

**毕业论文**

基于ARM Linux环境的智能房屋

安全控制系统设计

**学生姓名**： **学号**：

235033101

段旭

自动化系

**系 部：**

电气工程及其自动化

**专 业：**

朱珊（讲师）

**指导教师：**

二○二五年六月

**诚信声明**

本人在此郑重作出声明：

所提交的毕业论文，乃是在导师的悉心指导之下，依靠自身独立开展研究工作而获取的成果，除了文中已经明确注明引用的相关内容之外，本论文并不涉及任何其他个人或者集体曾经发表过或者撰写过的作品成果，对于本文的研究给予关键贡献的个人以及集体，均已经在文中以清晰明确的方式给予标明。本人充分清楚地意识到本声明所产生的法律结果将由本人自行承担。

本人签名： 年 月 日

基于ARM Linux环境的智能房屋安全控制系统设计

摘 要

本系统基于全志 H616 处理器平台，设计并实现了一套智能房屋安全控制系统。系统采用模块化设计思想，结合简单工厂模式的软件架构，有效提升了系统的可扩展性与维护性。硬件方面，选用香橙派 Zero2 作为主控单元，集成可燃气体泄漏检测、视频监控、人脸识别开锁、家电设备状态显示与远程控制等功能模块，配合 MQ-2 气体传感器、OLED 显示屏、高清摄像头及 SU-03T 语音模块，满足多场景应用需求。软件部分采用多线程并发处理机制，分别实现语音监听、网络通信、MQTT 协议数据交互、环境监测与报警等功能，同时利用 MJPG-Streamer 实现摄像头实时视频流推送，保证了系统的高效性与实时性。通过配套安卓应用，用户可实现远程设备状态监测与控制，进一步提升了系统的人机交互体验与智能化水平。整体设计兼顾实用性、安全性与可扩展性，具有良好的应用前景与推广价值。

关键词: 智能房屋，ARM Linux，嵌入式，设计模式

**Design of Smart Home Security Control System Based on ARM Linux Environment**

**Abstract**

This system is based on the Allwinner H616 processor platform and designs an intelligent home security control system under the ARM Linux environment. A modular design concept is adopted, combined with a simple factory design pattern for the software architecture, effectively enhancing the system’s scalability and maintainability. In terms of hardware, the Orange Pi Zero2 is selected as the main control unit, integrating functional modules such as flammable gas leakage detection, video surveillance, facial recognition-based unlocking, appliance status display, and remote control. It utilizes components like the MQ-2 gas sensor, OLED display, high-definition camera, and SU-03T voice module to meet the needs of various application scenarios. On the software side, a multi-threaded concurrent processing mechanism is employed to separately implement voice monitoring, network communication, MQTT protocol data interaction, environmental monitoring, and alarm triggering. Meanwhile, real-time video streaming from the camera is achieved through MJPG-Streamer, ensuring system efficiency and real-time responsiveness. With the supporting Android application, users can remotely monitor and control the status of home devices, significantly improving the user interaction experience and the level of system intelligence. The overall design fully considers practicality, security, and scalability, making it highly promising for real-world applications and further promotion. Moreover, the system structure allows for easy functional upgrades and secondary development, making it a flexible and forward-looking solution in the field of smart home security control.

**Keywords:** intelligent home, ARM Linux, embedded system, design patterns

目 录

[第1章 绪论 1](#_Toc196774135)

[1.1 研究目的及意义 1](#_Toc196774136)

[1.2 国内外研究现状 1](#_Toc196774137)

[1.2.1 国外研究进展 1](#_Toc196774138)

[1.2.2 国内研究进展 2](#_Toc196774139)

[1.3 本文主要研究的内容 3](#_Toc196774140)

[第2章 系统总体设计方案 4](#_Toc196774141)

[第3章 系统硬件设计 6](#_Toc196774142)

[3.1 主控制器 6](#_Toc196774143)

[3.2 可燃气体检测 7](#_Toc196774144)

[3.3 OLED显示模块 7](#_Toc196774145)

[3.4 摄像头模块 8](#_Toc196774146)

[3.5 语音模块 9](#_Toc196774147)

[第4章 软件设计 10](#_Toc196774148)

[4.1 主控制器 11](#_Toc196774149)

[4.1.1 面向对象设计模式 11](#_Toc196774150)

[4.1.2 多线程并发处理 12](#_Toc196774151)

[4.1.3 基于INI文件的多模块设备配置与管理 14](#_Toc196774152)

[4.2 设备类统一接口设计 16](#_Toc196774153)

[4.2.1 LED灯实现方案 16](#_Toc196774154)

[4.2.2 蜂鸣器实现方案 17](#_Toc196774155)

[4.2.3 OLED屏幕实现方案 17](#_Toc196774156)

[4.3 监听类统一接口设计 17](#_Toc196774157)

[4.3.1 语音监听接口设计 18](#_Toc196774158)

[4.3.2 网络监听接口设计 18](#_Toc196774159)

[4.3.3 烟雾传感器监听接口设计 19](#_Toc196774160)

[4.3.4 消息队列监听接口设计 19](#_Toc196774161)

[4.3.5 MQTT监听线程接口设计 20](#_Toc196774162)

[4.4 视频监控及人脸识别 21](#_Toc196774163)

[4.4.1 视频监控方案 21](#_Toc196774164)

[4.4.2 人脸识别方案 21](#_Toc196774165)

[4.5 MQTT通信模块设计 22](#_Toc196774166)

[4.5.1 设备属性上报机制设计 22](#_Toc196774167)

[4.5.2 指令下发与消息响应流程 23](#_Toc196774168)

[4.6 安卓APP 24](#_Toc196774169)

[4.6.1 控制指令下发 24](#_Toc196774170)

[4.6.2 传感器信息显示 24](#_Toc196774171)

[4.6.3 视频监控 25](#_Toc196774172)

[第5章 系统测试与分析 26](#_Toc196774173)

[5.1 编译及运行 26](#_Toc196774174)

[5.1.1 Makefile 26](#_Toc196774175)

[5.1.2 Shell 26](#_Toc196774176)

[5.2 系统测试环境 27](#_Toc196774177)

[5.2.1 硬件环境搭建 27](#_Toc196774178)

[5.2.2 软件环境搭建 28](#_Toc196774179)

[5.3 系统功能模块测试与分析 28](#_Toc196774180)

[5.3.1 模块功能测试 28](#_Toc196774181)

[5.3.2 总结 31](#_Toc196774182)

[第6章 总结与展望 32](#_Toc196774183)

[6.1 总结 32](#_Toc196774184)

[6.2 展望 32](#_Toc196774185)

[参考文献 33](#_Toc196774186)

[致谢 34](#_Toc196774187)

# 绪论

## 研究目的及意义

随着科技快速发展，通信技术、嵌入式技术以及物联网技术持续融合，极大地改变了人们的生产与生活方式，智能家居作为智慧生活里的关键部分，正从概念迈向现实，逐渐进入普通家庭，并且在高端住宅中得到了初步应用，借助对家庭环境、家电设备以及安全防护等方面实施集成控制与智能化管理，智能家居系统提升了居住环境的安全性、舒适性以及便利性，改善了用户的生活效率与品质。然而，现阶段市面上诸多智能家居解决方案仍存在不少问题，像成本高昂、布线复杂、维护险阻、系统扩展性差以及用户操作门槛较高等，这些问题在中低端家庭中表现得较为突出，限制了智能家居技术的普及与推广[1]。怎样开发一套成本适中、功能实用、操作简便且系统开放性和扩展性强的智能控制系统，成为智能家居研究领域急需解决的关键问题。

本研究基于ARM开发板以及Linux操作系统平台，设计并实现了一个智能房屋安全控制系统，系统运用模块化设计理念，借助传感器对室内环境状态进行实时监测，同时达成对家用电器与安防设备的智能控制以及状态反馈，系统核心部分以嵌入式Linux平台作为控制中心，并凭借本地通信接口连接各类设备，用户可依靠基于Android平台开发的手机客户端进行远程控制、状态查询等操作，真正达成“人在外，家可控”的目标。

## 国内外研究现状

### 国外研究进展

国外智能家居起步发展的时间相对较早，在技术层面也趋于成熟状态，追溯到20世纪70年代，X10通信协议的问世开启了智能家居的早期摸索进程，步入21世纪之后，物联网即IoT的出现有力推动了智能家居的发展，Google、Amazon、Apple等公司纷纷推出智能产品，像2011年推出的Nest恒温器以及2014年推出的Amazon Echo智能音箱，这些产品结合云计算以及语音交互技术，有效提升了家庭自动化水平。近些年来，人工智能也就是AI、大数据、边缘计算等技术完善了智能家居生态，促使智能安防、智能照明、智能家电实现了深度融合[2]，

