目 录

摘要..........................................................i

关键词.......................................................ii

[第一章 绪论 2](#_Toc481917343)

[1.1课题背景及意义 2](#_Toc481917344)

[1.2国内外发展现状 3](#_Toc481917345)

[1.3研究内容 4](#_Toc481917346)

[1.4 论文结构安排 4](#_Toc481917347)

[第二章 相关技术分析 5](#_Toc481917348)

[2.1 软件技术 5](#_Toc481917349)

[2.1.1 无线通信协议 5](#_Toc481917350)

[2.1.2 qt框架 6](#_Toc481917351)

[2.2 硬件技术 6](#_Toc481917352)

[2.2.1 Arduino开发板 6](#_Toc481917353)

[2.2.2 nano pi开发板 6](#_Toc481917354)

[2.2.3 ESP8266无线开发板 7](#_Toc481917355)

[2.3 本章小结 7](#_Toc481917356)

[第三章 功能需求 8](#_Toc481917357)

[3.1 系统说明 8](#_Toc481917358)

[3.1.1 系统目标 8](#_Toc481917359)

[3.1.2 运行环境 8](#_Toc481917360)

[3.2 业务需求 8](#_Toc481917361)

[3.2.1 用例分析 8](#_Toc481917362)

[3.2.2 业务流程 10](#_Toc481917363)

[3.3 系统功能结构 11](#_Toc481917364)

[3.3.1 手机端具体功能描述 11](#_Toc481917365)

[3.3.2 服务器端具体功能描述 11](#_Toc481917366)

[3.3.3 家庭网关具体功能描述 11](#_Toc481917367)

[3.3.4 设备具体功能描述 12](#_Toc481917368)

[3.4 性能要求 12](#_Toc481917369)

[3.4.1 可用性 12](#_Toc481917370)

[3.4.2 可靠性 12](#_Toc481917371)

[3.4.3 性能 12](#_Toc481917372)

[3.5 本章小结 13](#_Toc481917373)

[第四章 系统设计 14](#_Toc481917374)

[4.1 硬件整体结构 14](#_Toc481917375)

[4.2 设备硬件结构 14](#_Toc481917376)

[4.2.1 ESP8266 14](#_Toc481917377)

[4.2.2继电器 16](#_Toc481917378)

[4.2.3温湿度传感器 17](#_Toc481917379)

[4.2.4气体烟雾传感器 17](#_Toc481917380)

[4.3 家庭网关硬件结构 18](#_Toc481917381)

[4.4 总体设计 19](#_Toc481917382)

[4.4.1 手机端设计 19](#_Toc481917383)

[4.4.2 服务器端设计 19](#_Toc481917384)

[4.4.3 家庭网关设计 20](#_Toc481917385)

[4.4.4 设备设计 20](#_Toc481917386)

[4.4.5 传输协议设计 21](#_Toc481917387)

[4.5 模块设计 21](#_Toc481917388)

[4.5.1 手机端 21](#_Toc481917389)

[4.5.2 服务器端 21](#_Toc481917390)

[4.5.3 家庭网关 21](#_Toc481917391)

[4.5.4 设备 22](#_Toc481917392)

[4.5.5 数据库设计 22](#_Toc481917393)

[4.6 类的设计与定义 24](#_Toc481917394)

[4.7 本章小结 26](#_Toc481917395)

[第五章 功能实现 27](#_Toc481917396)

[5.1 总体实现 27](#_Toc481917397)

[5.2 模块实现 27](#_Toc481917398)

[5.2.1网络模块 27](#_Toc481917399)

[5.2.2 数据库模块 28](#_Toc481917400)

[5.2.3 事务管理模块 31](#_Toc481917401)

[5.2.4 界面显示模块 35](#_Toc481917402)

[5.3 实现效果 37](#_Toc481917403)

[5.4 总结与展望 40](#_Toc481917404)

[5.5 本章小结 41](#_Toc481917405)

[谢 辞 43](#_Toc481917406)

[附 录 44](#_Toc481917407)

**智能家居控制系统的设计与实现**

**摘要：**近年来我们的经济发展水平不断的提高，对生活的环境要求也是越来越高，美观，舒适，方便，智能，安全等成为我们对生活的环境基础要求。而以物联网，智能硬件为代表的智能家居，恰好能够满足我们对生活环境的需求。本文首先介绍该课题的研究背景和本课题所需要完成的内容。接着介绍了相关技术，系统需求，系统设计，接着介绍了本系统完成的关键代码和完成成果。最后对本课题进行总结和展望，提出不足和需要改进的地方。

**关键词：**智能硬件,物联网,手持设备,远程控制,自动化

**The design and implementation of smart home control system**

**Abstract:**Raise our level of economic development in recent years, environmental requirements of life is also more and more high, beautiful, comfortable, convenient, intelligent, security, become our basic environmental requirements for life. With the Internet of things, smart hardware is the smart home for us to meet the needs of our living environment. This paper first introduces the research background and the content of this topic. Then the related technologies, system requirements, system design, and then the key code and achievement of this system are introduced. Finally, the paper summarizes and prospected the project, and puts forward the deficiency and the improvement.

**Keywords:** Intelligent hardware,The Internet of things, Handheld devices, Remote control, Automation

# 第一章 绪论

## 1.1课题背景及意义

早在上世纪七十年代欧美等国家就已经出现了家居智能化的概念，并不断的传播到全世界。而智能家居在我国的出现在二十世纪末，可以说是比较晚的。

智能家居集合了多种方面的技术，如硬件制造，自动化控制，远距离通信技术，它能够自动管理家中的设备，对用户的指令做出反应，用户可以在任何可以联网的地方来控制相应的设备。并且在一些简单的场景中做出正确的反应。在家中的设备中增加了计算单元，使用有线或者无线的通信方式将所有的设备组建成为一个网络，进行统一管理。方便了生活，解放了双手，增强了安全性。目前智能家居技术已经应用于安防报警，自动控制，家居控制，紧急救助等。

不过目前仍然有很多问题没有解决，目前没有统一的行业标准，各个厂家各自有各自的电气标准和通信协议。系统和设备的选择上很大的局限性。非常不利于整个行业的发展。缺少实用性的功能，设备控制功能较少，自动化程度不高，过度依赖遥控器，安全性，没有大厂商实质性动作等都是阻碍智能家居推广的重要因素。

近年来随着我们的科技和经济水平的快速提高，对生活的要求不仅仅停留在了美观，时尚等审美理念上，对生活环境有了更深层次的要求--智能化。智能家居是正在冉冉升起的新兴产业，该概念也是炙手可热，有关机构统计表明，在未来的五年内，全球智能家居市场的年平均增长率为8%，产品营业额已经达到了几百亿美元，是一块众多企业盯着的“蛋糕”，蕴含着无限的商机，也是未来家居科技发展的必由之路。

## 1.2国内外发展现状

早在上世纪八十年代,西方发达国家就已经开始出现智能家居的概念和产品,虽然早就已经出现,但是当时的产品实用性不强,智能度很低,需要大量布线，基本上没有任何的推广价值。

近年来随着智能硬件和物联网，云计算等新技术的兴起，智能家居才算是焕然一新，实用性，智能化得到了飞速的发展。安装，使用变得方便快捷。具备了大面积推广的技术条件。比尔盖茨的豪宅的智能化程度更是让人看到了科幻片中的生活即将成为现实。

在我国，智能家居的概念出现不过短短的十几年，开始发展缓慢,各大厂商投入较少。不过近几年随着新技术的产生和发展，国内也得到了快速的发展。作为未来家庭的必然，目前也是各大厂商的发力的重点。

技术上嵌入式芯片计算能力的提高，复杂的智能化有了计算的基础，而新的联网技术的出现和发展，使得远程控制成为现实。嵌入式操作系统的普及，极大的提高软件的移植性，重用性。降低了开发的工作量和难度。目前来说技术已经不再是阻碍智能家居的发展的因素。

