西安邮电大学
毕业设计（论文）

题目：基于单片机的蔬菜大棚温度监测的设计

学院：自动化

专业：自动化

班级：自动 1403

学生姓名：邹智宏

学号：06141107

导师姓名：陈维佳　职称：助理工程师

起止时间：2017 年 12 月 5 日至 2018 年 6 月 10 日

毕业设计（论文）声明书

本人所提交的毕业论文《基于单片机的蔬菜大棚温度监测的设计》是本人在指导教师指导下独立研究、写作的成果，论文中所引用他人的文献、数据、图件、资料均已明确标注；对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式注明并表示感谢。

本人完全理解《西安邮电大学本科毕业设计（论文）管理办法》的各项规定并自愿遵守。

本人深知本声明书的法律责任，违规后果由本人承担。

论文作者签名：

日期：　　年　月　日

西安邮电大学本科毕业设计(论文) 选题审批表

<table>
<tr><td>申 报 人</td><td>陈维佳</td><td>职 称</td><td>助理工程师</td><td>学 院</td><td colspan="3">自动化</td></tr>
<tr><td>题目名称</td><td colspan="7">基于单片机的蔬菜大棚温度监测系统的设计</td></tr>
<tr><td>题目来源</td><td>科研</td><td colspan="2">是</td><td>教学</td><td></td><td>其它</td><td></td></tr>
<tr><td>题目类型</td><td>硬件设计</td><td>是</td><td>软件设计</td><td></td><td>论文</td><td>艺术作品</td><td></td></tr>
<tr><td>题目性质</td><td colspan="2">实际应用</td><td colspan="2">是</td><td colspan="2">理论研究</td><td></td></tr>
<tr><td>题目简述</td><td colspan="7">该课题可以使学生将课堂上学到的理论知识与实际应用结合起来，对电子电路、电子元器件、单片机等方面的知识进一步加深认识，同时在软件编程、排错调试、焊接技术、相关仪器设备的使用技能等方面得到较全面的锻炼和提高。使学生掌握单片机的内部功能模块的应用，如定时器/计数器、中断、片内外存贮器、I/O 口、串行口通信等。提高学生在单片机应用方面的实践技能和科学作风;培育学生综合运用理论知识解决问题的能力，实现理论结合实际，学以至用的原则。</td></tr>
<tr><td>对学生知识与能力要求</td><td colspan="7">传感器数据收集，能做到实时监测数据。数据上传的实现，通过网口无线传输到有接收器的主机端。数据处理的实现，通过后台将数据处理完毕存入数据库中。用户交互界面的程序编写，通过后台接口将数据库中的数据在网页中展示。</td></tr>
<tr><td>预期目标</td><td colspan="7">本题目最终将以网页展示的形式，实现对发送端数据与信息的实时监测</td></tr>
<tr><td>时间进度</td><td colspan="7">2018 年 2 月购买相关硬件设备，基本完成通信模块的硬件组装 2018 年 3 月-4 月基本完成数据处理模块的软件设计 2018 年 5 月完成用户交互界面模块的程序编写，将后台数据推送到网页渲染显示 2018 年 5 月底完成论文准备答辩。</td></tr>
</table>

系（教研室）主任 签字	2017 年 12 月 9 日	主管院长 签字	2017 年 12 月 9 日

西安邮电大学本科毕业设计（论文）开题报告

学号	06141107	姓名	邹智宏	导师	陈维佳
题目	基于单片机的蔬菜大棚温度监测系统的设计				

选题目的：

2009 年以来，物联网概念在国内甚至全球都成为热潮，物联网被称为继计算机、互联网之后世界信息产业第三次浪潮。而嵌入式设备在物联网的应用中占据极大的比例。

在网络应用日益普遍的今天，越来越多的嵌入式应用需要支持网络功能，比如，现在的许多智能设备需要实现远程的数据存储甚至远程控制，均离不开网络的支持。

由于市面上大多数存在的都是普通的检测报警器，并不能很方便的将信息传达给用户。为了解决这一问题，基于单片机的蔬菜大棚温度监测系统的设计将传感器采集的数据远程发送到服务器，务农人员可以实时打开网页进行监控多亩蔬菜大棚的温度和烟雾。高并发的 epoll 模型也为数据的安全传输提供了保障。

前期基础：

已学课程：《单片机原理及应用 A》《计算机控制技术》《C 语言程序设计》

掌握工具：示波器、电烙铁、电脑；

资料积累：《UNIX 环境高级编程》（（美）理查德·史蒂文斯 人民邮电出版社）

《深入理解 java 虚拟机》（周志明 机械工业出版社）

《effective java》（（美）布洛克 机械工业出版社）

《Andriod 应用程序开发》（王向辉 清华大学出版社）

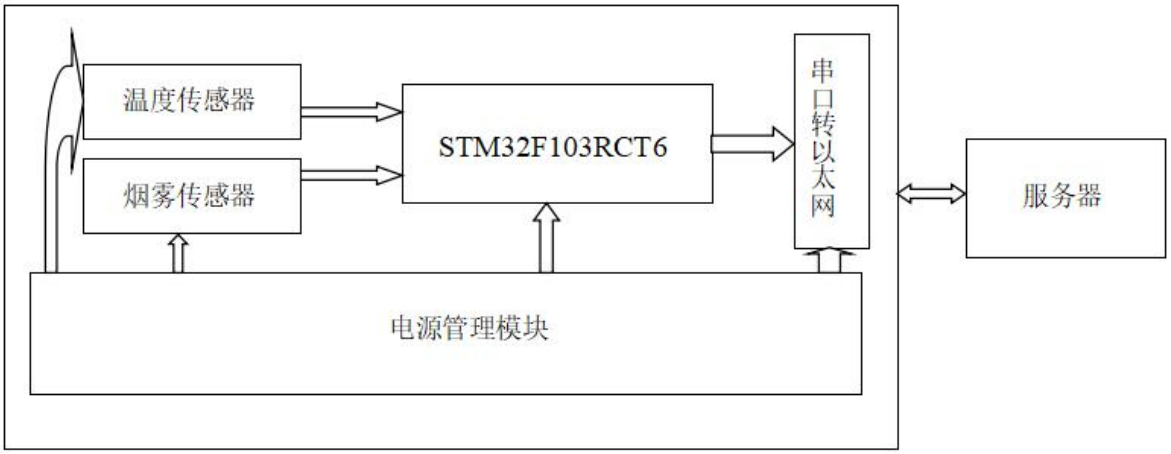
《MySQL 必知必会》（（英）福塔 人民邮电出版社）

要解决的问题：

通用性问题：我们采用 JSP 技术来进行动态 web 网页的开发。JSP 是用 Java 开发的，可以一处编写，到处运行，因此 JSP 与平台完全无关，可以支持多平台的移植。对于网络数据，则采用 XML 方式进行数据打包，由于 XML 简单灵活的文本格式，非常适合在网络上进行传输。

安全性问题：在 MySQL 中可用视图增强控制数据方式，增强了用户数据的安全性，同时用户和数据是分开的，访问方式设计上加强数据管理转变。

工作思路和方案：



系统功能架构设计图

数据收集模块通过硬件设备收集数据，将处理好的数据包发送到远端服务器，数据采集器和服务器之间遵循应用层的数据传输协议。数据打入后台数据库中，可以在 app 客户端进行显示。

第 1 周—第 2 周：　收集资料、理解题目、整理文献综述。

第 3 周—第 4 周：　题目分析、方案设计，分析各模块之间的数据交互关系。

第 5 周—第 7 周：　完成后台管理各模块的设计，并进行单元测试。

第 8 周—第 10 周：　完成硬件电路各模块设计，进行单元测试

第 11 周—第 12 周：进行模块联调，完成系统整体架构，完成整体测试。

第 13 周—第 14 周：撰写毕业论文。

指导教师意见

内容充实，重点突出，研究步骤基本合理，学生能够在预定时间内完成。同意该课题开题。

签字　　2018 年 1 月 9 日

西安邮电大学毕业设计 (论文)成绩评定表

<table>
<tr><td>学生姓名</td><td>邹智宏</td><td>性别</td><td>男</td><td>学号</td><td>06141107</td><td>专 业
班 级</td><td>自动 1403 班</td></tr>
<tr><td>课题名称</td><td colspan="7">基于单片机的蔬菜大棚温度监测系统的设计</td></tr>
<tr><td>指导教师意见</td><td colspan="7">评分（百分制）：_____ 指导教师(签字)：_____ _____年___月___日</td></tr>
<tr><td>评阅教师意见</td><td colspan="7">评分（百分制）：_____ 评阅教师(签字)：_____ _____年___月___日</td></tr>
<tr><td>验收小组意见</td><td colspan="7">评分（百分制）：_____验收教师(组长)(签字)：_____ _____年___月___日</td></tr>
<tr><td>答辩小组意见</td><td colspan="7">评分（百分制）：_____ 答辩小组组长(签字)：_____ _____年____月____日</td></tr>
<tr><td>评分比例</td><td colspan="7">指导教师评分(20%) 评阅教师评分(30%) 验收小组评分(30%) 答辩小组评分(20%)</td></tr>
<tr><td>学生总评成绩</td><td>百分制成绩</td><td colspan="2"></td><td>等级制成绩</td><td colspan="3"></td></tr>
<tr><td>答辩委员会意见</td><td colspan="7">毕业论文(设计)最终成绩(等级)：_____
学院答辩委员会主任(签字)：_____ _____年___月___日</td></tr>
</table>

目 录

摘　要

温室大棚在现代社会生产生活中应用广泛，如何利用自动检测与自动控制系统有效的控制好蔬菜大棚里的各种环境参数，以提高蔬菜大棚环境的控制精度和效果，对于我国温室业的发展有着不可估量的意义。所以针对这种情况我们很有必要设计出一种温度远程监控系统，减少火灾的发生，并应用到蔬菜大棚中提高产量。目前现有的蔬菜大棚温度监控系统，但是大部分的蔬菜大棚温度监测系统都难于统一管理。所以研制一款针对于家庭、蔬菜大棚等小环境研制出一款温度监测系统是非常重要的，让用户可以网页登录 app 实时监控。并且本次设计也简单实用。

该系统设计核心为单片机和 MQ-2 半导体气体烟雾传感器以及 DHT11 温度传感器，芯片我们采用 STM32F103RCT6 芯片，基于单片机的蔬菜大棚温度监测系统的设计中的电路包括传感器模块、单片机最小系统模块、串口转以太网模块、服务器模块等组成。最终经过调试，确定本设计方案可实现预设功能。

关键词：温度监测；单片机；数据采集；数据传输

ABSTRACT

Greenhouse are widely used in modern social production and life. How to use automatic control system of effective control of greenhouse trellis inside, in order to improve the environment factor trellis environment control precision of the greenhouse effect and has become the greenhouse industry research in China at present. In this case, it is necessary to design a remote monitoring system of temperature, and we can apply it into vegetable greenhouses to increase the production of vegetables. At present a lot of temperature monitoring system has been designed in our country, but most of them is difficult to manage conveniently. At this stage, it is very important to develop a temperature monitoring system for the small environment such as family and vegetable greenhouses, so that users can log in the web and monitor it in real time. This design is also concise and practical high performance.

The core system is designed by using single chip microcomputer and MQ - 2 semiconductor gas sensor and temperature sensor DHT11 smok , STM32F103RCT6 chip is adopted as well as, chip circuit including sensor module, temperature monitoring system for vegetable greenhouses of single chip microcomputer minimum system module, serial ports, Ethernet module, server module, etc. Finally, after debugging, it is feasible to determine the design scheme.

Keywords: Monitoring temperature; SCM; Data collection; Data transmission

第一章 引言

1.1 课题背景

这是一个科技的时代，科技的进步促使人们对于电器的需求也是与日俱增，家用电器、工厂使用的电器也层出不穷。也正是因为如此，这些电器的不当使用，或者操作人员的疏忽大意，很容易引起各种火灾，火灾的危害不仅仅是财产的流失更是对生命的一种威胁和损害。所以针对火灾发生的原因我们要引起重视，通过检测温度和烟雾浓度趁早发现火灾隐患，保护人民的安全。与此同时，在蔬菜大棚中对温度进行监测既能预防火灾，又能调节大棚中蔬菜的适宜温度，使得收成增加。在实际应用中，我们往往要对一些重要的设备进行监控，监控它们的各种状态，然后将检测的数据通过特定的装置进行处理和分析。很显然这是一种很笨的办法，对时间和人力或是财力都是一种浪费，往往还无法保证做到万无一失，得不偿失。

近些年来互联网的浪潮及至巅峰，软件的发展也推动着硬件的改革，物联网的大势所趋下，我们需要更加智能的方式去监控生活中可能带来隐患的场景，比如说通过网页进行远程监控。

国外的传感器技术早早发展起来，但是中国的技术在二十年后也迅速崛起，因为本身广泛的工业生产环境产生了一系列令外国赞叹的产品。1981 年后，随着 PC 终端的相对较多的使用。但是那个年代还是有其局促性，那个时候的中国互联网还没兴起，数据的传输快速优势还没得到广泛的应用，传感器的数据采集工作简单，实时性差，效率较低。随着互联网的崛起和大数据时代的来临，我们将数据的远程传输提上议程，对传感器的工艺也提出了更高的要求，这样能很好的为火灾信息监控和预防起到重要的作用。

本系统通过硬件获取数据，通过网线发送数据，通过网页监控数据，用户无论在何时何地，只要通过注册登入网站即可实现一个可实时监控、蔬菜大棚信息自动识别、报警等功能的基于单片机的蔬菜大棚温度监测系统。

