

---

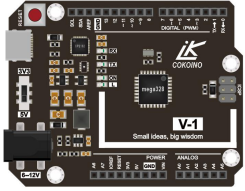
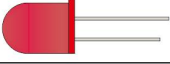

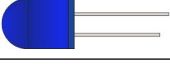

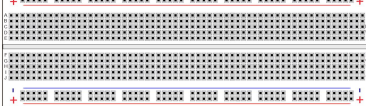

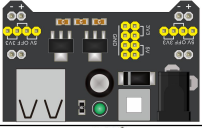
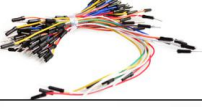


## 4. Two ways to control 4 LED lights

ABOUT THIS PROJECT:

### You will learn:

- ◆ Use the IO port of V-1 board control four LED lights
- ◆ Use the PWM port of V-1 board control four LED lights

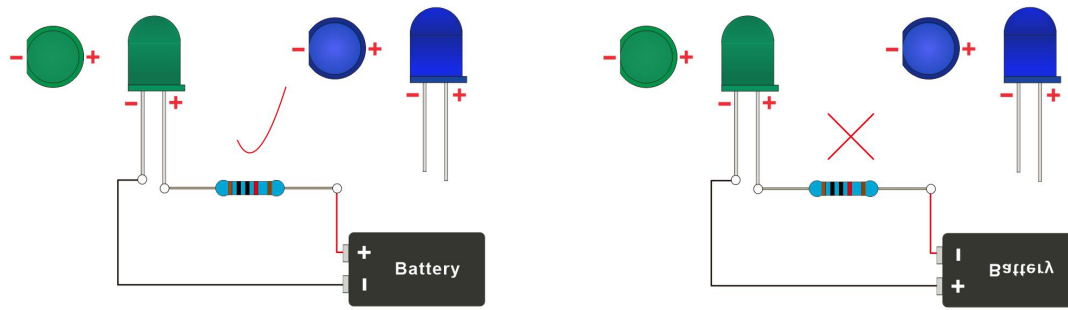
Things used in this project:

Hardware components	Picture	Quantity
V-1 control board		1 PCS
F3 Red LED Light		1 PCS
F3 Green LED Light		1 PCS
F3 Blue LED Light		1 PCS
F3 White 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		6 PCS
Type C USB Cable		1 PCS
220R Resistance		4 PCS

---

## 1. LED light introduction:

LED lights are also called light-emitting diodes. Generally, when the forward current is turned on, there will be a voltage drop of 1-3V at both ends. The current through it is generally required to be around 5-15 mA, so a resistor is often used in series with the circuit to achieve current limiting. The forward voltage can illuminate the LED lamp, the LED lamp will not work in reverse voltage, and if the reverse voltage connected exceeds its reverse withstand voltage value, the LED light may be permanently damaged.



## 2. Let' s make a simple IO control LED light

In this experiment, the high and low levels of the IO port of the V-1 board are used to control the LED on and off, to achieve the LED flow effect. The main statement of this program is :  
`digitalWrite(pin,LOW/HIGH);`