人类对美好生活的向往和追求从来没有停止过，智能家居能够提供舒适，安全，方便，全自动的生活环境，提升生活品质，简化生活，必将是未来的标配。

## 1.3研究内容

作为智能家居的灵魂，控制系统是智能家居中最重要的一环，本课题将要完成由手机控制，通过远程服务器来实现对家庭中设备进行控制的系统，控制的设备分为三类，开关型设备，调节型设备，采集型设备，安防型设备，可查看环境数据，控制家电的开关，调节家电的运行状态，对防火防盗等具有实时监控，及时告警等功能。用户通过手机端在有网络的地方即可对家中设备进行控制。

## 1.4 论文结构安排

本文共分为5章：

第一章：绪论。简单叙述了本课题的研究背景，发展趋势，需要完成的内容。

第二章：相关技术分析。分析了智能家居和本课题使用的硬件和软件技术。

第三章：功能需求。对本课题进行需求和功能上的分析与设计。

第四章：系统设计。详细介绍了功能模块的设计。

第五章：功能实现。介绍具体实现的结果和实现的关键代码。

# 第二章 相关技术分析

## 2.1 软件技术

### 2.1.1 无线通信协议

通信方式分为无线通信和有线通信两种方式，而智能家居作为家庭小范围设备集群，有线通信的方式显得线错综复杂，对房子破坏性大，装修起来困难等因素，所以有线通信的方式并不适合智能家居的业务需求，无线通信的优点恰恰克服了有线通信的缺点。

当然无线通信的缺点也是存在的，安全性历来都是无线通信方式的人们最担心的因素，不过随着加密技术的不断提高，目前WIFI都采用了WPA2加密方式，WPA2加密采用的是AES算法进行加密，该算法是目前公认的牢不可破的加密算法加密方式，并且广泛的应用于金融财务、在线交易、无线通信、数字存储等对安全要求很高的领域。基本上密码设置得无规律长度够长，基本上是没有可能被破解的。

目前比较主流的无线通信技术有蓝牙，zigbee，WIFI等，其中蓝牙因为其超高的功耗，传输速率低，传输距离近等因素没有被使用。

zigbee基于IEEE802.15.4的通信协议，具有实时性强，功耗低，能自组网等优点，但是作为智能家居来说，zigbee有传输速率低，覆盖面低，连接互联网需要额外的转换模块，价格略高等缺点，而其因为功耗低，自组网等优点更多的应用于物联网方面。

WIFI基于IEEE 802.11的通信技术，通信速率快，带宽高，安全性强，信号覆盖范围和功耗有良好的平衡，普及性高，作为有稳定电源供应的家庭来说WIFI无疑是最好的通信方式。

WIFI通信是网络链路层协议，而应用层协议多采用http协议或者是自定义协议，可以说HTTP协议是目前互联网上使用最多的协议，当然，在智能家居中http协议不再是请求网页等功能，而是传输数据的标准。http协议分为协议头和数据，协议头表明数据的类型和作用。

### 2.1.2 qt框架

Qt是奇趣开发的C++跨平台的图形用户界面应用程序开发框架，几乎可以在任何的计算设备上运行，一次编码，重新编译以后就能在相应的平台上运行，只需要少许改动甚至不用改动代码，其采用良好的封装机制，使得qt模块化程度很高，重用性很好，使用的安全的信号与槽的机制，既能方便的实现，而且还保证了回调函数的安全正确的使用，界面采用OpenGL渲染，具有较高的美观性和有效的降低了资源的消耗，适合嵌入式设备的开发。

## 2.2 硬件技术

### 2.2.1 Arduino开发板

有实力的公司一般都是自己设计硬件，在本研究课题中硬件并非是主要的，更多的是强调控制系统的实现，所以本课题中所有的硬件都是购买现成的开源硬件。

作为低成本，高性能的计算模块Arduino开发板一直都备受开源爱好者的喜爱。Arduino最早由意大利一所学校的老师开发，一般由一块Atmel AVR单片机、一个晶振或振荡器（一种天然的时钟，可以以特定的频率发出时问脉冲使Arduino以正确的速度运行）和一个5v的直流电源组成。所有硬件设计都是公开的，一般运行由C++语言编写的裸机程序。具有高封装，硬件控制简单，价格低，计算能力强，技术支持多等优点。

### 2.2.2 nano pi开发板

NanoPi是友善之臂公司开发的，以嵌入式Linux爱好者为对象的嵌入式ARM开发板，具有低功耗，体积小，计算能力强，兼容树莓派IO口等优点，采用全志高性能CPU，直接从sd卡中运行Ubuntu操作系统，该操作系统也是专门为ARM芯片适配的，基本和桌面版无异。很适合物联网，智能家居，嵌入式设备，甚至当作桌面版使用。

### 2.2.3 ESP8266无线开发板

ESP8266是WIFI方面比较理想的soc，包含了完整WIFI网络方案，也可以连接在别的设备上，直接将WIFI当作串口进行操作。

ESP8266的使用分为两种场景，一种直接提供计算功能和WIFI网络功能，直接使用芯片上的GPIO口连接，集成传感器，控制器等设备，实现了用低成本，低功耗，低耗资源的WIFI智能设备一体化解决方案。 另外一种方式就是通过调试串口发送AI指令使其执行相应的网络操作，这种方案通常是在其他更强的计算芯片的基础上，通过串口进行控制，将网络操作转化为串口操作。

ESP8266高度片内集成各种电路，如天线开关balun、电源管理转换器，所以在应用时仅需极少的外部电路，而且包括前端模块在内的实际解决方案，在设计时就可以将所占PCB空间降到最低。装有ESP8266片上系统的WIFI通信系统能够在睡眠/唤醒模式之间的快速切换、配合低功率操作的自适应无线电偏置、前端信号的处理功能、故障检测和排除，可以消除同为无线传输方案的蜂窝/蓝牙/DDR/LVDS/LCD干扰。

## 2.3 本章小结

本章主要介绍了智能家居通信的两种方式—有线通信和无线通信，并且分别介绍了两种通信的方式的优缺点，总的来说还是无线通信的方式优点更强大。接着介绍了主要使用到的编程框架，qt强大的可移植性是使用其开发智能家居控制系统不二选择。智能家居自然少不了硬件的参与，esp8266作为最实惠的WIFI片上系统绝对是设备控制器的首选。而家庭网关作为家庭中枢必须提供超强的计算功能，Nanopi开发板刚好提供了经济而又强大的ARM处理单元和嵌入式Linux系统。

# 第三章 功能需求

## 3.1 系统说明

### 3.1.1 系统目标

智能家居具备很好的发展前景，而家居控制系统是智能家居的核心，本课题将围绕着智能家居控制系统展开，全部采用远程的方式对家中设备包括家电设备，门禁，窗户等家居的控制，对环境数据包括温湿度，一氧化碳浓度，二氧化碳浓度等数据的采集，对安防如视频监控，烟雾监控，火焰监控，人体感应监控等功能。

