

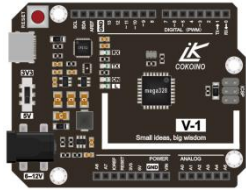
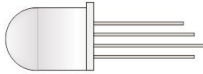
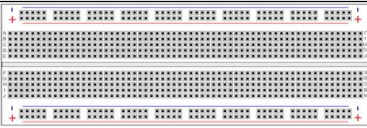

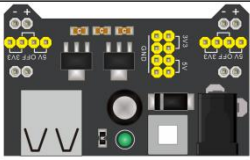
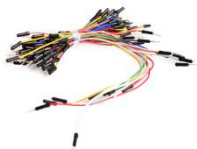


## 6.Cool RGB LED Light

### ABOUT THIS PROJECT:

#### You will learn:

- ◆ How does the RGB LED work
- ◆ Use the I/O port of V-1 board control the RGB LED light
- ◆ Use the PWM port of V-1 board control the RGB LED light

### Things used in this project:

Hardware components	Picture	Quantity
V-1 board		1 PCS
F5 common anode RGB LED light		1 PCS
Breadboard		1 PCS
9V Battery Snap Connector (you need to buy 9V battery yourself)		1 PCS
Breadboard power module		1 PCS
Male to Male DuPont Cable		5 PCS
Type C USB Cable		1 PCS
220R Resistance		3 PCS

---

---

## 2. About RGB LED light

The RGB LED can emit different colors by mixing the 3 basic colors red, green and blue. So it actually consists of 3 separate LEDs red, green and blue packed in a single case. That's why it has 4 leads, one lead for each of the 3 colors and one common cathode or anode depending of the RGB LED type.

An RGB LED is shown in the following figure:



### 2.1 How to create different colors?

With an RGB LED you can, of course, produce red, green, and blue light, and by configuring the intensity of each LED, you can produce other colors as well.

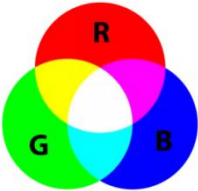
For example, to produce purely blue light, you'd set the blue LED to the highest intensity and the green and red LEDs to the lowest intensity. For a white light, you'd set all three LEDs to the highest intensity.

### 2.2 Mixing colors

To produce other colors, you can combine the three colors in different intensities. To adjust the intensity of each LED you can use a PWM signal.

Because the LEDs are very close to each other, our eyes see the result of the combination of colors, rather than the three colors individually.

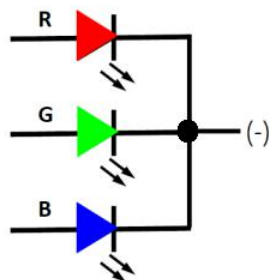
To have an idea on how to combine the colors, take a look at the following chart. This is the simplest color mixing chart, but gives you an idea how it works and how to produce different colors.



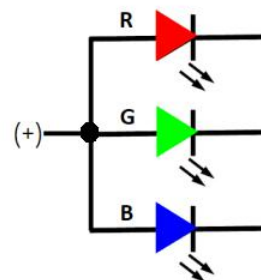
### 2.3 Common Anode and Common Cathode RGB LEDs

There are two kinds of RGB LEDs: common anode LED and common cathode LED. The figure below illustrates a common anode and a common cathode LED.

Common Cathode (-)



Common Anode (+)



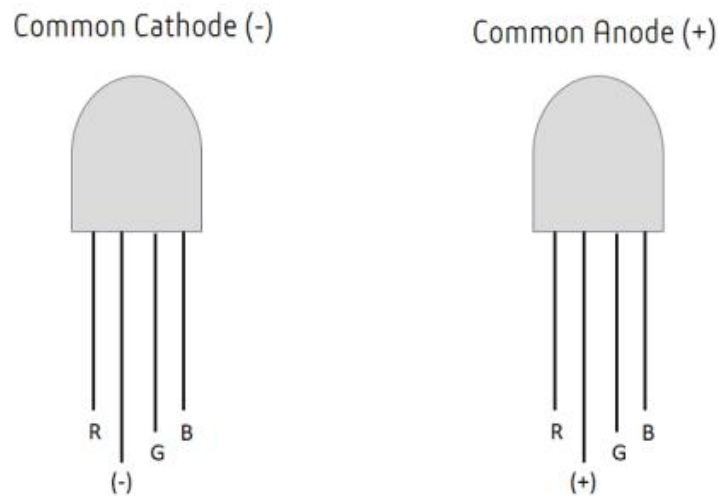
---

In a common cathode RGB LED, all three LEDs share a negative connection (cathode). In a common anode RGB LED, the three LEDs share a positive connection (anode).

