Infrared transmission experiment

Introduction the devices

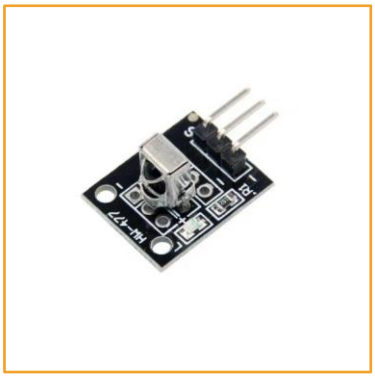
This time, we will introduce the infrared transmitter and receiver modules, which actually play an important role in our daily life. Now such a device is widely used in many home appliances, such as air conditioning, television, DVD, etc. It is based on wireless remote sensing, but also a remote control, it is necessary to study its principle and how to use.

Infrared transmitting tube and infrared receiving tube are devices that convert electric energy into near-infrared light directly. Its structure and principle are similar to ordinary light-emitting diodes, but the semiconductor material is different.

The infrared receiver is a receiving, amplifying and demodulating device. The internal integrated circuit has been demodulated and the output is digital signal.

Infrared receiver

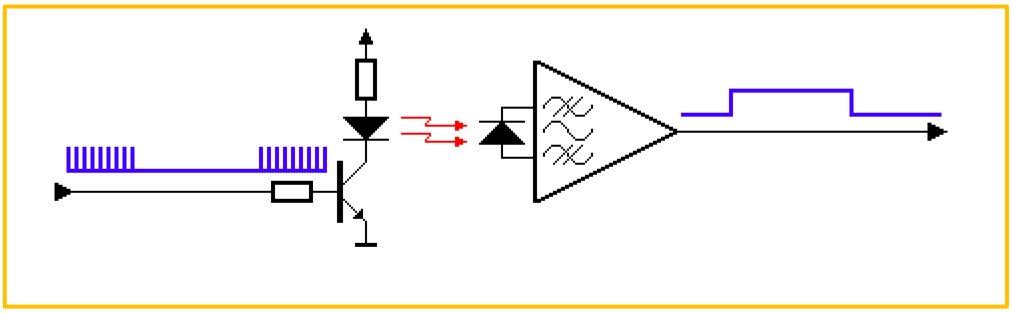
Infrared transmitter



Infrared receiver module

Working Principle

To understand the structure of infrared receiver: infrared receiver is composed of IC and PD. IC is the processing element of the receiver, mainly composed of silicon crystal and circuit. It is a highly integrated device. PD is a photodiode whose main function is to receive optical signals. The infrared emitting diode sends out the modulation signal, and the infrared receiving head recovers the signal after receiving, decoding, filtering and a series of operations.



Component List

- RaspberryPi mainboard
- Infrared transmitter *1
- Infrared receiver*1
- Several jumpers

