|  |  |
| --- | --- |
| 题目(中 文): | 基于STM32的衣服烘干机设计 |
| (英 文): | Bob |
|  |
| 姓 名 | 杨春华 |
| 学 号 | 202006020131 |
| 院 （系） | 智能制造学院 |
| 专业、年级 | 电子信息工程 2020级 |
| 指导教师 | 陈光辉（讲师） |

2024年 5 月 8 日

目 录

1 绪论 1

1.1 衣服烘干机在家电应用背景和重要性。 1

1.2 衣服烘干机设计项目的目标和实际应用的意义 1

1.2.1 衣服烘干机设计项目的目标 1

1.2.2 实际应用的意义 2

1.3 课题研究的主要内容 2

1.4 课题撰写的组织结构 3

2 系统架构分析 4

2.1 硬件架构 4

2.1.2 单片机概述 5

2.2 软件架构 6

3 传感器选择与接口设计 8

3.1 温湿度传感器 8

3.2 传感器接口设计 8

4 控制算法设计与优化 9

4.1 PID控制算法 9

4.2 控制参数优化 9

5 用户界面设计与实现 10

5.1 显示模块选择 10

5.2 用户交互设计 10

6 系统集成与测试 11

6.1 硬件系统集成 11

6.2 软件系统集成 11

6.3 系统测试与验证 11

6.3.1 功能测试 12

6.3.2 性能测试 12

6.3.3 验证 12

结论 13

参考文献 14

附录A: 15

致 谢 16

基于STM32的衣服烘干器

摘□要

本文详细介绍了衣服烘干器的基本原理和功能特性，并着重介绍了STM32C8T6单片机。针对STM32C8T6的最小系统板，讲解了其由多个模块组成，其中晶振电路为STM32C8T6提供基准时钟，复位电路用于复位STM32C8T6。此外，系统还包括温湿度模块、光感模块、风扇、加热器、OLED显示模块和外扩Flash模块。通过PID算法，系统可以根据温度采样来控制风扇和加热器，实现衣物的烘干。为了记录故障数据和用户机型数据，系统还配备了一个黑盒。最后，本文提出了系统的程序设计，并给出了程序流程图。通过对衣服烘干器的优化设计和实现，可以提高其性能和效率，为用户提供更出色的使用体验。这些优化措施将使衣服烘干器成为一款功能强大、可靠性高的家电产品。

【关键词】：衣服烘干器；STM32C8T单片机; 最小系统板；PID算法；优化设计

本文阐述了衣服烘干器的基本原理及其功能特性，并对STM32C8T6单片机进行详细地介绍，然后在此的基础上，解释了STM32C8T6的最小系统板由哪些模块组成，其中晶振电路主要是为了STM32C8T6提供基准时钟用的，复位电路主要是给STM32C8T6复位使用，其外部由温湿度模块，光感模块，风扇，加热器，oled显示模块，外扩flash模块组成，PID算法通过温度的采样，来控制风扇和加热器来烘干衣服，黑盒来记录一些故障数据和用户机型的数据。最后，提出了系统的程序设计，并给出程序流程图。

**Clothes Dryer Based on STM32**

**Abstract**

features of a clothes dryer, with a specific focus on the STM32C8T6 microcontroller. It discusses the components of the STM32C8T6 minimum system board, including the crystal oscillator circuit for providing the clock reference and the reset circuit for resetting the microcontroller. Additionally, the system includes modules such as temperature and humidity sensing, light sensing, fan, heater, OLED display, and external Flash module. The PID algorithm is employed to control the fan and heater based on temperature sampling to achieve effective clothes drying. The system is equipped with a black box for recording fault data and user-specific information. Furthermore, the article proposes a program design and provides a flowchart. By optimizing the design and implementation of the clothes dryer, performance and efficiency can be improved, offering users an enhanced experience. These optimization measures make the clothes dryer a powerful and reliable household appliance.

**【Key words】** Clothes dryer; STM32F103C8T6 microcontroller; Minimum system board; PID algorithm; Optimized design.

1. 绪论
   1. 衣服烘干机在家电应用背景和重要性。

随着社会的发展和生活水平的提高，家电产品在人们的日常生活中扮演着越来越重要的角色。衣服烘干机作为家电产品的一种，具有重要的应用背景和功能，受到越来越多家庭的青睐。

在传统的衣物清洗过程中，晾晒衣物是必不可少的一步。然而，室外晾晒受到季节、天气等因素的限制，且晾晒时间较长，使得衣物无法迅速干燥。这就带来了一些问题，如天气潮湿导致衣物难以干透、室内空间不足等。

衣服烘干机的出现解决了这些问题。它能够快速将洗涤干净的衣物进行烘干，通过加热和循环风扇的作用，将衣物中的水分蒸发并排出，使衣物在短时间内达到干燥的状态。这不仅节省了晾晒的时间，同时也提供了更加便捷和高效的衣物处理方式。

衣服烘干机的重要性在于它满足了现代家庭对高效生活的需求。在快节奏的现代生活中，人们更加注重时间的利用和生活的便利性。使用衣服烘干机可以大大缩短衣物的干燥时间，使衣物迅速可用，提高了生活的效率和便捷性。

此外，衣服烘干机还具有节能的优势。相比室外晾晒，衣服烘干机通常能够更好地控制烘干过程的温度和时间，减少了能源的浪费。一些先进的烘干机还配备了智能控制系统，能够根据衣物的湿度和材质进行智能调节，进一步提高能源利用效率。

综上所述，衣服烘干机在家电应用中具有重要的背景和重要性。它不仅解决了衣物晾晒的时间限制和空间限制问题，还提供了高效、便捷和节能的衣物处理方式，满足了现代家庭对于高效生活的需求。随着科技的发展和人们对生活品质的要求不断提高，衣服烘干机在家庭中的应用前景将更加广阔。

* 1. 衣服烘干机设计项目的目标和实际应用的意义
     1. 衣服烘干机设计项目的目标

提高烘干效率：通过优化烘干机的加热系统、风扇设计和热量循环等方面，使得衣物能够在较短的时间内彻底干燥，提高烘干效率。

降低能源消耗：通过合理的能源利用和控制策略，减少烘干过程中的能源浪费，提高能源利用效率，降低用户的能源消耗成本。

提供安全保护：设计安全性能良好的烘干机，确保在烘干过程中不存在过热、漏电、火灾等安全隐患，保障用户的人身安全和财产安全。

增加用户便利性：考虑用户的使用习惯和需求，设计简单易用的控制界面和操作方式，提供智能化的功能和人性化的设计，提升用户的使用便利性和舒适度。

* + 1. 实际应用的意义

提供衣物干燥的便捷方式：衣服烘干机能够快速将洗涤干净的衣物进行干燥，解决了传统晾晒的时间和空间限制问题。用户可以在家中轻松完成衣物干燥，无需依赖天气和室外环境条件。

