**中国海洋大学计算机科学与技术学院**

实验报告

**姓名：惠欣宇 年级：2021级 专业：计算机科学与技术**

**科目：物联网系统设计与开发 题目：温度数据采集及控制交互编程**

**实验时间：2024年 5月 23日 实验教师：郭忠文**

**目录**

[一、实验目的及要求 2](#_Toc167486535)

[1. 实验目的 2](#_Toc167486536)

[2. 实验要求 2](#_Toc167486537)

[二、实验内容 2](#_Toc167486538)

[三、实验设备 2](#_Toc167486539)

[四、通信协议 3](#_Toc167486540)

[五、实验步骤 4](#_Toc167486541)

[六、报告要求 10](#_Toc167486542)

[1. 数据采集模块可连接的传感器数量是多少 10](#_Toc167486543)

[2. 控制模块控制风扇开启和关闭的原理是什么 11](#_Toc167486544)

[3. 把控制风扇开启的语句及文本文件的内容附在实验报告中 11](#_Toc167486545)

[4. 写出调试过程遇到的主要问题和解决方法 13](#_Toc167486546)

[七、实验总结 14](#_Toc167486547)

一、实验目的及要求

1. 实验目的

该实验为串口通信实验，通过串口连接进行温度数据的采集、处理，通过控制风扇运行，模拟调节温度场景，了解真实设备与计算机互联的方式，培养学生编程能力。通过编写数据采集及控制软件，培养利用串行通信接口获取传感数据以及对设备进行控制的编程能力，为物联网系统综合数据采集软件的设计与开发打下基础。

2. 实验要求

根据实验内容，预先编写数据采集和控制软件，实验过程中认真调试、思考和记录，掌握利用软件自动获取串行接口传感设备数据以及对设备的控制方法， 撰写实验报告。

二、实验内容

1. 编写数据采集及控制程序，对于程序获取数据采集模块的温度，数据解析后写入笔记本电脑桌面的文本文件，文件以姓名加学号命名，每次采集的数据在文件中占据一行，每行的开头为采集数据的时间格式为“YYYY-MM DD,hh:mm:ss”，如：2017-03-26，15:28:32。不同数据项以“;”号分隔。
2. 当解析后温度数据大于10 ℃ 时， 通过控制模块控制风扇开启，当温度小于等于10 ℃ 时，控制风扇运行关闭。

三、实验设备

（1）TAS-RTU-432 采集模块一个；

（2）PT100 温度传感器和变送器一套；

（3）24V直流电源两个；

（4）USB转RS-485串口线一根；

（5）YF-60 控制模块一个；

（6）风扇一个；

（7）笔记本电脑一台。

四、通信协议

（1）TAS-RTU-432 采集模块数据获取命令为:01 04 00 00 00 08 F1 CC，该采集模块接收到命令后会返回测试数据，如：01 04 10 06 32 00 00 00 00 00 00 00 00 00 00 00 00 00 00 6A 1F，具体数据解析内容见表 5-1-1。

