为

**毕业论文**

**题 目： 室内空气质量检测系统的设计**

**姓 名： 蒋纯中**

**学 号： 162051510305**

**学 院： 电子与信息工程学院**

**专 业： 电子信息工程**

**指导教师： 徐荃**

**完成时间： 2020年5月30日**

**摘 要**

随着生活水平的日益提高，人们对绿色健康生活的意识在逐年提升，与此同时人们对室内空气质量的需求也越来越大。在诸多因素中，尤为受到人们关注的是室内空气温湿度的高低及有害气体的污染。研究表明，室内空气污染及温湿度过高或过低会引发各种疾病，严重影响人们的工作和生活。因此，设计一款能够实时检测室内空气质量的室内空气质量检测系统具有重大意义和实用价值。

基于此，本文以STM32F103系列单片机为主控，结合WIFI模块、DHT11温湿度传感器、MQ135有害气体传感器、显示模块以及电源模块等，开发了一种室内空气质量实时检测系统。该系统能够实现有害气体浓度和温湿度的实时采集，并通过液晶屏实时显示，同时将WIFI模块连接公网，以无线设备作为服务器端，在无线设备上接收环境数据。此外，对有害气体超标设置阈值，通过声光报警器进行报警，以提醒人们知晓室内空气质量情况。

**关键词：**环境检测；STM32F103；传感器；WIFI；无线设备；

**ABSTRACT**

With the improvement of living standards, people's awareness of green and healthy life is increasing year by year, at the same time, people's demand for indoor air quality is also increasing. Among the many factors, people pay more attention to the temperature and humidity of indoor air and the pollution of harmful gases. Studies have shown that indoor air pollution and too high or too low temperature and humidity will cause a variety of diseases, seriously affecting people's work and life. Therefore, the design of an indoor air quality detection system which can detect indoor air quality in real time is of great significance and practical value.

Based on this, this paper develops a real-time detection system of indoor air quality with STM32F103 series single-chip microcomputer as the main control, combined with WIFI module, DHT11 temperature and humidity sensor, MQ135 harmful gas sensor, display module and power supply module. The system can realize real-time acquisition of air index and temperature and humidity, and real-time display through LCD screen. At the same time, the WIFI module is connected to the public network, and the wireless device is used as the server to receive environmental data on the wireless device. In addition, a threshold is set for harmful gases exceeding the standard, and an alarm is made through an acousto-optic alarm to remind people of indoor air quality.

**Keywords:** environmental detection; STM32F103; sensor; WIFI; wireless device;

**目 录**

第一章 绪论 1

1.1 课题背景 1

1.2 本课题的研究现状及发展趋势 1

1.3 本课题的研究目的及意义 2

1.4 本课题的主要研究工作 3

第二章 方案设计 4

2.1 系统总体设计与分析 4

2.1.1 总体设计的方案 4

2.1.2 总体框图 4

2.2 方案选择 5

2.2.1 主控芯片选型 5

2.2.2 显示屏选型 6

第三章 硬件电路设计 7

3.1 硬件电路总原理图 7

3.2 STM32单片机主控模块 7

3.2.1 STM32单片机简介 7

3.2.2 STM32最小系统原理图 8

3.3 DHT11温湿度传感器 8

3.3.1 DHT11简介 8

3.3.2 DHT11电路原理图 9

3.4 MQ135空气质量传感器 9

3.4.1 MQ135简介 9

3.4.2 MQ135电路原理图 10

3.5声光报警模块 10

3.5.1 蜂鸣器简介 10

3.5.2 声光报警电路原理图 10

3.6 WIFI模块 11

3.6.1 ESP8266简介 11

3.6.2 ESP8266电路原理图 12

3.7 液晶模块 12

3.7.1 OLED显示屏简介 12

3.7.2 OLED显示屏电路原理图 12

第四章 软件程序设计 14

4.1 开发软件简介 14

4.2 系统的主程序设计 14

4.3 DHT11温湿度传感器模块设计 15

4.4 MQ135空气质量传感器模块设计 17

4.5 WIFI模块设计 19

4.6 OLED显示屏模块设计 21

4.7报警器模块设计 23

第五章 系统制作与调试 25

5.1 硬件制作 25

5.1.1 AD简介 25

5.1.2 系统PCB板的设计 25

5.2 系统硬件调试 26

5.2 软件及联机调试 27

5.2.1 主控程序调试 27

5.2.2 短信息发送调试 28

第六章 总结与展望 30

致谢 31

参考文献 32

附录 系统源程序 33

第一章 绪论

**1.1 课题背景**

近年来，我国工业化水平不断发展，人民生活水平随之提高。与此同时，环境污染也成为一个不容推延的问题。而在这些众多的环境污染中，空气污染对人们的影响最为严重，因此了解空气质量就显得尤为重要。有关专家发现，空气中含有大量的有机化合物，其中包括一些致癌物和病毒。越来越多的实际例子证明，空气污染已经成为人类健康的隐形杀手，空气检测也是目前我国必须面对的课题。据悉，华北地区一半以上的人口处于空气污染中，这也造成了大量呼吸道疾病感染者，对当地生产发展造成了不良影响。当前空气污染中最受关注的有害气体主要包括：二氧化碳，甲醛，二氧化氮，PM2.5。这些气体主要来源于家居装修、汽车尾气排放以及各种工厂带来的污染。

鉴于问题的严重性，许多方面已经开始采取不同的措施来挽救空气质量。例如，零污染装修的倡导者现在可以从源头上切断污染源，环境保护者提倡科学治理环境。此外，空气质量标准化检测也成为有力的防护武器。总之，人民迫切需要关注空气质量，掌握自我检测才能更好的保护自己和他人。

**1.2 [本课题的研究现状](#目录)****及发展趋势**

早些年，BBN公司联合麻省理工大学开发的无线传感器网络系统用于测量室内CO2浓度、室内温湿度、有害气体、烟尘、PM2.5等环境数据。随后，美国各大研究院相继开发了软硬件平台，利用CPR通信网络应用系统检测学生宿舍周围的气温、空气湿度、室外有害气体、风速等。葡萄牙科学家在文章《利用漂浮右植物组合检测水体框架中的湖泊生态状况说明：MAP组合表》中提出使用浮游植物监测房屋周围环境。日本东京科学家在《基于云框架的环境资源分析》一文中提出了一种基于云计数的管理系统。根据全球室内环境数据XBRL报告，通过对室内环境数据进行比对，起到预警报警的作用。

由于我国工业化进程发展较晚，与国外发达国家相比，我国积累的相关技术有限。目前有一种基于MSP430开发平台的环境检测系统，它是利用ds18B20温度传感器和HS1101传感器和BH1750光强传感器开发的。这个产品可靠耐用，功耗低。近年来，由于ARM产品在国内的风靡，市场上出现一款基于ARM11平台的环境检测设计。加上DHT90温湿度传感器模块和BH1750光照传感器，以及烟雾传感器，这款设计工作稳定、功耗低、功能性较强。看起来很完美，非常耐用。但众所周知，嵌入式作品价格较高。这些系统大多以单片机为主控制器，然后通过通信手段将微处理器与各个传感器连接起来，实现多个模块的通信和传输，具备联网功能的系统发展尚不成熟。由此可见，我国的室内环境检测还处于发展阶段，还在探索的研究道路上不断完善。

物联网技术的出现打破了这一窘境，在新老产品交替之间，物联网技术、电子信息技术和电子测量技术相结合的趋势逐渐显现。目前，很多企业研究院已经开始研究如何将物联网应用到我们的生活中。物联网的优势在于让产品变得更加“聪明”。就目前市场上类似的环境检测产品来看，功能单一、稳定性差限制了应用范围，操作复杂、价格昂贵令消费者望而却步。而随着物联网的良好发展趋势，只能实现上述单一功能的系统或稳定性差的产品将逐步淘汰，具备综合功能的系统且使用方便在未来必将占据市场。

虽然物联网具备以往产品不可比拟的优势，但作为一个新兴行业依然存在一一些需要解决的问题。在人工智能时代的不断牵引下，物联网定会朝着越来越好的方向发展。主要有以下几个方面：

1.稳定化。该产品面向的是广大市民，稳定易用是客户使用产品的直观感受；

2.标准化。物联网行业目前没有统一的行业规范，其产品开发难度及兼容性受到很大限制。建立健全统一标准，有利于产品的进一步升级；

3.成本低。物联网技术发展迅速，产品普及率较低，最大的原因就是成本太高。然而，随着元器件的不断更新换代和流水线生产技术的不断成熟，未来的低成本产品是必然结果。

**1.3本课题的研究目的及意义**

随着生活水平的不断提高，广大人民群众的绿色健康意识也逐步提高，党和国家大力倡导“绿水青山就是金山银山”的理念，室内空气检测与治理也逐渐成为一个新的国民经济产业。近年来，家居装修发展迅速，不规范的家装带来的负面影响十分严重，而目前建材市场的混乱，造成非环保、低劣质的家装材料进入家居，有害气体以甲醛、甲醚的污染为首。目前室内环境污染的四大毒气—甲醛、苯系物、氨、氡的检测是室内空气质量检测的主要项目，室内空气污染物也以这四大毒气为主。研究表明，室内空气污染可引发多种疾病，包括呼吸系统疾病、慢性支气管炎症、慢性肺部疾病等，严重影响人们的工作和生活。

