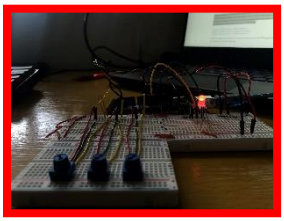
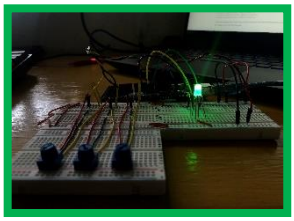
Extra Challenge: Find Your Favorite Color Using Potentiometer!

Designed by Muhammad Arsani

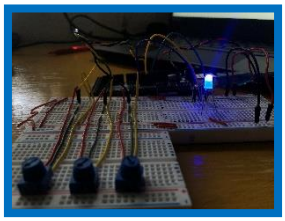
Objective: In this challenge, we are going to apply the concept of voltage divider by using potentiometer in our circuit. We will test our understandings in Pulse-Width Modulation (PWM) signals and how we use Arduino to adjust the duty cycle of the signals. We also want to preserve the vibrancy of the RGB LED using our knowledge of resistors in parallel and in series, then determine which one works best in this challenge. Finally, use your creativity to arrange the components on the breadboard so your hands have enough space and comfort when changing the potentiometer value.
The goal is to have three potentiometers controlling the intensity of the Red, Green, and Blue LED of the RGB LED to produce the 8 colors shown in the following pictures. Finally, you can adjust the intensity as much as you want until you found your favorite color!



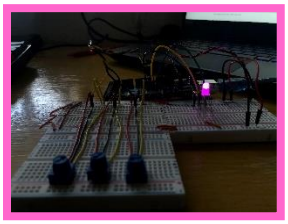
RED



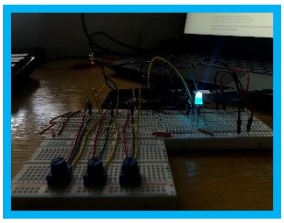
GREEN



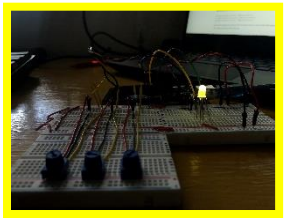
BLUE



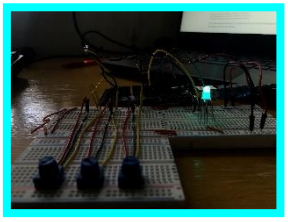
PINK



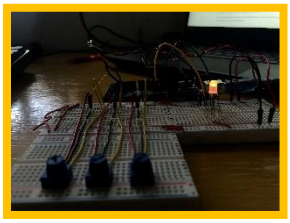
LIGHT BLUE



YELLOW



TURQUOISE



ORANGE

Components:
1.) 1 Arduino MEGA
2.) 1 RGB LED
3.) 3 Potentiometer
4.) 8 Jumper wires (or more, depends on the student)
5.) 3 Resistors (330 ohms)

