Practical Robotics Projects with Arduino
(CSE 4571)

Lab Assignment No – 03

Light & Sound Show

Submission Date: ______________

Branch: CSE	**Section: 2241026**	
Name	**Registration No.**	**Signature**
Dinanath Dash	2241004161	

Department of Computer Science and Engineering
Institute of Technical Education and Research (Faculty of Engineering)
Siksha 'O' Anusandhan (Deemed to be University)
Bhubaneswar, Odisha-751030.

Aim:

Light & Sound Show: To design and implement an intelligent lighting system using Arduino Uno that integrates analog voltage control, potentiometers, photoresistors, RGB LED color mixing, and audio synchronization, thereby enabling adaptive brightness, color variation, and sound response based on user input and ambient light conditions.

Objectives:

1) Develop a comprehensive understanding of analog voltage and potentiometers.

1.1) Read analog voltage using the "analogRead" command and convert the analog voltage to a voltage reading in volts

✓ Generate an analog voltage by building a voltage divider circuit with two resistors (330 ohm and 100 ohm) and connect the circuit to the 5V supply pin of the Arduino.

✓ Acquire knowledge of analog voltage reading by implementing the "analogRead" command on the Arduino platform.

✓ Utilize the "analogRead" command to measure and display the analog voltage readings on the Arduino Serial Monitor.

1.2) Read and interpret the voltage signals produced by a potentiometer.

✓ Demonstrate knowledge of the function and operation of potentiometers in electronic circuits.

✓ Write an Arduino sketch to read the analog value from the potentiometer connected to pin A0 and displays the value on the Serial Monitor.

1.3) Read the analog value from the potentiometer and converts it to a voltage reading in volts using Arduino and serial monitor

✓ Read the analog value from the potentiometer and converts it to a voltage reading in volts using the formula V = (ADC value * Vref) / 1023, where Vref is the reference voltage (5V in this case) and 1023 is the maximum value of the ADC. The voltage reading is then displayed on the Serial Monitor.

1.4) Controlling LED Brightness with Potentiometer and Serial Monitoring using Arduino

1.4.1) Controlling LED brightness with potentiometer without using map() function.

- Read the analog value from the potentiometer, calculates the proportional brightness value, sets the brightness of the LED, and displays the values on the serial monitor.

1.4.2) Controlling LED brightness with potentiometer using map() function.

- Read the analog value from the potentiometer, calculates the proportional brightness value, sets the brightness of the LED, and displays the values on the serial monitor.

1.5) Optimizing Electronic Control with Potentiometers: A Practical Approach to Controlling Speaker Volume and LED Brightness with Arduino

✓ Utilize the serial monitor in the Arduino sketches to accurately monitor the changes in analog values and ensure reliable control while adjusting the volume of a speaker and the brightness of an LED using potentiometer controls. The circuit connection for this sketch involves connecting a potentiometer for controlling speaker volume and another potentiometer for controlling LED brightness.

2) **Understand the function and operation of an LDR/Photoresistor in electronic circuits and its ability to measure changes in light intensity.**

   2.1) Comprehend the relationship between light intensity and the resistance of photoresistors and how they can be used to measure changes in light intensity.

   ✓ Develop a comprehensive understanding of the photoresistor and its ability to measure the brightness of the room, where the resistance of the photoresistor is inversely proportional to the brightness of the light.

   ✓ Hook up a photoresistor in series with a fixed resistor to obtain a measurable change in voltage across the series resistor.

   ✓ Develop proficiency in programming the Arduino to read and interpret the analog signals from the photoresistor through the serial monitor.

   2.2) Demonstrate knowledge of the practical application of photoresistors in electronic circuits by implementing a project involving street light control using a red and green LED.

   ✓ Read the voltage signal via an analog pin on the Arduino and use the information to control electronic components such as LEDs.

   ✓ Implement a practical application of a photoresistor by programming an Arduino to control a red and green LED based on the brightness of light, where the green LED is turned on when the light is on and the red LED is turned on when the light is off.

   ✓ Test and troubleshoot the photoresistor circuit and Arduino sketch to ensure accurate and reliable operation.

3) **Create an audible signal that changes tone based on the brightness of the light in the room by utilizing a photoresistor to measure light and a passive buzzer to produce the audible signal.**

   3.1) Effectively implement the photoresistor and passive buzzer in an electronic circuit, ensuring accurate and reliable measurement of the light and creation of the audible signal.

   ✓ Use the serial monitor to monitor the changes in the analog values of the photoresistor and ensure that the audible signal is proportional to the brightness of the room.

   3.2) Implementation of a Photoresistor-Based Audible Signal with control Button

   ✓ Using a button to start/stop the tone generation loop in an Arduino sketch. To start/stop the sound, you can add a conditional statement that checks for a certain condition, such as a button press, and sets a variable to start/stop the loop that generates the tone.

4) **Exploring RGB Mixing with Arduino: Connecting, Programming, and Controlling Primary and Intermediate Colors using PWM Signals**

   4.1) Develop the ability to experiment with RGB mixing and create other colors by varying the levels of Red, Green, and Blue channels.

✓ Learn the fundamental concepts of RGB LEDs and how they work to produce different colors.

✓ Understand the differences between common anode and common cathode RGB LEDs and learn how to connect a common cathode RGB LED to an Arduino.

✓ Program an Arduino to get the primary colors by mixing different combinations of Red, Green, and Blue channels using PWM signals.

4.2) Achieve the ability to connect an RGB LED and program it to display primary and intermediate colors based on user input.

✓ Develop a program that allows user for input to control the primary and intermediate colors displayed by the RGB LED.

✓ Create examples of primary colors, such as Cyan, Magenta, and Yellow, and explore the possibilities of mixing in-between colors.

✓ Test the program and RGB LED thoroughly to ensure accurate and reliable color mixing.

5) Create an audible signal that changes tone based on the brightness of the light in the room by utilizing a photoresistor to measure light and a passive buzzer to produce the audible signal.

✓ Write an Arduino sketch that reads the analog value from the LDR, uses this value to adjust the color and tone of the RGB LED and the pitch of the buzzer, and displays the light level and other relevant data on the serial monitor. The start/stop button allows the user to control the sound and light of the lamp. When the button is pressed, the flags for stopSound and stopLight are set to true, and the sound and light are stopped.

Pre-Lab Questionnaire:

1) What is the purpose of the "analogRead" command in Arduino?
2) How can you convert analog voltage to voltage reading in volts using Arduino?
3) Can you use the "analogRead" command to measure digital signals?
4) What is the maximum analog voltage that can be read by the Arduino?
5) What is the range of values that the "analogRead" command can output on the Arduino platform?
6) What is the function of the map() function in Arduino?
7) What is the relationship between light intensity and the resistance of a photoresistor?
8) How can a photoresistor be used to measure changes in light intensity?
9) What is the resistance range of a typical photoresistor in bright light conditions?

Answers to Pre-Lab Questions

Components/Equipment Required:

Sl. No.	Name of the Component / Equipment	Specification	Quantity
1)	Arduino UNO R3	16MHz	1
2)	Arduino UNO cable	USB Type A to Micro-B	1
3)	Trimmer Potentiometer	10k, Preset	2
4)	LDR/Photoresistor	5mm	1
5)	RGB Led (4 pin)	5mm, Common Cathode	1
6)	Resistors (carbon type)	¼ watt (100Ω)	1
		¼ watt (330Ω)	3
		¼ watt (10Ω)	1
7)	LED	Any 2 different colour of your choice	2
8)	Push Button	4-legged Tactile switch (5mm)	9
9)	Buzzer	5v, small	1
10)	Breadboard	840 Tie points	1
11)	Digital Multimeter	---------------------------	1
12)	Jumper Wire	---------------------------	As per requirement

Objective 1

Develop a comprehensive understanding of analog voltage and potentiometers.

1.1) Read analog voltage using the "analogRead" command and convert the analog voltage to a voltage reading in volts

1.2) Read and interpret the voltage signals produced by a potentiometer.

1.3) Read the analog value from the potentiometer and converts it to a voltage reading in volts using Arduino and serial monitor

1.4) Controlling LED Brightness with Potentiometer and Serial Monitoring using Arduino

1.4.1) Controlling LED brightness with potentiometer without using map() function.

1.4.2) Controlling LED brightness with potentiometer using map() function.

1.5) Optimizing Electronic Control with Potentiometers: A Practical Approach to Controlling Speaker Volume and LED Brightness with Arduino

Circuit / Schematic Diagram

1.1) Read analog voltage using the "analogRead" command and convert the analog voltage to a voltage reading in volts

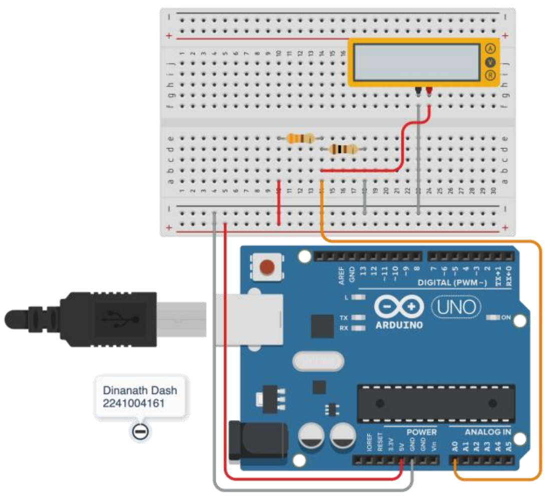


Figure 1.1: Schematic of reading analog voltage using the "analogRead" command and convert the analog voltage to a voltage reading in volts

Code

```
int sensorPin = A0;
float Vref = 5.0;
int adcValue;
float voltage;

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

void loop() {
  adcValue = analogRead(sensorPin);
  voltage = (adcValue * Vref) / 1023.0;
  Serial.print("ADC Value: ");
  Serial.print(adcValue);
  Serial.print("  Voltage: ");
  Serial.println(voltage);
  delay(500);
}
```

1.2) Read and interpret the voltage signals produced by a potentiometer.