This results in an LED that has 4 pins, one for each LED, and one common cathode or one common anode.

### RGB LED Pins

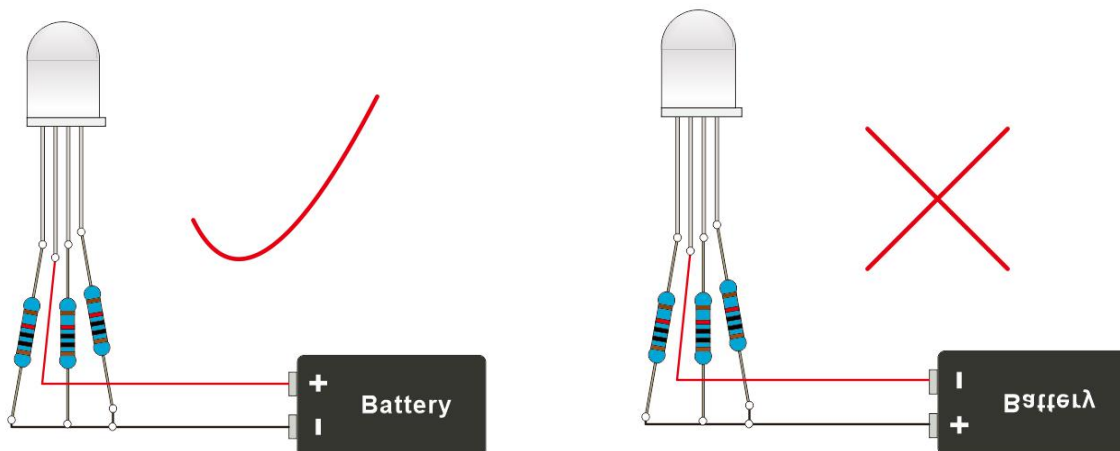
RGB LEDs have four leads—one for each LED and another for the common anode or cathode. You can identify each lead by its length, as shown in the following figure.



**In this tutorial I will be using a common anode one.**

When a single color lamp is lit, there is a voltage drop of about 1-3V at both ends, and the current is generally about 5-15 mA. When a single color lamp is lit, there is a voltage drop of about 1-3V at both ends, and the current is generally about 5-15 mA. The forward voltage can illuminate the LED, and the reverse voltage will not cause the LED to operate. If the reverse voltage connected exceeds its reverse withstand voltage, the LED may be permanently damaged.

### Instructions for use:



---

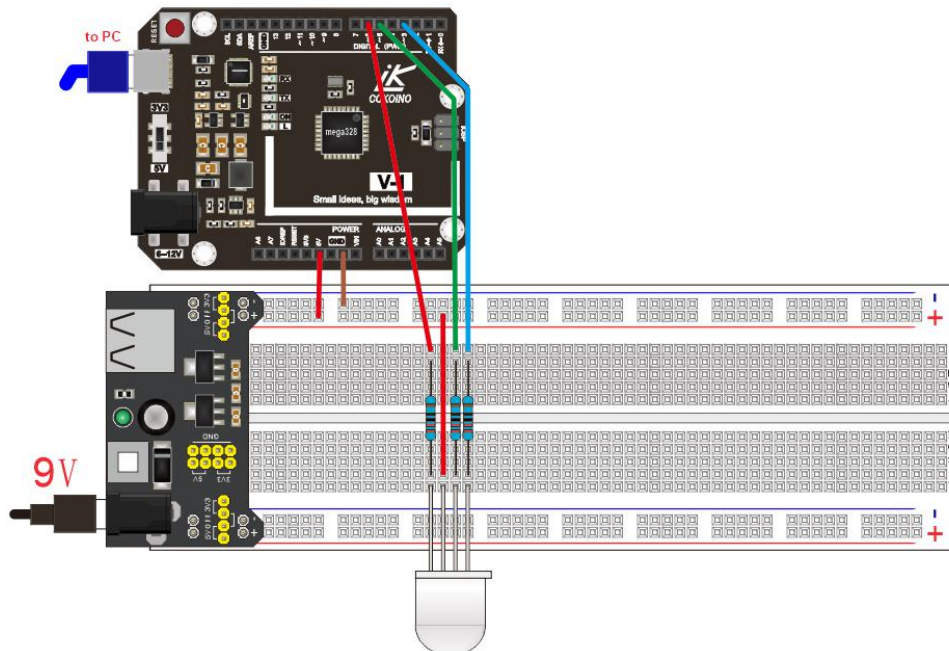
### 3. Use I/O Pins of V-1 board to Control RGB LED light