因此，设计一套能够实时检测室内空气质量的室内空气质量检测系统具有重要的意义和实用价值。

**1.4本课题的主要研究工作**

本文主要研究基于STM32单片机，结合物联网技术的环境检测系统，主要实现实时显示和远程监控功能。本设计的具体路线分为：STM32平台的选择和设计方案、学习STM32及其相关外设、采购元器件搭建硬件平台、软件编程(传感器模块、显示模块、WIFI模块和主控程序模块等)、程序及相关功能调试、绘制原理图生成PCB。

主要内容如下：

1.系统介绍。本章主要研究课题背景，国内外相关技术的对比，研究目的和意义，明确主要研究工作；

2.系统方案设计。本章主要设计该系统的总体设计方案，绘制系统总体框架图。进行初步设计，在此基础上，进行器材选型及方案设计；

3.设计硬件电路。本章进行模块选型，并查找模块相关资料。基于此设计原理图，生成PCB。然后设计了系统的总体硬件方案，开始手工焊接模块（最小系统、电路连接）；

4.编写程序。本章熟悉开发平台，了解KEIL软件开发环境。在此基础上，编写主程序和各模块程序（模块化编程）；

5.系统测试。本章对检测控制系统进行了测试，各模块工作正常，基本实现预期功能。

**第二章 方案设计**

**2.1 系统总体设计与分析**

**2.1.1 总体设计的方案**

本文旨在设计一种集数据采集、显示、报警和远程控制于一体的室内空气质量检测系统。系统以STM32及相关外设为主控，传感器采集环境数据，经单片机处理数据后，通过OLED显示屏实时显示，并设置阈值报警。同时，系统的WIFI模块接入网络，无线设备作为服务器与系统完成连接后，通过网络远程获取数据并实时显示。此外，服务器可以发送指令相应地改变LED灯的状态。本系统实现对室内空气中有害气体、温湿度等多个参数的实时检测和阈值报警，帮助人们随时了解室内空气质量。系统的设计由两部分组成：硬件部分和软件部分：

（1）硬件部分

室内空气质量检测系统硬件部分主要由STM32主控、传感器模块、WIFI模块、OLED显示模块、声光报警模块和电源模块六部分组成。控制单元由STM32F103C8T6及相关外设组成；传感器模块由DHT11温湿度传感器和MQ135空气质量传感器组成，用于数据采集；WIFI模块用于系统与无线设备之间的信息传输；OLED显示屏用于显示环境数据；声光报警器由无源蜂鸣器和LED灯组成，发出声光报警；电源模块为各个模块供电。

（2）软件部分

软件设计部分由四部分组成：传感器数据采集与处理部分、WIFI模块与服务器交互部分、显示部分、声光报警部分。由单片机设定采样频率间隔采集数据，经相关算法对数据进行处理后，发送至OLED显示屏。WIFI模块连接热点，与服务器建立连接后，获取数据及远程控制。整个过程中，OLED显示屏用于显示程序执行过程，在改变外界环境后，有害气体超标发出声光报警。

**2.1.2 总体框图**

本系统以STM32平台作为主控制器，主控制器与OLED显示屏通过IIC协议通信，与WIFI模块通过串口协议通信。采集传感器检测到的数据通过主控制器对数据进行处理，向OLED屏发送并显示程序执行状态和数据；WIFI模块连接网络及终端设备后，发送数据到终端设备。设置阈值，当数据值超过了阈值，主控制器驱动声光报警模块进行报警。

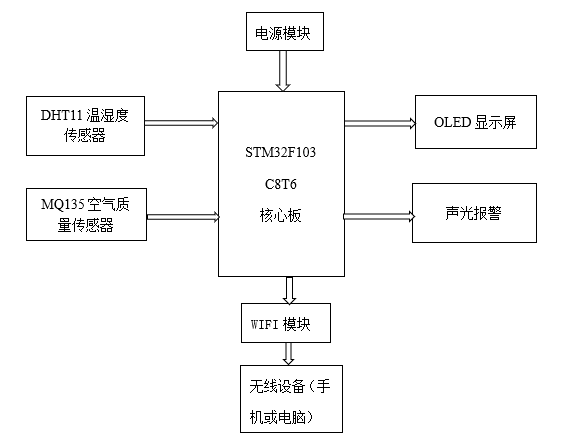
本系统总体框图如图2-1所示：

图2-1 总体框图

**2.2 方案选择**

**2.2.1主控芯片选型**

方案一、可编程逻辑控制器

可编程逻辑控制器(PLC)可以进行逻辑运算、设定时间运算、逻辑时序运算、计数运算和算术运算，并通过内部结构或机制和输出来控制各种器件和器件的运行。

可编程逻辑控制器的优缺点分明。它具有编程简单、功能性高、稳定性强、可靠性高、抗干扰能力强、安装调试方便等优点。缺点主要是目前软PLC技术尚有许多关键技术需要解决，现阶段的知识储备难以达到要求。所以如果以此为设计方案，简单的功能却要很大精力来实现，排除此方案。

方案二、89C51/2单片机

89C51/2单片机是在一片集成电路芯片上集成微处理器、存储器、I/O接口电路，从而构成了单芯片微型计算机。

它具有功耗低、价格低廉、稳定性强、可靠性高、抗干扰性强，编程方式及代码结构简单，应用成熟。但由于本设计中传感器模块和其他外设较多，89C52单片机的引脚不能满足要求，所以该方案不合适。

方案三、STM32系列单片机

与51单片机相比，STM32基本具备51单片机所有优势。对比运算速度，STM32是32位单片机，一次数据宽度为32位，而51一次只能处理8位，主频72 MHz、168 MHz，计算能力大大增强；对比存储空间，STM32内部RAM和ROM(FLASH)比51大得多。此外，STM32片上外设资源丰富，如ADC，IIC不需要外部扩展，大大节省了IO口引脚；特有的DMA控制器，使得CPU可以从繁忙的数据转换中解放出来；还有FAMC存储器接口，可用于扩展LCD液晶显示屏；其外部接口也非常丰富，多串口USB控制SPI、IIC等。

对比以上方案，本设计采用STM32系列的STM32F103C8T6单片机作为主控芯片。

**2.2.2显示屏选型**

方案一：字符液晶屏

如LCD1206 (12\*6 的像素)，LCD12864 (128\*64像素)，其特点是单色，像素粗糙，但是价格低廉，体积小，适合显示小数据用，常用于仪器仪表。

方案二：段码液晶

其特点是价格便宜，功耗低，可以作为数码管的替代品，在实际应用中一般需要定制来满足项目要求，像家中的冰箱或者空调遥控器等都会看到它的身影。

方案三：OLED 显示屏

现在流行的高端显示器和手机屏幕都是用的OLED屏幕，其实它的内部是由非常多的小LED灯组成的，因此它是自发光屏幕，它的优点是像素高，色彩还原度好，但是缺点也很明显，就是制造困难。

综上所述，本次实验所需的屏幕较小，选择性价比最高的是OLED显示屏。

**第三章 硬件电路**

**3.1 硬件电路总原理图**

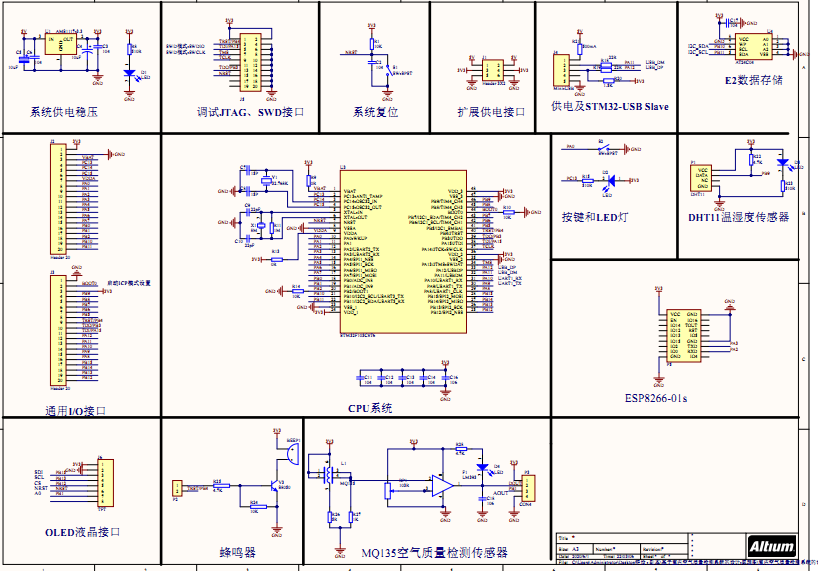


图3-1 MQ135硬件电路总原理图

**3.2 STM32单片机主控模块**

**3.2.1 STM32单片机简介**

本文选用了基于ARM Cortex-M3内核，由ST公司生产的的中低端32位ARM微控制器--STM32F1系列芯片。芯片集成了定时器、SPI、ADC、I2C、USB、USART等外设。