本课题主要针对节点数据的结构体构造和数据采集与传输进行研究。如图 1.1 为该系统的软件系统架构图。

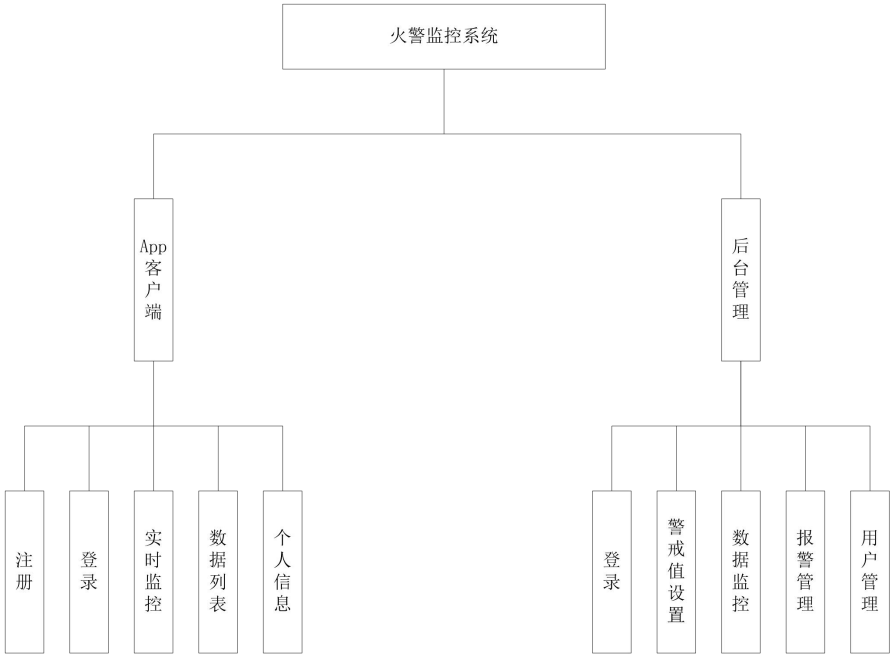


图 1.1　软件系统架构设计图

本课题系统设计理论为：通过传感器收集环境烟雾和温度的数据，将处理完成的数据通过网络发送至互联网中，用户通过登入网站获取实时信息，以达到实时监控的目的。

1.2 国内外发展现状

上世纪 30 年代初，国外就已经开始了对于烟雾传感器的研究和开发。那时的传感器市场已经在不断的发展，工艺也不断在提高，人们对于传感器又不断提出新的要求，此外，火灾频发也让人们感受到了威胁，政府和人民越来越重视这个事情，这也促使了烟雾传感器的飞速发展。这种发展体现在传感器体积趋于小型化，这也为我们的烟雾传感器和温度传感器在实际应用场景中的使用带来了便利，运输的成本较之以往也大大降低。

四十年之后，到上世纪七八十年代初，我国才开始逐步研究烟雾传感器，但是凭着科研人员的不懈奋斗，我们研究出多种多样的型号，针对不同烟雾的种类也在逐渐补全。我国的科研人员通过引进国外先进的烟雾报警仪器和技术来弥补我国在此项技术上的短板，通过对先进技术的学习和我国科研人员决心要自主研发的决心，我国最终还是研制出可以适应各种应急场合的烟雾报警装置。

一般可将我国的烟雾报警装置分为三类：民用烟雾报警器、工业用烟雾报警器和 有毒有害烟雾报警器。

（1）民用烟雾报警器

这种类型的烟雾报警装置不但会在烟雾浓度达到预设的报警限度时发出警报，

还会在空气中产生一种负离子用来和空气中的CO，反应后生成大气中最常见的CO_2。但是，这种民用的装置必须且只能安装在极有可能发生燃气泄漏的房间里，而且作用范围仅限于几十平方米，装置的灵敏度较低，一般不会检测出其他房间的泄漏。

(2) 工业用烟雾报警器

顾名思义，这种类型的烟雾报警装置只能检测在工业过程中生产的烟雾并在烟雾浓度达到预设的报警限度时发出警报。工业生产过程和内容十分复杂，所以在具体不同的工业生产中对装置的选择也是有区别的。燃气管道发生泄露时，检漏仪检测出来就会发出警报，显示烟雾浓度。在防爆现场人们一般采用探测器和控制器相结合的方式，两者之间用屏蔽电缆线连接，探测器放在危险的地方，而控制装置则是放在远离危险地区的值班室内，当煤气泄漏后由探测器给控制器发信号，控制器就会发出警报[9]。

(3) 有毒有害烟雾报警器

有毒有害烟雾报警器则是用来检测有毒有害烟雾，针对不同烟雾的毒性进行区分和探测。

1.3 论文结构

本文各章节安排如下：

第一章：引言；

第二章：介绍 STM32 主控芯片及烟雾温度传感器，讲述其技术特点及系统原理；

第三章：硬件电路的设计以及便宜工具的介绍；

第四章：软件编程及网页搭建；

第五章：调试过程和测试。

第二章 系统原理

2.1STM32F103RCT6 介绍

STM32F103RCT6 单片机是一种嵌入式-微控制器的集成电路，它拥有 48KB SRAM，该芯片性价比极高。MSP430 内部资源结构图如图 2.1 所示。

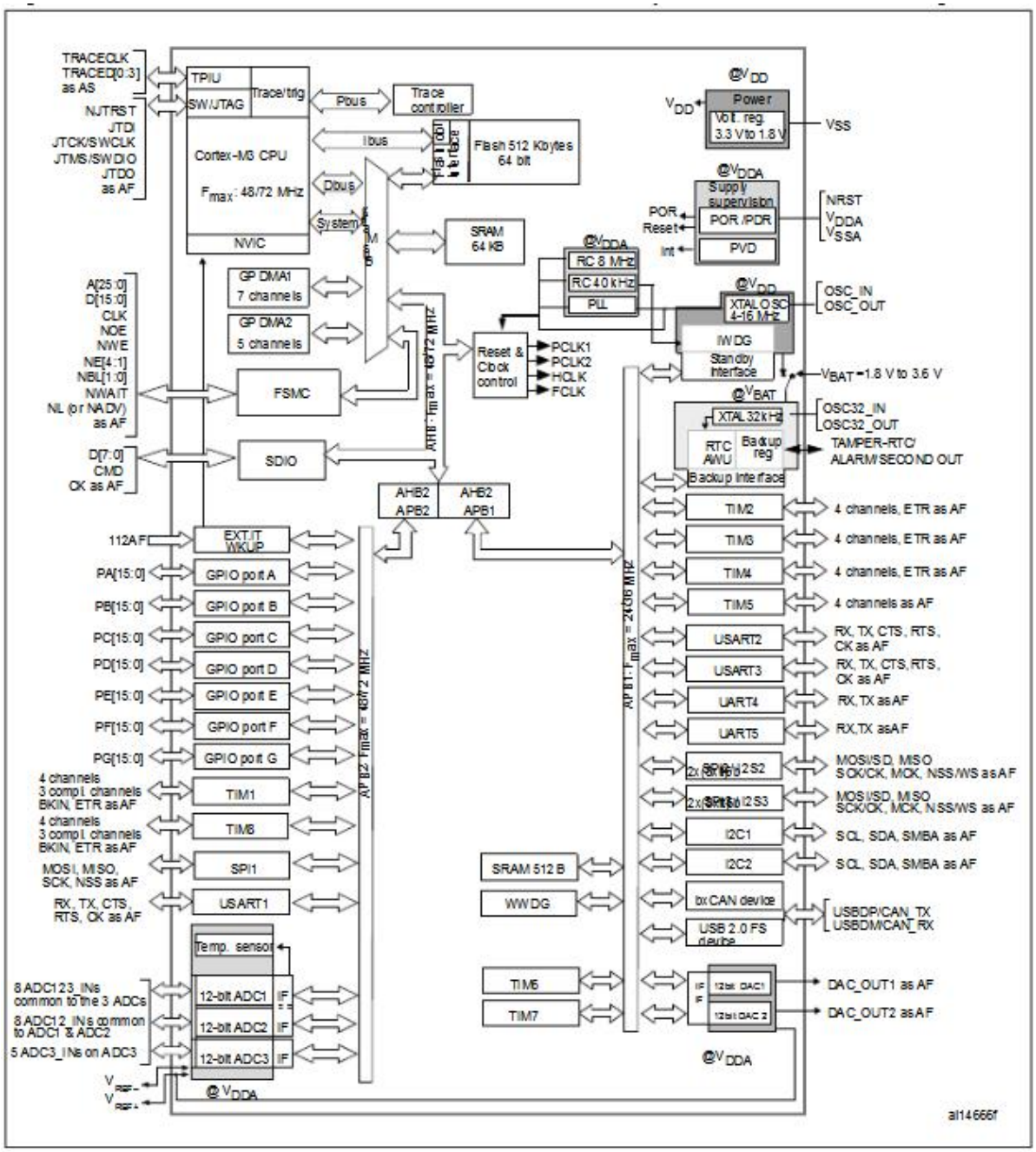


图 2.1　资源结构图

基本参数如下表 2.1。

表 2.1 基本参数表

系列	STM32
芯体尺寸	32 位
速度	72MHz
连通性	CAN,I2C,IrDA,LIN,SPI,UART/USART,USB
外围设备	DMA,电机控制 PWM,PDR,POR,PVD,PWM
输入/输出数	51
程序存储器容量	256KB
程序存储器类型	FLASH
RAM 容量	48K
电压-电源（Vcc/Vdd）	2V-3.6V
振荡器型	内部
工作温度	-40℃-85℃
封装/外壳	64-LQFP
包装	托盘

STM32 采用薄塑封四角扁平封装方式 LQFP，系统频率很高，完全可以满足我们在实验和设计过程中的开发需求。STM32F103RCT6 芯片封装图如下图 2.2。

2.2 嵌入式实时操作系统(FreeRTOS)介绍

实时操作系统具有实时性，要求及时性，这些可以选择合适的页面调度算法来进行系统响应。通过这样的方式来完成实时任务是可靠性较强的，效率有所增加并且将 cpu 利用率最大化[10]。

FreeRTOS 是轻量级的嵌入式实时操作系统内核，他核心的多线程调度算法十分高效，大大降低了开发难度，以及其具有四种内存管理方式，高效的利用了控制器的片内外存储介质。

2.2.1FreeRTOS 移植优势

FreeRTOS 的移植是 FreeRTOS 使用的关键，关于 FreeRTOS 的移植主要由三个文件实现，分别是.h 文件，.c 文件和.s 文件。高效的封装了底层 API，移植成功后只需调用上层 API，极大的降低了开发难度。除此之外，由于 FreeRTOS 的广泛使用，有关于这方面的资料也是非常详尽的。

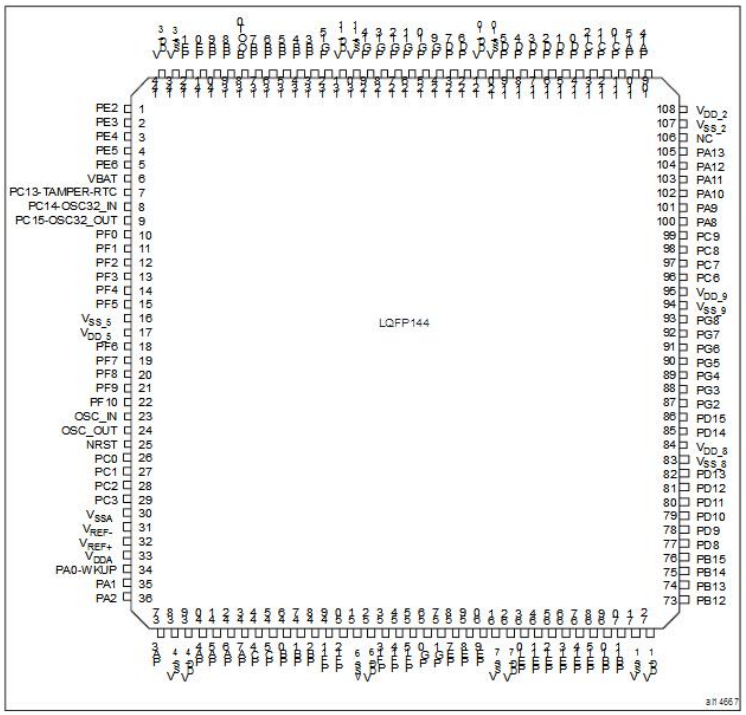


图 2.2　STM32F103RCT6 芯片封装图

2.2.2FreeRTOS 功能和特点

FreeRTOS 功能和特点如下表 2.2。

表 2.2　FreeRTOS 功能特点表

功能	特点
用户可配置内核功能	通过配置 FreeRTOS 的配置文件可以进行功能的定制
目标代码小，简单易用	FreeRTOS 的内核只有三个文件，可以节省开发时间
遵循 MISRA-C 标准的编程规范	符合一定的编码标准，也降低了对源代码阅读的难度
没有限制的任务数量与任务优先级	用户可以自己创建任务及设置任务的优先级，这大大增大了程序的灵活性
各任务之间数据传递方式	用户创建的任务之间可以通过队列进行数据交互，信号量等可以完成任务间的同步
免费的开源源代码	FreeRTOS 嵌入式实时操作系统，不仅可以供个人阅读学习源代码（方便定位问题），并且还可以免费用于商业用途（极大的降低了成本），相比于其他的实时操作系统是一个优势

2.2.3FreeRTOS 移植过程

FreeRTOS 的移植首先需要从官网上下载 FreeRTOS 的库文件，与移植相关的代码文件主要有 port.c,portasm.s,portmacro.h 三个部分，将文件移植妥当后，对 FreeRTOS 进行配置，主要配置文件为 FreeRTOSConfig.h，移植完成后就可以直接通过 API 进行任务创建，调度任务实现各个模块之间的交互。

FreeRTOS 的接口分层如图 2.3 所示。

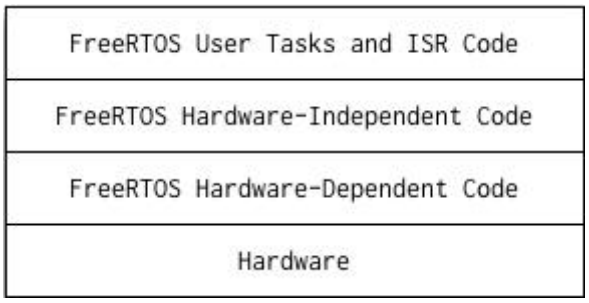


图 2.3　FreeRTOS 的软件层

2.3 嵌入式 TCP/IP 协议栈(Lwip)

