河南工业大学

《嵌入式系统》实验报告

专业班级：物联网 1603　　学号：201616070320　　姓名：郭治洪

实验一 嵌入式系统安装和使用

【实验目的】

1. 学会安装树莓派的操作系统 Raspbian
2. 学会开启 SSH 和 VNC 远程连接

【实验步骤】

1. 去官网下载 Raspbian 系统烧入 TF 卡（无截图）
2. 在 TF 卡的 Boot 分区新建 ssh.txt 或 ssh 文件
3. 打开虚拟机将 TF 读卡器连接虚拟机使用终端挂载系统分区/system，命令

 # root@localhost mkdir /root/pi

 # root@localhost fdisk -l

Device Boot Start End Sectors Size Id Type

/dev/sdb0 8192 97890 89699 43.8M c W95 FAT32 (LBA)

/dev/sdb1 98304 62333951 62235648 29.7G 83 Linux

 # root@localhost mount /dev/sdb1 /root/pi

 # root@localhost vim

/root/pi/etc/etc/wpa_supplicant/wpa_supplicant.conf

然后将笔记本热点的 WIFI 信息添加进去，效果是这样：

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=CN
```

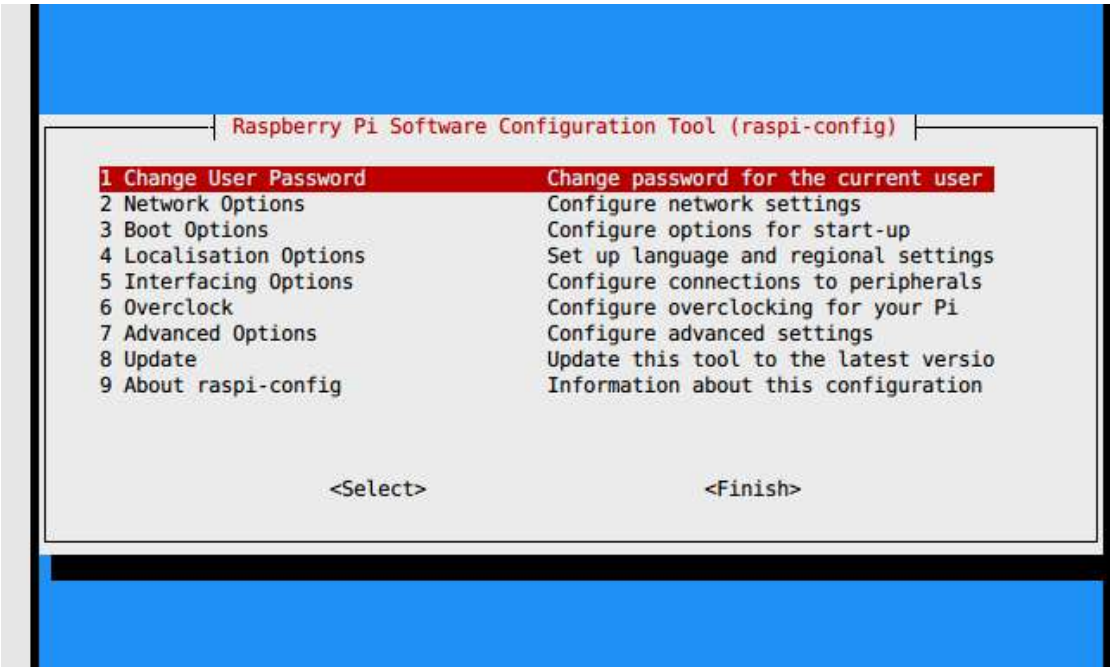
```
network={
        ssid="MR.Guo"
        psk="66786666"
}