STM32F103 系列微控制器有关资源和特点如下：

* 高达64KB内存，20KB SRAM
* 最高72MHZ的工作频率，其运算速度远远高于51，AVR和MSP430等单片机
* 3个通用定时器（TIM2/3/4/5）
* 1个滴答定时计数器
* 1个高级定时计数器（TIM1）
* 1个 RTC实时时钟
* 2个 SPI通讯接口
* 2个 I2C通讯接口
* 3个 UART通讯接口
* 1个 CAN接口（2.0B主动）
* 37个GPIO端口
* 2\*10路12位ADC
* 7通道DMA控制器
* 支持SWD & JATG仿真接口

**3.2.2 STM32最小系统原理图**

在本设计中，采用STM32F103C8T6最小系统作为主控制器，图3-2为最小系统电路的原理图（包括STM32F103C8T6主控芯片，复位电路，晶振电路，电源电路）。STM32F103C8T6最小系统原理图如图3-2所示：

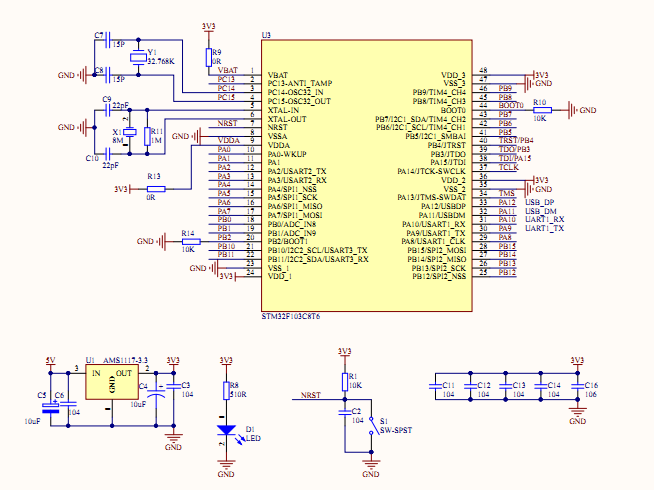


图3-2 STM32F103C8T6最小系统原理图

**3.3 DHT11温湿度传感器**

**3.3.1 DHT11简介**

DHT11数字式温湿度传感器是一种具有校准数字信号输出的温湿度复合传感器。

传感器特性：供电电压：3.3~5.5V DC；输出：单总线数字信号;测量范围：湿度20-90%RH，温度0~50℃；测量精度：湿度+-5%RH，温度+-2℃；分辨率：湿度1%RH，温度1℃。

**3.3.2 DHT11电路原理图**

单线串行接口，系统集成方便快捷。其超小尺寸和极低功耗使其成为此类应用甚至最苛刻应用的最佳选择。产品为4针单排引脚封装，连接方便。其中，VCC接3.3V，NC悬空，DATA为串行数据单总线引脚接PB9，仅占用一个IO口。DHT11电路原理图如图3-3所示：

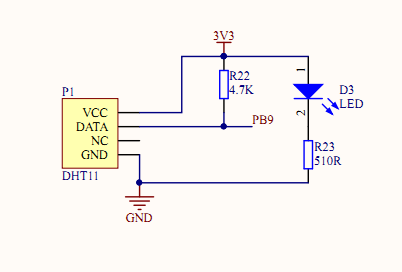


图3-3 DHT11电路原理图

**3.4 MQ135空气质量传感器**

**3.4.1 MQ135简介**

MQ135气体传感器对氨气、硫化物、苯系蒸汽的灵敏度高，对烟雾和其它有害气体的监测也很理想。这种传感器可检测多种有害气体，是一款多功能的低成本传感器。MQ135的基本参数如表3-1所示：

表3-1 MQ135基本参数

|  |  |  |  |
| --- | --- | --- | --- |
| 产品型号 | | | MQ135 |
| 产品类型 | | | 半导体气敏元件 |
| 标准封装 | | | 胶木（黑胶木） |
| 检测气体 | | | 氨气、硫化物、苯系蒸汽 |
| 检测浓度 | | | 10-1000ppm（氨气、甲苯、氢气） |
| 标准电路条件 | 回路电压 | VC | ≤24V DC |
| 加热电压 | VH | 5.0V**±**0.2V ACorDC |
| 负载电压 | RL | 可调 |
| 标准测试条件下气敏元件特性 | 加热电阻 | RH | 32Ω±**3**Ω（室温） |
| 加热功耗 | PH | 900mW |
| 敏感体表面电阻 | RS | 2KΩ-≤20KΩ（在100ppm NH3） |
| 灵敏度 | S | RS （在 air）/RS 大于等于5 |
| 浓度斜率 | a | ≤0.6（R100ppm/R50ppm NH3） |

**3.4.2 MQ135电路原理图**

MQ135模块使用3V3进行驱动，D0输出数字信号，A0输出模拟信号。D0输出相当于一个开关电源，到设定值进行跳转，在本设计中没有用到。此处使用A0 引脚进行模拟信号的输入与输出。产品为4针单排引脚封装，VCC接3.3V，AOUT连接ADC通道1（PB1），DOUT悬空。MQ135空气质量传感器如图3-4所示：

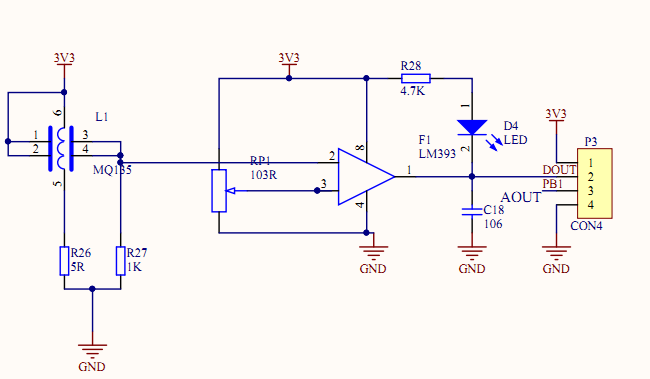


图3-4 MQ135空气质量传感器原理图

**3.5 声光报警模块**

**3.5.1蜂鸣器简介**

本设计采用无源蜂鸣器作为提醒音来使用，无源蜂鸣器内部不含振荡器，我们需要人为的提供驱动信号，其好处是可以提供更加灵活的声音定制，能够产生不同音调的声音（取决于驱动信号的频率）。

**3.5.2 声光报警电路原理图**

无源蜂鸣器VCC接3.3V电源供电，输入接PB4，LED接PC13。 声光报警电路原理图如图3-5所示：

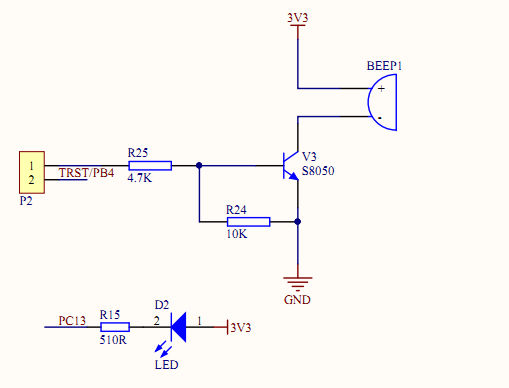


图3-5 声光报警电路原理图

**3.6 WIFI模块**

**3.6.1 ESP8266简介**

ESP8266是一款超低功耗的UART-WIFI透传芯片。ESP8266的封装方式多种多样，专门为移动设备和物联网应用开发。它可以自发地生成局域网，类似于路由器，还可以连接到WIFI无线网络进行局域网或远程通信，从而实现联网功能。

ESP8266可广泛应用于智能电源插头、家庭自动化、智能家居、手持设备、工业无线控制、可穿戴电子产品等。

本设计选用ESP8266-01S作为无线组网方式，ESP8266-01s参数如表3-2所示：

表3-2 ESP8266-01s基本参数

|  |  |  |
| --- | --- | --- |
| 模块 | 型号 | ESP8266-01s |
| 主芯片 | ESP8266 |
| 硬件参数 | 硬件接口 | UART，IIC，PWM，GPIO，ADC |
| 工作电压 | 3.3V |
| 工作电流 | 持续发送下  平均值：~70mA,峰值: 200mA  正常模式下  平均: ~12mA,峰值: 200mA  待机：<200uA， |
| 工作温度 | -40℃~125℃ |
| 存储环境 | 温度：<40℃，相对湿度：<90%R.H. |
| 尺寸 | 板载PCB天线：14.3mm\*24.8mm\*1mm； |
| 串口透传 | 传输速率 | 110-921600bps |
| TCP Client | 5个 |
| 软件参数 | 无线网络类型 | STA/AP/STA+AP |
| 固件升级 | 本地串口，OTA远程升级 |
| 网络协议 | IPv4, TCP/UDP/FTP/HTTP |
| 用户配置 | AT+指令集, Web 页面 Android/iOS 终端, Smart Link 智能配置APP |