### 2.1 Code

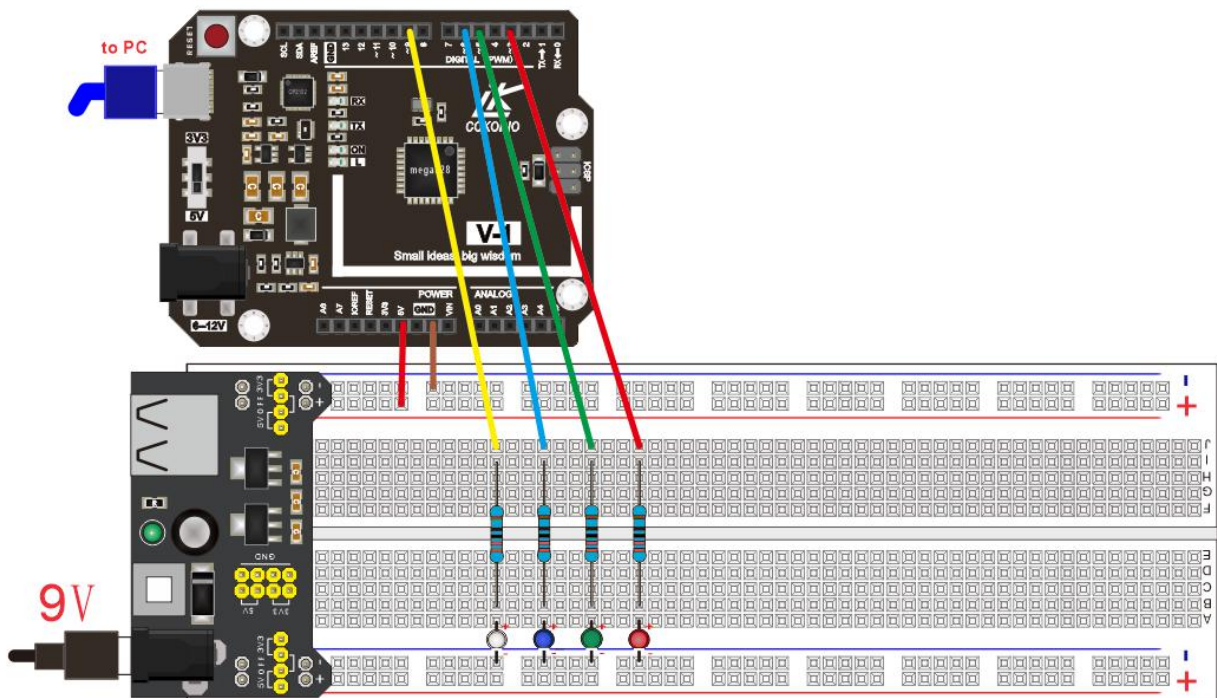
```
#define LED_R 3
#define LED_G 5
#define LED_B 6
#define LED_W 9
void setup() {
  pinMode(LED_R,OUTPUT);
  pinMode(LED_G,OUTPUT);
  pinMode(LED_B,OUTPUT);
  pinMode(LED_W,OUTPUT);
}
void loop() {
  digitalWrite(LED_R,HIGH); // turn the LED on
  delay(500);               // wait for a second
  digitalWrite(LED_G,HIGH);
  delay(500);
  digitalWrite(LED_B,HIGH);
  delay(500);
  digitalWrite(LED_W,HIGH);
  delay(500);
  digitalWrite(LED_R,LOW);  // turn the LED off
```

```

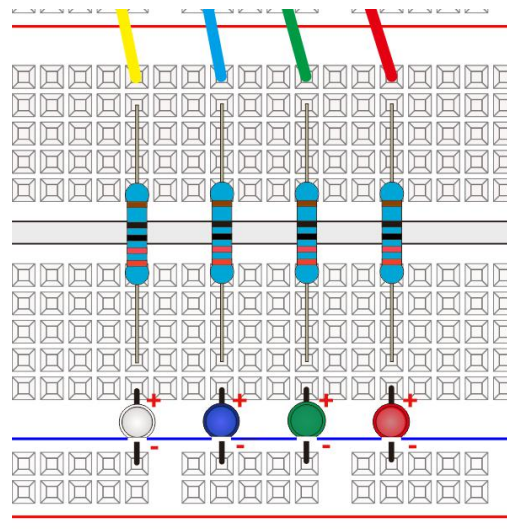
delay(500);
digitalWrite(LED_G,LOW);
delay(500);
digitalWrite(LED_B,LOW);
delay(500);
digitalWrite(LED_W,LOW);
delay(500);
}