network={
        ssid="Mr.Guo"
        psk="66786666"
        key_mgmt=WPA-PSK
}
```

而后:wq!退出并且卸载分区，将 TF 卡插入树莓派启动

后将会在热点管理器中看到树莓派的 ip 使用 SSH 登录

进入树莓派的 Bios 开启 VNC

 # pi@raspbian sudo raspi-config

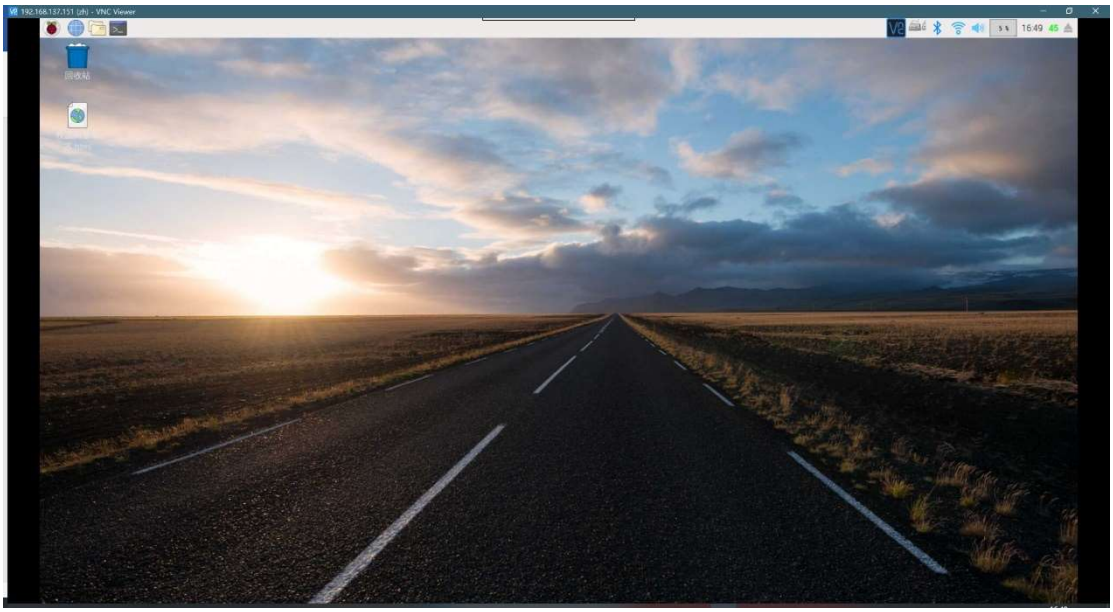


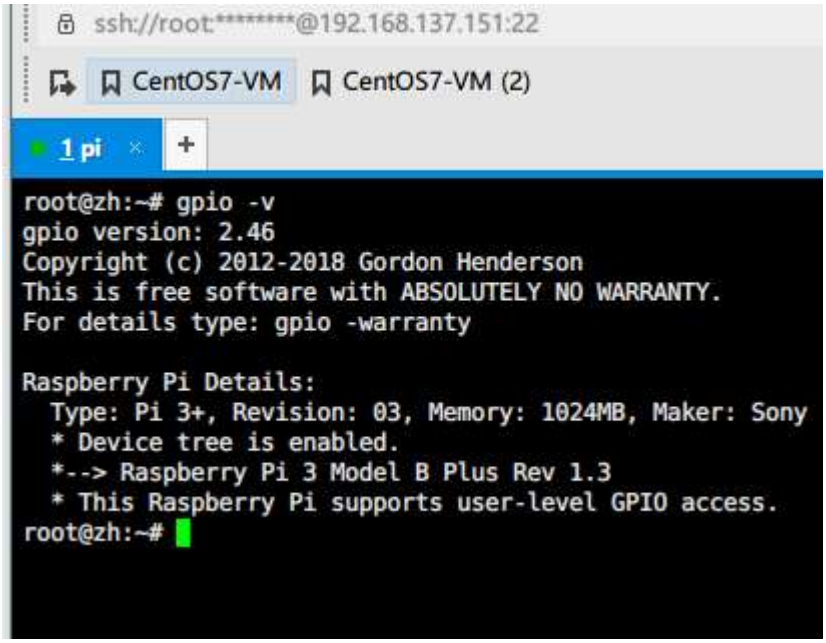
使用 VNC VIEWER 登入

个人为了使用方便更改了账户密码/root 密码/开启了 root 登录

同时下载并安装了 **wiringPi** 并且安装（无截图）

