**PRACTICAL 2 : RGB Light**

**Aim :** To control an RGB LED using Arduino and generate different colors.

**Overview :**

This project demonstrates how to control an RGB LED using Arduino. By adjusting the intensity of red, green and blue components, different colors can be generated. This experiment helps understand Pulse Width Modulation (PWM) and how it can be used to mix colors for various lighting applications.

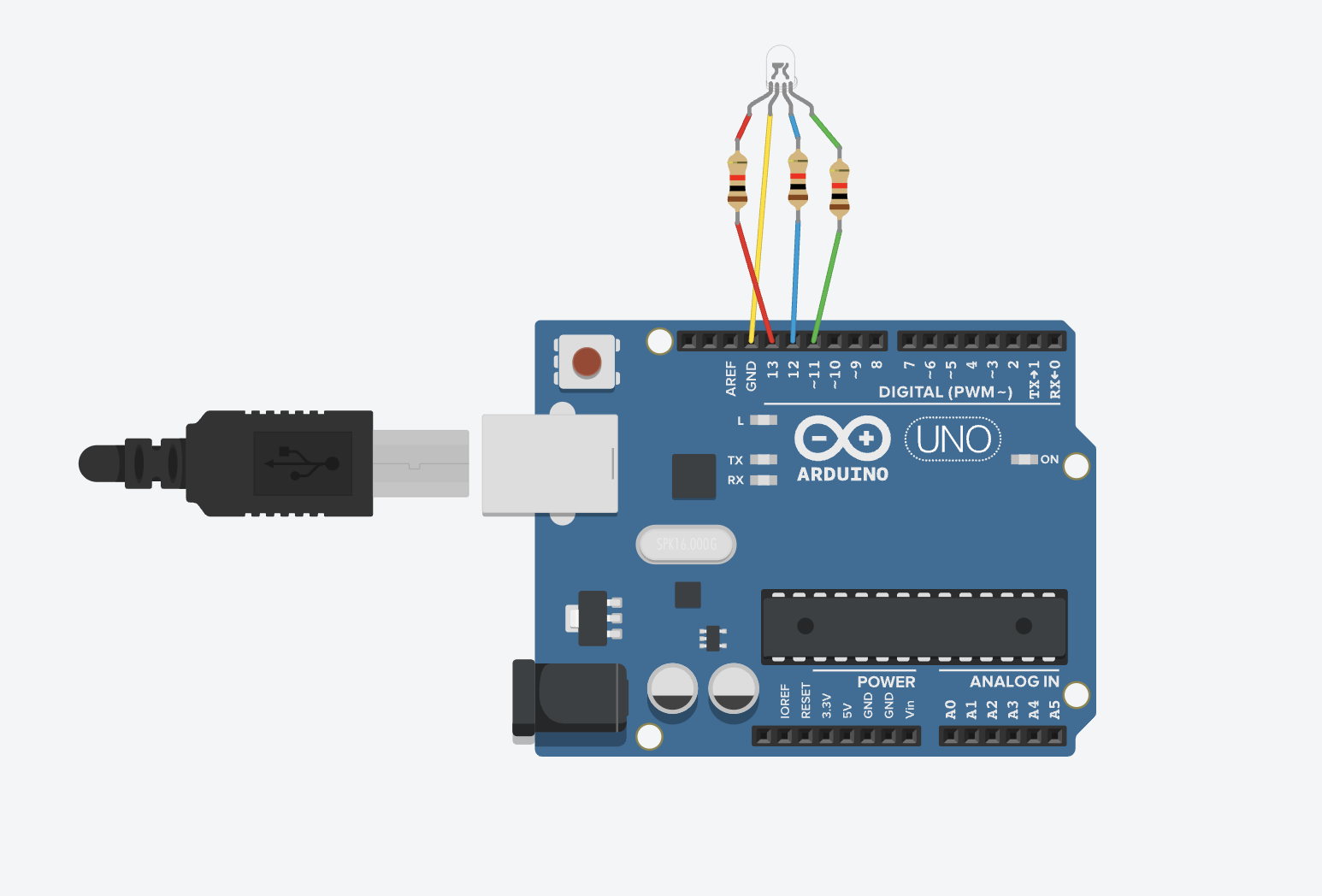
**Materials Required :**

* Arduino Uno R3
* 1 x RCBG LED RGB
* 3 x 1kΩ Resistor
* Jumper Wires
* Arduino IDE (Installed on your Computer)