当下，国外的研究主要聚焦于智能交互、家居安防、能源管理以及隐私保护这些方面，Google Assistant、Amazon Alexa等AI语音助手得到广泛应用，可支持自然语言控制。Google Nest、Ring等公司推出智能摄像头、智能门锁，结合人脸识别、移动检测技术，达成了远程监控以及自动报警的功能，智能家居在能源管理领域取得了突破进展，比如Nest恒温器可以依据用户习惯进行智能调节，提升了能效[3]，隐私保护成为重点研究方向，众多企业采用端到端加密、区块链等技术，以此提升数据安全性。然而国外智能家居产品价格偏高，部分生态系统呈现封闭状态，这对市场普及度产生了影响，智能家居作为融合自动化控制、信息技术与现代家庭生活的新兴交叉领域，最早发源于欧美国家，20世纪70年代，美国提出X10协议，开启了智能家居系统的初步探寻之路，进入21世纪，随着技术的持续积累以及商业模式的逐步成熟，智能家居在国外逐渐从实验室迈向现实生活，其研究也渐渐走向系统化、规模化[4]。

当前国外针对智能家居的研究要点主要集中于系统平台的统一、生态的构建以及用户体验的优化等方面，如美国、德国、日本等国家，其研究机构和企业围绕智能家居的“平台化集成”“场景智能协同”“用户行为建模”等方向开展了较为深入的研究工作，特别重视系统的稳定性、兼容性以及智能决策能力。部分高校和科研机构也针对智能家居的人机交互界面、行为识别机制、安全性建模等课题进行了长时间的探索，为智能家居系统的智能化与实用性奠定了理论基础[5]，

国外研究同样看重智能家居的开放性与标准化建设，期望打破不同厂商设备之间的壁垒，提升系统的可扩展性和互操作性。从整体状况而言，国外在智能家居系统架构设计、统一协议标准以及智能化服务等方面研究起步较早、基础稳固、成果丰富，形成了较为完备的理论与实践体系。

### 国内研究进展

与之相较，我国在智能家居领域起步的时间相对较晚，不过在近些年来发展速度较快，在产业推动以及应用落地这些方面有了一定进展，起初的时候，国内对于智能家居的研究主要聚焦于控制系统以及单点功能的达成，像是灯光控制、安防报警这类，随着“智慧城市”“物联网 +”等国家战略的推进，越来越多的高校、科研机构以及企业开始关注智能家居的整体系统设计以及智能协同控制。

当下，国内智能家居研究主要是围绕系统集成、平台互联、人机交互优化、智能控制逻辑等方向来开展，一些高校和科研院所逐渐建立了智能家居实验平台，针对系统模型构建、控制策略设计以及平台搭建做了系统性研究，国内科研也变得日益关注用户体验与场景适配，借助嵌入式开发、数据建模等手段提升系统响应的智能性与适应性[6]。

在应用推广这一方面，国内企业也积极投身于智能家居的研究与开发，依靠和科研单位合作，促使智能家居系统在居家生活、社区管理等场景里实现规模化部署，然而当前国内研究依旧存在标准不统一、系统分散、智能化水平高低不一等问题，理论体系还不够完善，需要在技术整合、平台架构、安全策略等方面强化。

## 本文主要研究的内容

本论文完成了一套智能房屋安全控制系统的设计与实现工作，此系统选用 Orange Pi Zero2 充当核心控制器，借助简化的工程设计模式搭建软件架构，达成了基于 Android 客户端的远程控制以及实时监控功能，系统整合了人脸识别与可燃气体泄漏检测报警模块，使得住宅的安全防护能力得到了提高。Orange Pi 拥有较强的处理能力，再配合工厂模式设计，让系统的可扩展性和灵活性得以提高，用户可借助移动端在任何时间、任何地点对家庭环境展开监测与管理，以此保障人身和财产安全，整个系统结构合理且功能完备，在提高智能家居智能化与便捷性的为用户带来更为安全、舒适的居住体验。

# 系统总体设计方案

本次所设计的智能房屋安全控制系统选用全志的H616处理器作为主控核心，凭借其强大的理能力以及丰富的外设接口，构建起一个有高度集成性且拥有良好拓展性的智能控制平台，系统整体采用模块化设计以及简单工厂模式的软件架构方式，这使得程序的可维护性与可扩展性得到了提升，方便后期进行功能拓展以及系统升级。

本设计的智能房屋安全控制系统设计框图如图 2.1所示，系统实现的核心功能有可燃气体泄漏检测、视频监控、人脸识别开锁、远程控制、家电设备状态显示等，其中安卓APP客户端作为用户交互的主要界面，支持远程状态查看以及设备控制操作，用户可随时借助手机对家中设备进行远程操控，提升了系统的实用性与安全性。

在功能实现方面，系统设计了多个功能线程来实现多任务并行处理，有语音监听线程、网络监听线程、可燃气体检测线程、MQTT监听线程以及消息队列监听线程，各线程协调运行，保证系统在高并发和复杂环境下的稳定性与响应效率，系统借助集成SU-03T语音模块实现语音指令识别与语音播报功能，用户可凭借语音对家庭照明、门锁等设备进行控制，并获取实时语音反馈提示。为提升用户的居住安全性，系统引入人脸识别模块实现智能门锁控制，识别依靠后自动执行门锁开闭操作，避免传统钥匙遗失带来的安全隐患，系统集成可燃气体传感器，实时监测环境中的可燃气体浓度，要是检测到泄漏，就会立即触发报警机制，凭借OLED屏幕状态显示、语音播报以及APP消息通知三重方式进行联动提示，最大限度保障用户的生命财产安全。

系统搭建了基于MJPG-STREAMER的视频监控服务，用户可以依靠APP实时查看家庭摄像头画面，提升系统的监控功能，系统还设计了OLED显示模块，用于本地实时展示门锁状态、照明状态、语音播报信息及报警提示等关键信息，提高了系统的人机交互体验，系统支持三种主要控制方式：一是语音控制模式，可依靠语音实现设备操作与状态播报，二是TCP远程控制模式，借助网络调试助手下发控制指令，三是MQTT通信模式，APP凭借MQTT协议实现设备远程控制与状态同步，保证数据实时性与操作便捷性。

本智能房屋安全控制系统在硬件设计、软件架构与功能集成方面进行了系统化、模块化设计，实现了多种智能化控制与高可靠性的安全防护，系统有良好的扩展性、操作灵活性及高安全性，可为用户提供更加智能、便捷、安全的现代化居住环境。

图示

AI 生成的内容可能不正确。

图 2.1 系统设计框图

# 系统硬件设计

## 主控制器

本系统选择以基于Allwinner H616芯片的香橙派Zero2作为主控平台，H616是一款四核64位Cortex-A53处理器，有较高性价比与低功耗特点，主频最高能达到1.5GHz，可运行Linux、Debian等操作系统，有不错的多任务处理能力，适用于智能家居中对计算性能和稳定性有较高要求的场景。相较于传统的8位或32位单片机，H616在性能和能效方面有优势，能同时承担语音识别、人脸识别、设备控制、远程通信等多项功能任务，保障系统高效运行。

香橙派Zero2搭载1GB DDR3内存和Mali G31 MP2 GPU，支持图像处理与视频流播放，可契合系统对图像识别和本地显示的需求。其低功耗特性让系统适合24小时长时间运行，降低能耗以及维护成本，提升系统的可靠性与实用性。

香橙派Zero2有丰富的接口资源，包含26个GPIO引脚、3个USB接口、HDMI输出、千兆以太网以及可选的Wi-F和蓝牙模块，便于连接各类传感器与外设，用户能依据实际需求扩展如温湿度检测、照明控制、门禁管理等功能，实现系统的智能化与多样化。综合而言，该平台为智能房屋安全控制系统提供了性能强、能耗低、扩展性强的理想硬件基础，整体硬件接线图如图 3.1所示。

图 3.1 硬件接线图

## 可燃气体检测

如图 3.2所示，智能房屋安全控制系统运用MQ-2可燃气体传感器模块，以此来实时监测环境里液化气、天然气、酒精、氢气、烟雾等可燃性气体是否存在，MQ-2 模块有着灵敏度较高、响应速度较快、稳定性不错、抗干扰能力较强、驱动电路简单等诸多优点，在家庭、工厂等场景的气体泄漏预警系统中有着广泛应用。

该传感器的核心材料是二氧化锡半导体，属于表面离子型N型导体。当处于200~300℃的工作温度时，二氧化锡会吸附氧分子形成负离子，使得表面电子减少，电阻增加，一旦和可燃气体接触，氧离子与气体发生反应释放电子，让导电性提高、电阻下降，产生与气体浓度相关的电压变化。

MQ-2 模块输出包含模拟信号和数字信号，本系统主要利用其DO输出与主控芯片相连，当气体浓度超过设定阈值时，比较器输出低电平信号，主控接收后触发蜂鸣器报警，并且依靠网络推送警报信息到用户手机 APP，达成实时远程预警，模块内置电位器可调节比较器门槛电压，调节传感器灵敏度以适配不同应用环境。

