## 嵌入式操作系统树莓派GPIO接口编程实验报告

课程名称： 嵌入式操作系统  
实验名称： 树莓派GPIO接口编程  
姓　　名： \_\_\_\_\_周唯\_\_\_\_\_\_  
学　　号： \_\_U202217266\_\_  
实验日期： \_\_2025.10.11\_\_\_

## ****一、实验目的****

1. 掌握安装和使用树莓派系统
2. 学习GPIO接口的基本原理和使用方法
3. 实现LED七色灯控制、按钮交互和超声波测距功能
4. 掌握Python图形界面编程与硬件控制的结合应用

## ****二、实验设备****

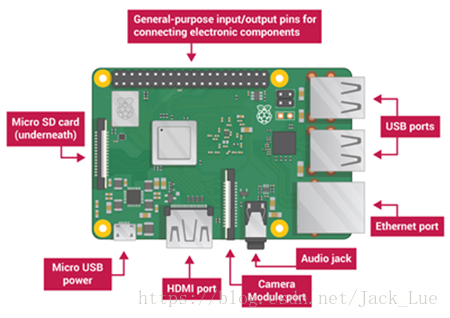
1. 树莓派3B × 1
2. T型板 × 1 + MB-102面包板 × 1 + 导线若干
3. 16GB Micro SD卡 × 1 + SD卡读卡器 × 1
4. RGB LED灯（四引脚CRGB） × 1
5. 轻触按键模块（三引脚+-S） × 1
6. 超声波测距模块（HC-SR04） × 1
7. 笔记本电脑（用于远程连接编程）