【实验结果】





【实验心得】

做这个实验做的比较早，当时汪洋同学迫不及待想让我一起灌入系统然后玩，结果当时灌完系统就断电了，显示器没法使用，我就上网查如何使用无显示起连接树莓派，网上给出了教程。结果系统分区 **Ext4** 在 **Windows** 下不是很好挂载，机制的我想到我有虚拟机连接读卡器使用 **Linux** 挂载，然后我就照做了，结果最后成功了，感觉很高兴！

实验二 GPIO 输入/输出设定

【实验目的】

1. 下载并且安装 wiringPi 并且检测是否安装正常
2. 学会书写简要的 GPIO 的程序并且编译测试

【实验步骤】

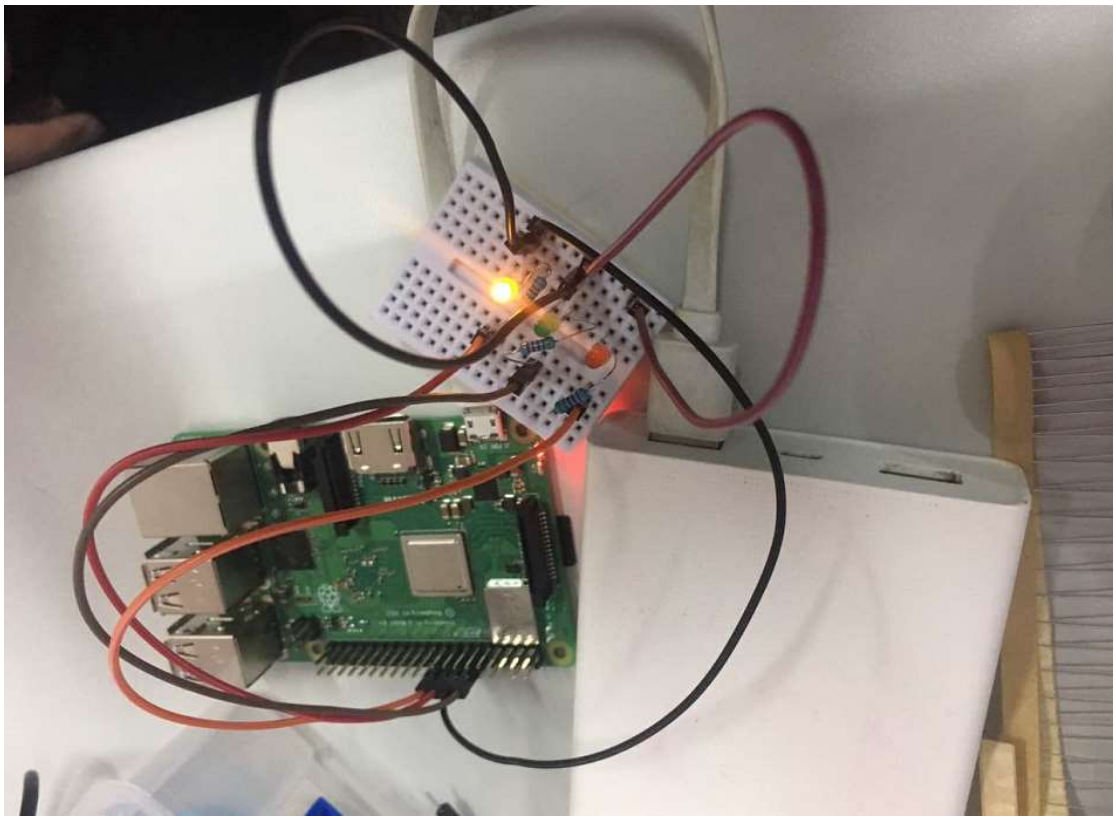
1. 断电情况下按照实验指导书连接导线
2. 书写程序，并且使用 g++编译