**3.6.2 ESP8266电路原理图**

外出引脚有8个，连线时中间四根线悬空，TXD和RXD与开发板串口引脚交叉连接。VCC和GND与开发板3V3与GND相连接。ESP8266电路原理图如图3-6所示：

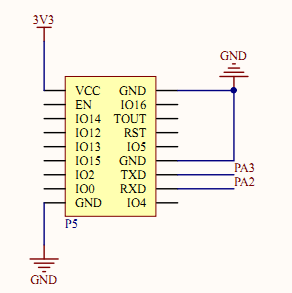


图3-6 ESP8266电路原理图

**3.7 液晶模块**

**3.7.1 OLED显示屏简介**

本文采用的是分辨率为128\*64的0.96英寸OLED屏幕，可显示图片、符号、中西文字等，采用SSD1306内部驱动屏幕，SSD1306支持6800/8000并口协议以及三线/四线SPI和二线制IIC，使用并行口的效果最好，它可以达到最大的刷新速度，适用于一些低帧率动画的显示，此硬件使用IIC通信。

**3.7.2 OLED显示屏电路原理图**

外出引脚有四个，VCC接3.3V，其中SDA、SCL接单片机IIC接口，即SCL接PB10，SDA接PB11。OLED显示屏电路原理图如图3-7所示：

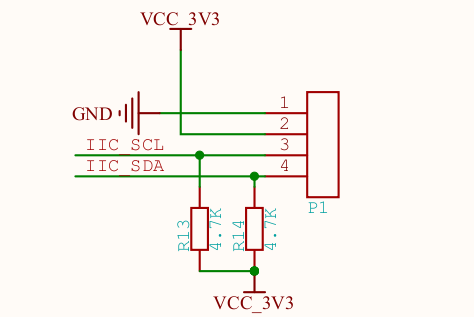


图3-7 OLED显示屏电路原理图

注意：OLED显示屏不同于LCD，OLED上电是没有反应的，需要程序驱动才会有显示！

**第四章 软件程序设计**

**4.1开发软件**

选择合适的开发环境，可以加快开发进度，为开发者提供更大的便利。目前，嵌入式开发工程师最常用的开发工具是keil5。Keil MDK-ARM是美国Keil软件公司（现已被ARM公司收购）出品的支持ARM微控制器的一款IDE（集成开发环境）。MDK-ARM包含了工业标准的Keil C编译器、宏汇编器、调试器、实时内核等组件。与51单片机的开发不同，STM32需要对Keil软件进行手动导包，keil5支持在线仿真和直接下载程序。此外，由于keil5软件本身具备局限性，市面上有很多基于java开发环境的辅助开发软件，这给嵌入式开发者再次带来了福音。

使用Keil来开发项目，开发周期较于其他软件平台相似，大致有以下几个步骤：

1.创建一个工程，选择一款开发芯片，设置启动文件等相关配置；

2.编写C或者汇编源文件；

3.编译应用程序；

4.修改源程序中的错误；

5.联机调试。

**4.2 系统的主程序设计**

本设计由STM32主控采集单通道ADC接口模拟输入电压，与MQ135空气质量传感器，根据模数转换算法，得到空气质量的值；连接单总线的DHT11数字式温湿度传感器，由内部时序获取温湿度；OLED显示屏通过IIC连接与单片机进行通信；WIFI模块通过串口发送AT指令进行配置，连接热点和服务器；将采集得到的环境数据发送至OLED显示屏上实时显示，同时将环境数据实时发送至无线设备；本设计还可以对空气质量划分等级和设置，有害气体超标触发声光报警器；此外，可以通过在服务器端发送相关指令控制开发板LED的状态变化。

主函数流程图如图4-1所示：

1.初始化定时器、系统时钟、设置中断的优先级，初始化各个模块，初始化串口1和串口2；

2.在程序中设置WIFI名称及密码，服务器端的IP地址及端口号，由STM32发送AT指令控制WIFI模块连接服务器；

3.如果OLED显示屏显示WIFI模块初始化成功、联网成功及连接服务器端成功，则开始将STM32采集到的环境数据通过IIC通信发送至OLED显示屏上实时显示数据，同时将数据通过WIFI模块发送至服务器端，在无线设备接收并实时显示环境信息；

4.调用MQ135空气质量传感器进行等级划分函数，如果有害气体超过阈值，触发声光报警；

5.此外，在本设计中增加通过在服务器端发送函数指令控制开发板LED灯的状态功能。

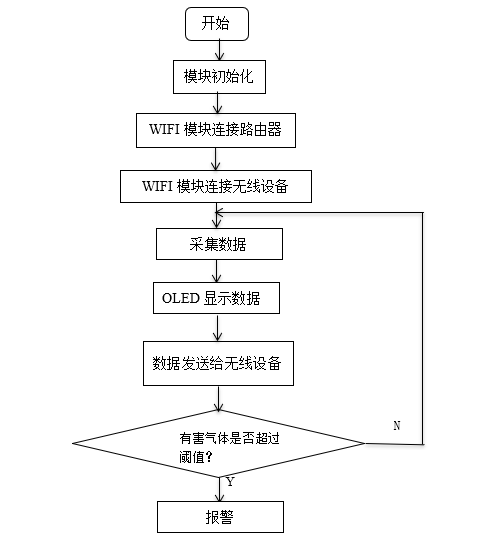


图4-1主函数流程图

**4.3 DHT11温湿度传感器模块设计**

DHT11温湿度传感器模块是集成模块，DATA引脚直接连接开发板即可。按照工作时序编写数据采集程序，采用库函数的方式编程。温湿度获取及校验部分程序如下所示：

/\*\*

\* 功能：读取DHT11采集到的数据

\* 参数：

\* mode：返回数据类型 用户可以指定返回温湿度当中的某一个或者所有

\* 返回所有数据时，湿度在高八位，温度在低八位

\* 返回值：采集到的字节数据

\*/

u16 readDHT11(void)

{

u8 i = 0;

u8 Hum\_H,Hum\_L,Temp\_H,Temp\_L,CheckByte; //依次是湿度整数、小数部分、温度整数、小数部分、校验和字

setLineOut(); //发起读取开始信号

LINE\_LOW();

Delay\_ms(25);

LINE\_HIGH();

setLineIn();

while(GPIO\_ReadInputDataBit(DATA\_LINE\_PORT,DATA\_LINE\_PIN) && ++i<10) //主机释放总线空闲阶段

{

Delay\_us(10);

}

i = 0;

while(!GPIO\_ReadInputDataBit(DATA\_LINE\_PORT,DATA\_LINE\_PIN) && ++i<10) //传感器响应阶段

{

Delay\_us(10);

}

i = 0;

while(GPIO\_ReadInputDataBit(DATA\_LINE\_PORT,DATA\_LINE\_PIN) && ++i<10) //传感器拉高通知主机准备阶段

{

Delay\_us(10);

}

Hum\_H=readByte(); //读取有效湿度值

Hum\_L=readByte(); //湿度小数部分为0

Temp\_H=readByte(); //读取有效温度值

Temp\_L=readByte(); //温度小数部分为0

CheckByte=readByte(); //校验和

if(Hum\_H+Hum\_L+Temp\_H+Temp\_L==CheckByte) //计算校验和

{

return Hum\_H<<8 | Temp\_H;

}else

{

return 0xFFFF; //返回无效值，标识采集有误

}

}

**4.4 MQ135空气质量传感器模块设计**

MQ135空气质量传感器是集成化模块，A0引脚直接连接开发板ADC\_IN9即可。分为ADC的初始化配置部分和获取ADC转换结果部分。

ADC的初始化配置步骤：初始化对应通道的GPIO位ADC输入模式；配置ADC， ADC时钟分频（官方建议不要超过14MHZ）,ADC 工作模式，是否连续转换，扫描模式，数据寄存器对齐模式转换序列等；校准ADC并开启ADC。初始化ADC1\_IN9如下所示：

/\*\*

\* 功能：初始化ADC1\_IN9

\* 参数：None

\* 返回值：None

\*/

void initADC(void)

{

GPIO\_InitTypeDef GPIO\_InitStructure; //定义GPIO初始化结构体

ADC\_InitTypeDef ADC\_InitStructure; //定义ADC初始化结构体

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOB|RCC\_APB2Periph\_ADC1 , ENABLE); //使能外设时钟 /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*配置PB1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_1;