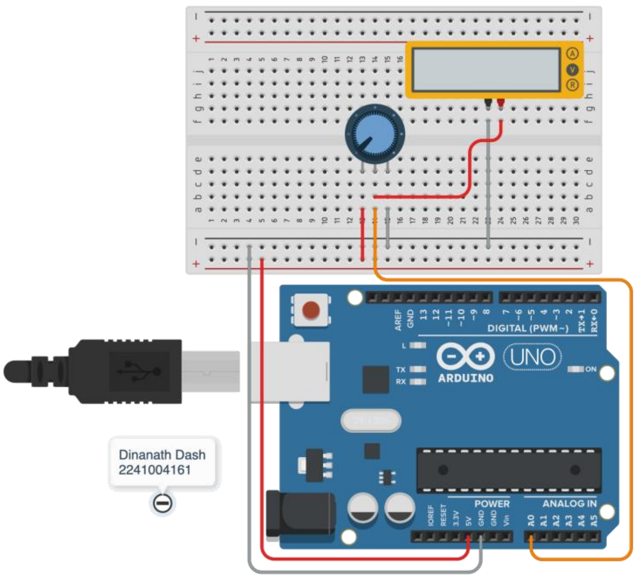


Figure 1.2: Schematic of reading and interpreting the voltage signals produced by a potentiometer.

Code

```
int potPin = A0;
int potValue;

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

void loop() {
  potValue = analogRead(potPin);
  Serial.print("Potentiometer Value: ");
  Serial.println(potValue);
  delay(500);
}
```

1.3) Read the analog value from the potentiometer and converts it to a voltage reading in volts using Arduino and serial monitor

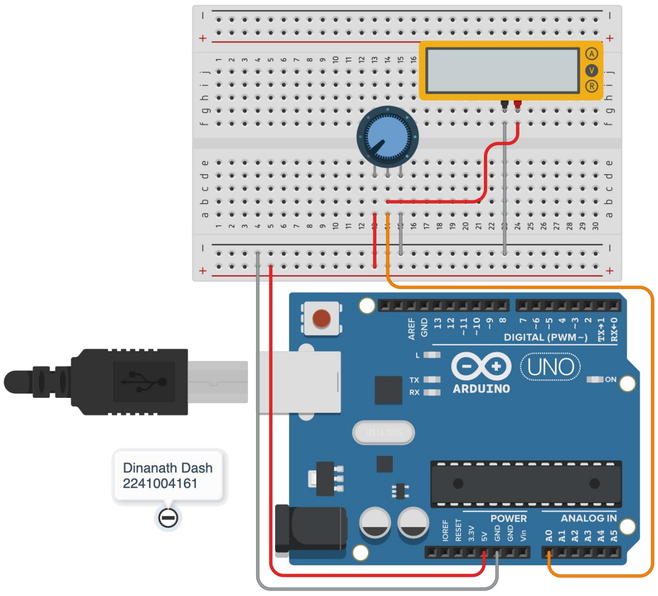


Figure 1.3: Schematic of reading the analog value from the potentiometer and converts it to a voltage reading in volts using Arduino and serial monitor

Code

```
int potPin = A0;
float Vref = 5.0;
int potValue;
float voltage;

void setup() {
  Serial.begin(9600);
}
void loop() {
  potValue = analogRead(potPin);
  voltage = (potValue * Vref) / 1023.0;
  Serial.print("ADC Value: ");
  Serial.print(potValue);
  Serial.print("  Voltage: ");
  Serial.println(voltage);
  delay(500);
}
```

1.4) Controlling LED Brightness with Potentiometer and Serial Monitoring using Arduino

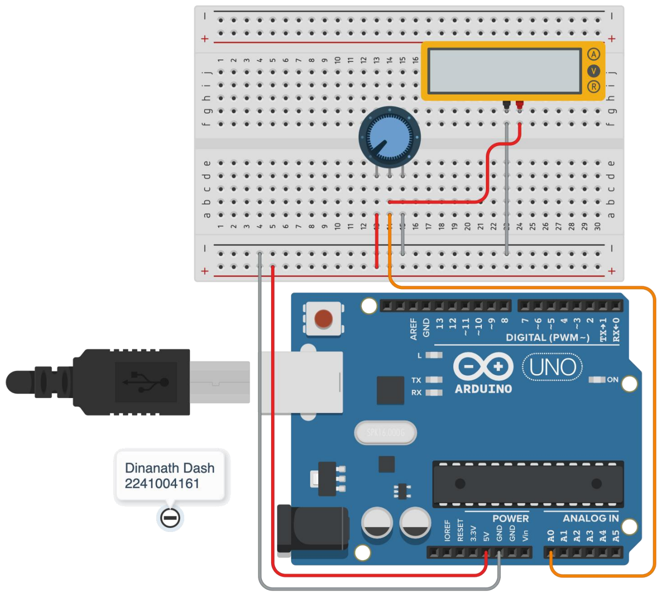


Figure 1.4: Schematic of controlling LED Brightness with Potentiometer and Serial Monitoring using Arduino

1.4.1) Controlling LED brightness with potentiometer without using map() function.

Code

```
int potPin = A0;
int ledPin = 9;
void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
}
void loop() {
  int potValue = analogRead(potPin);
  int brightness = potValue / 4;
  analogWrite(ledPin, brightness);
  Serial.print("Potentiometer: ");
  Serial.print(potValue);
  Serial.print("  Brightness: ");
  Serial.println(brightness);
  delay(500);
}
```

1.4.2) Controlling LED brightness with potentiometer using map() function.

Code

```
int potPin = A0;
int ledPin = 9;
void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
}
void loop() {
  int potValue = analogRead(potPin);
  int brightness = potValue / 4;
  analogWrite(ledPin, brightness);
  Serial.print("Potentiometer: ");
  Serial.print(potValue);
  Serial.print("  Brightness: ");
  Serial.println(brightness);
  delay(500);
}
```

1.5) Read the analog value from the potentiometer and converts it to a voltage reading in volts using Arduino and serial monitor

Code

```
int potPin = A0;
int ledPin = 9;
void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
}
```

```
void loop() {
  int potValue = analogRead(potPin);
  int brightness = map(potValue, 0, 1023, 0, 255);
  analogWrite(ledPin, brightness);
  Serial.print("Potentiometer: ");
  Serial.print(potValue);
  Serial.print("  Brightness: ");
  Serial.println(brightness);
  delay(500);
}
```

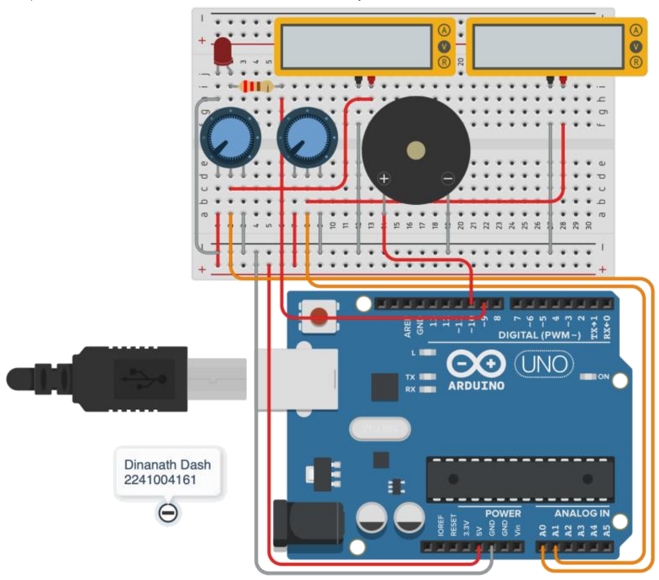
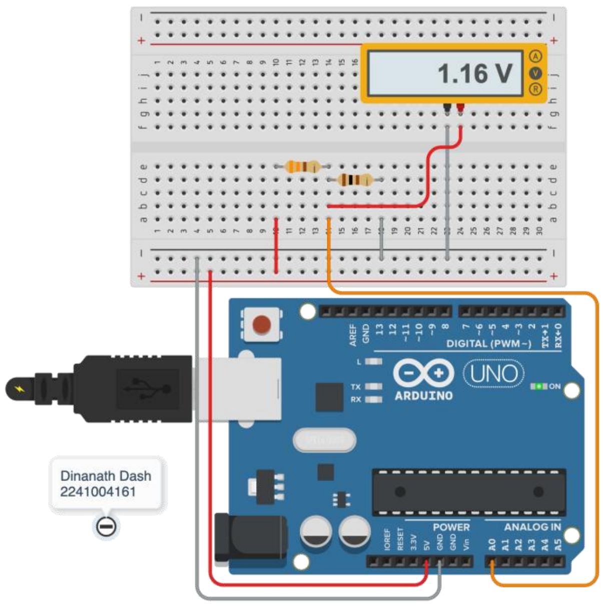
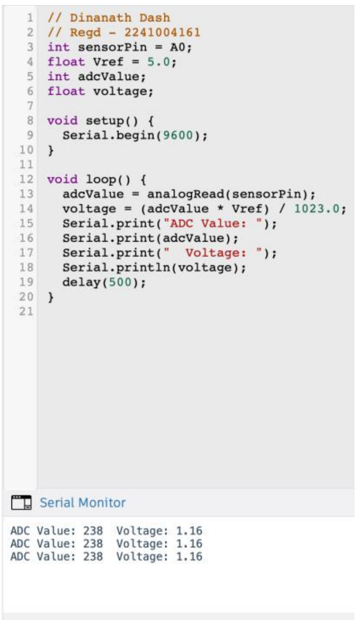


Figure 1.5: Schematic of optimizing Electronic Control with Potentiometers: A Practical Approach to Controlling Speaker Volume and LED Brightness with Arduin0

Observation

Figure 1.1: Simulation based reading analog voltage using the "analogRead" command and convert the analog voltage to a voltage reading in volts