```

## 2.2 Connection Diagram



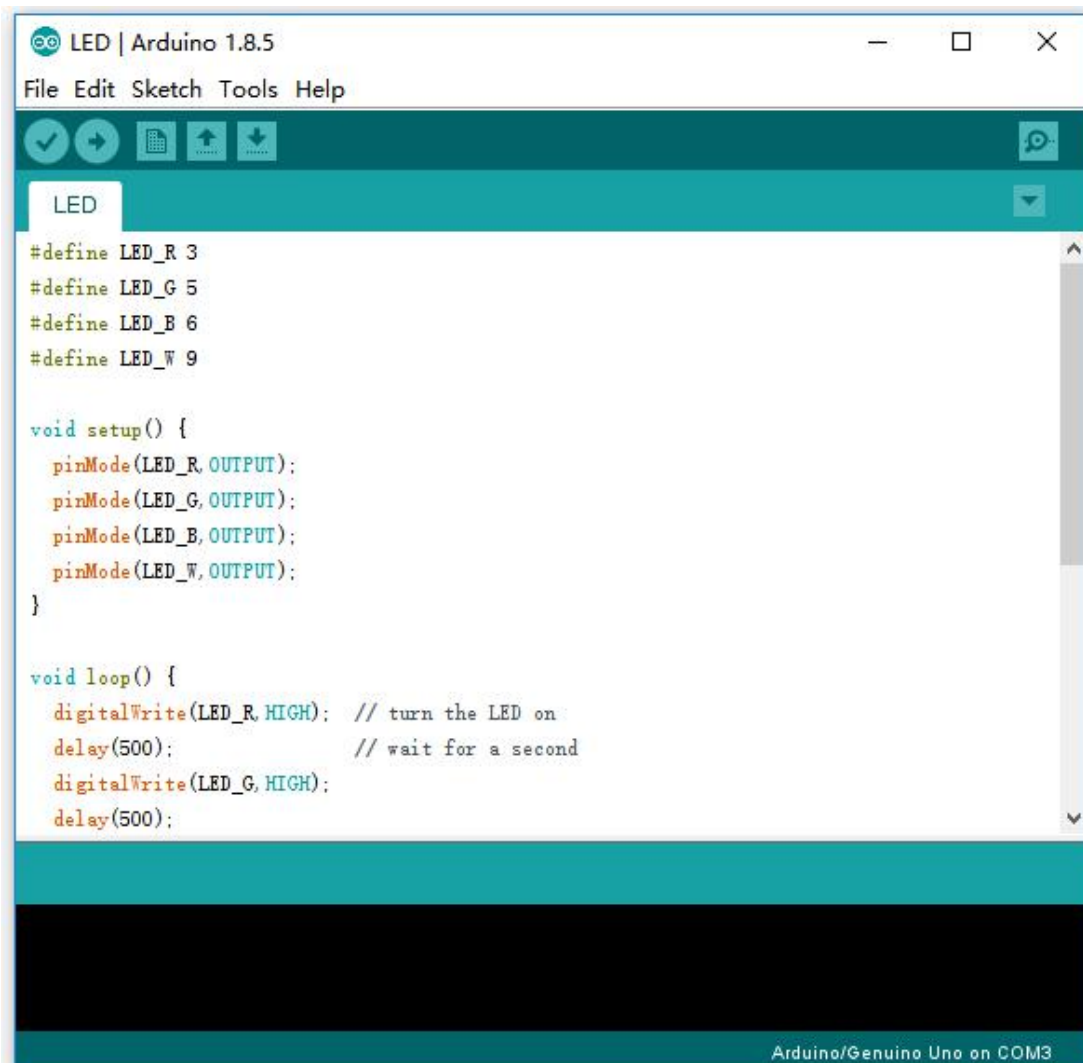
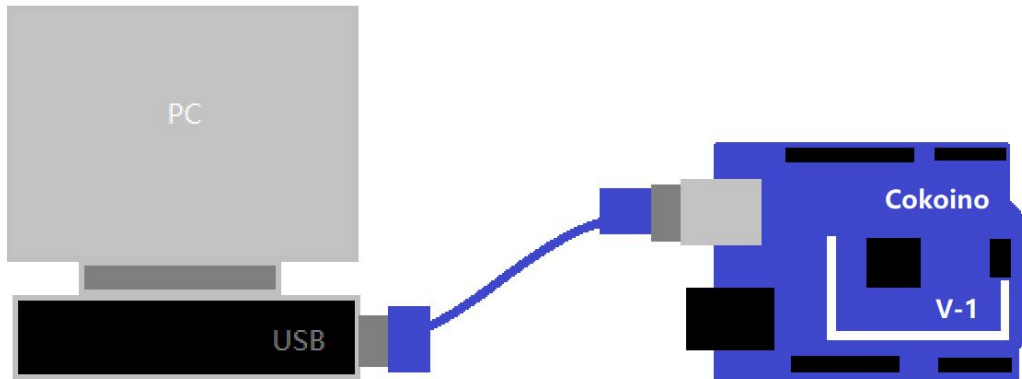
Note: The long pin of the led lamp is positive and the short pin is negative.