### 3.1.2 运行环境

手机：安卓系统，arm架构，SQLite3

服务器：linux，mysql，qt

家庭网关：嵌入式linux，qt，SQLite3

设备：C++裸机程序

## 3.2 业务需求

### 3.2.1 用例分析

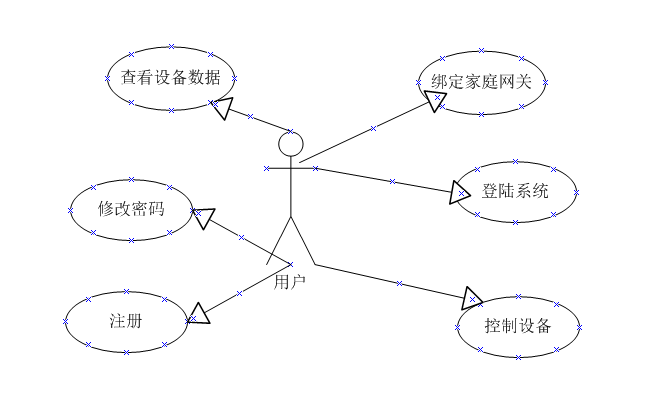


图 3-1 手机端用例图

手机端主要有用户管理：注册，修改密码，登陆身份验证，家庭网关的绑定，设备控制：查看设备采集，运行数据，对设备进行控制的功能。

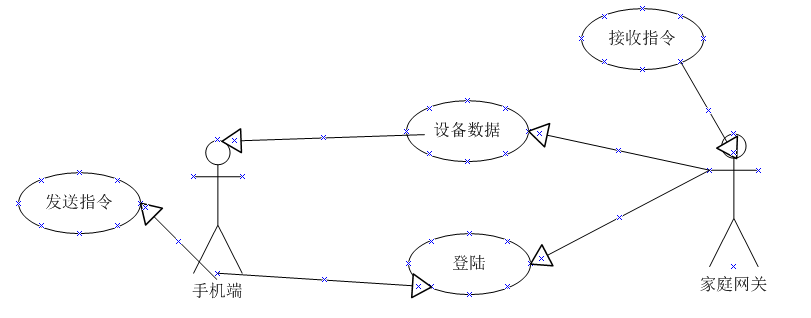


图 3-2 服务器端用例图

服务器作为连接用户和家庭的枢纽，具有用户，家庭网关登陆，身份验证，数据存储和转发等功能。

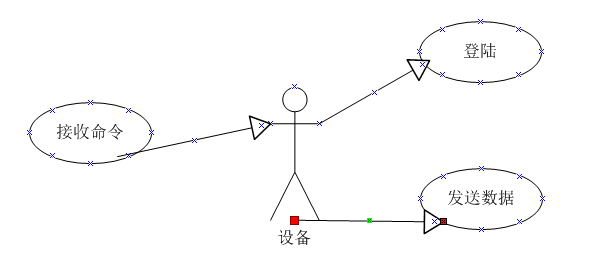


图 3-3 家庭网关用例图

家庭网关起到中转的功能，接收设备和服务器的信息，并相互转发。

### 3.2.2 业务流程



图 3-4 系统流程图

系统运行流程图，表明设备采集到数据发送到用户的过程

## 3.3 系统功能结构

### 3.3.1 手机端具体功能描述

* + 1. 获取设备信息和环境信息
    2. 对设备进行控制
    3. 显示警报信息

### 3.3.2 服务器端具体功能描述

* + 1. 与手机端进行通信
       1. 同步设备信息
       2. 实时更新设备信息
       3. 接收用户命令
       4. 用户登陆验证
       5. 同步个人信息
    2. 与家庭网关进行通信
       1. 同步设备信息
       2. 实时更新设备信息
       3. 传达用户命令

### 3.3.3 家庭网关具体功能描述

1. 获取每个设备的数据
2. 上传设备数据到服务器
3. 接收服务器传递过来的用户命令
4. 发送用户命令到具体设备

### 3.3.4 设备具体功能描述

* + 1. 共有功能
       1. 向家庭网关发送数据
       2. 接收并执行通过家庭网关发送过来的用户命令
    2. 设备分为开关型设备，采集型设备，交互型设备，安防型设备
       1. 开关型设备：根据用户命令来打开或者关闭设备
       2. 采集型设备：采集环境数据如温湿度，二氧化碳浓度，PM2.5浓度等
       3. 交互型设备：可根据用户命令动态调整设备状态，并实时反馈给用户，也可根据环境数据自动调整如下雨自动关窗等
       4. 安防型设备：对环境进行检测如烟雾检测，火焰检测，人体感应检测等，检测到异常立即发送告警信息给用户

## 3.4 性能要求

### 3.4.1 可用性

主要工作在设备上，直接与用户交互的手机端功能简单，提供GUI界面进行输入和显示提醒。具体根据界面上的提示操作即可，具有很好的操作性和可用性。

### 3.4.2 可靠性

控制，采集等信息因为数据实时更新所以允许低频率的数据丢失。

告警信息等实时性强，作用大的数据必须杜绝丢包的情况发生，传输层使用TCP协议进行传输，应用层必须用户确认，否则服务器将不断的超时重发。

### 3.4.3 性能

服务器采用异步，多线程，事件驱动等方式尽可能的提供并发数，一台主机在保证响应时间的情况并发数大概在1000左右。一般延迟500毫秒左右，因为从用户到设备经过了多个节点，而且网络状况也可能有不好的情况，所以最大延迟大概4秒左右。

## 3.5 本章小结

本章主要对本课题进行功能上的描述，描述了运行环境，要实现的功能，业务流程，性能，可靠性等硬性要求。为后面要实现的控制系统具体实现提供功能上的依据。方便后续的开发。

# 第四章 系统设计

## 4.1 硬件整体结构

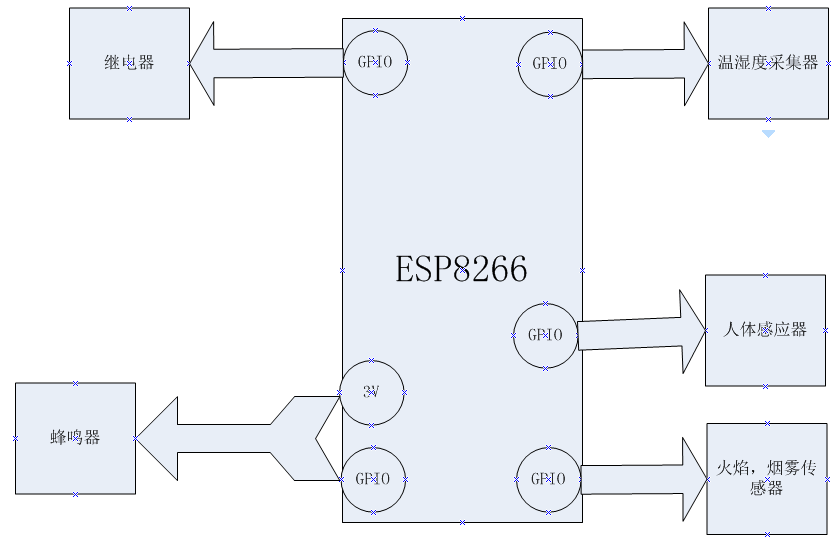


图 4-1 设备硬件体系结构

该硬件结构是控制系统控制的硬件核心，由esp8266提供电源，GND，使用GPIO进行控制和数据的读取，目前主要连接继电器，对设备进行简单的通电，断电操作，蜂鸣器，提供警报，各种传感器，采集器进行监控和环境数据采集。