或显示屏+HDMI双端线用于直接连接

## ****三、实验原理****

### ****3.1 树莓派GPIO接口****

树莓派的主板配置结构，示意图如下：



图中标注部位的基本功能如下：

**SD card slot：**插入SD卡的位置，SD卡用于存储操作系统及其他文件数据，作用相当于笔记本中的硬盘

**Micro USB power connector：**电源口，连接供电电源（官方提示：在连接好其他所有外设后再接通电源）

**USB Port：**用于连接各种USB外设，如U盘、键盘、鼠标等，功能和笔记本电脑上的USB口一样，这款树莓派有4个USB口

Ethernet port：以太网口，用于连接有线网

Audio jack：用于连接耳机或者其他声音设备

HDMI port：和电脑上的HDMI口一样，用于视频输出，即连接显示器，当然HDMI也支持音频输出

GPIO ports：连接你自己的外设，和单片机的GPIO一样（可以连LED灯，按键，传感器之类的）

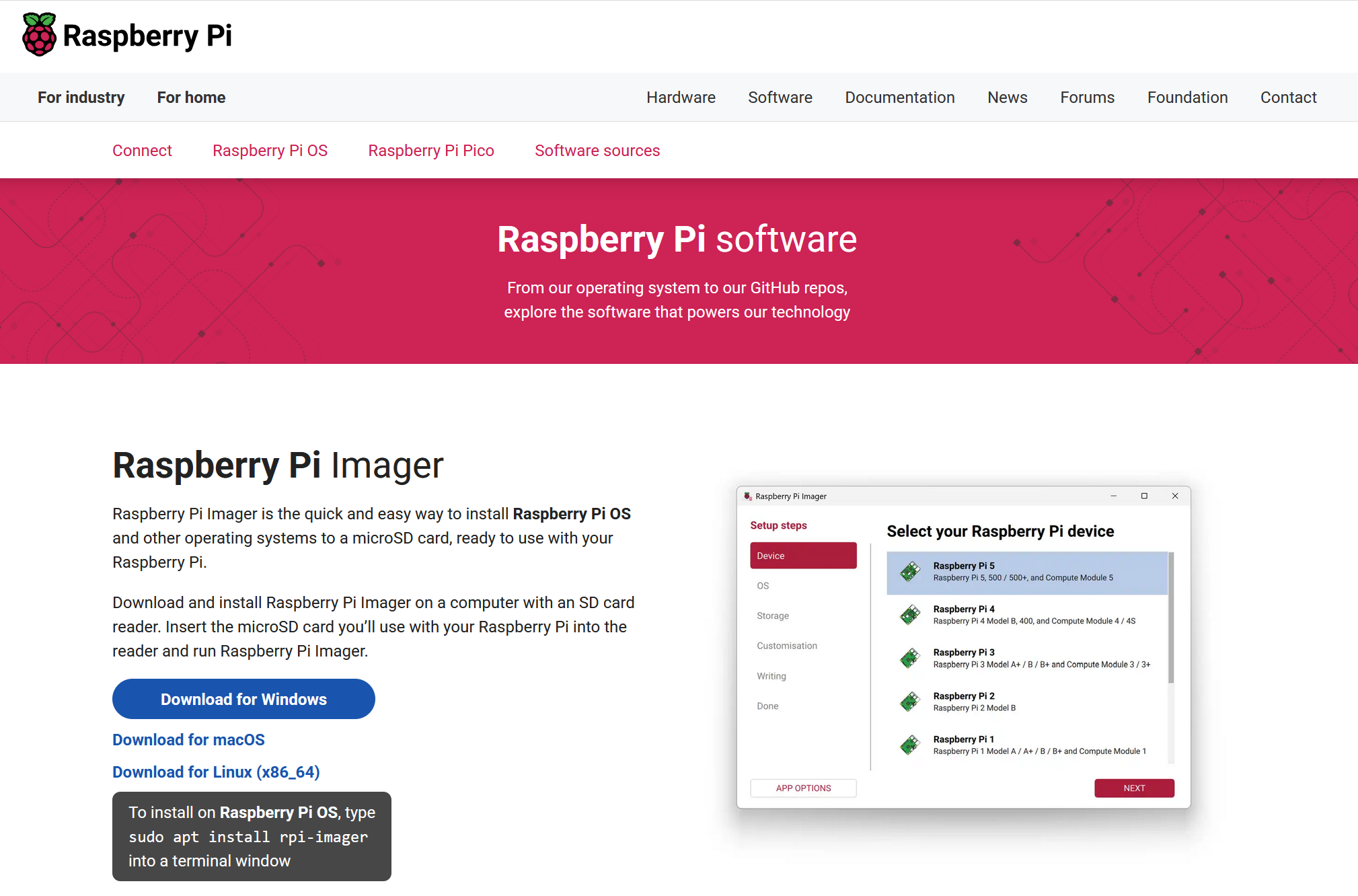
## ****四、实验步骤与结果****

### 4.1 树莓派系统安装

步骤一：准备好树莓派3B与一张SD卡（建议16GB以上），SD卡可预先格式化，也可随后安装系统时镜像烧录时再格式化。

步骤二：安装树莓派Imager

从电脑中进入树莓派官网<https://www.raspberrypi.com/software/> 安装树莓派Imager，

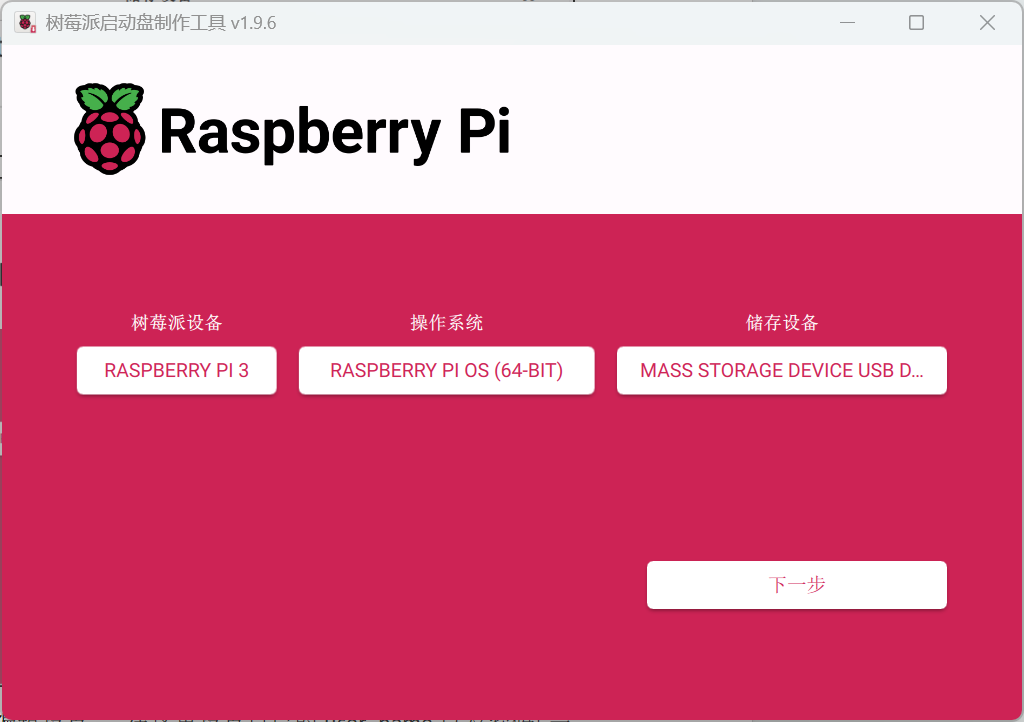


步骤三：烧录树莓派系统

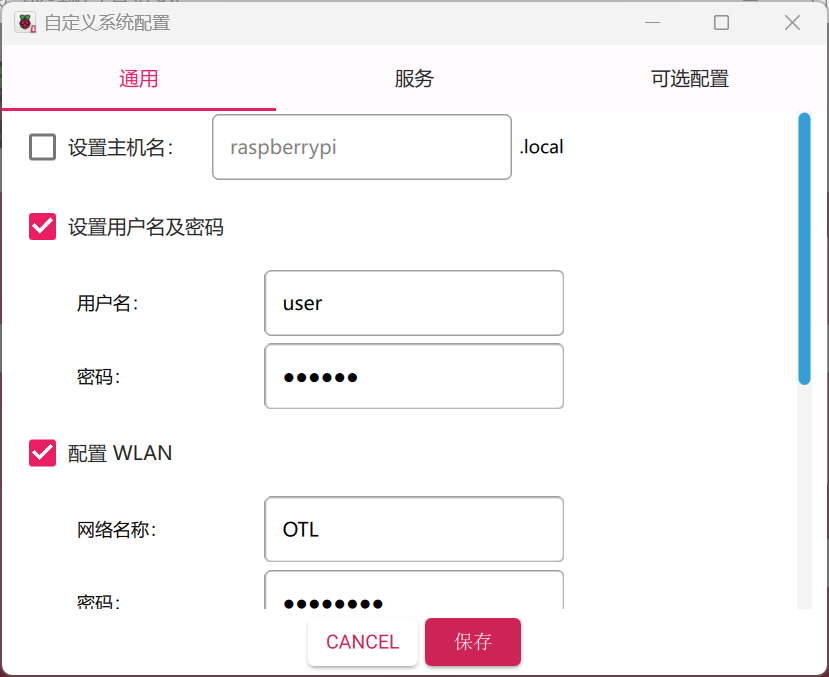
在打开之前，先**将SD卡插入SD卡读卡器中，并通过USB接口插入电脑中，**

开始进行镜像烧录，打开刚刚安装的Raspberry Pi Imager，Device选择自己的树莓派版本、操作系统、以及刚刚插入的SD卡作为存储设备，我这里使用的是树莓派3B、64bitOS。





选择完成后点击下一步，编辑设置，​在这里设置自己的user name以及密码（这里的用户名和密码用于后续的远程连接，需要记牢），设立的wifi以及wifi密码来设置（此处推荐使用手机热点，便于后续查询IP与MAC地址），修改好后点击保存即可（我们在这里修改后就**无需再设置ssh文件以及wpa\_supplicant.conf文件**了，简化步骤）。



然后就开始了镜像烧录程序，预计等待十多分钟左右，在烧录完成后就可以**拔下SD卡插入树莓派**了。

### 4.2 树莓派连接

树莓派可直接连接显示器或远程连接笔记本电脑。

#### 4.2.1 直连显示器（优先选择）

* 采用HDMI接口接上显示器，USB口插上鼠标键盘外设。
* 为树莓派与显示器接上电源线，按下开关，树莓派红灯（电源指示灯）亮，黄灯闪烁，系统安装没问题的话应该已经启动了。此处与电脑操作流程差不不大，可直接开始编程。