表格, 日历

描述已自动生成

（2）通道数据换算公式。

所求值 = 通道数据 X (量程上限 - 量程下限) / 0xFFFF + 量程下限；

例如：

通道数据是 0xCDD2(52690)，电流值量程是 0~20mA；

电流值 = 52690 x (20-0) / 65535 + 0 = 16.079mA。

通道数据是 0xCDD2(52690)，温度值量程是 0~100℃；

温度值 = 52690 x (100-0) / 65535 + 0 = 80.395℃。

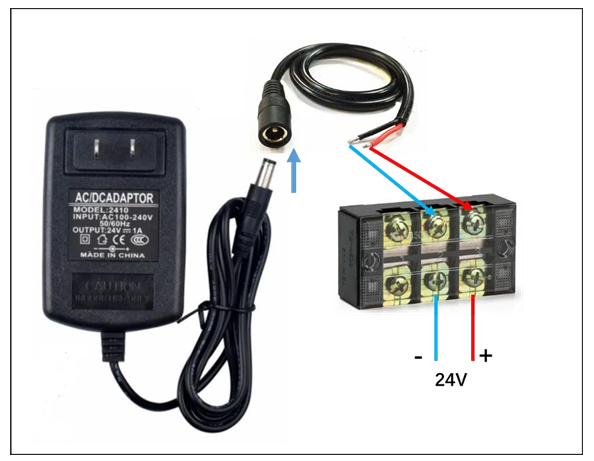
（3）YF-60 控制设备有两个控制输出口，控制指令见表5-1-2。

日历

描述已自动生成

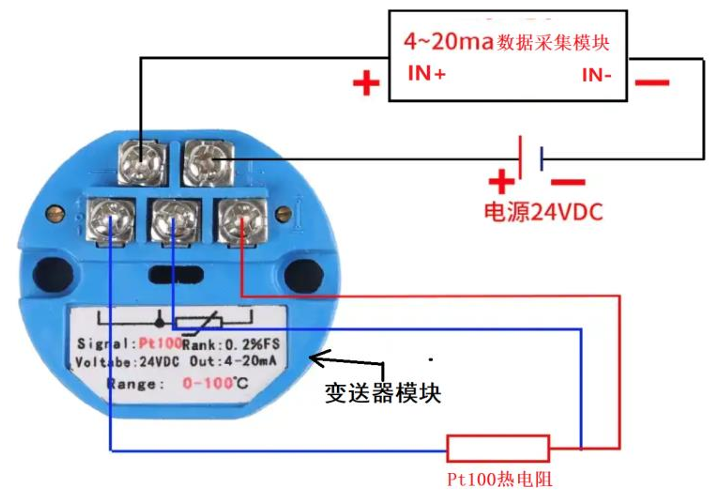
五、实验步骤

（1）实验系统提供了两个 24V 电源适配器，为了进行电源线连接，根据电 源线引出线路，如图5.1.1所示，建立两组电源引出电路。注意：线路全部连接完毕，检查线路正确后再通电。



**图5.1.1** 电源线引出电路

（2）系统提供了PT100与数据采集器硬件连线如图5.1.2所示，该图中Pt100 热敏电阻传感器与变送器模块连接，该模块负责把 Pt100 的输出信号转换成 4 20mA 的电流信号，与数据采集模块连接。采用实验系统提供的配件，根据图9.2 进行硬件线路连接。数据采集模块共有8个输入通道，既（IN0+,IN0-）至（IN7+,IN7-） 选择任一通道连接都可以。在连接线路中需要串接一个24V直流电源。数据采集 模块外观如图5.1.3所示，Pt100传感器外观如图5.1.4所示。



**图5.1.2** Pt100 与数据采集器硬件连线

形状

描述已自动生成

**图5.1.3** Pt100 与数据采集器硬件连线

图示

描述已自动生成

**图5.1.4** 数据采集模块外观

（3）YF-60 控制模块与风扇硬件连线如图5.1.5所示，根据该图进行硬件线 路连接。YF-60控制模块外观如图5.1.6所示，风扇外观如图5.1.7所示。

图示, 示意图

描述已自动生成

**图5.1.5**  YF-60 控制模块与风扇硬件连线

文本

描述已自动生成

**图5.1.6** YF-60 控制模块外观结构

图片包含 照片, 不同, 话筒, 钟表

描述已自动生成

**图5.1.7** 风扇外观结构

（4）在笔记本电脑Windows系统中利用设备管理器，查看是否有需要使用的串口，确认USB-RS485 转换器驱动已经安装。利用该转换器连接笔记本电脑、 数据采集模块YF-60控制模块，笔记本电脑与数据采集模块硬件连线如图5.1.8 所示。USB-RS485 转换器外观结构如图5.1.9所示。

图示, 示意图

描述已自动生成

**图5.1.8** 笔记本电脑与数据采集模块硬件连线

图示, 工程绘图

描述已自动生成

**图5.1.9** USB-RS485 转换器外观结构

（5）打开计算机桌面上的“琏雾实验系统\工具”文件夹中的串口助手软件，运行该软件进行通信调试，确保硬件通信线路正常。

（6）对根据实验内容编写数据采集控制程序。

我们要模拟上述温度数据采集以及设备控制的交互过程，这里我采用Python语言来实现这个模拟过程：

首先我们导入相应的包，分别有serial、time和datetime。serial用来进行串口通信的模拟，time用来进行程序与实验板的时间同步，datetime用来向目标文件写入当前时间。

1. import serial
2. import time
3. from datetime import datetime

然后设置串口参数并打开串口，这里我们需要检查设备管理器中对应的串口号是多少，这里我是COM9，并配置波特率为9600。

1. # 打开串口连接
2. ser = serial.Serial('COM9', 9600, timeout=1)

程序向实验板的发送命令函数send\_command() 的实现。本函数的功能是向设备发送命令以请求数据，并读取设备的响应。这个函数构建一个指定的16进制命令，将其转换为字节形式，通过串口发送该命令，然后读取设备返回的21字节响应数据，并将该响应数据返回给主程序。