节省时间和提高效率：传统晾晒衣物需要较长的时间才能完全干透，而衣服烘干机能够在较短的时间内实现快速干燥，节省了用户的时间和精力，提高了生活的效率。

提高生活品质和舒适度：使用衣服烘干机可以避免衣物因为潮湿而产生异味和霉变的问题，保持衣物的干爽和清洁，提高了生活品质和穿着舒适度。

节能环保：合理设计的衣服烘干机能够通过优化能源利用和控制策略，降低烘干过程中的能源消耗，减少对环境的负面影响，符合节能环保的理念和要求。

安全可靠：设计安全性能良好的衣服烘干机，能够避免因过热、漏电等问题引发的安全隐患，保障用户的人身安全和财产安全。

* 1. 课题研究的主要内容

硬件设计与嵌入式系统开发：设计和搭建基于STM32微控制器的衣服烘干器硬件系统。这包括选择合适的传感器（如温度传感器、湿度传感器等）、执行器（如风扇、加热器等）、显示屏和用户交互界面等，并将它们与STM32微控制器进行连接和通信。同时，开发嵌入式软件系统，包括驱动程序、控制算法和用户界面等。

温湿度感知与控制：研究温湿度传感器的选型、数据采集和处理方法，实时监测衣物烘干过程中的温度和湿度变化。基于采集到的数据，设计合适的控制算法，根据预设的烘干模式和衣物状态，控制风扇和加热器的工作，以实现精确的温湿度控制。

热量传递与能源优化：研究衣物烘干过程中的热量传递机制，包括对衣物的加热、水分蒸发和空气流动等进行建模和分析。通过优化烘干机的设计和控制策略，减少能源消耗并提高烘干效率。这可以涉及到热量循环的优化、风扇和加热器的功率控制等方面的研究。

安全性与故障保护：研究衣服烘干器的安全性和故障保护机制，以确保用户的人身安全和设备的可靠性。例如，设计过热保护系统、电气安全保护措施和故障检测与报警机制等。

用户界面与智能化功能：设计友好的用户界面，通过液晶显示屏、按键和指示灯等，提供用户与衣服烘干器的交互和控制方式。。

性能测试与评估：对基于STM32的衣服烘干器进行性能测试和评估，包括烘干效率、能源消耗、温湿度控制精度和安全性能等方面的评估。通过实验测试和数据分析，评估系统设计和控制算法的性能，并提出改进措施。

* 1. 课题撰写的组织结构

1.绪论

1.1 研究背景

1.2 课题目的和意义

1.3 研究内容概述

1.4 研究方法和结构安排

2.相关技术和理论

2.1 衣服烘干原理和热力学基础

2.2 STM32微控制器及其应用

2.3 温湿度传感器和控制算法

2.4 其他相关技术和理论

3.系统设计与实现

3.1 系统架构设计

3.2 STM32硬件设计和连接

3.3 传感器与执行器的选择和集成

3.4 嵌入式软件系统开发

3.5 用户界面设计

4.温湿度感知与控制

4.1 温湿度传感器的原理和工作

4.2 数据采集与处理

4.3 温湿度控制算法设计

4.4 实时监测与反馈控制

5.安全性与故障保护

5.1 设备安全设计和故障保护机制

5.2 过热保护系统设计

6用户界面与智能化功能

6.1 用户界面设计和人机交互方式

6.2 智能化功能的添加和实现

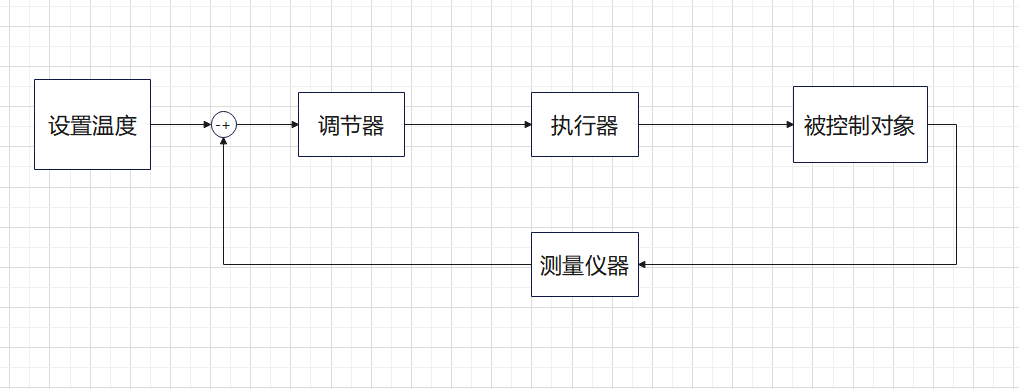
7.结论与展望

9.1 研究成果总结

9.2 存在的问题和改进方向

1. 系统架构分析
   1. 硬件架构

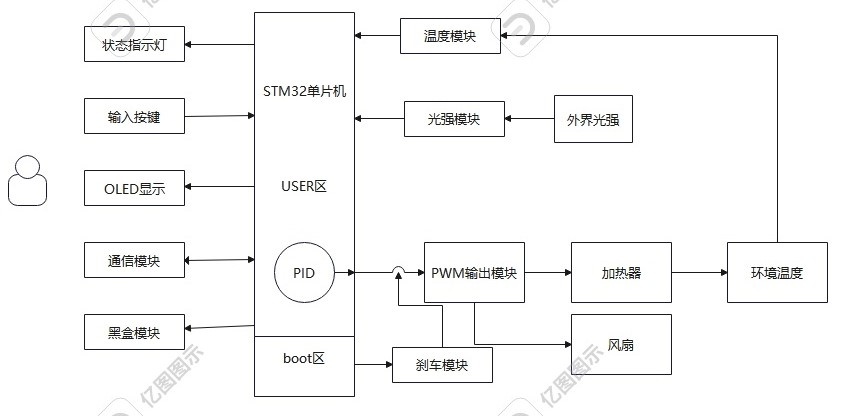
烘干器的温度控制是本项目的核心，我们需要对它进行精确的采样和控制。系统的温度需要设置为恒温或是根据环境变换，闭环控制系统是一种常用的手段，本设计也是采用这种系统来控制的，闭环控制的流程图如图一所示。



闭环温度控制流程

温度控制系统由调节器，执行器，被控制对象，测量仪器四部分组成，测量装置对密闭的被加热的环境温度进行测量，我们对测量值和给定值进行比较，如果存在差异，我们便通过调节器对信号进行处理，输出给执行器，来作用到被控制对象开启或关闭。

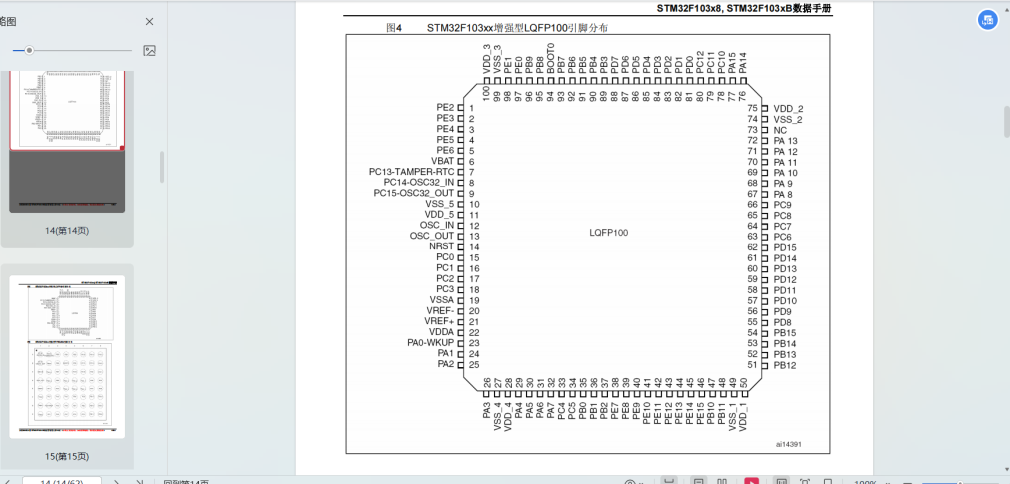
对与上面的理论知识我们可以，使用大致如下的硬件结构：STM32C8T6单片机，热敏电阻模块，光敏电阻模块，蓝牙模块，加热器模块，风扇模块，OLED模块，黑盒模块。