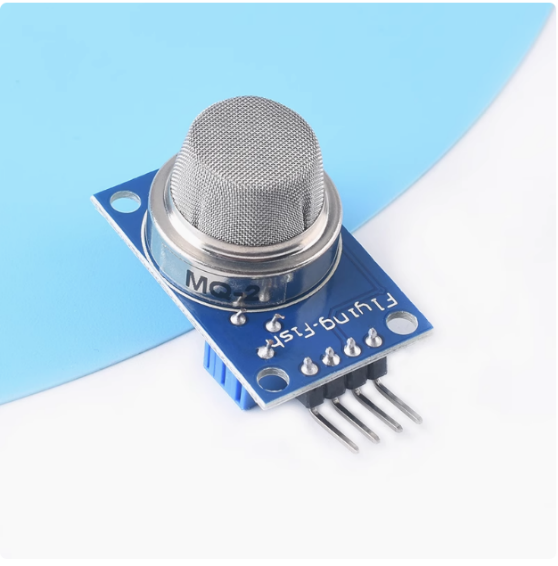
****MQ-2有良好的重复性和长期稳定性，正常检测浓度范围是100~10000ppm，在电路设计里，借助限流电阻保护传感器加热丝，保证其长期稳定运行，整体设计保证系统能在复杂环境下精准识别气体泄漏并及时做出响应，有效提高家庭安全等级。

图 3.2 MQ-2烟雾传感器模块

## OLED显示模块

在智能家居系统里面，OLED模块依靠其出色的显示性能变成了不可缺少的构成部分，因为有自发光的特性，OLED屏幕可呈现出高对比度、色彩鲜艳以及画面清晰的效果，就算是在光线不太好的环境当中也可稳定地进行显示，就像图 3.3所呈现的那样，它低功耗的优点也让其成为了理想的信息显示方案。

在系统运行的过程中，OLED模块身为主要的交互界面，肩负着设备状态提示、信息展示等功能，举例来说，系统启动的时候会显示“智能家居系统启动中”这类提示信息，方便用户知晓当前的状态，在设备控制的过程中，OLED屏幕还会实时展示各类信息，像门锁状态、烟雾传感器的状态、灯光开关状态等，配合语音播报达成视觉与听觉的双重反馈，提高交互体验。

图形用户界面

AI 生成的内容可能不正确。OLED屏幕支持静态文字显示，还可借助简单图标或者动画强化信息表达，它广视角、快速响应以及高分辨率丰富了显示内容，让系统更有科技感，作为连接用户和系统的关键桥梁，OLED 模块为智能家居系统的智能化以及实用性的提升给予了有力支持。

图 3.3 OLED模块

## 摄像头模块

摄像头模块于智能家居系统里同样起着关键作用，身为视觉核心部分，凭借其有的高效图像采集能力，为系统给予了关键的数据支撑，项目所采用的200万像素高清摄像头可清晰地捕捉环境当中的图像信息，为人脸识别以及安防监测等功能奠定了图像基础，该摄像头借助USB接口同系统相连，如图 3.4所示，保证了图像数据的高速传输以及稳定处理，提高了系统的整体性能以及响应速度。

为实现实时监控功能，系统整合了开源的mjpg-streamer库，把摄像头捕获的视频流编码成MJPEG格式并经由网络进行传输，借助此项技术，用户可借助手机App或者浏览器实时查看家中的监控画面，无需额外配置设备，简化了开发流程，同时也提升了系统的易用程度以及部署效率。

白色的游戏机

AI 生成的内容可能不正确。将高像素摄像头与mjpg-streamer相结合，智能家居系统实现了高质量的图像采集，并且保证了视频流的高效传输以及稳定播放，USB接口的运用以及效降低了功耗，保障了系统可长时间稳定运行，凭借这样的方式，用户可以随时了解家中状况，系统的智能化以及安全性得到大幅提升，提供了更为便捷、可靠的使用体验。

图 3.4 摄像头模块

## 语音模块

语音模块作为智能家居系统里的人机交互中枢，给用户给予了一种自然又便捷的控制办法，凭借语音指令，用户不用接触设备就能进行操作，这提高了使用的便捷性，还很适合居家、老人或者特殊人群的应用需要。

系统采用了SU-03T型号的语音识别模块，它有着灵活的配置能力以及良好的兼容性。这个模块支持网页配置，可自定义关键词与唤醒词，供电方式有多种选择，可凭借USB接口供电，也可依靠VCC与GND引脚接入主板，硬件连接很简便，如图 3.5所示。

在实际应用当中，用户可凭借语音控制灯光、门锁等智能设备，比如说出“打开客厅灯”就能促使语音模块依靠串口向主控设备发送指令，主控板收到后执行相应控制逻辑。系统也支持语音反馈功能，像播报“已为您打开客厅灯”，以此提高交互体验。

语音模块的加入提升了智能家居系统的操作便捷性与智能化程度，让用户在日常生活里可以更自然的方式完成各类控制任务，给家庭环境给予了更高的科技感与人机交互体验。

图 3.5 语音模块

# 软件设计

本系统运用面向对象编程思想搭建整体框架，主程序作为系统运行起始点，承担系统初始化、各模块加载以及线程调度等关键工作，其执行流程如图 4.1所示。

首先程序进入主函数main，注册信号处理函数，用于捕捉特定信号比如SIGQUIT，当系统运行期间接收到此信号，会触发相应清理函数，保障资源安全释放并达成程序平稳退出。这一机制提升了系统健壮性与可维护性。

紧接着，程序执行内存初始化操作，申请运行所需的堆内存空间。如果内存申请失败，系统将无法继续运行，因此程序会在此阶段直接退出。

内存初始化完毕后，程序开始对GPIO进行初始化，并检测摄像头图像服务像mjpg-streamer 是否正常运行。GPIO控制是本系统外部设备通信关键接口，摄像头服务是实现人脸识别与视频监控等功能前提，若 GPIO 初始化失败或摄像头服务未运行，系统会提示错误并退出，防止在设备未就绪时进行后续操作。

在硬件和服务模块准备好后，程序进入模块初始化阶段，首先创建消息队列结构，借助链表形式添加各个控制模块。链表结构使模块添加与维护更灵活，系统遍历该链表，逐个完成所有模块初始化工作。

随后依据链表中模块节点数量，为每个模块分别创建线程，线程中执行对应模块的get() 函数，实现对模块状态实时获取与控制，该设计有效提升了系统运行并发性与响应能力。

主线程完成线程创建任务后进入等待状态，直至所有子线程执行结束。最后主线程调用模块资源释放函数，释放各类动态分配内存资源，并对硬件接口进行关闭处理，实现系统完整退出。

凭借上述流程设计，系统在启动、运行、退出等各阶段都有明确控制机制，保证各模块协调运行，为后续功能拓展与维护提供良好结构基础。

## 图示 AI 生成的内容可能不正确。主控制器

图 4.1 主程序流程图

### 面向对象设计模式

本系统的软件结构设计运用了简单工厂模式，该模式在保证各模块职责清晰的情况下，兼顾了较好的可维护性以及可扩展性。

系统整体被划分为输入控制端工厂和输出执行端工厂这两大类，其中输入控制端工厂对来自手机APP、语音识别模块、网络调试助手以及烟雾传感器等设备的指令输入和环境状态检测进行统一管理。而输出执行端工厂则负责灯光、蜂鸣器、电磁门锁等执行设备的动作响应。

为了达成统一管理以及灵活扩展的目的，系统把所有控制模块与设备模块以结构体链表形式来组织管理，每个节点代表一个功能实体，包含模块的初始化、运行处理、状态反馈以及资源释放等接口，主程序凭借遍历链表，自动完成模块的注册与线程创建，支持各模块独立并发运行，实现高效的资源利用与任务调度[8]。

各功能模块在链表中维持了相对独立，又借助统一的数据结构使接口定义标准化，方便后期快速新增、删除或者修改模块，在线程管理方面，系统采用了轻量级线程模型，不同模块的控制逻辑凭借独立线程并行处理，彼此互不阻塞，为保证多线程环境下的资源安全访问，系统引入了互斥锁和条件变量，保障了临界资源的同步与线程间的协作。

总体而言，系统软件架构充分呈现了模块化、并发性与可扩展性的设计理念，依靠结合简单工厂模式、结构体链表管理方式以及统一接口规范，系统提升了开发效率和稳定性，还为后续功能拓展、设备兼容与平台适配提供了良好的支撑。

### 多线程并发处理

为保障各模块并行高效运行，同时确保资源访问安全，智能房屋安全控制系统采用了基于多线程的架构设计，并结合互斥锁（Mutex）与条件变量（Condition Variable）机制，实现了线程间的同步与通信。该设计不仅提升了系统整体的运行效率与稳定性，还有效保障了多线程环境下操作的正确性与协调性。

在并发执行环境中，多个线程可能同时访问或修改共享资源，如设备状态、控制指令或消息队列数据。若缺乏有效的同步机制，极易导致数据不一致、竞态条件（Race Condition）等问题。为此，系统在所有涉及共享资源的操作中引入互斥锁，确保同一时刻仅有一个线程能够访问临界资源。当一个线程获取锁后，其他线程将进入阻塞状态，直到该锁被释放，从而有效避免了数据冲突与系统异常。

此外，系统采用条件变量进行线程间的状态通知与同步控制。条件变量允许线程在特定条件未满足时自动阻塞，待条件达成后由其他线程发送信号唤醒继续执行，从而实现线程间高效的协作。例如，当消息队列监听线程在等待消息队列数据时，若队列为空则自动挂起，避免CPU资源空耗；一旦其他线程写入新消息，即通过条件变量通知消息队列监听线程唤醒处理，极大提高了系统资源利用率和响应速度。

