3.Read CPU temperature and adjust fan speed

RGB cooling HAT needs to be correctly inserted into the GPIO port of the Raspberry Pi and the I2C function of the Raspberry Pi must be turned on.

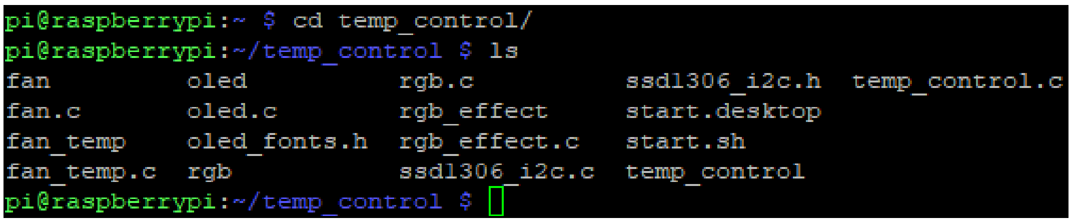
The phenomenon of this experiment is that reading and printing the Raspberry Pi CPU temperature, and adjusting the fan rotation speed according to the temperature. The higher the temperature, the faster the rotation speed.

1. Compile and run the program

1.1 Enter the folder and view the files in the current folder

```
cd temp_control/
```

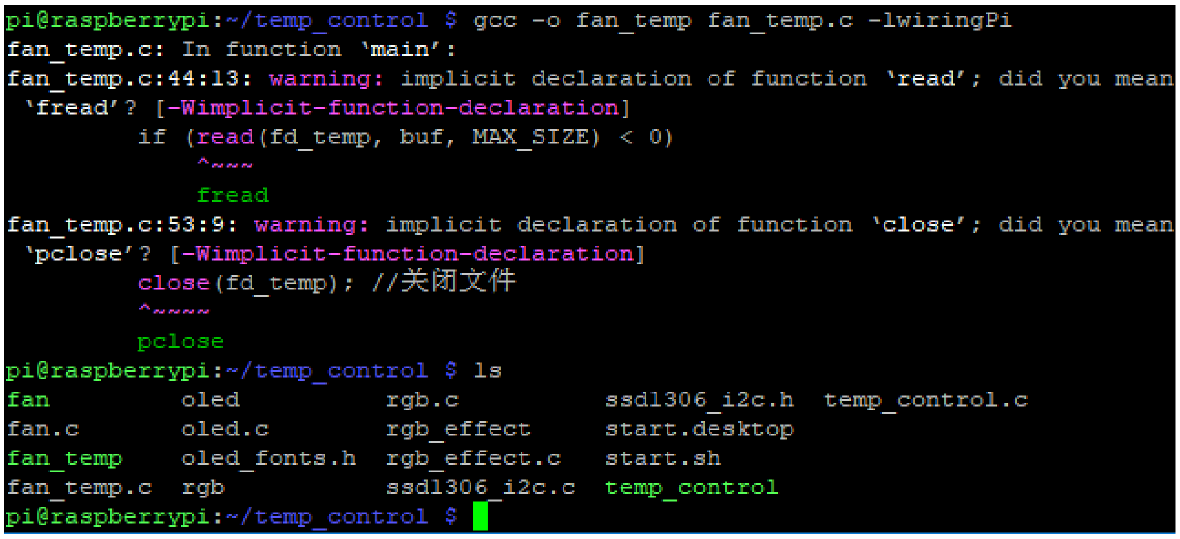
```
ls
```



```
pi@raspberrypi:~ $ cd temp_control/
pi@raspberrypi:~/temp_control $ ls
fan          oled           rgb.c          ssd1306_i2c.h  temp_control.c
fan.c        oled.c         rgb_effect     start.desktop
fan_temp     oled_fonts.h   rgb_effect.c   start.sh
fan_temp.c   rgb            ssd1306_i2c.c  temp_control
pi@raspberrypi:~/temp_control $ 
```