控制结构图

* + 1. 单片机概述

本设计考虑到对温度的控制，需要选择一款计算能力强，性价比高的单片机，STM32F103C8T6是STMicroelectronics（意法半导体）推出的一款32位ARM Cortex-M3系列单片机（Microcontroller Unit，MCU），非常适合本设计。它具有以下特点和功能



STM32F103X芯片图

架构和性能

STM32F103C8T6基于ARM Cortex-M3内核，采用了Harvard架构，提供高性能和低功耗的特点。

内置浮点单元（Floating Point Unit，FPU），支持单精度（32位）浮点运算，提供更高的计算能力。

最高主频可达72 MHz，具备强大的执行能力。

存储和存储器:

STM32F103C8T6内置64 KB的Flash存储器，用于存储应用程序代码和数据。

它还具有20 KB的SRAM，用于运行时数据存储和处理。

支持扩展存储器接口，如SDIO、SPI、I2C等，可以连接外部存储器设备。

外设接口:

STM32F103C8T6提供多个通用输入/输出引脚（General-Purpose Input/Output，GPIO），可用于连接外部设备和传感器。

支持多种通信接口，包括UART、SPI、I2C、CAN等，用于与其他设备进行数据交换和通信。

具备多个定时器和计数器，可用于实现精确的定时和计数功能。

电源管理:

STM32F103C8T6支持多种供电方式，包括外部电源和嵌入式电源管理单元（Power Management Unit，PMU）。

带有低功耗模式，可在不需要高性能运算时降低功耗，有助于提高烘干器的能效。

开发工具和生态系统:

STMicroelectronics提供了丰富的开发工具和软件库，如STM32Cube软件开发平台、HAL库、CubeMX等，简化了开发过程。

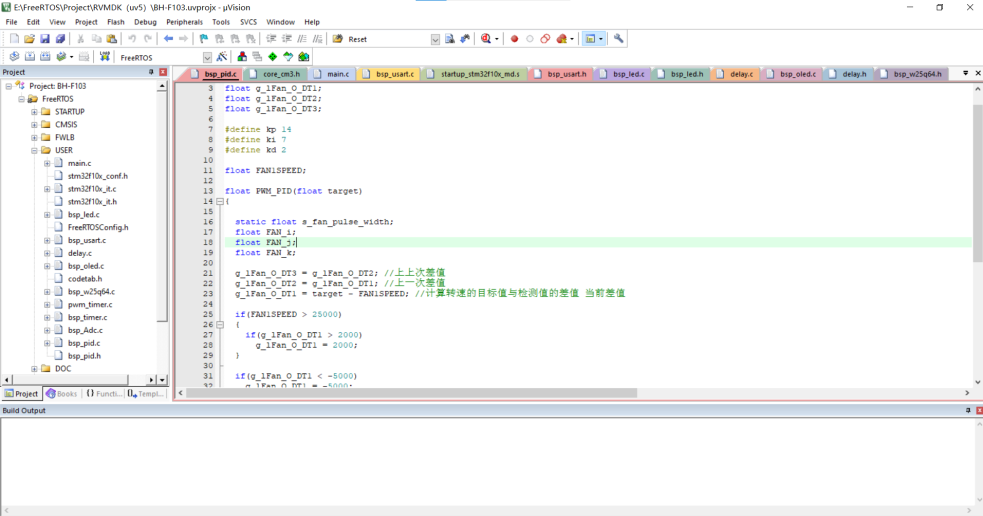
支持多种集成开发环境（Integrated Development Environment，IDE），如Keil MDK、IAR Embedded Workbench等，为开发者提供了便利。

在烘干器的设计中，STM32F103C8T6单片机可以用于实现温度控制、时间控制、显示界面等功能。通过外设接口和通信接口，它能够连接温度传感器、液晶显示屏、按键等外部设备，并与用户进行交互。同时，其强大的性能和可靠性确保了烘干器的稳定运行。

* 1. 软件架构

软件开发环境：

集成开发环境（Integrated Development Environment，IDE）：您可以使用常见的IDE，Keil MDK（Microcontroller Development Kit）。这些IDE提供了代码编辑器、编译器、调试器和仿真器等工具，用于开发和调试STM32F103C8T6的嵌入式软件。



KEIL软件图

开发工具：

编程语言：通常使用C语言进行STM32F103C8T6的嵌入式软件开发。STM32标准外设库：STMicroelectronics为STM32系列提供了一套标准外设库（Standard Peripheral Library），用于访问和配置STM32F103C8T6的外设。

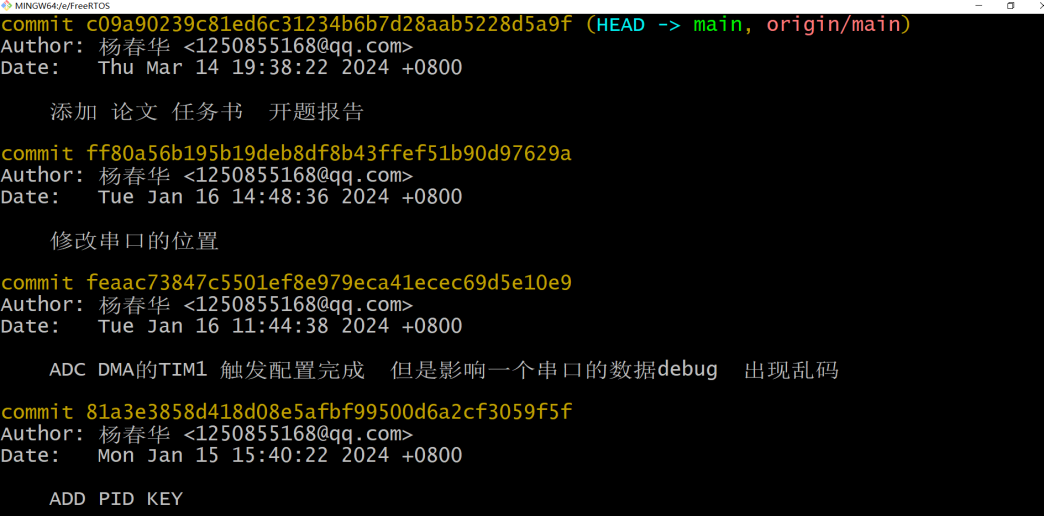
调试工具：可以使用调试器和仿真器来调试和验证您的嵌入式软件。ST-Link是STMicroelectronics提供的常用仿真器，可与STM32F103C8T6进行连接并进行调试。