1. # 发送获取数据的命令
2. def send\_command():
3. hex\_command = '01 04 00 00 00 08 F1 CC'  # 设定要发送的16进制命令
4. command\_bytes = bytes.fromhex(hex\_command.replace(' ', ''))  # 将16进制命令转换为字节
5. ser.write(command\_bytes)  # 通过串口发送命令
6. response\_data = ser.read(21)  # 读取设备响应，21个字节
7. return response\_data  # 返回响应数据

解析温度数据的函数parse\_temperature() 的实现。这个函数的功能是解析从设备响应中提取的温度数据。程序接收到设备的响应数据后，首先检查响应数据的长度是否至少为21字节。如果长度不足，则抛出异常。然后，从响应数据中提取温度信息（第4和第5字节），将其转换为整数值，并根据给定公式计算实际温度值，最终返回解析后的温度值。

1. # 解析温度数据
2. def parse\_temperature(responseForParse):
3. if len(responseForParse) < 21:
4. raise ValueError("返回数据长度非法")  # 如果数据长度不足21字节，抛出异常
5. data = responseForParse[3:5]  # 提取温度数据部分
6. data = int.from\_bytes(data, byteorder='big')  # 将数据转换为整数
7. parsed\_temperature = data \* (100.0 - 0.0) / 65535 + 8.3  # 计算温度值
8. return parsed\_temperature  # 返回解析后的温度值

控制风扇的开关状态的函数control\_fan() 的实现。这个函数根据传入的状态参数（"on"或"off"）决定发送哪个命令。如果状态是"on"，则发送启动风扇的命令；如果状态是"off"，则发送关闭风扇的命令。如果传入其他状态值，则抛出异常。发送命令后读取设备的响应，以确认命令已被执行。**这里需要注意最后需要将风扇开关后实验板向程序返回的数据从串口缓冲区读出来，返回数据为8字节。**

1. # 控制风扇
2. def control\_fan(state):
3. if state == 'on':
4. command = '03 06 00 02 00 01 E8 28'  # 风扇启动命令
5. elif state == 'off':
6. command = '03 06 00 03 00 01 B9 E8'  # 风扇关闭命令
7. else:
8. raise ValueError("非法指令")  # 如果状态不是'on'或'off'，抛出异常
9. command = bytes.fromhex(command.replace(' ', ''))  # 将命令转换为字节
10. ser.write(command)  # 通过串口发送命令
11. ser.read(8)  # 读取风扇控制的响应

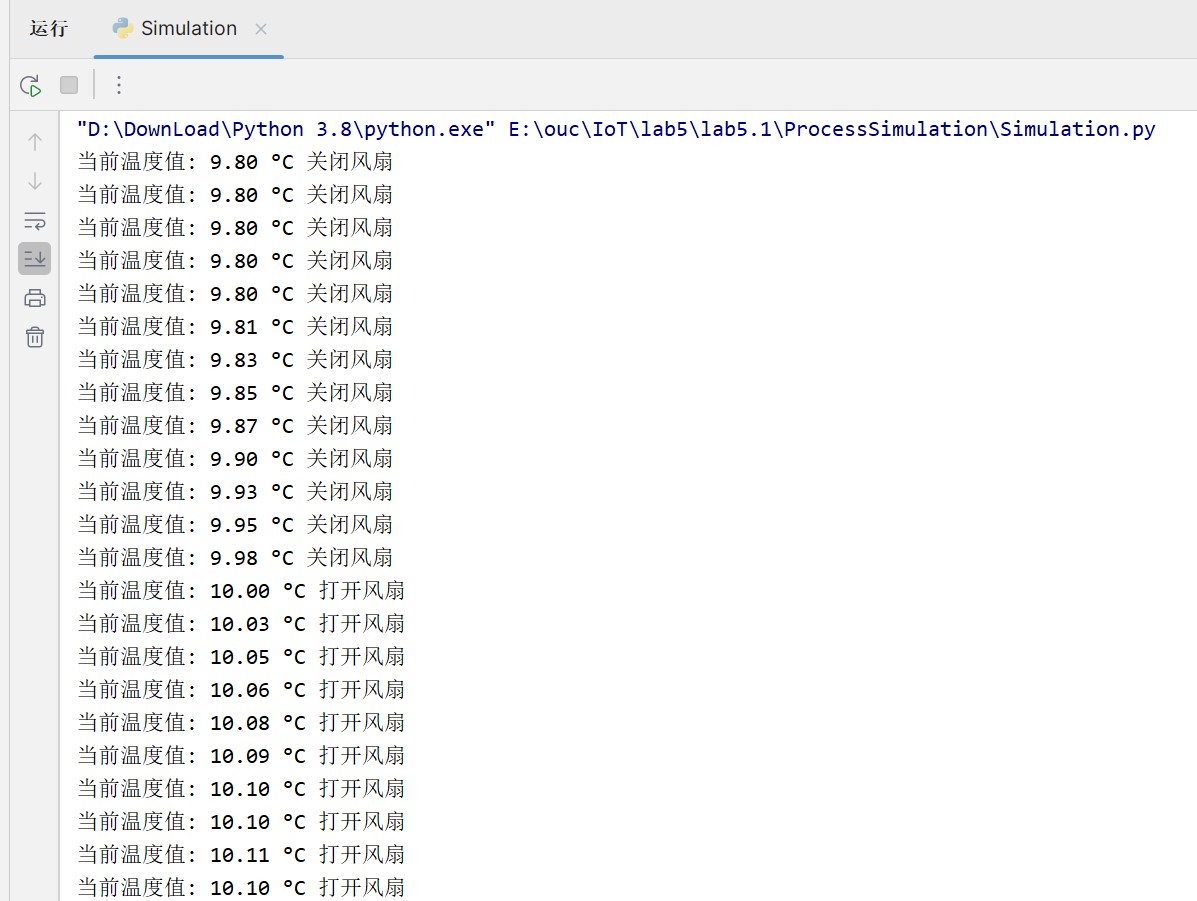
将温度数据记录到文件中的函数save() 的实现。这个函数打开一个指定的文本文件（追加模式），获取当前时间，并将时间和温度值格式化为字符串后写入文件。这样每次调用该函数都会将新的温度数据附加到文件末尾，记录下每次的温度测量结果。

