图片包含 图示

描述已自动生成

|  |  |
| --- | --- |
| 2025 | 届本科毕业论文（设计） |

|  |  |
| --- | --- |
| **题 目** | **基于STM32的多功能电子万年历** |
|  | **的设计与实现** |

|  |  |  |  |
| --- | --- | --- | --- |
| **学 院** | **大数据与智能工程学院**  **（工业软件产业学院）** | | |
| **年 级** | **2021级** | | |
| **专 业** | **电子科学与技术** | | |
| **学生姓名** | **张伟** | **学号** | **20230781136** |
| **指导老师** | **毛丽利** | **职称** | **硕士研究生** |

|  |  |
| --- | --- |
| **日期** | **2025年5月** |

重庆对外经贸学院  
本科毕业论文（设计）诚信声明

本人郑重声明：所呈交的本科毕业论文（设计）是本人在指导教师的指导下，独立进行研究工作所取得的成果。除了文中特别加以注明的地方外，论文（设计）中没有抄袭他人研究成果和伪造数据等行为。本人完全意识到本声明的法律结果由本人承担。

本科毕业论文（设计）作者签名：

年 月 日

致 谢

盛年不重来，一日难再晨，及时当勉励，岁月不待人。此时此刻，我将为我大学四年本科生涯画上圆满的句号。一路走来，并非容易，非常感恩我青葱岁月的所知、所学、所遇。

首先，我要感谢我本次毕业设计的导师，正是因为你对我细致的指导和严格的要求，我才能设计并完成本次设计，在此期间，我遇到的问题最多，您总是很有耐心的指导我，对于一些特别愚蠢的问题，您一直以温柔和蔼的方式给我讲解，告知我更多问题的解决方案，同时还为我提供了丰富的参考资料的途径，让我可以直接查阅和学习，在您身上，我不仅仅是学习到了专业知识，还看到了为人师表、终身学习的生活态度，我在心里由衷地尊重您，并且在未来会不忘您的教诲，以您为榜样继续学习。

其次，我要感谢四年的同窗们，特别是我的舍友，是你们给我这枯燥乏味的学习生活带来了乐趣，四年的时光中，我们绝大多数的时间一起学习，为我们各自的梦想而奋斗，同时，你们还在我平日的学习生活中有困难的时候，毫不犹豫地伸出援手，替我解答各种难题等，也希望未来你们在各自的道路上，心想事成、前程似锦，我们以后顶峰相见！

最后，我要感谢我的父母，是您们给我提供了一切基础，也是我坚实的后盾，有您们也才有我，在这大学四年里，您们毫无条件的支持我的各种选择，只为我能够健康快乐的成长，给我提供了最好的衣食住行，才能让我没有后顾之忧，大胆拼搏，所以，借此机会，把爱您们的话都写在了本文里。

青春结伴，我已拥有，是感激，是满足，无怨无悔。希望这四年我在乎的人都能够身体健康，在生活中都一帆风顺。

作者：

日期：

摘 要

随着“万物互联”概念的兴起和物联网（IoT）技术的迅速发展，智能家具、全屋控制的理念已经渗透到人们的日常生活中，而电子万年历则凭借其高精度、易查询、多功能等优势作为家用终端设备迅速崛起。本文从时间管理兼远程控制方面进行了进一步的优化，提出了一种具备低功耗、低成本的整改方案。整个系统也是从硬件实现在到软件设计，硬件层面选用STM32作为控制核心，采用AMS1117-3.3的就降压芯片将供电从12V控制到3-5V内给MCU供电，其中RTC时钟则是采用外接一纽扣电池防止断电后时钟数据丢失，也能够精准校准当前时间，年、月、日、时、分、秒以及星期在内的详细时间显示功能，并通过相关算法实现支持节气信息的展示。系统整体采用C语言进行编程开发，通过外接给类传感器如DHT11温湿度传感器、BH1750光照强度传感器等，将实时监控外部环境状态，最终通过显示屏显示给用户，或者通过蓝牙同步到用户手机APP上。手机APP的开发包含了蓝牙通信的建立、系统时间的校准、闹钟的设置和对环境数据的采集，并通过声光等形式来提醒用户。借助搭载的蓝牙模块，此系统也能更灵活的接入其他智能设备的生态网络中，从而降低用户对其的远程控制难度，显著的提升在生活中的便携性和智能水平。

关键词：蓝牙通信；远程控制；传感器；STM32；电子万年历

**Abstract**

With the rise of the concept of "Internet of Things" and the rapid development of Internet of Things (IoT) technology, the concepts of smart furniture and whole house control have penetrated into people's daily lives, while electronic perpetual calendars have rapidly emerged as home terminal devices with their advantages of high precision, easy query, and multifunctionality. This article further optimizes time management and remote control, and proposes a rectification plan with low power consumption and low cost. The entire system is also implemented from hardware to software design. At the hardware level, STM32 is selected as the control core, and the AMS1117-3.3 step-down chip is used to control the power supply from 12V to 3-5V for the MCU. The RTC clock uses an external button battery to prevent clock data loss after power failure, and can accurately calibrate the current time, including detailed time display functions such as year, month, day, hour, minute, second, and week. Relevant algorithms are used to support the display of solar term information. The overall system is programmed and developed in C language. By connecting external sensors such as DHT11 temperature and humidity sensor, BH1750 light intensity sensor, etc., the external environmental status will be monitored in real time, and finally displayed to the user through the display screen or synchronized to the user's mobile APP through Bluetooth. The development of mobile apps includes establishing Bluetooth communication, calibrating system time, setting alarms, collecting environmental data, and alerting users through sound and light forms. With the help of the Bluetooth module installed, this system can also flexibly connect to the ecological network of other smart devices, thereby reducing the difficulty of remote control for users and significantly improving its portability and intelligence level in daily life.