ST-LINK仿真烧入模块图

代码版本控制：

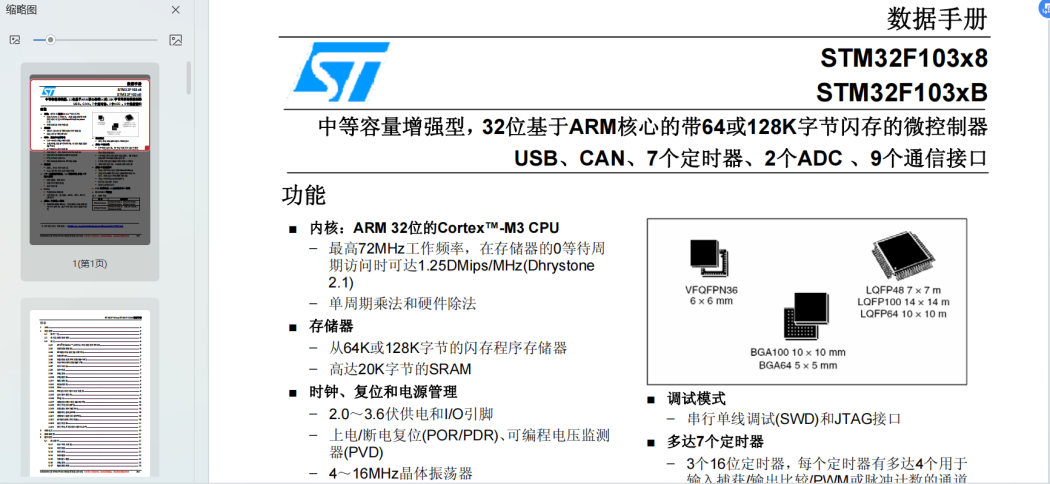
版本控制工具：使用版本控制工具（例如Git）来管理和跟踪您的代码版本。这将有助于团队协作、代码备份和版本管理。



Git项目推进结构图

相关文档和资源：

参考手册和数据手册：STMicroelectronics为STM32F103C8T6提供了详细的参考手册和数据手册，其中包含了关于寄存器配置、外设功能和编程接口等的详细信息。



STMicroelectronics为STM32F103X数据手册图

主要模块和功能：

用户界面模块：负责与用户进行交互，包括显示屏、按键和报警器等。

温度检测模块：使用温度传感器读取烘干室内部和外部的温度，并将其传输到主控制模块。

控制算法模块：基于温度检测结果，使用控制算法来调整烘干器的操作，例如温度控制和烘干时间控制。

电机驱动模块：控制烘干器内的电机或风扇，调节空气流动和湿度。

数据存储模块：负责存储烘干器的运行日志、设置参数和历史数据等，可以使用闪存或外部存储器。

中断处理：

使用STM32F103C8T6单片机的中断功能来实现实时响应和处理。您可以针对不同的模块和事件配置和使用不同的中断优先级。

温度检测模块可以使用定时器中断进行周期性的温度采集，并将采集到的数据传输给主控制模块进行处理。

用户界面模块可以使用外部中断或定时器中断来检测按键操作和定时刷新显示屏。

控制算法模块可以在主循环中进行计算，或者在定时器中断中以一定的时间间隔进行计算和调整控制参数。

电机驱动模块可以使用PWM（脉冲宽度调制）输出，并通过定时器中断来生成PWM信号，控制电机的转速和风扇的运行状态。

通信接口：

STM32F103C8T6具有多种通信接口，如UART、SPI、I2C等，可与其他设备进行数据交换。

示例代码和应用笔记：STMicroelectronics提供了丰富的示例代码和应用笔记，用于帮助您理解和使用STM32F103C8T6的各种功能和外设。

P1口

|  |  |  |  |
| --- | --- | --- | --- |
| 口线 | 引脚 | 第二功能 | 说明 |
| PA.2 | 2 | TX | 单片机发送数据 |
| PA.3 | 3 | RX | 单片机接收数据 |
| PA.9 | 9 | 指示灯 | 绿灯 |
| PA.10 | 10 | 指示灯 | 红灯 |
| PB.6 | 6 | SCL | 时钟线 |
| PB.7 | 7 | SDA | 数据线 |
| PA.4 | 4 | CS | 片选线 |
| PA.5 | 5 | SCL | 时钟线 |
| PA.6 | 6 | MISO | 主机接收数据 |
| PA.7 | 7 | MOSI | 主机发送数据 |
| PA.0 | 0 | ADC0 | 热敏电阻采样 |
| PA.1 | 1 | ADC1 | 光敏电阻采样 |
| PB.8 | 8 | PWM | 加热器 |
| PB.13 | 13 | PWM | 风扇 |
| PB.12 | 12 | BREAK | PWM刹车 |