#### 4.2.2 远程连接笔记本电脑（较为复杂）

参考资料：https://zhuanlan.zhihu.com/p/1935064046777902281

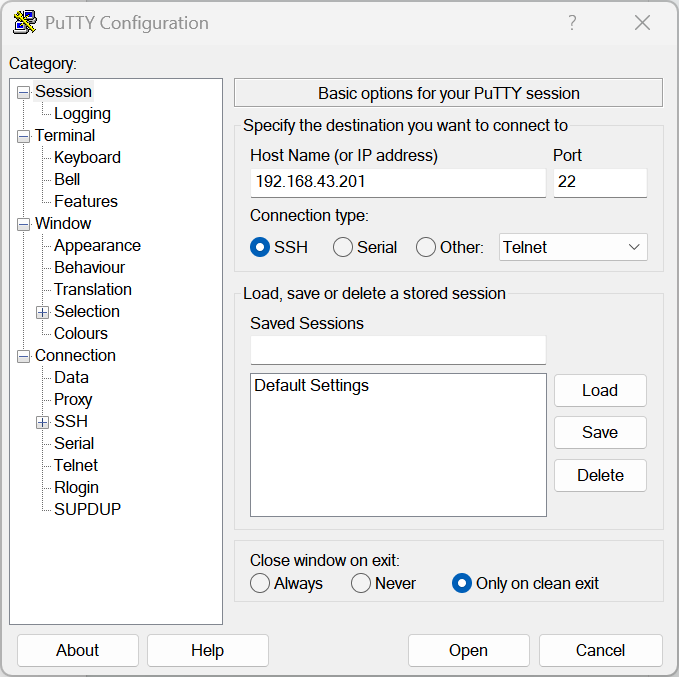
在给树莓派上电前先打开手机热点，确保手机热点打开后，**给树莓派上电，正常情况下树莓派的红灯会常亮，黄灯会闪烁，**此时观察手机热点的已连接设备，观察到树莓派连上后，让电脑也连上同一个热点。

此时在手机中查看热点已连接设备（以华为手机为例），若前述网络名称与密码配置正确则树莓派启动后会自动连接到热点，在手机连接设备中查看树莓派的IP地址。



此时电脑和树莓派都在同一个热点下，此时我们使用ssh让电脑连接树莓派，需要预先下载putty软件<https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>

这里我们直接点击putty打开软件，在这里输入树莓派的IP地址（手机内查询到的或5.2中自己配置的），点击OPEN即可连接，若出现warning，选择accept即可。

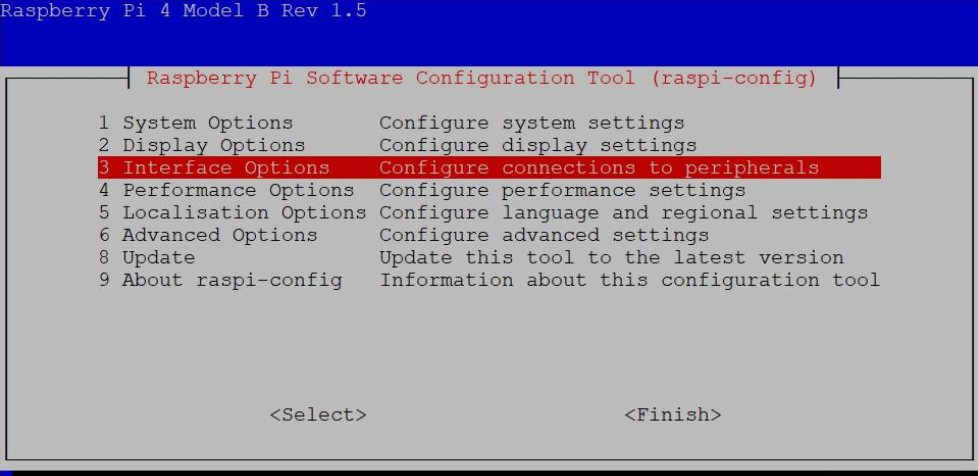


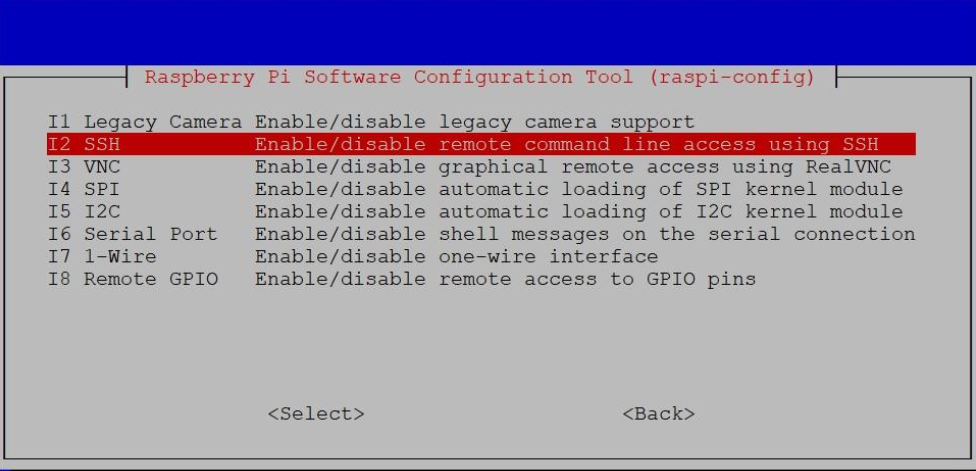
这里输入自己的user name以及对应的密码（密码默认不会显示，直接输即可），输完后enter即可。



连接成功后输入：sudo raspi-config

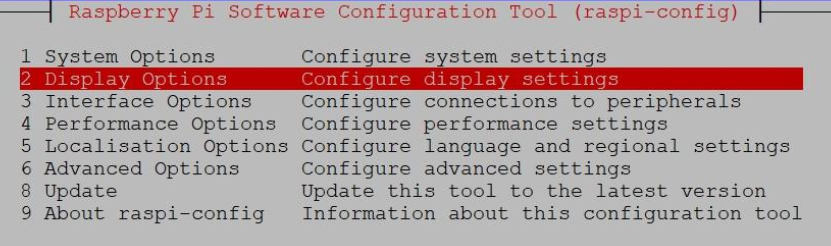
接下来通过上下左右以及enter键进行操作了，选择第三个，打开ssh，vnc，以及serial串口。



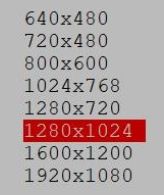
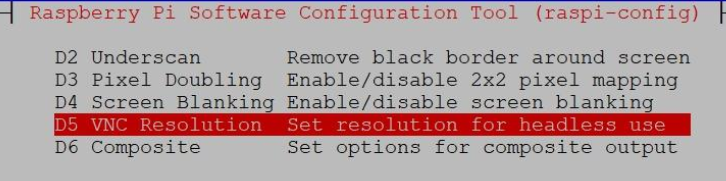