**Keyword:** Bluetooth communication; Remote control; Sensors; STM32; Electronic Perpetual Calendar

目 录

[1 绪论 1](#_Toc194704035)

[1.1 研究背景与意义 1](#_Toc194704036)

[1.2 国内外研究现状 1](#_Toc194704037)

[2 系统整体设计 2](#_Toc194704038)

[2.1 方案分析 2](#_Toc194704039)

[2.2 模块选型 4](#_Toc194704040)

[2.2.1 电子万年历MCU 4](#_Toc194704041)

[2.2.2 传感器模块 4](#_Toc194704042)

[2.2.3 显示屏选择 5](#_Toc194704043)

[2.2.4 通信方式选择 6](#_Toc194704044)

[2.2.5 声光提醒模块选择 7](#_Toc194704045)

[3 硬件电路设计 8](#_Toc194704046)

[3.1 MCU电路设计 8](#_Toc194704047)

[3.2 电源电路设计 9](#_Toc194704048)

[3.3 传感器模块电路设计 10](#_Toc194704049)

[3.3.1 DHT11传感器电路设计 10](#_Toc194704050)

[3.3.2 BH1750传感器电路设计 11](#_Toc194704051)

[3.3.3 按键模块电路设计 11](#_Toc194704052)

[3.3.4 显示模块电路设计 12](#_Toc194704053)

[3.3.5 时钟模块电路设计 12](#_Toc194704054)

[3.3.6 声光提醒电路设计 13](#_Toc194704055)

[3.3.7 蓝牙模块电路设计 14](#_Toc194704056)

[4 系统软件设计 15](#_Toc194704057)

[4.1 编译语言和环境 15](#_Toc194704058)

[4.2 系统程序设计 16](#_Toc194704059)

[4.2.1 主程序设计 16](#_Toc194704060)

[4.2.2 传感器程序设计 16](#_Toc194704061)

[4.2.3 显示程序设计 18](#_Toc194704062)

[4.2.4 声光提醒程序设计 18](#_Toc194704063)

[4.2.5 按键程序设计 19](#_Toc194704064)

[4.2.6 通讯程序设计 20](#_Toc194704065)

[5 系统调试与分析 22](#_Toc194704066)

[6 总结与展望 25](#_Toc194704067)

[参考文献 26](#_Toc194704068)

[附 录 27](#_Toc194704069)

# 1 绪论

## 1.1 研究背景与意义

时间作为一个描述世间万物的变化的计量名词，从古至今，贯穿每个人的一生，从幼年到中年最后到晚年，从最早的日晷到后来的水钟、沙漏、水运仪像台；再到后面的机械时钟、摆钟，在到如今各类常见的石英表，电子钟；从上看来无疑是人们在对待时间的精度上，越精益求精了。而精确的测量和高效的利用，在现如今的社会生产效率的高效启停，与个人生活的有序展开息息相关，在这样的背景下，万年历系统包含对时间呈现的信息，如公历日期、时间、星期及节气等；基于微控制器的万年历系统，凭借其低成本、高效能以及良好的扩展性等特色，近年来吸引了越来越多的关注和研究。

嵌入式技术依靠其体积小、成本低、功能强等特点，适应了智能化发展的最新要求。ARM微处理器资源丰富，具有良好的通用性。Cortex-M3是ARM公司最新推出的第一款基于ARMV7体系的处理器内核。它主要针对MCU领域，在存储系统、中断系统、调试接口等方面做了较大的改进，有别于过去的ARM7处理器；Cortex-M3具有高性能、低功耗、极低成本、稳定等诸多优点，非常适合汽车电子、工业控制系统、医疗器械、玩具等领域[1]。将STM32应用于万年历系统的开发，不仅允许对时间进行精准的记录和显示，同时也支持多种功能模块的扩展，比如温度传感器、闹铃提醒、语音报时等，从而为万年历系统增加了更多实用的选择，满足人们日益多样的需求。

在现如今科学洪流的推动下，各个领域都有卓越的突破，而人们对生活品质的日渐提高的如今，功能单一的日历产品早已被替代，而融入多用途、控制方便、可远程校正等多种人机互动的新型电子万年历渐渐进入人们的日常生活，为人们增添了更多便利和舒适感。

## 1.2 国内外研究现状

早期的实时时钟产品主要通过将振荡器产生的时钟信号进行分频和累加来确定时间，累加计算得出年月日以及时分秒，之后再通过计算机的通讯接口将这些信息传递给处理器进行进一步处理。随着芯片制作工艺技术的进步，内部功能的进一步完善，万年历和时钟功能愈加齐全，各个输出端口的配置也变得更加灵活多样。日本的RICOH公司推出了一种具有时间基准软件自动校准和振荡器静止自动检测功能的RTC芯片，其性能不仅有所提升，而且价格更具竞争力。最新推出的RTC芯片在继承了前几代产品完整功能的基础上，还引入了温度补偿、电源低压检测以及主备电源自动切换等新特性，同时其封装尺寸也有所缩小。

当前，国内外的RTC芯片制造商提供了超过一千种产品，本文对市场上最受欢迎的

国内外超过十家公司，例如Maxim、ST、TI、NXP以及上海贝岭等，推出了多款全新RTC芯片。总体来看，这些新型时时钟芯片具备的功能包括：万年历、闹铃、监控定时器（Watchdog Timer，WDT）、自动处理闰年、12小时和24小时制的转换、低电压监测、备用电源、电气可编程时钟输出以及晶振的温度补偿等。部分芯片，例如NXP公司的某种RTC设计，能够实现厘秒级的计时功能，并提供的100Hz时钟源的抖动大约为3.91毫秒；某款来自Maxim公司的RTC芯片配备了一个容量为4096K的SRAM存储器。众多厂商为RTC芯片增添了许多新的特性，这使得RTC芯片的功能远不止于简单的计时设备，它同时为微控制器（MCU）提供如存储和看门狗等扩展性的功能，

同时，Gajski和Vahid（n.d.）的嵌入式系统设计与建模方法为万年历的分成设计，将整体功能拨丝抽茧般的分成相互关联且又各自独立的抽象层级实现[2]。而《The Art of Designing Embedded Systems》中包含了嵌入式系统开发的详细流程，如硬件设计、软件开发、调试技巧和项目管理等内容。Ganssle，Jack通过丰富的开发经验撰写了如何在有限资源空间下优化系统性能并且提高系统可靠性的方法，为本次设计提供了全面的设计思路和实用技巧[3]。