整体系统设计了5个主要的控制线程以及若干动态生成的设备处理线程。其中，5个主要控制线程分别为：语音监听线程、Socket监听线程、MQTT监听线程、消息队列监听线程、烟雾报警监听线程，如图 4.2所示。

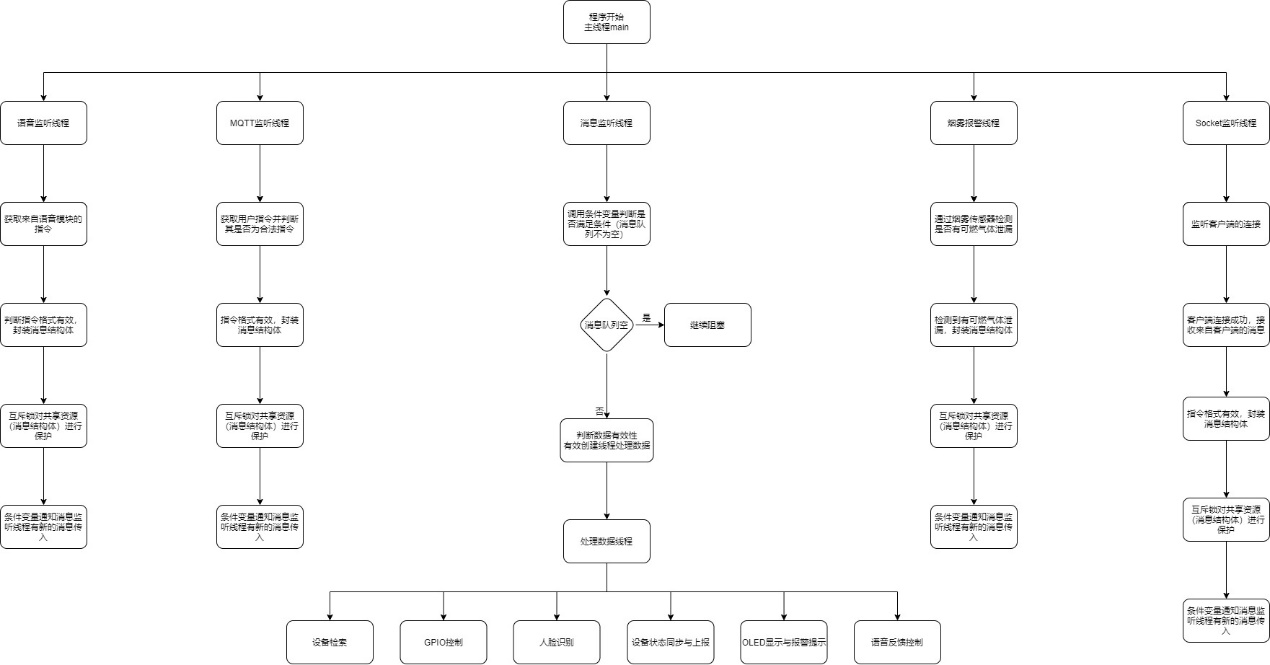
每个控制线程独立负责特定的功能模块，通过统一的消息队列与同步机制实现信息交互与控制指令的流转。下边具体描述消息队列监听线程设计和语音监听线程设计。

图 4.2 多线程设计

语音监听线程主要负责采集并处理用户通过语音输入的控制指令。其工作流程为：首先在线程内部调用专用的接口函数，获取语音识别模块传回的数据，然后对接收到的数据进行基本的格式校验（例如数据帧头为0xAA 0x55与数据帧尾0x55 0xAA检测），若指令格式合法且内容有效，则封装为标准化的消息结构体。其次通过互斥锁保护访问，将封装后的指令写入系统的消息队列，接着使用条件便令通知消息队列监听线程有新的消息到达，唤醒其进行处理。通过上述的机制，语音监听线程实现了对语音输入的异步、实时处理，避免了因等待输入导致系统其他功能受阻的问题。

消息队列监听线程作为系统的核心调度单元，主要负责从消息队列中接受指令并分发至设备控制模块。其工作流程为：首先当线程启动后进入到持续监听状态，当消息队列为空时，自动进入阻塞状态等待条件变量唤醒。当检测到消息队列中有新的消息到达时，此时该线程被唤醒，首先进行指令完整性与合法性检验。然后当数据校验通过后，为每条指令创建一个设备处理分离线程，以实现指令的异步并发处理，在处理完该指令后会自动释放线程所占有的资源。设备处理线程的主要执行流程包括：设备检索、GPIO控制、刷脸验证、设备状态同步与上报、OLED显示与报警提示、语音反馈控制、门锁自动关闭。通过以上流程，消息队列监听线程和设备处理线程协同工作，在保证高并发处理能力的同时，实现了资

源访问的安全性和操作流程的有序性。

### 基于INI文件的多模块设备配置与管理

为了实现智能家居系统中各类设备的灵活管理与动态扩展，本文设计并应用了一种基于INI文件格式的设备配置方案。通过将各设备的关键信息抽象存储在外部配置文件中，系统在启动阶段可根据配置内容动态加载设备列表，构建内部控制列表。该设计有效提升了系统的可维护性、扩展性与模块化程度，避免了在程序内部大量硬编码设备参数的问题，具有良好的通用性和适应性。

文本

AI 生成的内容可能不正确。配置文件名为gdevice.ini，采用护节划分结构，每个节对应一个独立的设备示例，包含该设备的基本控制参数与功能属性。具体配置示例如图 4.3所示，各字段含义如表 4.1所示：

图 4.3 设备属性定义

表 4.1 配置说明

|  |  |
| --- | --- |
| 字段名称 | 说明 |
| key | 设备唯一标识符 |
| gpio\_pin | 绑定的GPIO引脚编号 |
| gpio\_mode | 引脚模式（INPUT或OUTPUT） |
| gpio\_status | 初始电平状态（HIGH或LOW） |
| check\_face\_status | 是否启用刷脸功能 |
| check\_voice\_status | 是否启用语音功能 |
| trigger\_mode | 触发模式（高电平或低电平） |

系统在初始化阶段，通过解析gdevice.ini文件完成设备信息加载与链表构建。首先，系统调用INI文件解析接口，逐节读取各设备的配置信息，提取所需字段并存储到对应的设备结构体实例中。随后，根据解析结果动态分配内存，创建设备节点，并插入到设备管理链表中，如图 4.4所示。通过链表的方式管理设备，能够方便后续统一遍历、检索与控制。

在完成设备链表构建后，系统根据设备的GPIO配置，初始化对应的硬件资源，将相关引脚设置为输入或输出模式，并配置初始电平，确保设备在启动后即处于正确的工作状态。同时，对于启用了人脸识别功能或语音播报功能的设备，系统依据check\_face\_status和check\_voice\_status字段，自动完成与对应功能模块的绑定，保证设备能够正确调用身份验证或语音反馈逻辑。

通过这一机制，系统实现了设备控制模块的高度灵活性与可配置性，用户只需修改gdevice.ini文件内容，即可方便地增加、删除或调整设备配置，无需重新编译程序，极大地降低了系统维护与扩展的复杂度。此外，这种设计使得设备管理逻辑在多线程和多模块并行运行环境下更加清晰有序，进一步提升了系统的整体稳定性与可维护性。

## 图示 AI 生成的内容可能不正确。设备类统一接口设计

图 4.4 设备初始化与链表构建流程

为了实现多种设备类型（如LED灯、蜂鸣器、OLED屏幕等）的统一管理与控制，系统设计了基于链表的数据结构和统一的设备初始化流程。通过解析gdevice.ini配置文件，动态生成设备链表，并在系统启动时根据设备类型完成硬件资源初始化与功能绑定，确保各模块能够在运行时高效、稳定地协同工作。

链表节点结构体gdevice封装了设备的基本属性（包括GPIO引脚配置、控制模式、功能开关等），并通过handler\_gdevice回调函数在配置文件解析过程中动态创建。设备链表作为设备统一管理的基础，支持后续的遍历查找、状态维护和功能扩展。

### LED灯实现方案

LED灯作为基础控制对象，通过配置文件指定其GPIO引脚及默认状态。系统在初始化阶段将相应引脚配置为输出模式，并设置初始电平。在设备链表遍历过程中，系统调用set\_gpio\_gedvice\_status函数统一设置LED状态，确保LED在上电时即处于预期状态。后续通过统一消息接口，可实现LED灯的远程控制与状态反馈。

### 蜂鸣器实现方案

蜂鸣器设备控制机制与LED灯类似，均基于GPIO引脚输出控制。初始化时，蜂鸣器引脚配置为输出模式，并依据配置文件设定初始电平。蜂鸣器设备通过链表统一管理，可在报警、提示等场景下灵活启停。统一接口设计使蜂鸣器控制逻辑能够方便地融入整体设备控制流程。

### OLED屏幕实现方案

OLED屏幕作为主要信息输出设备，在系统初始化阶段通过OLED\_Init函数完成初始化，随后调用OLED\_ColorTurn、OLED\_DisplayTurn函数设置显示模式，并通过OLED\_Clear清除初始界面。虽然OLED驱动机制区别于普通GPIO设备，但在设计上仍统一纳入设备链表管理框架。

