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

时间一闪而逝，回顾往日，踏入校园的那一刻仿佛仍停留在昨日。此时此刻，我也即将为大学四年的生活画上满意的句号。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，我十分感激本次毕业设计的指导老师，正是有您细致的指导和严格的要求，我才能顺利的完成本次的设计，在这段时间，我有颇多的问题，您一直都十分有耐心的指导我，对于一些简单的问题，您也会相当细致的给我讲解，并告诉我很多的解决方案，同时还给我提供了参考资料的搜集途径，让我能快速查阅和学习，在您身上，我不仅仅是学习到了专业知识的运用，还看到了为人师表、终身学习、与人为善的生活态度，我在心里由衷地敬佩您，在未来的生活学习中也不会忘了您的教诲，以您为榜样砥砺前行。

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，是您们给了我一切，更是我坚实的后盾，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）技术的快速崛起，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。根据本系统的功能实现需求，将本系统分为硬件控制模块和远程控制模块进行设计与实现。硬件以STM32开发板为硬件控制主模块，使用C语言进行编程实现。通过DHT11和BH1750传感器模块采集处理得到当下周边环境中的温湿度等指标，通过集成的RTC实时时钟模块达到更为精确的当前时间校准，实现年、月、日、时、分、秒、星期时间的具体显示并能实现节气的显示功能，当达到设定的时间，可以通过声光报警提示方式提醒用户。以蓝牙通信为基础，在远程控制模块中与手机应用程序进行连接，实现远程控制本系统手机应用程序的功能，通过此模块可以完成对本系统万年历模块功能的调整，同时也能够与其他的智能设备进行相互的连接与控制，实现生活中的更便捷性、人性化。

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, electronic perpetual calendars have rapidly emerged as home terminal devices due to 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. According to the functional implementation requirements of this system, it is divided into hardware control module and remote control module for design and implementation. The hardware is controlled by the STM32 development board as the main module and programmed using C language. The temperature and humidity indicators in the surrounding environment are collected and processed through DHT11 and BH1750 sensor modules. The integrated RTC real-time clock module achieves more accurate current time calibration, realizing the specific display of year, month, day, hour, minute, second, and week time, as well as the display function of solar terms. When the set time is reached, the user can be reminded through sound and light alarm prompts. Based on Bluetooth communication, the remote control module is connected to the mobile application to achieve remote control of the system's mobile application functions. Through this module, adjustments can be made to the perpetual calendar module of the system, and it can also be connected and controlled with other smart devices, achieving greater convenience and humanization 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。该阶段的实时时钟(Real-Time Clock，RTC)设计采用并行传输数据方法，功耗较高，制造工艺仍然为普通CMOS，封装方法选取双列直插形式。而且该款产品的功能没有做到24H、12H时制之间的自动转换，也没有对闰年进行处理，不能很好地解决“千年虫”问题，最终被市场所淘汰。大约在1995年，RTC芯片逐步采用先进的CMOS技术，功耗电流典型值降至0.5HA以下，电源电压达到了1.4V的要求。采用串行通信接口与处理器连接，包含SPI总线和I2C总线等，且采用更加紧凑的。随着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，能够实现毫秒级的计时功能，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，Gengqi L和Jinjie B在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]。提及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。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。