## 4.2 设备硬件结构

### 4.2.1 ESP8266

设备控制器硬件主要采用ESP8266，采集器，继电器，控制器等全部连接在GPIO口上，通过改变输入输出方式来读取或写入高低电平，来完成我们需要的操作。

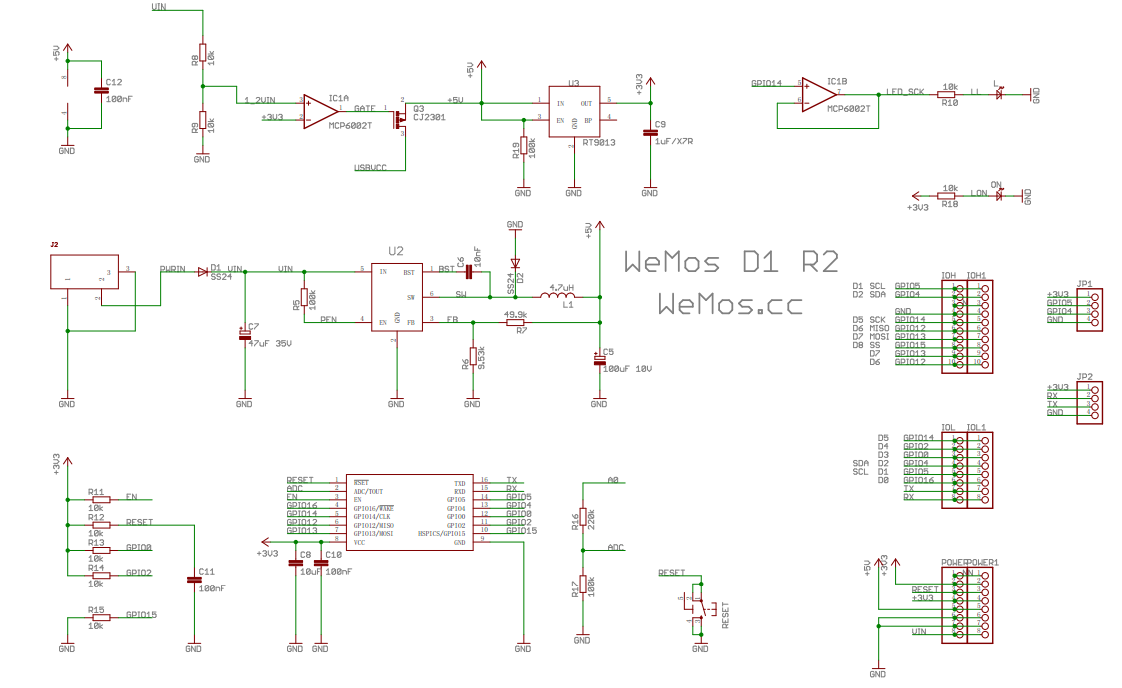


图 4-2 ESP8266开发板电路图1

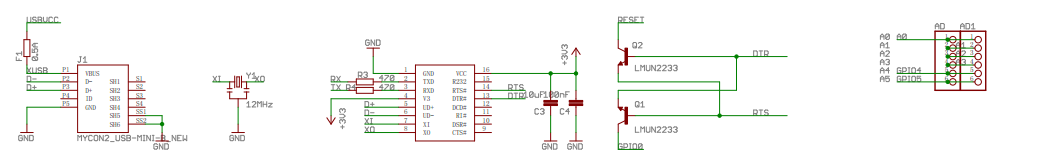


图 4-3 ESP8266开发板电路图2

表 4-1 ESP8266引脚表

|  |  |  |
| --- | --- | --- |
| **Pin** | **Function** | **ESP-8266 Pin** |
| TX | TXD | TXD |
| RX | RXD | RXD |
| A0 | Analog input, max 3.3V input | A0 |
| D0 | IO | GPIO16 |
| D1 | IO, SCL | GPIO5 |
| D2 | IO, SDA | GPIO4 |
| D3 | IO, 10k Pull-up | GPIO0 |
| D4 | IO, 10k Pull-up, BUILTIN\_LED | GPIO2 |
| D5 | IO, SCK | GPIO14 |
| D6 | IO, MISO | GPIO12 |
| D7 | IO, MOSI | GPIO13 |
| D8 | IO, 10k Pull-down, SS | GPIO15 |
| G | Ground | GND |
| 5V | 5V | - |
| 3V3 | 3.3V | 3.3V |
| RST | Reset | RST |

### 4.2.2继电器

继电器适合驱动大功率的电器，如电灯、电风扇甚至空调。单片机接继电器可以实现弱电控制强电。

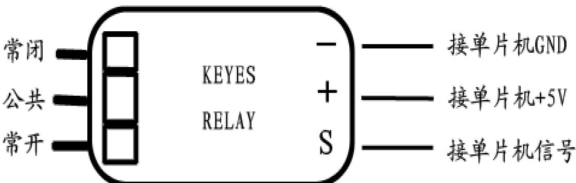


图 4-4 继电器原理图

如上图所示，将火线接在公共和常开端（没有控制信号时是断开的），然后S接单片机接数字GPIO口，根据单片机发送的高低电平，继电器进行通断，从而实现用低电压控制高电压的功能。

### 4.2.3温湿度传感器

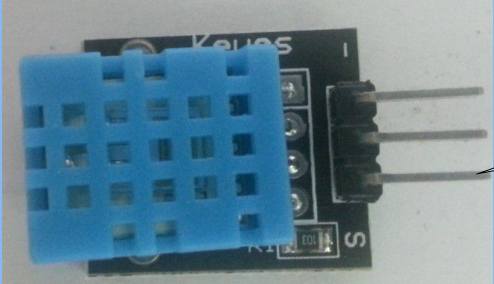


图　4-5温湿度传感器

DHT11数字温湿度传感器是一款温湿度复合传感器，可同时采集当前环境下的温度和湿度数值，而因为具有相应的数字转化电路，所以它支持数字信号的输出，无需自己采集模拟信号量再进行转化。简化了温湿度的数据解析的难度。

### 4.2.4气体烟雾传感器

MQ-2/MQ-2S气体传感器的主要原理为二氧化锡在清洁的空气中和可燃气体浓度较大的空气中所表现出来的不同且差别巨大的导电率。当传感器所处空气中存在可燃气体和烟雾时，传感器的电导率会随空气中可燃气体浓度的变化而表现出正相关的特性。因此只要检测二氧化锡的导电率即可完成对空气中可燃气体和烟雾的检测。

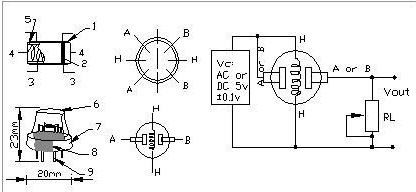


图 4-6 烟雾传感器电路图

## 4.3 家庭网关硬件结构

家庭网关采用Nanopi，该开发板没有带WIFI功能，但是该开发板性能很强，所以使用USB WIFI的方式进行连接WIFI。

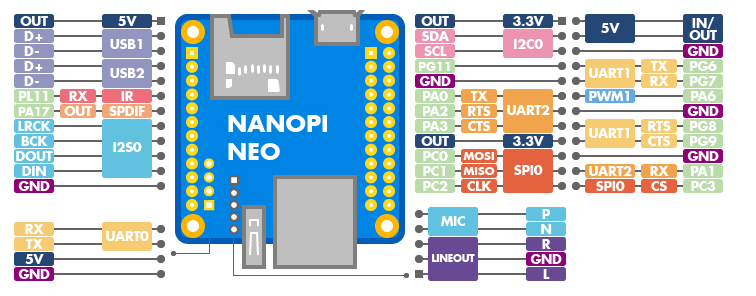


图 4-7 nanopi引脚图

表 4-3 nanopi引脚功能表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| Pin# | Name | Linux gpio | Pin# | Name | Linux gpio |
| 1 | SYS\_3.3V |  | 2 | VDD\_5V |  |
| 3 | I2C0\_SDA |  | 4 | VDD\_5V |  |
| 5 | I2C0\_SCL |  | 6 | GND |  |
| 7 | GPIOG11 | 203 | 8 | UART1\_TX/GPIOG6 | 198 |
| 9 | GND |  | 10 | UART1\_RX/GPIOG7 | 199 |
| 11 | UART2\_TX/GPIOA0 | 0 | 12 | PWM1/GPIOA6 | 6 |
| 13 | UART2\_RTS/GPIOA2 | 2 | 14 | GND |  |
| 15 | UART2\_CTS/GPIOA3 | 3 | 16 | UART1\_RTS/GPIOG8 | 200 |
| 17 | SYS\_3.3V |  | 18 | UART1\_CTS/GPIOG9 | 201 |
| 19 | SPI0\_MOSI/GPIOC0 | 64 | 20 | GND |  |
| 21 | SPI0\_MISO/GPIOC1 | 65 | 22 | UART2\_RX/GPIOA1 | 1 |
| 23 | SPI0\_CLK/GPIOC2 | 66 | 24 | SPI0\_CS/GPIOC3 | 67 |