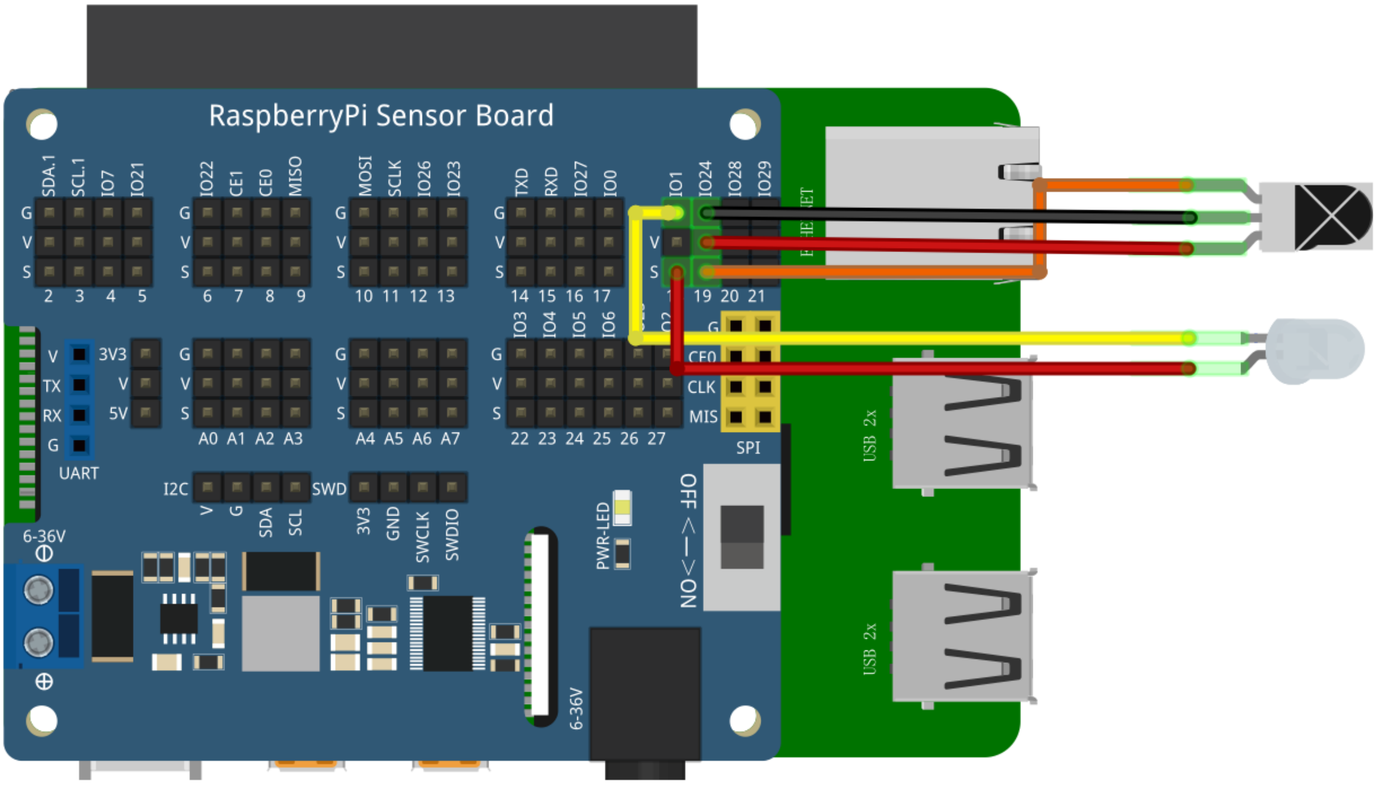
接线

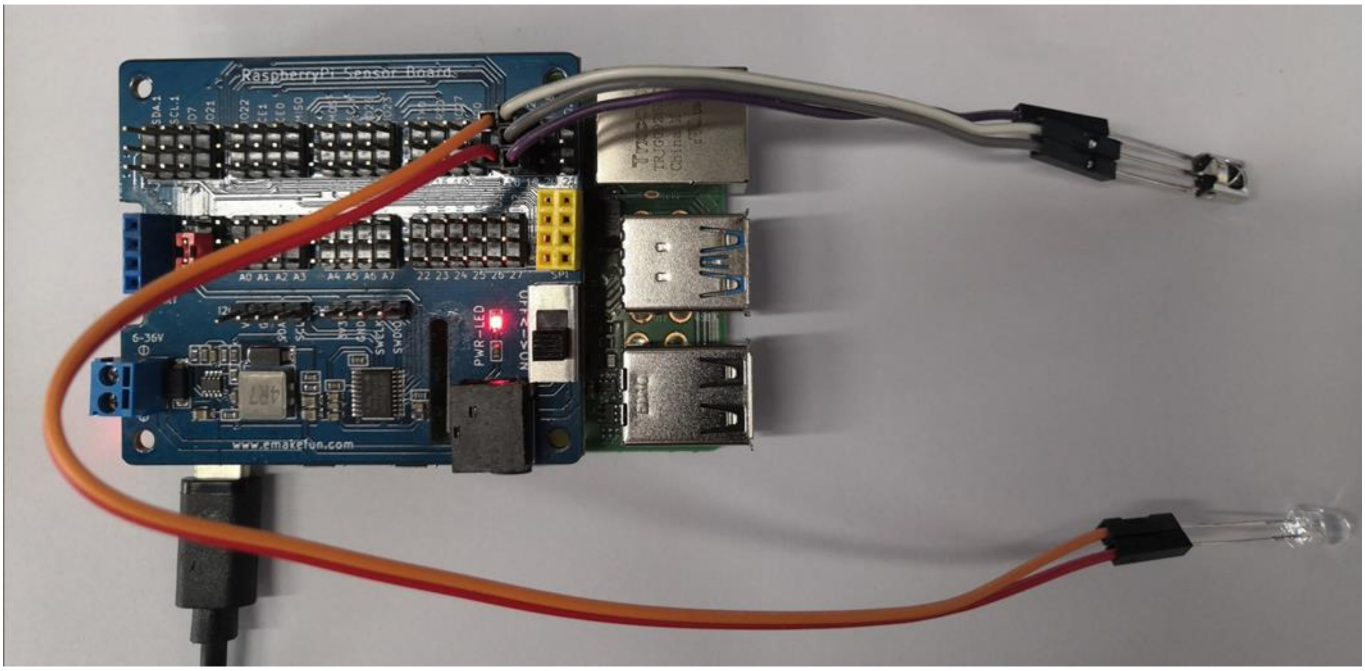
红外接收接线

RaspberryPi	Infrared receiver
IO24(wiringPi)/19(BCM)	S
GND	-
+5V	+

红外发送实验

RaspberryPi	Infrared transmitter
IO1(wiringPi)/18(BCM)	S
GND	GND


RaspberryPi Sensor Board
SDA.1 SCL.1 IO7 IO21
IO22 CE1 CE0 MISO
MOSI SCLK IO26 IO23
TXD RXD IO27 IO0
IO1 IO24 IO28 IO29
G
V
S
2 3 4 5
6 7 8 9
10 11 12 13
14 15 16 17
19 20 21
IO3 IO4 IO5 IO6
V 3V3
TX V
RX 5V
G
UART
A0 A1 A2 A3
A4 A5 A6 A7
22 23 24 25 26 27
CE0
CLK
MIS
SPI
I2C
V G SDA SCL
SWD
3V3 GND SWCLK SWDIO
PWR-LED
OFF <—> ON
6-36V
USB 2x
USB 2x


RaspberryPi Sensor Board
A0 A1 A2 A3
A4 A5 A6 A7
22 23 24 25 26 27
SPI
PWR-LED
6-36V
www.emakefun.com

Code

Receiver

C++   main program

```
#include "IR_REC.h"

int main()
{
    int key;
    if(wiringPiSetup() == -1){
       printf("setup wiringPi failed !");
       return 1;
    }
    while(1){
    key = GetKey();
       if (key != ERROR) {
           printf("key: %x \n",key);
       }
    }
}
```

Python program

```
#!/usr/bin/python
# -*- coding:utf-8 -*-
import RPi.GPIO as GPIO
import time
ERROR = 0xFE
PIN = 19
    #Define infrared receiver pin

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(PIN, GPIO.IN, GPIO.PUD_UP)   #Set the infrared receiving pin to pull-up mode

def getKey():
    byte = [0, 0, 0, 0]
    if IRStart() == False:     #Judging infrared guide pulse
       time.sleep(0.11)       # One message frame lasts 108 ms.
       return ERROR
    else:
        for i in range(0, 4):
                byte[i] = getByte()  # #Receive 32-bit infrared data (address, address