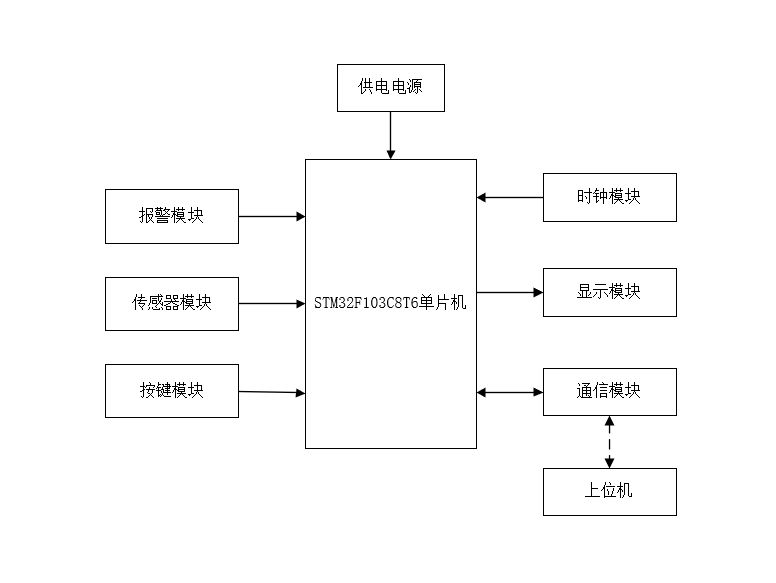
# 2 系统整体设计

## 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。本系统是不依靠人工参与的全自动系统，可以实现对日常生活中人们基本时间管控的合理要求，提供准确与完整的时间信息，从而提高人们的日常效率和工作生活。系统的组成主要由主模块控制器、感知系统、键盘以及OLED界面、时钟模块、显示模块、串口模块和报警模块组成。

在此次设计中，作为主体功能的控制器模块能发送与接收命令以及对获取的数据进行处理和计算；温湿度、光照量检测模块用传感器来监测所处环境，并在接收数据后传输给控制器模块显示；采用按键或移动终端对当前的时间进行人工更改和闹钟的设定，显示于显示模块上，以便人员查看，增加了程序的交互性

因此，日历系统每年的整体框图像显示在图2.1中。

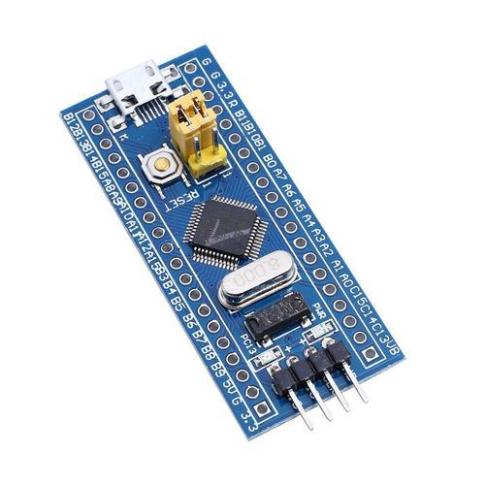


**图2.1 系统总体框图**

## 2.2 模块选型

### 2.2.1 电子万年历MCU

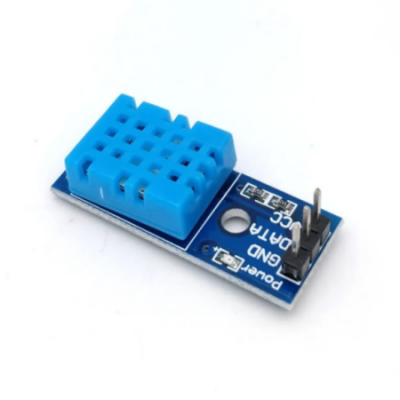
主控制器模块是整个系统的核心模块，通过中央处理器芯片执行各类程序命令，所有发送和接受的命令、运算的数据均由该模块控制，故选择性能先进的中央处理芯片在程序编写上可以起到事半功倍的作用。STM32F103C8T6微控制器是一款以ARMCortex-M3为架构的芯片，拥有超强的运算能力，能够有效处理各类中断指令，且拥有低功耗的优势。芯片运行频率为72MHz，可支持多种外设，例如ADC、DAC、I2C、SPI、USA、定时器等满足多项应用需求。除了具备各种各样的外设外，STM32F103C8T6芯片还可以多接口支持外部器件以及存储区的加入；同时，STM32F103C8T6芯片还具备低功耗的优势，可支持待命模式、休眠模式、停机模式等[3]。MCU硬件实现可见图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，其硬件实现如图2.3所示，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传感器中存在光敏二极管（PD）、运算放大器、模数转换器（ADC）以及内部晶振等器件。图2.4展示了其外观形态。其中的PD光敏二极管通过光电效应，可以把光信号调制为电信号。电信号在运算放大器电路进行增益放大处理后，电压被放大后会经过模数转换器转换。随后，转换后的电压信号在逻辑电路中被转换成16位二进制数，并保存在寄存器中（注：感光窗口获取的光线强度越强，得到的光电流和电压值就越大。因此，可通过电压的改变来判断光强度）。BH1750集成时钟线和数据线，微控制器可使用I2C协议控制BH1750模块，从而设置该模块的工作模式和读取光强度数据。