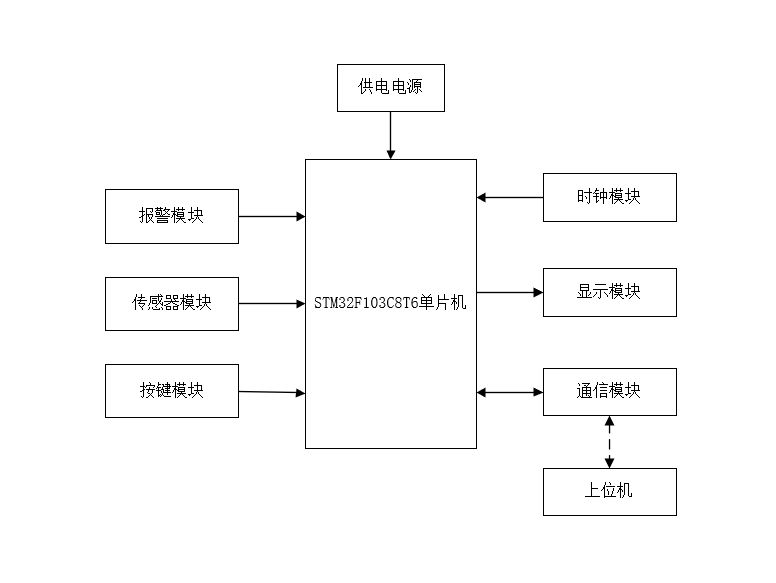
# 2 系统整体设计

## 2.1 方案分析

本项目以软硬件结合的方式，选择C语言作为程序硬件编码语言，以STM32单片机作为核心控制板，在数据传输节点上连接温湿度传感、光照传感器，分别对环境的温湿度、光照情况进行实时监测，结合RTC时钟模块来校准当前时间、显示当前时间，包括年月日、时分秒、星期等信息，且支持闹钟设定，当到达闹钟后，进行声光报警。通过无线通信技术，无需人为介入，数据可实时传输至管理终端，实现万年历的全天候监测，进而达成完全自动化的智能管理效果。此系统包含多个关键模块，如主控制单元、传感器组件、按键界面、时钟单元、显示屏、通信接口以及报警装置。

其中主控模块作为像人大脑一样的存在负责各个器件的协调和数据的处理与运算，并将处理结果传达给受控器件输出给用户，而用户也能操作APP或外部按键来控制其工作状态，增加程序的互动性。

因此，日历系统每年的整体框图像显示在图2.1中。

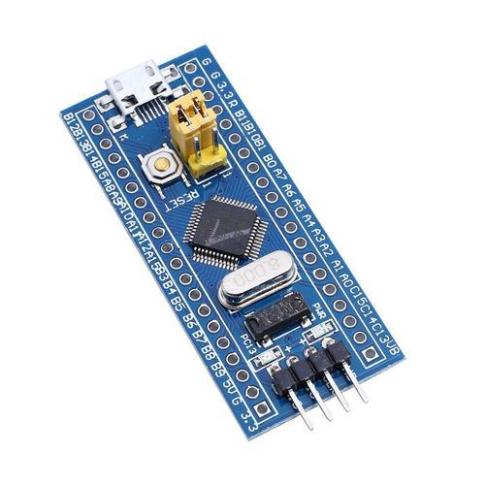


**图2.1 程序总体框图**

## 2.2 模块选型

### 2.2.1 电子万年历MCU

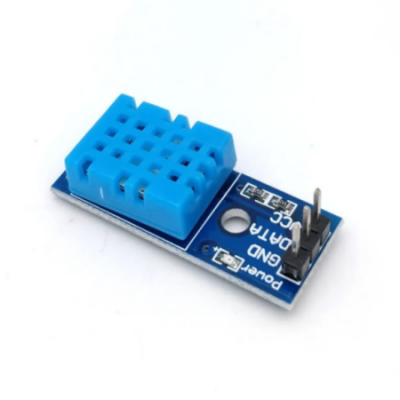
此次设计采用STM32F103C8T6为主控芯片，它是采用ARM Cortex-M3架构，具备超高的运算能力，能够快速响应处理各种中断。芯片的工作频率可达到72MHz，支持多种外设的连接，包括ADC、DAC、I2C、SPI、USA以及定时器等，除了具备丰富的外设支持外，芯片还提供额外的接口，以便连接更多的外部设备和存储单元；此外，STM32F103C8T6还具备低功耗特性，支持多种低功耗工作模式，包括待命模式、休眠模式和停机模式，MCU可见图2.2。



**图2.2 主控模块STM32F103C8T6实物图**

### 2.2.2 传感器模块

所谓传感器，是一种可以对物体的某种性质进行检测的设备，它是一种可以将被测物体的某些基本参数，依据确定的精确度，转化成为拥有相应关系的设备[4]。DHT11传感器是用于测量环境中温度和湿度，其具有很好的湿度和温度的分辨能力，二者的分辨率均为8位。至于湿度，测量的准确度为±4%RH，其响应时间在6到16秒之间，在温度方面，其精度为±1℃，可测量的温度范围为0~50℃，响应时间为6~30s，同时在1秒内能够完成一个采样周期。但由于其硬件设计上为单总线传输，所以就必须要使得在数据传输期间不会被其他中断事件所打断，DHT11传感器的数据格式是由8位的整数和小数构成，所有可以直接显示或者进行数据处理。

****

**图2.3 DHT11传感器图**

BH1750传感器的内部结构包括光敏二极管、运算放大器、模拟到数字转换器（ADC）及晶振等元件。PD二极管利用光生伏特现象，能够将光信号转化为电信号，经过运算放大电路的增益处理后，由模数转换器将电压采集。随后通过逻辑电路将其转化为16位二进制数，并存储于内部寄存器中（注：进入光窗内的光线强度越强，产生的光电流和电压亦随之增大，因此可以通过电压的变化来评估光照强度。BH1750提供了时钟线和数据线，单片机可通过I2C协议实现与BH1750模块的通信，能够选择该模块的工作模式，并提取光照度的注册数据。如图2.4所示。



**图2.4 BH1750传感器图**

### 2.2.3 显示屏选择

在本万年历系统中，显示组件作为其硬件输出装置，需要展现出清晰、全面且富有动态感的效果。在硬件显示技术中，常见的显示器件种类繁多，例如发光二极管（LED）、LCD1602液晶屏幕和OLED显示屏等。以下是两个方案。

（1）LCD液晶显示屏。其中包含TFT（TFT Thin Film Transistor）指薄膜液晶显示器。ILI9320液晶控制器自带大小为172820(240\*320\*18/8 )26万色的显存。模块各个像素均由红绿蓝三种颜色混合而成,16位数据线与显寸的对应关系高达565种[5]。这种显示器具有平面结构，被视为一种较为传统的显示技术，通过液晶材料的分子排列方式在电场作用下发生改变，以调制周围的光线，从而实现信息的展示。其使用的技术相对较为陈旧，厚度较大，显示的对比度不够明显，同时也存在延迟，界面反应速度无法满足按键切换的要求，因此已经在市场上逐渐被淘汰。

（2）OLED显示屏。这种设备在近年来被广泛应用于硬件程序中，由丰富的有机物质构成，具备鲜明、清晰的显示效果，并且界面相当精致。并且在操作中的响应速度与按键切换等方面保持一致，展现出高效和长期使用的优势特性。

因此，在此万年历系统中，决定使用一款0.96寸的OLED显示器作为数据输出终端，能够实时动态地展现所有信息，让用户直观地观察到数据的变化。如图2.5所示。



**图2.5 OLED显示屏图**

### 2.2.4 通信方式选择

本项目采用蓝牙实现与移动设备之间的数据交换，这能满足用户远程监控的需求，确保万年历能够与其他设备连接，增加系统的灵活性与易扩展性。在选择通信方式时，本文探讨了三种方案：ZigBee、WIFI和蓝牙，对比它们的特点，以最优的方案选择通信模块。