inversion, data, data inversion)
        if byte[0] + byte[1] == 0xff and byte[2] + byte[3] == 0xff:  # Check whether the
received data is correct
        print("right")
            return byte[2]
        else:
        print("error")
            return ERROR
    #return byte[2]
def IRStart():
    timeFallingEdge = [0, 0]
    timeRisingEdge = 0
    timeSpan = [0, 0]
    GPIO.wait_for_edge(PIN, GPIO.FALLING)
    timeFallingEdge[0] = time.time()
    GPIO.wait_for_edge(PIN, GPIO.RISING)
    timeRisingEdge = time.time()
    GPIO.wait_for_edge(PIN, GPIO.FALLING)
    timeFallingEdge[1] = time.time()

    timeSpan[0] = timeRisingEdge - timeFallingEdge[0]
    timeSpan[1] = timeFallingEdge[1] - timeRisingEdge
    print(timeSpan[0],timeSpan[1])
    if timeSpan[0] > 0.0085 and timeSpan[0] < 0.0095 and timeSpan[1] > 0.004 and timeSpan[1]
< 0.005:
    print("1")
        return True
    else:
    print("0")
        return False
def getByte():
    byte = 0
    timeRisingEdge = 0
    timeFallingEdge = 0
    timeSpan = 0
    for i in range(0, 8):
        GPIO.wait_for_edge(PIN, GPIO.RISING)
        timeRisingEdge = time.time()
        GPIO.wait_for_edge(PIN, GPIO.FALLING)
        timeFallingEdge = time.time()
        timeSpan = timeFallingEdge - timeRisingEdge
        if timeSpan > 0.0016 and timeSpan < 0.0018:
            byte  |= 1 << i
    return byte
print('IRM Test Start ...')
try:
    while True:
        key = getKey()     #Read infrared pulse
        if(key != ERROR):  #Print infrared pulse value
            print("Get the key: 0x%02x" %key)
except KeyboardInterrupt:
    GPIO.cleanup()
```