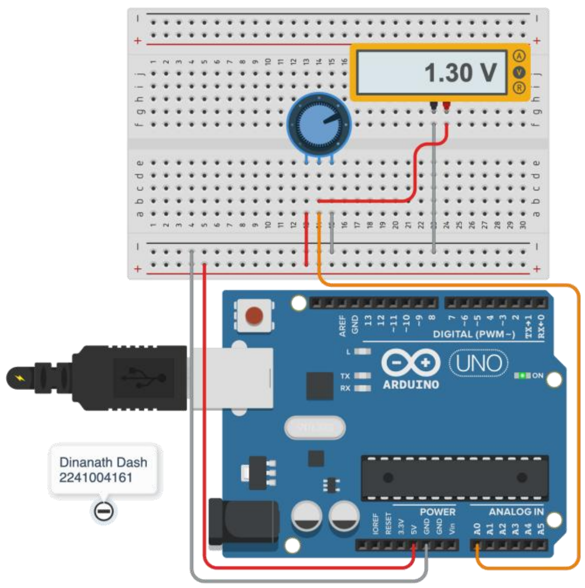


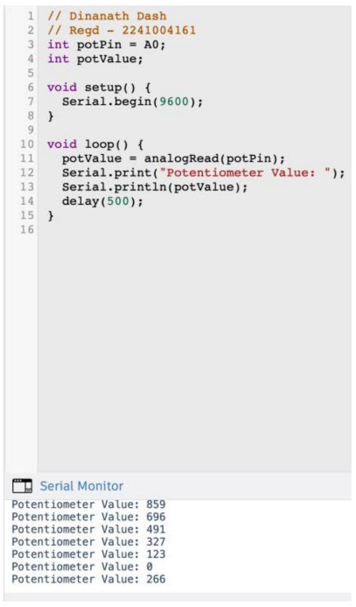
```
// Dinanath Dash
// Regd - 2241004161
int sensorPin = A0;
float Vref = 5.0;
int adcValue;
float voltage;

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

void loop() {
  adcValue = analogRead(sensorPin);
  voltage = (adcValue * Vref) / 1023.0;
  Serial.print("ADC Value: ");
  Serial.print(adcValue);
  Serial.print("  Voltage: ");
  Serial.println(voltage);
  delay(500);
}
```

Serial Monitor

```
ADC Value: 238  Voltage: 1.16
ADC Value: 238  Voltage: 1.16
ADC Value: 238  Voltage: 1.16
```

Figure 1.2: Simulation based reading and interpreting the voltage signals produced by a potentiometer.



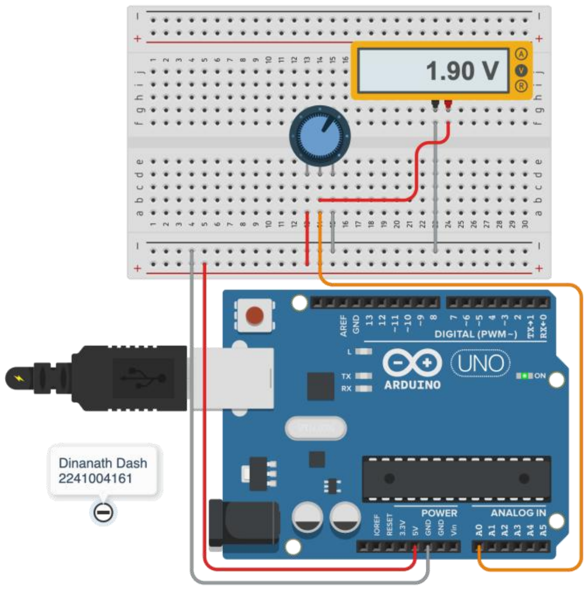


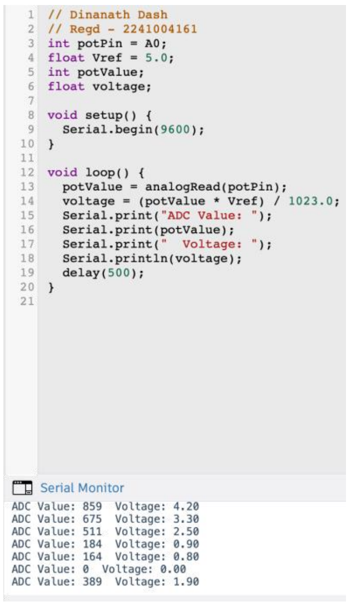
```
// Dinanath Dash
// Regd - 2241004161
int potPin = A0;
int potValue;

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

void loop() {
  potValue = analogRead(potPin);
  Serial.print("Potentiometer Value: ");
  Serial.println(potValue);
  delay(500);
}
```

Serial Monitor

```
Potentiometer Value: 859
Potentiometer Value: 696
Potentiometer Value: 491
Potentiometer Value: 327
Potentiometer Value: 123
Potentiometer Value: 0
Potentiometer Value: 266
```

Figure 1.3: Simulation based reading and interpreting the voltage signals produced by a potentiometer.



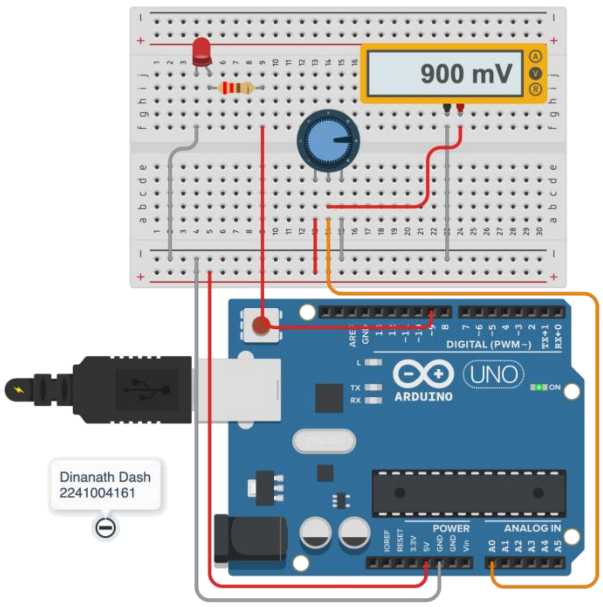


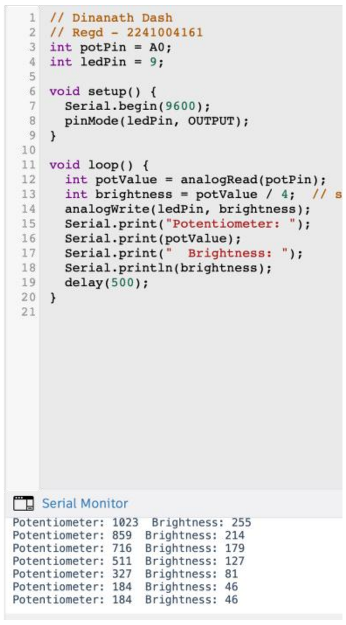
```
// Dinanath Dash
// Regd - 2241004161
int potPin = A0;
float Vref = 5.0;
int potValue;
float voltage;

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

void loop() {
  potValue = analogRead(potPin);
  voltage = (potValue * Vref) / 1023.0;
  Serial.print("ADC Value: ");
  Serial.print(potValue);
  Serial.print("  Voltage: ");
  Serial.println(voltage);
  delay(500);
}
```

Serial Monitor

```
ADC Value: 859  Voltage: 4.20
ADC Value: 675  Voltage: 3.30
ADC Value: 511  Voltage: 2.50
ADC Value: 184  Voltage: 0.90
ADC Value: 164  Voltage: 0.80
ADC Value: 0  Voltage: 0.00
ADC Value: 389  Voltage: 1.90
```

Figure 1.4: Simulation based controlling LED Brightness with Potentiometer and Serial Monitoring using Arduino



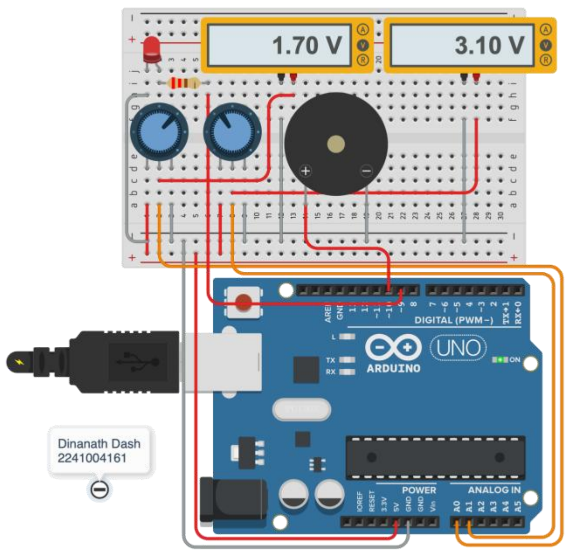


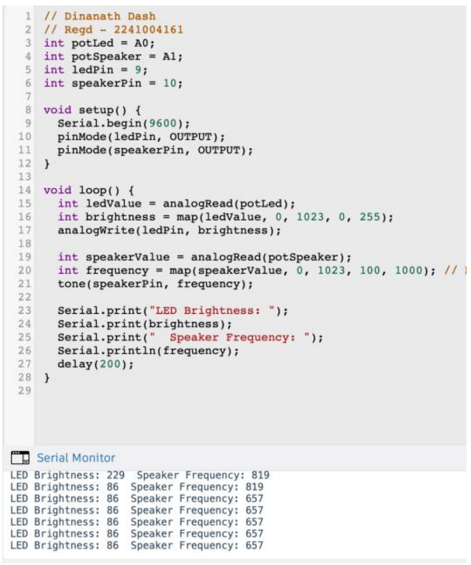
```
// Dinanath Dash
// Regd - 2241004161
int potPin = A0;
int ledPin = 9;

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

void loop() {
  int potValue = analogRead(potPin);
  int brightness = potValue / 4;  // s
  analogWrite(ledPin, brightness);
  Serial.print("Potentiometer: ");
  Serial.print(potValue);
  Serial.print("  Brightness: ");
  Serial.println(brightness);
  delay(500);
}
```