在数据传输速度方面，ZigBee的速率仅介于10~250kbps之间，而一般蓝牙的速率则在1Mbps到2Mbps，WiFi的传输速度则明显高于ZigBee和蓝牙，其基于802.11ax标准的最高速率可达到10Gbps。只有具备高传输速度的技术，才能够支持远程监控的实时进行，并满足大文件、视频和音频的在线处理及下载需求。

传输距离方面，通常情况下，ZigBee与蓝牙可实现10至100米之间的数据传输，而WIFI的覆盖范围则远超前者，达到300米，十分适合于进行远程监控。

在能量消耗方面，ZigBee和蓝牙是能够实现低能耗的，而WiFi则为了获得高数据传输速率和较远的通信距离，其能耗表现不及前者。在本设计中，电子万年历主要应用于家庭环境，所需的通讯距离并不需要像WiFi那样远，但对功耗的要求则相对较高。在这三种技术中，蓝牙的费用相对较低，且能够很好地满足低功耗和相对较大的数据传输需求。正是由于蓝牙技术的低成本、低功耗、高速率、高可靠性和兼容性等特点，使得基于蓝牙技术的智能设备能广泛的被人们接受[6]。

因此本次设计选择蓝牙无线通信方式。如图2.6所示。



**图2.6 HC-05通信设备图**

### 2.2.5 声光提醒模块选择

当设定的时间到达时，必须开启监控程序以启动报警，起到提醒的效果。本项目中采用了常见的声光报警设备，包括蜂鸣器和LED灯，以实现报警功能。在选择蜂鸣器时，可以考虑两种类型：无源蜂鸣器和有源蜂鸣器。

（1）无源蜂鸣器：此类蜂鸣器内部没有发声装置。电流信号经过蜂鸣器，它不会自动发出声音，需要通过方波信号并调整其频率来使无源蜂鸣器发出声音。由于缺乏内部振荡源，往往成本低于有源蜂鸣器，但是它的响应速度却不如有源蜂鸣器快。

（2）主动蜂鸣器：内部带又震荡器，被触发后，其内部的震荡源便会以固定的频率发出声音，从而实现报警功能。主动蜂鸣器的控制程序设计简单，易于操作，且声压较高，是一种非常理想的报警装置选择。

总之，系统选择主动蜂鸣器报警。如图2.7所示。



**图2.7 有源蜂鸣器图**

# 3 硬件电路设计

## 3.1 MCU电路设计

该电子万年历的核心控制单元采用的是STM32F103C8T6型号的微控制器，从图3.1可以看出，其内部集成了多种电路，包括电源模块、复位模块、ADC转换模块以及时钟模块等。电源部分负责为整个系统提供必要的能量，关于STM32F103C8T6芯片，规定的供电电压为3.3V，连接在38号引脚上。由于某些外设需要5V的电源，18号引脚接入5V电源后，通过内置的稳压器将其转化为3.3V，以满足STM32F103C8T6芯片的电源要求；STM32最小系统需要搭建好两个晶振模块，分别为32768HZ的低速晶振和8MMZ的高速晶振。然后接入复位电路，用于实现系统的复位功能。最后接入JTAG电路，就可以进行程序的下载和硬件仿真了[7]。复位电路负责为程序执行提供初始化的功能，若需重启系统，用户可通过复位电路上的按钮来进行相应操作，从而实现整个程序的初始化；ADC转换电路的作用是将传感器所采集的数据处理成可以被STM32F103C8T6芯片理解和计算的数字信号；该时钟系统中集成了一个振荡芯片，能够对程序执行期间进行时间管理，以确保程序按照预定的时间流程运行。主控模块的集成芯片展现出一体化、简化和灵活的特性，电路中的其他引脚和接口负责与执行器、显示单元、传感器等设备进行连接，以实现信号的传输与交流。



**图3.1 主控模块电路设计**

## 3.2 电源电路设计

由于各种传感器模块及MCU都需要稳定的3.3V或5V电源，来确保它们的内部电路能够正常工作。而常见的电源输入方式通常是通过计算机的USB接口或者是充电器头（电压约5V），所以电路设计时是采用一个降压芯片将5V电压降至3.3V。例如，低压降稳压芯片ME6211C33或者AMS1117-3.3等。AMS1117-3.3提供的电压输出包括1.2V、1.5V、1.8V、2.5V、3.3V和5.0V。芯片的热关断特性可以有效避免在过载或高温的情况下导致芯片损坏。

在电路中添加一个22uF的旁路电容以减少噪声，通常而言，线性稳压器的稳定性会随着输出电流增大的情况下逐渐降低。LDO有显著的低压降特性，并且能提供高精度的电压输出，尤其是在输入和输出电压差异较小的条件下，仍能维持稳定，输出精确的3.3V电压。

为了方便观察电源电路的运行状态，在电源电路中加入一个发光二极管（LED），经过一个限流电阻与AMS1117所输出的3.3V电压相串联连接。指示灯在电源正常的情况下会发光，表明电源线路运作正常。其电路设计如图3.2所示。



**图3.2 电源电路设计**

## 3.3 传感器模块电路设计

### 3.3.1 DHT11传感器电路设计

DHT11传感器用于采集环境中的温度和湿度，该传感器由电容感应元件与数字处理单元组成。电容感应元件包含一个热敏电阻和湿度的敏感装置，其中，热敏电阻其阻值与周围环境温度成反比，环境温度升高时，电阻值会下降；湿度敏感元件能够根据空气中水分的含量发生膨胀，并根据其膨胀的程度来计算出相应的电阻值，从而反映当前环境的湿度。DHT11传感器配备了一个16位的AD转换器能够将电信号转化为MCU处理的数字信号，能够灵活的监测环境的温度和湿度变化。在设计的电路中，该传感器含有电源引脚与数据引脚，其中第一引脚连接至5V电源，第三引脚接地，以形成一个完整的闭合电路，而第二引脚则作为数据输出端，以单总线传输方式将采集到的温湿度数字信号到单片机的PA4接口。与此同时，单片机也通过此引脚向传感器发送控制指令。其电路图如下图所示。