---

## 2.3 Steps

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

A screenshot of the Arduino IDE interface. The window title is 'LED | Arduino 1.8.5'. The menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. The toolbar shows icons for opening, saving, and running a sketch. The sketch editor displays the following code:

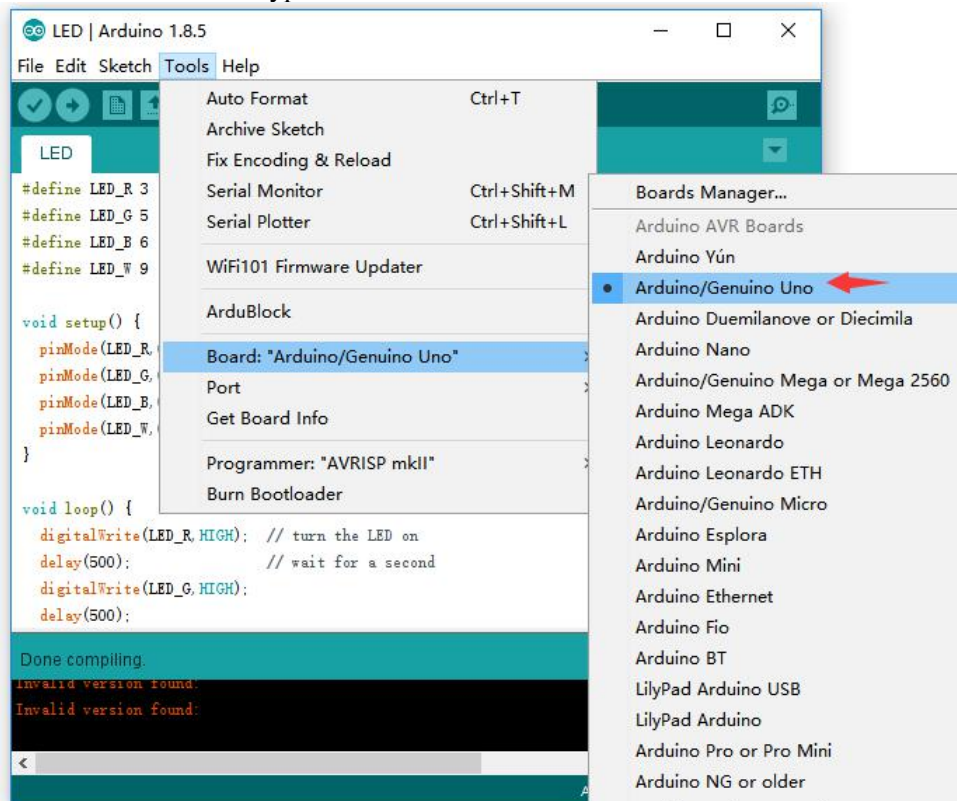
```
#define LED_R 3
#define LED_G 5
#define LED_B 6
#define LED_W 9

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

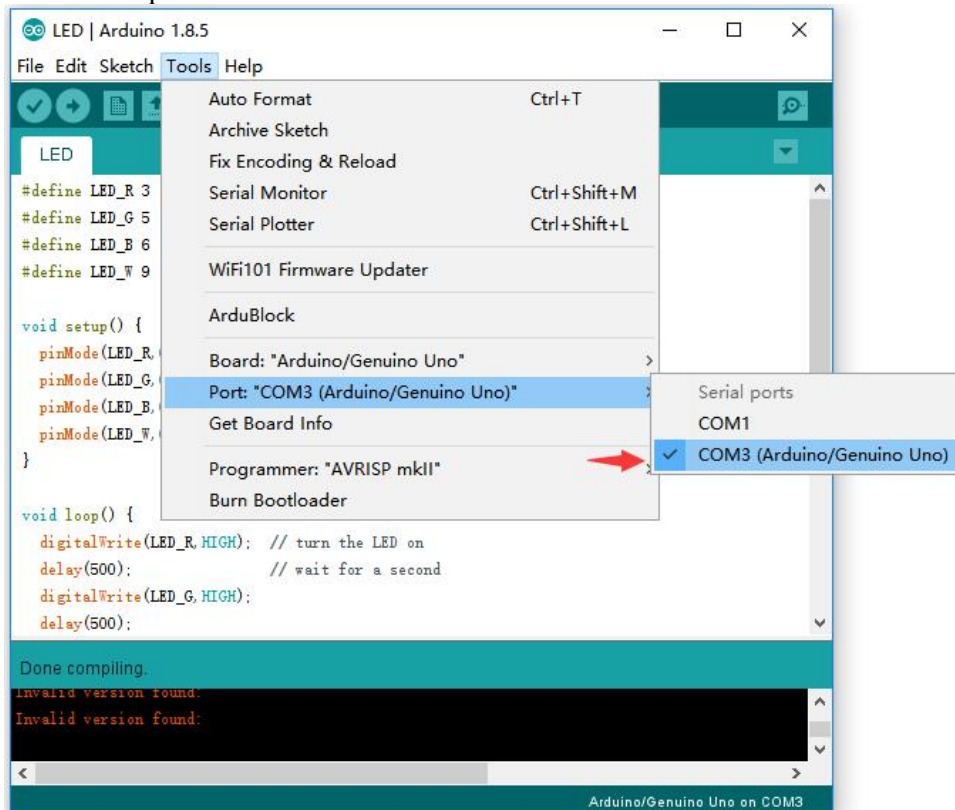
void loop() {
  digitalWrite(LED_R, HIGH); // turn the LED on
  delay(500);                // wait for a second
  digitalWrite(LED_G, HIGH);
  delay(500);
}
```