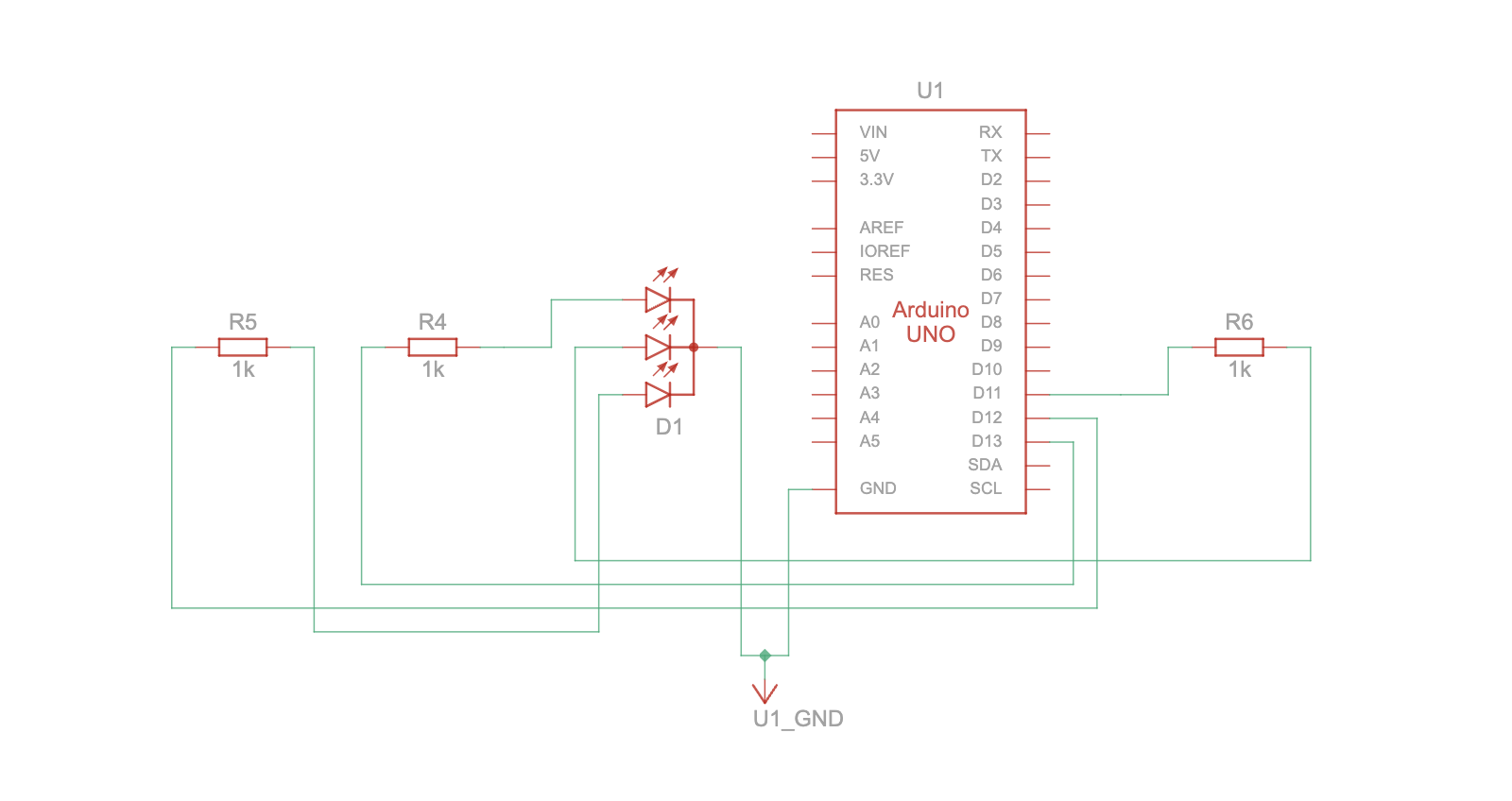
**Circuit Connection and Steps :**

1. **Connect the RGB LED to the Arduino :**
   * Insert the RGB LED into the breadboard. RGB LEDs have 4 pins :
     + The longest pin is the common cathode (ground pin).
     + The other three pins control the Red, Green, and Blue channels.
   * Connect the common cathode pin to the GND pin on the Arduino.
   * Connect the Red, Green and Blue pins of the RGB LED to pins 9, 10 and 11 of the Arduino, respectively.
   * Place a 1kΩ resistor in series with each of the Red, Green and Blue pins to limit the current.
2. **Set up the Arduino environment :**
   * Open the Arduino IDE on your computer.
   * Select the correct board and port from the "Tools" menu.