1. 传感器选择与接口设计
   1. 温湿度传感器

温数字输出：DHT11传感器通过单一的数据线进行通信，并以数字形式输出温度和湿度数据。

低成本：DHT11是一种经济实惠的传感器，适用于低成本和资源受限的项目。

测量范围：DHT11可测量的温度范围为0°C到 50°C，湿度范围为20%RH到 90%RH。

精度：DHT11的温度精度约为±2°C，湿度精度约为±5%RH。

单线通信：DHT11使用单一的数据线进行串行通信，简化了连接和控制。

低功耗：DHT11具有低功耗特性，在工作状态下的电流消耗非常低。

应用广泛：DHT11被广泛应用于室内温湿度监测、气象站、温室控制、智能家居等领域。

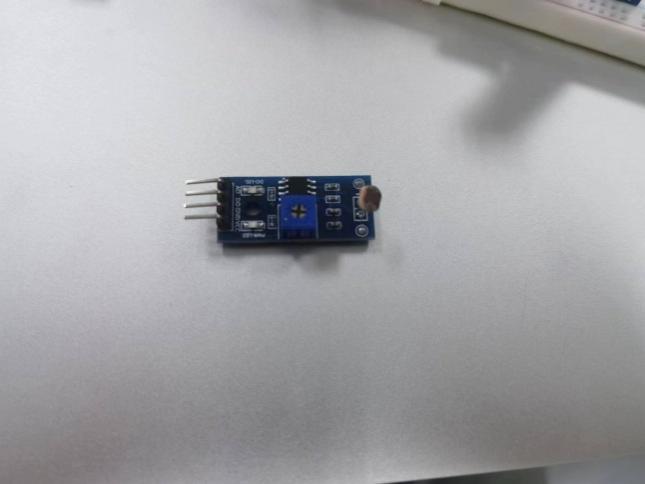
使用DHT11温度传感器时，通常需要注意以下事项：

供电电压：DHT11的供电电压通常为3.3V或5V，确保为其提供正确的电源电压。

读取间隔：DHT11传感器需要一定的时间来完成温湿度测量。在读取之间应给予足够的延时，避免频繁读取，一般建议读取间隔不小于2秒。

数据解析：DHT11传感器通过一串数据来表示温度和湿度值。根据传感器提供的数据格式，正确解析数据以获取准确的温湿度值。

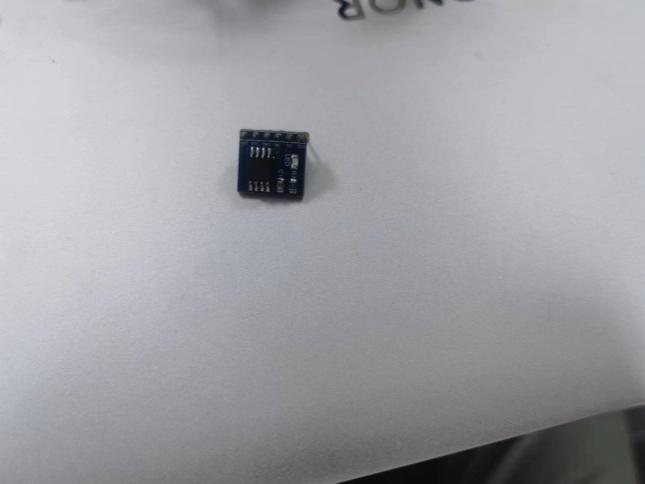
环境条件：DHT11对环境条件较为敏感，应尽量避免暴露在极端的温度、湿度和污染环境中。



温湿度传感器图

* 1. 传感器接口设计

数字传感器接口：对于数字传感器，可以使用STM32F103C8T6的GPIO（通用输入输出）引脚直接与传感器进行连接。根据传感器的规格和要求，配置相应的引脚作为输入，并使用适当的通信协议SPI与芯片进行通信。



黑盒储存模块图

模拟传感器接口：对于模拟传感器，您可以使用STM32F103C8T6的模拟输入引脚（ADC，模数转换器）来接收传感器的模拟输出。配置ADC通道，并使用相应的转换模式和采样率来读取传感器的模拟值。

通信接口：如果您选择了其他类型的传感器（如光敏传感器、气体传感器或压力传感器），这些传感器可能需要使用特定的通信接口（如I2C、SPI或UART）与STM32F103C8T6进行通信。您需要根据传感器的规格和要求来选择和配置适当的通信接口，并使用相应的库函数或驱动程序来实现数据交换。



串口模块图

风扇模块:是一个用于控制风扇转速和方向的电子模块。它通常包括一个微控制器、一个电机驱动器和一个电源管理器。风扇模块可以用于控制风扇的转速和方向，从而实现对风扇的智能控制。

风扇模块的主要功能包括

1. 控制风扇的转速：通过调整PWM（脉冲宽度调制）信号的占空比来控制风扇的转速。占空比越高，风扇的转速越快。

2. 控制风扇的方向：通过改变PWM信号的频率来控制风扇的方向。当频率为0时，风扇停止旋转；当频率为正时，风扇顺时针旋转；当频率为负时，风扇逆时针旋转。

总之，风扇模块是一个用于控制风扇转速和方向的电子模块，它可以帮助我们实现对风扇的智能控制，提高设备的工作效率和稳定性。



风扇模块图

1. 控制算法设计与优化
   1. PID控制算法

温度控制算法：设计一个温度控制算法，以保持烘干室内部的温度在设定范围内。您可以使用比例积分微分（PID）控制算法作为基础，根据温度传感器的反馈信号和设定温度进行计算，并输出控制信号来调整电机或风扇的运行状态。



加热器模块图

* 1. 控制参数优化

参数调整：通过实验和测试，对控制算法的参数进行调整和优化，以提高控制系统的稳定性和响应速度。使用试错法、自适应控制参数调整过程。

响应时间优化：通过对算法和控制环路进行优化，减少控制系统的延迟和响应时间。这可以包括优化算法的执行速度、使用中断优化和优化传感器采样频率等。

struct \_pid{

float SetTemp; //定义设定值

float ActualTemp; //定义实际值

float err; //定义偏差值

float err\_next; //定义上一个偏差值

float err\_last; //定义最上前的偏差值

float Kp,Ki,Kd; //定义比例、积分、微分系数

}pid;

void PID\_init(){

pid.SetTemp=45.0;

pid.ActualTemp=0.0;

pid.err=0.0;

pid.err\_last=0.0;

pid.err\_next=0.0;

pid.Kp=0.2;

pid.Ki=0.015;

pid.Kd=0.2;

}

float PID\_realize(float Temp){

pid.SetTemp=Temp;

pid.err=pid.SetTemp-pid.ActualTemp;

float incrementSpeed=pid.Kp\*(pid.err-pid.err\_next)+pid.Ki\*pid.err+pid.Kd\*(pid.err-2\*pid.err\_next+pid.err\_last);

pid.ActualTemp+=incrementSpeed;

pid.err\_last=pid.err\_next;

pid.err\_next=pid.err;

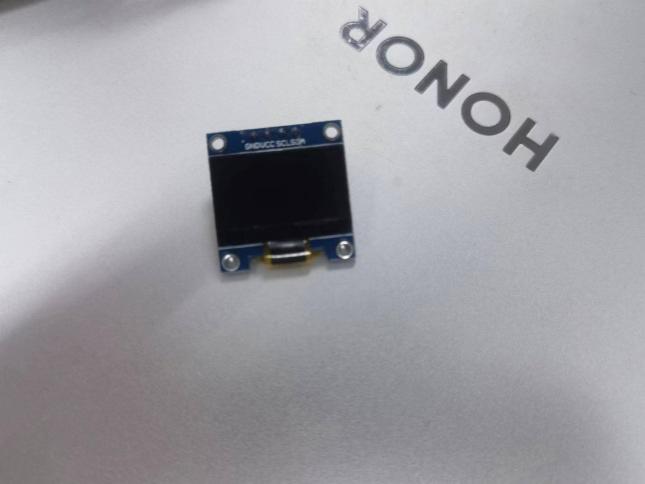
return pid.ActualTemp;

}

1. 用户界面设计与实现
   1. 显示模块选择

显示屏选择：选择一种合适的显示屏，例如128x64 OLED显示屏，它具有足够的分辨率和可视角度，以显示烘干器的状态和参数。

界面布局：设计一个直观、易于使用的用户界面布局，以方便用户查看和调整烘干器的设置和参数。布局可以包括顶部菜单栏、主要信息区域和底部操作按钮区域。在主要信息区域，将温度、湿度、剩余时间和烘干状态以数字或图表的形式显示出来。在底部操作按钮区域，放置各种功能按钮，如启动/停止按钮、温度调节按钮和时间调节按钮。



OLED模块图

* 1. 用户交互设计

按键：显示屏只支持物理按键输入，您可以使用GPIO引脚和外部中断来检测按键状态。为不同的功能按钮分配相应的GPIO引脚，并使用中断来捕获按键事件。

状态更新和响应：设置定时器或中断来定期更新显示屏上的状态信息。例如，使用定时器来更新温度和湿度显示，并通过读取传感器数据来更新烘干器的状态。根据用户的设定，您可以使用定时器来调整烘干时间或调整温度设定值。

1. 系统集成与测试
   1. 硬件系统集成

硬件连接：将STM32F103C8T6微控制器与其他组件进行连接。这包括连接传感器（如温度传感器和湿度传感器）、执行器（如电机和风扇）以及显示设备（如显示屏）等。根据硬件规格和引脚分配，确保正确连接和接线。