```
//g++ traffic.cpp -o traffic -l wiringPi
//嵌入式实验
//物联网1603 郭治洪 201616070320
#include <iostream>
#include <wiringPi.h>

using namespace std;

//12 GPIO 1 绿
//14 GPIO 4 黄
//16 GPIO 5 红
const int red=5; //GPIO 1 RED
const int yellow=4; //GPIO 4 YELLOW
const int green=1; //GPIO 5 GREEN

int main(int argc, char const *argv[])
{
    if(wiringPiSetup()==-1)
        return 0;
    pinMode(red,OUTPUT); //output
    pinMode(yellow,OUTPUT);
    pinMode(green,OUTPUT);
    while(1)
    {
        //绿灯 5s
        digitalWrite(red,0);
        digitalWrite(yellow,0);
        digitalWrite(green,1);
        delay(5000);
        //黄灯 闪烁4次 间隔0.5s
        digitalWrite(green,0);
        for(int i=0;i<4;i++) {
```

```
            digitalWrite(yellow,1);
            delay(500);
            digitalWrite(yellow,0);
            delay(500);
        }
        //红灯 5s
        digitalWrite(red,1);
        digitalWrite(yellow,0);
        digitalWrite(green,0);
        delay(5000);
    }
    return 1;
}
```

【实验结果】

三个灯交叉亮，按照设定的时间变化或者闪烁。



【实验心得】

这个实验，起初我没有搞懂面包板的原理然后连错了，后来发现问题后我重新连接了一遍，但是灯没有照着老师给的模式连。不过我想到硬件都是死的，软件是活的，我可以直接改引脚和代码做到同样的效果，这就是变通，因此我觉得我还是学的不少呢。

实验三 LED 数码管的驱动和控制实验（PWM）

【实验目的】

1. 了解 PWM 工作原理
2. 了解软件 PWM 的工作原理

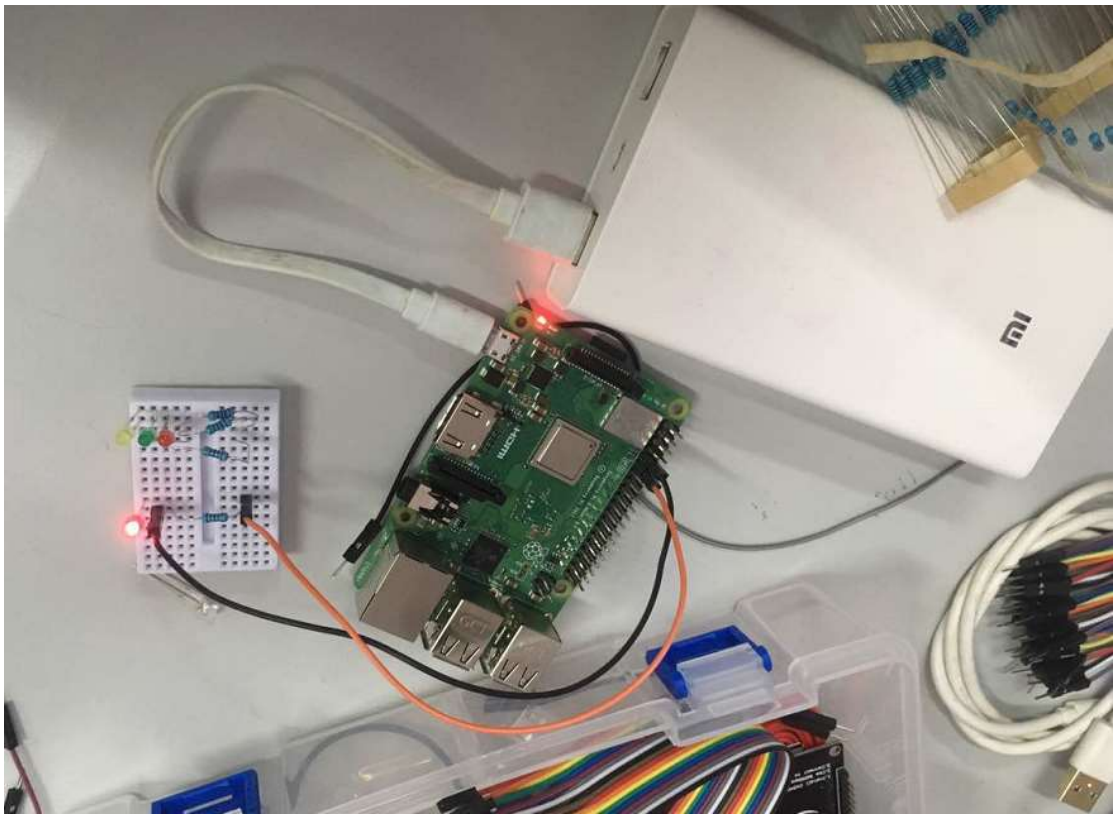
【实验步骤】

1. 断电情况下按照实验指导书连接导线
2. 书写程序，并且使用 g++编译

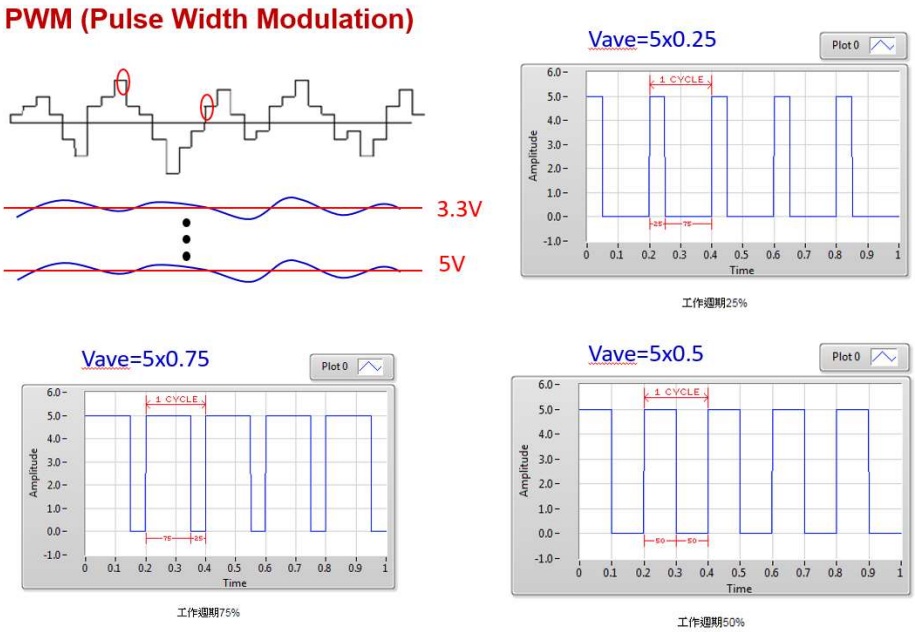
```
//g++ pwnLed.cpp -o pwnled -l wiringPi
//嵌入式实验
//物联网1603 郭治洪 201616070320
#include <iostream>
#include <wiringPi.h>
#include <softPwm.h>
using namespace std;
const int led =0; //GPIO0
int main(int argc, char const *argv[])
{

