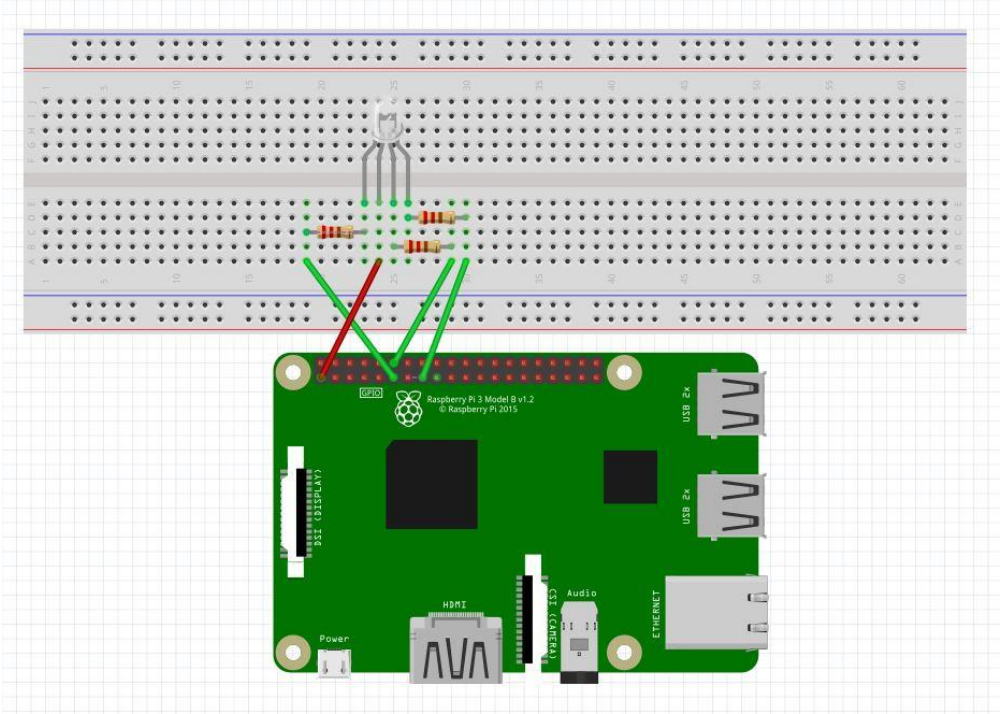
Internet of Things

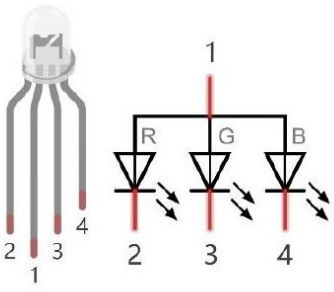
Practical No 4: Controlling an LED Using Raspberry Pi.

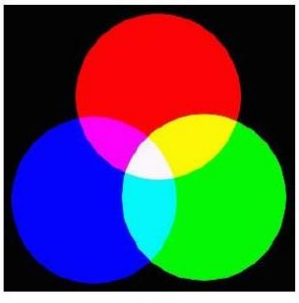
- We have to connect the Light Emitted Diode (LED) with bread-board and hard wired with Programmable Raspberry circuit. It will perform our code and blinks at interval like 5 seconds.



Main architecture

- As figure show, we connect register and LED light and Raspberry input port and Ground and Circuit switches.
  - I want to see RGB Colour as(RED GREEN BLUE)

