperature in Fahrenheit is: 113.00 deg F  
Temperature in Fahrenheit is: 113.00 deg F  
Temperature in Celsius is: 45.00 deg C  
Temperature in Celsius is: 45.00 deg C  
Temperature in Celsius is: 45.00 deg C
```

Figure 32: ss of serial port of temperature



Figure 33: lcd display of temperature in celsius

Workbook 5

Activity 5.1: RGB Led and switches

Fritzing

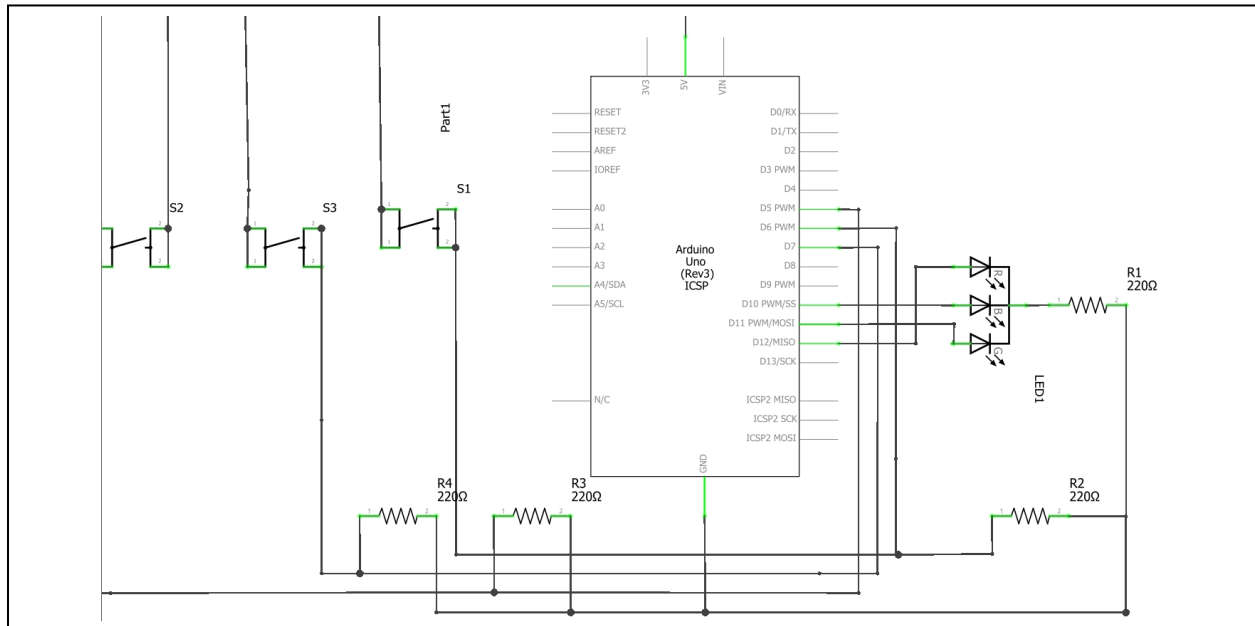


Figure 34: fritzing diagram of rgb led and switches

Arduino Program

```
//Student name:ElizA Gamal
// Student ID : 2331425
//Activity 5.1: RGB Led and switches

int redPin = 2;
int greenPin = 3;
int bluePin = 4;
int redButton = 5;
int greenButton = 6;
int blueButton = 7;
void setup() {
    pinMode(redPin, OUTPUT);
    pinMode(greenPin, OUTPUT);
    pinMode(bluePin, OUTPUT);
    pinMode(redButton, INPUT_PULLUP);
    pinMode(greenButton, INPUT_PULLUP);
    pinMode(blueButton, INPUT_PULLUP);
}
void loop() {
    for(int i=0; i<3; i++) {
        if (i== 0) { // red LED ON
            digitalWrite(redPin, HIGH);
            digitalWrite (greenPin, LOW);
            digitalWrite (bluePin, LOW);
            while(digitalRead(redButton) == HIGH) {}
        }
        else if (i == 1) { // green LED ON
            digitalWrite(redPin, LOW);
            digitalWrite (greenPin, HIGH);
            digitalWrite (bluePin, LOW);
            while(digitalRead(greenButton) == HIGH) {}
        }
        else if (i == 2) { // blue LED ON
            digitalWrite(redPin, LOW);
            digitalWrite (greenPin, LOW);
            digitalWrite (bluePin, HIGH);
            while(digitalRead(blueButton) == HIGH) {}
        }
    }
}
```

Activity 5.2: Distance Sensor

Arduino Code