Java program

```
import com.pi4j.wiringpi.Gpio;

public class IR_NEC {
        static int PIN = 24;
        static int ERROR = 0xfe, key;
        static long timeRisingEdge, timeFallingEdge, timeRising, timeFalling_0,
timeFalling_1;
        static long timeSpan_val = 0;
        static long[] time_span = new long[2];

    static {
        if (Gpio.wiringPiSetup() == -1) {
            System.out.println(" ==>> GPIO SETUP FAILED");
        }

        Gpio.pinMode(PIN, Gpio.INPUT);
    }

    public static boolean  IRStart() {
        while(!(Gpio.digitalRead(PIN) == 0));
        timeFalling_0 = gettimeofday();
        while(!(Gpio.digitalRead(PIN) == 1));
        timeRising = gettimeofday();
        while(!(Gpio.digitalRead(PIN) == 0));
        timeFalling_1 = gettimeofday();
        time_span[0] = timeRising - timeFalling_0;
        time_span[1] = timeFalling_1 - timeRising;

//      System.out.println("start_time " + time_span[0] + "," +time_span[1]);

        if (time_span[0] > 8500 && time_span[0] < 9500 && time_span[1] >= 4000 &&
time_span[1] <= 5000)
        {
//          System.out.println("start singe************");
            return true;
        }
        else    {
            return false;
        }
    }

    public static long gettimeofday() {
//      return System.currentTimeMillis() ;// +System.nanoTime() / 1000;
        return System.nanoTime() / 1000;

    }

    public static int GetByte() {
        int byte_val = 0;
        for (int i = 0; i < 8; i++) {
            while(!(Gpio.digitalRead(PIN) == 1));
            timeRisingEdge = gettimeofday();
            while(!(Gpio.digitalRead(PIN) == 0));
            timeFallingEdge = gettimeofday();
            timeSpan_val = timeFallingEdge - timeRisingEdge;
//          System.out.print("start byte  ");
//          System.out.println(timeSpan_val);

            if (timeSpan_val > 1500 && timeSpan_val < 1800)
                byte_val |= 1 << i;

        }
//  System.out.printf("byte_val: %x  \n", byte_val);

        return byte_val;
    }

    public static int GetKey() {
        int[] byte_val = new int[4];
        if (IRStart() == false) {
            Gpio.delay(108);
            return ERROR;
        } else {
            for (int i = 0; i < 4; i++) {
                byte_val[i] = GetByte();
//              System.out.printf("byte_val[%d]: %x \n",i, byte_val[i]);

            }
            if ((byte_val[0] + byte_val[1] == 0xff) && (byte_val[2] + byte_val[3] == 0xff))
{
                return byte_val[2];
            } else {
                    return ERROR;
              }
          }
     }

    public static void main(String args[]) {
        int rec_val;
        IR_NEC ir_nec = new IR_NEC();
        for ( ; ;) {
            rec_val = ir_nec.GetKey();
            if (rec_val != ERROR) {
                System.out.printf("key: %x \n",rec_val);
            }
        }
    }
}
```

Transmitter

C++ main program

```
#include "IR_SEND.h"

int main()
{
    if(wiringPiSetup() == -1){
        printf("setup wiringPi failed !");
        return 1;
    }
    while(1){
        IR_Send(0x45);
        delay(200);
    }
}
```