In this lab, you can learn to control the RGB LEDs to emit different colors of light through V-1 board. The main statement of this program is: `digitalWrite(pin, LOW/HIGH);`

#### 3.1 Code

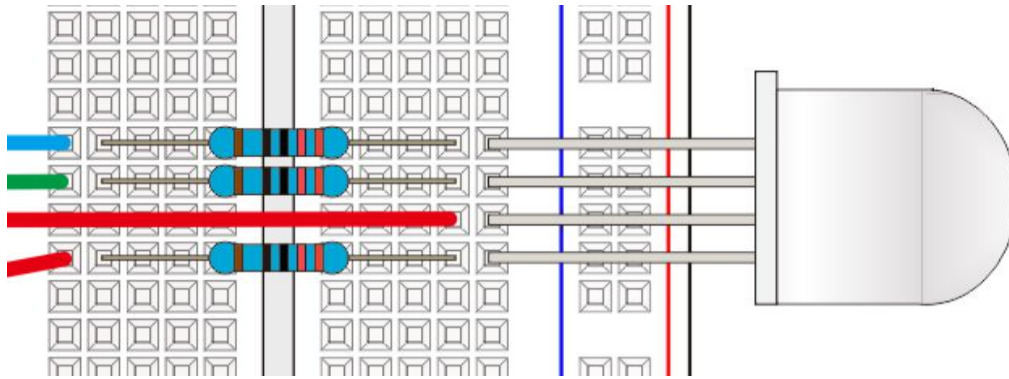
```
#define LED_B 3
#define LED_G 5
#define LED_R 6
void setup() {
  pinMode(LED_R,OUTPUT);
  pinMode(LED_G,OUTPUT);
  pinMode(LED_B,OUTPUT);
}
void loop() {
  digitalWrite(LED_R,LOW);    // red
  delay(1000);               // wait for a second
  digitalWrite(LED_R,HIGH);   // turn the LED off
  digitalWrite(LED_G,LOW);    //green
  delay(1000);
  digitalWrite(LED_G,HIGH);
  digitalWrite(LED_B,LOW);
  delay(1000);
  digitalWrite(LED_R,LOW);    //white
  digitalWrite(LED_G,LOW);
  delay(1000);
  digitalWrite(LED_R,HIGH);   //off
  digitalWrite(LED_G,HIGH);
  digitalWrite(LED_B,HIGH);
}
```

#### 3.2 Connection Diagram



Note: The longest pin of the RGB led lamp is positive, and the other three pins are the cathodes of red, green and blue lights.