Serial Monitor

```
Potentiometer: 1023  Brightness: 255
Potentiometer: 859  Brightness: 214
Potentiometer: 716  Brightness: 179
Potentiometer: 511  Brightness: 127
Potentiometer: 327  Brightness: 81
Potentiometer: 184  Brightness: 46
Potentiometer: 184  Brightness: 46
```

Figure 1.5.1: Simulation based optimizing Electronic Control with Potentiometers: A Practical Approach to Controlling Speaker Volume and LED Brightness with Arduino



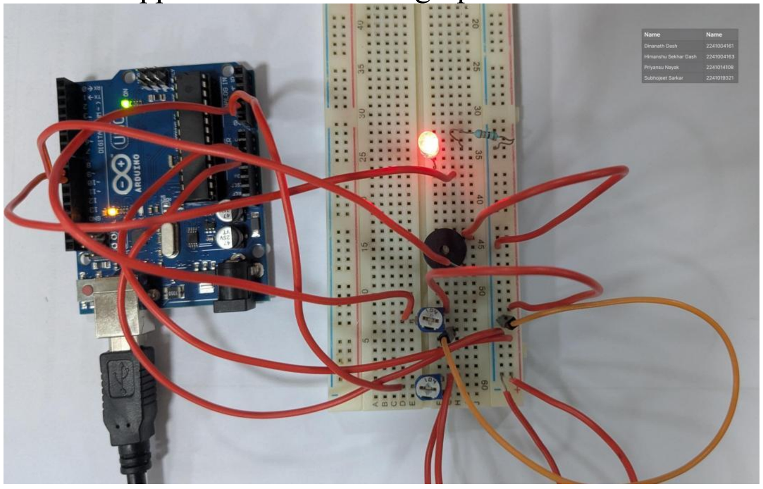


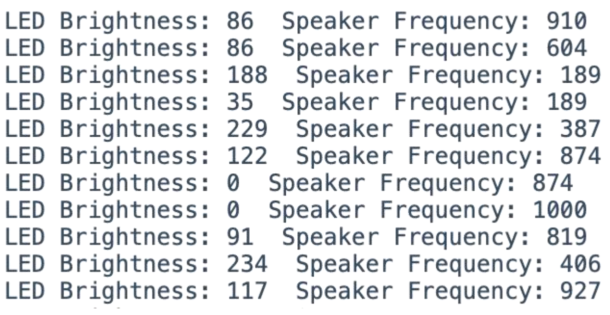
```
// Dinanath Dash
// Regd - 2241004161
int potLed = A0;
int potSpeaker = A1;
int ledPin = 9;
int speakerPin = 10;

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

void loop() {
  int ledValue = analogRead(potLed);
  int brightness = map(ledValue, 0, 1023, 0, 255);
  analogWrite(ledPin, brightness);

  int speakerValue = analogRead(potSpeaker);
  int frequency = map(speakerValue, 0, 1023, 100, 1000); //
  tone(speakerPin, frequency);

  Serial.print("LED Brightness: ");
  Serial.print(brightness);
  Serial.print("  Speaker Frequency: ");
  Serial.println(frequency);
  delay(200);
}
```

Serial Monitor

```
LED Brightness: 229  Speaker Frequency: 819
LED Brightness: 86  Speaker Frequency: 819
LED Brightness: 86  Speaker Frequency: 657
LED Brightness: 86  Speaker Frequency: 657
LED Brightness: 86  Speaker Frequency: 657
LED Brightness: 86  Speaker Frequency: 657
LED Brightness: 86  Speaker Frequency: 657
```

Figure 1.5.2: Hardware Implementation based optimizing Electronic Control with Potentiometers: A Practical Approach to Controlling Speaker Volume and LED Brightness with Arduino.





```
LED Brightness: 86  Speaker Frequency: 910
LED Brightness: 86  Speaker Frequency: 604
LED Brightness: 188  Speaker Frequency: 189
LED Brightness: 35  Speaker Frequency: 189
LED Brightness: 229  Speaker Frequency: 387
LED Brightness: 122  Speaker Frequency: 874
LED Brightness: 0  Speaker Frequency: 874
LED Brightness: 0  Speaker Frequency: 1000
LED Brightness: 91  Speaker Frequency: 819
LED Brightness: 234  Speaker Frequency: 406
LED Brightness: 117  Speaker Frequency: 927
```

Objective 2

Understand the function and operation of an LDR/Photoresistor in electronic circuits and its ability to measure changes in light intensity.

- **2.1) Comprehend the relationship between light intensity and the resistance of photoresistors and how they can be used to measure changes in light intensity.**
- **2.2) Demonstrate knowledge of the practical application of photoresistors in electronic circuits by implementing a project involving street light control using a red and green LED.**

Circuit / Schematic Diagram

2.1) Comprehend the relationship between light intensity and the resistance of photoresistors and how they can be used to measure changes in light intensity.

Code

```
int ldrPin = A0;
int ldrValue;
void setup() {
  Serial.begin(9600);
}
void loop() {
  ldrValue = analogRead(ldrPin);
  Serial.print("LDR Value: ");
  Serial.println(ldrValue);
  delay(500);
}
```

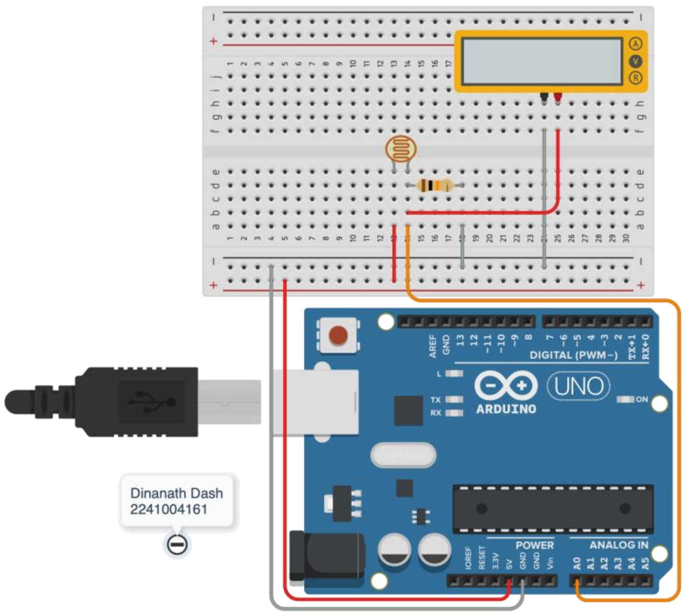


Figure 2.1: Schematic of comprehending the relationship between light intensity and the resistance of photoresistors and how they can be used to measure changes in light intensity.

2.2) Demonstrate knowledge of the practical application of photoresistors in electronic circuits by implementing a project involving street light control using a red and green LED.

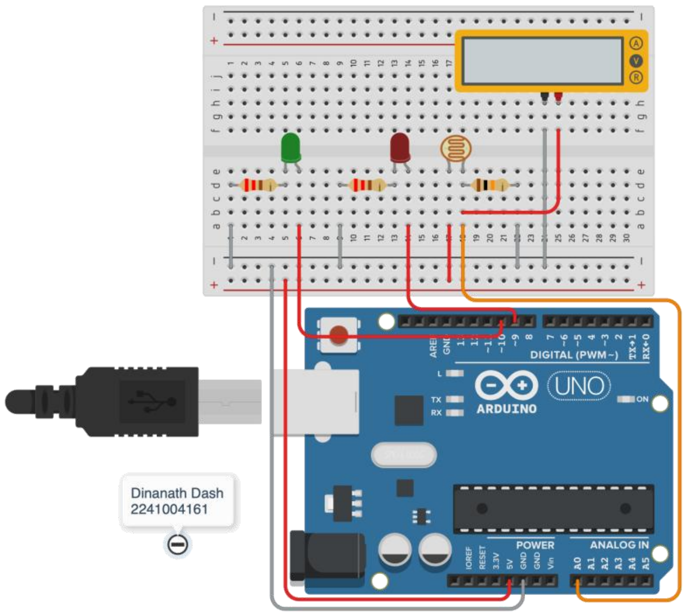


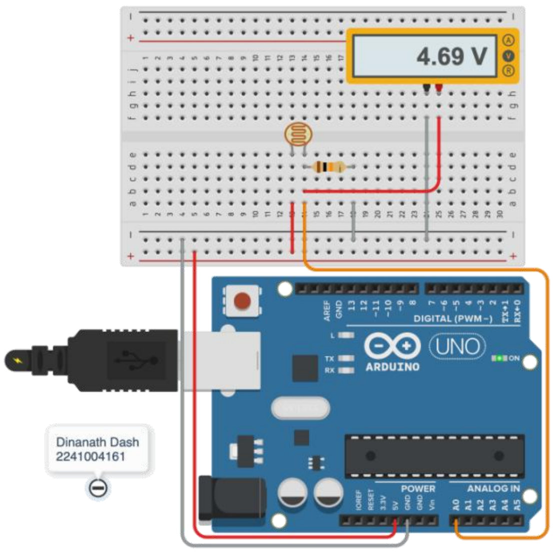
Figure 2.2: Schematic of demonstrating knowledge of the practical application of photoresistors in electronic circuits by implementing a project involving street light control using a red and green LED.