**Circuit Diagram :**



**Schematic Diagram :**

****

**Code :**

// C++

// Define the pin connections for the RGB LED

int redPin = 13;

int greenPin = 11;

int bluePin = 12;

// Setup function runs once when the program starts

void setup() {

// Set RGB pins as OUTPUT

pinMode(redPin, OUTPUT);

pinMode(greenPin, OUTPUT);

pinMode(bluePin, OUTPUT);

}

// Loop function runs repeatedly

void loop() {

// Red color

analogWrite(redPin, 255); // Full brightness for red

analogWrite(greenPin, 0); // No green

analogWrite(bluePin, 0); // No blue

delay(1000); // Wait for 1 second

// Green color

analogWrite(redPin, 0); // No red

analogWrite(greenPin, 255); // Full brightness for green

analogWrite(bluePin, 0); // No blue

delay(1000); // Wait for 1 second

// Blue color

analogWrite(redPin, 0); // No red

analogWrite(greenPin, 0); // No green

analogWrite(bluePin, 255); // Full brightness for blue

delay(1000); // Wait for 1 second

// Yellow color (Red + Green)

analogWrite(redPin, 255); // Full brightness for red

analogWrite(greenPin, 255); // Full brightness for green

analogWrite(bluePin, 0); // No blue

delay(1000); // Wait for 1 second

// Purple color (Red + Blue)

analogWrite(redPin, 255); // Full brightness for red

analogWrite(greenPin, 0); // No green

analogWrite(bluePin, 255); // Full brightness for blue

delay(1000); // Wait for 1 second

// Cyan color (Green + Blue)