```
// Student Name: Eliza Gamal
// Student id: 2331425
//Activity 5.2: Distance Sensor

const int trig_pin = 2;
const int pw_pin = 4;
const int trig_delay = 5; //Microseconds
void setup()
{
    Serial.begin(9600); //Microseconds
}
void loop(){
    long duration;
    float inch;
    float cm;
    //Tell distance to send out a pulse Value
    pinMode(trig_pin, OUTPUT);
    digitalWrite(trig_pin, LOW);
    delayMicroseconds(10);
    digitalWrite(trig_pin, HIGH);
    delayMicroseconds(trig_delay);
    digitalWrite(trig_pin, LOW);

    //Measure time of pulse on PW pin
    pinMode(pw_pin, INPUT);
    duration = pulseIn(pw_pin, HIGH);

    //Convert time to distance
    cm = duration / 58.8;
    inch = cm/2.54;
    Serial.print(cm);
    Serial.print(" cm ");
    Serial.print(inch);
    Serial.print(" inch ");
    Serial.print(duration);
    Serial.print(" Raw ");
    Serial.print("\n");
    delay(1000);
}
```

Take a picture of your distance sensor and include it here, please reduce the size and quality as it will be too large else 😊

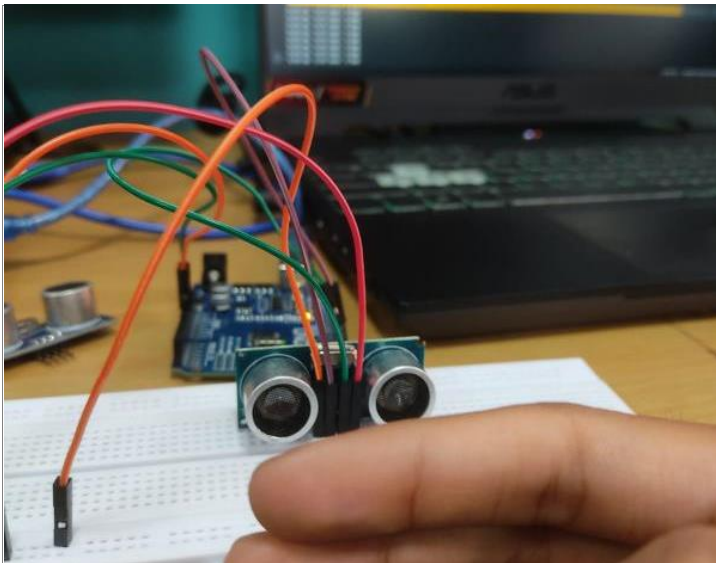


Figure 35: picture of distance sensor