选择ssh后点击yes，**随后对I3 VNC和I6 Serial同理选yes**

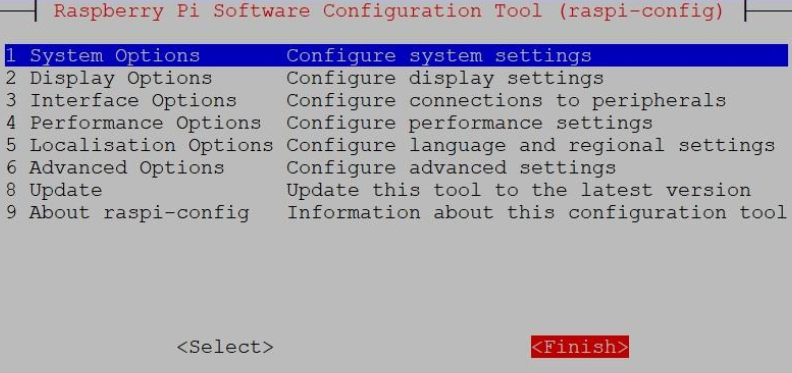
随后回到上级界面，调整vnc远程桌面的分辨率



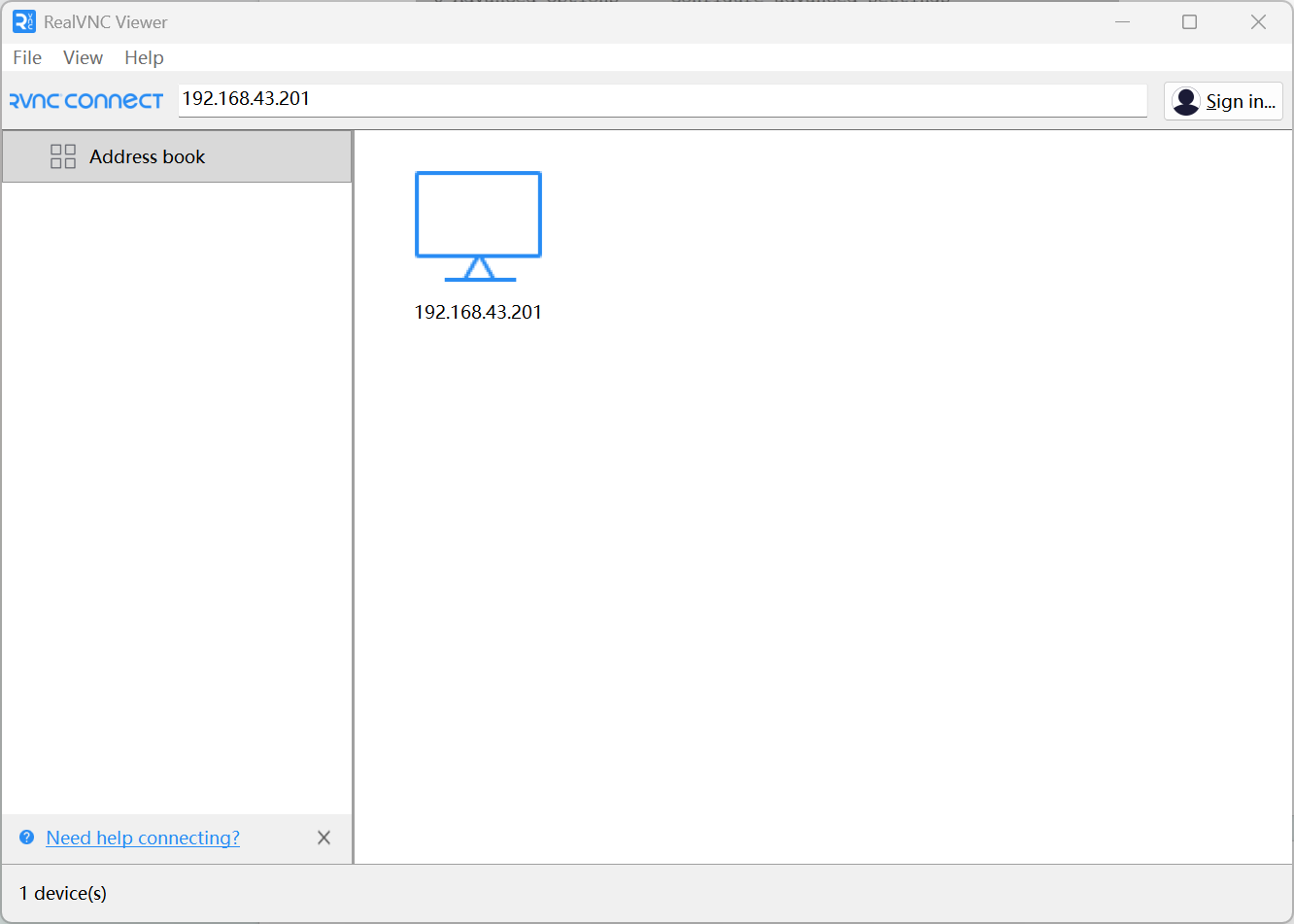
优先VNC resolution，分辨率自行选择。如果远程桌面打不开，可以回到这里修改分辨率。



修改完成后，从这里选择Finish，并重启。

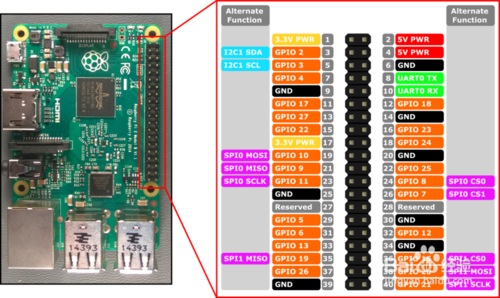


重启树莓派后重新建立连接（手机热点或其他方式），可以打开VNC进行远程连接。RealVNC Viewr下载地址：<https://www.realvnc.com/en/connect/download/vnc/>，双击打开运行即可，在下面输入树莓派的IP地址，连接continue即可，然后输入自己的user name以及密码，并勾选记住密码。



### 4.3 树莓派GPIO接口编程

树莓派接口图如下：

[](http://jingyan.baidu.com/album/0202781140cab61bcc9ce590.html?picindex=1)