## Detail enlargement



## 3.3 Step

3.3.1 Connect the computer and V-1 board with a USB cable and copy the above sample code to the Arduino IDE as shown below:

```
RGB_LED | Arduino 1.8.5
File Edit Sketch Tools Help

RGB_LED

#define LED_B 3
#define LED_G 5
#define LED_R 6

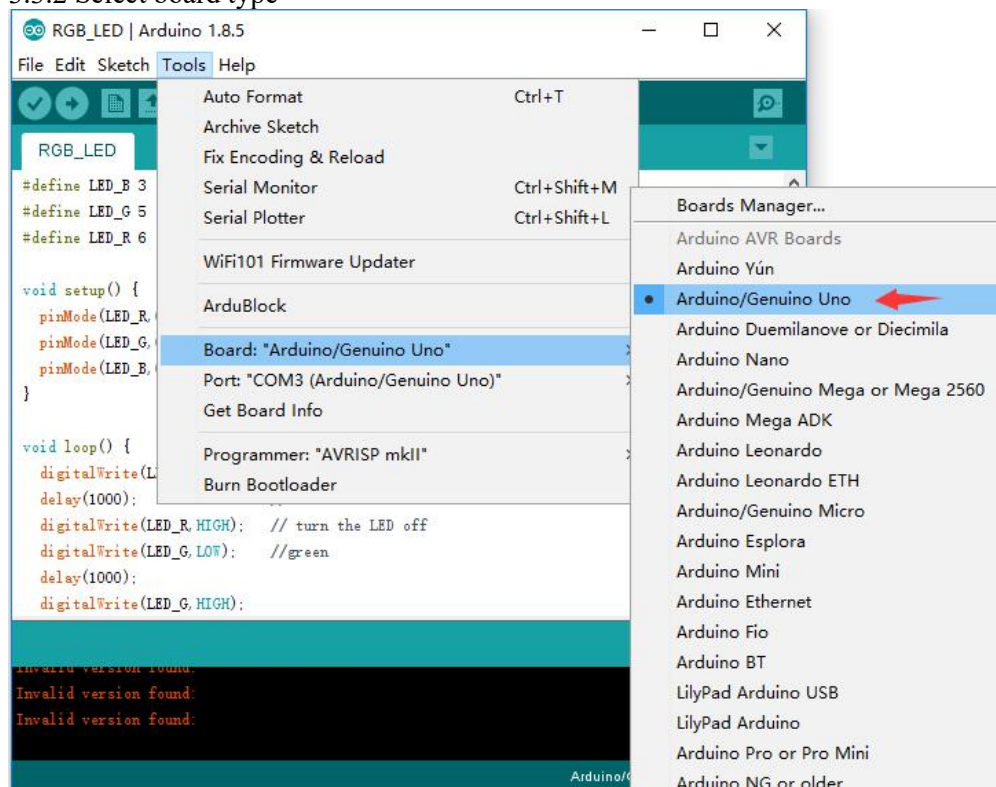
void setup() {
  pinMode(LED_R, OUTPUT);
  pinMode(LED_G, OUTPUT);
  pinMode(LED_B, OUTPUT);
}

void loop() {
  digitalWrite(LED_R, LOW);    // red
  delay(1000);                 // wait for a second
  digitalWrite(LED_R, HIGH);   // turn the LED off
  digitalWrite(LED_G, LOW);    // green
  delay(1000);
  digitalWrite(LED_G, HIGH);

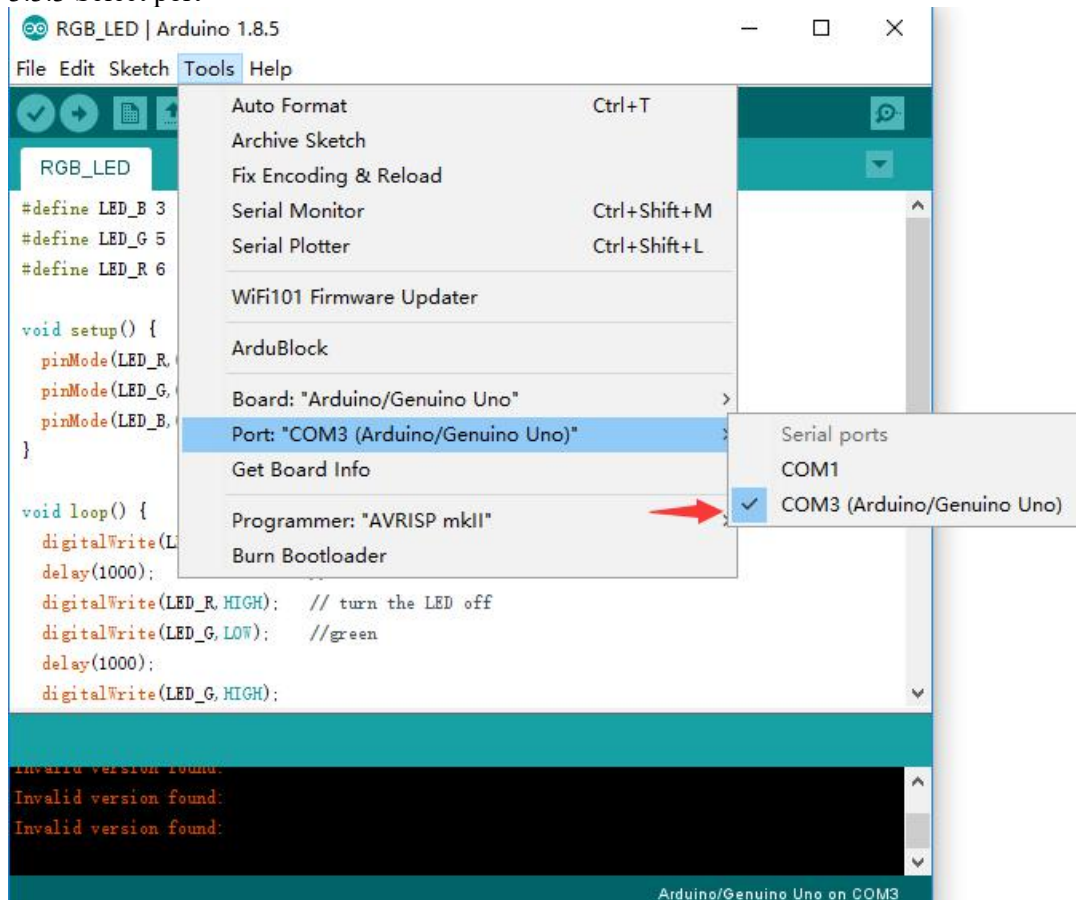
  Invalid version found:
  Invalid version found:
  Invalid version found:

Arduino/Genuino Uno on COM3
```