The status bar at the bottom indicates 'Arduino/Genuino Uno on COM3'.

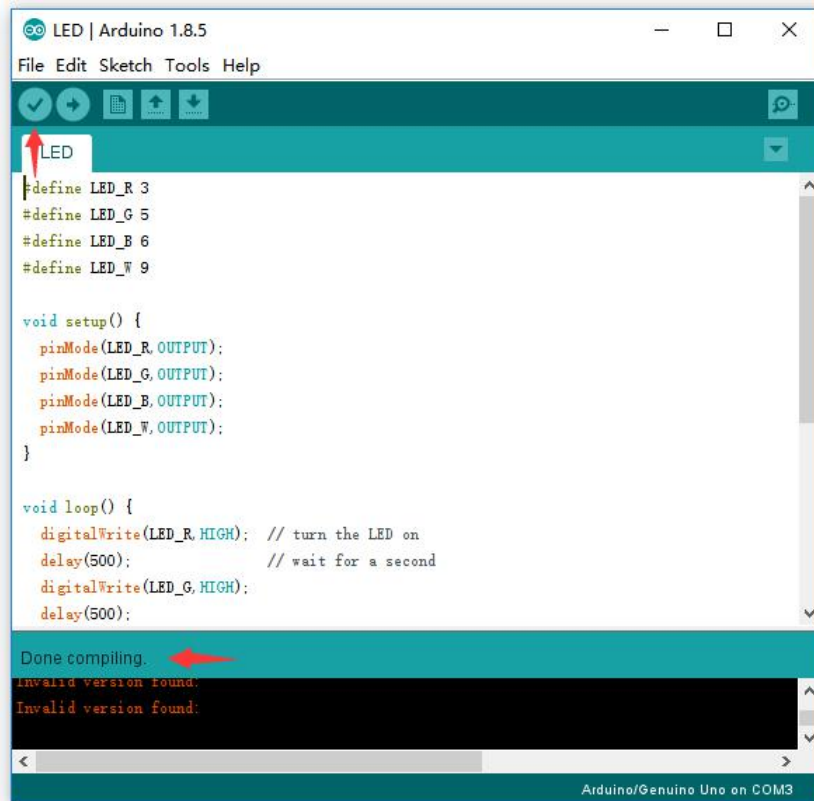
### 2.3.2 Select the board type



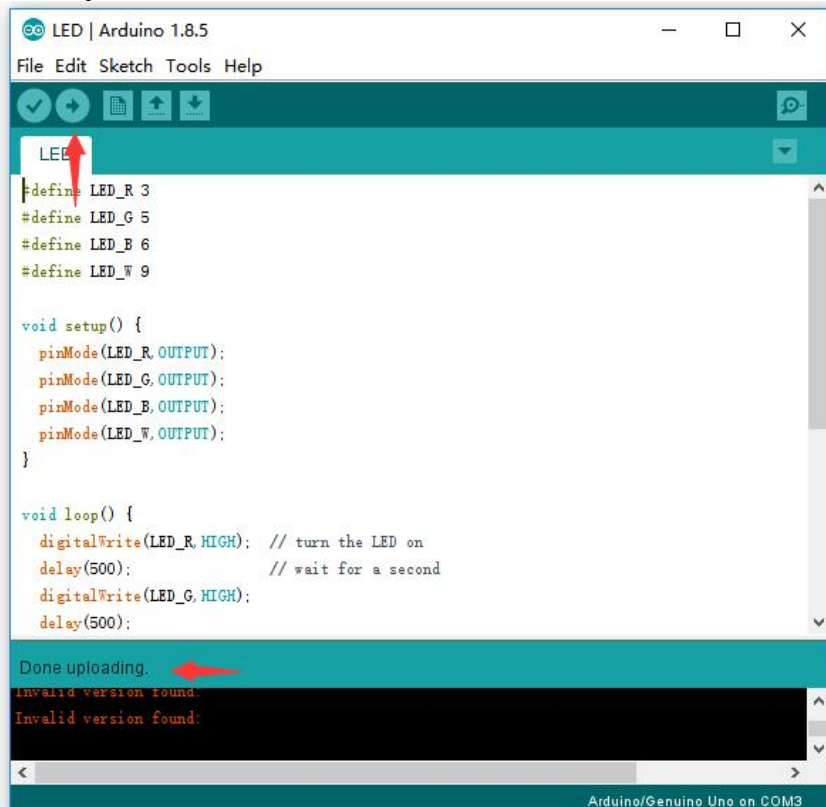
### 2.3.3 Select port



### 2.3.4 Compiling