**图3.3 DHT11传感器电路设计**

### 3.3.2 BH1750传感器电路设计

BH1750传感器用于光照的采集。当光线射向该传感器时，其中的光子能量会激发内部的电子，使其发生跃迁，从而产生一定量的电流。这个电流会对应产生特定的电阻值。当没有光线照射时，光敏电阻的阻值相对较高，而在感应到光线时，阻值则会下降。因此，光照强度与阻值呈现一种反向关系。在BH1750传感器的工作电路中，通常会需要设计一个偏置电路，以实现电压分配，从而提高该传感器的响应速度。该电路的设计如图3.4所示。在该电路中，正极引脚（引脚1）连接到5V电源，而引脚4（负极引脚）则接地。同时，引脚2和引脚3分别作为模拟输出引脚（AO）和数字输出引脚（DO），其中AO引脚负责输送光照强度的模拟信号到单片机的PA10端口，而DO引脚则向单片机的PA9端口传递数字信号，默认状态为高电平。



**图3.4 BH1750传感器电路设计**

### 3.3.3 按键模块电路设计

本设计将采用比较经济的独立式键盘，独立式键盘每个键单独占用一根I/O口线，它们彼此之间相互独立互不干扰[8]。该按键模块由五个不同的按键构成，每个按键负责不同的指示任务，通过按键的状态变化来控制程序电路的运行。然而，在未按下按键时，按键内部的接触点并未与程序电路建立连接，因此，当前的按键电路处于关闭状态；相反，若按键被激活，其内部触点将与程序电路发生连接，随即改变该电路的状态至闭合模式，并执行相应的指令，将信号传递给微控制器。微控制器将再次处理和响应该信号。其电路设计如图3.5所示。按钮1至按钮5通过接口PB12、PB13、PB14、PB15和PB1分别与主控电路相连，实现数据的传递。



**图3.5按键模块电路设计**

### 3.3.4 显示模块电路设计

此系统选用了0.96英寸的OLED显示器作为程序输出界面，其电路布局见图3.6。此显示器模块的供电电路连接至VCC，结构设计包括几个部分；该OLED显示屏配备了驱动电路，使得其像素能够发光，采用的是薄膜晶体管（TFT）作为其驱动器；并且通过I2C与主控模块的PA11、PA12接口连接；从而实现了程序显示在液晶屏上的功能。

而I2C总线只需要由两根信号线组成，一根是串行数据线SDA，另一根是串行时钟线SCL在系统中，C总线的典型接法如图2所示，其中Master和Slave分别代表主、从机，Rp是为了保证具有I2C总线器件的SDA和SCL漏极开路的上拉电阻,注意连接时Master和Slave需要共地[9]。

图示

AI 生成的内容可能不正确。

**图3.6 显示模块电路设计**

### 3.3.5 时钟模块电路设计

实时时钟（Real-Time Clock，RTC）模块用于在设备失去电力或重启后继续保持时间的准确性，包括日期、小时、分钟和秒钟的信息[10]。许多RTC具备报警（Alarm）功能，能够在设定的时刻触发中断，用以定时唤醒设备或执行其他操作。该功能还能用于安排系统的定期任务，比如每天执行日志备份或实现设备的自动开启与关闭。RTC具有在设备处于低功耗模式或失去电源后，根据预设的时间自动激活系统的功能，从而帮助降低能耗。在日志记录或传感器采集过程中，RTC能够提供高精度的时间标记，从而提升数据的管理和分析效率。RTC模块主要由以下部分组成：

（1）振荡器：一般采用高精度晶体振荡器，负责生成稳定的时钟信号，为RTC模块提供时间基准。

（2）计数装置：依据振荡器发出的时钟脉冲进行时间的统计，以便监测秒、分钟、小时、日子、月份以及年份等时间单位。

（3）记忆单元：专为记录当前时间、日期及可能的提醒设置等信息而设计。即便是当主电源关闭，这些数据依然会由备用电池维持。

控制逻辑负责接收外部设备传来的指令，例如时间读取、时间设定以及闹钟配置等，同时也管理计数器和存储器的操作。

接口电路的功能在于使外部设备（如微控制器）能够互相交流，通常通过I2C、SPI等协议实现。

此次选择了内部集成于STM32中的实时时钟模块，其电路布线如图3.7所展示。VCC引脚作为主电源连接了3.3V的电压，而GND引脚则用于接地，以此实现时钟模块的闭合电路。VBAT引脚则是备用电池的连接点，当主电源关闭时，能够维持时间和日期的正常运作。



**图3.7 时钟模块电路设计**

### 3.3.6 声光提醒电路设计

该系统的声光模块使用了主动式蜂鸣器和LED灯，主动式蜂鸣器内部集成有震荡电路及放大器电路。在此架构中，震荡电路的功能是生成声音信号。当电流通过主动式蜂鸣器时，其内设的振荡器会发出一个恒定的震荡频率，这样就能产生声音，起到推动蜂鸣器的作用。当发声装置输出音频信号后，该信号会经过放大器电路进行增强，以便生成警示音效。如图3.8所示，声光模块中所设计的蜂鸣器电路利用三极管8050作为增幅元件，并联接了适当的电阻。电源为蜂鸣器提供5V电压，并与主控模块的PA8接口相连接，以实现电流信号的有效传输。



**图3.8 声光模块蜂鸣器电路设计**

蜂鸣器发出提示的同时，LED灯也会闪烁，其驱动电路如图3.9所示。为了避免短路和确保LED灯能够正常亮起，LED灯与电阻成串联关系，并通过主控模块的PC13端口进行数字信号的传输，以为LED灯输入相应的电平。在信号输入完成后，会启动延时机制，从而实现LED灯的点亮或者关闭。



**图3.9 声光模块LED灯电路设计**

### 3.3.7 蓝牙模块电路设计

HC-05蓝牙模块是一款依据蓝牙协议设计的简易无线通讯设备。该模块采用BC417单芯片蓝牙集成电路，符合蓝牙v2.0标准，同时支持增强数据传输速率（EDR）技术，能够达到调制速率在2Mbps至3Mbps之间。其内部配备了2.4GHz的天线，用户无需进行额外调校，同时还具有自适应跳频功能，有效减少了干扰，确保了通信的可靠性[11]。HC-05模块支持两种模式，分别是命令响应模式以及自动连接模式。在命令响应工作模式中，模块能够执行AT指令，使得用户可以配置控制参数或下发控制命令；而在自动连接模式下，模块可以以主控（Master）、从属（Slave）或自环（Loopback）的方式进行信息传递。通信模块HC-05的电路设计如图3.10所示。



**图3.10 蓝牙模块电路设计**

# 4 系统软件设计

## 4.1 编译语言和环境

选择编程语言时，C语言[12]因其在高效性和简洁性方面的表现而受到青睐。这种语言不仅能够直接与硬件进行交互，还提供了结构化编程的支持。它拥有多样的运算符和数据结构，并且通过指针机制，使得内存管理变得更加灵活和精细。C语言展现出高度的跨平台能力，可以在不同的计算机系统上进行编译。此外，它还支持反汇编，其运行效率几乎可与用汇编语言编写的程序相媲美。