1.2 Compile program files

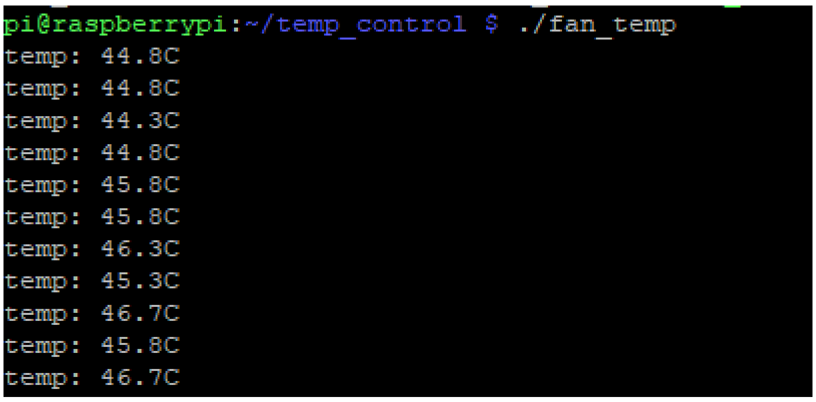
```
gcc -o fan_temp fan_temp.c -lwiringPi
```



```
pi@raspberrypi:~/temp_control $ gcc -o fan_temp fan_temp.c -lwiringPi
fan_temp.c: In function 'main':
fan_temp.c:44:13: warning: implicit declaration of function 'read'; did you mean
 'fread'? [-Wimplicit-function-declaration]
         if (read(fd_temp, buf, MAX_SIZE) < 0)
             ^~~~
             fread
fan_temp.c:53:9: warning: implicit declaration of function 'close'; did you mean
 'pclose'? [-Wimplicit-function-declaration]
         close(fd_temp); //关闭文件
         ^~~~~
         pclose
pi@raspberrypi:~/temp_control $ ls
fan          oled           rgb.c          ssd1306_i2c.h  temp_control.c
fan.c        oled.c         rgb_effect     start.desktop
fan_temp     oled_fonts.h   rgb_effect.c   start.sh
fan_temp.c   rgb            ssd1306_i2c.c  temp_control
pi@raspberrypi:~/temp_control $ 
```

1.3 Run program

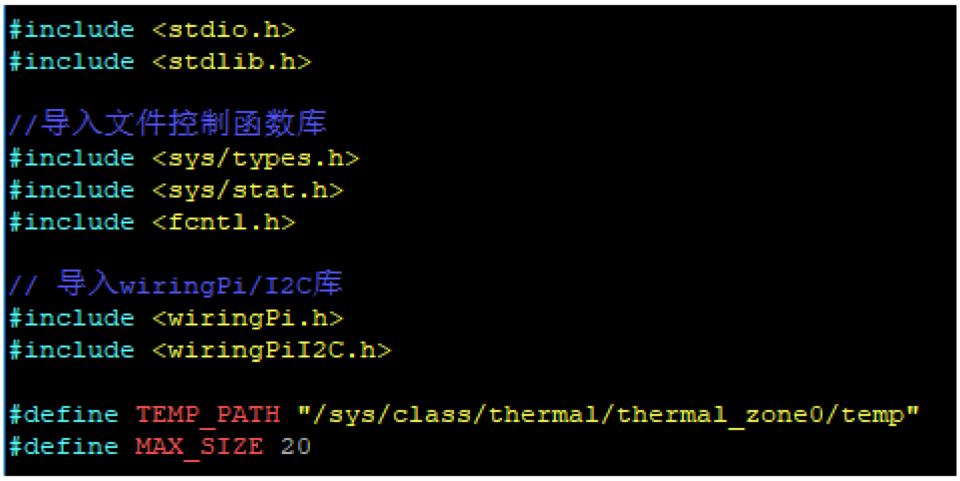
```
./fan_temp
```


```
pi@raspberrypi:~/temp_control $ ./fan_temp
temp: 44.8C
temp: 44.8C
temp: 44.3C
temp: 44.8C
temp: 45.8C
temp: 45.8C
temp: 46.3C
temp: 45.3C
temp: 46.7C
temp: 45.8C
temp: 46.7C
```

At this time, the terminal will print the current CPU temperature value. The higher the CPU temperature, the faster the fan speed.

2. Code analysis

2.1 First import the file control library and I2C library. The path for viewing the CPU temperature on the Raspberry Pi is defined as TEMP_PATH.