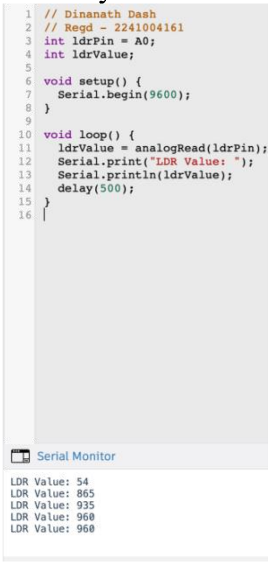
Code

```
int ldrPin = A0;
int redLED = 9;
int greenLED = 10;
int threshold = 500;  // Adjust as per lighting condition
void setup() {
  pinMode(redLED, OUTPUT);
  pinMode(greenLED, OUTPUT);
  Serial.begin(9600);
}
void loop() {
  int lightValue = analogRead(ldrPin);
  Serial.print("Light Intensity: ");
  Serial.println(lightValue);

  if (lightValue < threshold) {  // Dark → Red ON
    digitalWrite(redLED, HIGH);
    digitalWrite(greenLED, LOW);
  } else {  // Bright → Green ON
    digitalWrite(redLED, LOW);
    digitalWrite(greenLED, HIGH);
  }
  delay(500);
}
```

Observation

Figure 2.1: Simulation based comprehending the relationship between light intensity and the resistance of photoresistors and how they can be used to measure changes in light intensity.



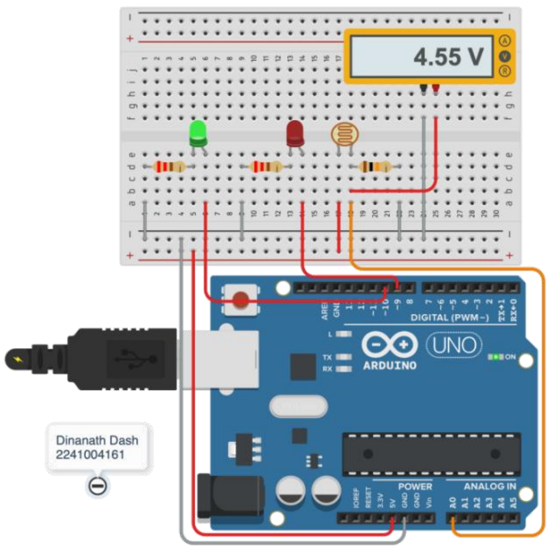


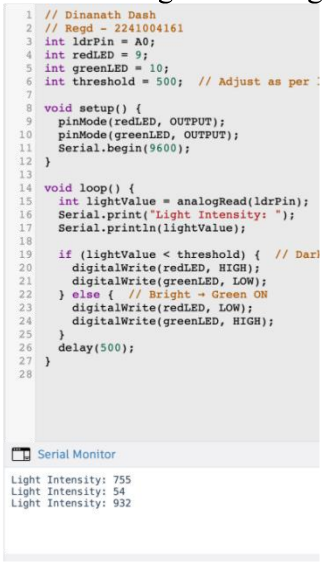
```
// Dinanath Dash
// Regd - 2241004161
int ldrPin = A0;
int ldrValue;

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

void loop() {
  ldrValue = analogRead(ldrPin);
  Serial.print("LDR Value: ");
  Serial.println(ldrValue);
  delay(500);
}
```

Serial Monitor

```
LDR Value: 54
LDR Value: 865
LDR Value: 935
LDR Value: 960
LDR Value: 960
```

Figure 2.2.1: Simulation based demonstrating knowledge of the practical application of photoresistors in electronic circuits by implementing a project involving street light control using a red and green LED.



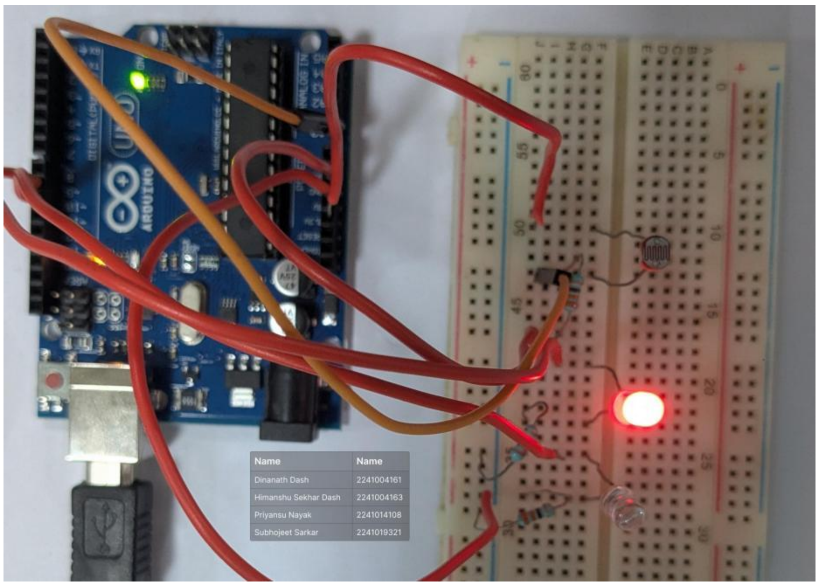


```
// Dinanath Dash
// Regd - 2241004161
int ldrPin = A0;
int redLED = 9;
int greenLED = 10;
int threshold = 500;  // Adjust as per 

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

void loop() {
  int lightValue = analogRead(ldrPin);
  Serial.print("Light Intensity: ");
  Serial.println(lightValue);

  if (lightValue < threshold) {  // Dar
    digitalWrite(redLED, HIGH);
    digitalWrite(greenLED, LOW);
  } else {  // Bright → Green ON
    digitalWrite(redLED, LOW);
    digitalWrite(greenLED, HIGH);
  }
  delay(500);
}
```

Serial Monitor

```
Light Intensity: 755
Light Intensity: 54
Light Intensity: 932
```

Figure 2.2.2: Hardware Implementation based demonstrating knowledge of the practical application of photoresistors in electronic circuits by implementing a project involving street light control using a red and green LED.



Name	Name
Dinanath Dash	2241004161
Himanshu Sekhar Dash	2241004163
Priyansu Nayak	2241014108
Subhojeet Sarkar	2241019321

```
Light Intensity: 935
Light Intensity: 969
Light Intensity: 883
Light Intensity: 973
Light Intensity: 54
Light Intensity: 579
Light Intensity: 435
Light Intensity: 54
```

Objective 3

Create an audible signal that changes tone based on the brightness of the light in the room by utilizing a photoresistor to measure light and a passive buzzer to produce the audible signal.

- **3.1) Effectively implement the photoresistor and passive buzzer in an electronic circuit, ensuring accurate and reliable measurement of the light and creation of the audible signal.**
- **3.2) Implementation of a Photoresistor-Based Audible Signal with control Button**

Circuit / Schematic Diagram

3.1) Effectively implement the photoresistor and passive buzzer in an electronic circuit, ensuring accurate and reliable measurement of the light and creation of the audible signal.

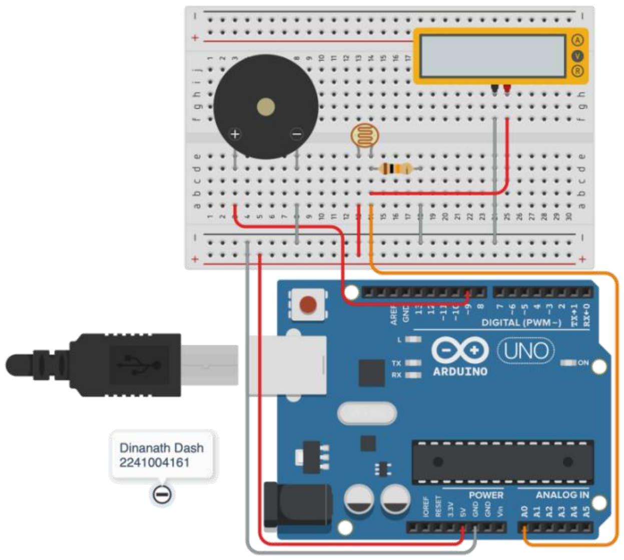


Figure 3.1: Schematic of effectively implement the photoresistor and passive buzzer in an electronic circuit, ensuring accurate and reliable measurement of the light and creation of the audible signal.

Code

```
int ldrPin = A0;
int buzzerPin = 9;
void setup() {
  Serial.begin(9600);
}
void loop() {
  int lightValue = analogRead(ldrPin);
  int toneFreq = map(lightValue, 0, 1023, 1000,
100); // inverse relation
  tone(buzzerPin, toneFreq);

  Serial.print("Light: ");
  Serial.print(lightValue);
  Serial.print("  Frequency: ");
  Serial.println(toneFreq);

  delay(200);
}
```

3.2) Implementation of a Photoresistor-Based Audible Signal with control Button

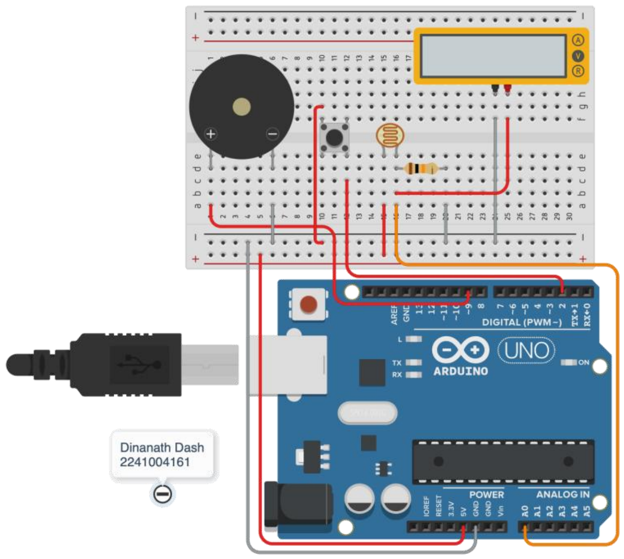


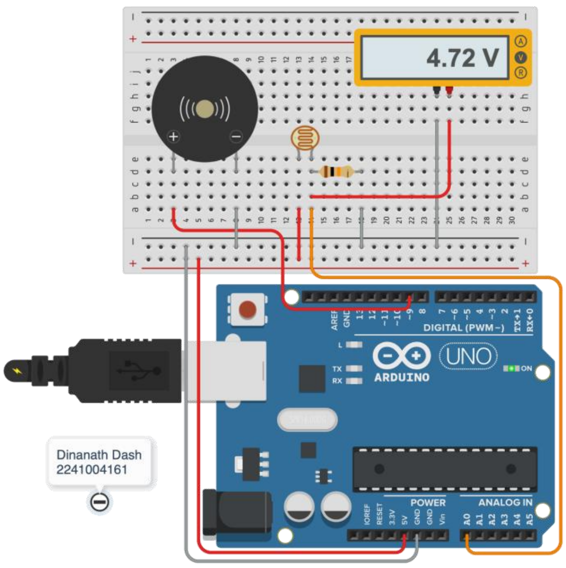
Figure 3.2: Schematic of implementation of a Photoresistor-Based Audible Signal with control Button