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_AIN; //模拟输入引脚

GPIO\_Init(GPIOB, &GPIO\_InitStructure);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*配置ADC\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

ADC\_InitStructure.ADC\_Mode=ADC\_Mode\_Independent;

//配置为独立工作模式

ADC\_InitStructure.ADC\_ScanConvMode=DISABLE; //设置ADC工作在单通道

ADC\_InitStructure.ADC\_ContinuousConvMode=DISABLE; //设置ADC为单次转换

ADC\_InitStructure.ADC\_ExternalTrigConv=ADC\_ExternalTrigConv\_None; //失能外部触发，使用软件人为触发

ADC\_InitStructure.ADC\_DataAlign=ADC\_DataAlign\_Right; //设置ADC数据寄存器右对齐

ADC\_InitStructure.ADC\_NbrOfChannel=1; //规则转换通道数，我们只使用一个

ADC\_Init(ADC1,&ADC\_InitStructure); //生效ADC设置

ADC\_Cmd(ADC1,ENABLE);

//设置ADC\_CR2中的ADON位，使其给ADC1上电 /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 1. 配置ADCx规则转换的通道以及转换序列

\* 2. 配置转换序列，因为我们只用到了一个通道，因此转换序列要设1

\* 3. 设置ADCx的采样周期

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

ADC\_RegularChannelConfig(ADC1,ADC\_Channel\_9,1,ADC\_SampleTime\_239Cycles5);

ADC\_ResetCalibration(ADC1); //复位ADC1校准

while(ADC\_GetResetCalibrationStatus(ADC1)); //等待复位校准完成

ADC\_StartCalibration(ADC1); //开始ADC1的校准

while(ADC\_GetCalibrationStatus(ADC1)); //等待校准完成

}

初始化好ADC1\_IN9后，获取ADC1\_IN9单次AD转换结果，获取ADC1\_IN9多次转换平均值。本设计需要给每个ADC通道提供电源，本次设计使用杜邦线从开发板的GND和3V3取信号。由于显示的电压值过程较小，所以在获取数据时将数据扩大1000倍。获取数据程序如下所示：

/\* AOUT输出模拟信号，模拟量输出随浓度增加而增加，浓度越高电压越高\*/

ADC\_data = getConvValueAve(5,1000)\*3.3/4096\*1000;

**4.5 WIFI模块设计**

ESP8266是一款UART-WIFI透传芯片。使能RX和TX引脚GPIO时钟和UART时钟，初始化GPIO，并将GPIO复用到UART上，配置波特率及其它参数，配置终端控制申请并使能UART接收中断，使能UART，最后在UART接收中断服务函数中实现数据接收与发送。串口2中断服务函数如下所示：

/\*\*

\* 功能：串口2中断服务函数

\* 参数：None

\* 返回值：None

\*/

void USART2\_IRQHandler(void)

{

static u8 i = 0;

if(USART\_GetITStatus(USART2, USART\_IT\_RXNE))

//判断接收数据寄存器是否有数据

{

RXBuffer[i++] = USART\_ReceiveData(USART2);

if(i==RXBUFFER\_LEN)

//超出接收缓冲范围,可能的情形是ESP8266复位,为防止溢出必须设置索引

{

i = RXBUFFER\_LEN-1;

}

}

if(USART\_GetITStatus(USART2, USART\_IT\_IDLE))

{

USART\_ReceiveData(USART2); //读一次UART可以清除空闲标志位

i = 0;

processServerBuffer();

}

}

本次设计直接使用乐鑫提供的库函数来编写程序，AT开发是使用外部单片机通过串口和WIFI模块通信，其通信的指令中包含AT前缀，这种开发方式虽然成本比SDK开发高，但是开发周期短。其常用AT指令：

1.测试模块是否正常（AT）；

2.开启/关闭 回显；

3.设置AP模式及AP参数（AT+CWMODE，AT+CWSAP\_DEF）；

4.设置为Station模式（AT+CWMODE=1,AT+CWJAP）；

5.使用SmartConfig为设备配网（AT+CWSTARTASMART=<type>）。

初始化ESP8266如下所示：

/\*\*

\* 功能：初始化ESP8266

\* 参数：None

\* 返回值：初始化结果，非0为初始化成功,0为失败

\*/

u8 initESP8266(void)

{

sendString(USART2,"+++"); //退出透传

Delay\_ms(500);

sendString(USART2,"AT+RST\r\n"); //重启ESP8266

Delay\_ms(500);

if(checkESP8266()==0) //使用AT指令检查ESP8266是否存在

{

return 0;

}

memset(RXBuffer,0,RXBUFFER\_LEN); //清空接收缓冲

sendString(USART2,"ATE0\r\n"); //关闭回显

if(findStr(RXBuffer,(u8\*)"OK",200)==0)

//如果在接收缓冲区中找不到"OK",则设置不成功

{

return 0;

}

return 1; //设置成功

}

**4.6 OLED显示屏模块设计**

通信方式：OLED显示屏使用I2C通信。I2C通信协议和其他通信协议的功能一样，都是以某种实现约定好的时许进行数据传输的通信协议，因此主机和多个从机之间的通信线只要两条就够了，分别为时钟线和数据线昆明，IIC代码主机通过各个从机的地址选通对应从机。

主从结构示意图如图4-2所示：

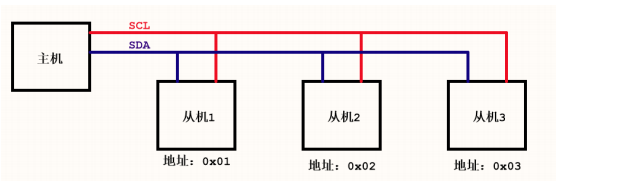


图4-2 I2C主从结构

通讯步骤：

1.主机发送一个起始信号通知各从机就位；

2.主机发送从机地址和读写标志位（从机地址和读写标志位一共占用8位，地址占用高8位，读写标志位占用最低位）；

3.从机给主机回复响应；

4.如果是写模式，主机发送一字节数据等待从机响应，主机收到响应之后如果还有数据要发送就继续发送第二段数据等待响应…直到发送完成；

5.如果是读模式，此时主机读取从机发来的数据，并给从机响应，如果从机还有数据要发送（接着汇报第二段），主机接着读取然后发送响应给从机；

6.主机给从机一个停止信号。

对于屏幕来说，主要功能就是显示ASCII字符，显示整数，显示汉字，显示图片，设置显示方向，息屏显屏操作以及屏幕反色。这里介绍OLED控制指令：

1.设置页地址（范围0-7）

设置页坐标就是设置OLED屏幕的y轴坐标，指令为0xB0+y，y是要设置的坐标。

2.设置段坐标（0-127）

即设置OLED的ｘ轴坐标，对应指令有两条，每条记录段坐标的４位，加起来一共8位。对应指令位设置低四位的0x00和设置高四位的0x10，即0x10 的低四位作为x轴坐标的高四位，0x00的低四位作为x轴坐标的低四位。

3.开启/关闭OLED显示0xAF为开启显示，0xAE为关闭显示。

4.设置OLED反色，正常显示为0xA6，反色显示为0xA7。

5.设置端重映射，即屏幕水平翻转，指令为A0/A1。

6.设置COM扫描方向，即垂直翻转，指令为C0/C8。

OLED初始化函数如下所示：

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*显示屏驱动函数\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\* \* 功能：初始化OLED

\* 参数：None

\* 返回值：None

\*/

void initOLED(void)

{

writeCommand(0x81); //设置亮度

writeCommand(0xFF); //亮度值最大 复位默认0x7F

writeCommand(0xA1); //设置段映射方式即设置是否水平翻转 A0表示翻转 通常和C0一起使用

writeCommand(0xC8); //设置COM扫描模式即设置是否垂直翻转 C0表示翻转 通常和A0一起使用

writeCommand(0x8D); //电荷泵使能

writeCommand(0x14);

writeCommand(0xAF); //开屏幕，默认是关闭的就和没上电一样，所以要手动开启

}

**4.7 报警器模块设计**

输出比较生成PWM的软件配置过程：

1.设置定时计数器初始化结构体，包括定时器时钟，周期，是否使能各个终端等；

2.设置输出比较初始化结构体，包括输出比较模式，输出的极性；

3.设置PWM输出对应的GPIO为服用推挽输出；

4.最后可以在程序运行时通过重装载寄存器ARR来改变PWM频率，通过更改捕获比较寄存器CCRx 来改变PWM占空比。

初始化函数分为两部分设置，第一部分时重映射TIM3到PB4引脚，并初始化PB4引脚为复用推挽输出，第二部分是设置蜂鸣器的音量和音调。蜂鸣器初始化函数如下所示：

/\*\* \* 功能：设置蜂鸣器的音量

\* 参数：

\* vol:音量大小 0-99 0就是静音 99就是最大音量

\* 返回值：None

\*/

void setBuzzerVol(u8 vol)