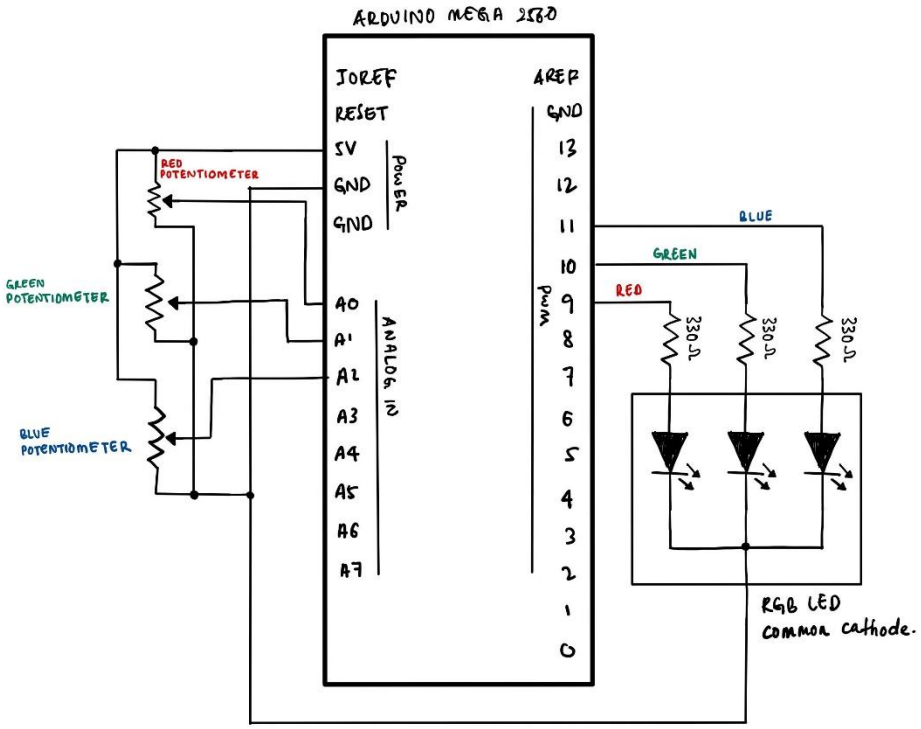
Coding/Programming: Analyze the code below and fill in the empty spaces. Make sure to put on your thinking hat and recall the functions and syntax you learned in previous lab sessions! Tip: Read the comments as they will help you connect the dots!

```
//Assigning the Pin for each color of your RGB
int RGB_RedPin = _;
int RGB_GreenPin = _;
int RGB_BluePin = _;

void setup()
{
 Serial.begin(9600); // initialize serial communications at 9600 bps
 //Tell the Arduino that our RGB LED is the output
 pinMode(RGB_RedPin, ______);
 pinMode(RGB_GreenPin, ______);
 pinMode(RGB_BluePin, ______);
}

void loop()
{
 int Potentiometer_RedPin = _____;
 //tell the arduino to read the voltage across the wiper terminal of our potentiometer at the Analog pin as stated
 int PWM_RedPin = _____;
 //We divide by 4 to convert the values to the 0-255 range
 //since we are using arduino mega which has PWM function of 0-255 range
 //isntead of the 10-bit ADC resolution 0-1023
 int Potentiometer_GreenPin = _____;
 int PWM_GreenPin = _____;
 int Potentiometer_BluePin = _____;
 int PWM_BluePin = _____;

 /*
 //Use this if your RGB LED is COMMON CATHODE
 analogWrite(RGB_RedPin, PWM_RedPin);
 analogWrite(RGB_GreenPin, PWM_GreenPin);
 analogWrite(RGB_BluePin, PWM_BluePin);
 */

 //Use this code for RGB LED is COMMON ANODE
 analogWrite(RGB_RedPin, ___-PWM_RedPin);
 analogWrite(RGB_GreenPin, ___-PWM_GreenPin);
 analogWrite(RGB_BluePin, ___-PWM_BluePin);

 delay(2);
}
```

Instructor's note: *Solution is as follows. Take note of the two blocks of codes in the void loop, one commented with "Use this if your RGB LED is COMMON CATHODE". Students need to identify what kind of RGB LED they are using and determine which codes to use and which codes need to be commented.*