```
The distance in inches is:4  
The distance in centimeter is:10  
The distance in inches is:3  
The distance in centimeter is:8  
The distance in inches is:4  
The distance in centimeter is:10  
The distance in inches is:6  
The distance in centimeter is:16
```

Figure 36: ss of serial port of the distance

Activity 5.3: 1602 LCD Display

Fritzing

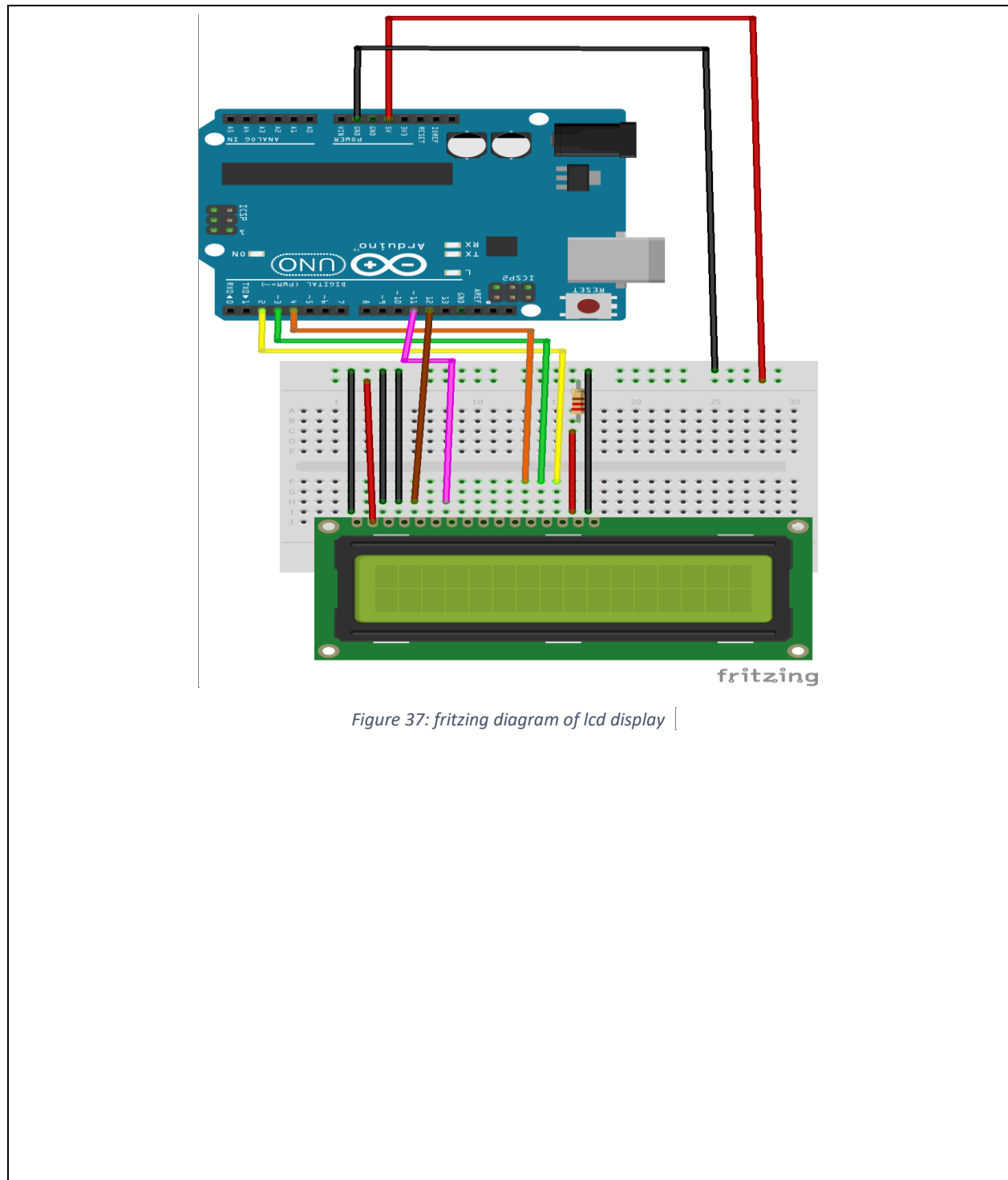


Figure 37: fritzing diagram of lcd display

Arduino Program