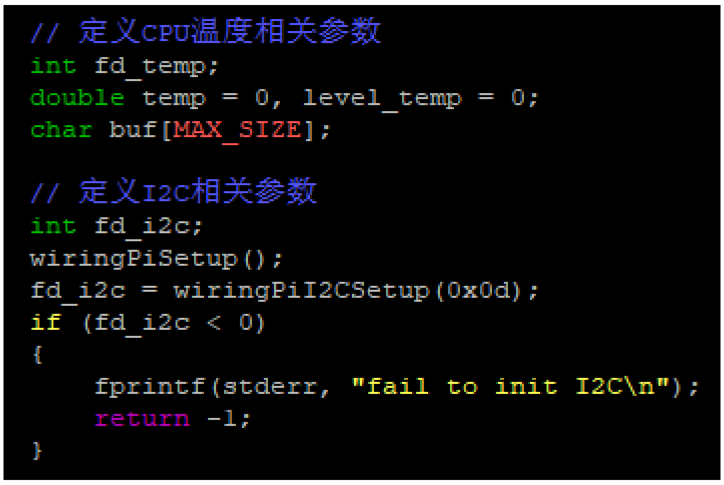

```
#include <stdio.h>
#include <stdlib.h>

//导入文件控制函数库
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

// 导入wiringPi/I2C库
#include <wiringPi.h>
#include <wiringPiI2C.h>

#define TEMP_PATH "/sys/class/thermal/thermal_zone0/temp"
#define MAX_SIZE 20
```

2.2 Define CPU temperature-related parameters and I2C-related parameters in the main function.

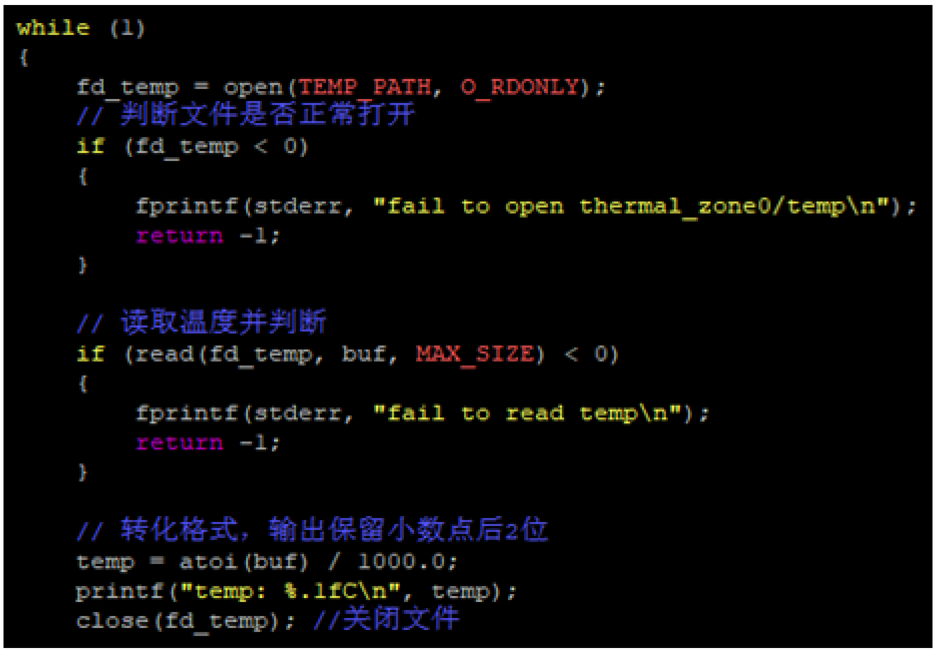

```
// 定义CPU温度相关参数
int fd_temp;
double temp = 0, level_temp = 0;
char buf[MAX_SIZE];

// 定义I2C相关参数
int fd_i2c;
wiringPiSetup();
fd_i2c = wiringPiI2CSetup(0x0d);
if (fd_i2c < 0)
{
    fprintf(stderr, "fail to init I2C\n");
    return -1;
}
```

2.3 Open the CPU temperature file in a loop and save fd_temp. If the opening fails, -1 is returned. Next, the temperature is read and saved in buf. If it fails, -1 is returned.

When the temperature is successfully read and saved to buf, since the value is relatively large, the current temperature is obtained by dividing it by 1000, in degrees Celsius, and is saved to temp.

Every time you finish reading a file, you must run the close() function to manually close the file.