1. # 写入数据到文件
2. def save(result):
3. with open('惠欣宇-21030031009.txt', 'a') as file:  # 以追加模式打开文件
4. current\_time = datetime.now().strftime("%Y-%m-%d,%H:%M:%S")  # 获取当前时间并格式化
5. file.write(f"{current\_time}; {result:.2f}°C\n")  # 将时间和温度值写入

最后就是实现程序的主循环了，我们需要不断循环获取温度数据并根据温度值控制风扇，同时记录数据。在一个无限循环中，每秒钟执行一次操作。首先调用send\_command()函数发送命令并接收响应，然后等待1秒以确保设备有时间响应。如果接收到响应数据，调用parse\_temperature()函数解析温度值，并将结果打印出来和保存到文件中。根据温度值决定是否调用control\_fan()函数来控制风扇的开关。整个循环每秒执行一次，直到捕获到键盘中断（Ctrl+C），此时打印终止信息并关闭串口连接。

1. # 主程序循环
2. try:
3. while True:
4. response = send\_command()  # 发送获取数据的命令并接收响应
5. time.sleep(1)  # 等待设备响应1秒
6. if response:
7. temperature = parse\_temperature(response)  # 解析温度数据
8. print(f"当前温度值: {temperature:.2f} °C", end=" ")  # 打印当前温度
9. save(temperature)  # 将温度数据保存到文件
10. if temperature > 10:
11. control\_fan('on')  # 如果温度大于10度，打开风扇
12. print("打开风扇")
13. else:
14. control\_fan('off')  # 否则，关闭风扇
15. print("关闭风扇")
16. time.sleep(1)  # 每秒采集一次数据
17. except KeyboardInterrupt:
18. print("程序终止")  # 捕获键盘中断异常，打印终止信息
19. finally:
20. ser.close()  # 关闭串口连接

下面给出控制台输出的解析数据的过程截图，如图5.1.10、5.1.11所示。



**图5.1.10** 温度数据控制台运行结果(1)

表格

描述已自动生成

**图5.1.11** 温度数据控制台运行结果(2)

六、报告要求

1. 数据采集模块可连接的传感器数量是多少

根据实验步骤（2）中的介绍我们可以知道，数据采集模块共有8个输入通道，即（IN0+,IN0-）至（IN7+,IN7-）。因此，数据采集模块理论上可以连接8个传感器。

2. 控制模块控制风扇开启和关闭的原理是什么

控制模块通过发送特定的指令到风扇设备来控制其开启和关闭。当温度数据大于10℃时，发送开启风扇的指令，实验板接收到该指令之后对命令进行解析，然后打开风扇进行降温；当温度小于等于10℃时，发送关闭风扇的指令实验板接收到该指令之后对命令进行解析，然后关闭风扇停止降温。这是基于温度阈值的自动控制逻辑。