analogWrite(redPin, 0); // No red

analogWrite(greenPin, 255); // Full brightness for green

analogWrite(bluePin, 255); // Full brightness for blue

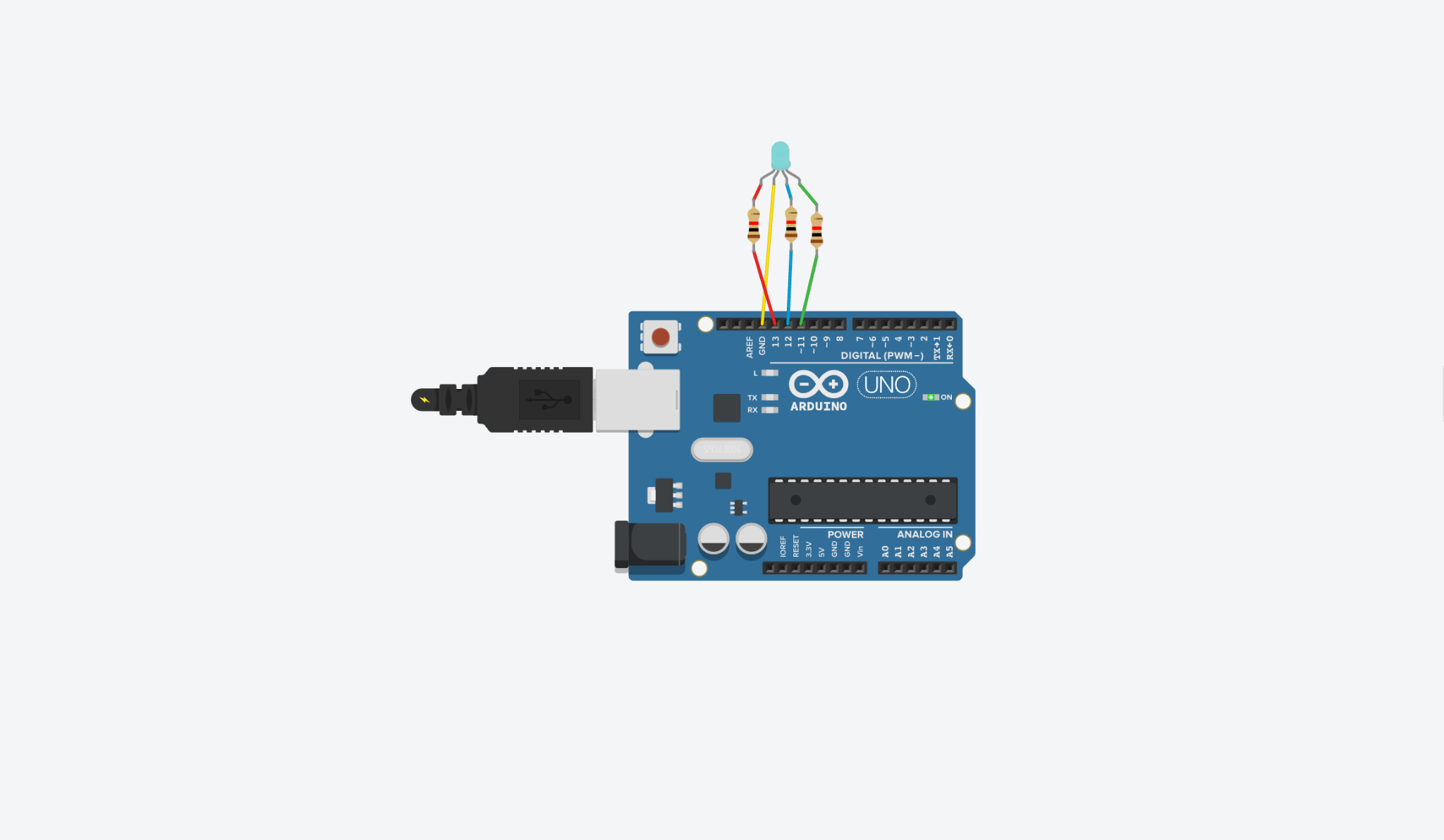