从上到下，从左至右物理接口编号从1到40，其中，8个GND为接地线，另有4个电源接口，2个保留接口，其余26个为GPIO接口。

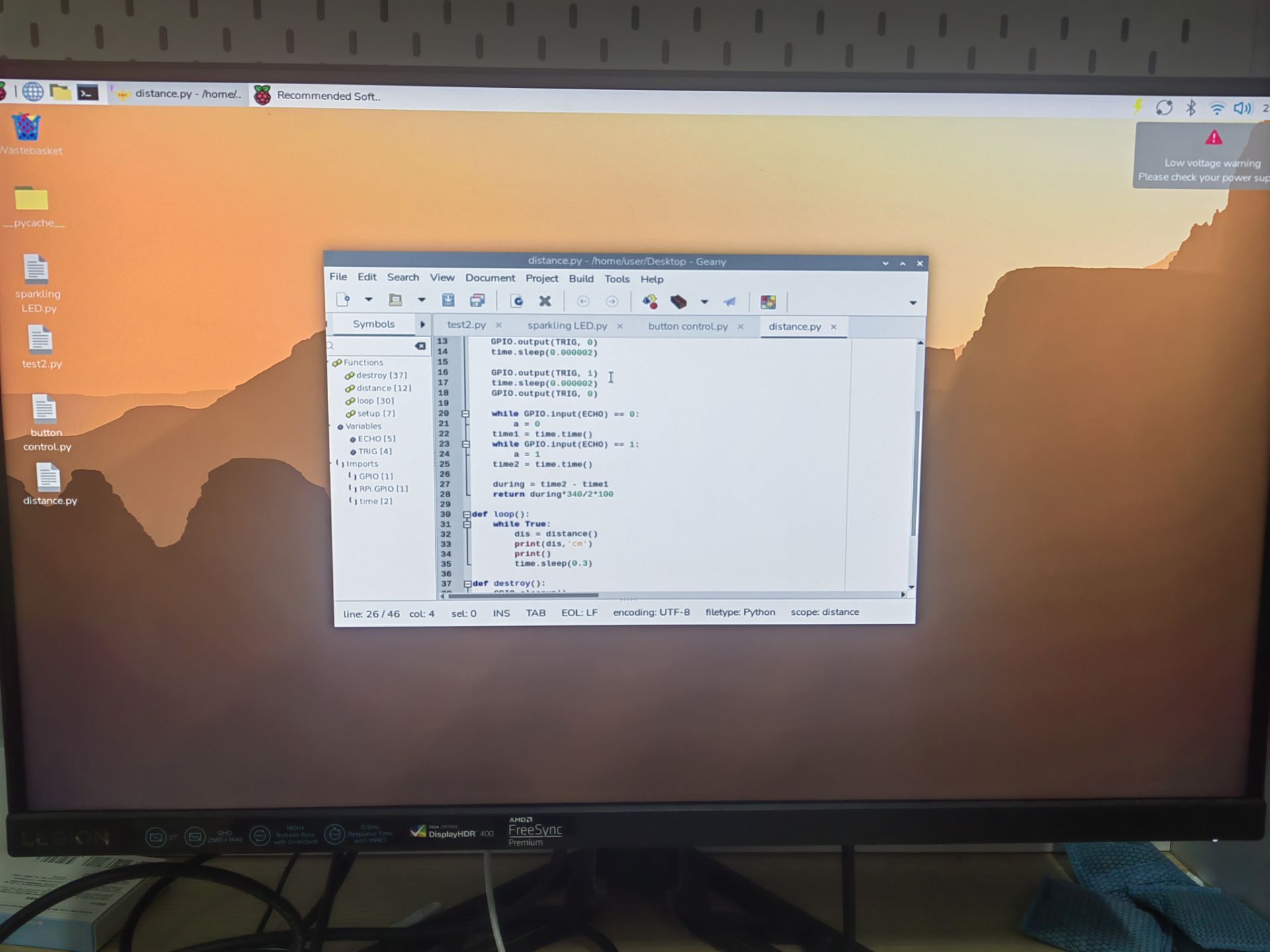
在实验开始前，先将T型板插到树莓派接口上，与MB-102面包板相接：





如上图所示T型板上标注了各个接口的作用，正确插入面包板中后，可以将设备直接与相应接口相连，也可插入面包板中间与T型板接口直接相连的接口，一横行5个插口同时与T型板、树莓派的一个物理接口相连。

随后按照前述方式与树莓派连接，显示器连接可插入鼠标和键盘开始编程。



#### 4.3.1 七色LED灯实验

在树莓派中创建后缀为.py的Python文件，代码如下：

import RPi.GPIO as GPIO

import time

GPIO.setmode(GPIO.BCM)

GPIO.setup(17,GPIO.OUT)

GPIO.setup(18.GPIO.OUT)

GPIO.setup(27.GPIO.OUT)

t=0.5

def loop():

while True:

GPIO.output(17,GPIO.HIGH)

time.sleep(t)

GPIO.output(27,GPIO.HIGH)

time.sleep(t)

GPIO.output(17,GPIO.LOW)

time.sleep(t)

GPIO.output(18,GPIO.HIGH)

time.sleep(t)

GPIO.output(27.GPIO.LOW)

time.sleep(t)

GPIO.output(17,GPIO.HIGH)

time.sleep(t)

GPIO.output(18,GPIO.LOW)

GPIO.output(17,GPIO.LOW)

time.sleep(t)

def destroy():

GPIO.cleanup()

if \_\_name\_\_ == '\_\_main\_\_':

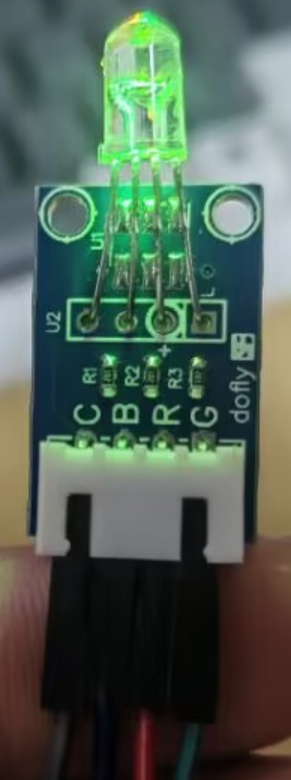
try:

loop()

except KeyboardInterrupt:

destroy()

其中17,18,27代表的是GPIO17,GPIO18和GPIO27接口，即物理接口11,12,13.将LED灯的四引脚中C连接到任一GND引脚，RGB三引脚分别连接到GPIO17,GPIO18和GPIO27接口。编译并运行该程序，程序会不断循环，LED灯会交替闪烁，间隔为t=0.5s。



#### 4.3.2 轻触按键控制 LED 灯实验

Python代码如下：

#!/usr/bin/env python

import RPi.GPIO as GPIO

import time

# 引脚定义（物理引脚编号）

BtnPin = 11 # 物理引脚11

Gpin = 13 # 物理引脚13

Rpin = 12 # 物理引脚12

def setup():