2.3.1 简介

Lwip 是轻量型的 TCP/IP 协议栈，他在保持主要功能的基础上减少对 RAM 的占用，他只需十几 KB 的 RAM 就能够运行，并且 Lwip 尽量避免内存复制，避免了此操作产生的性能损失[5]。

2.3.2 网络协议分层

网络开发过程中，了解网络分层的概念能帮助我们更好的理解软件的设计。数据链路层的数据流通是对于用户不可见的，用户能看到的就只有应用层面的操作。我们在网页设计时，也不需要将复杂的后台功能的实现展现给用户，我们只需要将最为关键的数据信息传递给用户就可以了。网络分层如下图 2.3 所示。

2.3.3TCP 报文格式

IP 数据包封装了 TCP 报文，在网络层进行分组转发，如图 2.4 所示。

表 2.3　网络分层表

链路层（设备驱动及接收卡）	包括设备驱动程序及网络接口卡
网络层（IP、ICMP 和 IGMP）	位于第二层，若数据包超过了系统规定的最大值，则需要将数据包进行拆分，拆分后分别发送每一部分。在 TCP/IP 协议族中，第二层包括 IP、ICMP、IGMP。
运输层（TCP 和 UDP）	运输层为同一主机上的两个不同进程之间进行网络通信或者在不同的物理主机上的两个进程进行网络通信。IP 层通过将数据报分片（一般将超过最大传输单元 MTU 的数据包进行分解，再加上一些信息，成为小的报文分别进行发送）。
应用层（Http 和 FTP 等）	应用层一般为用户实现，用户可以在该协议层获取期望的数据。

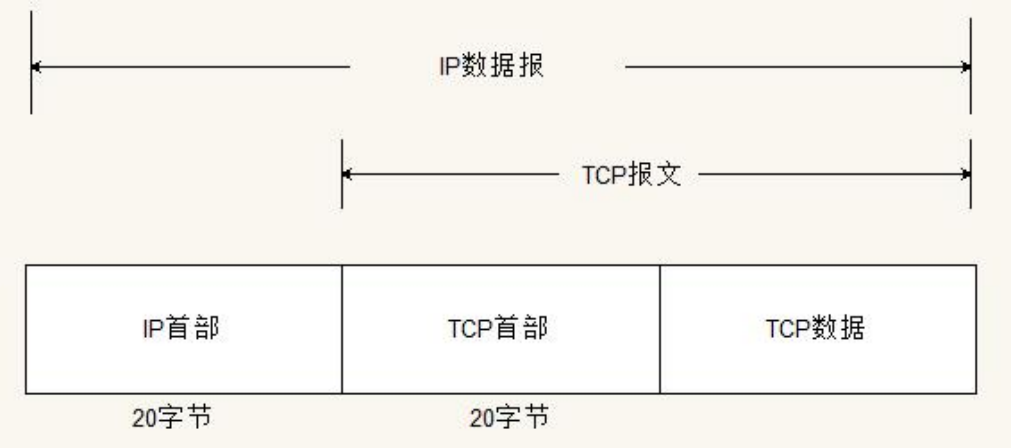


图 2.4　IP 数据报中的内部封装

TCP 数据包格式如图 2.5 所示。

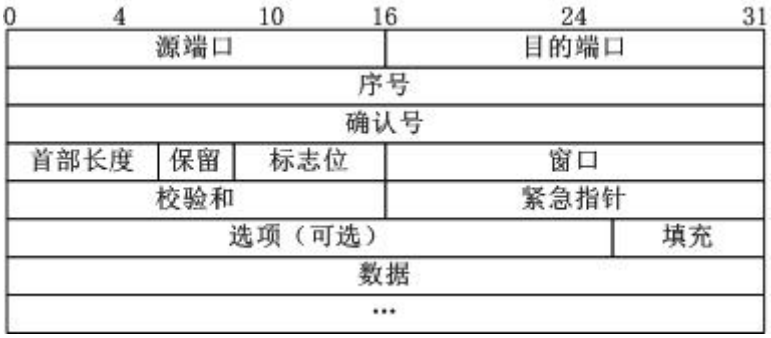


图 2.5　TCP 报文格式

- 序号：其实就是包的序号，为了保证传输的数据包不会发生丢失和乱序。
- 确认序号：确认收到数据包并请求下一个序号数据包返回给发送方的一个信号。
- 标志位：
  - URG：紧急指针
  - ACK：确认标志
  - PSH：快速转发
  - RST：重置连接
  - SYN：发起一个新连接
  - FIN：释放一个连接

2.3.4Lwip 模式

Lwip 提供以下编程接口：

RAW 接口：应用和协议栈共同发挥作用，有数据来临时，应用程序就向 Lwip 协议栈中添加一个函数，只有当事件发生变化或者文件描述符状态发生改变，该接口被调用时，实际上是调用指针指向的方法。内核的速度是最快的，用这个接口就是将小型程序移植到内核空间运行，速度也会变快。

Lwip 接口：该接口将 receive 与 send 函数可能会造成资源抢占和死锁，会导致响应较慢或者阻塞的情况，而该接口可以让接受的数据存放在一个流动窗口，程序从中获取。

BSD 套接字接口：这是 UNIX 系统中常见的用于网络通信的接口，允许不同主机进行通信，而管道需要血缘关系并且只能单方面发送信息，效率是所有通信方式中最快的[6]。

2.3.5Lwip 特性

- 在网络层，可以支持不同接口的数据包转发。
- RAW API 接口可以极大加快程序的运行，并且维持在一个稳定安全的状态。

- 可以完成正常的网络通信
- BSD 套接字接口

第三章 系统硬件设计

该系统包括 STM32F103RCT6 微控制器、温度传感器、烟雾传感器、串口转以太网模块、网络模块。火警智能警报系统设计框图 3.1.

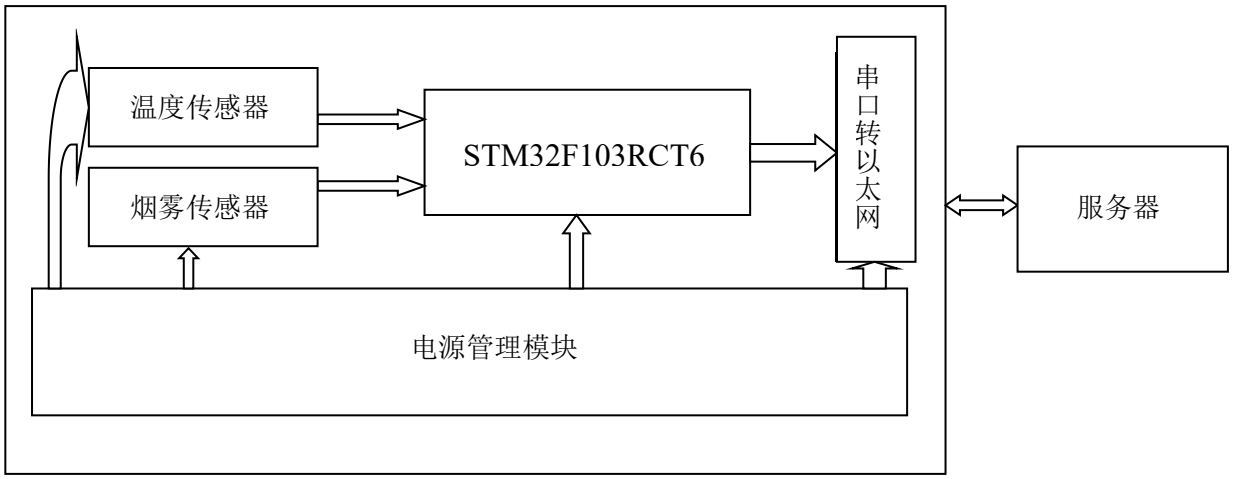


图 3.1　火警智能警报系统设计框图

该系统架构中包括传感器模块、STM32F103RCT6 主控模块、电源管理模块及转换模块。传感器采集数据发送给 STM32 控制器，然后发送到串口，串口将数据进行打包通过互联网上传给服务器，服务器再用数据处理函数对发送过来的数据进行处理。对于发送的数据，STM32F103RCT6 通过中断等待我们目的数据的返回[4]。