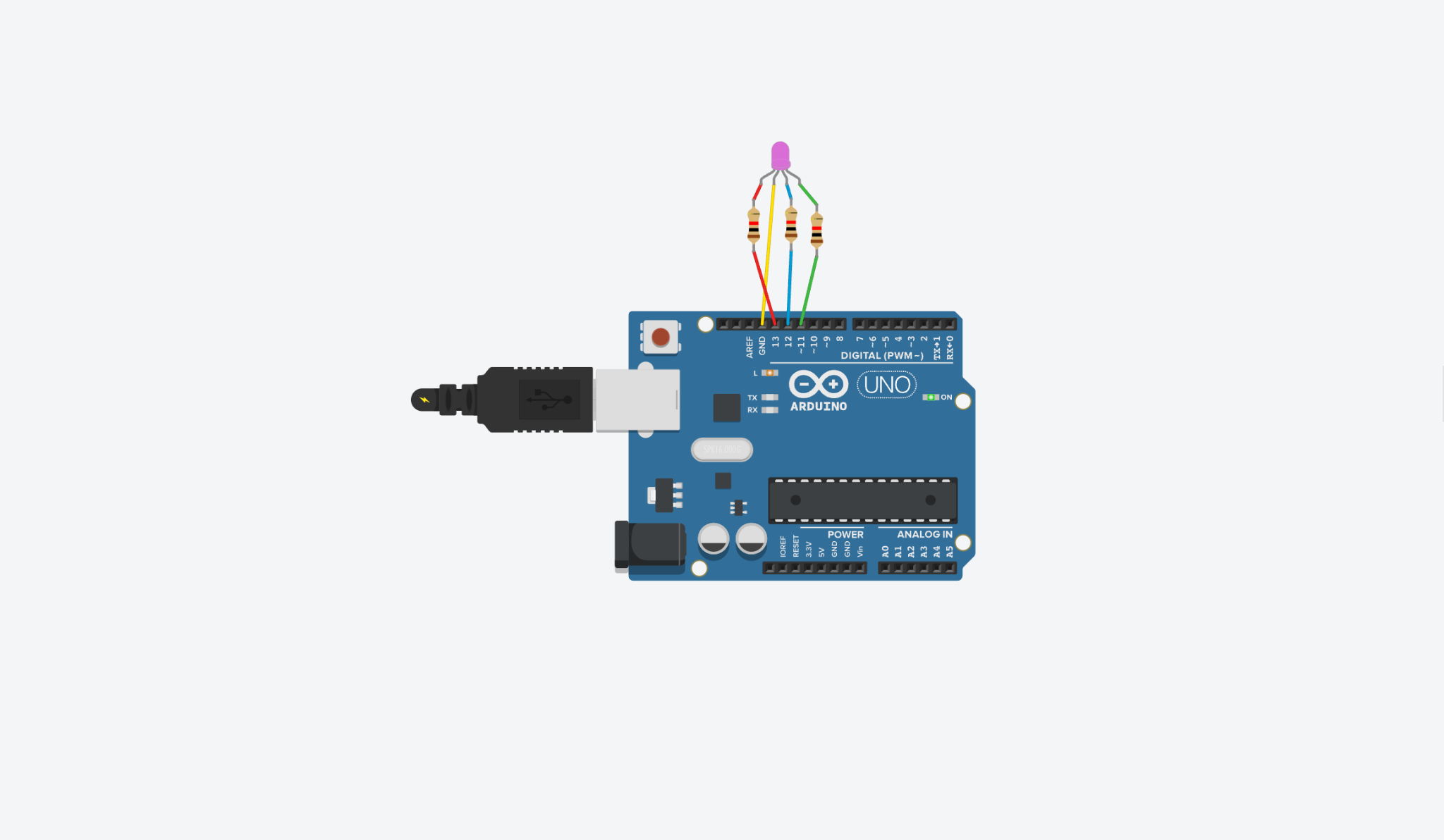
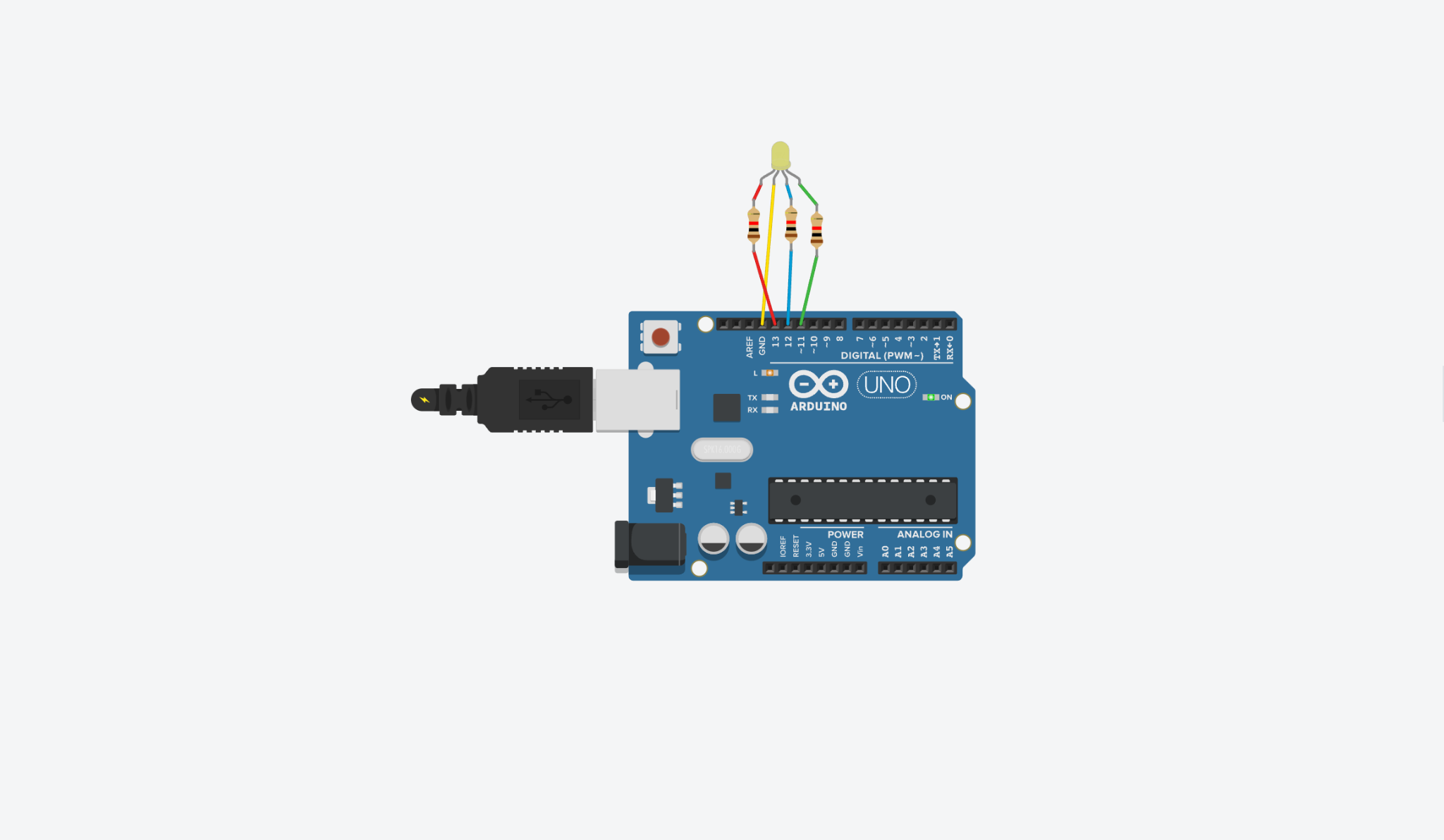
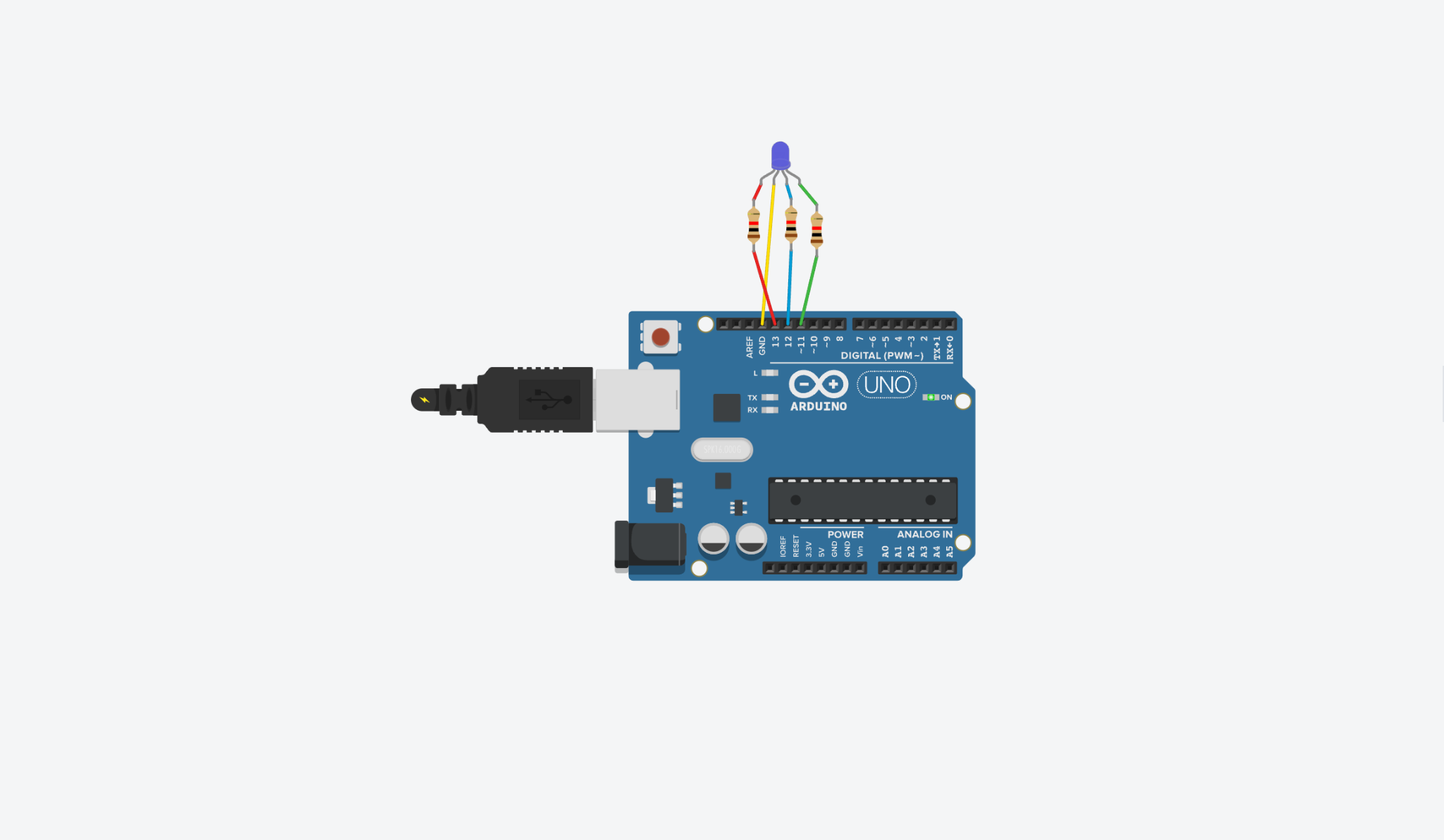
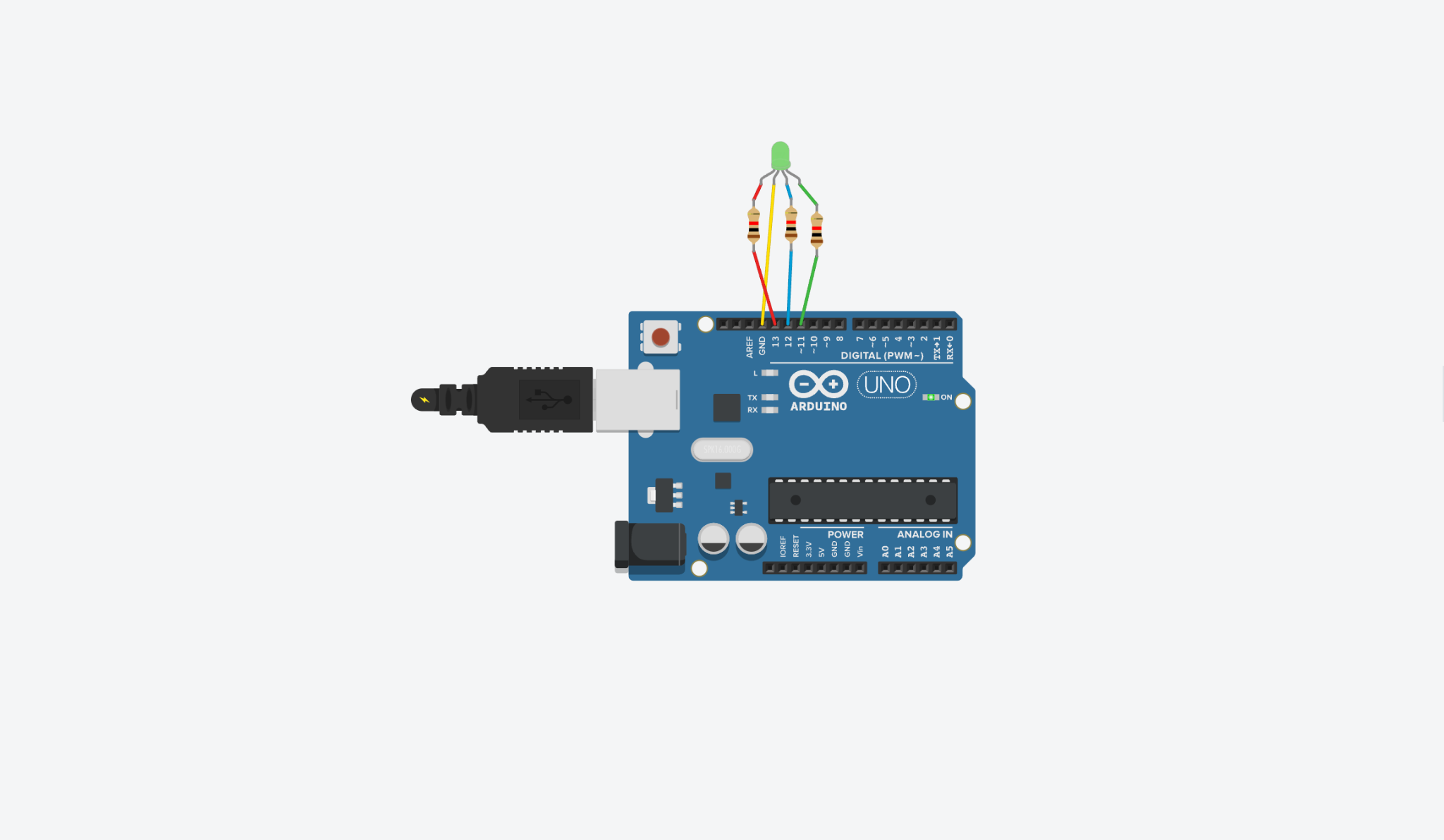
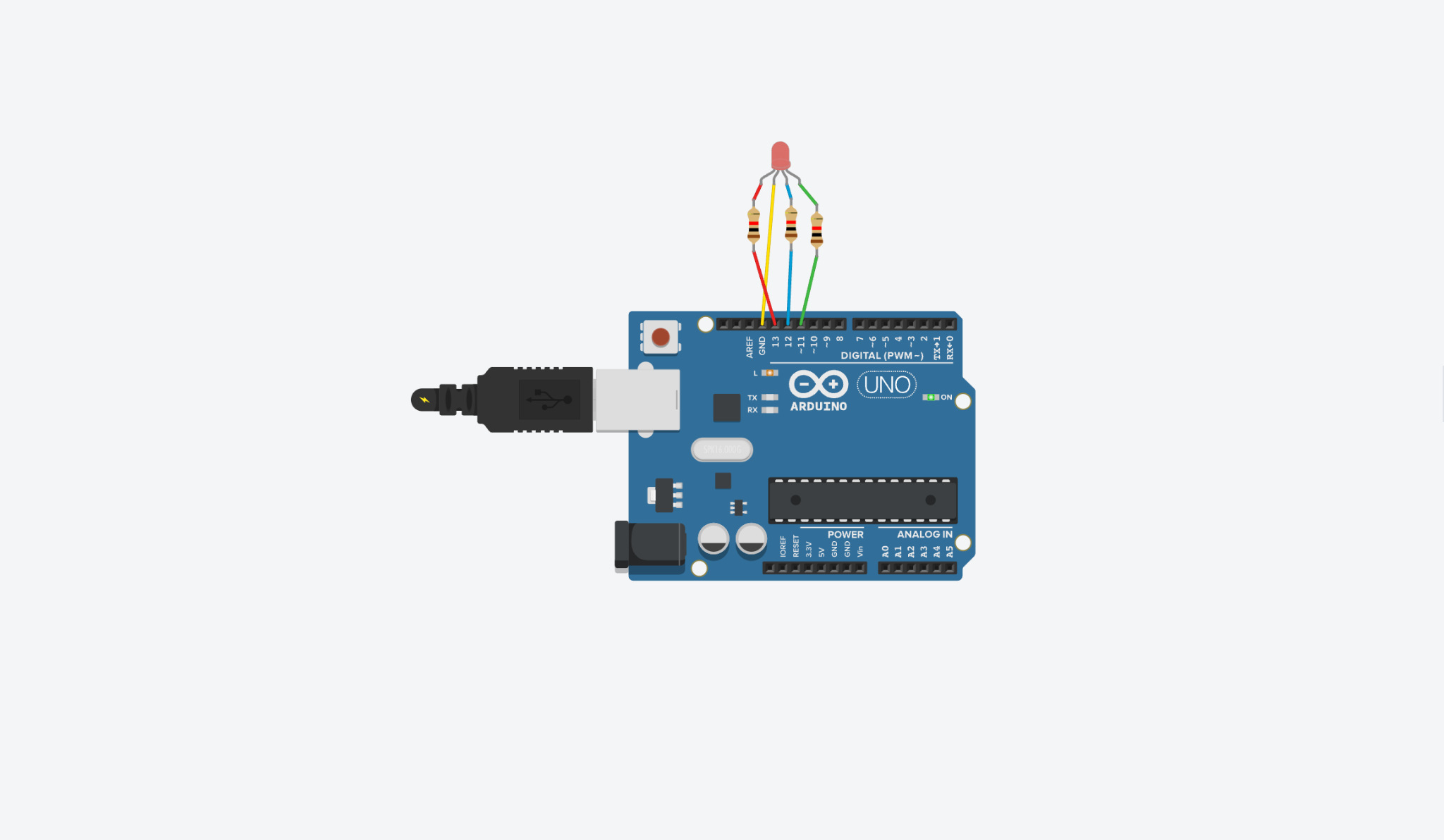
delay(1000); // Wait for 1 second

}

**Results :**

The RGB LED will cycle through colors such as Red, Green, Blue, Yellow, Purple and Cyan. The intensity of each color channel is controlled by the analogWrite() function using PWM,

and the LED will display each color for 1 second.

****

**Conclusion :**

The RGB Light project effectively showcases the use of PWM to control the color of an RGB LED. By adjusting the intensity of red, green and blue channels, various colors are generated. This experiment strengthens the understanding of LED control and prepares learners for more advanced IoT-based lighting and display applications.