    if(wiringPiSetup()==-1)
        return 0;
    softPwmCreate(led,0,100);
    while(1)
    {
        int val;
        for(val=0;val<=100;val++)
        {
            softPwmWrite(led,val);
            delay(20);
        }
         for(val=100;val>0;val--)
        {
            softPwmWrite(led,val);
            delay(20);
        }
    }
    return 1;
}
```

【实验结果】

灯的亮度不断改变，到达最亮值或者最暗值后然后进行反向。



【实验心得】



http://www.electronicwings.com/raspberry-pi/raspberry-pi-pwm-generation-using-python-and-c

PWM 是利用元件对电压不敏感特性，改变每个周期的电压占空比实现。软件 PWM 使用软件完成这样的动作，而其中的延时工程师有经验的人根据系统时钟控制来计算出，使之体现出最好的效果，另外不建议分的太细或太粗，否则效果不好。

实验四 定时器（或 A/D 转换器）的 I/O 界面的使用

【实验目的】

1. 了解光敏电阻，A/D 转换器的原理
2. 了解 SPI 连接协议的工作原理

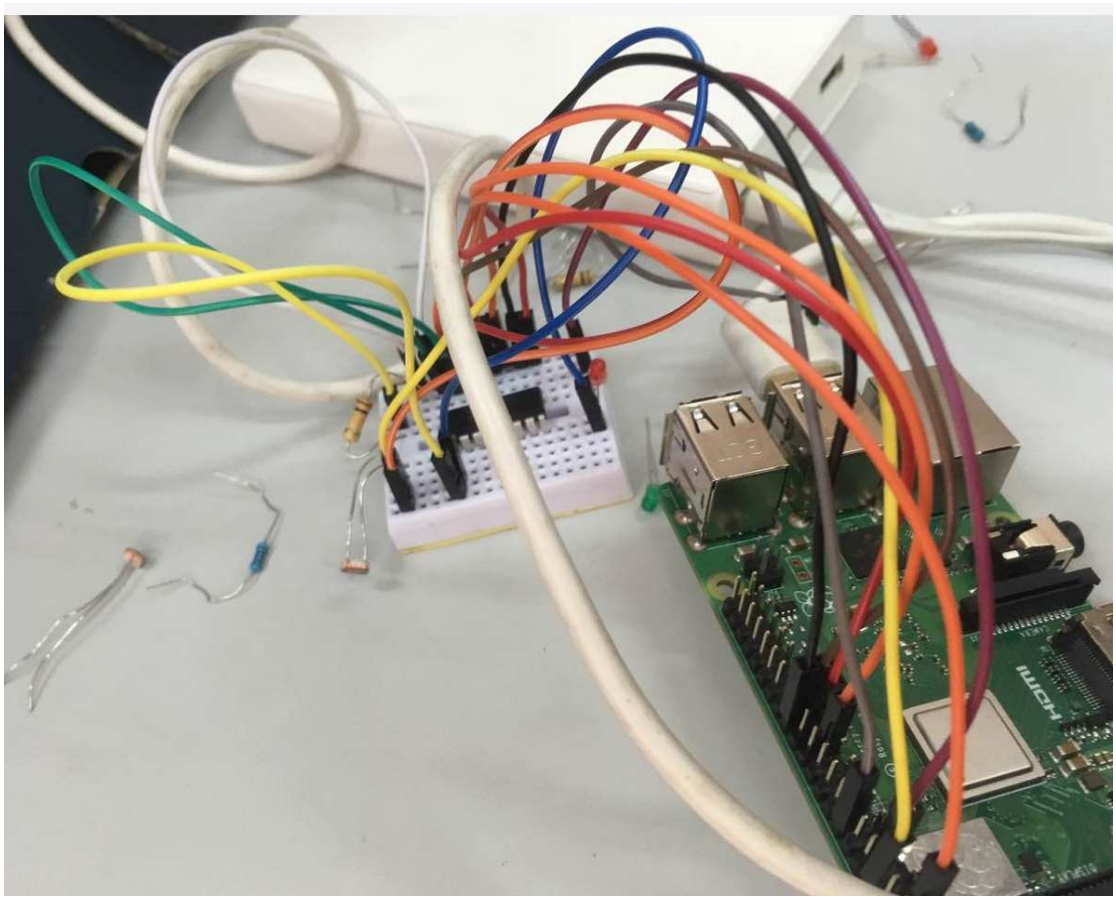
【实验步骤】

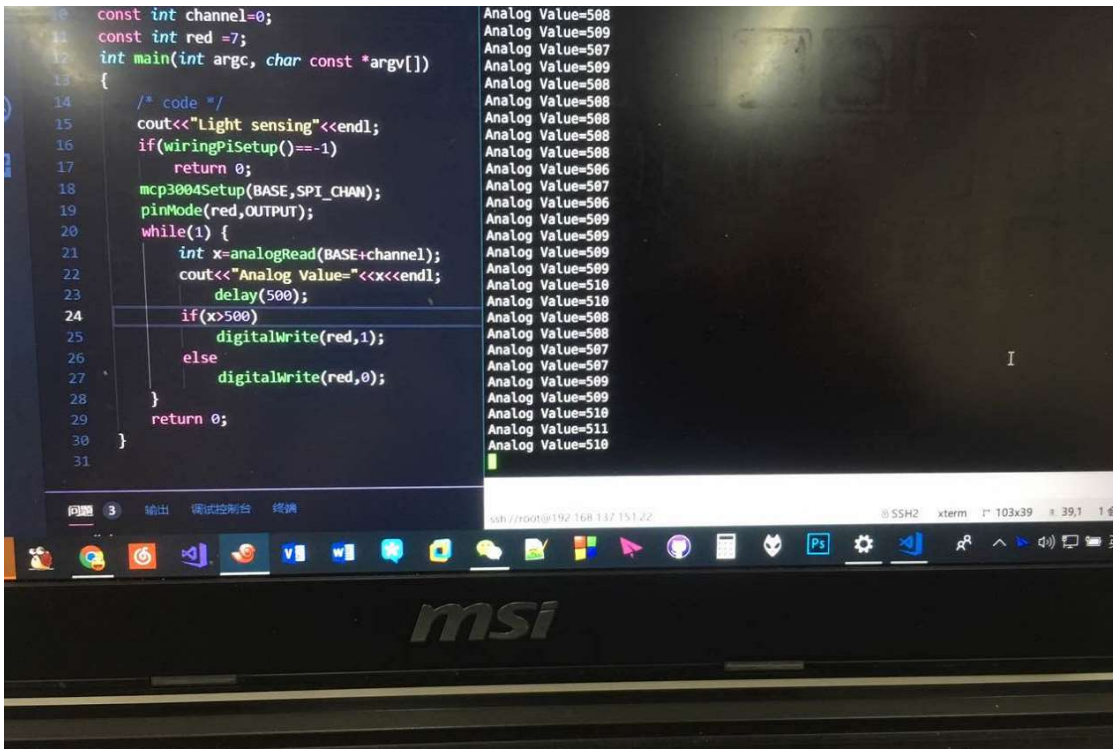
1. 断电情况下按照实验指导书连接导线
2. 书写程序，并且使用 g++编译