3.1STM32F103RCT6 最小系统电路设计

该模块主要包括了电源电路设计、时钟电路设计、下载电路设计、复位电路设计、按键电路设计以及接口电路设计。STM32 引脚连接图如图 3.2。

各个模块的具体介绍如下：

3.1.1 电源电路设计

电源电路保证的整个系统供电，对整个系统的安全起着决定性作用。电路图如图 3.3 所示。

OLED 模块接口图如图 3.4 所示，本接口支持 0.49-1.3 寸所有 OLED 模块，使用时 GND、VCC 一定要对应上，不可以插反。

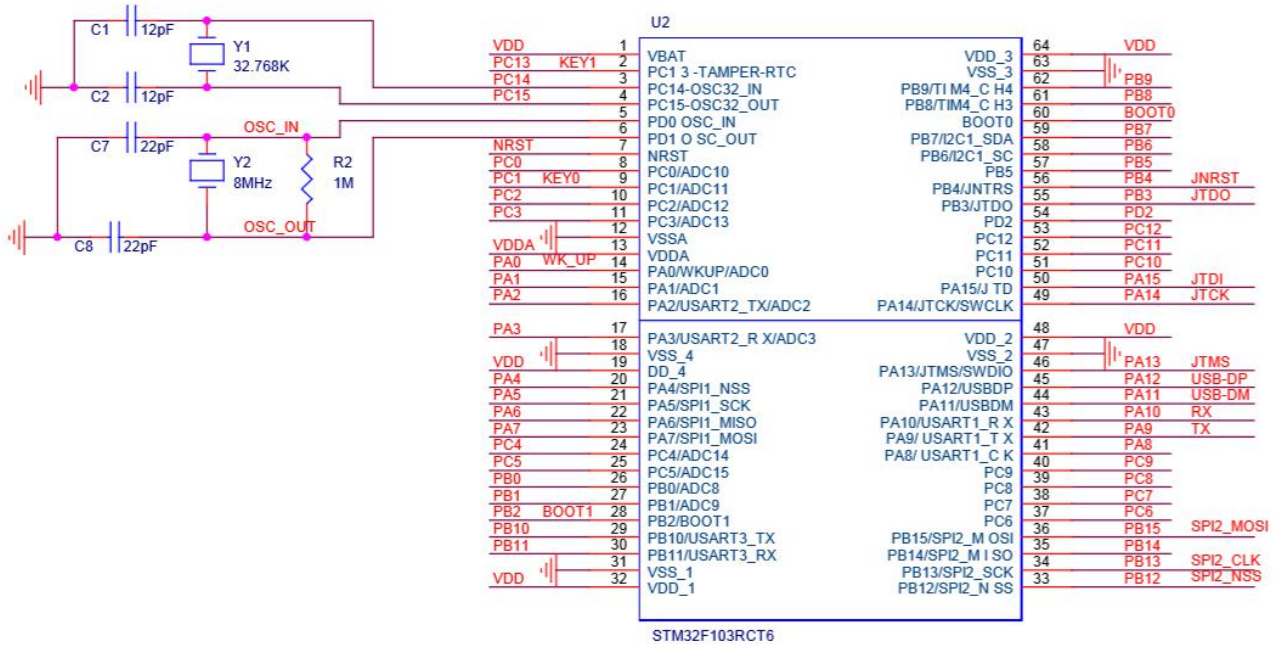


图 3.2　STM32 引脚连接图

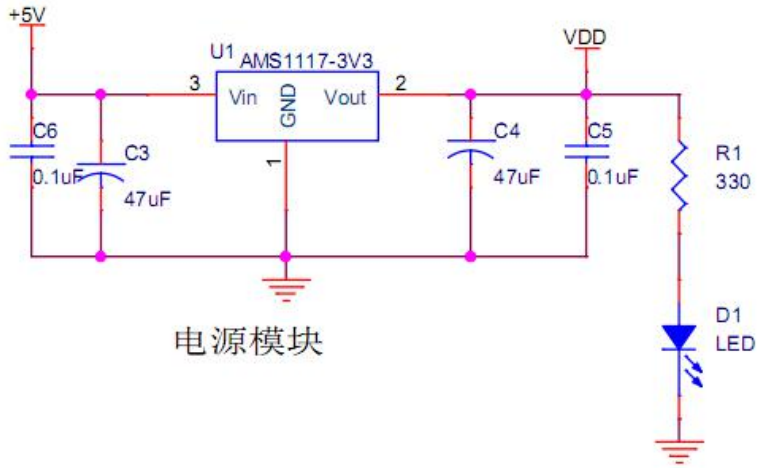


图 3.3　电源电路

3.1.2 时钟电路设计

RTC 时钟电路如图 3.5 所示，系统时钟电路如图 3.6 所示。

3.1.3 下载电路设计

STM32 使用 JTAG 接口的下载电路设计如图 3.7 所示。

STM32 使用 ISP 串口下载电路设计如图 3.8 所示。

图 3.4　OLED 模块接口图

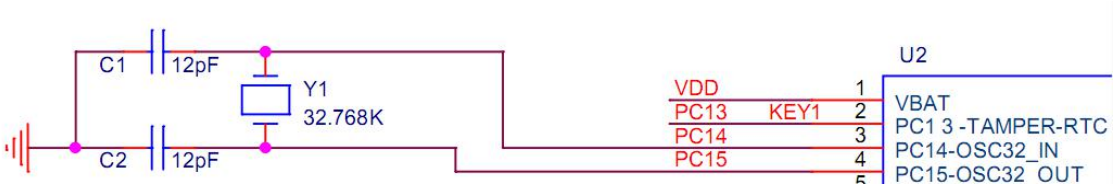


图 3.5 RTC 时钟

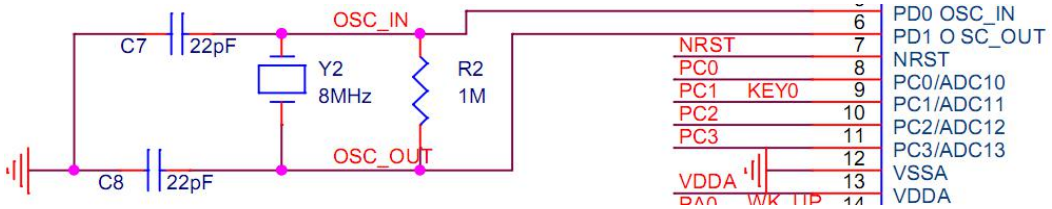


图 3.6 系统时钟

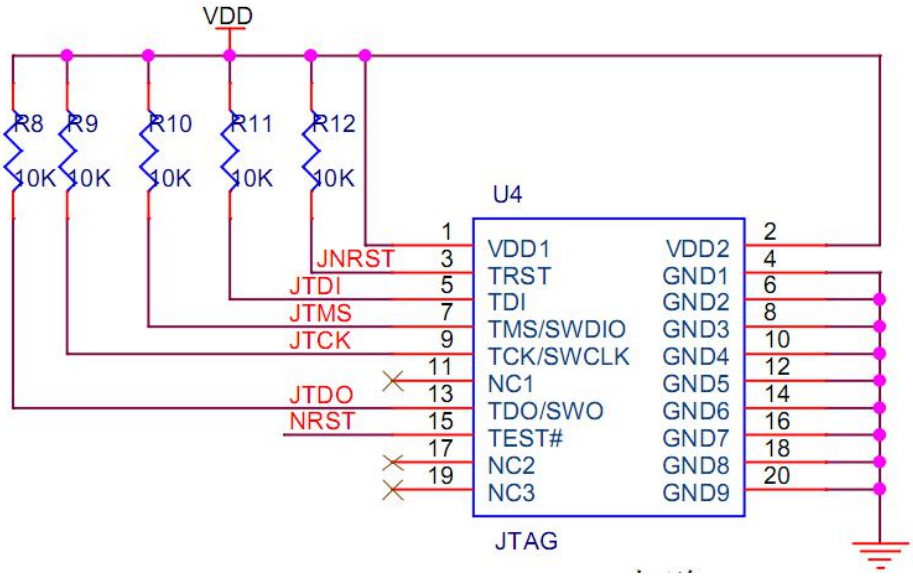


图 3.7　下载电路设计

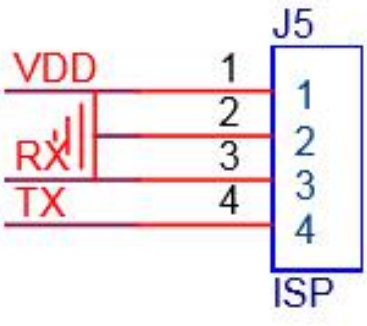


图 3.8　ISP 串口下载电路

3.1.4 复位电路设计

为了保证电路稳定可靠工作，必须利用复位电路将电路调节到初始状态。如图 3.9 为复位电路设计图。

按下复位键后程序会重新开始执行，复位引脚为低电平，松开后电容开始充电。

3.2 串口转以太网模块

该系统采用串口转以太网 USR-TCP232-T2 模块。我的设计需要将传感器采集的数据通过互联网传送出去，而该模块可以很好的将 TCP/UDP 数据包与 RS232 接口数据进行传输，并且该模块体积小方便运输，低功耗方便使用，稳定性好，速度快，搭载 ARM 处理器。图 3.10 为该芯片实物图。

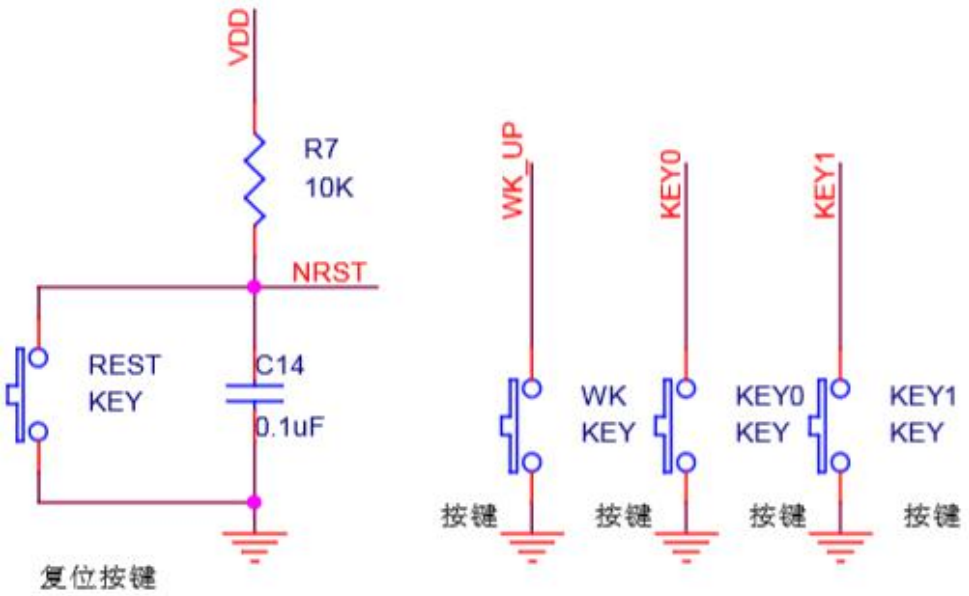


图 3.9　复位电路设计

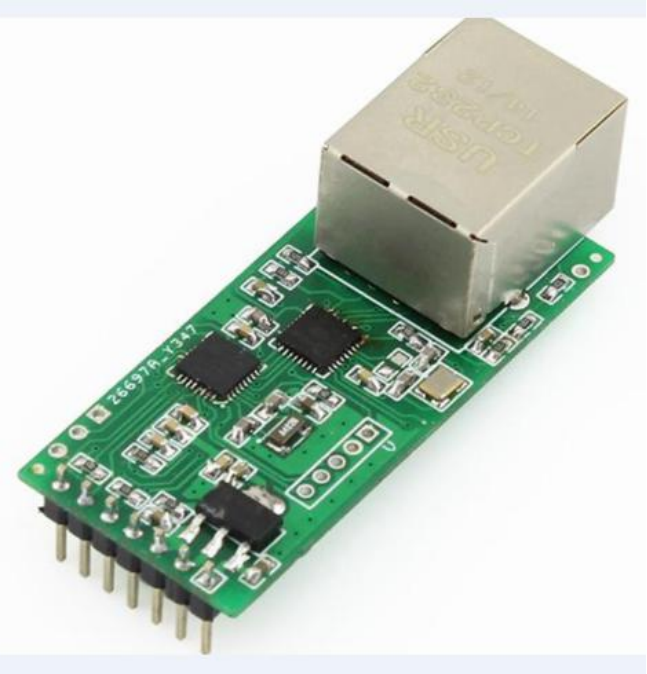


图 3.10　USR-TCP232-T2 模块实物图

该芯片的主要特点：

- 10/100M 自适应以太网接口
- 波特率 300 到 234000 可设置
- 支持虚拟串口工作模式
- 工作端口，目标 IP 地址和端口均可轻松设置
- -25℃至+75℃扩展级工业温度范围

USR-TCP232-T2 模块引脚图如图 3.11 所示。引脚 1 和引脚 2 为供电管脚，分别是 5V 和 3.3V，3 脚接地，4 为复位脚，567 分别为发送数据管脚、接收数据管脚以及配置引脚。

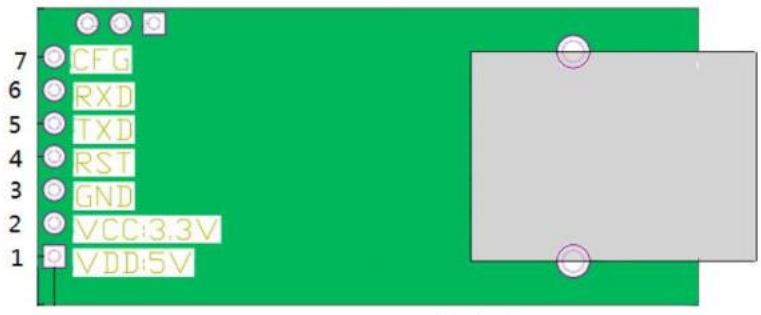


图 3.11　MAX485 芯片引脚

典型应用硬件连接如图 3.12。

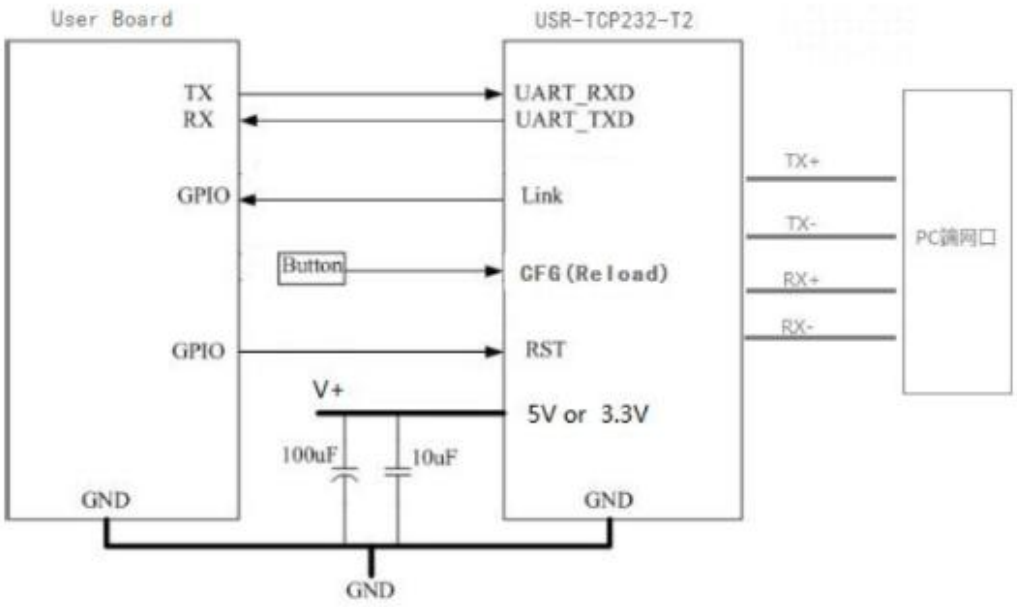


图 3.12 转换模块电路

3.3 传感器模块

本系统主要使用了两种传感器，温度传感器和烟雾传感器。

3.3.1 温度传感器

温度传感器本设计选用 DHT11。DHT11 是一款性价比较高的可以测量温湿度的传感器，它还含有数字信号输出。采用简单的单总线即可完成通信和交换数据[1]。该传感器的湿度误差在 5%RH，温度误差在 2 摄氏度，湿度能力在 20%-90%RH，温度能力在 0-50 摄氏度。产品为 4 针单排引脚封装，连接方便。而本系统选用的温度传感器模块实物图如 3.13 所示。它的 1 脚接 VCC，3.3V~5V，2 脚接地，3 脚为数据口。

3.3.2 烟雾传感器

本设计选用的烟雾传感器是 MQ-2 型烟雾传感器，这是二氧化锡半导体气敏材料。模拟量输出 0-5v 电压，气体浓度越高，模拟量输出电压也越高[2]。MQ-2 烟雾传感器模块的实物图如 3.14 所示。它有四个脚，1 脚接 5V 电源，2 脚接地，3 脚为 TTL 开关信号的输出，4 脚为模拟信号的输出[3]。