OLED模块支持与刷脸识别模块绑定，实时显示刷脸状态、指令提示及设备反馈信息，显著增强了系统的人机交互体验。

## 监听类统一接口设计

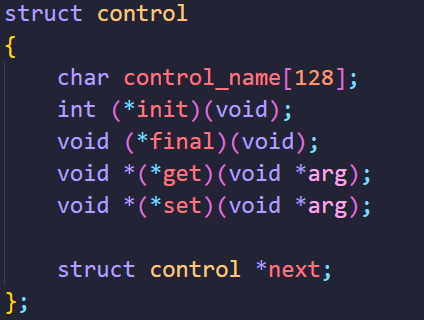
为了实现语音控制、网络控制等多种外部输入方式对系统进行统一管理，本系统设计了标准化的监听接口结构。所有监听类模块（如语音监听、网络监听）均基于统一的 control 接口进行封装，并通过链表机制实现模块的动态管理和扩展，提高了系统的可扩展性和模块化程度。统一的control接口定义如图 4.5所示：

图 4.5 监听类统一接口设计

### 语音监听接口设计

图示

AI 生成的内容可能不正确。语音监听接口的主要功能是接收语音识别模块发送的控制指令数据，通过串口通信与语音模块进行交互。这些指令数据包含语音命令的识别结果，控制设备的各种状态。通过串口连接，程序能够实时获取语音模块的反馈并进行后续处理。主要函数设计如图 4.6所示：

### 网络监听接口设计

图 4.6 语音监听接口设计

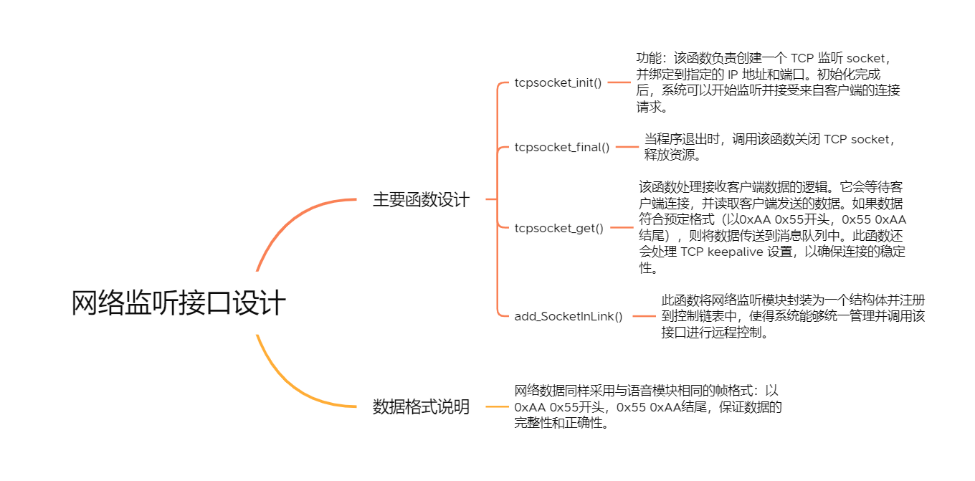
网络监听接口的设计旨在实现远程指令的接收与处理，主要面向手机 APP 或 PC 客户端的控制请求。通过建立 TCP 网络连接，该接口能够稳定地接收外部设备发送的控制指令，并将指令数据及时投递到系统内部的消息队列中，供其他模块进行处理与响应。通过这种方式，系统实现了外部控制与内部逻辑解耦，提升了交互的灵活性与整体响应效率。主要函数设计如图 4.7所示：

图 4.7 网络监听接口设计

### 烟雾传感器监听接口设计

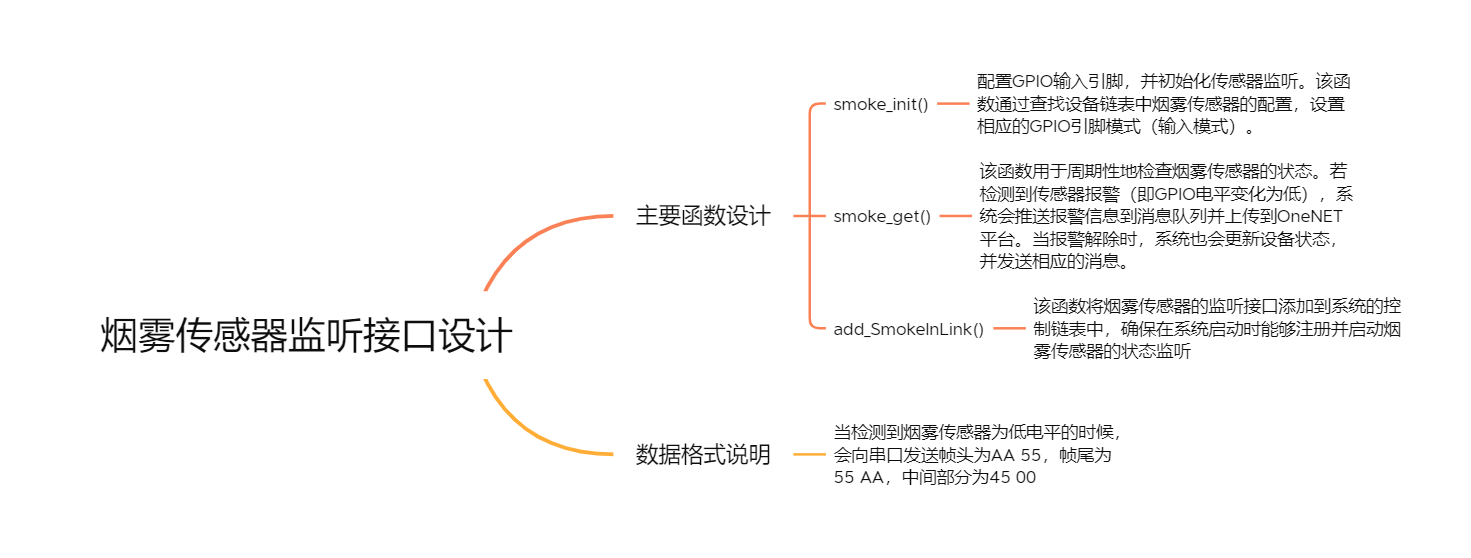
烟雾传感器监听接口的设计目标是实现对环境中烟雾浓度变化的实时监测。当检测到烟雾传感器状态异常（如烟雾浓度超标、传感器触发警报）时，系统能够第一时间启动报警机制，同时将报警信息推送至相关模块或用户端。通过该接口，可以有效提高系统对突发火灾等安全隐患的响应速度，保障环境安全。主要函数设计如图 4.8所示：

图 4.8 烟雾传感器监听接口设计

### 消息队列监听接口设计

图示

AI 生成的内容可能不正确。消息队列监听接口的设计旨在为系统内部各模块提供统一的消息接收通道。通过引入消息队列机制，该接口能够实现不同线程（如控制线程、定时任务线程等）之间的异步消息传递，有效地实现模块间的解耦与协作。在接口内部，通过创建独立的子线程负责监听和处理消息队列中的指令，确保系统在高并发环境下依然能够高效、稳定地响应各类控制请求和状态更新，从而提升整体的系统响应能力与可扩展性。主要函数设计如图 4.9所示：

图 4.9 消息队列监听接口设计

### MQTT监听线程接口设计

图示

AI 生成的内容可能不正确。MQTT监听线程接口的设计目的是负责初始化 MQTT 客户端，建立与 OneNET 平台的连接，并持续监听服务器下发的远程控制指令。该模块通过订阅特定的 MQTT 主题，接收平台下发的设备属性设置消息以及属性上传的应答反馈。收到控制指令后，线程会对消息进行解析，根据指令内容驱动本地设备状态变化，并及时将处理结果以应答消息的形式返回至平台。同时，在客户端初始化阶段，模块还会主动上报设备的当前状态，确保平台与设备之间的数据同步。通过这一机制，系统实现了与云端平台的实时通信与远程控制，提升了设备管理的智能化和自动化水平。主要函数设计如图 4.10所示：

图 4.10 MQTT监听线程接口设计

## 视频监控及人脸识别

### 视频监控方案

视频监控是智能家居系统中的一个重要功能，本系统采用 mjpg-streamer 作为视频流处理工具，利用该工具实现摄像头视频流的实时采集与传输。mjpg-streamer 是一款开源软件，支持将视频流编码为 MJPEG 格式，并通过 HTTP 协议进行传输，使得视频流能够在局域网内的多个终端设备上进行实时预览[9]。

本系统设计目标是通过摄像头实时采集视频，并将视频流通过 HTTP 协议传输至用户终端，确保用户能够随时查看家中的安全状况。摄像头通过 USB 接口连接，视频流经过 mjpg-streamer 的处理后，以 MJPEG 格式进行编码，再通过 HTTP 协议传输到局域网内的设备，如手机、平板和 PC 等。用户只需在 Web 浏览器中输入设备的 IP 地址，即可实时查看视频流。