```
//g++ light.cpp -o light -l wiringPi
//嵌入式实验
//物联网1603 郭治洪 201616070320
#include <iostream>
#include <wiringPi.h>
#include <mcp3004.h>

using namespace std;

#define BASE 200
#define SPI_CHAN 0

const int channel=0;
const int red =7;
int main(int argc, char const *argv[])
{
    /* code */
    cout<<"Light sensing"<<endl;
    if(wiringPiSetup()==-1)
        return 0;
    mcp3004Setup(BASE,SPI_CHAN);
    pinMode(red,OUTPUT);
    while(1) {
        int x=analogRead(BASE+channel);
        cout<<"Analog Value="<<x<<endl;
            delay(500);
        if(x>500)
            digitalWrite(red,1);
        else
            digitalWrite(red,0);
    }
    return 0;
}
```

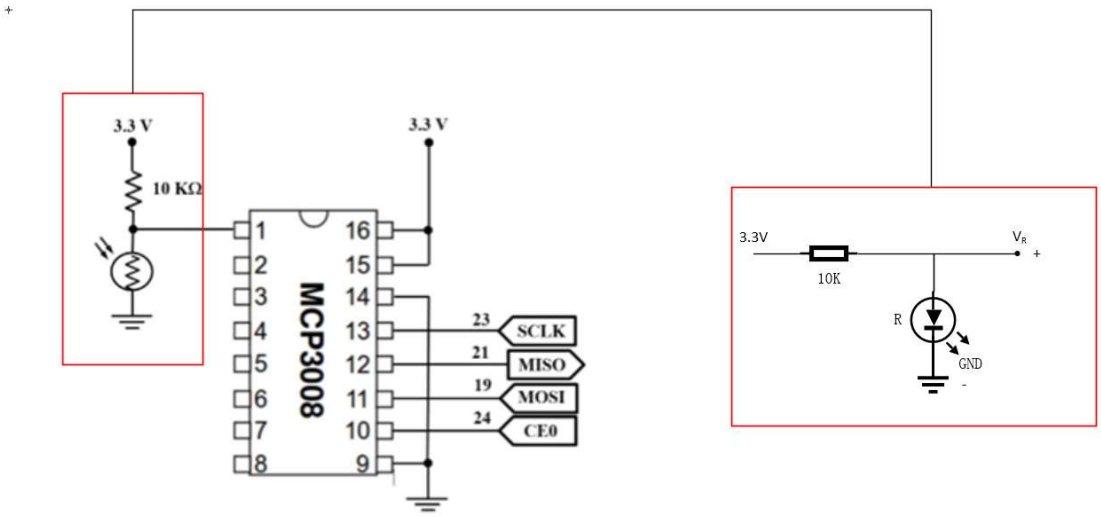
【实验结果】

光越强测得值越小，灯越暗(或者不亮)，光越弱值越大，灯越亮。





【实验心得】



MCP3008 其实是一个 A/D 转换器，用来将模拟量转成数字量，通过 SPI 协议与树莓派通信完成测量动作，其中测量电路红框可以转成右边电路简图。下面是分析过程：

$$\mathbf{R_T = 10K + R}$$

$$\mathbf{I = \frac{3.3V}{10K + R}}$$

$$\mathbf{V_R = \frac{3.3V}{10K + R} * R = 3.3V - 10K * I = 3.3V - \frac{3.3V * 10K}{10K + R}}$$

光敏电阻光线越暗电阻越大，光线越亮电阻越小。

光线暗，R 增大，后一项减少，则测得值$\mathbf{V_R}$增大，指示 LED 灯越亮；

光线亮，R 减少，后一项增大，则测得值$\mathbf{V_R}$减少，指示 LED 灯越暗。

这里老师取得有基值，阈值，但是在我们的实验中阈值调小效果比较好。另外我们的指示 LED 效果并不是很好，我尝试换了一个 GPIO 接口显示略有好转，但是仍未达到十分明显的效果，不过测量电路没有问题，经过老师的讲解我也搞懂了电路连接，了解了如何利用 A/D 转换器，SPI 协议进行数据的传输，感谢老师讲解的这么深。

实验五 嵌入式 Linux 下遍程（Shell/编译）

【实验目的】

1. 书写一个简单的 Shell 脚本并且尝试运行
2. 书写一个简单的 C 程序并且尝试编译运行

【实验步骤】

1. 书写一个简单的 Shell 脚本并且尝试运行