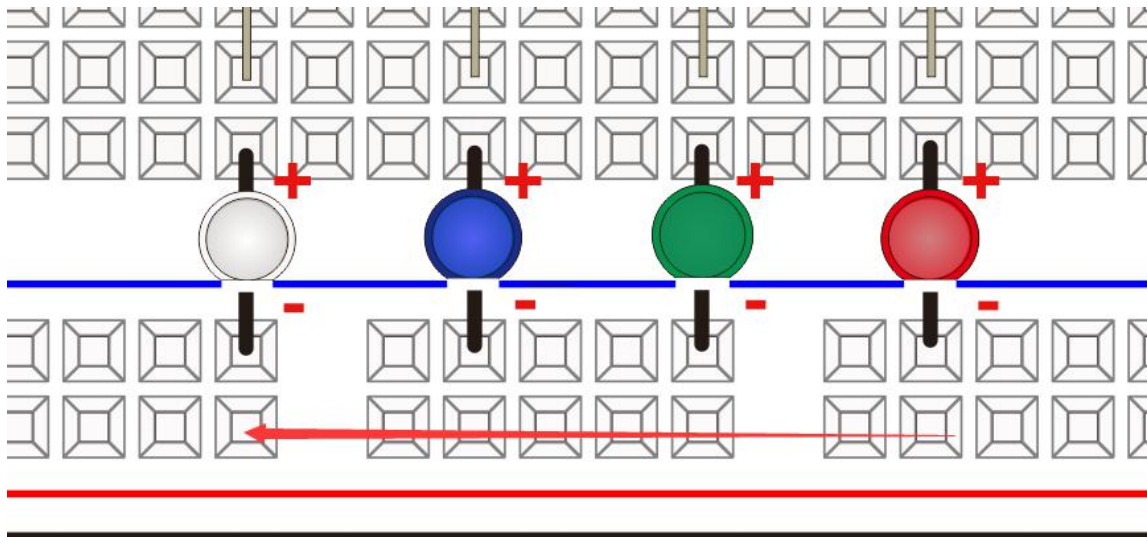


### 2.3.5 Upload the code





2.3.6 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 LED light on the breadboard will light up and go out from right to left, as shown below.:



