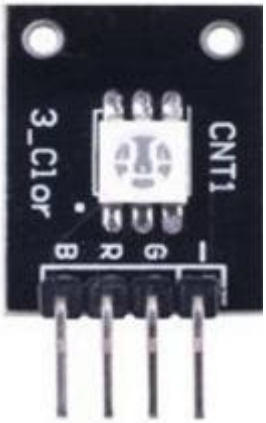
www.kumantech.com

RGB_SMD_LED

(aka 3-Color LED)



Overview

The “3-Color LED” and “3_Clor LED” are conventional names for an RGB LED in a surface-mount device (SMD) that has been packaged onto a breakout board. Like the glass bulb RGB LED, three-color LEDS contain three separate light-emitting diodes in red, green, and blue colors. In different intensities, these three component colors fuse together to form millions of separately perceivable colors. In this experiment, you’ll learn to connect and program the three-color LED to produce a flashing sequence of six different colors.

Experimental Materials

```
Raspberry Pi      x1
Breadboard        x1
Three-color LED   x1
Resistors (330Ω)  x3
Dupont jumper wires
```

Experimental Procedure

1. If you have not done so already, prepare your development system by installing the Python interpreter, RPi.GPIO library, and wiringPi library as described in READ_ME_FIRST.TXT.
2. Install the three-color LED in your breadboard and use resistors and Dupont jumper wires as illustrated in the Wiring Diagram below.

3. Execute the sample stored in this experiment's subfolder.
If using C, compile and execute the C code:

```
cd Code/C
gcc 3colorLED.c -o 3colorLED.out -lwiringPi
./3colorLED.out
```

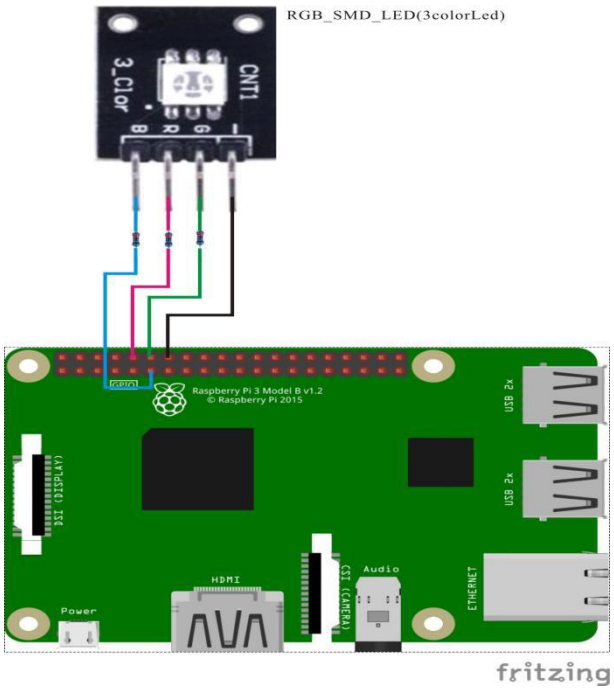
If using Python, launch the Python script:

```
cd Code/Python
python 3colorLED.py
```

4. Make experimental observations. The LED should alternately flash red, green, blue, yellow, magenta and cyan in a cycle.

The sample code uses pulse-width modulation (PWM) to vary the output voltage of the red, green, and blue GPIO pins, varying the intensity of light on the LED. In software, each individual component can be set to 101 different values (using Python's ChangeDutyCycle() or C's softPwmWrite() routines), so the LED can produce 101^3—over a million—distinct color combinations. LEDs are ubiquitous illumination sources for their durability and energy efficiency; and RGB is an additive color model drawn from the human perception of colors and widely used in electronic color imaging.