## 4.4 总体设计

### 4.4.1 手机端设计

手机端直接面向用户，不直接对设备进行操作，通过服务器来执行读取数据，对设备进行控制等操作，通过界面来提供友好的显示和操作功能。

手机端分为网络服务模块，数据库操作模块，事务管理模块，界面显示模块这四个模块组成。

### 4.4.2 服务器端设计

服务器端主要起到用户和家庭链接的作用，主要为远程操作而设立。主要是对数据进行存储，解析用户和设备数据，保证告警信息等重要信息的送达，将用户和家庭网关进行对应，分类管理，确保每个用户都只能操作自家的设备，用户管理等。

服务器端主要分为网络服务模块，数据库操作模块，用户管理模块，数据解析模块这四个模块组成。

### 4.4.3 家庭网关设计

家庭网关的设立主要是每个家庭有很多个设备，如果每个设备直接连接到服务器，将对服务器的资源，性能形成严峻的考验和造成巨大的浪费，降低服务器的可连接的家庭数量，减低了服务器的性能，延长了响应时间。而采用家庭网关的方式，一个家庭只有一个家庭网关和服务器进行通信，而设备统一受家庭网关的管理，降低远程服务器的运行压力，也方便服务器对每户家庭的设备进行统一的管理。由此可见家庭网关的作用主要还是在降低服务器的运行压力上，属于数据中转，可供数据存储，并在每个家庭中只留一个数据出口，方便管理，提高安全性。

家庭网关主要分为网络模块，数据库模块这两个模块。

### 4.4.4 设备设计

因为设备的种类多样，每种设备的实现都有些不同，所以下面将对每种设备进行分析。

* + 1. 开关型设备：
       1. 功能：提供开关控制和状态查询功能
       2. 模块：网络服务模块，GPIO控制模块。
    2. 采集型设备：
       1. 功能：从传感器中采集环境数据，并将数据上报给用户。
       2. 模块：网络服务模块，传感器数据采集模块。
    3. 交互型设备：
       1. 功能：开关控制和设备状态调整，实时数据更新给用户，对环境数据变化自动进行处理。
       2. 模块：网络服务模块，设备状态调整模块，传感器数据采集模块。
    4. 安防型设备：
       1. 功能：警告开关控制，传感器数据采集，蜂鸣器警告提醒等功能。
       2. 模块：网络服务模块，传感器数据采集模块，告警模块。

### 4.4.5 传输协议设计

传输协议借鉴了HTTP协议，由于传输内容简单，所以做了简化。格式如下：

设备标识，数据长度，协议头，命令码，具体数据。

一条数据之间使用\r\n分隔，多条数据使用\r\n\r\n分隔。警告信息超时重传，手机端收到数据以后回复数据接收标识。

## 4.5 模块设计

### 4.5.1 手机端

1. 网络服务模块：采用tcp socket进行通信，为程序提供网络数据读写服务，对网络相关的功能进行管理。
2. 数据库模块：采用SQLite3数据库，主要存储设备运行数据，用户数据，为程序提供数据读写服务。
3. 事务管理模块：协调，管理其他所有模块，所有的界面显示，消息，数据读写全部通过该模块进行管理。是所有模块相互连接的桥梁。
4. 界面显示模块：显示界面，提供用户和程序进行交互的接口。

### 4.5.2 服务器端

* 1. 网络服务模块：采用tcp server socket进行通信，作为服务端，监听端口，建立连接，读写网络数据。
  2. 数据库模块：采用Mysql数据库，主要存储设备运行数据，用户数据，为程序提供数据读写服务。
  3. 用户管理模块：对用户和设备进行管理和数据对应的操作。
  4. 数据解析模块：解析用户和设备发送过来的数据，执行对应的操作，作为主要模块。

### 4.5.3 家庭网关

* 1. 网络服务模块：对设备作为服务端，对服务器作为用户端，起到初步解析和透传的功能。
  2. 数据库模块：将设备数据存储起来。

### 4.5.4 设备

* 1. 网络模块：提供网络数据传输服务。
  2. GPIO控制模块：通过对指定IO口读入高低电平来读取采集器数据，设备运行数据等，通过对指定IO口写入高低电平来控制继电器等设备的工作。
  3. 数据处理模块：对读取回来的网络数据，高低电平进行处理
  4. 告警模块：实现异常告警功能，以多种蜂鸣器，实时告警信息等方式提醒用户。

### 4.5.5 数据库设计

**4.5.5.1 设备信息表设计**

服务器，手机端，家庭网关为保持解析数据的一致性而使用相同的设备数据库表结构。表结构定义如下：

-- 设备存储

create table alldev\_info(

type varchar(2),

-- 设备类型

id varchar(5),

-- 该设备的id号

user\_id varchar(15),

-- 该设备所属的用户的id（主账户ID）

info varchar(200),

-- 该设备的开关组状态

expl varchar(400)

-- 该设备的说明,如该设备的位置和所执行的功能

);

**4.5.5.2 警告信息表设计**

-- 待发送给用户的数据(如警报,提醒等)

create table toBe\_sent(

user\_id varchar(15),

-- 该消息要发送到的用户的id

id varchar(5),

-- 设备号

mess varchar(200),

-- 要发送的内容

date varchar(15)

-- 该消息生成的时间

);

**4.5.5.3 用户信息表设计**

而服务器因为涉及到用户管理，所以服务器将比其他的两个终端数据库中多一个用户信息表，而对家庭网关只是数据中转，所以直接使用绑定的主账户登陆服务器即可，无需存储家庭网关的信息，其用户表定义如下：

-- 用户表:存储用户信息

create table user\_info(

id varchar(15) not null ,

-- 使用手机号码当做账户名

pwd varchar(33),

-- 用户密码

main\_id varchar(15)

-- 主账户ID，一个家庭只能有一个主账户，其他账户均在该主账户下，主账户将同样的存储在家庭网关中，家庭网关使用该主账户与服务器进行通信

primary key (id)

);

## 4.6 类的设计与定义

#### 4.6.1手机端

PNetWork：网络管理类，继承于QObject，采用事件驱动，异步的方式进行数据读写，主要的方法：

void back\_login(QByteArray back);//将服务器返回的传递给登录界面

void warn\_get(QByteArray warnMess);//提醒管理类接收到了警告信息

void info\_get(QByteArray infoMess);//提醒管理类有数据获得

void info\_up(QByteArray upMess);//提醒管理类有设备数据更新

void no\_net();//提醒所有的类当前无网络

PSql：数据库管理类，主要提供数据的存储和读取，主要方法：

void give\_dev(QList<struct dev\_info> info);//将查询到的数据返回给上层

void update\_info(QByteArray type,QByteArray id,QByteArray sta);//更新设备信息

void insert\_info(struct dev\_info info);//存储设备信息

void insert\_allinfo(QList<dev\_info> info);//保存所以设备信息

PManage：整个程序的管理类，对所有其他类进行管理，消息传递等。主要方法：

void info\_give(QList<dev\_info> list\_info);//更新数据

void makeView(QString which\_view,QByteArray arg);//根据参数创建界面

QMainWindow \*view\_this;//所有界面的基类，用于指向子类界面

PSql \*database;//数据库操作类

PNetWork \*net;//网络服务类

DevView \*d;//特殊界面类，其有QMainWindow所没有但是在管理类中使用的方法。

QList<dev\_info> list\_info;//存储设备数据

#### 4.6.2服务端

Server\_base：所有网络服务的基类，继承于QObject，主要实现了网络数据读写，一些公用的信号与槽。采用事件驱动，异步的方式进行数据读写。主要方法：

void byte\_int(QByteArray \*Bdata,int \*Idata,int wdo);//接收数据的准备工作

void len\_data\_process(QByteArray \*data);//发送数据之前的数据加工

Server\_dev：继承于Server\_base，主要新增了对数据的分析和对应的操作，对设备提供服务。

Server\_user：继承于Server\_base，主要新增了对数据的分析和对应的操作，对用户提供服务。

RWSql：数据库主要操作类。提供数据库的数据存储与查找，提供用户登录的密码验证。

#### 4.6.3家庭网关

communication\_far：作为客户端连接远程服务器，解析服务器发送过来的数据，也是设备发送给用户数据的传输通道。主要方法：

void read\_data();//异步的方式对数据进行读取

communication\_home:作为服务端接受设备的连接，解析设备发送过来的数据，也是用户发送数据给设备的传输通道，采用事件驱动，异步的方式进行数据读写。