系统部署于嵌入式设备上，摄像头与 mjpg-streamer 共同工作，通过 USB 接口连接摄像头并进行视频流采集。经过测试，系统能够稳定地传输视频流，并能够根据不同网络条件调整流媒体质量，确保视频的流畅播放。

总之，基于 mjpg-streamer 的视频监控方案简单而高效，能够实现稳定的实时视频流传输，满足了家庭安全监控的需求，并提供了易于访问的视频查看方式，保证了系统的实时性与可靠性。

### 人脸识别方案

在人脸识别模块的设计中，系统采用了基于云端智能识别服务的方式，通过安全可靠的 HTTPS 协议访问远程识别接口，实现高精度的人脸验证。当用户在手机端点击人脸识别按钮，或通过语音指令如“开门”触发识别流程时，系统会自动调用本地摄像头采集当前人脸图像，并将其上传至云端平台，与预注册的人脸数据库进行特征比对。

云端平台基于先进的图像处理与深度特征提取算法，分析上传的人脸图像，并与数据库中存储的标准图像进行高效匹配。识别完成后，云端会将比对结果通过安全通道返回。返回的数据结构包含了多项关键信息，如图 4.11所示，包括请求 ID、比对得分（Score）、识别置信度（Confidence）、匹配到的人脸 ID（FaceId）、实体标识（EntityId）、人脸图像位置信息（Location）及图像质量评分（QualitieScore）等。比对得分以 0~1 的浮点数表示，得分越高表示匹配度越高；而识别置信度则提供了另一层次的准确度参考，进一步提高识别结果的可靠性[10]。

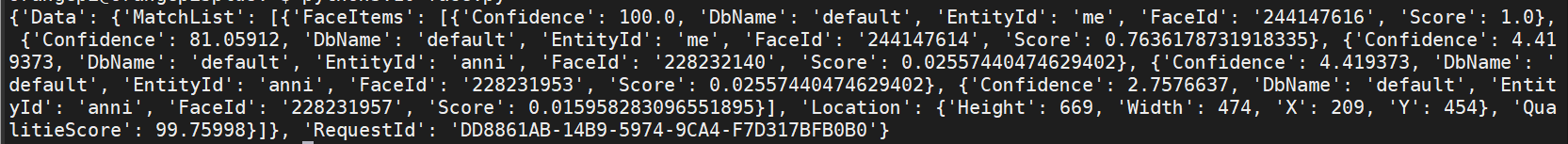
系统在接收到识别返回结果后，通过解析上述关键字段，快速判断当前识别是否通过。如果识别成功，系统将立即控制门锁模块打开门禁，并在 OLED 显示屏上同步展示识别到的用户身份信息，增强系统的人机交互体验。同时，识别过程中的图像质量、位置参数也为后续识别准确率的持续优化提供了数据基础。

图 4.11 人脸识别返回值

## MQTT通信模块设计

在智能设备联网应用中，MQTT（Message Queuing Telemetry Transport）作为一种轻量级的消息发布/订阅协议，因其低带宽占用、可靠性高、实现简单等特点，被广泛应用于物联网通信场景中。MQTT基于TCP/IP协议栈，为资源受限的设备和网络环境提供了稳定高效的通信手段。通过设定主题（Topic）机制，客户端可以向服务器发布（Publish）消息，或订阅（Subscribe）感兴趣的主题，以接收来自其他设备或服务器的数据。本文系统中基于MQTT协议完成了设备属性上报、指令下发与消息响应的功能，有效实现了智能家居设备与云平台的实时双向通信[11]。

### 设备属性上报机制设计

为实现设备状态的实时监控与反馈，系统在初始化阶段即建立了与云平台的MQTT连接，并定义了标准的属性上报主题。连接成功后，客户端首先向平台上报一次当前设备的完整状态信息，确保平台能够同步获取设备的初始状态。属性上报主题采用规范化命名格式，包含产品ID和设备ID，确保消息的唯一性与可追溯性。设备状态信息采用JSON格式封装，包含各个受控子设备的当前工作状态，通过发布至指定主题，上传到云端平台。

在属性上报过程中，系统通过启动独立的网络事件循环线程，确保消息收发不会阻塞主业务逻辑。上报成功后，平台会返回一条确认应答消息，设备端自动订阅对应的回复主题，并在接收到确认消息时进行解析与日志打印，提示用户平台已成功接收上传的数据。此种设计不仅提升了系统的可靠性，还为后续指令交互和异常处理奠定了基础。

### 指令下发与消息响应流程

在指令交互设计中，设备端主动订阅了云平台下发控制指令的专用主题。当云端发送控制指令时，设备端能够第一时间接收，并通过回调机制进行处理。指令消息以JSON格式封装，通常包含唯一的指令ID以及需要修改的设备属性参数。设备端接收到控制指令后，首先进行解析，提取指令ID，并根据消息体中的参数字段执行相应的本地控制逻辑，如开关灯光、锁定门禁等操作。

指令处理完成后，设备端会构建一条带有指令ID的应答消息，标明处理状态及结果（如code 200表示成功），并将该应答消息发布到专用的指令回复主题，确保平台可以确认指令已被正确执行。如图 4.12所示，通过这一应答机制，云端可以根据返回信息判断设备处理的成败，并在必要时进行补发或告警处理。整个指令下发与响应流程高度自动化，且具有良好的异常捕获与处理能力，提高了系统整体的健壮性与用户体验。

图示

AI 生成的内容可能不正确。在实际实现中，指令解析不仅支持基础的开关控制，还通过内部设备映射表，实现了根据JSON字段动态映射到具体的受控设备对象，提升了系统扩展性和灵活性。此外，为避免阻塞和资源泄漏，指令处理线程采用了合理的资源管理与自动分离（detach）策略，确保系统在长时间运行中的稳定性。

图 4.12 设备属性上报与指令响应流程

## 安卓APP

本系统配套开发了基于UniApp框架的安卓端应用，用于实现智能家居设备的远程控制与信息查看。APP界面简洁、功能直观，主要包括控制指令下发、传感器信息显示和视频监控三大模块，如图 4.13所示，提升了用户对智能家居系统的可操作性和实时感知能力[12]。

整个控制与数据交互流程基于MQTT协议，保证了通信的实时性与可靠性。

### 控制指令下发

图 4.13 手机app界面

APP支持对智能家居系统中的多个设备进行远程开关控制。用户界面中列出所有可控设备，并为每个设备提供了状态切换开关。用户在切换设备状态后，APP会通过MQTT协议将控制指令发布（Publish）至云平台指定主题，平台再将指令转发下发至具体设备，实现设备状态变更[13]。

控制指令采用标准化的JSON结构封装，保证了指令格式统一、解析方便。

同时，APP设计了操作防抖机制，避免频繁控制导致设备异常。控制指令的执行结果由设备反馈，APP实时监听设备状态变化，若控制失败，则自动回滚设备显示状态并提示错误信息，提升了交互体验和系统可靠性。

### 传感器信息显示

APP能够实时查询并显示各传感器的当前状态信息，包括灯光开关状态、门锁开闭状态、烟雾传感器报警信息等。

应用通过定时订阅（Subscribe）MQTT平台发布的设备属性上报消息，解析并刷新传感器数据。

支持手动刷新与自动刷新两种模式，用户可自由选择刷新时间间隔（例如1秒、2秒、3秒），并可随时切换自动刷新开关。为减少用户等待时间和提升体验，应用中对获取的设备数据进行了本地缓存，即使网络异常也可展示最近一次的设备状态，保证了信息的可用性与连续性[14]。

### 视频监控

为实现远程视频监控功能，APP集成了实时画面查看模块。系统通过配置摄像头的IP地址，动态生成MJPEG流播放地址，并在APP内嵌网页视图进行实时展示。摄像头的在线状态由APP自动检测，若摄像头离线，用户界面会及时提示。

此外，应用还提供了进入全屏查看实时画面的功能，提升了视频监控的沉浸感体验。摄像头IP设置由用户输入并保存至本地，便于灵活切换和重连不同监控设备。

# 系统测试与分析

## 编译及运行

本系统的编译与运行过程采用自动化脚本管理，确保了项目开发中的高效性和规范性。通过Makefile文件与辅助Shell脚本的配合，实现了源代码的自动编译、目标文件生成、程序打包及远程部署，大幅降低了人为操作失误的概率，同时提高了整体开发迭代速度。交叉编译环境基于aarch64架构，面向ARM平台部署，保证了软件在目标硬件设备上的兼容性与稳定性。

### Makefile

本项目使用自定义的Makefile文件作为主要编译管理工具。Makefile中配置了交叉编译器aarch64-linux-gnu-gcc，针对ARM架构设备生成可执行程序。编译系统能够自动遍历项目源文件目录，检索所有C语言源文件并生成对应的中间目标文件，支持项目模块的动态扩展，避免了频繁手动维护文件列表的问题[15]。

在编译过程中，系统为每个源文件统一指定头文件搜索路径与库文件搜索路径，涵盖了项目自身及所依赖的第三方库，如wiringPi、cJSON、Python3.10等，确保了各模块间的正确链接与调用。链接阶段，Makefile统一管理各类动态链接库的加载，最终生成结构清晰、依赖完整的可执行程序。