**图2.4 BH1750传感器图**

### 2.2.3 显示屏选择

万年历系统中，显示组件的作用尤为重要，作为与用户直接沟通的桥梁，它需要有高清晰度、全面且富有动态感的效果。在显示器件中，显示器件类型相当丰富，例如发光二极管（LED）、LCD1602液晶屏幕和OLED显示屏等。

（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液晶控制芯片有172820(240\*320\*18/8)26万色的存储能力。模块的每一个像素点都是以红绿蓝三种颜色混合而成，16位数据线与显存的对应关系高达565种[5]。这种显示器采用平面结构，是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力的作用下发生改变，改变反射的光线角度，从而实现信息的显示。其使用的技术相对较落后，厚度较大，显示分辨率偏低，同时也存在延迟，界面反应速度无法满足快速切换的要求，因此已经在市场上逐渐被淘汰。

（2）OLED显示屏。这种设备在近年在硬件应用中被广泛使用，它是以有机材料组成，能够提供生动清晰的显示效果，界面精致。此外，在操作过程中的响应速度以及界面切换方面十分稳定，展示了其高效性和长期使用的价值。

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。OLED实物如图2.5所示。



**图2.5 OLED显示屏图**

### 2.2.4 通信方式选择

本此项目利用蓝牙通信方式实现与移动终端的通信，可以较好地满足程序远程监控需求，确保万年历能与其他的装置连接，选择的通信方式有ZigBee、WIFI和蓝牙，下面我们分析其特点。

从数据传输速率上来说，ZigBee速率一般为10~250kbps，普通蓝牙传输速率一般为1Mbps~2Mbps，而WiFi速率相对于ZigBee和蓝牙传输速率远远领先，以802.11ax为标准，最高可以达到10Gbps，规模迥异。仅有传输速率高的才能支持远程监控实现实时传输，并且满足远程的大数据、音视频在线处理及下载的需求。

通讯距离方面，一般来说，ZigBee跟蓝牙能实现10至100米之间的数据传输，但是WIFI的覆盖范围可以达到300米，很适合远程监控的无线传输方式。

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。HC-05模块实物如图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。此项目所用的声光报警器材常采用蜂鸣器与LED灯两种形式完成警报。蜂鸣器种类包括无源蜂鸣器与有源蜂鸣器两种，可根据产品需要选择。

（1）无源蜂鸣器：由名称可以看出，该蜂鸣器没有内置的晶振。无源蜂鸣器在通电的情况下无法自行发声，需要对蜂鸣器加以方波信号，并对其进行频率改变才能使得无源蜂鸣器发出警报声。因其内置没有振荡源，成本上低于有源蜂鸣器。除此之外，无源蜂鸣器直接反应能力较弱，因此其反应速度比有源蜂鸣器较慢。

（2）有源蜂鸣器：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，系统选择如图2.7的有源蜂鸣器报警。



**图2.7 有源蜂鸣器图**

# 3 硬件电路设计

## 3.1 MCU电路设计

本次万年历的系统核心使用STM32F103C8T6微控制器。如图3.1所示，它拥有相当丰富的内部集成电路，例如电源模块、复位模块、模数转换（ADC）模块以及时钟模块等。电源电路提供了整个系统运转所需的能量。对于STM32F103C8T6芯片而言，其规定的供电电压为3.3V，连接到第38引脚。但是一些外部模块器件需要5V电源才能工作，在将5V电源连接到第18引脚后，稳压器件会将其降到3.3V，来达到STM32F103C8T6芯片的供电需求。