```
#!/bin/bash
COUNTER=0
#GPIO 0
echo 0 > /sys/class/gpio/export
sleep 1

echo high > /sys/class/gpio/gpio0/direction
sleep 1

while [ $COUNTER -lt 10 ]; do
 echo 0 > /sys/class/gpio/gpio0/value
 sleep 1
 echo 1 > /sys/class/gpio/gpio0/value
 sleep 1
 echo The counter is $COUNTER
 let COUNTER=COUNTER+1
done
echo 0 > /sys/class/gpio/unexport
```

这个因为没有驱动所以只能计数运行，无法在硬件显示。

2. 书写一个简单的 C 程序并且尝试运行

```
//gcc px.cpp -o px
//嵌入式实验
//物联网1603 郭治洪 201616070320
#include <stdio.h>
int main()
{
 int number[10]={6,2,3,3,1,5,4,0,-1,-2};
 int i=0,j=0;
 int temp=0;
 for(i=0;i<10;i++)
 {
   for(j=i;j<10;j++)
   if(number[j]<number[i])
```

```
    {
      temp=number[j];
      number[j]=number[i];
      number[i]=temp;
    }
  }
  for(i=0;i<10;i++)
  {
    printf("%d\n",number[i]);
  }
  return 0;
}
```

【实验结果】

1. 实验内容 1

```
root@zh:~/src# ./shell.sh
The counter is 0
The counter is 1
The counter is 2
The counter is 3
The counter is 4
The counter is 5
The counter is 6
The counter is 7
The counter is 8
The counter is 9
root@zh:~/src#
```

2. 实验内容 2