### 3.3.2 Select board type

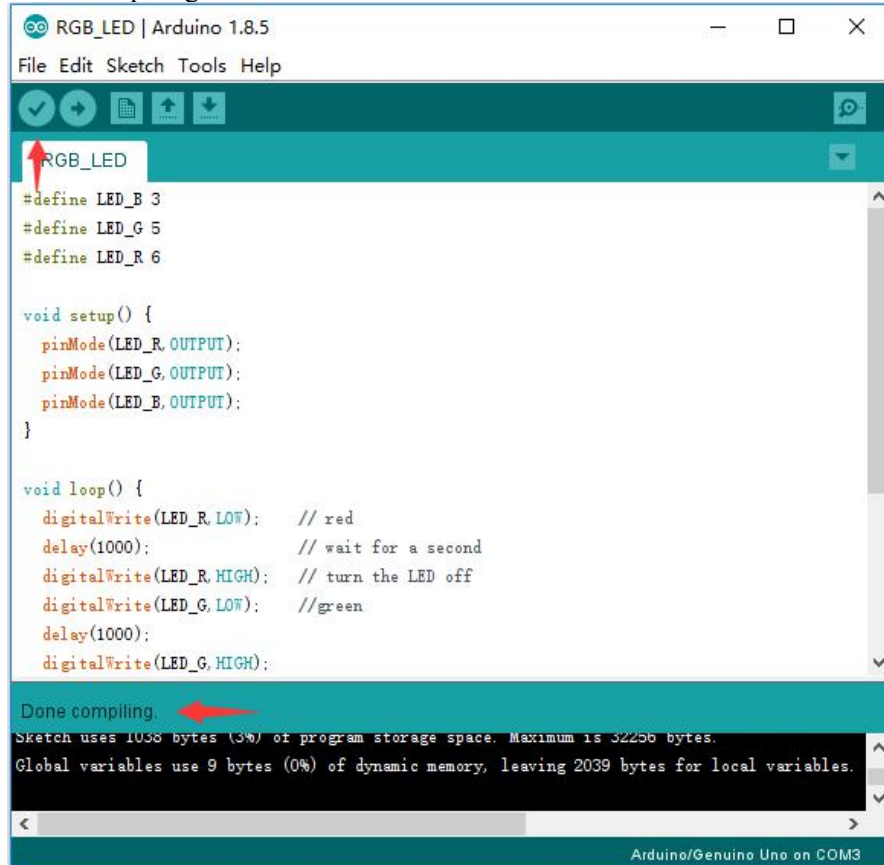


### 3.3.3 Select port





### 3.3.4 Compiling



RGB\_LED | Arduino 1.8.5

File Edit Sketch Tools Help

RGB\_LED

```
#define LED_B 3
#define LED_G 5
#define LED_R 6

void setup() {
  pinMode(LED_R, OUTPUT);
  pinMode(LED_G, OUTPUT);
  pinMode(LED_B, OUTPUT);
}

void loop() {
  digitalWrite(LED_R, LOW); // red
  delay(1000); // wait for a second
  digitalWrite(LED_R, HIGH); // turn the LED off
  digitalWrite(LED_G, LOW); //green
  delay(1000);
  digitalWrite(LED_G, HIGH);
```