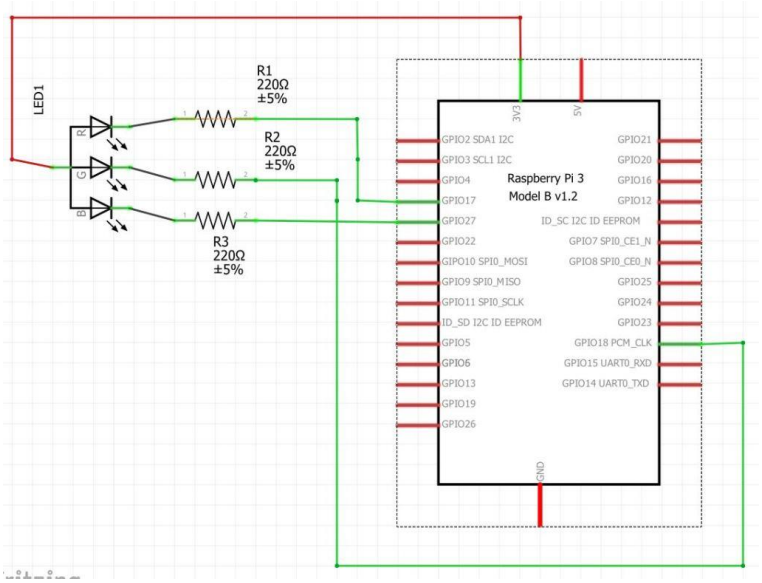




RGB

- **RGB LED**

- After this we Show pins of Raspberry GPI017 and GPI027 to switch with Register



Circuit Diagram

Python Script to Control RGB LED with Raspberry Pi

- Since each of the LED in an RGB LED is controlled via PWM, we will control 3 GPIO pins to generate PWM Signal.
- First, initialize all of the utilized GPIO resources as a PWM channel. We defined a tuple named pins to abstract the GPIO pins that correspond to the RGB pins of the LED. Also, since this is a Common Anode RGB LED, make sure you set the initial signal too HIGH to turn off the LED.
- We will be displaying different colours for each second in this script. Its also a great practice to keep your code simple by defining functions such as setColor**()**. This function sets the duty cycle for each of the PWM channels.

```
From gpiozero import LED    # It is for just one colour LED
Import time
K = input (“Sleep Time:”)
Led = LED (18)
While True:
        led.toggle()
        time.sllep(k)
        led.toggle()
        time.sleep(k)
```

CODE

```
import RPi.GPIO as GPIO

import time
import random

pins = (11,12,13) # R = 11, G = 12, B = 13

def setup():
    global pwmR, pwmG, pwmB
    GPIO.setmode(GPIO.BOARD)
    for i in pins:  # iterate on the RGB pins, initialize each and set to
HIGH to turn it off (COMMON ANODE)
        GPIO.setup(i, GPIO.OUT)
        GPIO.setup(i, GPIO.HIGH)
    pwmR = GPIO.PWM(pins[0], 2000)  # set each PWM pin to 2 KHz
    pwmG = GPIO.PWM(pins[1], 2000)
    pwmB = GPIO.PWM(pins[2], 2000)
    pwmR.start(0)   # initially set to 0 duty cycle
    pwmG.start(0)
    pwmB.start(0)

def setColor(r, g, b):  # 0 ~ 100 values since 0 ~ 100 only for duty cycle
    pwmR.ChangeDutyCycle(r)
    pwmG.ChangeDutyCycle(g)
    pwmB.ChangeDutyCycle(b)

def displayColors():
    setColor(100, 0, 0) #   red color
    time.sleep(1)   # 1s
    setColor(0, 100, 0) # green
    time.sleep(1)   # 1s
    setColor(0, 0, 100) # blue
    time.sleep(1)   # 1s
    setColor(100, 100, 0) # yellow
    time.sleep(1)   # 1s
    setColor(0, 100, 100) # cyan
    time.sleep(1)   # 1s
    setColor(100, 0, 100) # magenta
    time.sleep(1)   # 1s
    setColor(50, 0, 0) # maroon
    time.sleep(1)   # 1s
    setColor(50, 0, 50) # purple
    time.sleep(1)   # 1s
    setColor(0, 0, 50) # navy
    time.sleep(1)   # 1s

def destroy():
    pwmR.stop()
    pwmG.stop()
    pwmB.stop()
    GPIO.cleanup()

if __name__ == '__main__':
    setup()
    displayColors()
    destroy()
```