```
root@zh:~/src# gcc px.c -o px
root@zh:~/src# ./px
-2
-1
0
1
2
3
3
4
5
6
root@zh:~/src#
```

【实验心得】

这个没什么难度啦。但是老师说 Shell 脚本没有驱动无法使用硬件，这点很不开心，而且这些程序都十分简单，还没有深入，我现在经历也不够，我一定会在闲下来深入下去研究。

实验六 嵌入式 Linux 下遍程（Hello/中断）

【实验目的】

1. 了解中断的原理以及实现

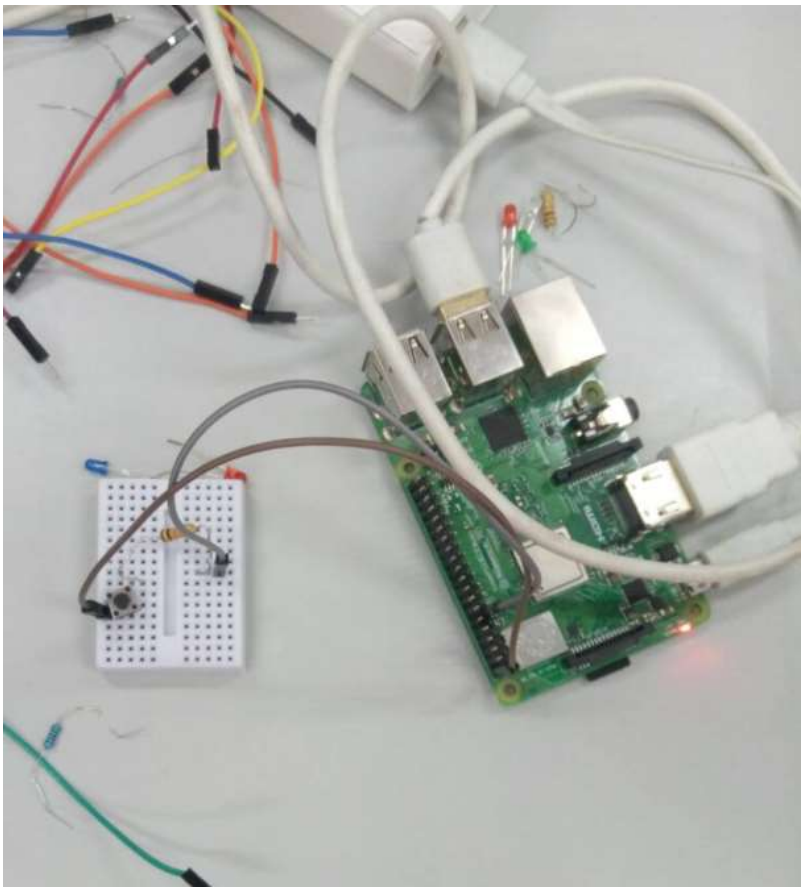
【实验步骤】

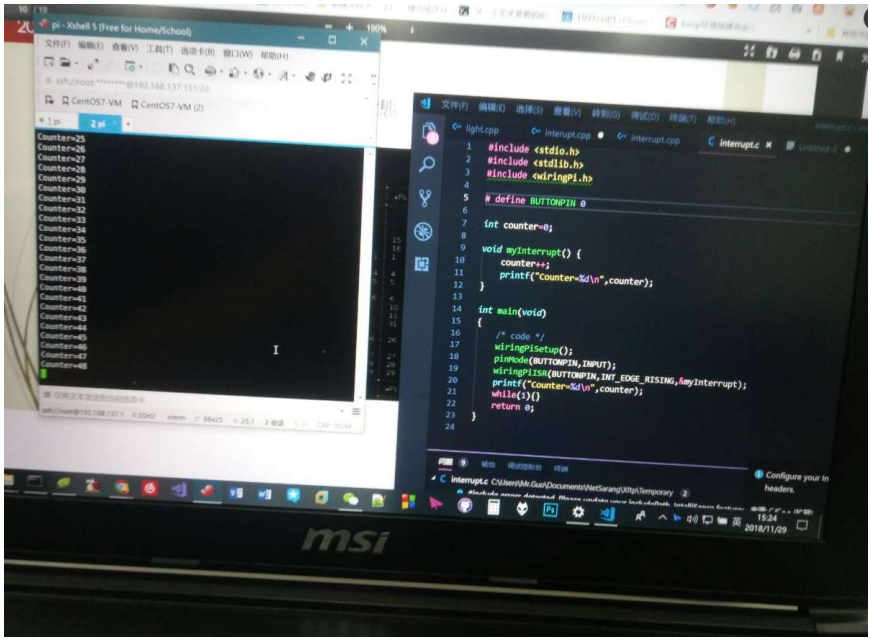
1. 断电情况下按照实验指导书连接导线
2. 书写程序，并且使用 gcc 编译

```
//gcc interrupt.cpp -o interrupt
//嵌入式实验
//物联网1603 郭治洪 201616070320
#include <stdio.h>
#include <stdlib.h>
#include <wiringPi.h>

# define BUTTONPIN 0

int counter=0;

void myInterrupt() {
    counter++;
    printf("Counter=%d\n",counter);
}

int main(void)
{
    /* code */
    wiringPiSetup();
    pinMode(BUTTONPIN,INPUT);
    wiringPiISR(BUTTONPIN,INT_EDGE_RISING,&myInterrupt);
    //printf("Counter=%d\n",counter);
    while(1){}
    return 0;
}
```

【实验结果】

观察到起初计数为=0，当按钮按下在上升沿会产生一个中断，调用自定义中断处理函数处理，计数每次加一并且输出。我们发现主程序的输出是 Counter=0 初始情况，而自定义中断函数处理时每次会加一并且输出，然后中断完成后返回调用它的函数（主函数），这就是中断。





【实验心得】

通过这次实验，我又对中断的了解加深了一步。中断我们在单片机就学过，当时可能不太理解，但是通过嵌入式学习和实验更加深入了。感谢老师！您辛苦了！我的实验还有很多不足，请老师指正，有时间我一定会更深入的！