Wiring Diagram



Three-color LED pin position:

R ↔ Raspberry Pi pin 10 (through resistor)

G ↔ Raspberry Pi pin 12 (through resistor)

B ↔ Raspberry Pi pin 11 (through resistor)

\- ↔ Raspberry Pi GND

Sample Code

Python Code

```
#!/usr/bin/env python
import RPi.GPIO as GPIO
import time

colors = [0xFF0000, 0x00FF00, 0x0000FF, 0xFFFF00, 0xFF00FF,
0x00FFFF]
pins = {'pin_R':10, 'pin_G':12, 'pin_B':11}  # pins is a dict

GPIO.setmode(GPIO.BOARD) # Numbers GPIOs by physical location

for i in pins:
   GPIO.setup(pins[i], GPIO.OUT)   # Set pins' mode is output

p_R = GPIO.PWM(pins['pin_R'], 2000)  # set Frequece to 2KHz
p_G = GPIO.PWM(pins['pin_G'], 2000)
p_B = GPIO.PWM(pins['pin_B'], 2000)

p_R.start(0)      # Initial duty Cycle = 0(leds off)
p_G.start(0)
p_B.start(0)

def map(x, in_min, in_max, out_min, out_max):
   return (x - in_min) * (out_max - out_min) / (in_max - in_min)
+ out_min

def setColor(col):
   R_val = (col & 0xFF0000) >> 16
   G_val = (col & 0x00FF00) >> 8
```

```
    B_val = (col & 0x0000FF) >> 0

    R_val = map(R_val, 0, 255, 0, 100)
    G_val = map(G_val, 0, 255, 0, 100)
    B_val = map(B_val, 0, 255, 0, 100)

    p_R.ChangeDutyCycle(R_val)     # Change duty cycle
    p_G.ChangeDutyCycle(G_val)
    p_B.ChangeDutyCycle(B_val)

try:
    while True:
        for col in colors:
            setColor(col)
            time.sleep(0.5)
except KeyboardInterrupt:
    p_R.stop()
    p_G.stop()
    p_B.stop()
    for i in pins:
        GPIO.output(pins[i], GPIO.HIGH)    # Turn off all leds
    GPIO.cleanup()
```

C Code

```
#include <wiringPi.h>
#include <softPwm.h>
#include <stdio.h>

typedef unsigned char uchar;

#define LedPinRed    16
#define LedPinGreen  1
#define LedPinBlue   0

void ledInit(void)
{
    softPwmCreate(LedPinRed,  0, 100);
    softPwmCreate(LedPinGreen,0, 100);
    softPwmCreate(LedPinBlue, 0, 100);
}
```

```
uchar map(uchar val, uchar in_min, uchar in_max, uchar out_min,
uchar out_max)
{
   uchar tmp = 0;
   tmp = (val - in_min) * (out_max - out_min) / (in_max - in_min)
+ out_min;

   return tmp;
}

void ledColorSet(uchar r_val, uchar g_val, uchar b_val)
{
   uchar R_val, G_val, B_val;
   R_val = map(r_val, 0, 255, 0, 100);
   G_val = map(g_val, 0, 255, 0, 100);
   B_val = map(b_val, 0, 255, 0, 100);

   softPwmWrite(LedPinRed,   R_val);
   softPwmWrite(LedPinGreen, G_val);
   softPwmWrite(LedPinBlue,  B_val);
}

int main(void)
{
   int i;

   if(wiringPiSetup() == -1)
   {
      printf("setup wiringPi failed !");
      return 1;
   }

   ledInit();
   while(1)
   {
      ledColorSet(0xff,0x00,0x00);   //red
      delay(1000);
      ledColorSet(0x00,0xff,0x00);  //green
      delay(1000);
      ledColorSet(0x00,0x00,0xff);  //Blue
      delay(1000);
      ledColorSet(0xff,0xff,0x00);  //yellow
```

```
        delay(1000);
        ledColorSet(0xff,0x00,0xff);  //Magenta
        delay(1000);
        ledColorSet(0x00,0xff,0xff);  //Cyan
        delay(1000);
    }

    return 0;
}
```