3.4 硬件设计环境

Altium Dedigner 又被叫做 Protel，常用来做电路设计。我们经常用它来设计原理图、做仿真电路、绘制 PCB 等工作。界面如图 3.15 所示。

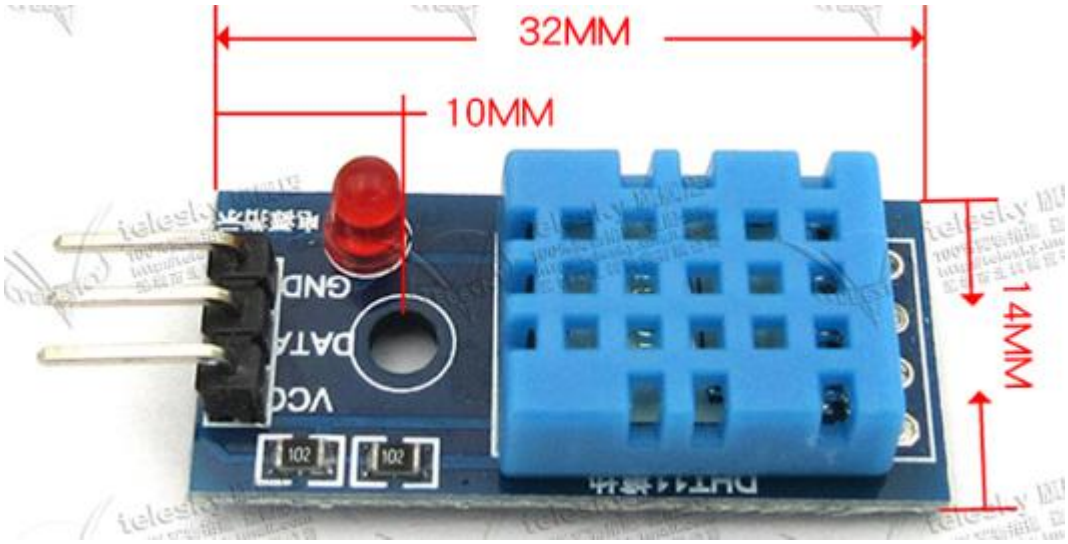


图 3.13 温度传感器模块实物

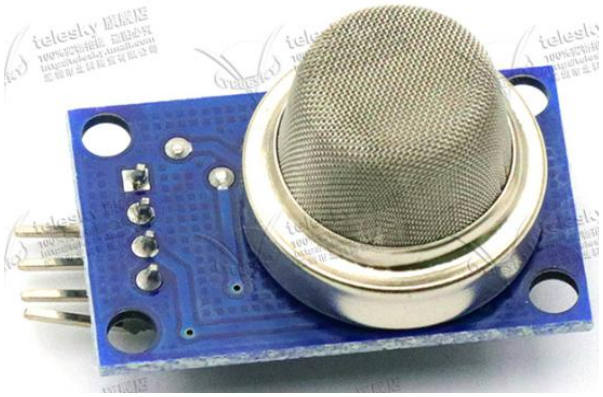


图 3.14 烟雾传感器模块实物

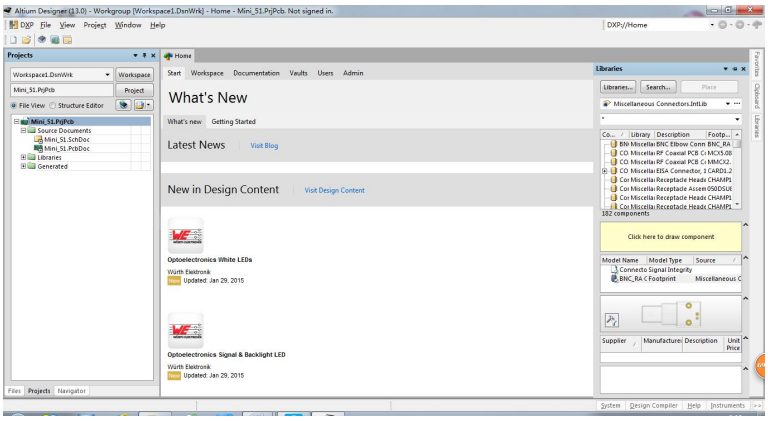


图 3.15 Altium Designer 环境界面

3.5 硬件设计小结

系统设计的硬件有 STM32F103RCT6、传感器模块、串口转以太网模块、电源模块。通过阅读官方文档和用户手册，开始逐步了解芯片和各引脚的作用，STM32 主控模块主要用于定时获取传感器的数据，并将数据进行封装转发到串口转以太网模块，并上传至服务器。传感器模块将获取的温度湿度数据以及烟雾数据通过串口发送给主控模块。串口转以太网模块是将主控模块封装好的数据发送到写好的服务器上。电源模块是给各模块进行供电。

硬件设计的过程并不是一帆风顺的，万事开头难，由于太过于在乎框架的搭建与设计，所以前期的时间和精力一直耗费在这上面，耽误了时间，影响了我后期的任务的进行进度。

第四章　系统软件设计

4.1FreeRTOS 系统设计

本系统的结构设计分为了多个部分，一是 FreeRTOS 系统，二是网络程序，三是数据处理，最后还有一个服务器设计。网页的构造需要从整体到局部细节一层层去架构。总体的设计流程图如 4.1 所示。

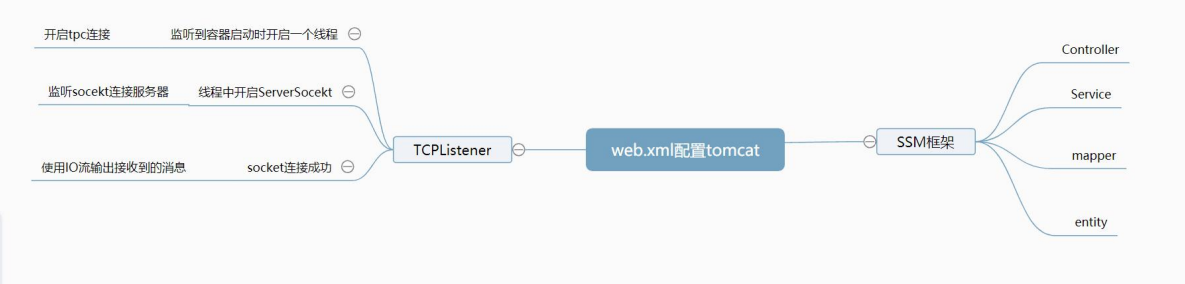


图 4.1　流程图

4.1.1FreeRTOS 程序结构

MDK 编译器中添加文件如图 4.2 所示。

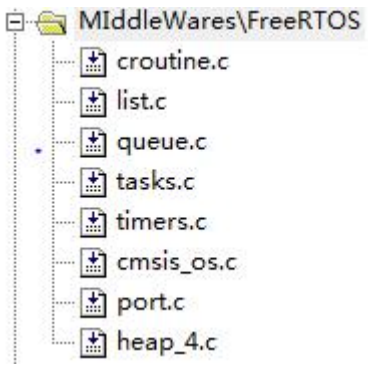


图 4.2 FreeRTOS 工程文件

由于 FreeRTOS 的内核体积并不大，大部分的任务操作都可以通过下面几个文件来完成：

- task.c：负责任务的管理和调度等。
- list.c：存储任务的链表。
- tasks.c：任务的消息队列文件。
- croutine.c：进程管理的文件。

4.1.2 FreeRTOS 任务管理

在 FreeRTOS 这种小型的嵌入式系统之中，它不同于传统的 Linux 操作系统和常见的 Windows 操作系统，FreeRTOS 由于只有一个 CPU，所以没有办法在 CPU 之间

切换任务和进程。FreeRTOS 切换任务实质上是在单个进程中对可执行程序进行换入换出磁盘和内存的操作，把一段可执行程序的代码加载到内存，然后运行它，就成了 FreeRTOS 任务的运行状态。把该程序换出内存，该任务程序就处于非运行态。

本系统切换 FreeRTOS 的任务使用的是 FreeRTOS 提供的一些用户态回调的接口，用户将自己实现的一些业务逻辑的回调函数，通过系统调用注册在 FreeRTOS 的内核中。每个任务的小程序是由 FreeRTOS 的任务的实现的，并且这些小程序不能退出，运行在一个死循环之中。在这里值得注意的是，该操作系统不允许用任何方法从函数中退出，另外操作系统可以调用一些函数，用来删除各种已经注册的任务。

创建 FreeRTOS 任务的方式有两种，一种是通过调用普通的函数在普通的函数中创建一个任务，另外的一种是在创建的任务之中直接调用一次创建任务的函数即可，这些都是 FreeRTOS 创建任务的方式。

如图 4.3 所示，任务在运行状态和非运行状态之间切换。

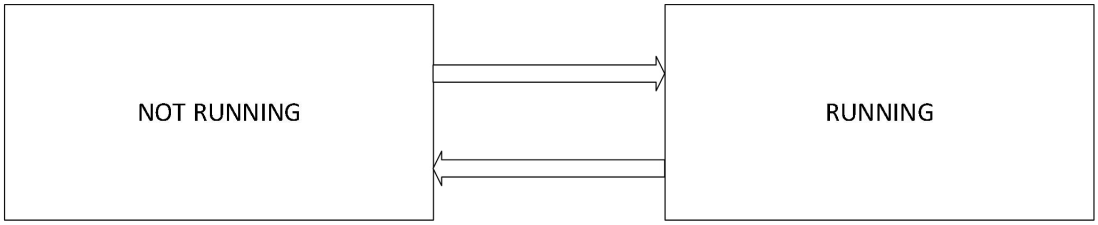


图 4.3　任务运行状态

在多任务进程当中，正在运行的任务处于运行状态，其它的就处于非运行状态，线程状态机如图 4.4 所示。

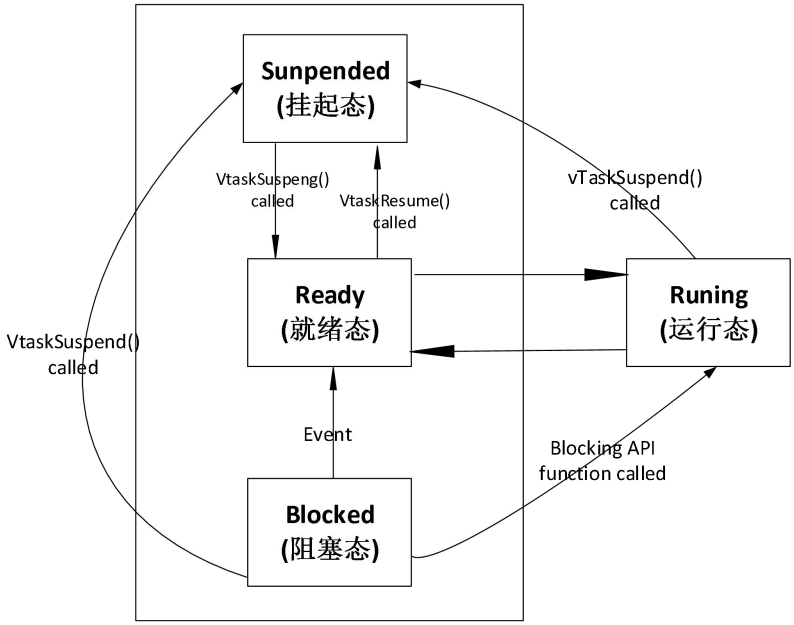


图 4.4　FreeRTOS 任务状态机

FreeRTOS 状态机介绍如下：

- 运行态：正在运行中的任务所处状态。
- 阻塞态：当一个任务等待某个事件发生时的状态。
- 挂起态：不需要竞争资源的状态。
- 就绪态：任务所需条件已经准备好只欠缺 CPU 资源，这时候就叫就绪态。

4.1.3FreeRTOS 任务间通信

本系统使用了 FreeRTOS 的消息队列来实现进程间通信。

任务间通过队列传输数据如图 4.5 所示。

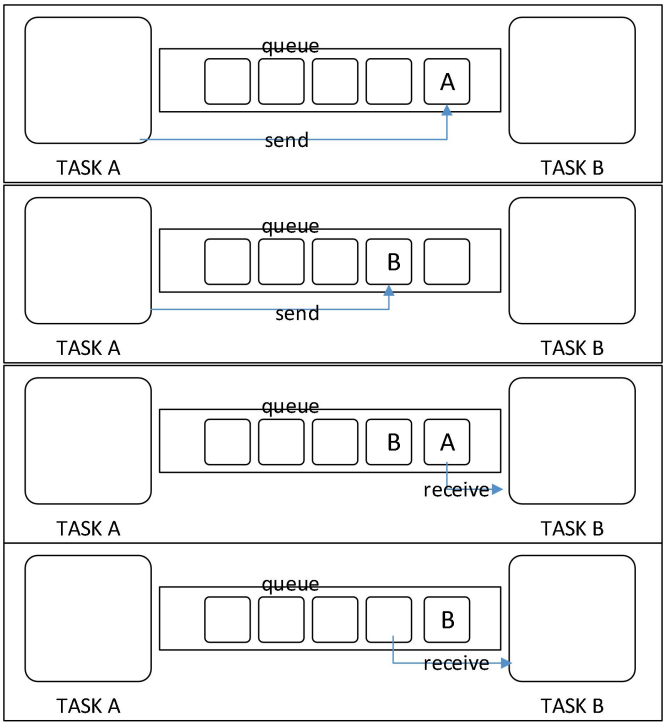


图 4.5　队列读写过程

4.1.4FreeRTOS 模块程序设计

程序框图设计如图 4.6 所示。

本系统从上到下进行任务的创建，采集，存储，以及执行一些网络任务等。本系统通过一些调用回调，一是进行任务的创建，二是实现任务的具体细节，比如任务需要完成的一些功能。

4.2 网络模块软件设计

4.2.1BSD socket 接口

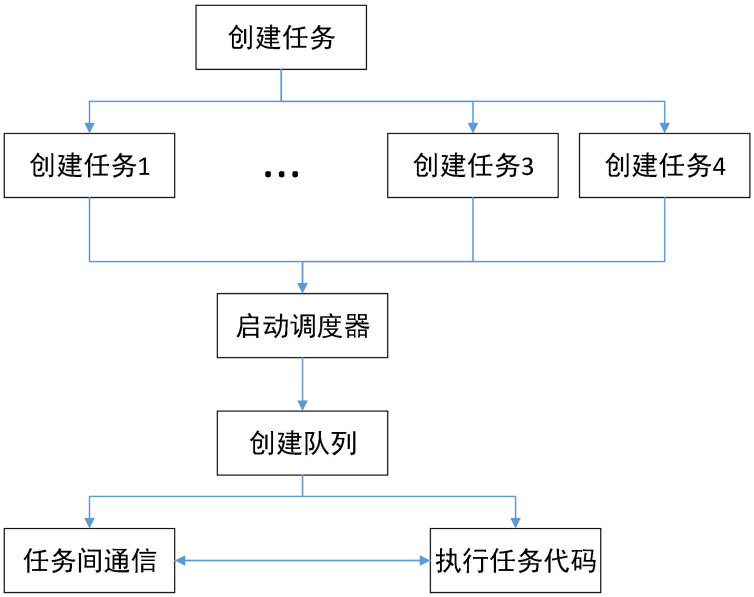


图 4.6　FreeRTOS 模块流程图

本项目的软件模块设计中有一个重点，就是本项目的网络模块。这里使用 BSD 的 socket 接口实际上是起初为 Linux 操作系统开发的一些网络接口，后来由于 Linux 操作系统发展的越来越强大，它的 socket 接口被众多后继出现的操作系统包括嵌入式操作系统所仿效，现在这些 socket 接口已经成为了一种标准接口，所以使用该接口开发

Socket 套接字实际上在操作系统包括 FreeRTOS 之中都是一个文件描述符，这符合 Linux 操作系统一切皆文件的哲理。在一个操作系统中，为 socket 分配的文件描述付一般情况下都是递增的，每一个 socket 分配不同的文件描述符，对这个文件描述符进行读和写就可以进行与客户端的通信工作。