{

TIM\_SetCompare1(TIM3,getPeriod(TIM3)\*vol/100);

}

蜂鸣器音调设置函数，其原理就是设置定时器的重装载寄存器ARR的值，定时器时钟设置周期是10us，记100000次就是1s即1HZ，由此而知，nHZ对应的周期值应该是100000/n。蜂鸣器音调设置函数如下所示

/\*\* \* 功能：设置蜂鸣器的音调

\* 参数：

\* tone:音调大小单位Hz，建议100-10000

\* 另外我们封装了七音阶音调，方便大家使用

\* 注意： 这里是以定时器时钟频率10us进行计算的

\* 返回值：None

\*/

void setBuzzerTone(u16 tone)

{

setPeriod(TIM3,100000/tone);

}

**第五章 系统制作与调试**

**5.1 硬件制作**

**5.1.1 AD简介**

Altium Designer提供唯一的统一应用解决方案，集成了电子产品集成开发所需的所有必要技术和功能。它的前身就是Protel，在学校里学习的就是Protel。现在很多工程师还是使用Protel设计PCB，AD是Protel的升级版。AD画PCB图步骤：

1．新建封装库，在封装库中新建元件封装；

2．新建原理图库，在元件库中新建元器件；

3．新建工程组，在工程组中新建PCB工程，在PCB工程中添加原理图文件；

4．在原理图文件中添加元器件，连线等进行原理图设计；

5．在PCB工程中添加PCB文件，设置板框；

6．将原理图导入到PCB；

7．设置设计规则，在PCB文件中布局，布线，铺铜等进行PCB设计。

**5.1.2 系统PCB板的设计**

系统PCB版图如图5-1所示：

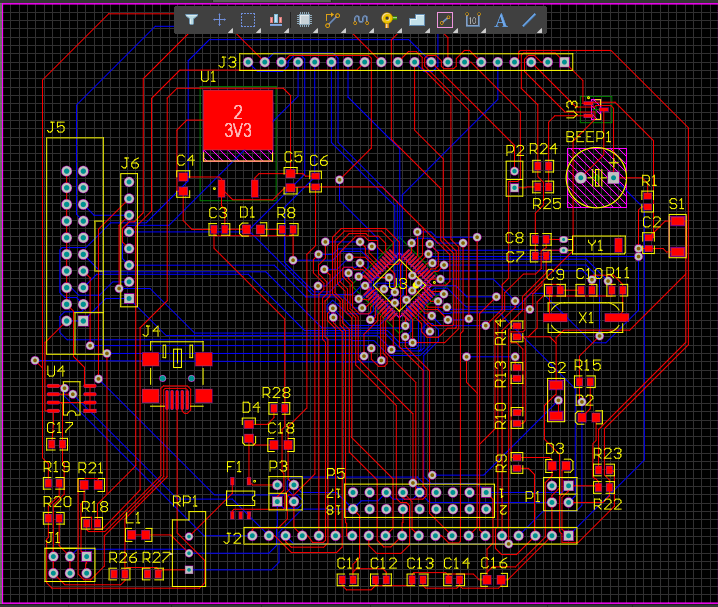


图5-1 系统PCB版图

**5.2 系统硬件调试**

本设计以STM32F103C8T6为开发平台，使用其相关外设完成环境检测并将数据发送至OLED显示屏及终端设备实时显示，最后基于环境检测数据提供阈值报警功能。

其中下载接口使用SWD接口，OLED显示屏连接IIC2的SCL和SDA（PB11和PB10），WIFI模块使用串口2 交替连接RX和TX（PA2和PA3），DHT11模块的总线连接PB9，MQ135空气质量传感器连接ADC通道9（PB1），蜂鸣器IO连接PB4，其余VCC和GND使用开发板引出电源3V3和GND。

实物图正面和背面分别如图5-2、5-3所示：

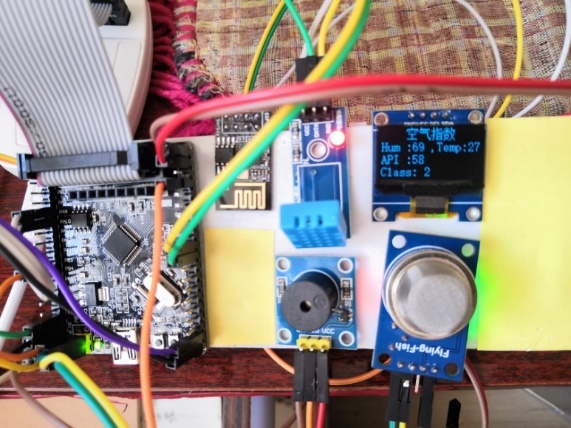


图5-2 实物图正面

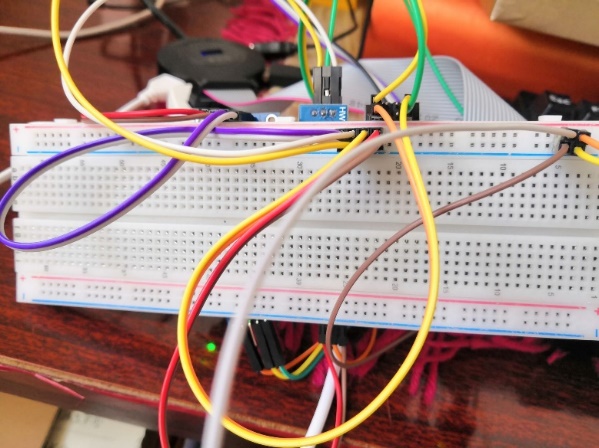


图5-3 实物图背面

使用USB接口为开发板供电，开发板LED灯亮，表示开发板成功上电；杜邦线的母头连接开发板，公头连接实物图背面面包板，使用杜邦线为其他模块供电，此时观察各个模块，模块电源指示灯亮起，表示各个模块均成功供电。

**5.3 软件及联机调试**

**5.3.1 主控程序调试**

本系统使用OLED显示屏来反映主控程序执行过程，程序烧录开发板，复位后，屏幕上依次显示init ok!, conn ap ok!, con server ok! 表示WIFI模块初始化成功，WIFI模块连接热点成功，WIFI模块连接服务器成功。此时程序跳转环境数据界面，如图5-4所示：