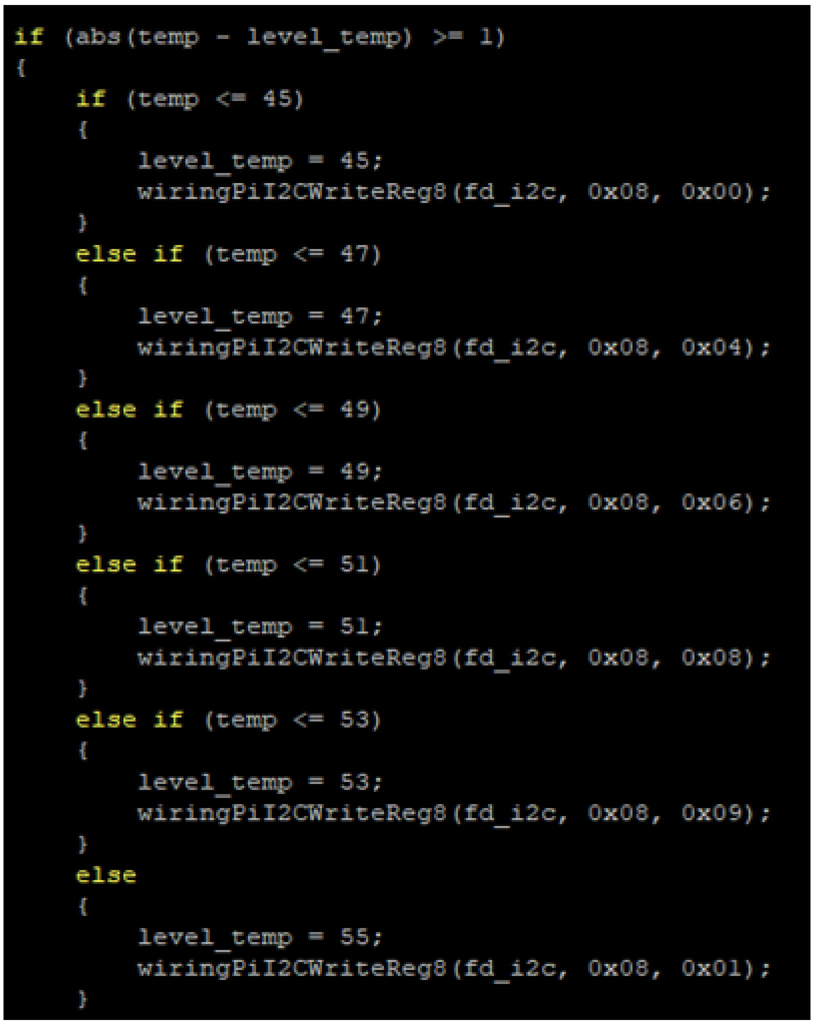


```
while (1)
{
    fd_temp = open(TEMP_PATH, O_RDONLY);
    // 判断文件是否正常打开
    if (fd_temp < 0)
    {
        fprintf(stderr, "fail to open thermal_zone0/temp\n");
        return -1;
    }

    // 读取温度并判断
    if (read(fd_temp, buf, MAX_SIZE) < 0)
    {
        fprintf(stderr, "fail to read temp\n");
        return -1;
    }

    // 转化格式，输出保留小数点后2位
    temp = atoi(buf) / 1000.0;
    printf("temp: %.1fC\n", temp);
    close(fd_temp); //关闭文件
```

2.4 After obtaining the temperature, determine the temperature value and modify the fan speed.



```
if (abs(temp - level_temp) >= 1)
{
    if (temp <= 45)
    {
        level_temp = 45;
        wiringPiI2CWriteReg8(fd_i2c, 0x08, 0x00);
    }
    else if (temp <= 47)
    {
        level_temp = 47;
        wiringPiI2CWriteReg8(fd_i2c, 0x08, 0x04);
    }
    else if (temp <= 49)
    {
        level_temp = 49;
        wiringPiI2CWriteReg8(fd_i2c, 0x08, 0x06);
    }
    else if (temp <= 51)
    {
        level_temp = 51;
        wiringPiI2CWriteReg8(fd_i2c, 0x08, 0x08);
    }
    else if (temp <= 53)
    {
        level_temp = 53;
        wiringPiI2CWriteReg8(fd_i2c, 0x08, 0x09);
    }
    else
    {
        level_temp = 55;
        wiringPiI2CWriteReg8(fd_i2c, 0x08, 0x01);
    }
```