Keil5是由Keil公司特别为嵌入式系统的开发人员设计的一款综合性开发平台（IDE）。嵌入式系统开发该软件能够兼容多种微控制器，尤其是ARM架构的芯片，使得进行硬件开发的过程变得更加高效。MDK-ARM作为一部分，其包含了工业标准的C语言编译器、调试工具、宏汇编器与实时操作系统等重要元素，这些元素为工程师们提供了一套高效且稳定的开发工具链。Keil[13]具有强大的代码编辑和编译功能。它还配备了强大的代码编辑器，允许用户进行在线编程、编译、模拟和调试，能够快速识别并修复代码中的问题。在这个集成开发环境中，用户能够直接撰写C语言代码，并借助内置的编译工具将其编译为可在目标硬件上执行的应用程序。凯尔还拥有许多调试工具。此环境还配备了源代码调试工具，能够模拟多种MCU硬件平台，使得开发者在编写代码时能够进行实时的调试工作。通过设置调试点，程序员能够精准找到代码中的问题，从而提升开发效率。

VSCode[14]是一款轻便且开放源代码的跨平台开发工具，能够在Windows、macOS及Linux操作系统上运行。它拥有丰富的插件体系，能够兼容多种编程语言，如JavaScript、Python和C++等。VSCode具备智能代码补全、语法着色、调试工具，以及与版本控制系统的无缝集成。其灵活多变的界面和功能配置能够让开发者根据需求优化他们的工作空间。此外，VSCode同样具备与Git的合作、终端功能，还表现出快速的启动速度和优异的运行效率。

## 4.2 系统程序设计

### 4.2.1 主程序设计

在此次设计中，main.c文件负责各功能模块的初始化工作，并为每个设备的I/O端口进行设定，所有功能代码被封装以便运行。程序启动后，用户需在手机上开启APP，搜索蓝牙HC-05并输入相应的密码，以建立手机APP的连接并启动各项功能模块。传感器则用于监测环境中的温度、湿度及光照强度。在监测过程中，用户可以通过按键或APP手动调整当前时间并设定闹钟。所有处理后的数据将通过OLED显示。当设定的闹钟时间到达时，报警模块将被激活以提示用户。采集的数据通过单总线方式传输至单片机，单片机接收后对其进行整理和分析，以实现声光警示的功能。主要程序过程设计如图4.1所示。



**图4.1 主程序设计流程图流程图**

### 4.2.2 传感器程序设计

该传感器模块用于获取温度和湿度以及光照信号，并经过一系列的放大、滤波和数模转换，最终将这些信号转化为数字信号。首先，MCU对DHT11传感器和BH1750传感器进行控制以使其启动。在启动后，传感器内的应变电阻获取相应的电压，通过内部处理将其转化为电信号，最终通过引脚传输转换后的数字信号给MCU进行计算和分析。

在这其中，AD模块的关键作用是将传感器输出的模拟信号转化为主控芯片可以处理的数字信号。在STM32微控制器中，AD转换模块通常是集成在芯片内部的，因此无需通过外部引脚进行扩展连接。这使得可以直接在寄存器中进行模拟信号的读写并完成转换，同时，STM32的IO端口具备多路选择开关的复用特性，从而提升了单片机的整体运行效率。

模拟信号的数字化处理过程可分为三个主要环节：首先进行采样，接着是量化，最后进行编码。首先，获取一段时间内连续的模拟信号，以便生成一系列离散样本；然后，将这些连续出现的模拟信号的幅度值逐一转化为离散的数字形式；最后将得到的离散样本数值转化为单片机能够识别的二进制格式，以完成编码过程，从而实现模拟信号向数字信号的转变。

传感器模块的程序设计如图4.2所示。



**图4.2 传感器模块程序设计流程图**

### 4.2.3 显示程序设计

本系统的显示组件采用了OLED显示技术，利用半导体材料发出光线，以实现显示效果，其工作原理如图4.3所示。一旦程序供电，电流便会分别从OLED显示屏的正负两个极点注入至其内部的有机半导体材料，形成激子并有规律地移动，以便找到自己的位置，最终以光子的形式释放能量，从而让OLED显示屏发出相应的光亮。

OLED本身是没有显存的，它的显存是依赖于SSD1306提供的而SSD1306提供一块显存。SSD1306的显存总共为128\*64bit大小，SSD1306将这些显存分为了8页。每页包含了128个字节，总共8页，这样刚好是128\*64的点阵大小。而其显示原理也就是修改MCU缓存（实际上就是SRAM），在修改完了之后，一次性把STM32上的缓存数据写入到OLED的GRAM。最终将数据在OLED上展示出来。



**图4.3 显示模块程序设计流程图**

### 4.2.4 声光提醒程序设计

一旦达到用户设定的闹钟时刻，微控制器将激活报警程序，通过电流驱动蜂鸣器，启动声音震荡装置，随即经过放大器处理后产生报警音，同时向LED灯发出高电平信号，从而实现声光警示效果。而对于实现的效果可以通过PWM（脉宽调制）可以用来控制蜂鸣器的音量和LED的亮度，通常通过调节PWM信号的占空比来实现。蜂鸣器的音量和LED亮度与其供电电压的平均值相关，而PWM信号通过快速切换电源开关的方式改变供电的平均电压，从而影响音量跟亮度。占空比越高，提供的平均电压越高，蜂鸣器的音量和LED的亮度也就越大。报警模块的软件工艺设计如图4.4所示。



**图4.4 报警模块软件流程设计**

### 4.2.5 按键程序设计

按键作为用户与设备的交互界面，使得对程序硬件进行数据输入成为可能。当用户按键时，内部的触点会与电路接触，形成闭合电路，发送相应的指令给微控制器，从而产生相应的反应。在这个系统中，按键的功能包括校准时间、设置闹钟和关闭警报提示等。而在软件实现上通过扫描按键端口状态，来判断按键是否被按下，通过不同的按键状态来实现与之对应的模式切换与时间设置，首先需要读取按键的当前状态，判断按键是被按下、释放还是没有变化。通过设计状态机来管理按键的状态转换，比如从按下到释放的变化。当按键发生状态变化时，要进行防抖处理，以避免因按键的机械抖动导致错误的多次触发。最终，根据不同的按键状态，执行相应的操作，如响应按键按下、松开等事件，确保按键扫描的准确性和稳定性。按键模块的程序设计如图4.5所示。



