

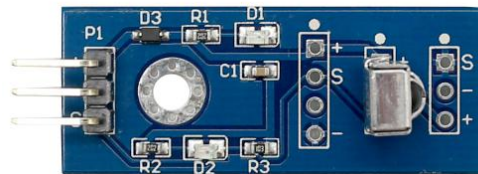
# 15.IR\_Receiver

## Introduction

Each button of an IR remote control (as shown below) has a string of specific encoding. When a button is pressed, the IR transmitter in the remote control will send out the corresponding IR encoding signals. On the other side, when the IR receiver receives certain encoding signals, it will decode them to identify which button is pressed.

## Hardware Required

- ✓ 1 \* Raspberry Pi
- ✓ 1 \* Breadboard
- ✓ 1 \* Network cable (or USB wireless network adapter)
- ✓ 1 \* IR Receiver module
- ✓ 1 \* IR Remote
- ✓ Several Jumper Wires

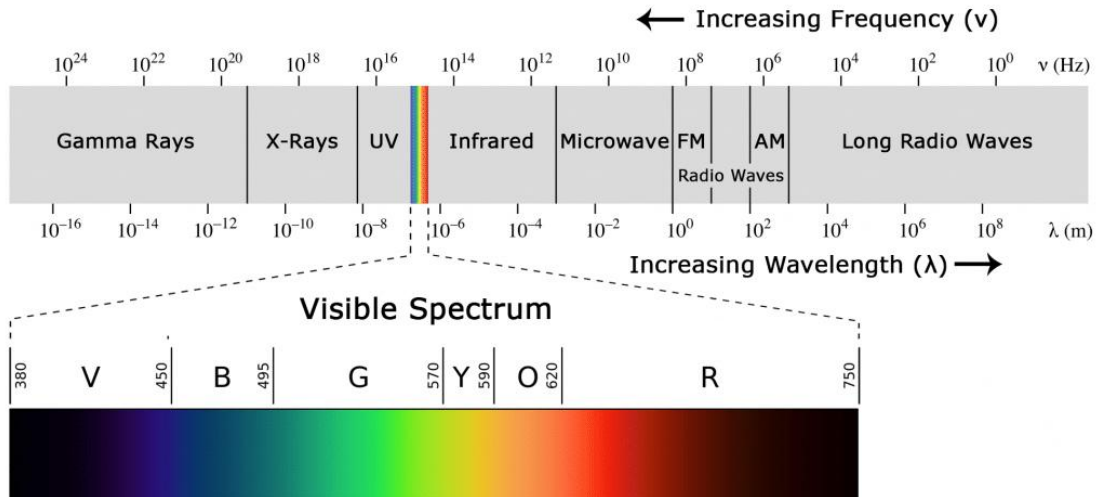


## Principle

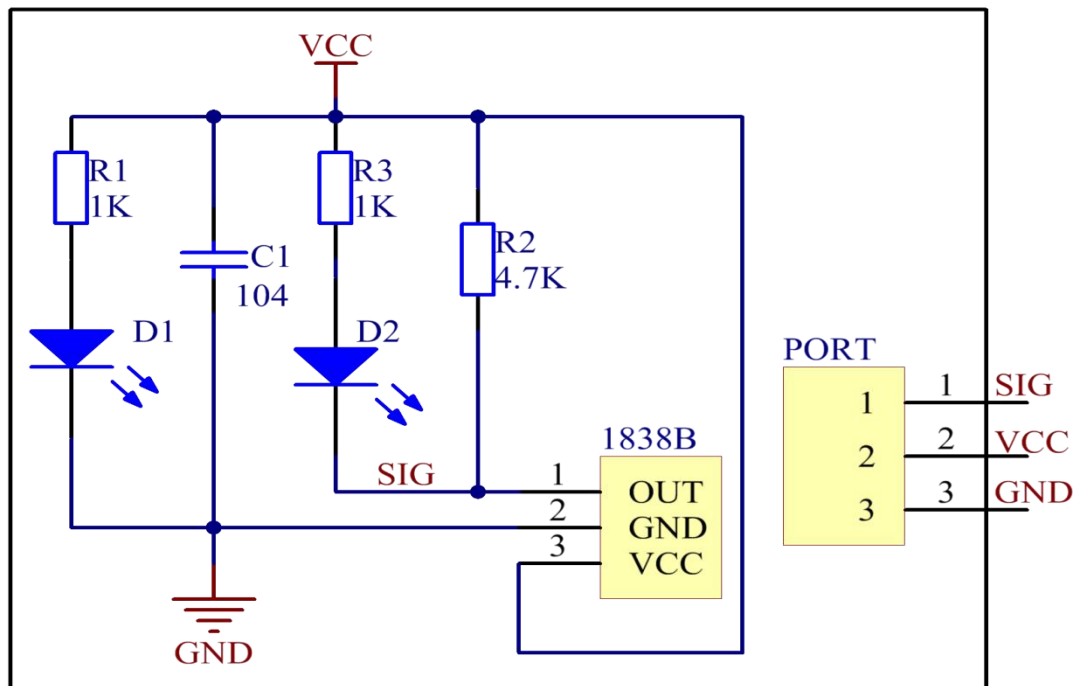
### IR Receiver Module

Infrared (IR) communication is a widely used and easy to implement wireless technology that has many useful applications. The most prominent examples in day to day life are TV/video remote controls, motion sensors, and infrared thermometers.