套接字流程如图 4.7 所示。

4.2.2Lwip 模块软件设计

整个网络模块的任务就是连接服务器，通过与服务器进行交互，将采集到的信息上传到服务器。

程序的设计思想流程图如图 4.8 所示。

这里解释一下上面的流程图，首先进行一次 socket 的初始化，这里在内核中为 socket 分配一些内核资源，比如文件描述符。这个 socket 之后就相当于对应上了一个文件描述符。然后开始进行发起连接，这里使用的 connet 函数是发起一个 TCP 连接，它所需要的参数是对端的地址和 IP 等信息。在连接成功之后，可以接受另外一端发送的一个序列码，这是一个随意的值。客户端然后对这个随机的值进行摘要，比如使用 MD5 算法进行摘要。MD5 算法是一个摘要算法，可以将输入转化为定长的字符串，这个字符串相当于输入的摘要。MD5 是一个不可逆的算法，只能进

行摘要编码，并不能进行反编码解析工作。在对端，可以读取一下回包的数据，并且比较一下 MD5 值，如果这个值是一样的，那么视为请求合法，登陆成功。否则视为登陆失败，上面的一系列操作必须重新执行一遍。

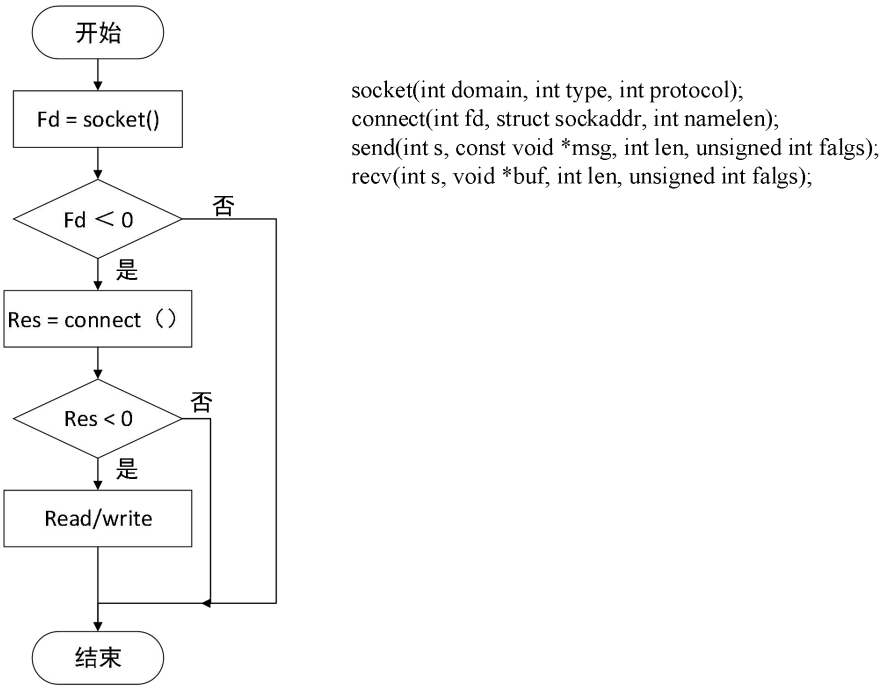


图 4.7　socket API 及设计流程

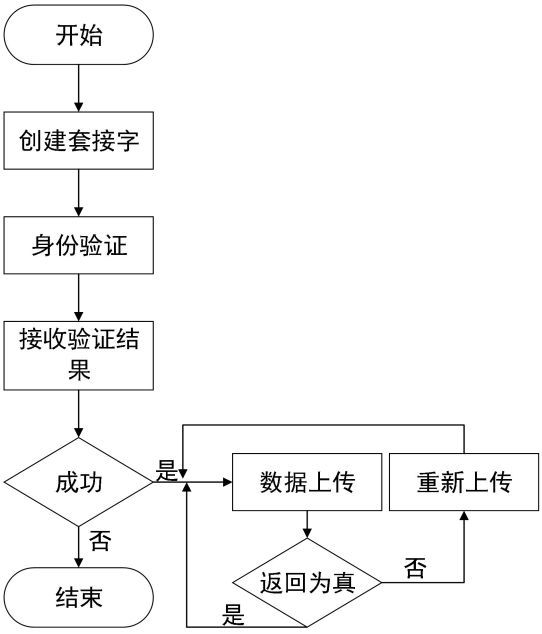


图 4.8　程序设计流程

4.3 数据处理软件设计

4.3.1XML 格式

本系统的另外一个重要的模块就是数据处理模块，数据处理模块主要是处理采集的数据的，本系统采用 xml 格式处理这些采集到的数据。Xml 格式虽然现在已经被 json 逐步替代，但是 xml 格式仍然有自身的优势比如定义结构功能更强大，仍旧有着有广泛的应用。使用 xml 格式的目的是将一些参数配置文件化，这样用户都可以方便的修改这些配置文件，达到修改软件的功能的目的。使用 xml 更加灵活，因为 xml 自身有着强大的解析库，开发时也可以轻松处理 xml 解析的工作。

4.3.2 数据处理

因为在网口传输过程中，读出的数据还需要进一步做处理，这些处理方式也根据要测量的电压、电流、功率的不同而需要进行不一样的计算，才能得到我们最终的所求。具体的处理方式如下：

(1)读出的电压为二次侧的电压值，固定 1 位小数位，二次侧的电压值=读出值/10，一次侧的电压值=读出值×PT 变比/10。

(2)读出的电流为二次侧的电流值，固定 3 位小数位，二次侧的电流值=读出值/1000，一次侧的电流值=读出值×CT 变比/1000。

(3)读出的功率为二次侧的功率值，固定 1 位小数位，二次侧的功率值=读出值/10，一次侧的功率值=读出值×PT 变比×CT 变比/10。

(4)频率固定 2 位小数位，频率值=读出值/100。

(5)功率因数固定 3 位小数位，功率因数值=读出值/1000。

(6)电能值由 3 个寄存器(Word0、Word1、Word2)组成，前 2 个寄存器组成一个长整数，表示整数部分的值，后 1 个寄存器组成一个整数，表示小数部分的值，为 3 位的小数。电能值=Word0×65536 + Word1 + word2/1000。

对于这些处理方式，需要在程序中实现。

4.3.3XML 打包

本系统最终的目的是将数据上传到服务器端，这就需要对 xml 进行打包，并利用网络模块进行 xml 文件的发送，具体流程如图 4.9 所示。

4.4 服务器软件设计

4.4.1epoll 模型

Epoll 是 Linux 的 I/O 多路服用机制之一。Linux 中有三种 I/O 多路服用机制，一是 select，二是 poll，最后一个就是 Epoll 模式。该模式克服了 select、poll 模式需要遍历内核中的文件描述符，效率底下的缺点[7]。Epoll 不需要进行遍历描述符，当 I/O 或者说文件描述符可读可写的时候，内核会使用 mmap 将可读可写的

文件描述直接从内核态映射到用户态，所以 Epoll 模式不需要遍历所有的文件描述符集合，只需要遍历这些已经活跃并且准备好被读写的文件描述符就可以了。

epoll 的接口包括三个函数，如图 4.10。

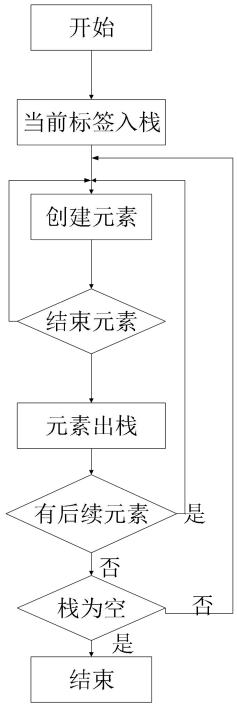
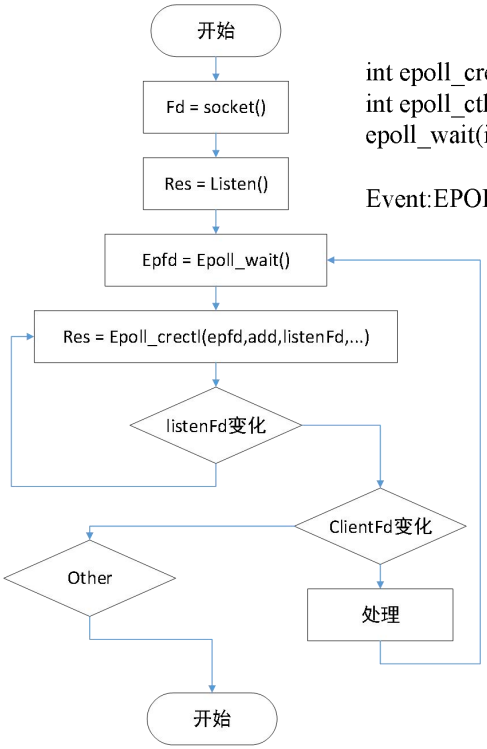


图 4.9 XML 打包流程



int epoll_create(int size);
int epoll_ctl(int epfd, int op, int fd, struct epoll_event *event);
epoll_wait(int epfd, struct epoll_event *events, int maxevents, int timeout);

Event:EPOLLIN 、EPOLLOUT、EPOLLPRI、EPOLLERR、EPOLLET

图 4.10 epoll 模型及 API

4.4.2 软件设计

服务器设计流程如图 4.11 所示。

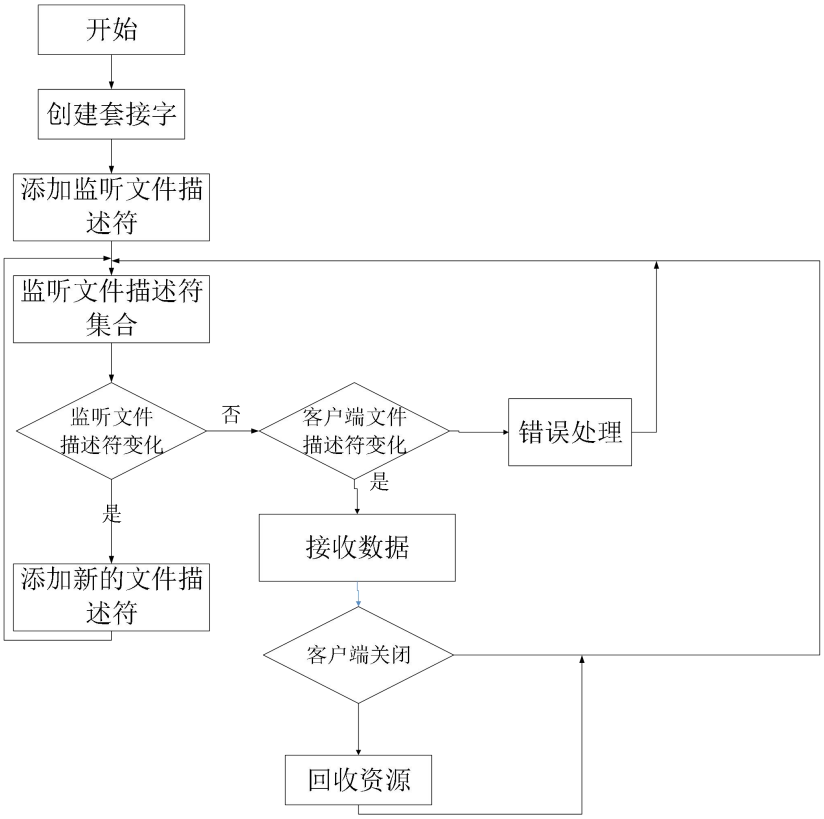


图 4.11 服务器设计流程

上述流程图的实现如下：函数 epoll_wait()表示阻塞知道有数据发送过来或者有新的文件描述符进来，建立连接时进入第一个 if，将该文件描述符加入集合当中，内核来监听这个集合中文件句柄的变化，如果是发送数据，则进入 else if 中，接受数据并进行处理。服务器首先进行 socket 的监听，然后会使用到 epoll。在 epoll 线程中阻塞，等待客户端的连接[8]。如果有新的客户端链接上来，那么会将客户端的文件描述符加入到监听队列当中。当这个文件描述符可读时，就说明客户端发送来信息，和客户端进行相应的交互，进行一些逻辑上的处理即可。

4.5 软件编译环境

本系统使用了一个非常流行方便使用的开发软件即 Keil uVision4，这个软件嵌入式开发中几乎是必备的。在开发过程中，使用该软件就可以完成所有的嵌入式程序的设计。