**图4.5 按键模块程序设计流程图**

### 4.2.6 通讯程序设计

实现APP与万年历之间的连接可以通过以下五个步骤和方法来达成。首先，需要通过UART通信协议建立HC-05模块与主控MCU之间的连接。当手机APP与MCU通过PIN（个人识别码）建立连接后，手机APP能够通过SDP（服务发现协议）协议，来选择将MCU所采集到的数据通过那种服务来实现，而这些数据包括传感器测得的数值以及设备的运行状态[15]。在本设计中，APP能够提取温度、湿度和光照等传感器数据。在数据采集的过程中，需遵循APP设定的触发条件和时间间隔，方能确保数据的可靠性和精确性。

APP在从MCU获取监测信息后，这些数据尚不能立即使用，而需经过一系列处理步骤，例如数据的解析、转换和滤波等，只有在这些步骤完成后，才能进行远程监控或预警控制等操作。

此外，移动应用程序还具备向MCU发送指令的能力。在这一实施环节中，需依赖工业控制算法，并将控制软件从上位机下载到下位机的存储装置中，以满足远程监控的需求，例如进行时间同步和设定闹钟等功能。

通信模块的软件过程如下图4.6所示。



**图4.6 通信模块程序设计流程图**

# 5 系统调试与分析

5.1 系统硬件调试

本万年历系统的调试主要用于对代码的编译与执行，首先需要检查代码中是否存在缺陷或错误，并对发现的问题进行修正，直到所有Bug都被消除。代码调试完成后，依据代码进行硬件设备的操作，以确认能够有效实现环境温湿度、光照的监测，以及时间校准和闹钟功能的设置等操作。实验结果如下表5.1所示。

**表5.1 系统硬件调试结果**

|  |  |  |
| --- | --- | --- |
| 调试用例 | 预期结果 | 实际结果 |
| 主控模块 | 能够成功发送、接收程序指令，并控制各连接外设，以及可以成功计算处理程序数据 | 能够成功发送、接收程序指令，并控制各连接外设，以及可以成功计算处理程序数据 |
| 传感器模块 | DHT11传感器能够成功检测环境温湿度信号，BH1750传感器能够成功检测光照情况，并完成数模转换传输至主控模块 | DHT11传感器能够成功检测环境温湿度信号，BH1750传感器能够成功检测光照情况，并完成数模转换传输至主控模块 |
| 按键模块 | 按下按键时，程序能够作出正确的输出结果 | 按下按键时，程序能够作出正确的输出结果 |
| 显示模块 | 启动程序后，OLED液晶屏会显示正确的字符串，包括采集到的环境信息和时间等 | 启动程序后，OLED液晶屏会显示正确的字符串，包括采集到的环境信息和时间等 |
| 时钟模块 | 可以对当前时间进行校准、设定闹钟 | 可以对当前时间进行校准、设定闹钟 |
| 通信模块 | 数据成功传输至上位机，手机App可以查看到正确的数据等信息 | 数据成功传输至上位机，手机App可以查看到正确的数据等信息 |
| 声光提醒模块 | 当到达所设定的闹钟后，驱动蜂鸣器和LED灯进行声光报警提示 | 当到达所设定的闹钟后，驱动蜂鸣器和LED灯进行声光报警提示 |

5.2 系统软件调试

本次针对万年历设计的软件调试，主要聚焦于程序功能特点的用例测试。看是否可以通过相关接口如按键、APP等来控制硬件功能的实现，而这其中最为关键的是测试蓝牙是否能与其APP建立通信连接，这些测试涵盖了数据查询和闹钟设置两个核心功能。数据查询是通过硬件搭载的蓝牙模块通过一定形式将数据同步到APP，APP通过数据组包格式解析并呈现在APP中，同理，闹钟设置只是主从关系颠倒一下，将APP中接受的用户输入信息发送给硬件，通过MUC处理后在OLED中显示出来、控制LED和蜂鸣器在到达时间后做出对应的警示，同时，这些软件测试也可帮助确认系统通信模块的工作是否正常，只有当通信模块有效，软硬件方能实现互相配合。并且，也能借机检测系统稳定性如何，是否存在数据传输量大而导致系统相关的堆栈溢出，导致系统崩溃从而影响用户体验。软件调试结果如下表5-2所示。

**表5.2 系统硬件调试结果**

|  |  |  |
| --- | --- | --- |
| 调试用例 | 预期结果 | 实际结果 |
| 数据查询模块 | 手机APP与程序连接成功后，在首页可以查看到温湿度、光照、当前时间，同时可以进行时间的一键校准 | 手机APP与程序连接成功后，在首页可以查看到温湿度、光照、当前时间，同时可以进行时间的一键校准 |
| 闹钟设置模块 | 手机APP与程序连接成功后，可以通过滑动按钮来开启闹钟，可以分别输入三个不同的时间作为闹钟，输入完成后改变报警模块的驱动条件 | 手机APP与程序连接成功后，可以通过滑动按钮来开启闹钟，可以分别输入三个不同的时间作为闹钟，输入完成后改变报警模块的驱动条件 |

5.3 实物测试

本万年历系统的软件设计部分涵盖了多个模块，包括MCU、蓝牙组件、显示模块、温湿度传感器模块、蜂鸣器模块和按键模块等。MCU能够针对不同的模块采用多种通讯方式来制定通讯协议，并实现对各种外设指令的发送和接收，同时进行相应的数据逻辑运算和处理；通过连接主控模块的接口引脚，能够完成数据的传送与控制，成功地获取并转换环境中的温度、湿度以及光照强度；

按键模块由五个按钮构成，每个按钮执行不同的操作，按下按钮会闭合电路，实现电流的流动，从而完成用户与设备的交互，使得用户能够设定时间和闹钟；该设备采用OLED屏幕来呈现当前时间、闹钟及监测数据等信息；报警系统由一个有源蜂鸣器和LED指示灯组成，MCU通过调节电流来驱动蜂鸣器一旦达到预设的闹钟时间，同时向LED指示灯发送高电平信号，以实现声光报警的效果；

时钟部分采用RTC模块，能够校准当前的程序时间，并负责程序的计时功能；最终通过蓝牙将数据传输至手机应用，实现了系统软硬件之间的互联，为用户提供了远程监控的基础。该系统通过前端与后端的轮流检测来对各个模块进行任务的安排，依据设定的中断优先级优先处理高优先级的信号。最终通过APP交互实现用户远程操控电子万年历从而实现对应的功能。

综上所述，下表展示了本万年历系统的硬件与手机APP界面。

**表5.3 系统硬件实现图**

|  |  |
| --- | --- |
| 系统数据同步 | 闹钟设置 |
| 1741694829126 | e494f47b3fe4a0ef960f4a919d1f331 |