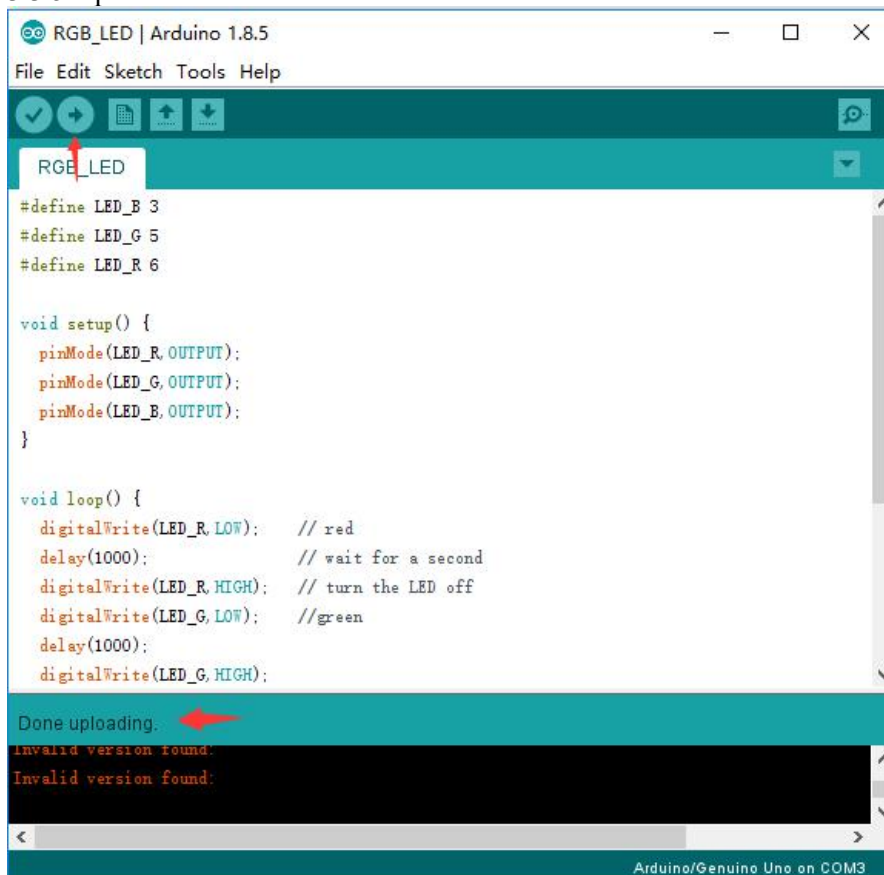
Done compiling.

Sketch uses 1038 bytes (3%) of program storage space. Maximum is 32256 bytes.

Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables.

Arduino/Genuino Uno on COM3

### 3.3.5 Upload the sketch



RGB\_LED | Arduino 1.8.5

File Edit Sketch Tools Help

RGB\_LED

```
#define LED_B 3
#define LED_G 5
#define LED_R 6

void setup() {
  pinMode(LED_R, OUTPUT);
  pinMode(LED_G, OUTPUT);
  pinMode(LED_B, OUTPUT);
}

void loop() {
  digitalWrite(LED_R, LOW); // red
  delay(1000); // wait for a second
  digitalWrite(LED_R, HIGH); // turn the LED off
  digitalWrite(LED_G, LOW); //green
  delay(1000);
  digitalWrite(LED_G, HIGH);
```

Done uploading.