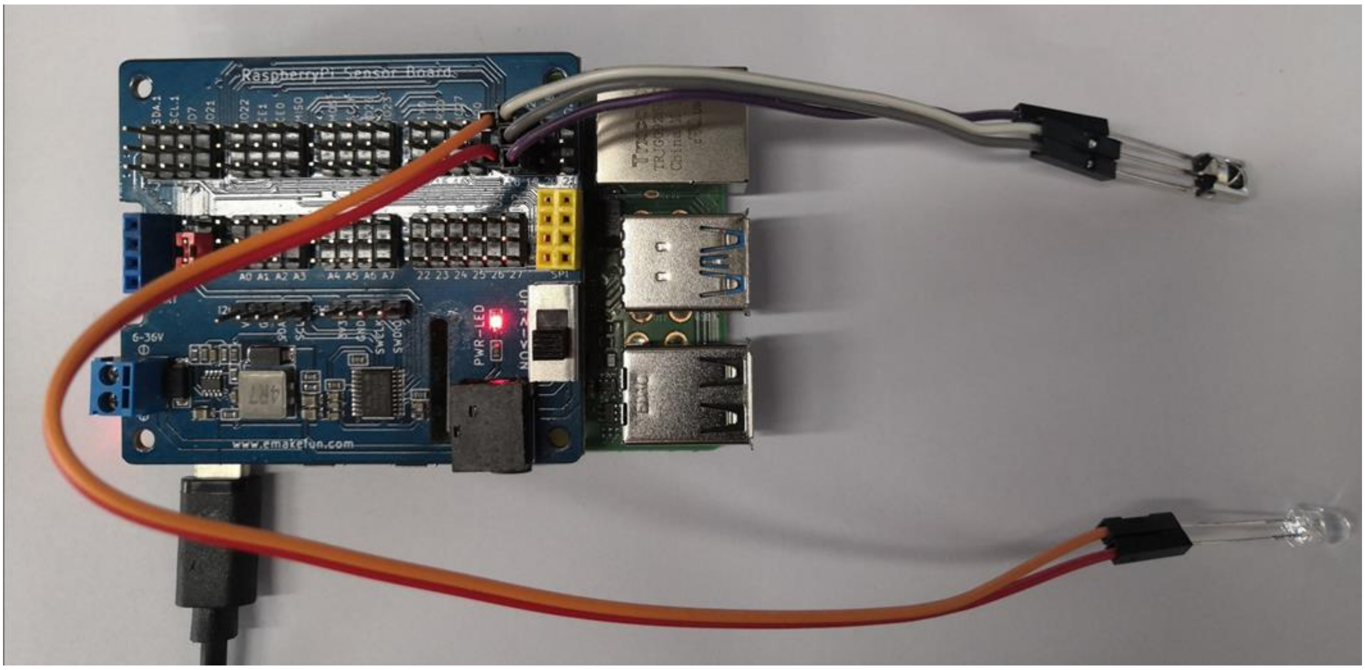
Python program

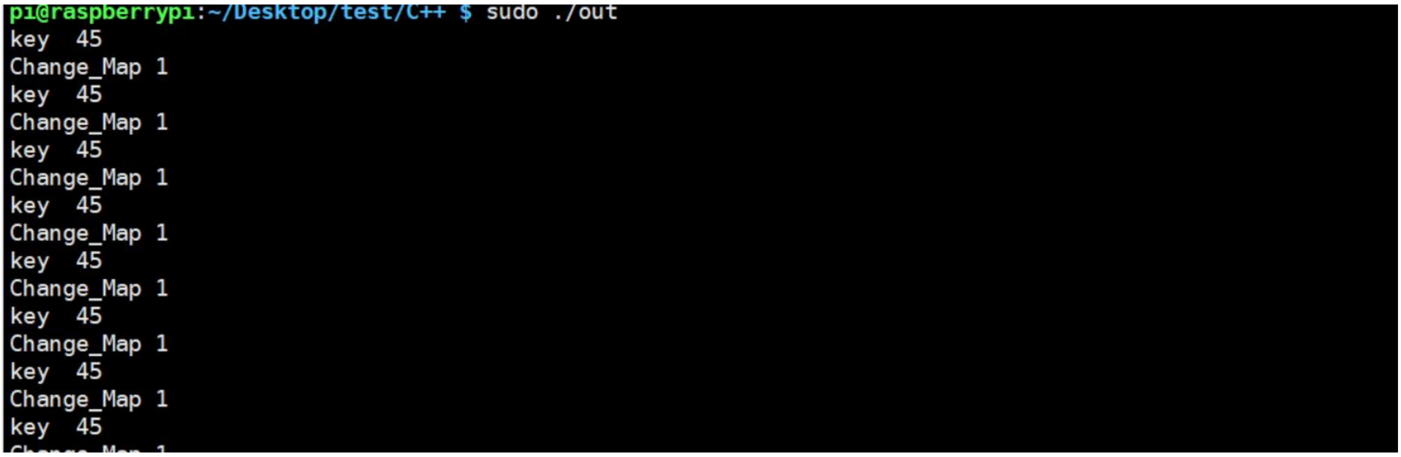
```
import ctypes
import time

so = ctypes.cdll.LoadLibrary
lib = so("./libpycallclass.so")

while True:
    lib.IR_Send(0x45)
    time.sleep(1)