本系统的上层软件以移动应用的形式进行展示。该系统具备数据查询和闹钟设置功能，使得用户能够进行24小时的远程监控。首先，通过手机应用程序与设备进行蓝牙连接，实时同步硬件所采集的数据。在界面的顶部，用户可以查看到温度、湿度、光照等指标的当前状况，以及实时的时间信息。在查看当前时间的同时，还可以进行一键校准，并且能够为硬件设置三个不同的闹钟。软件功能实现如表5.4所示。

**表5.4 系统软件功能实现图**

|  |  |
| --- | --- |
| 系统数据同步 | 闹钟设置 |
|  |  |

# 6 总结与展望

论文针对市面上的数字时钟进行考察调研，了解实际需求，提供了一套解决方案，分别从其课题研究背景与意义、国内外研究现状、以及所涉及关技术和主要内容进行详细介绍与设计。把该系统划分为两个不同设计主体，一方面为硬件控制主体，通过C语言来编码实现，以STM32开发板为核心控制器，在各数据节点上连接DHT11传感器、BH1750传感器，分别检测环境温湿度和光照强度，结合RTC时钟模块来校准当前时间、显示当前时间，包括年月日、时分秒、星期等信息，且支持闹钟设定，当到达闹钟后，进行声光报警。另一方面为远程控制主体，通过蓝牙通信技术进行连接，让用户能够在一定范围内通过手机APP对万年历实现远程监控。由主控模块、传感器模块、按键模块、时钟模块、显示模块、报警模块、通信模块组成。

经过不断调试与验证，该系统成功通过测试，实现了一种实用可靠、操作方便的智能时钟，市场潜力无限，前景广阔无比。但是，由于时间与个人能力的限制，本系统并没有很完美，本系统传感器节点的设计还不够完善，本文的研究重点在如何实现温湿度、光照的检测，以及对采集到的信号完成数模转换这几个部分，对于各传感器节点的布置和电路设计还存在一定的缺陷，所以在后期还需要在这方面下功夫，来提高程序的性能。

综上，即使本万年历系统得到了成功设计与实现，对本人的程序开发能力起到了帮助和提高作用，也在时间管理领域中起到了促进作用，但是，在以后的时间里，本人还可以继续针对以上提出的问题进行研究，争取让本系统的功能更加完善，为我国现代化发展尽一份绵薄之力。

# 参考文献

1. 范书瑞,李琦,赵燕飞.Cortex-M3嵌入式处理器原理与应用[M].北京:电子工业出版社,2011.
2. Gajski, D. D., & Vahid, F. Embedded System Design: Modeling, Synthesis, and Verification [M]. Norwell, MA: Kluwer Academic Publishers, 2001.
3. Ganssle,Jack The Art of Designing Embedded Systems[M].the Netherlands: Newnes, 2008.
4. 刘婷.传感器设计中应用单片机技术的分析[J].数码设计,2022,6(09):85-85.
5. 李宏,张家田.液晶显示器件应用技术[M].北京:机械工业出版社,2004.
6. 洗伟铨,钟玉琢.蓝牙(Bluetooth)在数字化信息家电中的应用[C]//第九届全国多媒体技术学术会议论文集.北京:中国计算机学会, 2001:432-435.
7. 刘磊.基于STM32的电子万年历系统设计[J].科技资讯,2015,13(1):20-20.
8. 王锐.基于单片机的万年历设计[J].信息技术,2014(9):213-216.
9. 张攀峰.基于LPC922单片机和I2C总线的低功耗万年历的设计[J].陕西科技大学学报（自然科学版）,2010,28(2):118-121.
10. 安会,蒲禹辰,李纪榕,马红静,李莉,张静.基于单片机的电子时钟设计与制作[J].电子制作,2022,30(1):71-74100
11. Zhao, Y. Bluetooth Low Energy: The Developer's Handbook[M]. Hoboken, NJ: Wiley, 2017.
12. 谭浩强.C程序设计[M].北京:清华大学出版社,2005:18-62.
13. Valvano, J. W. ARM Cortex-M4 Cookbook [M]. San Mateo, CA: Morgan Kaufmann, 2014.
14. O'Reilly, Tim, and Greg Danyluk. Visual Studio Code: The Definitive Guide[M]. Sebastopol: O'Reilly Media, 2020.
15. Sauter, M. Bluetooth Essentials for Programmers [M]. Sebastopol, CA: O'Reilly Media, 2017.

# 附 录

附录A系统原理图

图示, 示意图

AI 生成的内容可能不正确。

附录B PCB版图（正反面）

电子设备的屏幕

AI 生成的内容可能不正确。

**图B1（a） PCB版图正面**

电子设备的屏幕

AI 生成的内容可能不正确。

**图B1（b）PCB版图反面**

附录C C语言程序代码

#include "sys.h"

#include "usart3.h"

SENSOR SensorData;//传感器结构体定义

SYSTEM System;//系统标志位结构体定义

void mqttPublic(void);

void Threshold\_Init(void);

int main(void)

{

delay\_init();//延时函数初始化

NVIC\_Config(); //中断优先级配置

NVIC\_PriorityGroupConfig(NVIC\_PriorityGroup\_2);

My\_USART1();//串口1初始化 （调试打印用） 波特率：9600

Usart3\_Init(9600);//串口3初始化 （与HC-05通信用）波特率：115200

LED\_GPIO\_Config();//LED灯初始化

Beep\_Init();

oled\_Init();//oled初始化

BH1750\_Init();//BH1750初始化

DHT11\_Init();//温湿度传感器初始化

KEY\_Init();//按键初始化

Threshold\_Init();//传感器阈值及执行器件开关变量初始化

TIM1\_Int\_Init(99,3599);//定时器1初始化 定时扫描按键

MyRTC\_Init();//RTC时钟初始化

while(1)

{

SensorData.LightVal = Light\_Intensity();

DHT11\_Read\_TempAndHumidity(&dht11Data);//获取温湿度;

Mode\_Decide();//模式判定 （按键1选择模式）

if(System.BT\_Pub)

{

u3\_printf("Sensor1:%d#,Sensor2:%d#,Sensor3:%.2f#",dht11Data.temp\_int,dht11Data.humi\_int,SensorData.LightVal);

System.BT\_Pub=0;

}

}

}

//传感器阈值及执行器件开关变量初始化

void Threshold\_Init(void)

{

System.Switch1=System.Switch2=System.Switch3=System.Switch4=0;

System.BT\_Pub=0;

System.ClearFlag=0;

Alarm1.Hour=10;

Alarm1.Minute=20;

Alarm1.Second=0;

Alarm2.Hour=11;

Alarm2.Minute=20;

Alarm2.Second=0;

Alarm3.Hour=12;

Alarm3.Minute=20;

Alarm3.Second=0;

}