```
//Student-name: Eliza Gamal
```

```
//Student-id: 2331425
```

```
// include the library code:
```

```
#include <LiquidCrystal.h>
```

```
// initialize the library by associating any needed LCD interface pin
```

```
// with the arduino pin number it is connected to
```

```
const int rs = 12, en = 11, d4 = 4, d5 = 5, d6 = 6, d7 = 7;
```

```
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
```

```
void setup() {
```

```
    // set up the LCD's number of columns and rows:
```

```
    lcd.begin(16, 2);
```

```
    // Print a message to the LCD.
```

```
    lcd.print("4CS016");
```

```
    lcd.setCursor(0,1);
```

```
    lcd.print("Embedded Systems");
```

```
}
```

```
void loop() {
```

```
}
```

Take a picture of your LCD and include it here, please reduce the size and quality as it will be too large else 😊

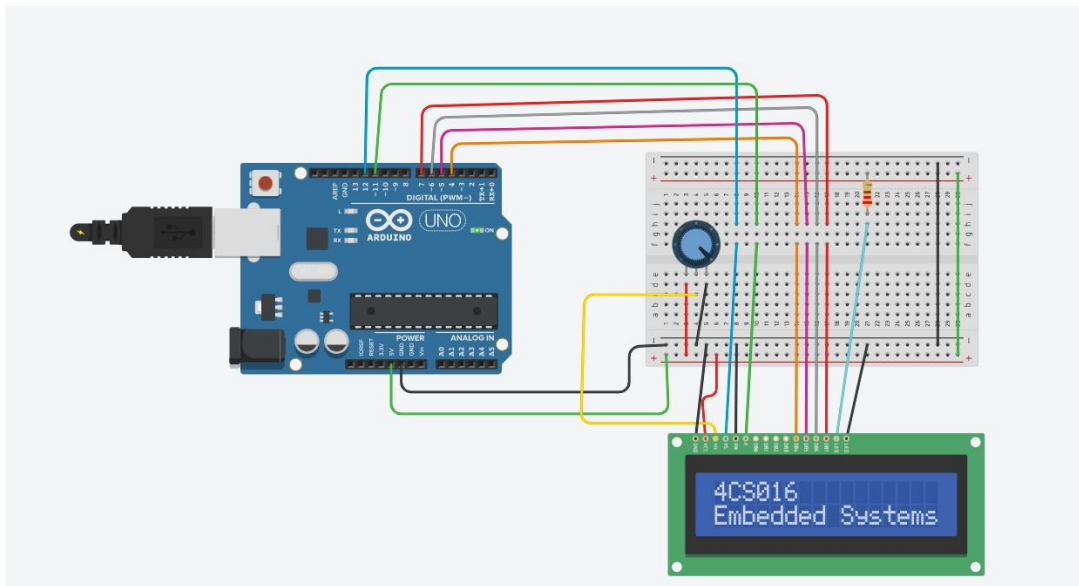


Figure 38: lcd display in tinkercad

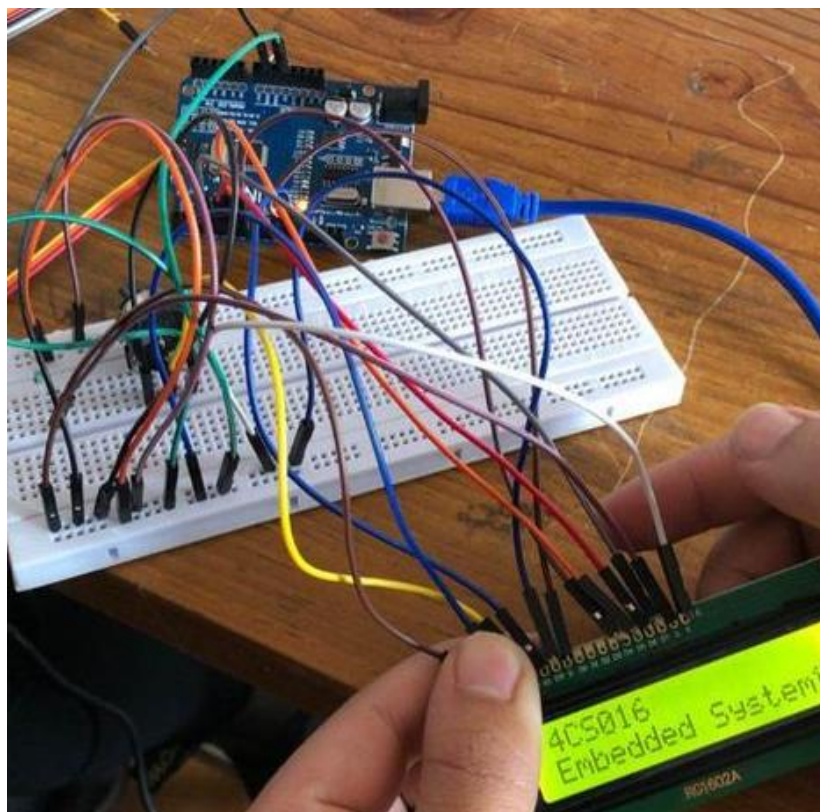


Figure 39: lcd display of module name

Workbook 6

Activity 6.1: PWM

Fritzing

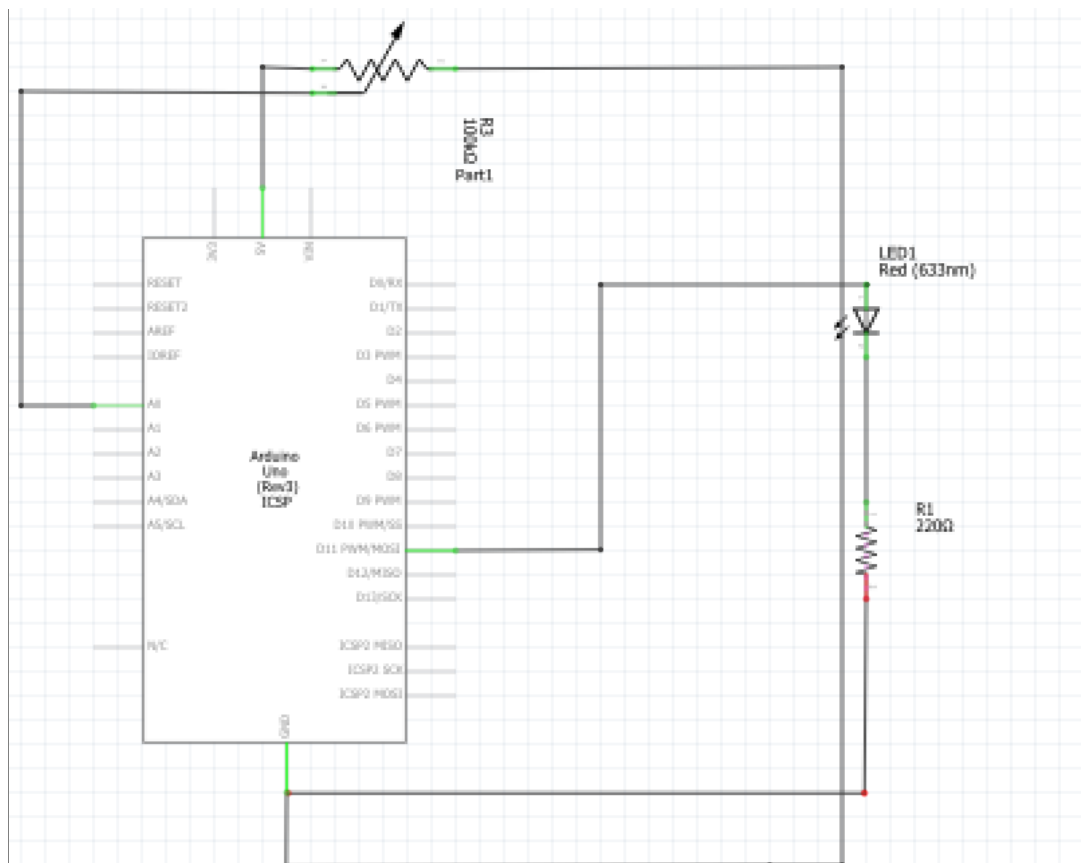


Figure 40: fritzing diagram of pwm

Arduino Program