RWSql：数据库操作类，进行数据的存储和读写。

#### 4.6.4设备

因为性能低下，相对简单，单一，并且没有操作系统的支持，所以并没有在设备中编写类。使用面对过程的编程思想实现。采用读取数据，发送数据，跟控制器，采集器交互排队轮询的方式实现。

## 4.7 本章小结

本章首先展示了本控制系统用到的硬件电路图，引脚图，实物图，从该三个方面介绍所用到的硬件，目前本系统并不涉及到电路的修改，所以电路仅仅是为了了解硬件电路设计，而引脚图这是最为关键的图表，引脚图标注了开发板和控制器，采集器等硬件的所有功能引脚，由开发板从电源中取电，通过变压电路转化为3V和5V电源给控制器，采集器等提供电力，而控制输出，数据采集输入则通过GPIO口进行通信，读取或者输出高低电平来进行二进制的传输。

而系统软件方面的设计则包含了控制系统各个节点，手机端，服务器端，家庭网关，设备，详细的描述了各个节点中的类的定义，作用，各个模块的作用。并且介绍了根据HTTP协议改编的自定义协议。由此通过软硬件的详细设计介绍，基本上确定了开发的方向，技术的具体实现。为后面的实际开发提供了理论基础。

# 第五章 功能实现

## 5.1 总体实现

* 1. 实现环境：
     1. 设备：直接运行裸机程序。
     2. 家庭网关：嵌入式Linux。
     3. 服务器：Linux桌面版。
     4. 手机：安卓系统
  2. 实现程度：
     1. 基本实现开关型设备的控制，基本实现环境数据采集，感应器基本可以工作，手机可以查看到数据和进行控制。

## 5.2 模块实现

### 5.2.1网络模块

//创建TCP服务端

Server\_dev::Server\_dev()

{

//TCP服务相关设置

ser\_dev = new QTcpServer(this);

if( !this->ser\_dev->listen( QHostAddress::Any,DEVPORT ) ){//创建监听

qDebug()<< this->ser\_dev->errorString();//创建失败,输出错误信息

ser\_dev->close();//关闭已经打开的套接字

return;

}

QObject::connect( ser\_dev , SIGNAL( newConnection() ) , this , SLOT( new\_con() ) );

//定时器相关设置

check\_time\_out = new QTimer( this );

QObject::connect( check\_time\_out , SIGNAL(timeout()) , this , SLOT(check\_time() ) );

}

//连接远程服务器

void unusefulview::connect\_far() {

if( farServer\_sock->isOpen() ) {

//该函数会在网络断开以后仍然会执行,所以需要检测套接字是否已经打开,打开的套接字需要及时的被关闭

farServer\_sock->close();

}

#ifdef DEBUG

qDebug()<< "connect far server" ;

#endif

islogin = false;

isconnect = false;

farServer\_sock->connectToHost( FARIP , FARPORT );//连接远程服务器

heart\_timer->start( 30000 );//启动定时

heart\_time\_out = 0;

}

### 5.2.2 数据库模块

//数据库操作，打开数据库，打开MySQL数据库

RWSql::RWSql( QObject \*parent ) : QObject( parent )

{

//打开数据库

db = QSqlDatabase::addDatabase( "QMYSQL" ); //加载驱动

db.setHostName( DATABASEHOST ); //设置数据库网络位置

db.setDatabaseName( DATABASE ); //设置使用的数据库名称

db.setUserName( DATABASEUSER ); //设置数据库登陆的用户名

db.setPassword( DATABASEPWD ); //设置该用户名的密码

bool ok = db.open();//打开数据库

if(!ok){//判断数据库是否成功打开

qDebug()<<"database opened failed\n "<< db.lastError().text();//输出错误信息

}

query = new QSqlQuery( db ); //新建sql执行类，用来与数据库进行交互

}

//将设备数据读取出来

void RWSql::show\_alldev\_info( QByteArray userid, QTcpSocket \*socket ) {

QString line\_sql = "select type , id , info , expl from alldev\_info where user\_id='" + userid + "'"; //组合成sql语句

QList<QByteArray> info; //保留结果

QByteArray one\_dev\_info = " "; //中间变量

query->clear(); //因为该变量重复使用，为防止查询结果混乱，使用前应先清空结果

query->exec( line\_sql ); //执行语句

while( query->next() ) { //循环读取返回的结果集

//添加设备类型

one\_dev\_info.append( query->value(0).toByteArray() );

//添加设备编号

one\_dev\_info.append( query->value(1).toByteArray() );

//添加设备描述

one\_dev\_info.append( query->value(3).toByteArray() );

//添加分隔符

one\_dev\_info.append( "\r\n" );

//添加设备数据

one\_dev\_info.append( query->value(2).toByteArray() );

//添加数据尾部

one\_dev\_info.append("\r\n\r\n");

}

//将数据添加到容器中

info.append( one\_dev\_info );

emit to\_back\_devinfo( DEVIN , socket , info );//发送信号通知槽已经将数据读取出来了

}

//添加设备信息

void RWSql::add\_dev\_info( dev\_info info ) {

QString line\_sql = "delete from alldev\_info where type='" + info.type + "' and id='"

+ info.id + "'"; //因为不知道该设备的信息是否已经在数据库中，所以每次插入的时候统一先试着删除该设备的数据，然后在插入数据以保证数据的成功插入

query->clear();

query->exec( line\_sql );

query->clear();

line\_sql = "insert into alldev\_info values ('" + info.type + "','" + info.id

+ "','" + info.user\_id + "','" + info.info + "','" + info.exp + "')";

query->exec( line\_sql );

}

### 5.2.3 事务管理模块

//使用信号与槽的机制对其他类进行管理

PManage :: PManage( QObject \*parent ) : QObject( parent )

{

Database = new PSql( this ); //创建数据库类

net = new PNetWork( this ); //创建网络服务类

QObject::connect(net , SIGNAL( info\_get( QByteArray ) ) , this , SLOT( info\_get(QByteArray) ) ); //建立信号与槽的连接

QObject::connect( net , SIGNAL( info\_up( QByteArray ) ) , this , SLOT( info\_up( QByteArray ) ) );

QObject::connect( database , SIGNAL( give\_usr( QByteArray,QByteArray ) )

,this , SLOT( getusr( QByteArray,QByteArray ) ) );

QObject::connect( database , SIGNAL( give\_dev( QList<dev\_info> ) )

