# 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.

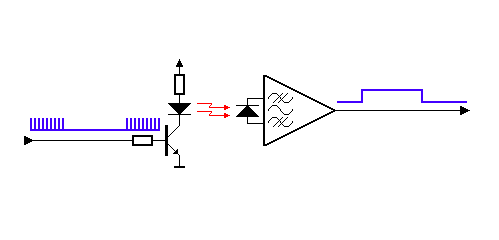
The infrared receiver is a receiving, amplifying and demodulating device. The internal integrated circuit has been demodulated and the output is digital signal.

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

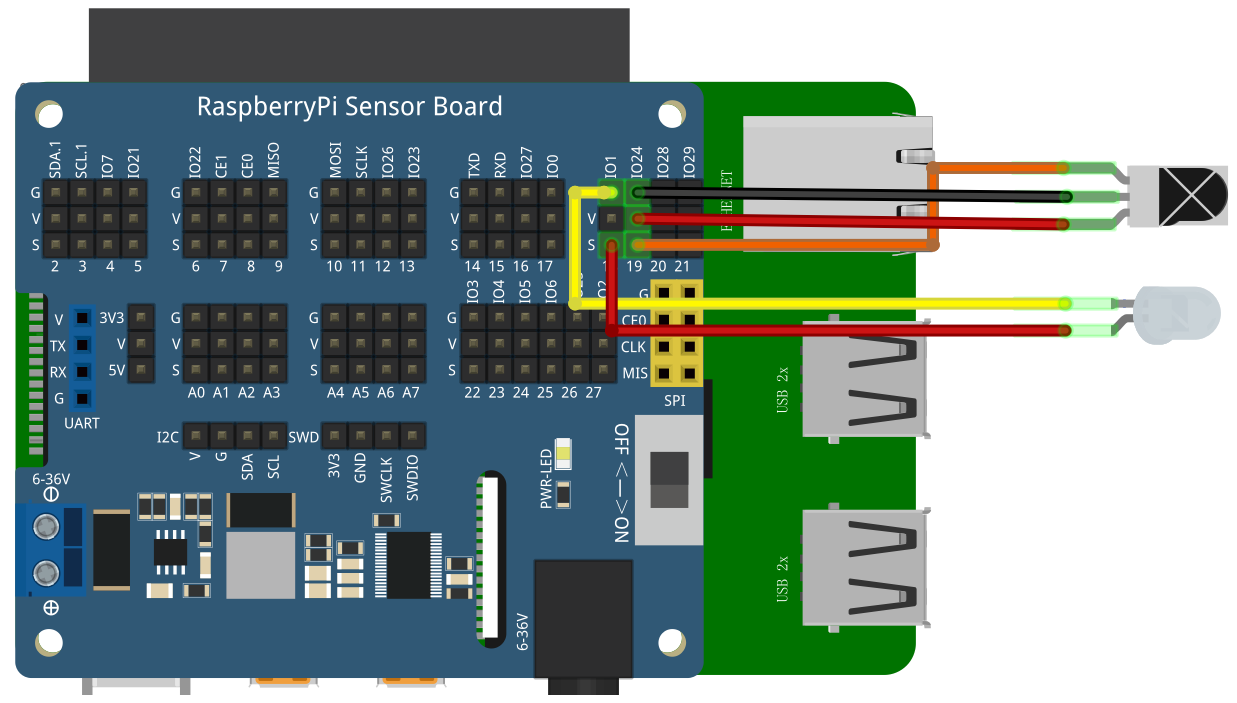
## 接线

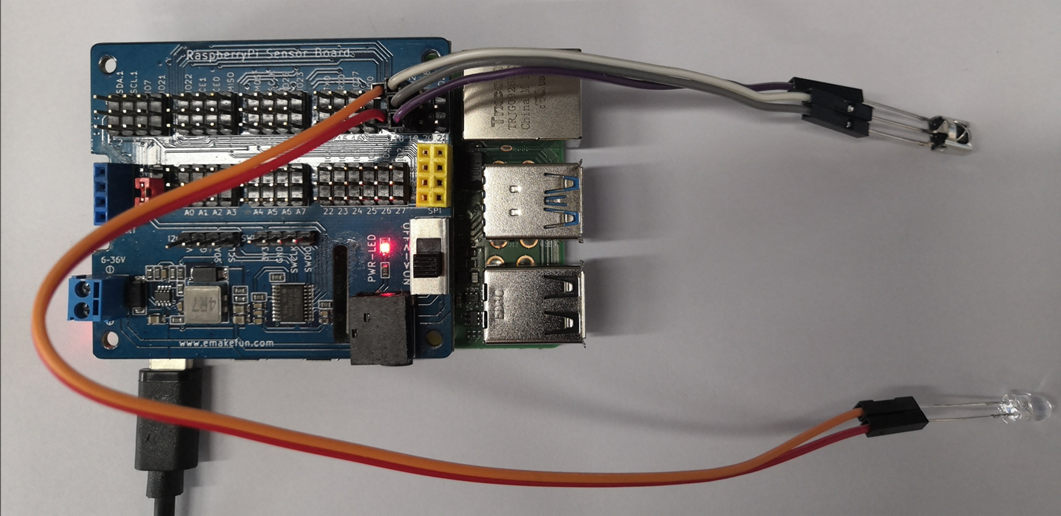
### 红外接收接线

|  |  |
| --- | --- |
| 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 |





## 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