```

// Student name: Eliza Gamal
// Student id: 2331425

// PWM
#include <LiquidCrystal.h>

// Initialize the LCD library
LiquidCrystal lcd(11, 10, 5, 4, 3, 2);

// Analog input pin for the potentiometer
int potPin = A0;
// Digital output pin for the LED
int ledPin = 9;
// Variable to store the current brightness value
int brightness = 0;

void setup() {
// Set the LED pin as an output
  pinMode(ledPin, OUTPUT);
  // Initialize the LCD screen with 16 columns and 2 rows
  lcd.begin(16, 2);
  // Display the text "Brightness:" on the first row of the LCD
  lcd.print("Brightness:");
}

void loop() {
  // Read the potentiometer value and divide by 4 to get a value between 0 and 255
  brightness = analogRead(potPin);
  // Set the LED brightness to the potentiometer value
  analogWrite(ledPin, brightness);
  // Set the cursor to the second row of the LCD
  lcd.setCursor(0, 1);
  // Clear the previous brightness value on the LCD
  lcd.print("    ");
  // Set the cursor to the second row of the LCD again
  lcd.setCursor(0, 1);
  // Display the current brightness value on the LCD
  lcd.print(brightness);
  // Delay to avoid flickering of the LCD
  delay(10);
}

```

Workbook 7

Activity 7.1: 2 Arduinos – using Digital Pins

Fritzing

Arduino Program

Activity 7.2: 2 Arduinos – using Serial I/O

Fritzing

1

Arduino Program

```


```


Workbook 8

Activity 8.1: Stepper Motor Circuit Diagram

I

Circuit Diagram

Arduino Program

Activity 8.2: 2 Stepper Motors

Arduino Program

```
1
```

Workbook 9

Activity 9.1: Windscreen Wiper Code using Servos & Temperature Sensor

Arduino Code

Individual Project (50%)

Rationale

Throughout the module you have used a range of sensors and actuators with an Arduino to complete weekly tasks. For the mini project we would like you to research and create a small embedded project in an area of your choice, such as:

- Games
- Networking
- IT Security
- Systems Engineering
- Smart Technology
- Artificial Intelligence

Previous projects have included a reaction game that gives a score depending on how fast you hit a button, this has buttons to restart the application, and an LCD to show scores, and information.

This project should be your own work, **YOU MUST NOT COPY A PROJECT FROM THE INTERNET.**

Timescales

This project should be started around week 5 and continue until the deadline, when it will be submitted in the Portfolio.

Equipment

You are free to use Tinkercad, or your own kit.

The Project

Step 1 produce a detailed description of your project.

This should clearly describe what you are intending to build and may contain some diagrams of how the sensor/switches input is to be processed by the Arduino. Then what kind of output is intended to be seen or heard by the user. Please mention any tools you intend to use.

Step 2 Circuit Diagram & Fritzing Schematic

You are required to produce a circuit diagram of your work showing any calculations you made, so these might be suitable resistor values for any LED's you use. These calculations are covered on the module. The circuit diagram should not be hand drawn but should follow the format of circuits from the module.

Step 3 A Program

You will need to write some software for this project and a listing of the code with suitable comments will need to be included.

Step 4 Testing

You will be required to produce some suitable test data that you would expect to be able to measure such as voltages, test code.

Once your prototype is complete you will be expected to test your circuit and compare the actual values to your initial test data, and comment on the results.

Step 5 Conclusions

You are required to write a summary of the work along with a short half page reflection on how you found the work.

Layout

The report should be suitably laid out for a report, using headings, references if required in Harvard style, and appendices used for any lengthy code. All diagrams should be produced on a PC, and hand-written work is not acceptable.

Marking

All sections carry equal marks.