STM32最小系统所需的两个晶振模块：一个是32768Hz的低速晶振和一个8MHz的高速晶振[7]。来满足对不同频率的需求，系统复位外接一个按键连接复位电路按键按下实现系统复位功能。而程序下载和硬件仿真则是以JTAG电路以实现。如果需要重启系统，可以给系统断电在上电或者是使用复位电路上的按钮，从而初始化整个程序。ADC转换电路用于将传感器读取到的模拟电压数据处理为STM32F103C8T6芯片能够识别处理的数字信号。时钟系统包含一个振荡器芯片，用于在程序执行期间管理时间，确保程序按照预定的时间顺序执行。主控模块的集成展现出一体化、简化和灵活的特性，电路中的其他引脚和接口负责连接到显示单元、传感器等，以达到控制信号的传输。



**图3.1 主控模块电路设计**

## 3.2 电源电路设计

一方面，各类传感器模块和MCU都需要稳定的电源，如3.3V或5V，保证内部电路正常工作；另一方面，由于常见的电源输入类型常为计算机USB接口供电或充电器头（电压5V左右），因此电源电路通常设计一个降压变换电路，将5V转换为3.3V输出。比如，低压差稳压芯片，如ME6211C33、AMS1117-3.3等，可以完成电压的转化。AMS1117-3.3提供的电压输出为1.2V、1.5V、1.8V、2.5V、3.3V和5.0V。IC的热关断功能可以阻止由于输入过载或在高温条件下结温过高的情况。

再电路中加入22uF的旁路电容降低噪声，通常线性稳压器的稳定性会随输出电流增大而减小。LDO具有低压降，且输出精度高，在输入与输出的压差较小的情况下也能实现输出3.3V稳定电压的目的。

电源电路为方便直观观察工作状态，一般会在电源电路加入指示灯，指示灯一般使用LED二极管，且指示灯与AMS1117输出的3.3V通过限流电阻进行串联接线。

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传感器用于采集温度和湿度，在传感器中由电容式感应元件和数字处理器构成。在传感器中，电容式感应元件包括温度和湿度传感器，温度传感器为热敏电阻，其电阻值与周围环境的温度呈反比，环境温度的增加，电阻变小，测得电阻的变化量即可知道环境的温度；湿度传感器具有一定吸湿能力，会根据空气中含水量的多少而伸缩，根据其伸缩的多少而测得电阻，即可知道湿度的大小。DHT11传感器本身可将电转换为数字信号，因为传感器自身设置有一个16位的AD转换器，可以将温度和湿度的变化数据量灵敏地和直接进行监测。在所设计的电路中，DHT11传感器中有电源脚和数据脚，第一个脚用于连接3.3V的电位，第三个脚接地，两者形成闭合电路，第二个脚用于提供数据，单总线将采集的温湿度数字信号传递给单片机的PA4接脚，同时，单片机也可以通过该接脚向传感器传递控制信号。另外为了保证传感器未使用时接脚2是高电位，则设置了上拉电阻。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传感器感知光照强弱变化。它内部电子会在光照下处与激发状态，在电子跃迁时会产生微弱的电流变化，而这个电流对应有个特定的电阻值。在没有光照时，传感器内部的光敏器件的电阻值较高；而当检测到光线时，电阻会随之降低。因此，光强与电阻值之间呈反比。通常BH1750传感器的工作电路中，设计了一个偏置电路来进行分压处理，从而提高传感器的响应速度。该电路的设计如图3.4所示，正极引脚（引脚1）连接到5V电源，而引脚4（负极引脚）接地。同时，引脚2和引脚3分别作为串行时钟线引脚（SCL）和串行数据线引脚（SDA）。SDA引脚负责将光强数据传输到微控制器的PA10端口，而SCL引脚则与微控制器进行时钟信号同步。这两个引脚默认处于高电平状态。