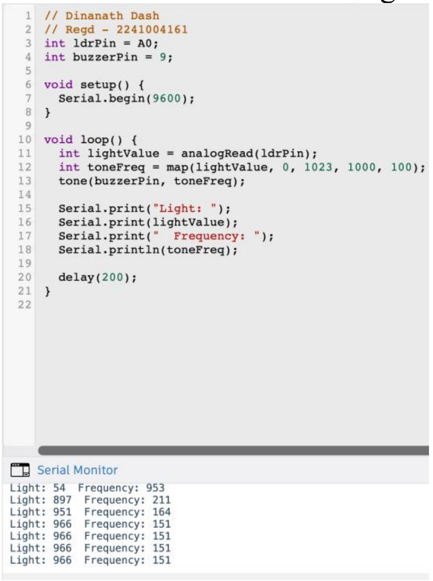
Code

```
const int ldrPin = A0;
const int buzzerPin = 9;
const int buttonPin = 2;
bool soundEnabled = false;
int ldrValue = 0;
void setup() {
  pinMode(buzzerPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);
  Serial.begin(9600);
}
void loop() {
  if (digitalRead(buttonPin) == LOW) {
    soundEnabled = !soundEnabled; // toggle on/off
    delay(300); // debounce delay
  }
  ldrValue = analogRead(ldrPin);
  Serial.print("LDR Value: ");
  Serial.println(ldrValue);
  if (soundEnabled) {
    int frequency = map(ldrValue, 0, 1023, 1000, 100);
    tone(buzzerPin, frequency);
  } else {
    noTone(buzzerPin);
  }
  delay(100);
}
```

Observation

Figure 3.1: Simulation based effectively implement the photoresistor and passive buzzer in an electronic circuit, ensuring accurate and reliable measurement of the light and creation of the audible signal.



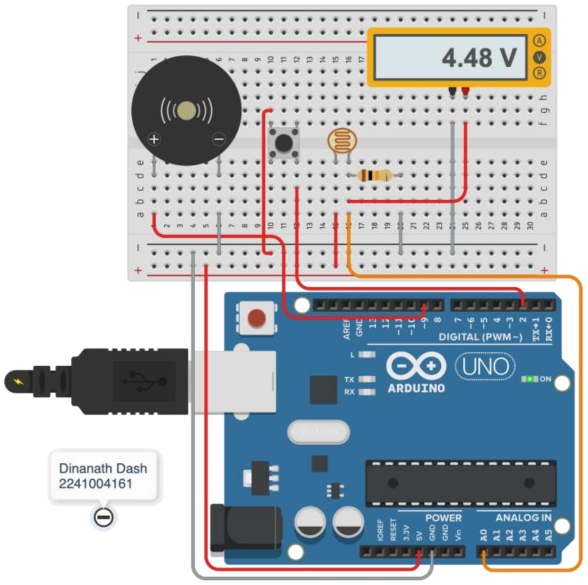


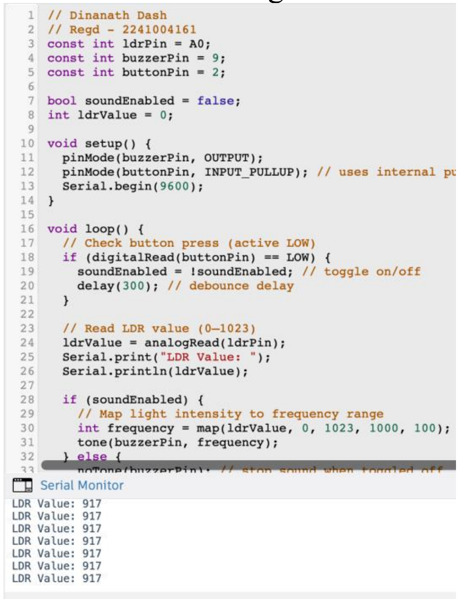
```
// Dinanath Dash
// Regd - 2241004161
int ldrPin = A0;
int buzzerPin = 9;

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

void loop() {
  int lightValue = analogRead(ldrPin);
  int toneFreq = map(lightValue, 0, 1023, 1000, 100);
  tone(buzzerPin, toneFreq);

  Serial.print("Light: ");
  Serial.print(lightValue);
  Serial.print("  Frequency: ");
  Serial.println(toneFreq);

  delay(200);
}
```

Serial Monitor

```
Light: 54  Frequency: 953
Light: 897  Frequency: 211
Light: 951  Frequency: 164
Light: 966  Frequency: 151
Light: 966  Frequency: 151
Light: 966  Frequency: 151
Light: 966  Frequency: 151
```

Figure 3.2.1: Simulation based implementation of a Photoresistor-Based Audible Signal with control Button.



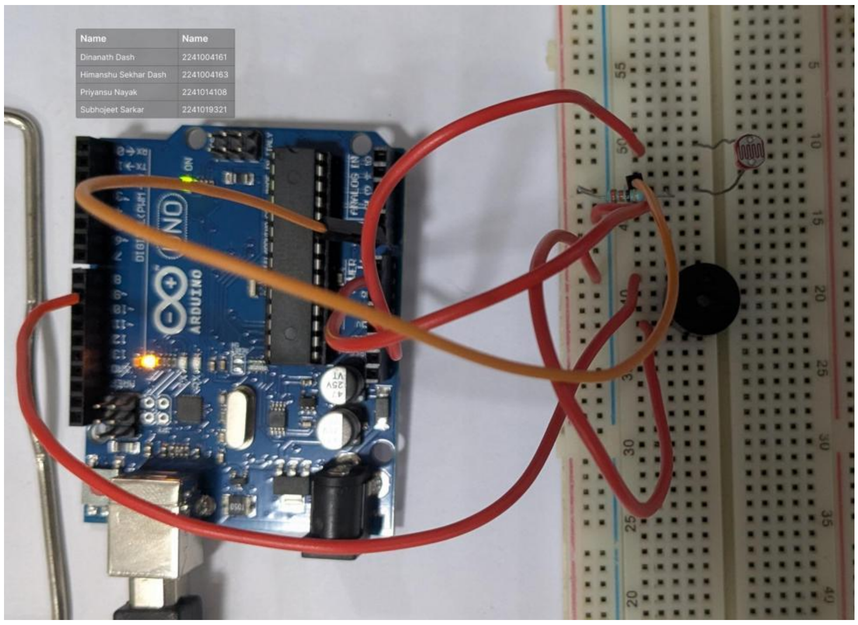


```
// Dinanath Dash
// Regd - 2241004161
const int ldrPin = A0;
const int buzzerPin = 9;
const int buttonPin = 2;

bool soundEnabled = false;
int ldrValue = 0;

void setup() {
  pinMode(buzzerPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP); // uses internal pu
  Serial.begin(9600);
}

void loop() {
  // Check button press (active LOW)
  if (digitalRead(buttonPin) == LOW) {
    soundEnabled = !soundEnabled; // toggle on/off
    delay(300); // debounce delay
  }

  // Read LDR value (0–1023)
  ldrValue = analogRead(ldrPin);
  Serial.print("LDR Value: ");
  Serial.println(ldrValue);

  if (soundEnabled) {
    // Map light intensity to frequency range
    int frequency = map(ldrValue, 0, 1023, 1000, 100);
    tone(buzzerPin, frequency);
  } else {
    noTone(buzzerPin); // stop sound when toggled off
```

Serial Monitor

```
LDR Value: 917
LDR Value: 917
LDR Value: 917
LDR Value: 917
LDR Value: 917
LDR Value: 917
LDR Value: 917
```

Figure 3.2.2: Hardware Implementation based implementation of a Photoresistor-Based Audible Signal with control Button.



Name	Name
Dinanath Dash	2241004161
Himanshu Sekhar Dash	2241004163
Priyansu Nayak	2241014108
Subhojeet Sarkar	2241019321

```
LDR Value: 970
LDR Value: 951
LDR Value: 951
LDR Value: 897
LDR Value: 807
LDR Value: 54
LDR Value: 54
LDR Value: 54
LDR Value: 807
LDR Value: 784
```

Objective 4

Exploring RGB Mixing with Arduino: Connecting, Programming, and Controlling Primary and Intermediate Colors using PWM Signals

4.1) Develop the ability to experiment with RGB mixing and create other colors by varying the levels of Red, Green, and Blue channels.

4.2) Achieve the ability to connect an RGB LED and program it to display primary and intermediate colors based on user input.

Circuit / Schematic Diagram

4.1) Develop the ability to experiment with RGB mixing and create other colors by varying the levels of Red, Green, and Blue channels.