### 3.PWM Control 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.

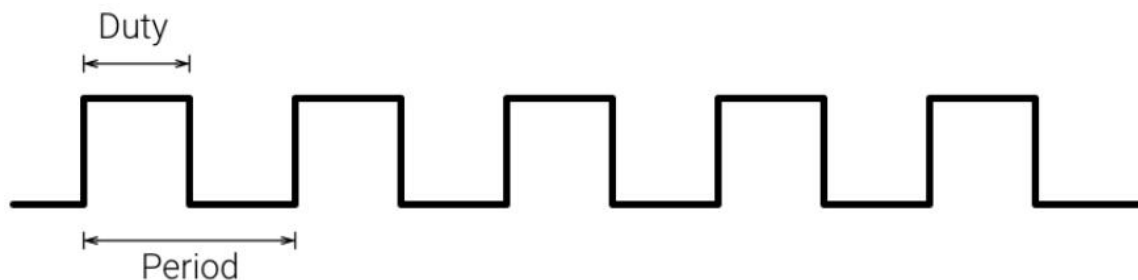
PWM is a digital (i.e. square wave) signal that oscillates according to a given frequency and duty cycle.

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

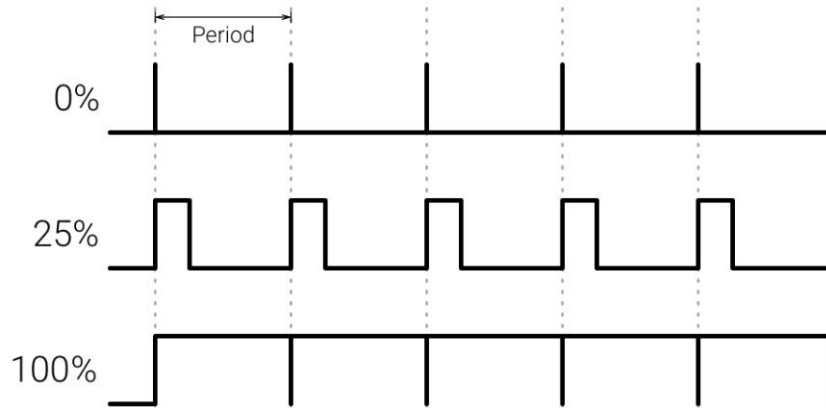
The period is the time each cycle takes and is the inverse of frequency.

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.

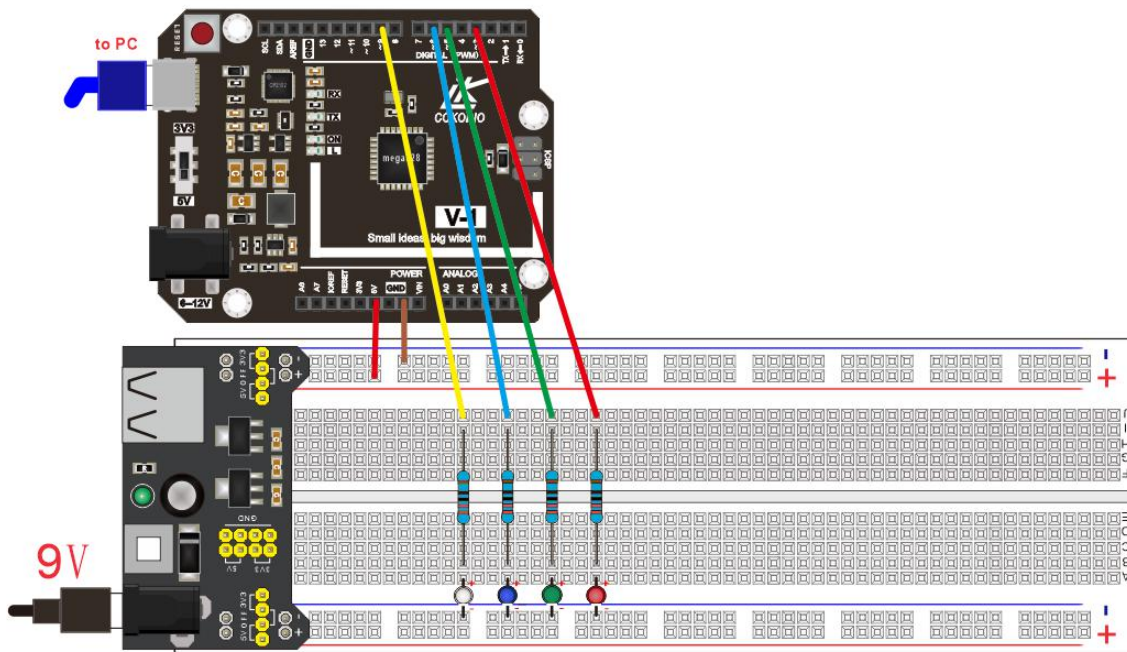
V-1 board only has 3, 5, 6, 9, 10, 11 pins to output PWM signal, and the default frequency of the signal is 1K Hz. When the PWM signal outputs high level, the LED light is on, and when the output is low, the LED is off. Controlling the high and low duty cycles controls the brightness of the LEDs. The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