**图3.4 BH1750传感器电路设计**

### 3.3.3 按键模块电路设计

用户操作上，我使用比较便宜的按键控制，每个按键单独用一根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，每个按键具有不同的功能，MCU将判断按键的状态变化来控制相应的程序电路执行。然而，按键未按下时，按键内部的接触点并没有跟相关电路建立连接，处于开路状态。因此，当前的按键电路处于关闭状态；相反按键被激活，其内部触点将和电路发生连接，MCU的I/O口电压会随之被拉低，而我们的微控制器通过判断I/O口的电压状态来判断按键是否被按下。从而响应该信号并执行相应的处理程序。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，实现信号的传递。



**图3.5 按键模块电路设计**

### 3.3.4 显示模块电路设计

0.96英寸的OLED显示器可以满足当前系统程序输出的需求，其电路布局见图3.6。显示屏的供电使用外部5V电路连接至VCC，OLED显示屏通过驱动电路，使得屏幕内的集成的有机材料图层能够发光，并且它不需要背光片，驱动采用的是常见的led驱动芯片SSD1306，并且通过I2C与主控模块的PA11、PA12接口连接；将需要显示的内容传输显示在液晶屏上。

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

实时时钟模块可以在设备失去电力供应或重启后持续提供日期、小时、分和秒的准确时间信息，许多RTC还具有一种称为报警(Alarm)的功能，当达到预置时间时触发中断以完成定时唤醒或其它操作[10]。同时，RTC还可用于安排系统的定期功能，例如在每天的日志备份或设备注定每天自动开启、关机等。RTC能在设备进入睡眠模式或失去电力供应后，根据预置时间自动唤醒系统，可以有效降低耗电量，RTC亦可根据设备记录日志或接收传感信息的时间设置，为数据的管理和分析提供高精度的时间戳，提高数据管理分析性能。RTC模块由下面几个部分组成。

（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。

本次采用集成在STM32内部的实时时钟模块，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，VBAT引脚连接到备用电池，确保在系统主电源关闭时，能够继续维持当前时间和日期的正常运转。



**图3.7 时钟模块电路设计**

### 3.3.6 声光提醒电路设计

本次系统的声光模块使用了有源蜂鸣器和LED灯，有源蜂鸣器包含有震荡电路跟放大器电路。在此架构中，震荡电路的功能是生成震荡电流。当电流通过蜂鸣器时，蜂鸣器的振荡器会产生一个恒定的震荡频率，这样就能让蜂鸣器产生声音。当发声装置输出音频信号后，此信号会经过放大器电路进行增强，从而生成警示音效。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，防止进入基极的电流过大损坏三极管。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**

当蜂鸣器发出声音的同时，LED灯也会闪烁，其驱动电路如图3.9所示。为了防止输出到LED灯的电流过大使用了一个1K的限流电阻以确保LED灯的工作环境良好。主控模块通过改变PC13端口的输出电压来控制LED的工作状态。在电信号输出完成后，会启动一个延时机制，方便用户观察到LED灯的点亮或熄灭。



**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蓝牙模块是一种蓝牙协议的简单无线通信模块。该蓝牙模块采用了单芯片蓝牙IC芯片BC417，遵循蓝牙V2.0协议，且支持增强数据率(EDR)，其调制速率为2Mbps~3Mbps。模块内部具有2.4GHz的天线，无须用户调整，模块还具有自适应的跳频功能，使其避免相互干扰，保证了通信的稳定性[11]。HC-05模块分为命令回应工作模式与自动连接工作模式。在命令回应的工作模式下，模块可以运行AT指令，允许用户设置控制参数与下达控制指令；在自动连接的工作模式下，模块可以处于主机(Master)、从机(Slave)或者自环(Loppback)模式进行传输信息。通信模块HC-05的电路如图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C语言编程主要在于C语言的高效性和简洁性，能直接与硬件进行交互；具有结构化编程的优势；有多种运算符、多种数据结构；能用指针对内存进行灵活精确控制；C[12]语言具有很强的跨平台能力，可以在不同类型的计算机系统中编译；能进行反汇编；与汇编语言编写的程序运行效率近乎一致。