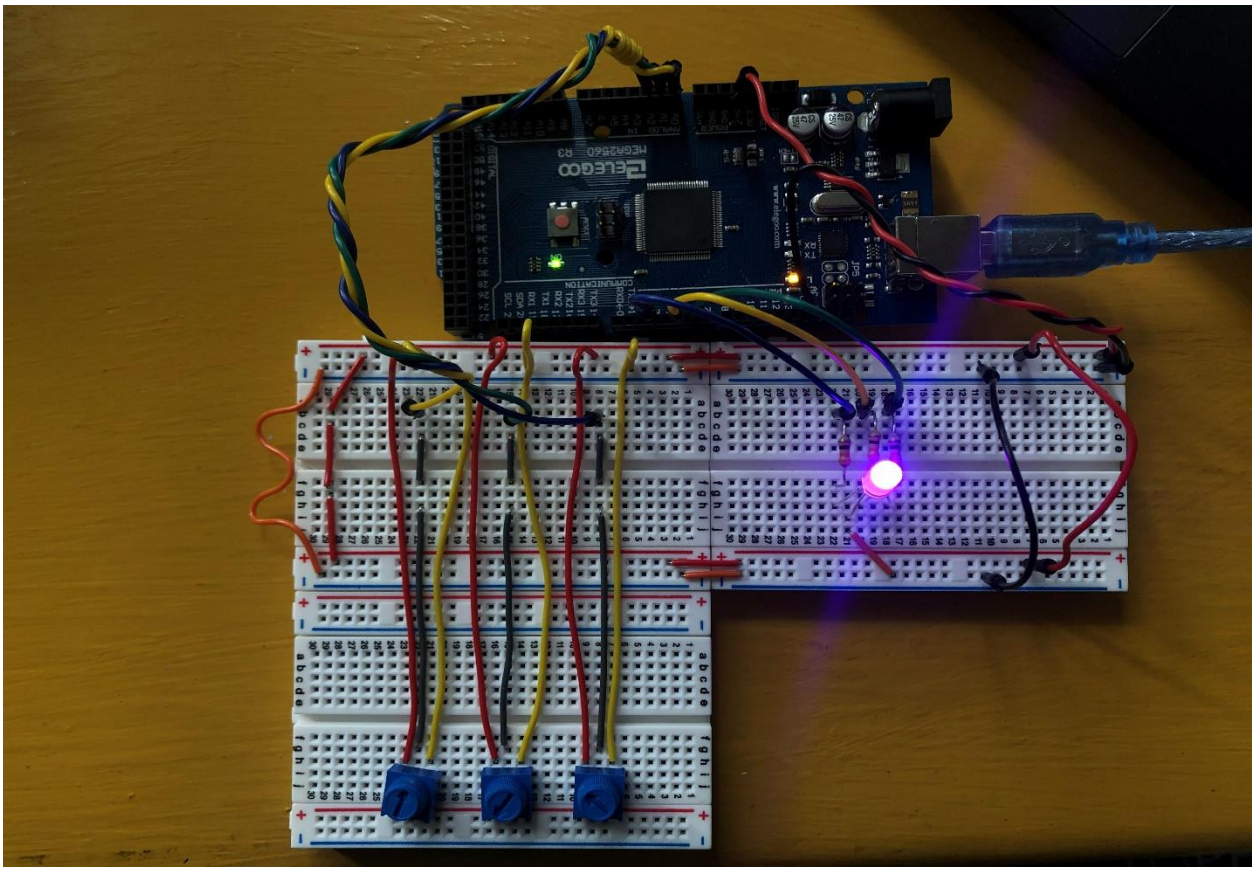
```
//Assigning the Pin for each color of your RGB
int RGB_RedPin = 3;
int RGB_GreenPin = 5;
int RGB_BluePin = 6;

void setup()
{
 Serial.begin(9600); // initialize serial communications at 9600 bps
 //Tell the Arduino that our RGB LED is the output
 pinMode(RGB_RedPin, OUTPUT);
 pinMode(RGB_GreenPin, OUTPUT);
 pinMode(RGB_BluePin, OUTPUT);
}

void loop()
{
 int Potentiometer_RedPin = analogRead(A0);
 //tell the arduino to read the voltage across the wiper terminal of our potentiometer at the Analog pin as stated
 int PWM_RedPin = Potentiometer_RedPin/4;
 //We divide by 4 to convert the values to the 0-255 range
 //since we are using arduino mega which has PWM function of 0-255 range
 //isntead of the 10-bit ADC resolution 0-1023
 int Potentiometer_GreenPin = analogRead(A1);
 int PWM_GreenPin = Potentiometer_GreenPin/4;
 int Potentiometer_BluePin = analogRead(A2);
 int PWM_BluePin = Potentiometer_BluePin/4;

 /*
 //Use this if your RGB LED is COMMON CATHODE
 analogWrite(RGB_RedPin, PWM_RedPin);
 analogWrite(RGB_GreenPin, PWM_GreenPin);
 analogWrite(RGB_BluePin, PWM_BluePin);
 */

 //Use this code for RGB LED is COMMON ANODE
 analogWrite(RGB_RedPin, 255-PWM_RedPin);
 analogWrite(RGB_GreenPin, 255-PWM_GreenPin);
 analogWrite(RGB_BluePin, 255-PWM_BluePin);

 delay(2);
}
```

Circuit Diagram:



Instructor's note: *Before building the circuit, students are encouraged to practice drawing the schematic diagram of their circuit using Fritzing or Eagle if they have learned how to use this software.*

Completed breadboard prototype example:



Congratulations, you've finished! Enjoy your customizable RGB LED!

Extra challenges/questions:

- Observe what happens if you don't divide your PWM value by 4.
- Try to incorporate button in the circuit so that the LED won't turn on even if you change the potentiometer value, unless you push the button.
- Use this project as a decoration in your room! Start with designing the PCB of the circuit with EAGLE, have it delivered to your home, and then solder all the components needed! You can also make a casing for it so that it looks like you bought it from a gift shop.