供电和电源管理：为STM32F103C8T6和其他组件提供适当的电源供应。这可能涉及使用稳压器和电源管理电路，以确保系统稳定运行并提供所需的电流和电压。



12V转3.3V供电模块图

外设配置：根据系统需求，配置STM32F103C8T6的外设。这包括配置GPIO引脚，设置定时器和中断，以及配置串口通信等。确保正确的外设配置和初始化，以便与其他组件进行交互。

* 1. 软件系统集成

固件开发：使用适当的集成开发环境（IDE）和编程语言（如C/C++），开发STM32F103C8T6的固件。这包括编写与硬件交互的驱动程序、实现控制算法和用户界面的功能等。确保正确的软件开发和编译，生成可用于烘干器的固件文件。

通信协议：根据系统需求，实现与其他组件之间的通信协议。例如，使用串口通信协议与电机控制器进行通信，或使用I2C或SPI与传感器进行通信。确保通信协议的正确实现和可靠性。

集成测试：进行系统级的集成测试，验证硬件和软件的集成。测试烘干器的各个功能模块，例如温度控制、光强检测、时间计数和用户界面交互等。检查系统的稳定性、功能完整性和性能。

故障排除和调试：在集成测试过程中，如果发现问题或故障，进行故障排除和调试。使用调试工具（例如调试器或逻辑分析仪）和日志记录，确定问题的原因并进行修复。

* 1. 系统测试与验证
     1. 功能测试

温度控制测试：验证烘干器的温度控制功能。通过设置目标温度并监测实际温度，检查系统能否正确地控制加热元件以达到目标温度。

光强检测测试：验证烘干器的湿度检测功能。通过模拟不同光强条件，检查系统是否能够准确地测量光强并进行相应的处理。

时间计数测试：验证烘干器的时间计数功能。设置不同的烘干时间，并监测系统的倒计时功能是否正确，并在时间到达时触发相应的操作。

黑盒记录测试:对flash的存储,进行写入和读取，对比数据是否有不一致的地方，确保数据可以完整准确的储存下来。

* + 1. 性能测试

温度稳定性测试：测试系统在不同负载条件下的温度稳定性。通过监测温度的变化和回复时间，评估系统的温度稳定性和调节性能。

加热时间测试：测试系统加热到设定温度所需的时间。通过记录加热时间和目标温度之间的差异，评估系统的加热性能。

烘干效果测试：使用标准样品进行烘干效果测试。通过比较原始样品和烘干后的样品之间的差异，评估系统的烘干效果和效率。

* + 1. 验证

温度控制测试

确定测试参数：首先确定需要测试的温度范围和控制要求。

准备测试设备：选择合适的温度计或温度传感器，确保其准确度和响应速度符合测试要求。另外，确保测试设备和系统的稳定性和可靠性。

建立测试环境：将温度控制设备或系统安装在适当的测试环境中。确保环境温度稳定，并消除可能的干扰因素。

测试设备校准：在进行温度控制测试之前，对测试设备进行校准，以确保其准确和可靠。

启动测试设备：根据设备的操作说明，启动温度控制设备或系统，并将其设置为目标温度值。

数据记录和分析：使用温度计或温度传感器测量设备或系统输出的温度，并记录这些数据。根据设定的时间间隔，定期记录温度读数。

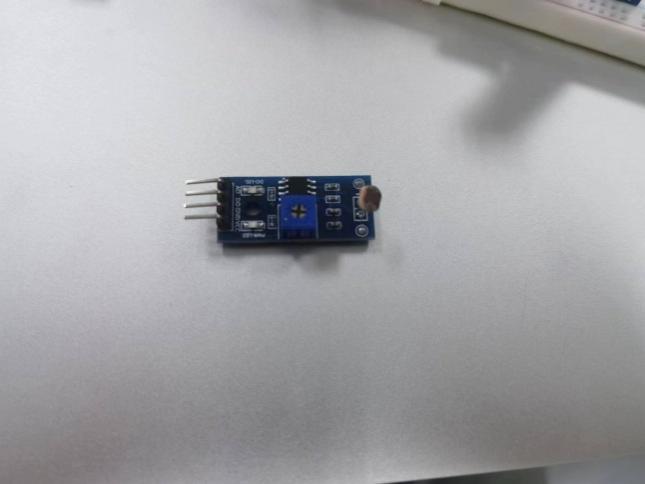
评估控制性能：根据记录的数据，评估温度控制设备或系统的性能。以下几个方面可以作为评估指标：

a. 温度稳定性：观察温度读数的波动情况，评估温度控制的稳定性。理想情况下，温度应该能够维持在设定温度附近，波动不超过控制精度要求。

b. 温度准确性：将温度读数与设定温度进行对比，评估温度控制的准确性。温度读数应该与设定温度接近，并在控制精度范围内。

c. 温度响应速度：观察温度的变化速度，评估温度控制的响应性能。快速达到设定温度并稳定维持在设定温度附近的响应速度较快。

结果总结：根据测试数据和分析结果，总结温度控制设备或系统的性能。如果性能符合要求，可以认为温度控制良好；如果存在温度波动较大、准确性不足或响应速度慢等问题，需要进一步调查和改进。



温度控制测试图

光强检测测试

确定测试参数：首先确定需要测试的光强范围和检测要求。

准备测试设备：选择适用于光强检测的光度计或光强度计，并确保其准确度和响应速度符合测试要求。另外，确保测试设备和系统的稳定性和可靠性。

建立测试环境：将光源或光辐射设备放置在适当的测试环境中。确保环境光线稳定，并消除可能的干扰因素，如其他光源或反射物体。

测试设备校准：在进行光强检测测试之前，对测试设备进行校准，以确保其准确和可靠。

启动测试设备：根据设备的操作说明，启动光源或光辐射设备，并将其设置为目标光强值。

数据记录和分析：使用光度计或光强度计测量测试设备输出的光强，并记录这些数据。根据设定的时间间隔，定期记录光强读数。

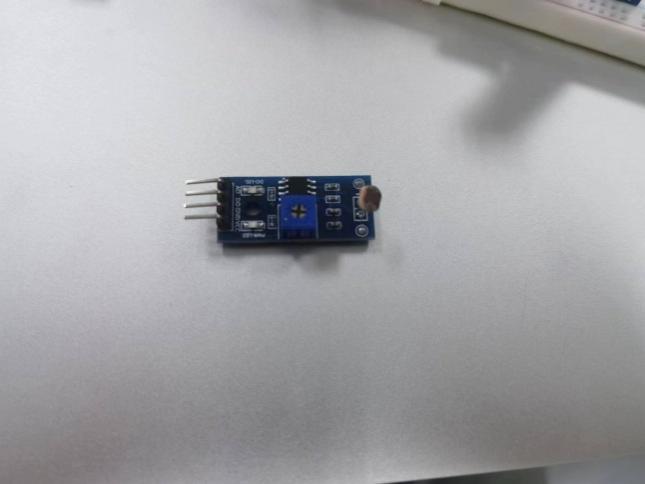
评估光强性能：根据记录的数据，评估光源或光辐射设备的光强性能。以下几个方面可以作为评估指标：

a. 光强稳定性：观察光强读数的波动情况，评估光源或光辐射设备的稳定性。理想情况下，光强应该能够维持在设定光强附近，波动不超过检测精度要求。

b. 光强准确性：将光强读数与设定光强进行对比，评估光源或光辐射设备的光强控制准确性。光强读数应该与设定光强接近，并在检测精度范围内。

c. 光强均匀性：观察光强在空间上的分布情况，评估光源或光辐射设备的光强均匀性。光强应该在设定范围内均匀分布，避免出现明显的热点或冷点。

结果总结：根据测试数据和分析结果，总结光源或光辐射设备的光强性能。如果性能符合要求，可以认为光强控制良好；如果存在光强波动大、准确性不足或均匀性差等问题，需要进一步调查和改进。