Keil5是Keil公司特为嵌入式系统的开发者打造的一体式开发软件(IDE)。嵌入式系统开发利用该开发软件可以与多种微控制器开发，特别是对基于ARM体系架构下的芯片，极大的提升了硬件的开发效率。MDK-ARM即包含工业标准C语言编译器和调试器、宏汇编语言编译器及实时操作系统，给工程师提供了一个强大且稳定的工具链。Keil[13]能实现强大的代码编辑、编译功能，其中它还拥有强大的代码编辑器，实现在线编程、编译、模拟和调试，能及时找到代码中的漏洞并对其予以修复。在这个开发环境中，它能由使用者直接输入C语言编码，并通过自带的编译器将其转化为能够在目标硬件平台上运行的程序。Keil还拥有众多的调试工具，它还配备有源代码的调试工具，它能够模拟多种MCU硬件系统，使开发者编写代码时就能开展实时的调试工作。通过调试点设置，能够让程序员准确找到代码中存在的漏洞，有效提高了程序员的工作效率。

VSCode[14]是免费开源跨平台、轻量的开发IDE，支持Windows、macOS、Linux，支持JS、Python、C++等语言的多种插件，支持智能代码补全、代码高亮、代码调试器，支持版本控制系统等。可以灵活地调整界面和功能，为个人的工作环境进行优化。另外，支持VSCode集成Git、终端，启动快速，运行高效。

4.2 系统程序设计

### 4.2.1 主程序设计

此次设计的main.c主要是对各个功能模块进行初始化设置，并且设置每一种设备的I/O端口，并且把其中所有功能代码进行封装以便运行。在程序运行开始，用户通过手机打开APP搜索蓝牙HC-05并输入正确的密码与之建立手机APP连接并开始各个功能模块。其中传感器负责对环境的温度、湿度、光照强度进行采集，然后在环境采集的过程中，可以使用按键方式或者APP手动设置当前时间并进行设定闹钟。数据处理结果利用OLED显示。设定的闹钟时间到了时启动报警模块以提示用户。单片机通过单总线的方式将其传输至单片机，接收后整理分析数据来实现声光报警。设计的主要程序过程如图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

传感模块用来采集温度湿度、光照信息等，经一系列放大、滤波、数模转换等处理成数字信号。MCU控制DHT11传感器以及BH1750传感器使之运作，之后传感器内的应变电阻采集相应的电压值，通过内部处理将电压值转换成电信号，最后通过引脚转换成电信号传送给MCU进行处理和解析。

AD模块主要负责将传感器输出的模拟信号转变为微控制器能够识别的数字信号。STM32微控制器内的AD转换模块通常内置在芯片内部，无须通过外部引脚进行扩展接线。故可以直接在寄存器内进行模拟信号的读写操作，并完成转化过程，此外STM32的IO端口还带有复用功能的多路选择开关，在这不仅使单片机的运行效率得以保障，同时还实现了一片单片机多个功能的整合。

模拟信号向数字信号进行转换的过程，是先采样、再量化、最后编码的三部分组成。首先是提取一定时间范围内的连续信号，进而产生了一组离散值；接着把信号在实际环境中出现的所有模拟信号的幅值连续分立为数字值；再将所得到的离散样本值用单片机能够认识的形式进行转换，也就是编码转换，于是完成了从模拟信号向数字信号的转换。

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

按键作为用户和设备之间的交互枢纽，能够将用户的想法传递给硬件设备进行相应功能的执行。当用户按下按键时，内部触点与电路接触，形成闭合电路，并向微控制器发送切换显示界面或者是调整当前时间跟日期的指令，微控制器随后根据接收到数据响应。按键的功能有时间校准、设置闹钟以及警报通知等。在软件实现方面，通过扫描按键端口的状态来判断按键是否被按下。利用不同的按键状态来实现相应的模式切换和时间设置。首先，需要读取按键的当前状态，以确定按键是被按下、释放还是保持不变。设计了一个状态机来管理按键的状态转换，例如从按下状态变为释放状态的变化。当按键状态发生变化时，会进行消抖处理，以防止因按键的机械弹跳而导致的错误多次触发。最后，根据不同的按键状态，执行相应的操作，例如响应按键按下和释放事件，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与万年历通讯的途径有五步和五种方法，首先使用UART通讯协议对MCU主控单元与HC-05建立连接；当手机APP和MCU使用PIN（个人识别码）建立连接后，手机APP通过SDP（服务发现协议）协议选择用那种服务将MCU上采集的数据传送，这些数据包含了传感器采集的数值、设备运行信息[15]。APP在此设计中可以读取到温度、湿度、光照等传感器信息；数据采集过程中须按照APP的触发条件及时间间隔，保证数据的准确性。