"""初始化设置"""

GPIO.setmode(GPIO.BOARD)

GPIO.setup(Gpin, GPIO.OUT)

GPIO.setup(Rpin, GPIO.OUT)

# 配置按钮引脚，启用内部上拉电阻

GPIO.setup(BtnPin, GPIO.IN, pull\_up\_down=GPIO.PUD\_UP)

# 添加事件检测，双边沿触发

GPIO.add\_event\_detect(BtnPin, GPIO.BOTH, callback=detect, bouncetime=200)

# 初始状态：绿灯亮

GPIO.output(Gpin, GPIO.LOW) # 绿灯亮

GPIO.output(Rpin, GPIO.HIGH) # 红灯灭

def Led(state):

"""根据按钮状态控制LED"""

if state == 0: # 按钮按下

GPIO.output(Rpin, GPIO.LOW) # 红灯亮

GPIO.output(Gpin, GPIO.HIGH) # 绿灯灭

print("按钮按下 - 红灯亮")

else: # 按钮释放

GPIO.output(Rpin, GPIO.HIGH) # 红灯灭

GPIO.output(Gpin, GPIO.LOW) # 绿灯亮

print("按钮释放 - 绿灯亮")

def Print(x):

if x==0:

print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

print(' Button Pressed! ')

print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

def detect(chn):

"""按钮事件回调函数"""

Led(GPIO.input(BtnPin))

Print(GPIO.input(BtnPin))

def loop():

"""主循环"""

while True:

pass

def destroy():

"""清理函数"""

GPIO.output(Gpin, GPIO.HIGH)

GPIO.output(Rpin, GPIO.HIGH)

GPIO.cleanup()

if \_\_name\_\_ == '\_\_main\_\_':

setup()

try:

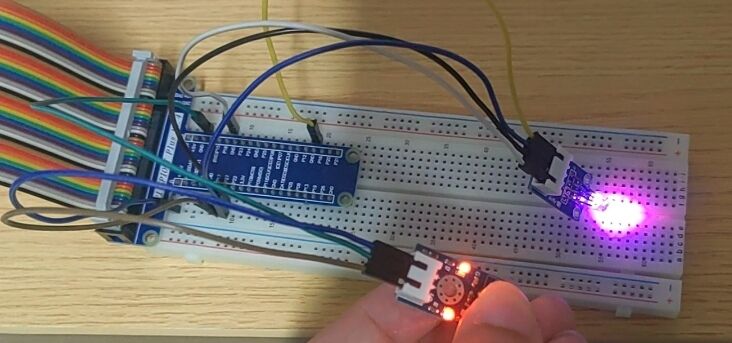
loop()

except KeyboardInterrupt:

destroy()

此程序中预先定义了引脚BtnPin、GPin、RPin物理引脚11、13、12，即GPIO17,GPIO27和GPIO18，将按钮+极插入物理接口1或17的3.3V电压上，-极插入任一GND接口中，S极插入物理引脚11即GPIO17中；

再将LED的C连接到任一GND引脚，G三引脚连接到GPIO27接口，将R引脚连接到GPIO18接口（也可自行调节RGB三引脚在GPIO27和GPIO18上的分布实现不同效果）（如下图将红蓝同时接入GPIO18接口发出紫光）。



运行结果： 初始状态绿灯亮，按下按钮时红灯亮绿灯灭，释放按钮时恢复绿灯亮红灯灭。

#### 4.3.3 超声波测距模块实验

Python代码如下：

import RPi.GPIO as GPIO

import time

TRIG = 11

ECHO = 12

def setup():

GPIO.setmode(GPIO.BOARD)

GPIO.setup(TRIG, GPIO.OUT)

GPIO.setup(ECHO, GPIO.IN)

def distance():

GPIO.output(TRIG, 0)

time.sleep(0.000002)

GPIO.output(TRIG, 1)

time.sleep(0.000002)

GPIO.output(TRIG, 0)

while GPIO.input(ECHO) == 0:

a = 0

time1 = time.time()

while GPIO.input(ECHO) == 1:

a = 1

time2 = time.time()

during = time2 - time1

return during \* 340 / 2 \* 100

def loop():

while True:

dis = distance()

print(dis, 'cm')

print()

time.sleep(0.3)

def destroy():

GPIO.cleanup()

if \_\_name\_\_ == '\_\_main\_\_':

setup()

try:

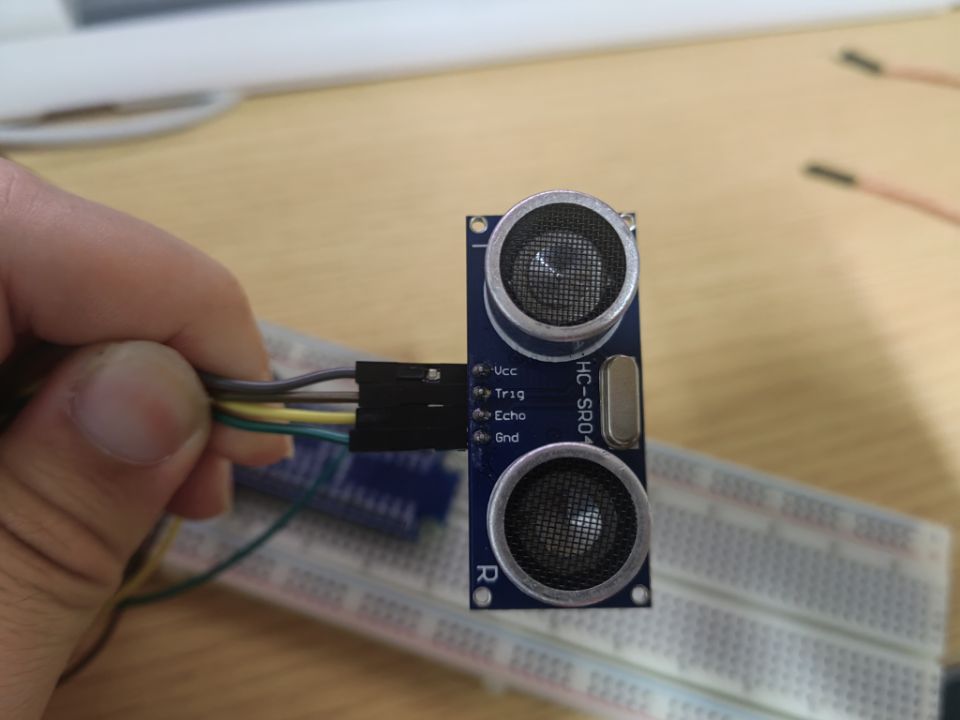
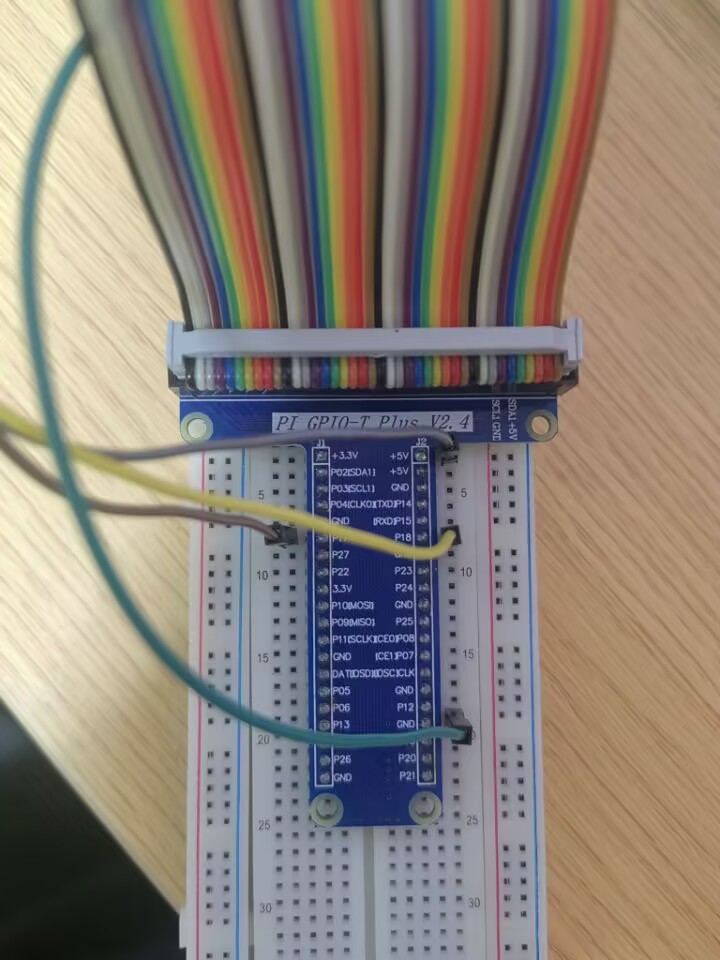
loop()

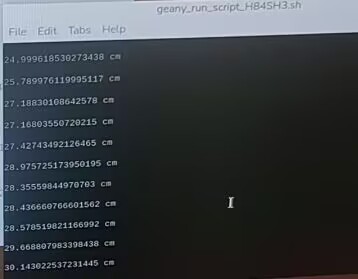
except KeyboardInterrupt:

destroy()

此程序中定义TRIG与ECHO物理引脚11、12，即GPIO17和GPIO18，将超声波测距仪HC-SR04的TRIG与ECHO两引脚分别接入，VCC引脚接+5V电源，GND接任意GND地线即可。

运行结果：每隔0.3秒输出一次测距结果于屏幕上，典型误差±0.3cm（近距离），±1cm（远距离），有效2-400cm。



## ****问题汇总****

### 5.1 树莓派远程ssh失败access denied

**原因：**

树莓派官方在2022年4月7日进行了一次系统更新，**这次更新删除了默认账户pi。**原文链接：<https://www.raspberrypi.com/news/raspberry-pi-bullseye-update-april-2022/>

而**旧的方法（当前不适用！）:**

1.把烧录好的sd卡插入电脑；

2.配置ssh文件和wpa\_supplicant.conf文件，复制到sd卡中；

3.将sd卡插入树莓派，连接电源启动树莓派；

4.查到树莓派ip地址，用远程ssh工具putty打开，输入ip地址。

这就是为什么我们按照网上的教程，配置SSH文件和wpa\_supplicant.conf文件，树莓派仍然无法联网的原因。

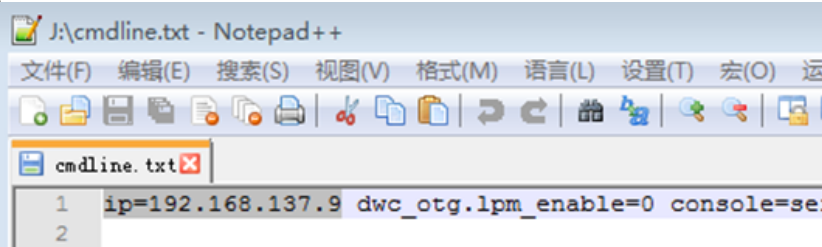
**解决方案：**

最主要的问题是不能用第三方软件进行烧录，用官方的烧录软件Rasberry Pi Imager，可以无需配置ssh文件和wpa\_supplicant.conf文件

### 5.2 树莓派未能自动连接到WIFI热点

目前原因不明，曾尝试修改修改局域网地址、调试手机热点等方式均无法解决，可使用备用方案：**手动设置IP并用光纤网线连接。**

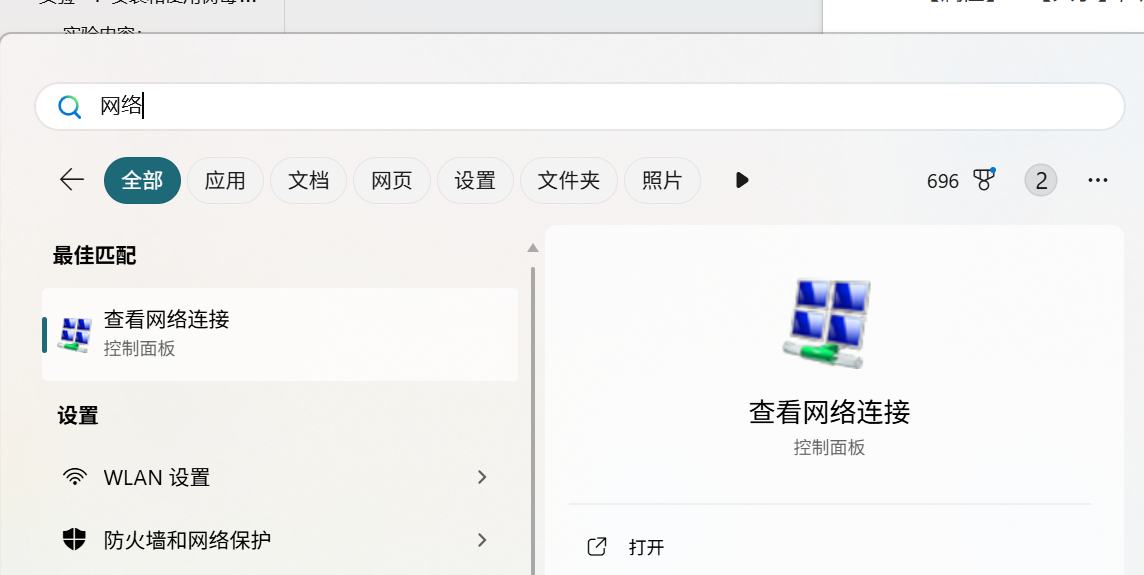
**步骤一：**将SD卡通过SD卡读卡器插入电脑，打开SD卡中的cmdline.txt文件，在其中写入ip=192.168.137.9（任一本局域网中合理，不被占用的地址）