, this , SLOT( get\_dev( QList<dev\_info> ) ) );

database->get\_info(); //数据库类读取存储的设备的数据

}

//创建界面和初始化的等操作

void PManage::makeView( QString which\_view , QByteArray arg ){

If ( which\_view == "FIRST" ){ //创建第一屏

view\_this = new FirstView(); //创建对象

QObject::connect( this ,SIGNAL( gomain( QByteArray ) ), view\_this , SLOT( goWhere ( QByteArray ) ) ); //信号与槽建立连接

database->get\_usr();

} else if ( which\_view == "LOGIN" ) { //登陆界面

view\_this = new LoginView();

//登陆返回的信息

QObject::connect( net , SIGNAL( back\_login( QByteArray ) ), view\_this

,SLOT( back\_login( QByteArray ) ) );

//将登陆成功的用户名和密码写入数据库

QObject::connect( view\_this , SIGNAL( addUsr( QByteArray , QByteArray ) ) , database

,SLOT( input\_usr( QByteArray , QByteArray ) ) );

}else if( which\_view == "REG" ) { //注册界面

view\_this = new RegView();

QObject::connect( net , SIGNAL( back\_band( QByteArray ) ), view\_this

,SLOT( back\_band( QByteArray ) ) );

QObject::connect( net , SIGNAL( back\_reg( QByteArray ) ), view\_this

,SLOT( back\_reg(QByteArray) ) );

} else if ( which\_view == "MAIN" ) { //主界面

view\_this = new MainView();

//接收警告信息

QObject::connect( net , SIGNAL( warn\_get(QByteArray)),view\_this

,SLOT( get\_warn( QByteArray ) ) );

MainView \*d = (MainView \*)view\_this;

d->get\_initinfo(arg);

}else if( which\_view == "DEVLIST" ){//设备列表

view\_this = new DevListView();

DevListView \*d = ( DevListView \* ) view\_this;

QObject::connect( this , SIGNAL( info\_give( QList<dev\_info> ) ), view\_this

,SLOT( get\_listinfo( QList<dev\_info> ) ) );

d->init\_info( arg , list\_info );

} else if ( which\_view == "VIEW" ) { //具体的设备的界面

view\_this = new DevView();

//截取设备类型和id

QByteArray type = arg.mid( 0 , 1 );

arg = arg.mid(1);

//查询设备信息,并将设备信息发送到具体设备控制界面

for(int i = 0 ; i < list\_info.size() ; i++ ){ //根据传递进来的参数进行查找对应的设备的数据

if( type == list\_info.at(i).type ){

if( arg == list\_info.at(i).id ){

dev\_info info = list\_info[i];

d = ( DevView \* ) view\_this;

d->getView( info );

}

}

}

}

//创建界面的信号与槽

QObject::connect( view\_this , SIGNAL( giveView( QString , QByteArray ) ), this

,SLOT( makeView( QString , QByteArray ) ) );

//网络状态

QObject::connect( net , SIGNAL( no\_net() ),view\_this,SLOT( back\_net() ) );

//发送数据

QObject::connect( view\_this , SIGNAL( sendToFar( QByteArray ) ) , net , SLOT( sendToFar( QByteArray ) ) );

view\_this->show();

}

### 5.2.4 界面显示模块

<!-- 界面使用XML样式文件进行创建 -->

<?xml version="1.0" encoding="UTF-8"?>

<ui version = "4.0" >

<class> FirstView </class>

<widget class = "QMainWindow" name = "FirstView" >

<property name = "geometry" >

<rect>

<x> 0 </x>

<y> 0 </y>

<width> 1080 </width>

<height> 1920 </height>

</rect>

</property>

<property name = "windowTitle" >

<string> FirstView </string>

</property>

<property name = "styleSheet" >

<string notr = "true" > background-image: url(:/new/prefix1/image/first.jpg); </string>

</property>

<widget class = "QWidget" name = "centralWidget" >

<layout class = "QVBoxLayout" name = "verticalLayout" >

<item>

<widget class = "QLabel" name = "label">

<property name = "font" >

<font>

<family> 华文行楷 </family>

<pointsize> 50 </pointsize>

<italic> false </italic>

<underline> false </underline>

<strikeout> false </strikeout>

</font>

</property>

<property name= "text" >

<string> EasyLife </string>

</property>

<property name= "alignment" >

<set> Qt::AlignCenter </set>

</property>

</widget>

</item>

</layout>

</widget>

</widget>

<layoutdefault spacing= "6" margin= "11" />

<resources/>

<connections/>

</ui>

//在头文件中使用命名空间的方式使用该界面类

namespace Ui {

class FirstView;