另外使用的一款重要的编码软件是 VIM。这是 Linux 开发人员必备的开发软件，该软件可以安装许多丰富的插件，效果在某些情况下甚至比 IDE 更棒。

界面如图 4.12 所示。

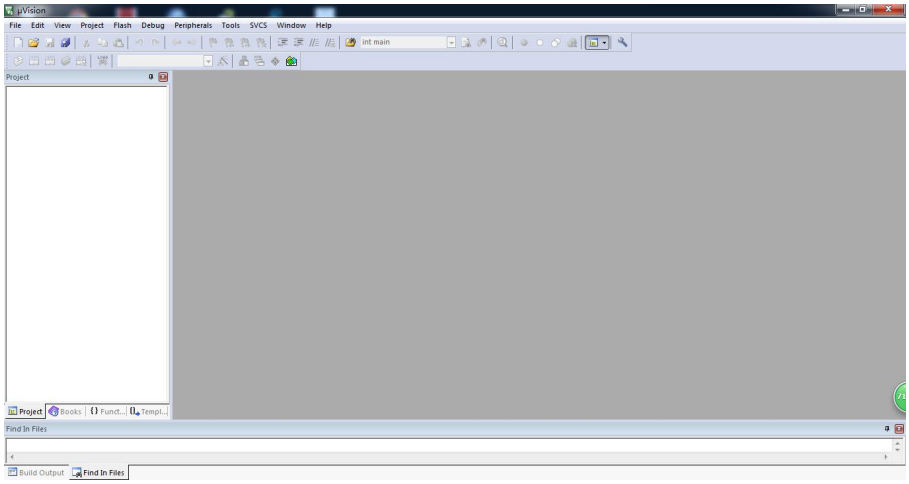


图 4.12 Keil uVision4 for ARM

采用 gcc 编译器进行代码编译。gcc 版本如图 4.13 所示。

```
root@Guanweibo 11:08:13#7:~/workspace/GraduationProject# gcc --version
gcc (Debian 4.9.2-10) 4.9.2
Copyright (C) 2014 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

图 4.13 gcc version

4.6 软件设计小结

本系统的软件设计部分主要分成了两个大模块，一个模块是嵌入式软件，使用 keil4 这个编译器来编译嵌入式软件的代码。这其中的内容包括 FreeRTOS 嵌入式操作系统的移植工作，Lwip 协议栈的移植工作等等。开发过程中，使用由大及小，我们需要控制传感器传给主控模块的数据应该怎样加以区别，他们的类型，节点位置等等，还要考虑怎么将这些数据进行打包，发送到串口转以太网模块，服务器网页部分首先要搭建后台功能，基本功能和框架实现后，首先完成操作系统，对操作系统的任务进行管理，以及完成操作系统任务之间的通信工作。通信工作主要使用的是消息队列。搭建完成这样的框架，然后开始实现具体的代码逻辑，实现具体细节的功能。另一部分是高并发服务器的设计，Linux 服务器端下 I/O 多路复用机制是解决高并发问题的有效办法，本系统采用的是 epoll 模型。并通过 j-link 的使用和 gdb 进行调试。

在开发整套软件的过程中，使用了 keil4 编辑器，他和 Linux 平台有互通的库，所以我直接在 Linux 系统上进行编程，然后在 Linux 操作系统上进行调试，然后进行编译。虽然出现了一些问题，但是经过调试以及代码的修改，测试功能正常无误后，软件得以正常上线。

第五章　系统调试结果

5.1 调试环境

5.1.1j-link

J-Link 可以和 keil4 编译环境无缝连接，使用非常方便，可以直接烧录程序还不用担心单步调试时程序跑飞。程序编译无误后，先 Flash-Download，下载程序再进入调试界面进行调试。j-Link 设备如图 5.1 所示。

图 5.1　j-link

5.1.2USR-TCP-232-Test

这个工具很强大的地方在与一方面他能读取串口转以太网模块的数据，另一方面还能获取和传递网络层数据，界面如图 5.2 所示。

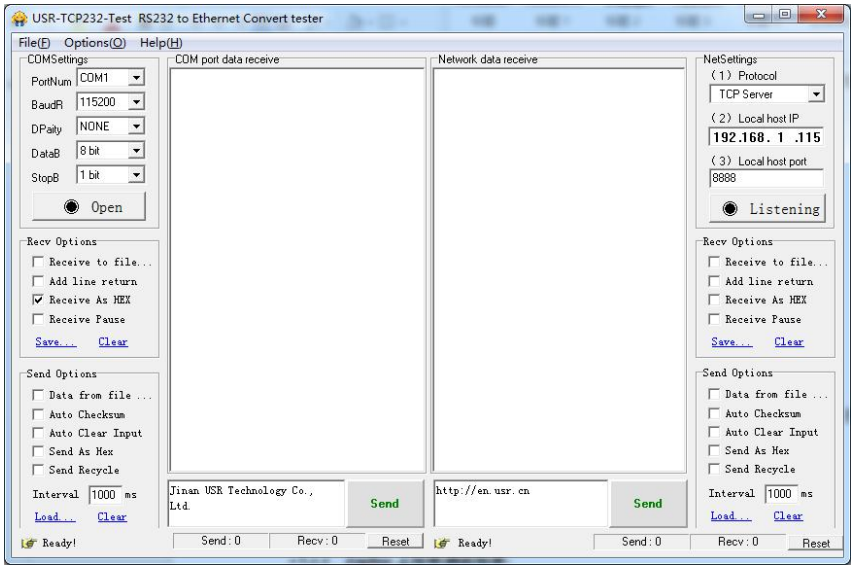


图 5.2　USR-TCP-232-Test

5.2 调试结果

5.2.1 数据采集调试结果

通过串口发送数据到主控单元。图 5.3 为串口发送数据并测试结果。

图 5.3　串口调试界面

5.2.2 网络发送调试结果

我们使用该工具监控 101.201.237.254ip 下 8080 端口的数据，如图 5.4 所示。

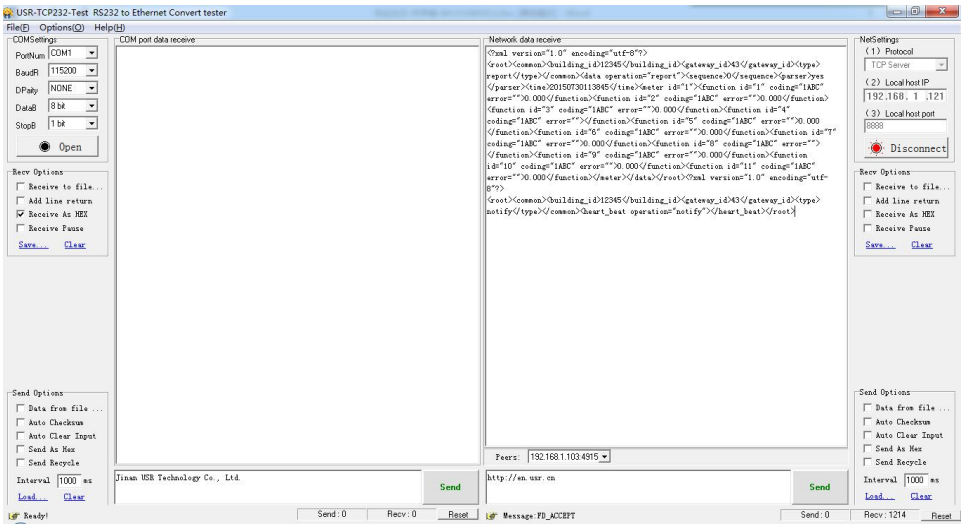


图 5.4　网络发送数据

ip 地址和端口成为一个套接字，如果我们有两个套接字加上传输协议就可以进行数据传输，这称为三元组，服务器接收数据监控，如图 5.5 所示。

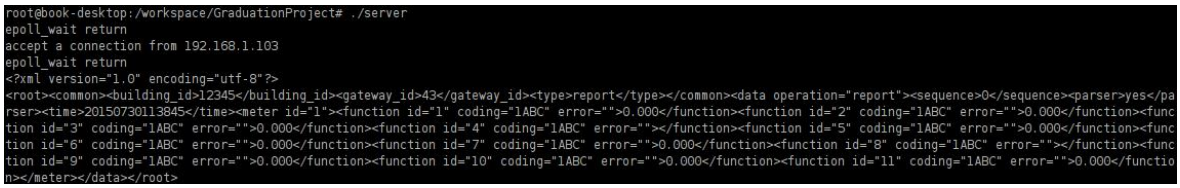


图 5.5　服务器接收数据

5.2.3 系统实物调试结果

传感器采集温度和烟雾信息，传递给主控模块，主控模块将这些数据封装后发送给串口，串口通过以太网传送到我们的服务器上，服务器再将数据进行解析整理显示在网页上。

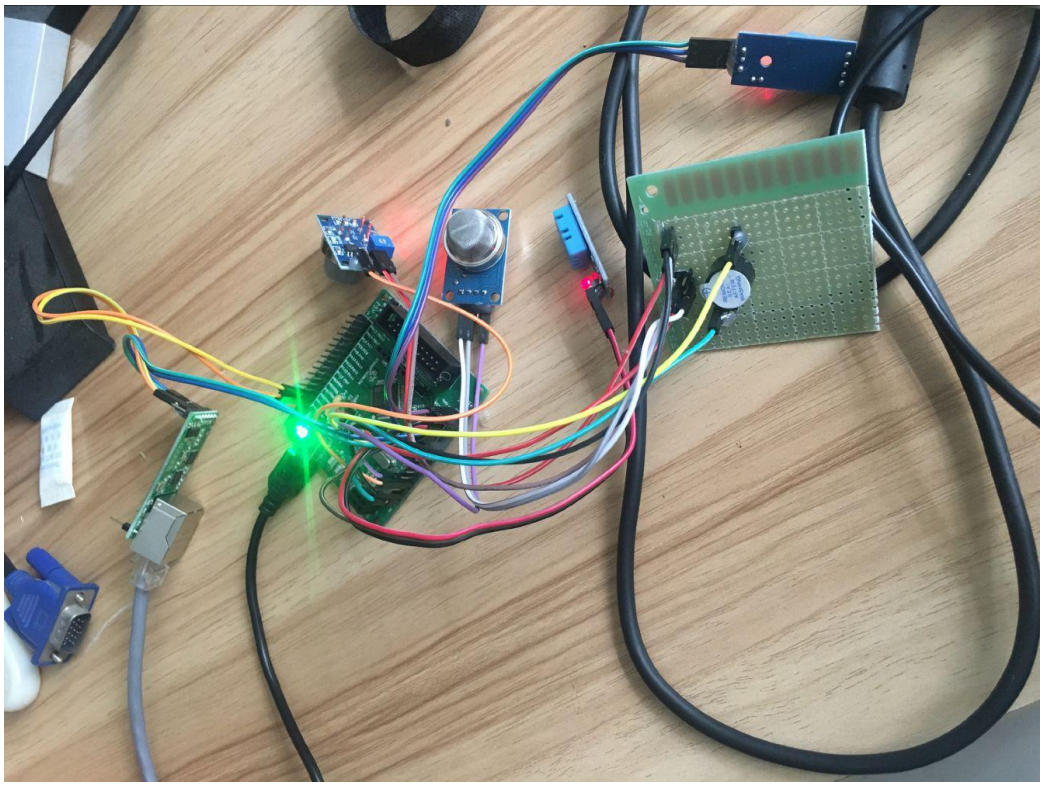


图 5.6　数据采集上传

如图 5.6 所示，图中使用网口传输采集的数据，通过互联网将数据发送到服务器中。服务器端接收数据后对页面进行渲染显示。

结束语

从毕业设计选题成功到最后毕业设计完成，这个过程中我学到了很多新知识。

基于单片机的蔬菜大棚温度监控系统是物联网时代的重要产物，本系统需要将节点数据打包发送到服务器，所以采用 BSD socket 接口，这是 UNIX 下标准的网络编程接口。对所需要的知识掌握之后，接下来考虑的就是整体的设计与实现。

硬件设计部分主要包含两个节点，一个芯片，一个串口转以太网 USR-TCP232-T2 模块。节点当中包含一个烟雾传感器和一个温度传感器。在硬件设计过程中，期间串口转以太网模块出现过问题，原因是外围电路存在问题，修改电路之后解决了问题。软件设计部分一方面需要在硬件上采集两个节点的温度数据和烟雾数据，并将之打包传送到服务器，这时候还要注意将这个节点打造成一个结构体时要区分彼此的位置和类型。这样服务器才知道你传递来的这个数据是温度的还是烟雾的，是第一个节点的还是第二个节点的。

在整体系统完成之后，需要对整体进行测试，接上网线，插上电源，后面发现数据没有发送到服务器，网页上的数据没有实时更新，后面检查发现是服务器上的上传代码出了一点问题，重新上传之后就好了，成功进行了数据的发送。

当然，由于本人的研究能力和研究时间有限，本系统还有颇多不足之处：

网络的稳定性会影响数据传输的稳定性，硬件部分所处场地也必须有路由器和网络的要求，其次，服务器应对网络攻击的能力还有待商榷，对于恶意占领服务器连接数的情况没有应对措施，需要每隔一段时间进行更新一次，清理缓存，我的设计还没加入数据备份的功能。

致　谢

在本次毕业设计中，我得到了指导老师陈维佳老师的耐心指导，陈老师在论文的设计过程中提出了许多合理的建议，帮助我解决了毕业设计中遇到的各方面的问题，并不断向我传授分析和解决问题的办法，给我指出了正确的方向，使得论文不断完善。在这里非常感谢陈老师的指导和帮助，在此向陈老师表达最真挚的感谢！

同时，论文的顺利完成，也离不开同学和朋友们的关心与帮助。在整个的论文写作中，班里的同学和舍友们积极帮我查找相关资料，提供了许多有用的建议，在他们的帮助下，我才能最终完成整篇论文。在此一并感谢！

大学四年的时光如同白驹过隙，在西安邮电大学我学习了很多，也成长了很多。感谢大学四年遇到的所有人和事，希望我的母校能越来越好！

参考文献

[1] 刘迎春.传感器原理设计及应用[M].哈尔滨工业大学出版社.

[2] 余成波.传感器与自动检测技术[M].高等教育出版社.

[3] 尤建军.吸气式感烟火灾探测器（单管/双管）等 12 种产品介绍[J].消防技术与产品信息.2016(03):20-30.

[4] 曾海峰.基于以太网的嵌入式系统远程固件更新设计与实现[J].网络安全技术与应用.2015(01): 131-134.

[5] 张青青.lwip 协议栈的移植[J].信息系统工程.2015(08):65-67.