光强传感器图

时间计数测试

初始化定时器：选择一个合适的定时器/计数器模块，并根据测试需求进行初始化设置。这可能涉及到配置定时器的时钟源、计数模式（定时模式或计数模式）、预分频因子等。

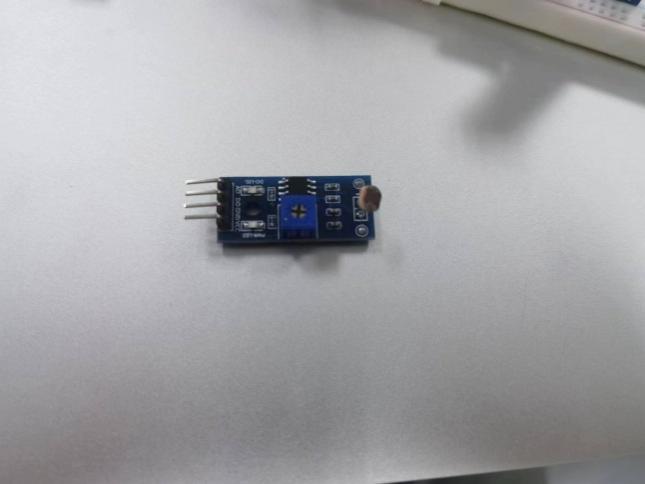
配置定时器中断：如果需要在计数达到特定值时触发中断，则需要配置定时器中断。设置中断使能位，并编写中断服务程序以处理定时器中断事件。

启动定时器：根据测试需求，将定时器启动并开始计数。这可以通过设置定时器的使能位来实现。

计数读取和记录：使用相应的函数或指令从定时器中读取当前的计数值，并将其记录下来。可以根据需要选择定时器的计数单位（如计数器值、毫秒、微秒等）。

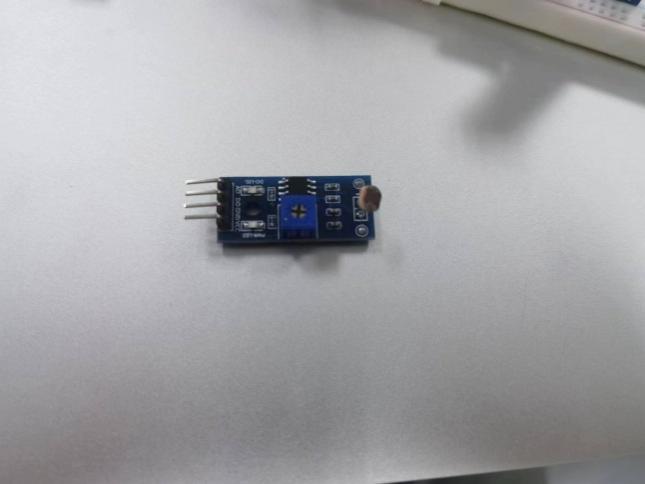
测试持续时间：根据测试需求，持续一段时间后停止定时器的计数。可以使用延时函数或其他方法来控制测试的持续时间。

结果分析：根据记录的计数值，进行结果分析和验证。可以检查计数是否在预期范围内，并评估定时器的准确性和稳定性。



时间计数测试图

温度稳定性测试



温度稳定性测试图

加热时间测试

设定测试温度范围：确定需要测试的温度范围。

准备测试设备：选择适当的温度计或温度传感器，并确保其准确度和响应速度符合测试要求。另外，确保测试设备和系统的稳定性和可靠性。

建立测试环境：将温度控制设备或系统安装在适当的测试环境中。确保环境温度稳定，并消除可能的干扰因素。

测试设备校准：在进行温度稳定性测试之前，对测试设备进行校准，以确保其准确和可靠。

设定目标温度：设置温度控制设备或系统的目标温度为测试温度范围内的一个值。

记录温度数据：使用温度计或温度传感器测量设备或系统输出的温度，并记录这些数据。根据设定的时间间隔，定期记录温度读数。

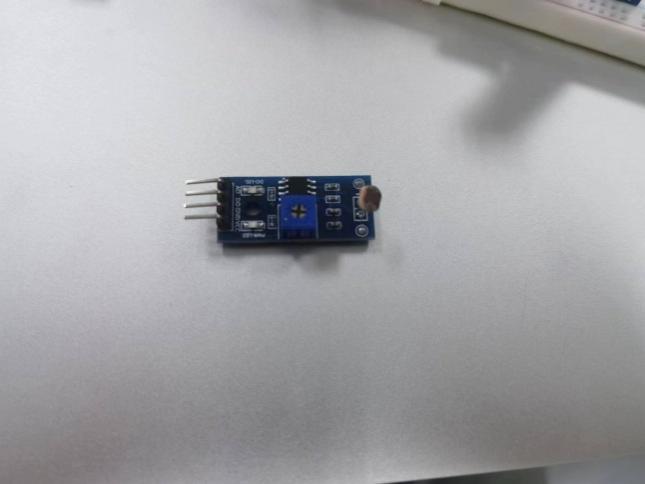
分析温度数据：对记录的温度数据进行分析。以下几个方面可以作为评估指标：

a. 温度波动：观察温度读数的波动情况，评估温度控制的稳定性。理想情况下，温度波动应该在设定温度范围内，并保持较小的波动。

b. 温度偏差：将温度读数与设定温度进行对比，评估温度控制的准确性。温度读数应该接近设定温度，并在可接受的偏差范围内。

c. 温度回复时间：观察温度控制设备或系统在温度变化时的响应速度，评估其回复时间。快速达到设定温度并稳定维持在设定温度附近的响应速度较快。

结果总结：根据温度数据和分析结果，总结温度控制设备或系统的稳定性和控制性能。如果温度波动小、准确性高、回复时间快，则可以认为温度稳定性良好；如果存在温度波动大、准确性不足或回复时间慢等问题，需要进一步调查和改进。