}

## 5.3 实现效果



图 5-1 手机主界面

手机端的主界面，可以选择控制，查看不同类型的设备。



图 5-2 设备列表界面

选择不同的类型的设备后，显示对应类型的所有设备。



图 5-3 开关型设备控制界面

开关类型设备的控制界面，控制开关型设备的通断。



图 5-4 交互型设备控制界面

交互类型的设备控制界面，可以和设备进行交互达到自己想要的状态。

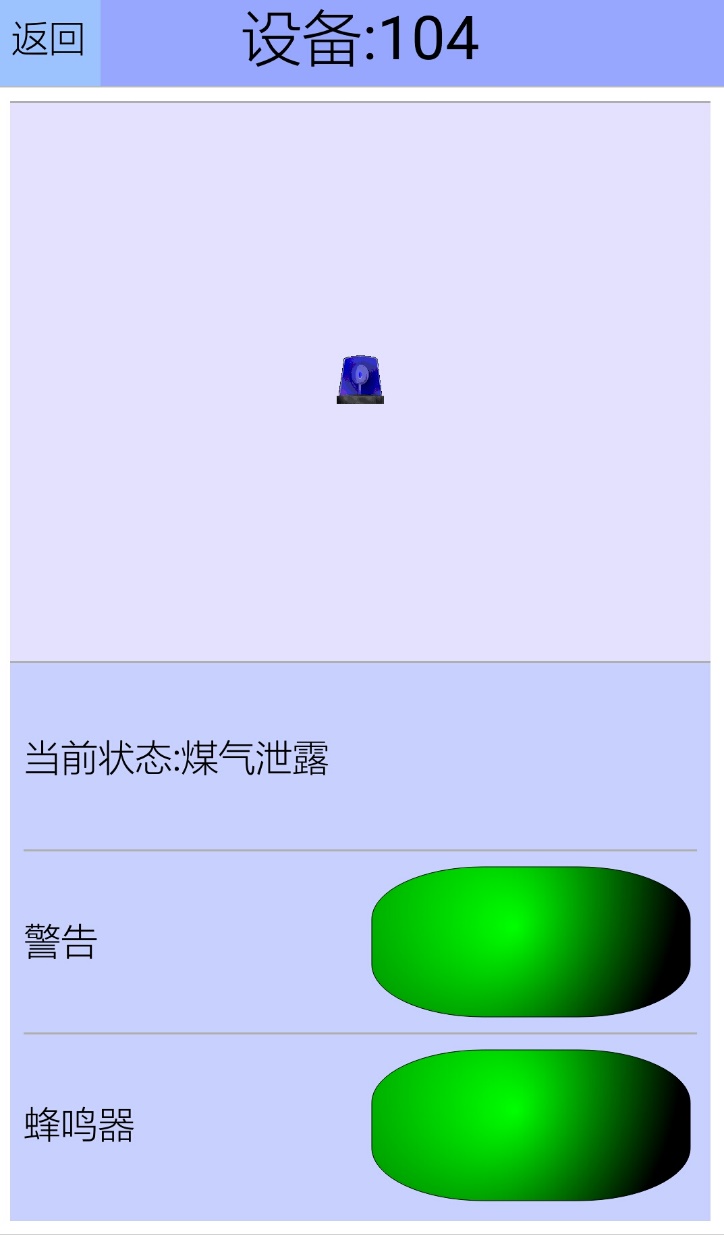


图 5-5 安防型设备控制界面

安防型设备控制界面，可以控制安防功能是否启动，蜂鸣器是否打开，而且还有详细的警报信息。

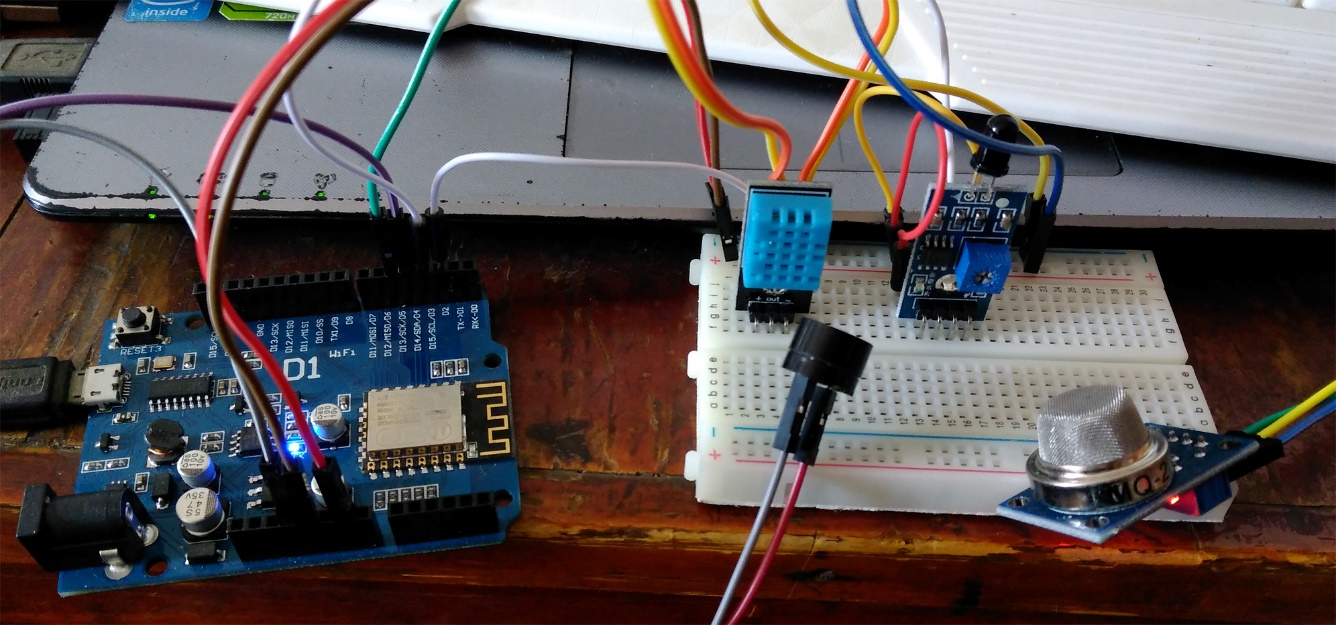


图 5-6 设备连接实物

设备实物连接图，使用esp8266开发板，通过USB口对设备进行供电，开发板对其他器件，采集器，传感器进行供电。上图连接了温湿度传感器，烟雾传感器，火焰传感器，蜂鸣器等 器件。

## 5.4 总结与展望

至此本智能家居控制系统的开发基本结束了，实现了对开关型设备的控制如电灯的开关和其他可以通过继电器进行控制的设备，环境数据采集型设备，可以对环境温湿度进行读取，安防型设备的控制，实现了烟雾和火焰报警，漏水检测，下雨检测，人体红外检测报警等功能。当然其中也存在很多的不足，交互型设备因为需要了解到相应的控制电路和协议，所以并没有实现功能，只能进行软件模拟，安全性有待加强，应该对报文的传输进行非对称加密等方式保证数据的安全，手机app端界面和功能过于简陋等都是存在的不足和需要改进的地方。

随着技术的不断发展，智能设备将会呈现出井喷的态势，而其中的控制系统作为用户和设备之间，设备和设备之间的纽带，更是具有极高的市场价值。我们也要随着技术的发展，不断的进行学习，将控制系统使用更新的技术，实现更加方便，快捷，智能，安全的功能，去探索更广阔的天空

## 5.5 本章小结

本章主要介绍了本控制系统的具体实现和具体实现的关键源码，并展示了控制系统的完成图片，由此正式的完成了该控制系统开发工作，基本上实现了对设备的控制，环境数据的采集等功能，也在实际中得到了验证，由此该控制系统已经具备了智能家居的应用的能力，不过该系统也存在着过于粗糙，细节仍需改进，安全性有待增强等缺点。

**参考文献**

1. 刘刚,赵剑川.Linux系统移植.北京:清华大学出版社 ,2011.1
2. 赵廷涛,刘冰,孙兴义,刘涛.Linux下的MySQL数据库编程.北京:清华大学出版社,2010.6
3. 丁林松 ,黄丽琴. Qt4图形设计与嵌入式开发 . 人民邮电出版社,2009.4
4. 安晓辉. Qt on Android核心编程.北京:电子工业出版社.2015.1
5. 安晓辉.Qt Quick 核心编程.北京:电子工业出版社.2015.1
6. 伍之昂.Linux Shell编程从初学到精通.北京:电子工业出版社
7. Marc Gregoire,Nicholas A.Solter,Scott J.Kleper,C++高级编程.北京:清华大学出版社,2012.10
8. Michael McRoberts .Arduino从基础到实践.北京:电子工业出版社,2013.3
9. Kenneth A.Reek.C和指针.北京:人民邮电出版社,2008.4
10. Hanson.C语言接口与实现.北京:机械工业出版社,2004.1
11. Gourley .HTTP权威指南.北京:人民邮电出版社,2012.9
12. Bovet.深入理解Linux内核.北京:中国电力出版社,2001
13. 宋宝华.Linux设备驱动开发详解.北京:人民邮电出版社,2010.11
14. Ezust.C++ QT设计模式.北京:电子工业出版社,2012.7
15. Martin Reddy.C++API设计.北京:人民邮电出版,2013.3
16. 张波.Qt中的C++技术.北京:电子工业出版社,2012.7
17. Grant Allen.SQLite权威指南.北京:电子工业出版社,2012.1
18. 韦东山.嵌入式Linux应用开发手册.北京:人民邮电出版社,2008.8
19. 王江伟,刘青.玩转树莓派.北京:北京航空航天大学出版社,2013.9
20. Bruce Molay.Unix/Linux编程实践教程.北京:清华大学出版社,2004.10
21. John Boxall. Arduino Workshop. San Francisco: No Starch Press,2013
22. Deitel . Associates , C++ How to Program , Prentice Hall,2011
23. Mark Summerfield, Advanced Qt Programming: Creating Great Software with C++ and Qt 4, Prentice Hall,2010

**谢 辞**

终于论文结束了，这也将意味着我的大学生活即将结束。四年来，我从一个脸庞有些幼稚的学生蜕变到即将踏入社会的成人，在这里我学到了很多很多的东西，不论是知识还是为人处事的道理，都是老师和同学给我的最大的财富。四年来的一幕幕都在脑海中留下深刻的印象，也将使我永远铭记。在此，我借此机会向所有帮助过我的老师和同学致以最真挚的感谢。

首先要感谢我的毕业设计指导老师余颖老师，她兢兢业业，耐心指导，经常关心我们的进度，不懂的问题也是及时，耐心，快速的回复，为我们的前进指引方向。使我在完成课题的过程中受益匪浅，这解决问题的方法和态度将是我受益终身的财富。

其次我要感谢其他的老师们，四年来你们教会我知识，教我道理，在我遇到问题时伸出的援手，让我在成长的道路上走的更快，更远。

我还要感谢我的同学们，团结有爱，相互帮助，我们这个大集体共同度过了美好的时光。也为即将走出校园的我，提供了融入社会的基础。

附 录