[6] 张青青.lwip 协议栈现状[J].现代工业经济和信息化.2015(10):107-109.

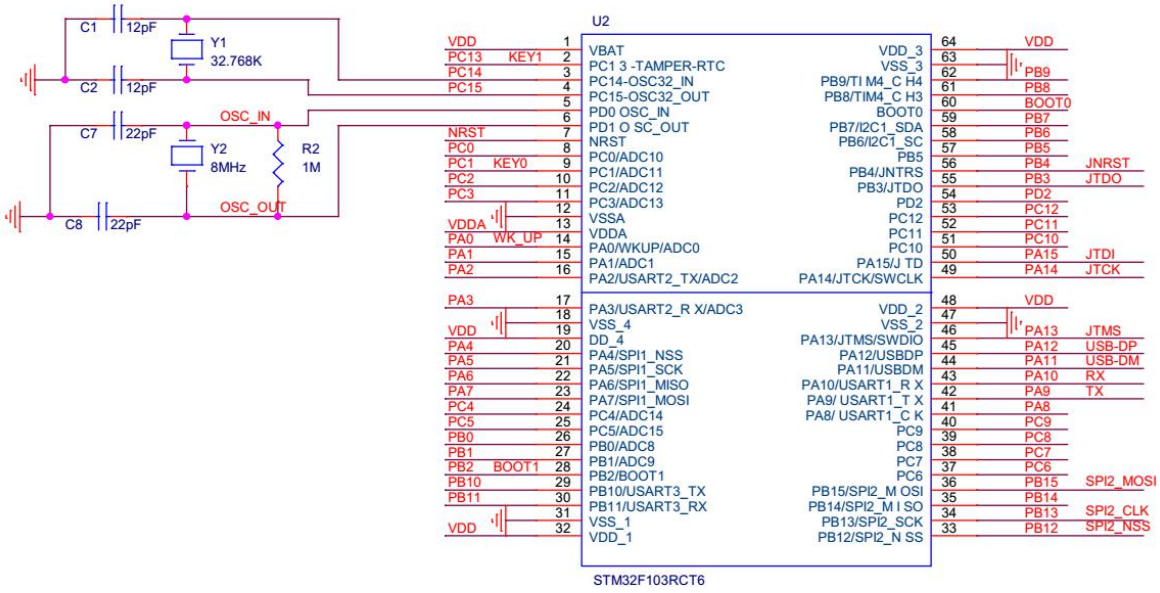
[7] 李伙钦.基于 WebSocket 的实时消息推送的设计与实现[J].科技视界. 2015(03):122-126.

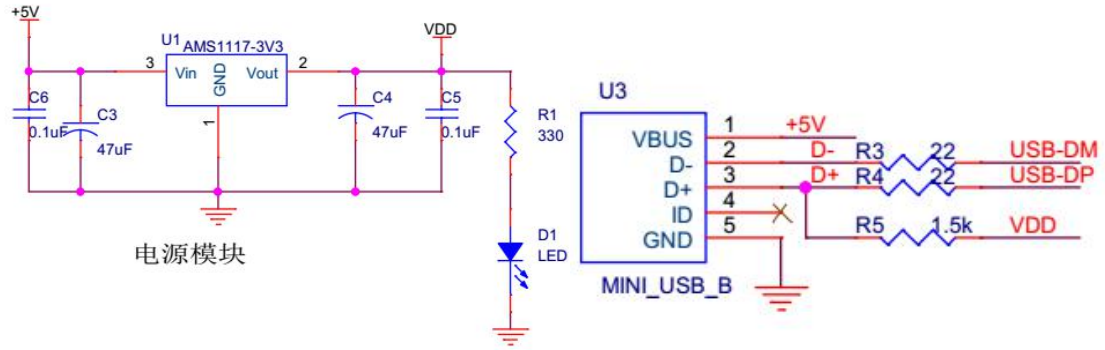
[8] Guo X L, Fei L, Bing Q H, Long C. Design of fire alarm system in laboratory[J]. Trans Tech, 2014, 912(2): 1232-1235.

[9]Jamri M.S. et al. Development of Novel Fire Alarm Warning System Using Automated RemoteMessaging Method[J]. Trans Tech, 2015, 793(3): 578-584.

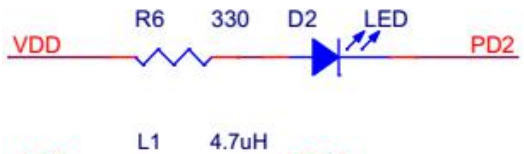
[10]Zhang M. Discussion of Optimize Method of Fire Alarm Dispatching Based on Operation Reasearch Principle[J]. Elsevierjournal, 2011, 714(4): 689-694.

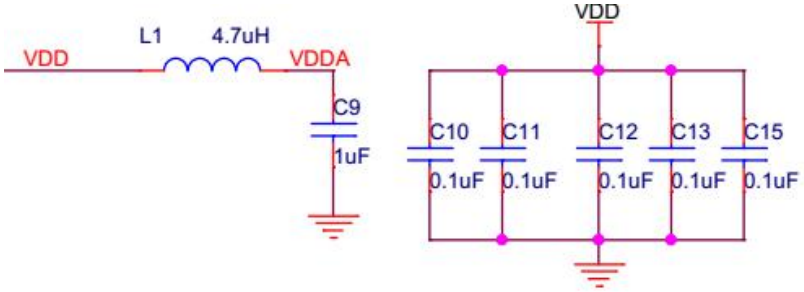
附录 A

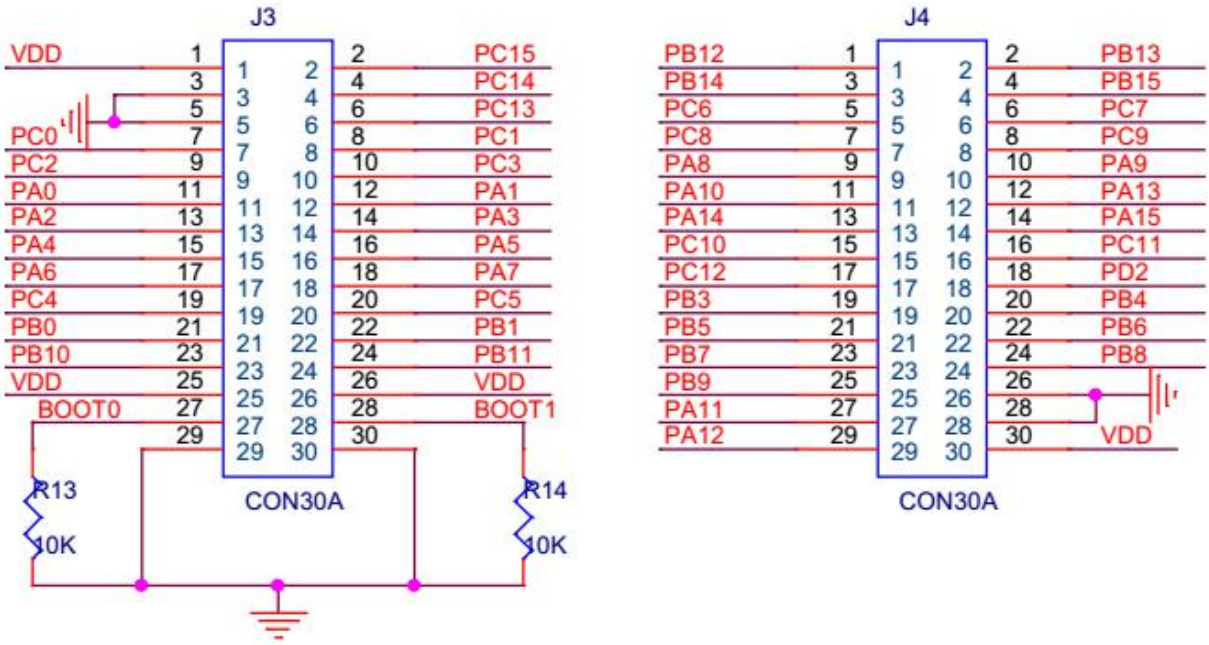

U2
C1 12pF
C2 12pF
Y1 32.768K
C7 22pF
Y2 8MHz
R2 1M
C8 22pF
OSC_IN
OSC_OUT
VDD 1
PC13 KEY1 2
PC14 3
PC15 4
5
6
NRST 7
PC0 8
PC1 KEY0 9
PC2 10
PC3 11
VDDA 12
PA0 WK_UP 13
PA1 14
PA2 15
PA3 16
17
18
VDD 19
PA4 20
PA5 21
PA6 22
PA7 23
PC4 24
PC5 25
PB0 26
PB1 27
PB2 BOOT1 28
PB10 29
PB11 30
31
VDD 32
VBAT
PC1 3 -TAMPER-RTC
PC14-OSC32_IN
PC15-OSC32_OUT
PD0 OSC_IN
PD1 O SC_OUT
NRST
PC0/ADC10
PC1/ADC11
PC2/ADC12
PC3/ADC13
VSSA
VDDA
PA0/WKUP/ADC0
PA1/ADC1
PA2/USART2_TX/ADC2
PA3/USART2_R X/ADC3
VSS_4
DD_4
PA4/SPI1_NSS
PA5/SPI1_SCK
PA6/SPI1_MISO
PA7/SPI1_MOSI
PC4/ADC14
PC5/ADC15
PB0/ADC8
PB1/ADC9
PB2/BOOT1
PB10/USART3_TX
PB11/USART3_RX
VSS_1
VDD_1
VDD_3
VSS_3
PB9/TI M4_C H4
PB8/TIM4_C H3
BOOT0
PB7/I2C1_SDA
PB6/I2C1_SC
PB5
PB4/JNTRS
PB3/JTDO
PD2
PC12
PC11
PC10
PA15/J TD
PA14/JTCK/SWCLK
VDD_2
VSS_2
PA13/JTMS/SWDIO
PA12/USBDP
PA11/USBDM
PA10/USART1_R X
PA9/ USART1_T X
PA8/ USART1_C K
PC9
PC8
PC7
PC6
PB15/SPI2_M OSI
PB14/SPI2_M I SO
PB13/SPI2_SCK
PB12/SPI2_N SS
64 VDD
63
62 PB9
61 PB8
60 BOOT0
59 PB7
58 PB6
57 PB5
56 PB4 JNRST
55 PB3 JTDO
54 PD2
53 PC12
52 PC11
51 PC10
50 PA15 JTDI
49 PA14 JTCK
48 VDD
47
46 PA13 JTMS
45 PA12 USB-DP
44 PA11 USB-DM
43 PA10 RX
42 PA9 TX
41 PA8
40 PC9
39 PC8
38 PC7
37 PC6
36 PB15 SPI2_MOSI
35 PB14
34 PB13 SPI2_CLK
33 PB12 SPI2_NSS
STM32F103RCT6

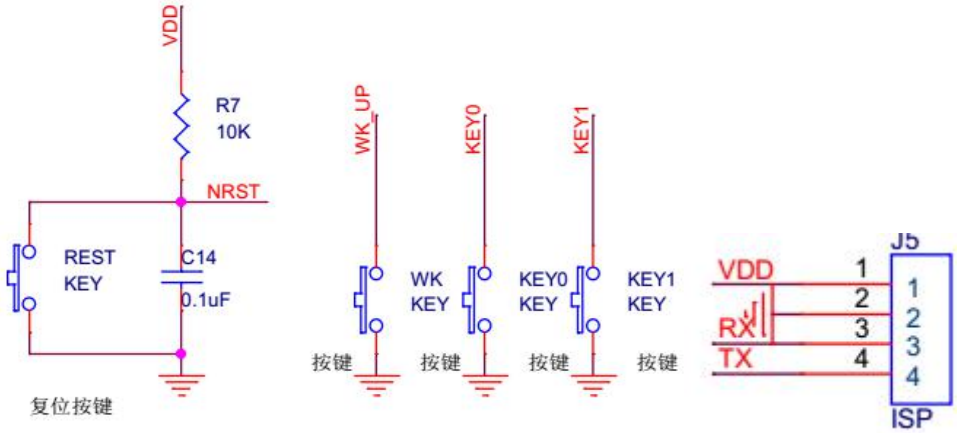

+5V
U1 AMS1117-3V3
3 Vin GND Vout 2
C6 0.1uF
C3 47uF
C4 47uF
C5 0.1uF
VDD
R1 330
D1 LED
电源模块
U3
VBUS 1 +5V
D- 2 D- R3 22 USB-DM
D+ 3 D+ R4 22 USB-DP
ID 4
GND 5
R5 1.5k VDD
MINI_USB_B

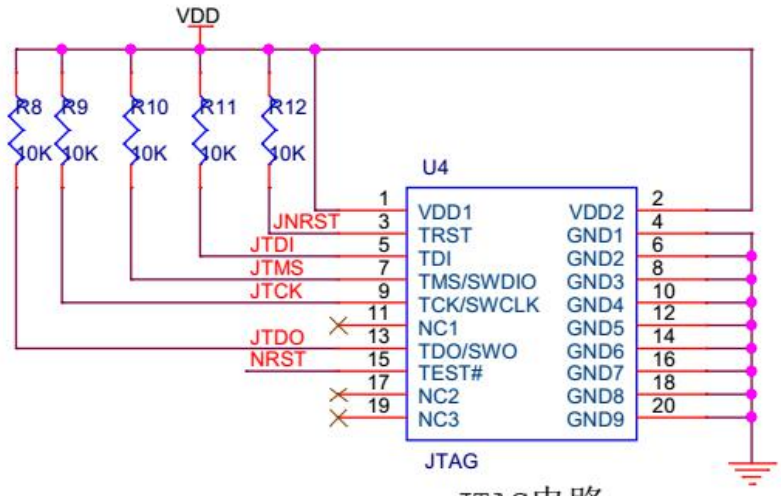

J1
GND 1
VDD 2
SPI2_CLK 3
SPI2_MOSI 4
PB11 5
PB10 6
SPI2_NSS 7
PB1 8
CON8
OLED模块接口


R6 330 D2 LED
VDD
PD2


L1 4.7uH
VDD
VDDA
C9 1uF
VDD
C10 0.1uF
C11 0.1uF
C12 0.1uF
C13 0.1uF
C15 0.1uF


J3
VDD
PC0
PC2
PA0
PA2
PA4
PA6
PC4
PB0
PB10
VDD
BOOT0
PC15
PC14
PC13
PC1
PC3
PA1
PA3
PA5
PA7
PC5
PB1
PB11
VDD
BOOT1
R13
10K
R14
10K
CON30A
J4
PB12
PB14
PC6
PC8
PA8
PA10
PA14
PC10
PC12
PB3
PB5
PB7
PB9
PA11
PA12
PB13
PB15
PC7
PC9
PA9
PA13
PA15
PC11
PD2
PB4
PB6
PB8
VDD
CON30A


VDD
R7
10K
NRST
REST
KEY
C14
0.1uF
复位按键
WK_UP
KEY0
KEY1
WK
KEY
KEY0
KEY
KEY1
KEY
按键
按键
按键
按键
J5
VDD
RX
TX
ISP


VDD
R8
R9
R10
R11
R12
10K
U4
JNRST
JTDI
JTMS
JTCK
JTDO
NRST
VDD1
TRST
TDI
TMS/SWDIO
TCK/SWCLK
NC1
TDO/SWO
TEST#
NC2
NC3
VDD2
GND1
GND2
GND3
GND4
GND5
GND6
GND7
GND8
GND9
JTAG

JTAG电路