3. 把控制风扇开启的语句及文本文件的内容附在实验报告中

* **控制风扇开启关闭的语句**

1. # 控制风扇
2. def control\_fan(state):
3. if state == 'on':
4. command = '03 06 00 02 00 01 E8 28'  # 风扇启动命令
5. elif state == 'off':
6. command = '03 06 00 03 00 01 B9 E8'  # 风扇关闭命令
7. else:
8. raise ValueError("非法指令")  # 如果状态不是'on'或'off'，抛出异常
9. command = bytes.fromhex(command.replace(' ', ''))  # 将命令转换为字节
10. ser.write(command)  # 通过串口发送命令
11. ser.read(8)  # 读取风扇控制的响应

主程序中调用上面函数的代码如下，这里进行开关逻辑的实现：

1. if temperature > 10:
2. control\_fan('on')  # 如果温度大于10度，打开风扇
3. print("打开风扇")
4. else:
5. control\_fan('off')  # 否则，关闭风扇
6. print("关闭风扇")

* **文本文件内容**

2024-05-23,16:28:25; 9.80°C

2024-05-23,16:28:27; 9.80°C

2024-05-23,16:28:29; 9.80°C

2024-05-23,16:28:31; 9.80°C

2024-05-23,16:28:34; 9.80°C

2024-05-23,16:28:36; 9.81°C

2024-05-23,16:28:38; 9.83°C

2024-05-23,16:28:40; 9.85°C

2024-05-23,16:28:42; 9.87°C

2024-05-23,16:28:44; 9.90°C

2024-05-23,16:28:46; 9.93°C

2024-05-23,16:28:48; 9.95°C

2024-05-23,16:28:50; 9.98°C

2024-05-23,16:28:52; 10.00°C

2024-05-23,16:28:54; 10.03°C

2024-05-23,16:28:57; 10.05°C

2024-05-23,16:28:59; 10.06°C

2024-05-23,16:29:01; 10.08°C

2024-05-23,16:29:03; 10.09°C

2024-05-23,16:29:05; 10.10°C

2024-05-23,16:29:07; 10.10°C

2024-05-23,16:29:09; 10.11°C

2024-05-23,16:29:11; 10.10°C

2024-05-23,16:29:13; 10.10°C

2024-05-23,16:29:15; 10.09°C

2024-05-23,16:29:17; 10.09°C

2024-05-23,16:29:20; 10.09°C

2024-05-23,16:29:22; 10.08°C

2024-05-23,16:29:24; 10.08°C

2024-05-23,16:29:26; 10.07°C

2024-05-23,16:29:28; 10.06°C

2024-05-23,16:29:30; 10.05°C

2024-05-23,16:29:32; 10.05°C

2024-05-23,16:29:34; 10.04°C

2024-05-23,16:29:36; 10.03°C

2024-05-23,16:29:38; 10.03°C