### 3.1 Code

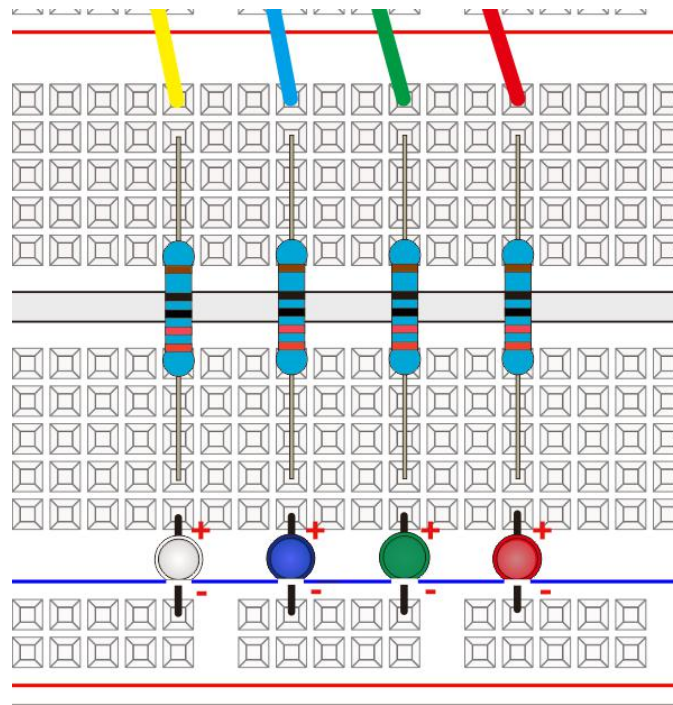
```
#define LED_R 3
#define LED_G 5
#define LED_B 6
#define LED_W 9
int PWM_data;
void setup() {
    pinMode(LED_R,OUTPUT);
    pinMode(LED_G,OUTPUT);
    pinMode(LED_B,OUTPUT);
    pinMode(LED_W,OUTPUT);
}
void loop() {
    for(PWM_data=0;PWM_data<=255;PWM_data++){
        analogWrite(LED_R,PWM_data); // PWM
        analogWrite(LED_G,PWM_data);
        analogWrite(LED_B,PWM_data);
        analogWrite(LED_W,PWM_data);
        delay(10);
    }
    for(PWM_data=255;PWM_data>=0;PWM_data--){
        analogWrite(LED_R,PWM_data);
        analogWrite(LED_G,PWM_data);
        analogWrite(LED_B,PWM_data);
        analogWrite(LED_W,PWM_data);
        delay(10);
    }
}
```



### 3.2 Connection Diagram



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



---

### 3.3 Experiment

After uploading the code, unplug the USB cable from the V-1 board, connect the external power supply to the power module, and then turn on the switch on the power module. The LED on the breadboard will change from dark to bright, and then from bright to dark, as shown below :