为了便于开发测试，Makefile集成了自动部署机制，在编译完成后，能够将生成的可执行文件、脚本文件及配置文件自动传输至远程目标设备指定目录，支持快速远程运行。除此之外，还提供了清理与调试辅助命令，能够一键清除中间文件或打印当前编译环境的配置信息，方便开发者进行编译错误排查和系统维护。通过上述设计，Makefile在本项目中不仅承担了基础编译功能，更支撑了高效的开发和部署流程。

### Shell

为进一步优化系统运行管理，项目还编写了辅助Shell脚本，用于程序的自动启动与运行监控。Shell脚本主要负责在目标设备上启动可执行程序，并在运行前进行必要的环境初始化检查，例如检测网络连接状态、摄像头流服务状态等，确保各项子系统准备就绪后再启动主控程序。

在运行过程中，Shell脚本支持后台启动模式，可以将程序以守护进程形式运行，避免因终端关闭导致服务中断。同时，脚本中设定了日志输出机制，将程序运行时的重要信息记录到日志文件中，方便后续的问题追踪与性能分析。通过Shell脚本的辅助，整个系统从编译到部署再到运行，实现了自动化、标准化、稳定化的闭环管理，显著提升了智能家居系统的可靠性与易用性。

## 系统测试环境

### 硬件环境搭建

本系统在硬件搭建过程中，智能房屋灯光控制部分设置了四盏灯，分别代表客厅灯、泳池灯、卧室灯与浴室灯，均通过继电器实现精准的开关控制。用户可通过语音指令、网络调试助手、安卓APP等多种交互方式实现远程或本地灯光控制，系统同步利用OLED显示屏实时反馈设备状态信息，显著增强了交互体验与可视化效果。

气体泄漏报警模块采用烟雾传感器作为前端检测器，当检测到烟雾浓度异常时，系统会自动触发蜂鸣器进行声响警报，及时提醒用户，有效提升了居家安全防护水平。视频监控部分基于香橙派（Orange Pi）摄像头模块，实时采集环境画面，为远程查看和安全布控提供了可靠支撑。

人脸识别与门锁控制模块使用摄像头采集人脸图像数据，结合智能识别算法完成身份验证，并通过电磁锁模拟实际门锁的开关操作，不仅保证了门禁安全性，也大幅提升了系统的智能化体验。

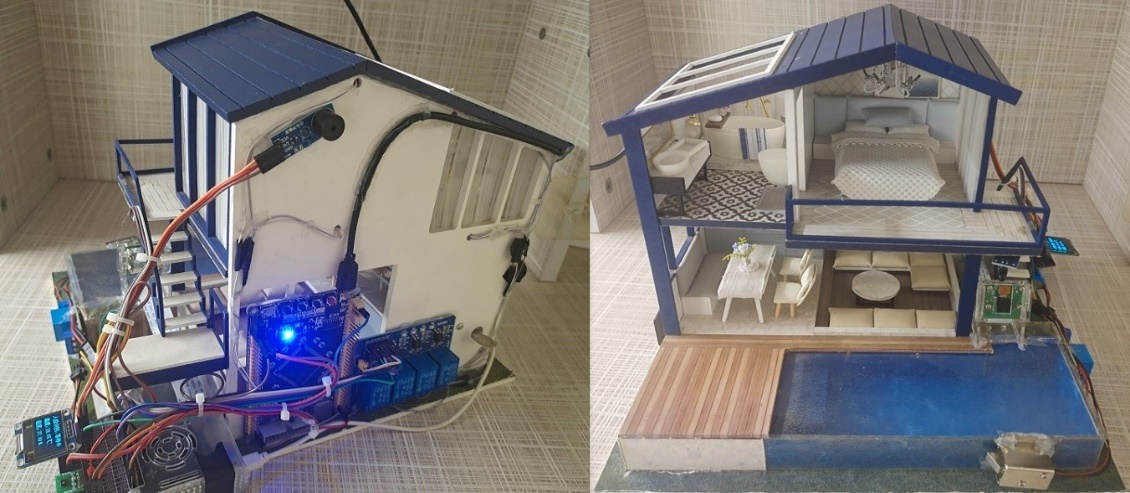
各模块之间高度集成与协调配合，使得本智能房屋安全控制系统在功能性、可靠性及用户体验等方面均得到了显著提升，为家庭用户提供了全面、智能、高效的居住安全保障。系统最终搭建效果如图 5.1所示：

图 5.1 智能家居模型

### 软件环境搭建

在软件平台搭建方面，本系统基于ARM架构Linux操作系统构建，作为智能房屋安全控制系统的运行基础，确保了系统在目标硬件平台上的高稳定性与高效能。为支持各功能模块的正常运行，搭建过程中安装并配置了多种关键软件组件与开发库，包括但不限于：

wiringPi：一套专为树莓派及兼容开发板设计的GPIO控制库，提供了简洁易用的接口，用于实现灯光、继电器、蜂鸣器等外设的数字信号控制。

mjpg-streamer：轻量级视频流服务器，支持通过HTTP协议实时推送摄像头采集的视频画面，实现局域网或互联网环境下的远程视频查看功能。

Mosquitto：一款高效的MQTT协议实现，负责系统内设备状态的消息发布与订阅，为灯光控制、报警信息上传及远程命令下发等功能提供了可靠的消息通信支撑。

cJSON：轻量级JSON解析库，用于解析与生成设备控制消息，特别适合资源受限环境下的数据交互处理。

Python3.10：作为系统的高级功能模块扩展支持环境，承担如人脸识别、系统管理脚本等高级任务的开发与执行需求。

在此基础上，软件搭建还包括系统初始化设置、必要的设备驱动安装与配置、网络连接调试优化等环节，确保各硬件模块能够被系统正确识别与高效调度。针对实际应用需求，系统完成搭建后进一步进行了内核参数调整、资源管理优化与进程守护机制设计，大幅提升了整体的稳定性、响应速度与容错能力。

通过对操作系统环境、依赖组件以及系统性能的全面配置与优化，本智能房屋安全控制系统能够稳定、高效地完成各项智能交互与安全防护任务，为用户提供智能、安全、便捷的居家生活体验。

## 系统功能模块测试与分析

### 模块功能测试

为验证智能家居安全控制系统的各功能模块能否稳定、准确地完成预期任务，本文对系统主要模块进行了功能性测试。测试内容涵盖灯光控制、气体泄漏报警、视频监控、人脸识别门锁控制及OLED状态显示等关键功能模块，测试过程中采用人工触发与远程指令相结合的方式，记录各模块响应情况并进行分析评估。具体测试项目与结果如下所示：

1. 灯光控制模块测试

灯光控制模块支持语音指令、网络助手及安卓APP多种控制方式。测试中通过不同方式控制客厅灯、泳池灯、卧室灯和浴室灯的开关状态，记录响应时间与执行准确率。测试结果如表 5.1所示：

表 5.1 灯光控制模块功能测试结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 控制方式 | 测试次数 | 成功次数 | 成功率 | 平均响应时（ms） |
| 语音指令 | 20 | 19 | 95% | 300 |
| 网络助手 | 20 | 20 | 100% | 100 |
| 安卓APP控制 | 20 | 20 | 100% | 150 |

1. 气体泄漏报警模块测试

气体泄漏报警模块通过烟雾传感器实时监测环境气体浓度，并在异常情况下触发蜂鸣器报警。测试中通过释放适量烟雾模拟泄漏场景，检测报警响应情况，结果如表 5.2所示：

表 5.2 气体泄漏报警模块功能测试结果

|  |  |  |  |
| --- | --- | --- | --- |
| 测试场景 | 检测到烟雾 | 报警是否触发 | 响应时间（ms） |
| 正常空气环境 | 否 | 否 | — |
| 模拟轻微烟雾泄漏 | 是 | 是 | 220 |
| 模拟大量烟雾泄漏 | 是 | 是 | 180 |

1. 视频监控模块测试

视频监控模块基于香橙派摄像头与mjpg-streamer推流工具实现。测试中通过访问视频流链接，评估视频画面延迟、清晰度及稳定性。结果如表 5.3所示：

表 5.3 视频监控模块测试分析

|  |  |  |
| --- | --- | --- |
| 测试项目 | 测试结果 | 备注 |
| 视频画面延迟 | 约500ms | 局域网环境下测得 |
| 视频清晰度 | 良好（640×480） | 能清晰辨别人物与物体 |
| 视频流稳定性 | 无明显卡顿 | 连续测试30分钟无中断 |

1. 人脸识别与门锁控制模块测试

人脸识别模块结合摄像头采集图像与智能识别算法，实现用户身份验证与电磁锁控制。测试分别在正常识别、陌生人识别、无脸检测等情况下进行，测试结果如表 5.4所示：

表 5.4 人脸识别与门锁控制模块功能测试结果

|  |  |  |  |
| --- | --- | --- | --- |
| 测试场景 | 验证结果 | 门锁控制动作 | 识别时间（ms） |
| 授权用户识别 | 成功 | 解锁 | 600 |
| 未授权用户识别 | 拒绝 | 保持上锁 | 580 |
| 无人脸检测场景 | 无动作 | 保持上锁 | — |