```

Java program

```
import com.pi4j.wiringpi.Gpio;

public class IR_SEND_OBJ {
        static int irsys = 0xfe, ircode;
        static int PIN = 1;

    public static void IR_Send_Start() {
        Gpio.pwmWrite(IR_SEND_OBJ.PIN,22);
        Gpio.delayMicroseconds(4500);
        Gpio.delayMicroseconds(4500);
        Gpio.pwmWrite(IR_SEND_OBJ.PIN,0);
        Gpio.delayMicroseconds(4500);
    }

    public static void Send_Byte(){
        for (int i = 0; i < 8; i++) {
            if((IR_SEND_OBJ.ircode & 0x01) == 1) {
                Gpio.pwmWrite(IR_SEND_OBJ.PIN,22);
                Gpio.delayMicroseconds(560);
                Gpio.pwmWrite(IR_SEND_OBJ.PIN,0);
                Gpio.delayMicroseconds(1690);
            } else {
                Gpio.pwmWrite(IR_SEND_OBJ.PIN,22);
                Gpio.delayMicroseconds(560);
                Gpio.pwmWrite(IR_SEND_OBJ.PIN,0);
                Gpio.delayMicroseconds(560);
            }
            IR_SEND_OBJ.ircode = IR_SEND_OBJ.ircode >> 1;
        }
    }

    public static void IR_Send(int date) {
        Gpio.pinMode(IR_SEND_OBJ.PIN,Gpio.PWM_OUTPUT);
        Gpio.pwmSetMode(Gpio.PWM_MODE_MS);
        Gpio.pwmSetRange(45);
        Gpio.pwmSetClock(32);
        IR_SEND_OBJ.ircode = IR_SEND_OBJ.irsys;
        IR_SEND_OBJ.IR_Send_Start();
        IR_SEND_OBJ.Send_Byte();
        IR_SEND_OBJ.ircode = ~IR_SEND_OBJ.irsys;
        IR_SEND_OBJ.Send_Byte();
        IR_SEND_OBJ.ircode = date;
        IR_SEND_OBJ.Send_Byte();
        IR_SEND_OBJ.ircode = ~date;
        IR_SEND_OBJ.Send_Byte();
        Gpio.pwmWrite(IR_SEND_OBJ.PIN,22);
        Gpio.delayMicroseconds(400);
        Gpio.pwmWrite(IR_SEND_OBJ.PIN,0);
    }

    public static void main(String args[]) {
        // setup wiring pi
        if (Gpio.wiringPiSetup() == -1) {
            System.out.println(" ==>> GPIO SETUP FAILED");
        }
        while (true){
            IR_SEND_OBJ.IR_Send(0x45);
            Gpio.delay(1000);
        }
    }
}
```

Experiment Result





```
pi@raspberrypi:~/Desktop/test/C++ $ sudo ./out
key  45
Change_Map 1
key  45
Change_Map 1
key  45
Change_Map 1
key  45
Change_Map 1
key  45
Change_Map 1
key  45
Change_Map 1
key  45
Change_Map 1
key  45
Change_Map 1
```