Code

```
int redPin = 9;
int greenPin = 10;
int bluePin = 11;
void setup() {
 pinMode(redPin, OUTPUT);
 pinMode(greenPin, OUTPUT);
 pinMode(bluePin, OUTPUT);
}
void loop() {
 analogWrite(redPin, 255);   // Red
 analogWrite(greenPin, 0);
 analogWrite(bluePin, 0);
 delay(1000);
 analogWrite(redPin, 0);     // Green
 analogWrite(greenPin, 255);
 analogWrite(bluePin, 0);
 delay(1000);
 analogWrite(redPin, 0);     // Blue
 analogWrite(greenPin, 0);
 analogWrite(bluePin, 255);
 delay(1000);
 analogWrite(redPin, 255);   // White
 analogWrite(greenPin, 255);
 analogWrite(bluePin, 255);
 delay(1000);
}
```

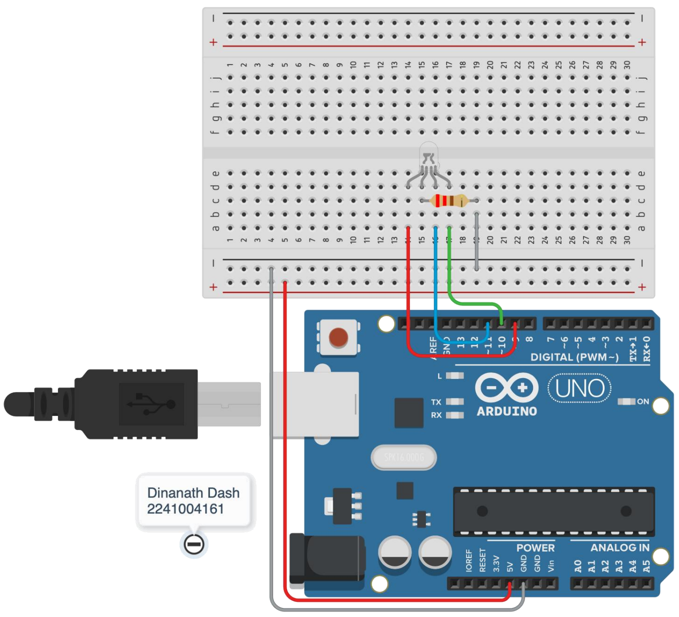


Figure 4.1: Schematic of developing the ability to experiment with RGB mixing and create other colors by varying the levels of Red, Green, and Blue channels

4.2) Achieve the ability to connect an RGB LED and program it to display primary and intermediate colors based on user input.

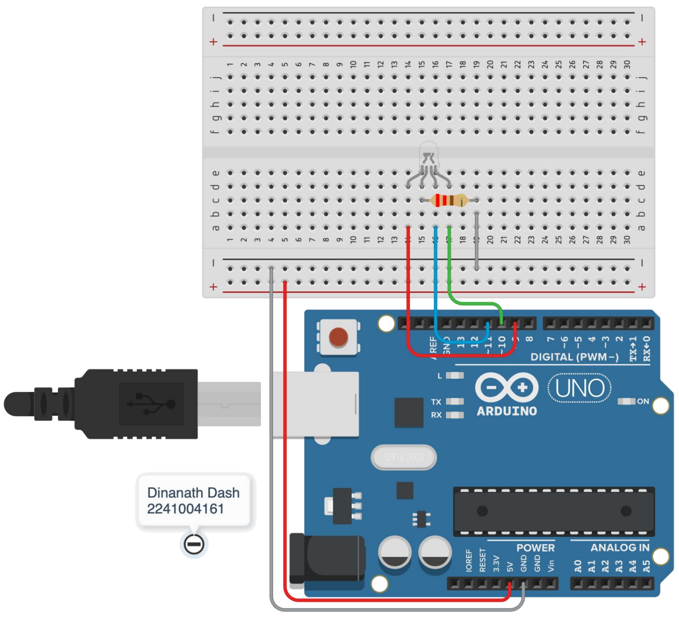


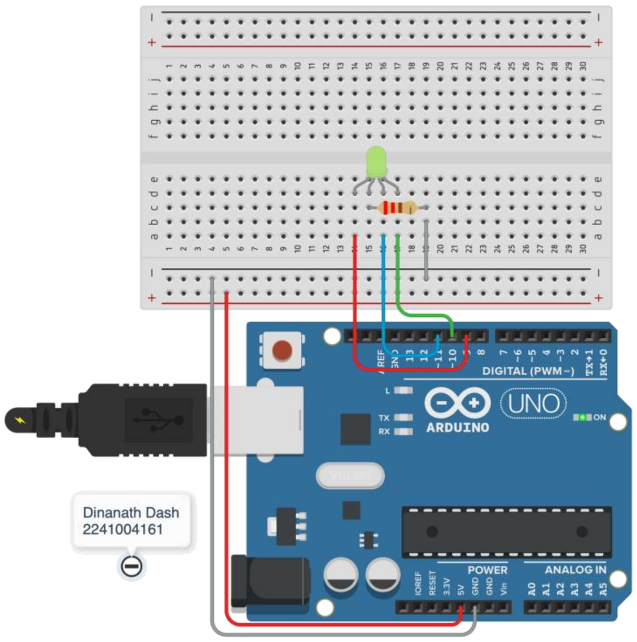
Figure 4.2: Schematic of achieving the ability to connect an RGB LED and program it to display primary and intermediate colors based on user input.

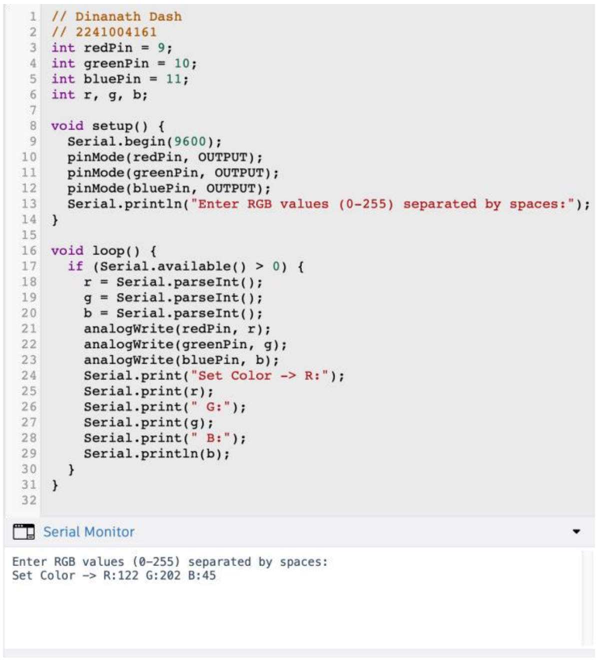
Code

```
int redPin = 9;
int greenPin = 10;
int bluePin = 11;
int r, g, b;
void setup() {
  Serial.begin(9600);
  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);
  Serial.println("Enter RGB values (0-255) separated by spaces:");
}
void loop() {
  if (Serial.available() > 0) {
    r = Serial.parseInt();
    g = Serial.parseInt();
    b = Serial.parseInt();
    analogWrite(redPin, r);
    analogWrite(greenPin, g);
    analogWrite(bluePin, b);
    Serial.print("Set Color -> R:");
    Serial.print(r);
    Serial.print(" G:");
    Serial.print(g);
    Serial.print(" B:");
    Serial.println(b);
  }
}
```

Observation

Figure 4.1.1: Simulation based developing the ability to experiment with RGB mixing and create other colors by varying the levels of Red, Green, and Blue channels and display primary and intermediate colors based on user input.



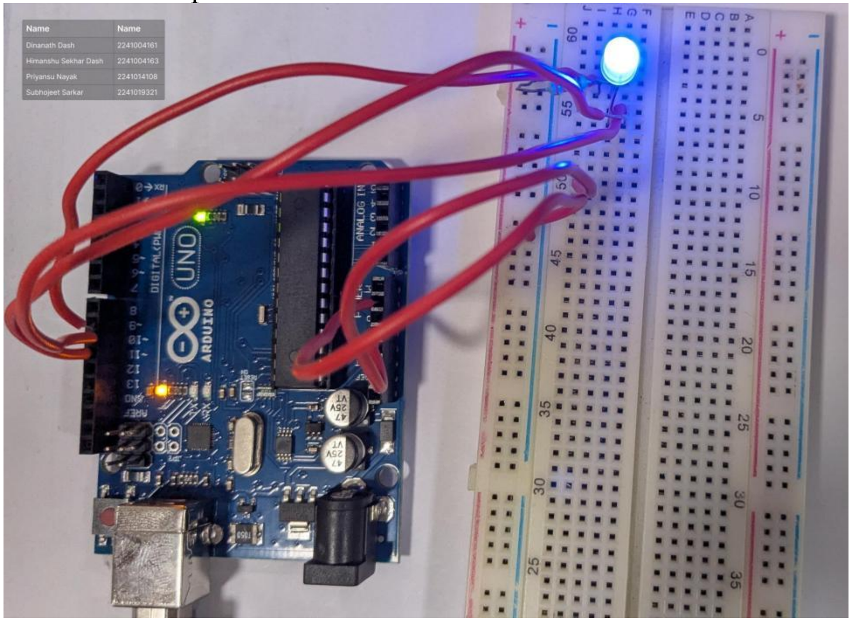


```
// Dinanath Dash
// 2241004161
int redPin = 9;
int greenPin = 10;
int bluePin = 11;
int r, g, b;

void setup() {
  Serial.begin(9600);
  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);
  Serial.println("Enter RGB values (0-255) separated by spaces:");
}

void loop() {
  if (Serial.available() > 0) {
    r = Serial.parseInt();
    g = Serial.parseInt();
    b = Serial.parseInt();
    analogWrite(redPin, r);
    analogWrite(greenPin, g);
    analogWrite(bluePin, b);
    Serial.print("Set Color -> R:");
    Serial.print(r);
    Serial.print(" G:");
    Serial.print(g);
    Serial.print(" B:");
    Serial.println(b);
  }
}
```

Serial Monitor

```
Enter RGB values (0-255) separated by spaces:
Set Color -> R:122 G:202 B:45
```

Figure 4.1.2: Hardware Implementation based developing the ability to experiment with RGB mixing and create other colors by varying the levels of Red, Green, and Blue channels and display primary and intermediate colors based on user input.