2024-05-23,16:29:40; 10.02°C

2024-05-23,16:29:43; 10.02°C

2024-05-23,16:29:45; 10.01°C

2024-05-23,16:29:47; 10.00°C

2024-05-23,16:29:49; 10.00°C

2024-05-23,16:29:51; 9.99°C

2024-05-23,16:29:53; 9.99°C

2024-05-23,16:29:55; 9.98°C

2024-05-23,16:29:57; 9.98°C

2024-05-23,16:29:59; 9.98°C

2024-05-23,16:30:01; 9.97°C

2024-05-23,16:30:03; 9.97°C

4. 写出调试过程遇到的主要问题和解决方法

* **问题和解决方法1**

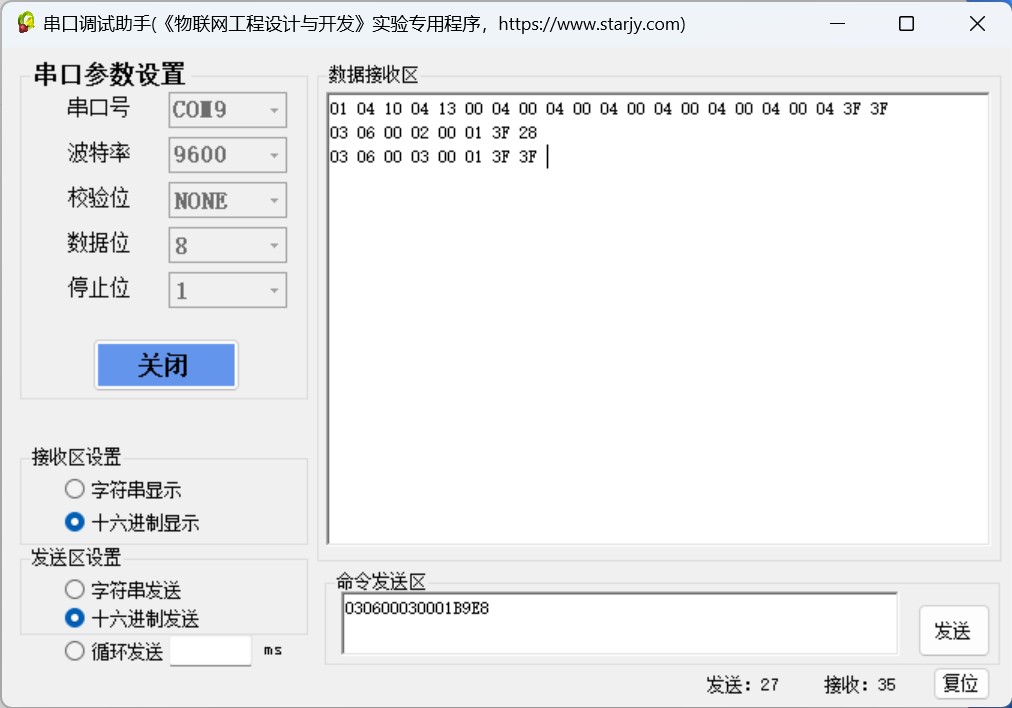
**问题：**在我们编辑好通信代码后，进行数据采集与解析，这个时候发现实验板并不能按照预期期望做出相应的动作响应，也就是说，当测得的温度大于10摄氏度时实验板的风扇不开启。

**解决方法：**经过研究后，我们首先在控制台输出每一次实验板的返回数据，这里发现程序只有第一次收到的返回数据是21字节的，而后续收到的返回数据都是29字节的。在这29字节中我们发现后面21个字节是有效的温度返回数据，而前面8个字节看似是无效的。这里我们想到是否向实验板发送打开或关闭灯泡的命令后实验板会有返回数据。这里我们通过串口助手来进行验证。如图5.1.12、5.1.13所示，向实验板发送风扇打开命令后，实验板会通过串口向我们返回一个8字节的数据，这个数据会缓存在串口中，风扇关闭命令也同理。

图形用户界面, 文本, 应用程序

描述已自动生成

**图5.1.12** 风扇打开命令的数据返回



**图5.1.13** 风扇关闭命令的数据返回

那么我们的问题就迎刃而解，只需要将每一次执行开关风扇的命令之后的返回数据从串口中读出即可，解决部分的代码高亮给出。

1. # 控制风扇
2. def control\_fan(state):
3. if state == 'on':
4. command = '03 06 00 02 00 01 E8 28'  # 风扇启动命令
5. elif state == 'off':
6. command = '03 06 00 03 00 01 B9 E8'  # 风扇关闭命令
7. else:
8. raise ValueError("非法指令")  # 如果状态不是'on'或'off'，抛出异常
9. command = bytes.fromhex(command.replace(' ', ''))  # 将命令转换为字节
10. ser.write(command)  # 通过串口发送命令
11. ser.read(8)  # 读取风扇控制的响应

七、实验总结

在完成这次《物联网系统设计与开发》课程中的“温度数据采集及控制交互编程”实验后，我深刻体会到了理论知识与实践技能相结合的重要性。实验过程中，编写并调试了一个能够实时采集温度数据并根据数据控制风扇开关的程序，这不仅加深了我对物联网系统中数据采集和设备控制原理的理解，也锻炼了我的编程能力和问题解决能力。通过解决风扇控制命令响应的问题，我学会了如何分析串口返回的数据，使用串口助手工具，并优化了程序以确保硬件设备能够正确响应。此外，我还将采集到的温度数据以指定格式记录在文本文件中，为后续的数据复查和分析提供了便利。尽管实验取得了成功，但我也意识到在代码结构和效率上还有改进空间，未来我希望能够进一步模块化代码，提高程序的可读性和可维护性，并扩展程序功能，增加对更多类型传感器的支持，实现更复杂的控制逻辑，以提升系统性能。这次实验不仅让我对物联网技术有了更加深入的了解，也为我未来的学习和研究打下了坚实的基础。