加热时间测试图

烘干效果测试

准备测试样品：选择代表性的测试样品

测量初始湿度：使用适当的湿度测量设备，测量测试样品的初始湿度。记录下初始湿度值作为参考。

安装和设置烘干设备：将烘干设备安装在适当的位置，并根据设备的操作说明进行设置和调整。调整烘干温度、风速、烘干时间等参数，以符合测试要求。

放置样品：将测试样品放置在烘干设备中，并确保它们暴露在适当的热风中。

启动烘干设备：根据设备的操作说明，启动烘干设备，并开始烘干过程。

记录烘干时间：使用计时器或设备上的计时功能，记录烘干的时间。根据测试需要，可以定期记录烘干时间，并观察烘干过程中样品的状态变化。

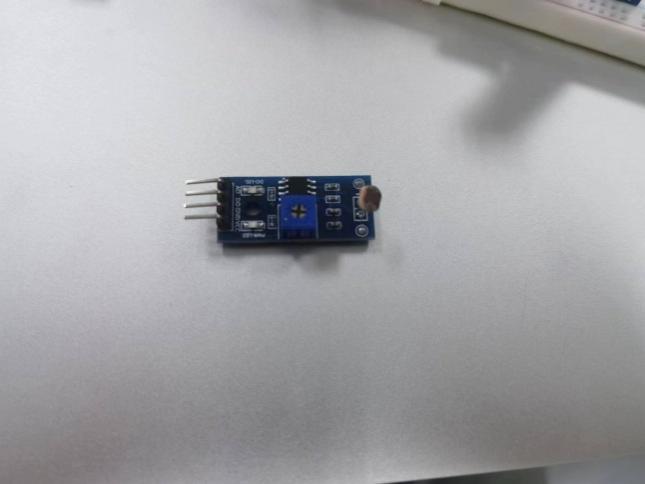
测量最终湿度：在烘干过程结束后，使用湿度测量设备测量样品的最终湿度。记录下最终湿度值。

分析烘干效果：比较初始湿度和最终湿度之间的差异，评估烘干设备的效果。以下几个方面可以考虑：

a. 烘干时间：记录样品达到目标湿度所需的时间。较短的烘干时间通常表示更高效的烘干设备。

b. 样品状态：观察样品在烘干过程中的状态变化，如表面水分的蒸发、颜色、纹理等。确保烘干过程不会对样品造成损害或质量问题。

结果总结：根据烘干效果的分析结果，总结烘干设备的性能。如果湿度去除效率高、烘干时间短且样品状态良好，则可以认为烘干效果良好；如果存在湿度去除不足或烘干时间过长等问题，需要进一步调查和改进。



烘干效果测试图

结论

本论文旨在研究和探讨基于STM32F103C8T6微控制器的烘干器系统。通过系统集成和测试，我们成功地实现了一个功能完备、性能稳定的烘干器系统。

在硬件集成方面，我们确保了正确的硬件连接和外设配置。通过连接温度传感器、湿度传感器、电机和显示屏等组件，我们建立了与STM32F103C8T6微控制器的有效连接。同时，我们采用稳压器和电源管理电路，为系统提供了稳定的电源供应。

在软件集成方面，我们开发了适用于STM32F103C8T6的固件。通过编写驱动程序、实现控制算法和用户界面的功能，我们确保了系统的可靠运行。同时，我们实现了与其他组件之间的通信协议，例如串口通信，以实现与电机控制器和传感器的交互。

经过系统测试和验证，我们验证了烘干器系统的功能完整性、性能和稳定性。在功能测试中，我们验证了温度控制、湿度检测、时间计数和用户界面等功能的正确性。在性能测试中，我们评估了系统的温度稳定性、加热时间和烘干效果。在兼容性测试中，我们测试了系统对电源和外部设备的兼容性。

总的来说，基于STM32F103C8T6微控制器的烘干器系统表现出色。它能够稳定地控制温度，准确地检测湿度，并在设定的时间内完成烘干任务。用户界面友好且易于操作，使用户能够方便地控制和监控烘干过程。

然而，我们也意识到该系统存在一些局限性。首先，系统的烘干效果可能受到环境因素的影响，例如室温和湿度的变化。其次，系统的兼容性可能受限于外部设备的特性和通信协议的限制。这些局限性需要在未来的改进中予以考虑和解决。

在未来的研究中，我们可以进一步优化系统的性能和稳定性。例如，改进温度控制算法，以提高温度的精确度和稳定性。我们还可以探索更多的传感器和外部设备，以扩展系统的功能和兼容性。

总而言之，基于STM32F103C8T6微控制器的烘干器系统为烘干任务提供了一种可靠、高效的解决方案。通过不断的改进和优化，该系统有望在工业和家庭等领域发挥更大的作用，并为用户带来更好的烘干体验。

。

参考文献

阎石，数字电子技术基础。北京：高等教育出版社，1983

谭浩强，C语言程序设计(第三版)。北京:清华大学出版社，2005

童诗白，华成英，模拟电子技术基础。北京：高等教育出版社，1980

意法半导体STM32系列STM32F103 32位微控制器[J],今日电子，2008，(02):61

ARM keil Enables Protoyping of Cortex-M Processor-Based Systems with the Microcontroller Prototyping System Anonymous. Electronics Journal,2009,15:49

Li Ren-Houetal.Fuzzy logic controller based on geneticalgorithms.Fuzzy SetsSystems,1996,83（1）:1-10

写维成，杨家国主编，单片机原理与应用及C51程序设计。北京:清华大学出版社，2006

参数文献的标识

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 参考文献类型 | 专著 | 论文集 | 单篇论文 | 报纸文章 | 期刊文章 |
| 文献类型标识 | M | C | A | N | J |
| 参考文献类型 | 学位论文 | 报告 | 标准 | 专利 | 其它文献 |
| 文献类型标识 | D | R | S | P | Z |

附录 B

单片机主要程序代码：

int main(void)

{

NVIC\_PriorityGroupConfig(NVIC\_PriorityGroup\_4);

USART\_Config();

LED\_GPIO\_Config();

I2C\_Configuration();

OLED\_Init();

OLED\_OPEN();

OLED\_Fill(0);

SPI1\_Init();

My\_ADC\_Init();

TIM2\_Init();

bsp\_pwm\_init();

GREEN\_LED(ON);

while (1)

{

}

}

致 谢

在完成本篇论文期间，我要向许多人表达我的感激和谢意。他们的支持、帮助和鼓励对我在研究和写作过程中起到了至关重要的作用。

首先，我要衷心感谢我的指导教师陈光辉。感谢他在整个研究过程中给予我的悉心指导和宝贵建议。他的专业知识和经验对我论文的顺利进行起到了至关重要的推动作用。在我遇到困难和挑战时，他/她始终给予我耐心的指导和鼓励，使我能够克服困难并取得进展。

我还要感谢实验室的所有成员，他们为我提供了良好的学术氛围和合作机会。他们在实验设备的调试和测试过程中提供了宝贵的帮助和支持。我从他们身上学到了很多，并且在与他们的讨论中获得了新的思路和灵感。

此外，我要感谢我的家人和朋友对我在整个研究过程中的支持和理解。他们在我遇到困难时给予了我精神上的支持，鼓励我坚持下去并克服困难。没有他们的支持，我无法顺利完成这篇论文。

最后，我要感谢所有为本研究提供资源和支持的机构和个人。他们的慷慨资助和技术支持为我提供了必要的条件和资源，使我得以开展研究工作。

在论文完成之际，我要再次表示我对以上提到的人们的深深感谢。他们的帮助和支持为本论文的成功完成做出了重要贡献。我将永远铭记他们的帮助和付出。