Name	Name
Dinanath Dash	2241004161
Himanshu Sekhar Dash	2241004163
Priyansu Nayak	2241014108
Subhojeet Sarkar	2241019321

```
Enter RGB values (0-255) separated by spaces:
Set Color -> R:12 G:50 B:112
```

Objective 5

Create an audible signal that changes tone based on the brightness of the light in the room by utilizing a photoresistor to measure light and a passive buzzer to produce the audible signal.

Circuit / Schematic Diagram

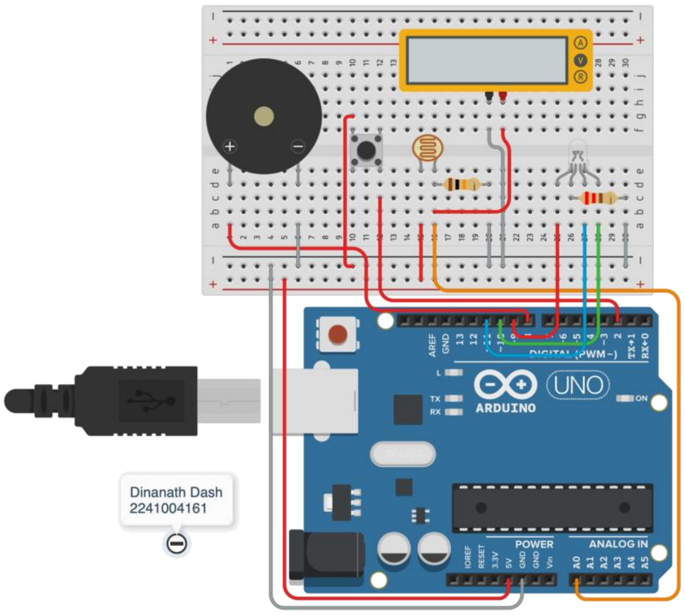


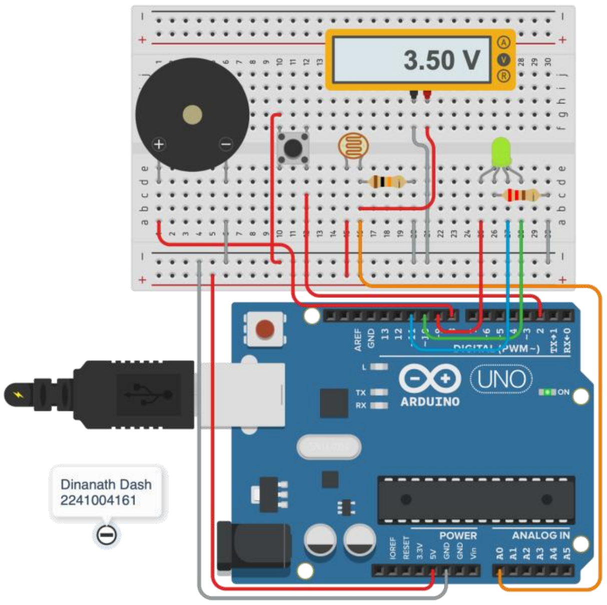
Figure 5.1: Schematic of implementation of an Arduino Ambient Light Mood Lamp with Start/Stop Button Control

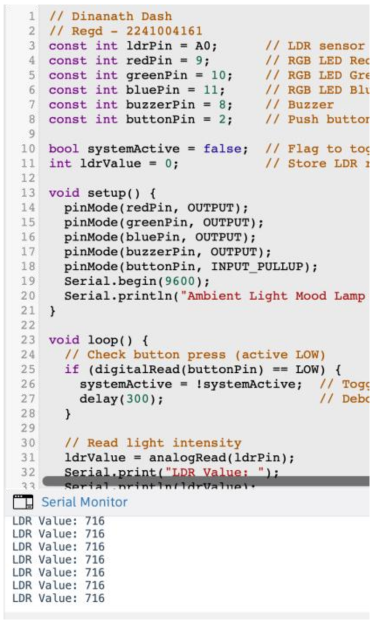
Code

```
const int ldrPin = A0;
const int redPin = 9;
const int greenPin = 10;
const int bluePin = 11;
const int buzzerPin = 8;
const int buttonPin = 2;
bool systemActive = false;
int ldrValue = 0;
void setup() {
  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);
  pinMode(buzzerPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);
  Serial.begin(9600);
  Serial.println("Ambient Light Mood Lamp -
Start/Stop Control");
}
void loop() {
  if (digitalRead(buttonPin) == LOW) {
    systemActive = !systemActive;
    delay(300);
  }
  ldrValue = analogRead(ldrPin);
  Serial.print("LDR Value: ");
  Serial.println(ldrValue);
  if (systemActive) {
    int redVal = map(ldrValue, 0, 1023, 255, 0);
    int greenVal = map(ldrValue, 200, 800, 0, 255);
    int blueVal = map(ldrValue, 600, 1023, 0, 255);
    redVal = constrain(redVal, 0, 255);
    greenVal = constrain(greenVal, 0, 255);
    blueVal = constrain(blueVal, 0, 255);
    analogWrite(redPin, redVal);
    analogWrite(greenPin, greenVal);
    analogWrite(bluePin, blueVal);
    int frequency = map(ldrValue, 0, 1023, 1000,
100);
    tone(buzzerPin, frequency);
  }
  else {
    analogWrite(redPin, 0);
    analogWrite(greenPin, 0);
    analogWrite(bluePin, 0);
    noTone(buzzerPin);
  }
  delay(100);
}
```

Observation

Figure 5.1.1: Simulation based implementation of an Arduino Ambient Light Mood Lamp with Start/Stop Button Control



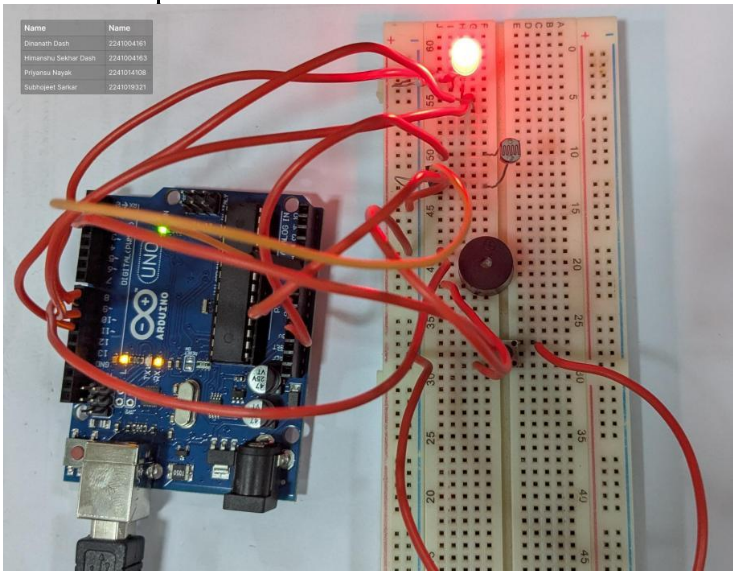


```
// Dinanath Dash
// Regd - 2241004161
const int ldrPin = A0;        // LDR sensor
const int redPin = 9;         // RGB LED Rec
const int greenPin = 10;      // RGB LED Gre
const int bluePin = 11;       // RGB LED Blu
const int buzzerPin = 8;      // Buzzer
const int buttonPin = 2;      // Push buttor

bool systemActive = false;    // Flag to tog
int ldrValue = 0;             // Store LDR r

void setup() {
  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);
  pinMode(buzzerPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);
  Serial.begin(9600);
  Serial.println("Ambient Light Mood Lamp
}

void loop() {
  // Check button press (active LOW)
  if (digitalRead(buttonPin) == LOW) {
    systemActive = !systemActive;  // Togg
    delay(300);                    // Debo
  }

  // Read light intensity
  ldrValue = analogRead(ldrPin);
  Serial.print("LDR Value: ");
```

Serial Monitor

```
LDR Value: 716
LDR Value: 716
LDR Value: 716
LDR Value: 716
LDR Value: 716
LDR Value: 716
LDR Value: 716
```

Figure 5.1.2: Hardware Implementation based implementation of an Arduino Ambient Light Mood Lamp with Start/Stop Button Control


Name Name
Dinanath Dash 2241004161
Himanshu Sekhar Dash 2241004163
Priyansu Nayak 2241014108
Subhojeet Sarkar 2241019321

```
Ambient Light Mood Lamp - Start/Stop Control
LDR Value: 54
LDR Value: 54
LDR Value: 54
LDR Value: 54
LDR Value: 54
LDR Value: 54
LDR Value: 54
LDR Value: 54
LDR Value: 54
LDR Value: 929
LDR Value: 662
LDR Value: 945
LDR Value: 968
LDR Value: 854
```

Conclusion

Precautions

Post Experiment Questionnaire:

1) If the "analogRead" command outputs a value of 512, what is the corresponding voltage reading in volts?
2) How does the value of the analog voltage reading change as the resistance of the second resistor in the voltage divider circuit is increased?
3) How do you calibrate a potentiometer reading in an Arduino sketch?
4) If the analogRead() function on the Arduino returns a value of 1023, what is the corresponding voltage reading in volts assuming a 5V reference voltage?
5) What is the maximum value of the ADC in Arduino?
6) If the ADC value of a potentiometer is 256 and the reference voltage is 5V, what is the voltage reading in volts?
7) If the reference voltage is changed to 3.3V, what is the voltage reading in volts for an ADC value of 750?
8) If the ADC value of a potentiometer is 1023, what is the voltage reading in volts for a reference voltage of 5V?
9) What is the formula to convert ADC value to voltage reading in volts?
10) What is the resolution of the analog-to-digital converter (ADC) in the Arduino board?
11) How does the Arduino handle noisy analog signals?
12) How can you adjust the range of LED brightness values using the map() function?

Answers to Post-Lab Questions

(Signature of the Faculty)	**(Signature of the Student)**
	Name:
Date:	**Registration No.:**
	Branch:
	Section