There are plenty of interesting Arduino projects that use IR communication too. With a simple IR transmitter and receiver, you can make remote controlled robots, distance sensors, heart rate monitors, DSLR camera remote controls, TV remote controls, and lots more. Infrared radiation is a form of light similar to the light we see all around us. The only difference between IR light and visible light is the frequency and wavelength. Infrared radiation lies outside the range of visible light, so humans can't see it:



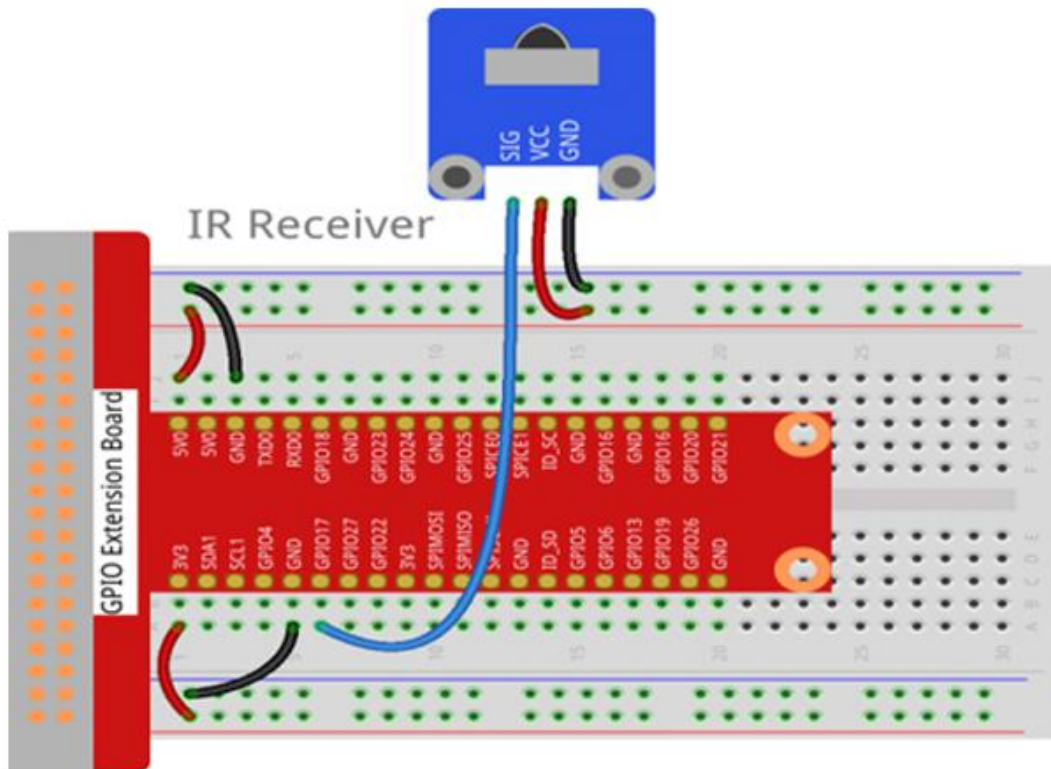
## Schematic Diagram



## Experimental Procedures

Raspberry Pi	T-Cobbler	IR Receiver Module
GPIO0	GPIO17	SIG
5V	5V0	VCC
GND	GND	GND

## Step 1: Build the circuit



## For C Language Users

### Step 2: Open the code file.

```
cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/C/15.IR_Receiver
```

Note: Change directory to the path of the code in this experiment via cd.

### Step 3: Compile the code.

```
gcc 15.IR_Receiver.c -o IR_Receiver.out -lwiringPi
```

### Step 4: Run the executable file .

```
sudo ./IR_Receiver.out
```

Here you can see the LED on the module blinking, and "Received infrared. cnt = xxx" will be printed on the screen.

## Code

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

#define IR 0

int cnt = 0;
```

```

void myISR(void)
{
    printf("Received infrared. cnt = %d\n", ++cnt);
}

int main(void)
{
    if(wiringPiSetup() == -1){ //when initialize wiring failed,print message to
screen
        printf("setup wiringPi failed !");
        return 1;
    }
    if(wiringPiISR(IR, INT_EDGE_FALLING, &myISR) == -1){
        printf("setup ISR failed !");
        return 1;
    }
    //pinMode(IR, INPUT);
    while(1);
    return 0;
}

```

## For Python Language Users

### Step 2: Open the code file.

```
cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/Python
```

### Step 3: Run the code.

```
sudo python3 15.IR_Receiver.py
```

### Code

```

#!/usr/bin/env python3
import RPi.GPIO as GPIO
IrPin = 11

```

```

count = 0

def setup():
    GPIO.setmode(GPIO.BOARD)          # Numbers GPIOs by physical location
    GPIO.setup(IrPin, GPIO.IN, pull_up_down=GPIO.PUD_UP)

def cnt(ev=None):
    global count
    count += 1
    print ('Received infrared. cnt = ', count)

def loop():
    GPIO.add_event_detect(IrPin, GPIO.FALLING, callback=cnt) # wait for falling
    while True:
        pass          # Don't do anything

def destroy():
    GPIO.cleanup()          # Release resource

if __name__ == '__main__':          # Program start from here
    setup()
    try:
        loop()
    except KeyboardInterrupt:      # When 'Ctrl+C' is pressed, the child program
destroy() will be executed.
        destroy()

```

## Phenomenon Picture