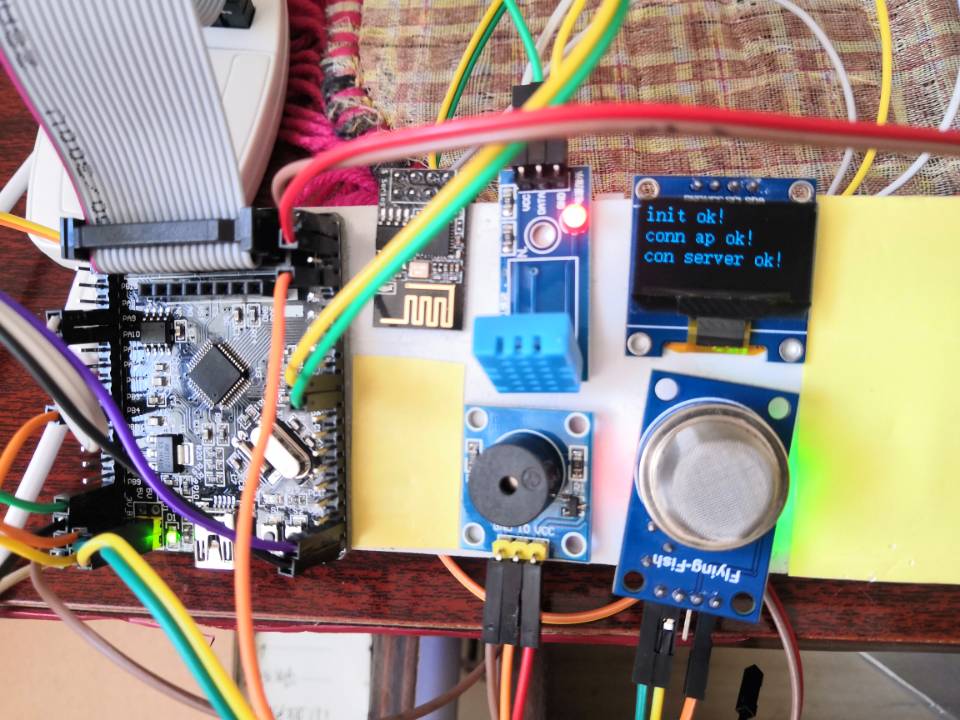
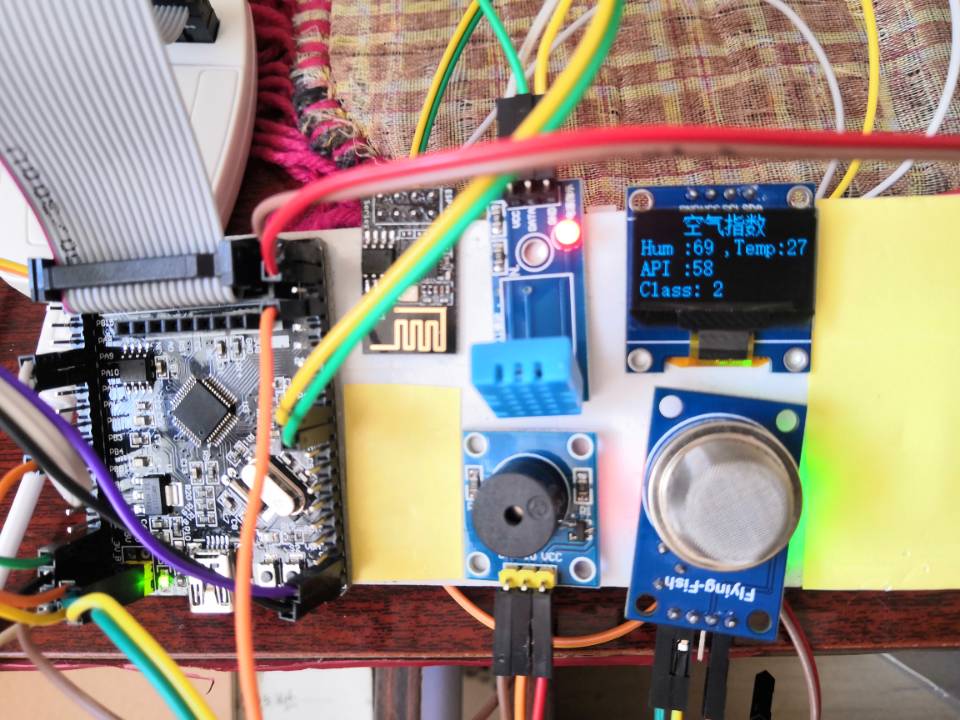
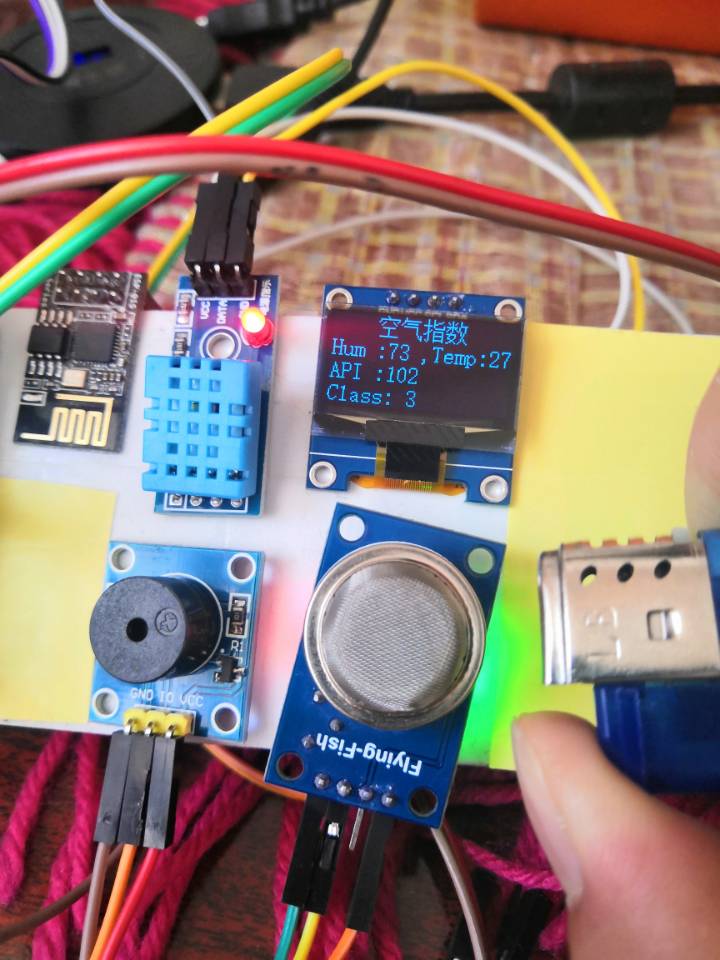


图5-4 WIFI模块初始化

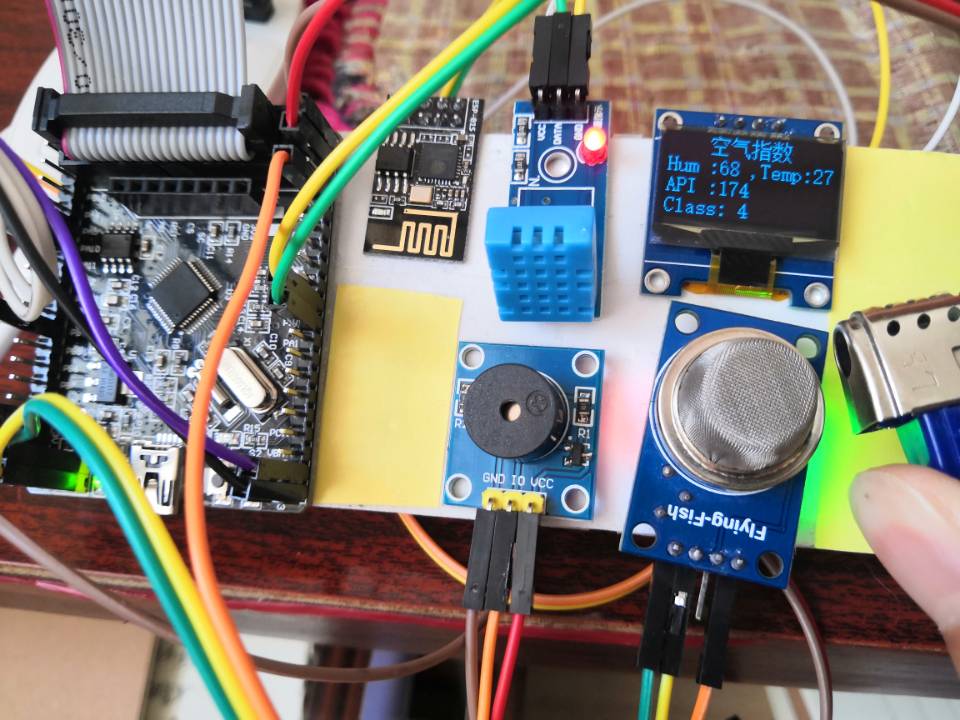
与此同时使用打开打火机对准MQ135空气质量传感器模块，观察OLED显示屏上API数据及对应CLASS等级，当等级增加到5或者6时，观察蜂鸣器发出声音。打火机释放可燃气体观察OLED显示屏空气指数（API）：



（a）Class为2时



（b）Class为3时



（c）Class为4时

图5-5 有害气体变化过程

**5.3.2 短信息发送调试**

在OLED显示屏上显示初始化成功，连接热点成功及连接服务器成功，即可在安心可透传云上显示实时环境数据，如图5-6所示：



图5-6 终端设备实时显示

为了展示可以通过在远程检测到环境数据，并且远程执行操作，本次实验通过在服务器端发送指令控制LED灯的状态变化。发送LED\_ON开发板LED灯亮，发送LED\_OFF 此时LED灯熄灭，发送blinkLED此时LED闪烁10 次后熄灭。发送指令控制开发板LED灯如图5-7所示：

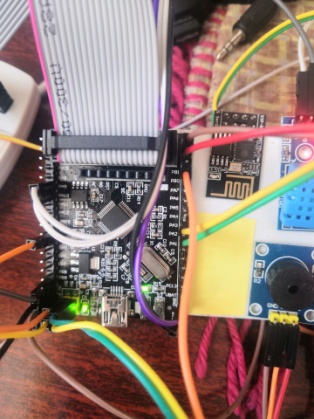


图5-7 发送指令控制开发板LED灯

**第六章 总结与展望**

本次毕设历时一学期，从前期课题选题、查阅文献、中英文翻译、撰写开题报告。确定好课题方向之后，在指导老师的指导下，开始学习相关技术（包括STM32及相关外设的使用、传感器相关知识和AD绘制原理图、制作PCB）。中期答辩之后，购买元器件，绘制原理图，着手软硬件设计。经调试，基本功能均已实现，汇总研究成果，撰写毕业论文。

本设计主要实现的功能如下：

（1）OLED显示屏实时显示环境温湿度、空气指数；

（2）无线设备通过公网连接，获取环境温湿度、空气指数；

（3）自动监测有害气体，默认空气指数阈值为200，超过自动报警；

（4）无线设备发送相应指令，远程控制开发板LED灯的亮灭。

本次设计适合广大想要用于家居室内环境检测的人群。其体积小、功能完善，并配有远程显示与控制的检测系统。

整个系统虽然基本实现了功能要求，但仍有部分内容需要改善。一个产品想要市场化必然需要符合用户的使用习惯，本系统的远程监测与控制仅以LED状态展示功能，应该设计一种对有害气体开关的控制系统；其次，本次设计中WIFI模块连接热点和服务器，仅可以在程序中实现，缺乏用户配网功能，应该使用物联网相关技术加以完善。希望通过未来的努力，该系统的功能更完善、实用性更强。