此处可预先下载Advanced IP Scanner（<https://www.advanced-ip-scanner.com/cn/download/>），在软件中查询IP地址找到空置IP，再写入。

**步骤二：**修改笔记本电脑中的本地局域网地址

打开笔记本中的【网络和共享中心】->【更改适配器设置】，右键单击【本地连接】->【属性】。



此处，WIN10以上可搜索网络连接，进入后查看【以太网】属性



打开【共享】，然后勾选这两个选项。

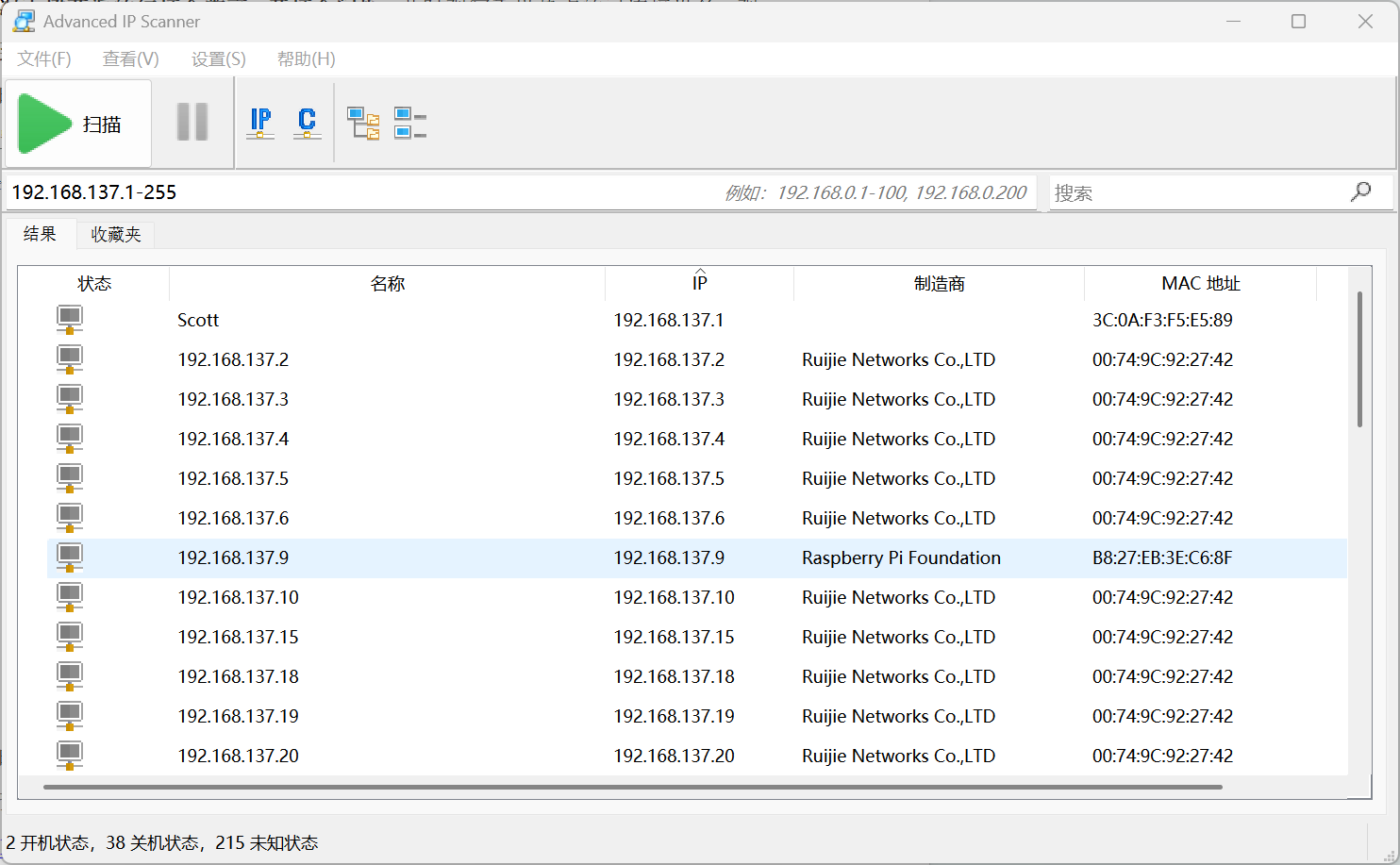


再选择【网络】->【TCP/IPv4】->【属性】，设置ip地址为192.168.137.1（与步骤一中写的树莓派IP地址位于同一局域网内（前三段相同）） 。子网掩码会自动填充或手动输入255.255.255.0



如图，单击【确定】退出。

**步骤三：给树莓派通上电**，使用光纤网线连接树莓派与笔记本电脑，在Advanced IP Scanner中查询192.168.137.1-255范围内的IP（或仅查询192.168.137.9即可），找到树莓派对应的设备，左侧显示屏图标表示当前状态，亮起则说明已成功连接，灰暗则说明连接已中断，没有这一表项说明未能成功连接过。



若果操作一切正常且树莓派本身无硬件问题，那么应该能成功连接，可跳转至[4.2.2](#_4.2.2 远程连接笔记本电脑（较为复杂）)小节继续连接树莓派进行操作。

### 5.3 上述步骤二中笔记本电脑网络共享无法打开

原因：笔记本电脑中安装虚拟机占用了该局域网段

解决方案：

**方案一**（**推荐**）：**可尝试从步骤一开始更换其他局域网**（与先前的192.168.137不同）

方案二：修改虚拟机的网络配置

参考资料如下：<https://blog.csdn.net/qq_43546676/article/details/126834953>