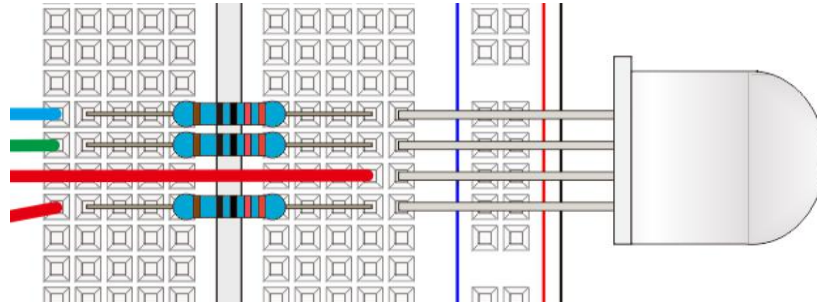
Invalid version found:

Invalid version found:

Arduino/Genuino Uno on COM3

### 3.3.6 Result

Unplug the USB cable from the V-1 board, connect the power module to the external power supply, and then turn on the switch of the power module on the breadboard. The RGB LED on the breadboard will emit red, green, blue, and white light, as shown below:



## 4. Use the PWM port of V-1 board control the RGB LED light

Pulse Width Modulation (PWM) is a common method used to apply a proportional control signal to an external device using a digital output pin. For example, servo motors use the pulse width of an incoming PWM signal to determine their rotation angle. LCD displays adjust their brightness based on a PWM signal's average value.

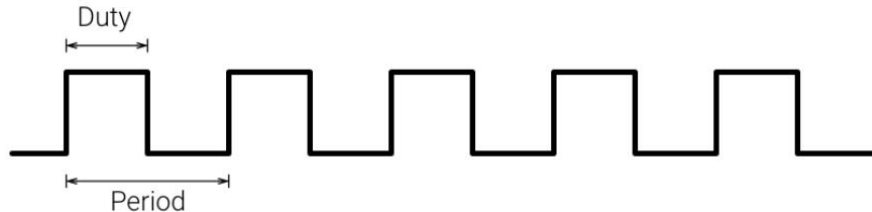
PWM is a digital (i.e. square wave) signal that oscillates according to a given frequency and duty cycle.

The frequency (expressed in Hz) describes how often the output pulse repeats.

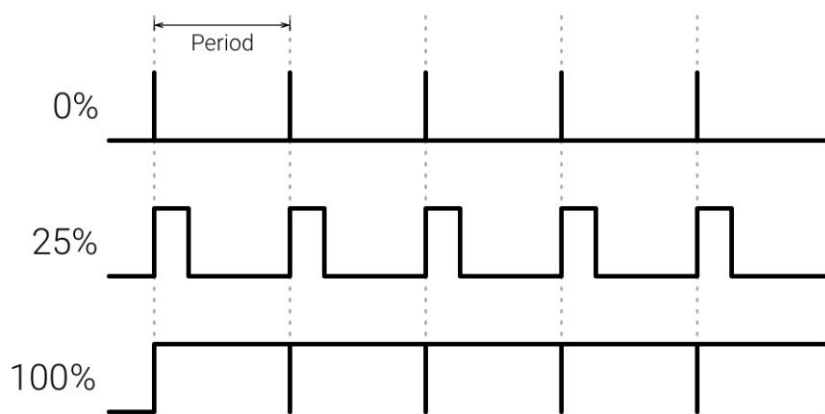
The period is the time each cycle takes and is the inverse of frequency.

The duty cycle (expressed as a percentage) describes the width of the pulse within that frequency window.

For example, a PWM signal set to 50% duty is active for half of each cycle:



You can adjust the duty cycle to increase or decrease the average "on" time of the signal. The following diagram shows pulse trains at 0%, 25%, and 100% duty:



Note: Most PWM hardware has to toggle at least once per cycle, so even duty values of 0% and 100% will have a small transition at the beginning of each cycle.