**致谢**

本设计从选题、构思、资料收集到最终毕业论文的各个方面始终是在导师的精心指导和周密安排下进行的。在这一个学期里，导师严谨的学习态度、丰富的学科知识、科学的学术思维、端正的工作态度、一丝不苟的科研精神，给我留下了深刻印象，感谢导师给我树立了一个终身学习的榜样。同时，我要感谢那些教会我知识的老师，正是他们孜孜不倦的教学给了我接触科研、完成学业的机会。在我大学的四年里，辅导员总是督促我在学习和生活中取得进步，让我在有限的青春书写青春无悔。在此，我向各位老师表示由衷的感谢。

我要感谢同学们在设计中给予的极大鼓励和帮助。在与他们的交流中，我开阔了眼界，开阔了思路，这样我才能更好地完成本次毕设。

最后，我要感谢所有在百忙之中抽出时间来审阅和评论这篇论文的老师们，感谢你们参与我的论文答辩！

**参考文献**

[1] 孙占鹏, 李佳, 欧文. 多传感器室内环境监测系统[J].传感器与微系统, 2017, 36(1): 87-90.

[2] 彭煜. 基于STM32的便携式室内空气质量监测器的研究[D]. 暨南大学, 2013.

[3]范奥博. 基于ARM的室内空气质量在线监测系统研究[D]. 浙江理工大学, 2014.

[4] 沈家磊. 环境实时监测及控制系统[D].天津科技大学, 2015.

[5] 于波。基于单片机的室内环境监测系统设计[D]。中国海洋大学，2011。

[6] 彭江鹏. 基于STM32的室内空气监测系统的设计与实现[D]. 华中师范大学, 2015.

[7] 郭联金, 虞晓琼等. 室内空气质量监测系统的设计与实现[J]. 微型机与应用, 2016, 35(18): 99-102.

[8] 高凡, 张希, 陈辰星等. 穿戴式实时环境气体监测系统[J].中国生物医学工程学报, 2015, 34(6): 654-661.

[9] 卢俊杰, 陈友荣, 张磊等. 基于无线传感器的室内环境监测系统设计[J]. 浙江树人大学学报, 2016, 16(4): 1-5.

[10] 吴沛. 基于ZigBee家居环境实时监测的研究与实现[D]. 杭州电子科技大学, 2015.

[11] 岳锐. 基于物联网的居家环境监测系统设计[D]. 中北大学, 2017.

[12] 刘坤, 高征红, 晁阳. Protel 99SE电路设计实例教程[M]. 北京: 清华大学出版社, 2010.

[13] 宋雪松. 手把手教你学51单片机[M]. 北京: 清华大学出版社, 2014.

[14] WenJie Tian, Yu Geng. Environment Monitoring System of Household Security Robot Based on Wireless Mesh Network[C]. International Conference on Networks Security, Wireless Communications and Trusted Computing, 2009: 176-179.

[15] Anuj Kumar, I.P.Singh, S.K.Sud. Energy Efficient and Low-Cost Indoor Environment Monitoring System Based on the IEEE 1451 Standard[J]. IEEE SENSORS JOURNAL, 2011, 11(10): 2598-2610.

**附录 系统源程序**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 文件：main.c

\* 功能：主函数入口

\* 日期：2018-02-16

\* 作者：zx

\* 版本：Ver.1.0 | 最初版本

\* 官博：http://fengmeitech.club

\* Copyright (C) 2017 zx. All rights reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "stdio.h"

#include "string.h"

#include "DELAY/Delay.h"

#include "ADC/ADC.h"

#include "LED/LED.h"

#include "KEY/Key.h"

#include "NVIC/NVIC.h"

#include "IIC/IIC.h"

#include "OLED/OLED.h"

#include "DHT11/DHT11.h"

#include "RELAY/Relay.h"

#include "UART/uart.h"

#include "NetWork.h"

#include "BUZZER/Buzzer.h"

#include "TIMx/TIMx.h"

#include "BUZZER/MusicScore.h"

static u8 Air\_Just(u16 data);

extern u8 TXBuffer[TXBUFFER\_LEN];

int main(void)

{

u8 i = 0;

u8 key\_value;

u8 class;

u16 DHT11\_data; //存储DHT11传感器采集数据

u16 ADC\_data; //存储ADC数据

/\*初始化各外设\*/

initSysTick();

initADC();

initUART();

initUART2();//单片机与ESP8266通信方式

initLED();

initKey();

initRelay();

initDHT11();

initIIC();

initOLED();

initNVIC(NVIC\_PriorityGroup\_2);//开启UART2的接收和空闲中断

/\*定时器3时钟周期10us,中断周期1ms,即1KHz，禁止更新中断和捕获通道1中断\*/

initTIMx(TIM3,719,99,TIM\_IT\_Update|TIM\_IT\_CC1,DISABLE);

TIM\_ARRPreloadConfig(TIM3,ENABLE);

initTIM3OC1(90);

/\*蜂鸣器默认最大音量，频率1000Hz\*/

initBuzzer(MUTE,1000);

/\*使用ADC初始化函数只是为了将连接PA15的LED灯关掉\*/

initADC();

formatScreen(0x00);//清屏

/\*打印ESP8266启动信息到OLED\*/

if(initESP8266()!=0)

{

showString(0,0,(u8\*)"init ok!",FONT\_16\_EN);

}else

{

showString(0,0,(u8\*)"init error!",FONT\_16\_EN);

}

if(connectAP(SSID,PWD)!=0)

{

showString(0,2,(u8\*)"conn ap ok!",FONT\_16\_EN);

}else

{

showString(0,2,(u8\*)"con ap error!",FONT\_16\_EN);

}

if(connectServer (TCP,IP,PORT)!=0)

{

showString(0,4,(u8\*)"con server ok!",FONT\_16\_EN);

}else

{

showString(0,4,(u8\*)"con server error!",FONT\_16\_EN);

}

Delay\_ms(1000);//让数据保持一段时间

formatScreen(0x00);

showCNString(32,0,(u8\*)"空气指数",FONT\_16\_CN);

showString(0,2,(u8\*)"Hum :",FONT\_16\_EN);

showString(0+8\*8,2,(u8\*)",Temp:",FONT\_16\_EN);

showString(0,4,(u8\*)"API :",FONT\_16\_EN);

showString(0,6,(u8\*)"Class:",FONT\_16\_EN);

while (1)

{

if(++i>=200) //每隔2S左右采集一次 建议采集间隔最小不要小于2S

{

i = 0;

/\*采集传感器数据\*/

DHT11\_data = readDHT11();

/\* AOUT输出模拟信号，模拟量输出随浓度增加而增加，浓度越高电压越高\*/

ADC\_data = getConvValueAve(5,1000)\*3.3/4096\*1000;

class = Air\_Just(ADC\_data);

if(class == 5||class == 6)

{

setBuzzerVol(MAXVOL);

setBuzzerTone(100);

}

else

{

setBuzzerVol(MUTE);

}

/\*输出传感器数据到服务器\*/

memset(TXBuffer,0,TXBUFFER\_LEN);

sprintf((char\*)TXBuffer,"Hum :%d,Temp :%d\nAPI:%d\nClass:%d\n",DHT11\_data>>8,DHT11\_data&0x00FF,ADC\_data,class);

sendBuffertoServer(TXBuffer);

/\*同时显示在OLED上\*/

showNumber(40,2,DHT11\_data>>8,DEC,3,FONT\_16\_EN);

showNumber(0+8\*8+8\*6,2,DHT11\_data&0x00FF,DEC,3,FONT\_16\_EN);

showNumber(0+8\*5,4,ADC\_data,DEC,4,FONT\_16\_EN);

showNumber(0+8\*7,6,class,DEC,2,FONT\_16\_EN);

}

key\_value = getKeyValue(KEY\_RELEASE);

//UP键按下断开TCP连接

if(key\_value==KEY\_UP)

{

disconnectServer();

}

/\*

//DOWN键按下恢复出厂

if(key\_value==KEY\_DOWN)

{

restoreESP8266();//恢复出厂同时重启单片机

}

\*/

Delay\_ms(10);//基础延时

}

}

static u8 Air\_Just(u16 data)

{

if(data<=50)

{

return 1;

}

else if((data>50)&&(data<=100))

{

return 2;

}

else if((data>100)&&(data<=150))

{

return 3;

}

else if((data>150)&&(data<=200))

{

return 4;

}

else if((data>200)&&(data<=300))

{

return 5;

}

else

{

return 6;

}

}