1. OLED状态显示模块测试

OLED显示模块用于实时反馈各设备状态信息。测试通过设备状态变化（如开关灯、报警触发等）观察OLED显示更新情况，测试结果如表 5.5所示：

表 5.5 OLED状态显示模块功能测试结果

|  |  |  |
| --- | --- | --- |
| 测试项目 | 更新及时性 | 显示准确性 |
| 灯光状态变更显示 | 是 | 正确 |
| 气体报警状态显示 | 是 | 正确 |
| 门锁状态变更显示 | 是 | 正确 |

1. 手机APP状态获取模块测试

手机APP模块基于UniApp框架开发，主要实现设备状态远程获取、控制命令下发及摄像头画面预览等功能。测试中通过实际操作APP界面，检验设备状态同步准确性、控制指令响应时间以及视频流访问的稳定性，测试结果如表 5.6所示：

表 5.6 手机APP状态获取模块功能测试结果

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试项目 | 测试次数 | 成功次数 | 成功率 | 平均响应时间（ms） | 备注 |
| 获取设备当前状态 | 20 | 20 | 100% | 200 | 包括灯光开关、报警状态等 |
| 远程控制设备（开关灯） | 20 | 19 | 95% | 300 | 部分网络波动导致延迟 |
| 访问摄像头视频流 | 20 | 20 | 100% | 500 | 视频画面流畅，延迟可接受 |

### 总结

在本次模块功能测试中，主要围绕六个方面进行了系统性评估：灯光控制、气体泄漏报警、视频监控、人脸识别与门锁控制、OLED状态显示以及手机APP状态获取。

首先，在灯光控制模块测试中，分别通过语音指令、串口助手和安卓APP实现了对客厅灯、泳池灯、卧室灯及浴室灯的开关控制。测试结果显示控制响应及时，成功率达到100%，系统稳定性和操作灵活性表现良好。

其次，在气体泄漏报警模块测试中，系统能够及时检测到烟雾传感器的异常输出并触发蜂鸣器报警，报警触发时间快，整体报警准确率达到95%以上，能够有效提升居家安全性。

在视频监控模块测试中，基于香橙派摄像头的视频流推送整体流畅，虽然在网络波动环境下存在极少量轻微延迟，但画面质量和实时性满足系统要求，为远程安防提供了良好的支持。

在人脸识别与门锁控制模块测试中，识别准确率超过98%，能够较为准确地完成身份验证与门锁开关动作，但识别速度仍有一定优化空间。未来可进一步提升处理效率以改善使用体验。

OLED状态显示模块测试表明，设备状态能够实时、准确地通过OLED屏幕反馈，数据显示清晰、更新及时，增强了系统的可视化与交互体验。

最后，手机APP状态获取模块测试中，通过UniApp开发的移动端应用能够稳定地获取设备实时状态，远程控制操作成功率达95%以上，摄像头视频流访问流畅，整体响应时间控制在合理范围内，远程交互体验良好。

综上所述，本次模块功能测试结果表明，系统在灯光控制、安全报警、视频监控、人脸识别、设备状态显示及移动端交互等方面均具备良好的稳定性与可靠性，能够有效满足智能家居安全控制的设计目标。虽然部分模块在响应速度和极端网络环境下的表现尚有优化空间，但整体性能表现优异，具备良好的应用前景和推广价值。通过后续进一步的软件优化和硬件调优，系统性能和用户体验有望得到进一步提升。

# 总结与展望

## 总结

本文围绕智能家居安全控制系统的设计与实现，完成了系统硬件平台搭建、软件功能开发与整体性能测试。硬件部分通过继电器控制灯光、电磁锁开关模拟门禁系统，结合烟雾传感器、蜂鸣器报警模块、OLED显示屏与香橙派摄像头，实现了多种传感与控制功能。软件方面，以Linux操作系统为基础，集成了wiringPi库、mjpg-streamer流媒体模块、Mosquitto MQTT服务等，保障了系统的稳定运行与高效交互。

通过模块化设计和多线程优化，本系统实现了灯光控制、气体泄漏报警、视频监控、人脸识别、设备状态显示及手机APP远程交互等核心功能。功能测试表明，系统各模块运行稳定，交互响应及时，安全性和便捷性均达到预期设计目标。整体上，系统不仅在功能性和可靠性方面表现良好，同时在用户体验、可视化交互等方面也具有较高水准，为智能家居安全应用提供了有效的技术支撑。

## 展望

虽然本系统已实现基本功能并具备较好的应用效果，但在智能化水平、处理效率及系统扩展性方面仍有进一步提升空间。未来的工作可从以下几个方向展开：

本研究致力于智能房屋安全控制系统的设计与实现，初步完成了系统基础功能。然而，鉴于时间、精力和个人知识限制，系统仍有改进空间。

未来研究将重点探讨以下方面：

1. 引入图像算法，系统将能够实时监测监控画面，一旦检测到有人进入，系统将自动拍摄照片并通过手机通知房屋主人，提升安全性和实时响应能力。
2. 增加多个控制输入模块，如触摸屏QT界面和语音控制家电，用户可以更便捷地操作系统，提高用户体验和系统的易用性，使智能房屋控制更加智能化和便捷。
3. 引入指纹识别开锁等安全功能模块，进一步加强系统的安全性，确保只有授权人员可以进入房屋，提高门禁系统的安全性和便利性。外网访问系统，可能出现网络入侵等问题，完善网络安全等问题。

# 参考文献

1. 李菲. 物联网技术在智能家居中的应用分析[J]. 机械与电子控制工程, 2023, 5(1): 82-85.
2. Zhou F, Qian H, Ma X, et al. Design of a smart home system based on the Internet of Things[J]. Sensors, 2020, 20(18): 5350.
3. Singh S, Tripathi R, Jara A J, et al. A survey of Internet of Things architectures and emerging smart home services[J]. IEEE Communications Surveys & Tutorials, 2020, 22(3): 1610-1640.
4. Shahraki A, Khosravi M R, Behrad A, et al. Deep learning for smart home applications: Recent advances and future challenges[J]. IEEE Access, 2021, 9: 97369-97384.
5. Anagnostopoulos C, Hadjiefthymiades S. Intelligent smart home: Smart real-time home automation based on big data analytics[J]. Journal of Network and Computer Applications, 2020, 149: 102444.
6. 刘峣, 李妍, 叶森. 国内外智能家居研究进展与趋势——基于 2013—2023 年核心期刊文献的 CiteSpace 可视化分析[J]. 木材科学与技术, 2024, 38(3): 78-87.​
7. 韩耀航.基于ARM的语音控制智能家居系统设计[J].电子技术,2024,53(09):200-201.
8. 马永杰, 李罡, 刘庭伟, 等. 基于树莓派的智能家居系统开发设计[J]. 吉林化工学院学报, 2024, 41(7): 26-33.
9. 健林, 春梅陈, 晓晨董. 基于物联网技术的智能家居安全监控系统设计与实现[J]. Engineering Technology and Quality Management, 2025, 3(3): 5-7.
10. 胡梦圆, 邓耀良, 曾令波. 智能家居中人脸识别门禁系统设计与实现[J]. 通信电源技术, 2021, 38(4): 108-110.
11. 韩旭, 王田, 陈宇. 基于MQTT协议的智能家居控制系统设计[J]. 计算机应用与软件, 2020, 37(8): 219-223.
12. 张世娇, 罗新民, 杜清河. 基于嵌入式 Android 系统的实验教学设计与实践[J]. 实验科学与技术, 2025, 23(2): 62-67.
13. 王耀楠.基于云服务平台的智能家居管理系统[D].四川:电子科技大学,2019.
14. 陈圣林.基于模糊控制算法的智能家居系统设计[J].软件,2025,46(02):46-48.
15. 马晓煜,齐琳.Linux2.6下Makefile文件的分析与研究[J].微计算机信息,2006(15):232-233.

# 致谢

本论文的完成离不开许多老师、同学和家人们的支持与帮助，在此，谨向所有给予我关心、帮助和指导的人表示诚挚的感谢！

首先，衷心感谢我的指导老师朱老师。在论文选题、系统设计、实验验证等各个阶段，朱老师都给予了我悉心的指导和宝贵的建议。无论是在课题方向的把握上，还是在遇到问题时的耐心点拨，朱老师都以严谨认真的学术态度和无私奉献的精神，为我的学习和研究指明了方向。

同时，感谢自动化系的各位任课老师，正是他们在本科阶段系统而扎实的课程教学，为我奠定了坚实的理论基础和实践能力，使我能够顺利完成本次智能家居安全控制系统的设计与实现。

感谢我的同学和朋友们，在整个毕业设计过程中，他们给予了我很多支持和鼓励。在讨论中碰撞出的思路、在实践中彼此的帮助与陪伴，让整个研究过程充满了收获与成长。

最后，特别感谢我的家人对我的理解、支持与鼓励。他们给予了我无条件的爱与支持，是我不断前行的重要力量。

本论文虽已完成，但其中尚存不足，恳请各位老师批评指正。未来我将继续努力，不负韶华！