APP获取的由MCU监控的数据获取后还无法直接利用，须经过解析、转换、滤波等一系列数据处理程序后，才能进行远程控制或预警监控等工作。

最后，APP还可实现对MCU的控制。该阶段需要使用工业控制算法并将控制程序下载至下位机中存储器，实现远程控制，如时间同步、闹钟设置等。

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，根据代码进行硬件设备的控制，确保可以对环境温湿度、光照的监测，跟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

本次万年历系统软件设计分为MCU、蓝牙模块、显示模块、温湿度传感器模块、蜂鸣器模块和按键模块。MCU能对各模块采用不同的通讯方式制定通讯协议及进行各外设指令的发送、接收及数据逻辑运算与处理；通过主控制模块的接口引脚实现数据的传输与控制，可很好地获取及转换环境温湿度及光照强度。

按键模块由5个按键组成，按哪个键实现什么功能，闭合按下的开关按钮，流过电流，完成了用户与设备的沟通，通过按键设置时间和闹钟功能；设备通过OLED显示当前时间、闹钟以及监控数值等信息。

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**

|  |  |
| --- | --- |
| 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程序的形式呈现。系统提供数据查询和警报设置功能，使用户能够进行24小时远程监控。首先，用户在移动应用程序与设备之间建立蓝牙连接，以实时同步硬件收集的数据。在界面顶部，用户可以查看温度、湿度、光照强度等指标的当前状态，以及实时时间信息。在查看当前时间的同时，用户还可以进行一键校准，并为硬件设置三个不同的警报。软件功能的实现如表5.4所示。

**表5.4 系统软件功能实现**

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

# 6 总结与展望

本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，本系统传感器节点和APP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. 张云,曹剑侠,李景景.基于单片机的多功能万年历设计[J].信息与电脑(理论版),2022,34(12):137-139.
2. Gengqi L,Jinjie B,Yi Y.Design of Perpetual Calendar Based on STC12C5A60S2 Microcontroller[J].Academic Journal of Engineering and Technology Science,2023,6(6):13-15.
3. Li J,Zhu Q,Gao Y,et al.Design of Energy-Saving Control System for Intelligent Street Lamps Based on STM32[J].Journal of Engineering System,2025,3(1):16-18.
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. 邸茵,刘莹,吴作用等.液晶光重构显示技术的发展与应用[J].液晶与显示,2024,39(11):1454-1462.
6. 龙子鸣,徐建,肖霖怡等.基于蓝牙可穿戴式心电监测系统的设计[J].物联网技术,2025,15(03):27-30.
7. 袁颖,熊洁.基于STM32单片机的智能手环设计[J].科技与创新,2025,(07):75-77.
8. 张梦君,任莉.基于STM32的宠物远程监控装置设计[J].科技与创新,2025,(05):63-65.
9. 刘秀娟,李朋龙,邵娟等.多功能电子万年历设计[J].计算机时代,2022,(05):73-75.
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):71-74.
11. Koulouras G,Katsoulis S,Zantalis F.Evolution of Bluetooth Technology:BLE in the IoT Ecosystem[J].Sensors,2025,25(4):996-996.
12. 谭浩强.C语言实战[M],电子工业出版社,2021.
13. Yiu,J.Embedded systems design with the ARM Cortex-M3[M].The Netherlands:Elsevier,2020.
14. O'Reilly,Tim,and Greg Danyluk.Visual Studio Code:The Definitive Guide[M].Sebastopol:O'Reilly Me ia,2020.
15. Afaneh,M.Intro to Bluetooth Low Energy (2nd ed.)[M].Joseph-Beth Booksellers,2023.

# 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;

}