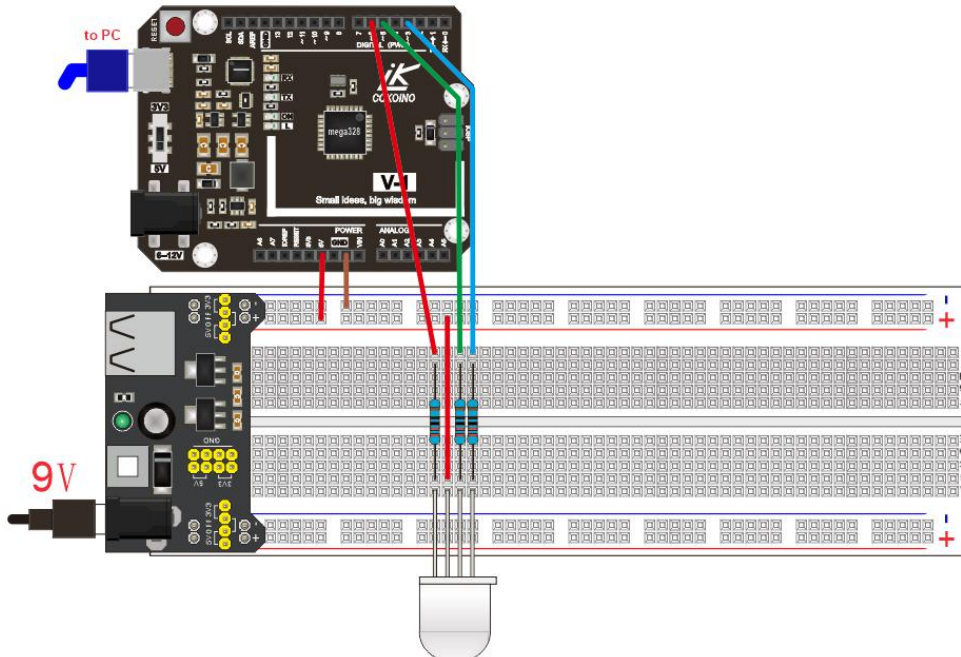
V-1 board only has 3, 5, 6, 9, 10, 11 pins to output PWM signal, and the default frequency of the signal is 1K Hz. When the PWM signal outputs high level, the LED light is on, and when the output is low, the LED is off. Controlling the high and low duty cycles controls the brightness of the LEDs. The main statement of this program is: `analogWrite(pin, PWM_data);` PWM\_data takes 0-255.



## 4.1、Code

```
#define LED_B 3
#define LED_G 5
#define LED_R 6
int PWM_data;
void setup() {
  pinMode(LED_R,OUTPUT);
  pinMode(LED_G,OUTPUT);
  pinMode(LED_B,OUTPUT);
}
void loop() {
  for(PWM_data=255;PWM_data>=0;PWM_data--){
    analogWrite(LED_R,PWM_data);  // PWM
    analogWrite(LED_G,0);
    analogWrite(LED_B,0);
    delay(10);
  }
  for(PWM_data=255;PWM_data>=0;PWM_data--){
    analogWrite(LED_R,0);
    analogWrite(LED_G,PWM_data);
    analogWrite(LED_B,0);
    delay(10);
  }
  for(PWM_data=255;PWM_data>=0;PWM_data--){
    analogWrite(LED_R,0);
    analogWrite(LED_G,0);
    analogWrite(LED_B,PWM_data);
    delay(10);
  }
}
```

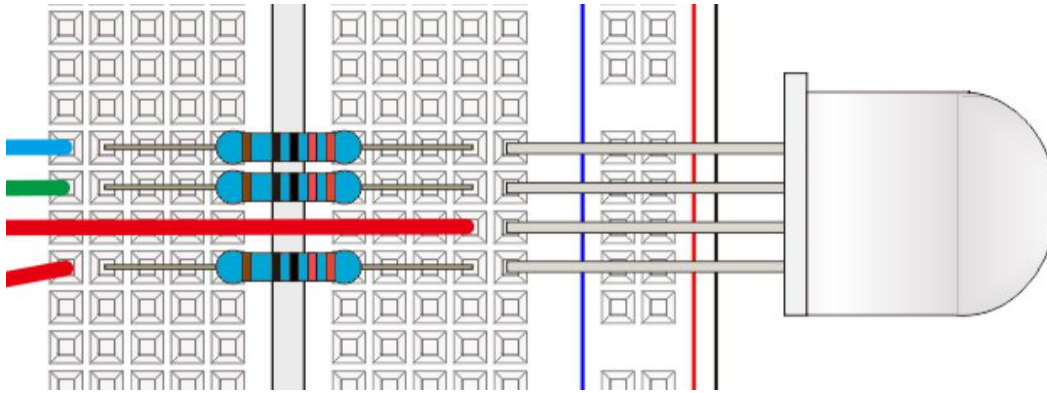
## 4.2 Connection Diagram



Note: The long pin of the led lamp is positive and the short pin is negative.

---

## Detail enlargement



### 4.3 Experiment

Unplug the USB cable from the V-1 board, connect the power module to the external power supply, and then turn on the switch of the power module on the breadboard. The RGB LED lights on the breadboard will slowly emit different colors of light, as shown below:

