中图分类号：TP3

论 文 编 号 ：10006ZF1821334

北京航空航天大學

专业硕士学位论文

基于语音的家居控制系统的设计与实现

作 者 姓 名　　张 加 杰

专 业 名 称　　软件工程

指 导 教 师　　王 丽 华

培 养 学 院　　软 件 学 院

The Design and Implementation of Home Control System Based on Voice

A Dissertation Submitted for the Degree of Master

Candidate：Zhang Jiajie

Supervisor：Prof. Wang Lihua

School of Software

Beihang University, Beijing, China

中图分类号：TP3

论 文 编 号 ：10006ZF1821334

硕 士 学 位 论 文

基于语音的家居控制系统的设计与实现

作者姓名	张加杰	申请学位级别	工程硕士
指导教师姓名	王丽华	职　　称	教　授
专业名称	软件工程	研究方向	人工智能
学习时间自	2018 年 9 月 10 日	起至	2021 年 12 月 11 日止
论文提交日期	2021 年 11 月 22 日	论文答辩日期	2021 年 12 月 11 日
学位授予单位	北京航空航天大学	学位授予日期	年　月　日

关于学位论文的独创性声明

本人郑重声明：所呈交的论文是本人在指导教师指导下独立进行研究工作所取得的成果，论文中有关资料和数据是实事求是的。尽我所知，除文中已经加以标注和致谢外，本论文不包含其他人已经发表或撰写的研究成果，也不包含本人或他人为获得北京航空航天大学或其它教育机构的学位或学历证书而使用过的材料。与我一同工作的同志对研究所做的任何贡献均已在论文中作出了明确的说明。

若有不实之处，本人愿意承担相关法律责任。

学位论文作者签名：_____ 日期： 年 月 日

学位论文使用授权书

本人完全同意北京航空航天大学有权使用本学位论文（包括但不限于其印刷版和电子版），使用方式包括但不限于：保留学位论文，按规定向国家有关部门（机构）送交学位论文，以学术交流为目的赠送和交换学位论文，允许学位论文被查阅、借阅和复印，将学位论文的全部或部分内容编入有关数据库进行检索，采用影印、缩印或其他复制手段保存学位论文。

保密学位论文在解密后的使用授权同上。

学位论文作者签名：_____ 日期： 年 月 日

指导教师签名：_____ 日期： 年 月 日

摘　要

随着移动互联网的成熟，智能化已经成为新的发展趋势和主要需求。在大规模数据和大规模算力的驱使下，人工智能技术进入了智能化引领的发展阶段，人工智能技术正在越来越广泛地应用在移动互联网领域。语音识别技术作为人机交流接口的关键技术也在根本性地改变人们使用互联网的方式，当前各互联网厂商推出的人工智能音箱是人工智能语音识别与智能家居结合的典型应用，不仅具有语音控制的能力还能控制智能家居设备。简单高效的控制智能的语音对话为我们提供了便利同时对家居控制不够完善，强依赖网络等问题也凸显出来。本文目标是设计一个基于语音的控制系统，可以控制尽量多家居设备并且具有一定的离线控制能力。

本文基于当前市面上智能音箱语音交互和家居控制功能，针对上述问题的解决进行设计实现。通过需求分析和整体架构的调研，确定了以嵌入式系统为基础的软硬件相结合的方式实现。硬件上在保证算力和多种外设接口的需求下，确定了开发板的类型，开发板需要同时支持红外、WiFi、蓝牙等外围接口，操作系统选用 Linux。软件层面语音唤醒采用 Snowboy 工具包进行热词定制，语音识别采用 Kaldi 作为离线识别解决方案，同时接入了百度开放平台支持在线识别。硬件接口层和控制层使用 C 语言实现，应用层使用 Python 和 C++语言进行处理。针对上述模块深入分析后，进行了详细的设计和实现，最后针对本系统进行全面测试和分析评估，最终的测试结果可以证明该系统满足需求。

该系统在落地实现后可以控制家电产品，包括一些非智能产品，本系统安装在家中实现了电视空调电灯冰箱热水器等设备的控制，切实做到语音控制解放双手，并且在没有外网的情况仍然可以使用。

关键词：语音识别，家居设备，嵌入式系统

Abstract

With the maturity of the mobile Internet, intelligence has become a new development trend and main demand. Driven by large-scale data and large-scale computing power, artificial intelligence technology has entered a stage of development led by intelligence, and artificial intelligence technology is being more and more widely used in the field of mobile Internet. Speech recognition technology, as the key technology of human-machine communication interface, is also fundamentally changing the way people use the Internet. Currently, the artificial intelligence speakers introduced by various Internet manufacturers are a typical application of artificial intelligence speech recognition combined with smart home. It not only has the ability of voice control, but also controls most smart home devices. Simple and efficient control, intelligent voice dialogue, etc., provide us with convenience. At the same time, the problem of insufficient home control and strong dependence on the network has also been highlighted. The goal of this article is to control all home equipment based on voice and ensure a certain offline control ability.

This article combines the capabilities of smart speakers and smart homes on the market to design and improve the system for the above problems. Through demand analysis and overall architecture research, we first determined the embedded system that combines software and hardware as the carrier. On the hardware, under the requirements of ensuring computing power and supporting all types of homes, we determined the selection of the development board and the needs of the development board. At the same time, it supports infrared, WiFi, Bluetooth and other peripheral interfaces, and the software system uses Linux. Voice wakeup uses Snowboy toolkit for hot word customization, voice recognition uses Kaldi as an offline recognition solution, and at the same time accesses Baidu's open platform to support online recognition. The hardware interface layer and control layer are implemented in C language, and the application layer is processed in Python and C++. After in-depth analysis of the above modules, detailed design and implementation were carried out. Finally, a

comprehensive test, analysis and evaluation was carried out for the system. The final test results can prove that the system meets the requirements.

After the system is implemented, it can control home appliances, including some non-intelligent products. The system is installed in the home to realize the control of TV, air conditioner, light, refrigerator, water heater and other equipment, and it can actually achieve voice control to liberate hands, and it can still be used when there is no external network.

Key words: Speech recognition, Home equipment, Embedded systems

目　　录

图清单

表清单

第一章 绪论

1.1 研究背景及意义

本课题主要是受当前流行的智能语音对话音箱启发，结合我当前从事的公司相关项目，我相继在百度智能生活事业群组小度智能音箱业务部和语音技术部门工作，主要工作内容是小度智能音箱的开发，先后开发了小度智能音箱的语音软件开发包，OTA（Over the Air）无线空中升级模块，蓝牙配网，底层 Linux 系统和驱动等功能。当前智能音箱普遍重云轻端，控制功能薄弱，强依赖网络，将主要算法如 ASR（Automatic Speech Recognition），NLP（Natural Language Processing）等放在云端，音箱端仅负责数据采集和语音合成 TTS（Text To Speech）播放，端云之间通过一套自研协议进行通信，基于这种架构形式可以有效削减硬件成本，相应的音箱端可实现的功能受到限制，无法发挥嵌入式系统控制优势，音箱端可新增功能受限，降低了灵活性。基于此，设计一款基于深度学习的嵌入式高性能语音控制系统非常有必要，应用场景可涵盖手机，车载，智能家居，行业智能终端等，该系统可将唤醒，识别和控制功能全部放在音箱本地系统中，可实现离线语音唤醒和识别功能，并能极大程度增加系统的控制功能，对该控制系统进行有效设计和实现是本课题的主要内容。

1.1.1 研究背景

在移动互联网上一轮的发展周期中，最具代表性的 PC 互联网时代，人们利用键盘、鼠标和显示器来获取信息，而且可以很好的解决人机交互问题。随着移动互联网的新一轮发展周期，语音技术和智能家居的进一步发展，人们通过移动互联网，可以随时随地的利用各种移动终端设备访问网络。但是伴随着移动互联网的快速发展，原来获取信息和人机的沟通的方式已经不能得到充分满足，社会向着信息化和智能化方向快速的发展，新的一种人机交互方式产生了，人类一直以来通过语音进行交流，是人类最自然最便捷的信息获取和沟通的方式，人机通过语音交互的方式引起了业界的广泛关注。另外一个方面，随着人工智能领域的发展，机器学习、深度学习技术的突破，大数据技术以及自然语言理解语音合成等能力的提升，进一步带动了一波产业热潮。国内包括继科大讯飞、捷通华声之后，互联网巨头像阿里，百度，腾讯等都在往智能语音领域发力。具有代表

性的基于语音交互的智能音箱已经走入千家万户，越来越多的人们习惯于用口语化的指令进行查询和接收信息，如播放音乐和智能音箱聊天对话，通过语音进行查询天气情况，还可以设置日程提醒、预订机票和酒店网上购物等非常实用的生活服务和功能[1]。伴随着智能家居产业的蓬勃发展，制造业，家电设备，安防，工控等方面的成熟，带动主要制造商雨后春笋般推陈出新了许多智能家居产品，通过无线传感技术来增加使用体验和舒适度。对于大众用户来说，随着消费需求逐渐升级成熟，通过语音进行集中控制更多的智能产品已日渐成为主流趋势。

1.1.2 研究意义

本选题在智能音箱普遍重云轻端的大背景下，实现一套在线离线一体的语音控制系统，相较于市面语音智能音箱具有更强力的控制功能，即使在一些没有网络的场景下仍然能离线识别控制，稳定性强，识别率高，适应场景能力更强。另一方面，基于算法和工程的完整系统可以回馈算法，使工程和算法互补互益，更好的提升算法能力和效率。基于当前工作项目，目前主要实现工程化及算法周边内容，对一些优秀的开源项目进行调研，利用算法和深度学习的强大能力进行深入的设计和实现，基于此选题，可以更深入的学习深度学习在智能语音交互系统中的实现，通过自己的思考和实现可以巩固自己所学和工作任务，更希望能在这个领域做出自己的一份贡献。

1.2 国内外相关研究现状及对比分析

1.2.1 国外研究现状

目前国外语音技术与智能家居的结合产物最具代表性的也属智能语音音箱，而且已经被广泛的推广，如 Amazon 的 Echo，谷歌的 GoogleHome，苹果的 HomePod 等均已发布上市，备受消费者欢迎。亚马逊在 2014 首次发布基于语音控制的智能音箱 Echo，该产品让人眼前一亮，亚马逊将其定位为一位家庭助理，赋予其语音交互和灵活处理语音需求的能力，它装载了一个名为 Alexa 的语音交互系统。这个系统在后续的迭代中能力不断增强，又赋予了它家居控制的能力，不仅可以通过手机进行被动控制，还可以通过语音进行高级的外部智能家居设备进行操控[2]。不止扮演了家庭助理的功能，它还可以像一个知心的朋友一样随时随地的跟人沟通交流，甚至在你无聊的时候给你讲笑话。另一方面，不管是购物还是家居控制，包括一些信息查询等功能都是基于语音的交互来

完成，这样全新的交互方式已经突破人们的传统观念。语音交互一直处于实验阶段，一直到 Echo 的成功发布和售卖，也同样带动了语音交互的发展。随后各个互联网巨头和厂家开始推出自己的造型各异的语音交互机器人。这些机器人更加注重家居场景，而从手机端到智能家居控制的垂直场景的过度，最根本的是各大厂商从技术角度突破并解决了远场拾音与多种场景下的语义理解问题，利用先进的声学算法和硬件创新使得语音交互成为可能，并且在家庭场景中和智能家居结合，面向真实客户群进入千万家庭，为人们生活提供了更多乐趣和便利。

1.2.2 国内研究现状

目前语音交互和家居控制在国内发展日益蓬勃，当前国内各厂家的普遍具有自己的语音交互和控制能力。家居生态中首屈一指的是小米科技，其推出米家家居生态做的比较全面，而家居生态中以小爱音箱为核心，打造基于语音的家居控制生态。科大讯飞联合京东在 2015 年首次推出国内的第一款基于语音交互的智能音箱“叮咚”，其在功能和形态上都比较类似亚马逊的 Echo。虽然比国外稍微晚了一些，但国内的发展势头非常强劲，从 2016 年开始，语音交互和家居市场突然爆发，各大厂家均发布了自己的带有家居控制的语音交互音箱，而且在很短的时间内各个厂家的各类语音和家居交互产品的版本都会进行迭代，不断优化用户体验，而且效果都会出现非常大的提升，产品的落地和用户的使用和反馈推动了语音交互和家居系统的不断完善，使语音交互技术链条越来越成熟。而家居方面各个厂家开始向智能方向转型。作为语音交互的基础底座，语音前端信号处理是一项极具挑战技术难点，声智科技和科大讯飞在这方面首先发力，并持续改进这项技术。随着市面上新产品的落地和数量的增加，真实场景下的用户数据和音频被不断的生产和收集，这对于互联网大规模的数据云存储能力得到了用武之地，可以轻松存储终端产品在使用过程中上报上来的数据进行收集并加以筛选和利用[3]。反过来大量的数据正是人工智能的前提，机器学习和深度学习在海量数据的加持下不断训练优化模型，目前小米、阿里、百度等语音识别准确率高达 97%，各互联网厂家在智能语音的基础上不断拓展家居生态，海尔推出了海尔智家，小米推出的小米生态链，阿里基于天猫精灵的智能家居等。

1.2.3 对比分析

全球范围内智能音箱销量都在持续增长，带动了智能家居生态的发展，据报告显示，以智能家居为核心的语音控制音箱在全球市场的规模达到 4000 万。中国已经成为仅次于美国的全球第二大智能产品消费市场，其中国产的智能音箱更是占据了大头。随着人工智能语音技术的不断突破，国外的互联网公司如亚马逊，谷歌，国内的互联网公司如百度，阿里巴巴，小米等均推出了自己特色的基于语音的智能家居生态体系，如亚马逊的 Amazon Robot，百度的语音交互系统 DuerOS 和小度智能家居生态，小米的小爱同学与小米生态链等都非常强劲，支持的家居种类也非常全面。

1.3 研究目标及研究内容

1.3.1 研究目标

本论文的研究目标是基于语音相关技术实现一套由语音进行家居控制的智能控制系统，探索最前沿的技术和解决方案，并加以研究和优化，并增进工程落地能力。具体的研究目标如下：

1）嵌入式开发板和控制系统的选型

2）语音关键词唤醒功能设计实现

3）语音识别的设计实现

4）控制系统的设计实现

5）语音合成功能设计实现

1.3.2 研究内容

本基于语音的家居控制系统的研究内容根据研究目标可分为具体下列几个方面:

1）嵌入式开发板和系统的选型

依据市面已有的智能音箱进行开发板对比，确定开发板型号，结构设计上对麦克风采集到的信号的影响，包括结构震动，腔体效应，导音孔，音腔谐振自激放大，现在大多数产品同时有麦克风和扬声器，而扬声器放出的声音会被麦克风采集到，回采（硬件或者软件实现）到的扬声器信号对最终的声学回声消除 AEC(Acoustic Echo Cancellation)影响比较大。另外操作系统可选择 Linux 系统来搭建开发和执行环境。

2）语音唤醒功能

对于智能产品的用户来说,唤醒就是语音交互的第一入口,唤醒效果的好坏直接影响到用户的第一体验。用户使用时通过指定唤醒词将系统唤醒，如“小贾小贾”。该功能依托音频的声学算法和深度学习，对声学算法 AEC 进行研究和调优，设计关键词唤醒训练模型并进行调参训练。

3）语音识别

用户唤醒系统后，发起语音请求控制家居设备，如“打开电灯”，系统收到指令音频，经过一系列声学算法，最后送入识别引擎转成文字。该模块同唤醒模块一样，需要研究声学算法并进行语音识别模型训练。经过声学算法对音频进行降噪增强，在语音识别中声学算法+深度学习模型具有非常好效果，这一部分主要研究语音识别相关解决方案和开源工具，以及如何进行音频处理和并根据语音识别模型搭建有效的识别引擎。

4）控制系统

语音识别结果为文本内容，通过字典匹配查找待执行指令，匹配成功后通过局域网通信协议将指令发与外设，电灯成功打开。这一部分需要研究局域网通信协议如 MQTT 或蓝牙 BLE，红外控制等，需要在开发板选型时考虑芯片是否支持相关功能,电灯打开后发送打开成功指令给系统，系统通过语音合成合成打开成功音频并播放，该模块可采用默认音频+简易语音合成合成引擎结合实现。

5）语音合成

许多互联网公司都提供了 AI 开发平台，平台提供了语音合成功能，本次研究会体验该功能并成功应用到当前系统。

1.4 本文组织结构

第一章是绪论。绪论部分主要讲述了我的论文基于语音的家居控制系统需要进行的研究背景和意义的探索分析，梳理并分析市面上控制系统的发展和现有技术成果。

第二章是系统需求分析。本章主要阐述了围绕基于语音的家居控制系统的核心功能进行相关的需求分析，并将系统需求从功能角度划分为功能性需求和非功能性需求，本系统的功能性需求比较注重该系统支持的功能和承载该功能所需要的底层功能性需求，最后说明非功能性需求，该需求强调系统性能和系统的实现效率等，与功能性需求相辅相成。

第三章是系统总体设计。本章从系统架构角度切入，对系统的功能结构，网络拓扑，系统接口等进行了设计。

第四章是系统详细设计与实现。本章对系统各个模块进行了详细的设计并进行开发和落地实现。另外本章还对语音唤醒，语音识别等关键技术进行了分析并给出了解决方案。

第五章是系统测试分析。本章从系统的角度出发，设计了比较全面的测试用例和测试方法，对系统功能和非功能需求进行了摸底和测试，测试结果符合预期。是系统设计后对系统整体实现完整度和能力的评估，是比较重要的一环。

第六章是总结与展望。本章对整个系统从需求到设计再到实现进行了整体总结，从总结内容得到后续需要继续完善的细节，并做出了具体的规划。对后续的系统落地和商业化进行了构思和展望。

1.5 本章小结

本章首先介绍了基于语音的家居控制系统需要进行的意义背景等方面的研究，根据当前的技术和产品方面的现状进行了国内和国外的对比，另外对于本论文的研究目标进行了设定，基于研究目标，设计了本系统在研究过程中需要实现的内容。本章是整个论文的基础，对后续的研究和设计工作具有指导意义。

第二章 系统需求分析

2.1 现状分析

目前市面上的音箱具有重云轻端的特点，往往丢失了嵌入式系统本身灵活和控制的特点。目前家居虽然向着智能的方向发展，支持移动互联网的远程控制[4]。但仍有大量的家居设备是按键控制，好一点的是红外或蓝牙，但相对功能单一，无法通过一个智能的语音中控进行统一控制。另一方面，智能语音的代表智能音箱，都是在有网络的环境下才能使用，在弱网或没有网的情况下基本罢工。

针对上述问题，本文从各个角度考虑，目标是设计一套能尽量多覆盖家居家电设备，对其进行控制，并且能够在有线离线环境下都能正常工作的家居控制系统。该系统具有如下特点：

1）尽量多的支持控制家居家电，包括已经支持 WiFi，红外，蓝牙的偏智能家居类产品，以及不支持无线控制的家居，可以使用智能插座来控制开关。

2）可以胜任有网和无网环境，在弱网或没有网络的情况下仍然支持最基本的开关和简单控制指令。在有网的条件下可以支持更复杂的指令，并且可以做到网络实时监测，动态切换有网无网模式。

2.2 功能性需求分析

基于语音的家居控制系统功能性需求比较明确，示例如图 1 所示，用户通过“小贾小贾”唤醒系统，语音指令是“打开空调”，系统将识别该语音并分析用户意图，发送红外控制指令给空调，空调打开后，系统会通过扬声器播放“空调已打开”的提示术语。

由系统用例图，该系统功能性需求分为如下几个部分，各部分内容如下：

1）音频处理需求分析

音频处理模块包括麦克风音频数据采集功能和音频降噪处理功能。当系统未被唤醒时应处于静默状态，麦克风则应处于持续收音状态，系统会开启音频采集功能，采集的音频有三路，双路麦克风音频和一路参考信号，该部分需要注意硬件上要支持参考路信

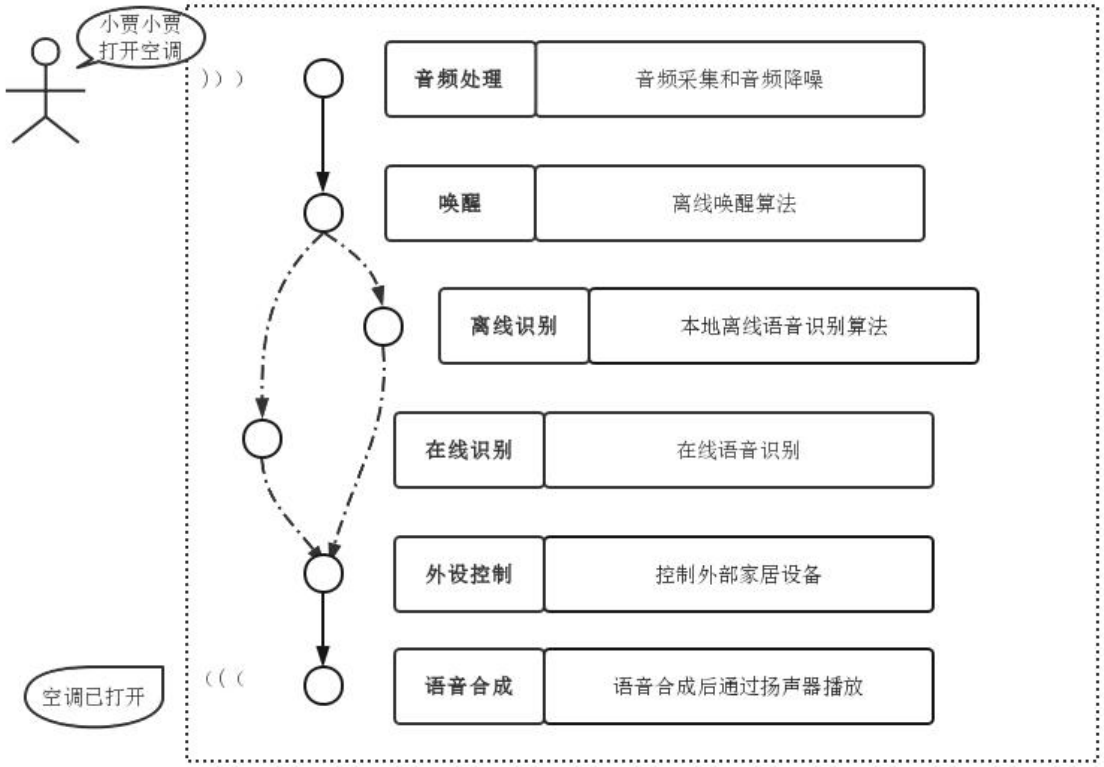

图 1　基于语音的家居控制系统功能示意图

号回采，否则做不了 AEC 降噪，无法在有内噪的情况下唤醒系统[5]。三路信号通过降噪算法进行降噪处理，处理后的音频送入离线唤醒引擎。用例如图 2 所示。

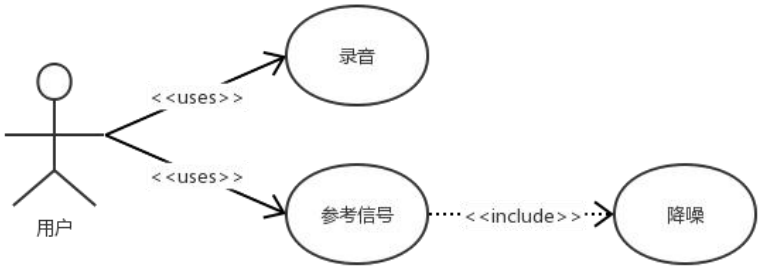

图 2　音频处理用例图

2）语音唤醒功能

作为整个系统的触发入口，唤醒功能是必需功能。当用户喊出语音关键词时可以唤醒系统。如功能示意图中的“小贾小贾”即为唤醒关键词，当系统接收到该关键词时，

经过唤醒算法计算，判定系统被唤醒，系统唤醒后，通过网络探测模块镜像网络通路判断，将录音送入下一阶段，进行离线或在线语音识别。语音唤醒功能用例如图 3 所示。

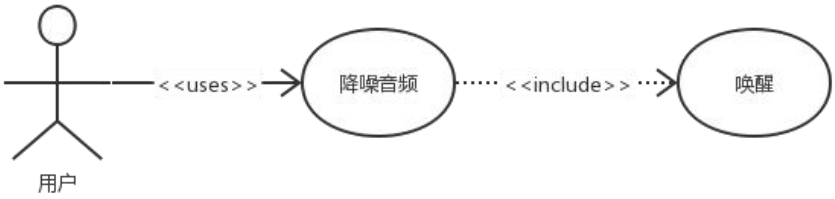

图 3　语音唤醒功能用例图

3）语音识别功能

该功能的作用是将音频转化为文字。该功能根据系统当前网络状况分为在线语音识别和离线语音识别。当系统网络通畅的情况下，采用在线语音识别，在线语音识别的优点是识别率高，识别准确，该功能的实现选择三方平台支持，如使用百度 AI 开发平台提供的在线语音识别服务，该平台提供了优秀的开发服务，具有比较全面的 API 接口[6]。

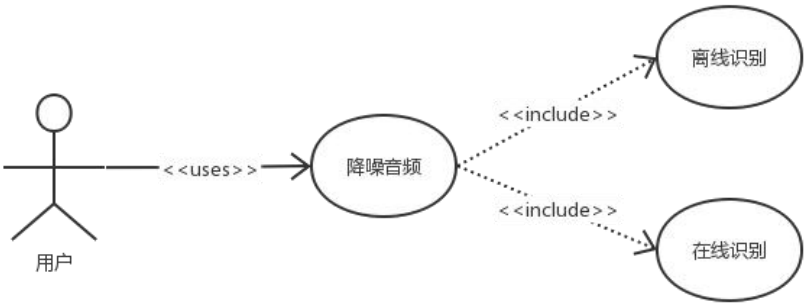

图 4　语音识别功能用例图

当系统网络不好的情况下，采用离线语音识别，离线语音识别的优点是不需要依赖网络，使系统具有一定的抗干扰能力。该功能需要处理后的音频，需在录音后将音频经过 AEC 算法进行内噪消除，之后将音频经过离线识别引擎，该离线识别引擎基于深度学习，通过采集语料训练识别模型。语音识别功能用例图 4 所示。

4）网络探测功能

该功能可以是一个独立任务，每隔一分钟进行网络状态判断并设置网络状态标志位。网络状态的判断可以通过访问因特网上特定网址进行判断，如 www.baidu.com 或

www.sina.com。当用户发起识别请求时能立即拿到相应状态进行在线离线模式切换，并且网络探测与状态获取应设置为异步模式，防止因网络探测造成的系统卡顿[7]。

5）指令识别与外设控制

系统应对语音识别转化后的文字进行有限的关键词分类和匹配，匹配到可控制设备关键词和动作关键词后查找对应设备的控制指令，这一块可通过预先设置字典记录设备关键词，关键动作和控制指令。作为一个嵌入式系统，该系统硬件上应具备最基本的控制功能模组，如红外，蓝牙，WiFi。开发板选型时应当注意支持的外设类型及相关驱动模块。

系统通过匹配到的设备关键词，关键动作和控制指令对设备发起控制指令[8]。这一块要实现对应的内设的驱动和与外部设备交互的协议。

控制模块可以接收外部设备发送回来的状态，同样的通过字典的形式匹配对应状态文本。

6）语音合成与音频播放功能

该功能的作用是文本转音频，将传入的文本经过处理后转化为音频结果，在处理状态文本时可以通过判断网络状态选择在线语音合成或离线音频，语音合成将文本转化为可播放的音频文件。在线功能同样可接入三方平台，使用其提供的语音合成功能[9]。离线音频为系统默认提供音频，数量比较有限。语音合成功能为用户提供可识别的控制结果，是与用户交互非常重要的一环。

9）音频播放功能

系统唤醒后，不管语音识别与控制成功与否，都应有相应的提示话术，如“空调打开成功”“空调打开失败”“指令无法匹配”等。该功能可以通过 TTS 语音合成技术将文本转换成音频，可以将录制好的音频存储下来，根据系统运行状态按序调用相关音频。系统将音频文件解码，送入 Audio 驱动，通过扬声器播放音频数据。用例如图 5 所示。

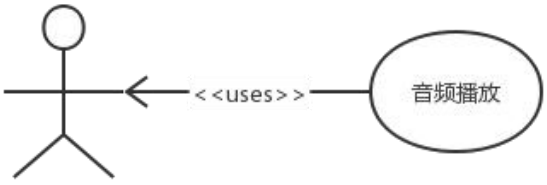

图 5　音频播放功能用例图

2.3 非功能性需求分析

1）唤醒率与误唤醒率需求

系统唤醒率与误唤醒率非常重要，唤醒是该语音控制系统的入口，若唤醒率低，会严重影响系统的可用性，要保证唤醒率在 90%以上（100 次唤醒有 90 次以上可以唤醒）。另一方面，误唤醒率高，也会在用户没有操作的情况下对用户造成干扰，误唤醒率应控制在 2 次每 12 小时（电视噪音下）以下。

2）识别率，字准句准需求

系统的识别率是影响用户控制体验的另一个关键指标，在线识别率依赖于服务商，识别率会比较高，离线识别字准应在 90%以上，句准应在 80%以上[10]。

3）安全性需求

安全性需求是一个比较重要的问题，因为系统会不断录音（送入唤醒引擎），会有泄露用户隐私的风险。本地录音数据应使用 AES(Advanced Encryption Standard)加密存储。另外防止系统入侵，应关闭相关登录接口，如串口，telnet，ssh 等服务[11]。

4）外设控制满足度需求

外设控制应满足用户需求，在用户表达不够明确的情况，应尽量揣摩用户心理，如“小贾小贾，空调”，若当前空调处于关闭状态，可以发送空调打开指令，用户控制满足度应在 80%以上。

5）完整性需求

该系统应满足至少三个智能家电的控制，并且控制能力包括开、关、一项自带功能控制三种能力。

2.4 本章小结

本章首先对市面上的一些智能家居语音控制系统进行了现状分析，针对分析出的痛点进行需求分析，将本系统的需求分为功能性需求和非功能性需求系统。从功能性需求角度将系统分为几个功能模块，并对功能模块进行一一分析。从非功能性需求角度分析，本系统应具有一定性能和使用性。

第三章 系统总体设计

3.1 系统设计原则

本系统设计应遵循以下几个原则：

系统性原则。在本控制系统设计中，遵循系统性的原则，从整个语音控制系统的角度进行设计，系统分为多个子系统，每个子系统又分为几个模块，子系统之间和模块之间都要相互配合，按流程进行设计和实现保证系统的完整性一致性等。

灵活性及可变性原则。本控制系统要有一定的灵活性，随着可控制外设的增加，要有一定的灵活性和扩展性，能够对外界环境变化的适应能力，比如在网络可用的情况下支持在线功能，在网络不稳定或断连的情况下支持离线模式。

可靠性原则。本控制系统要有一定的抗干扰能力，语音唤醒和语音识别对于外部环境具有一定的要求，在系统的设计过程中为了加强噪音情况下的唤醒率和语音识别准确度，要考虑增加一些降噪功能，在硬件和软件算法上都可以采取必要的降噪或语音增强方法能够抵御外界干扰和受外界干扰时的恢复能力。

经济性原则。本控制系统硬件上在能够满足性能开销的基础上尽量采用经济性的芯片和外网控制模组，这样可适当节约成本，减少系统不必要的开销，提高性价比，为以后商业提供基础。

3.2 系统架构设计

根据系统需求分析，我们将系统的整体结构进行了细化设计，如图 6 所示的基于语音的家居控制系统的功能示意图，该图包含了音频数据从麦克风收集经 AUDIO 驱动进入系统后又通过了 AEC 算法进行了消噪处理送入唤醒引擎，当唤醒引擎检查到唤醒词后根据网络状态分别进行了离线识别和在线识别得到识别文字结果，中断控制模块解析文字结果后选择性的控制外部设备。外设返回的控制结果又经控制模块解析后通过 TTS 模块将文字转换为音频数据，最后经 AUDIO 驱动送入扬声器进行播放。

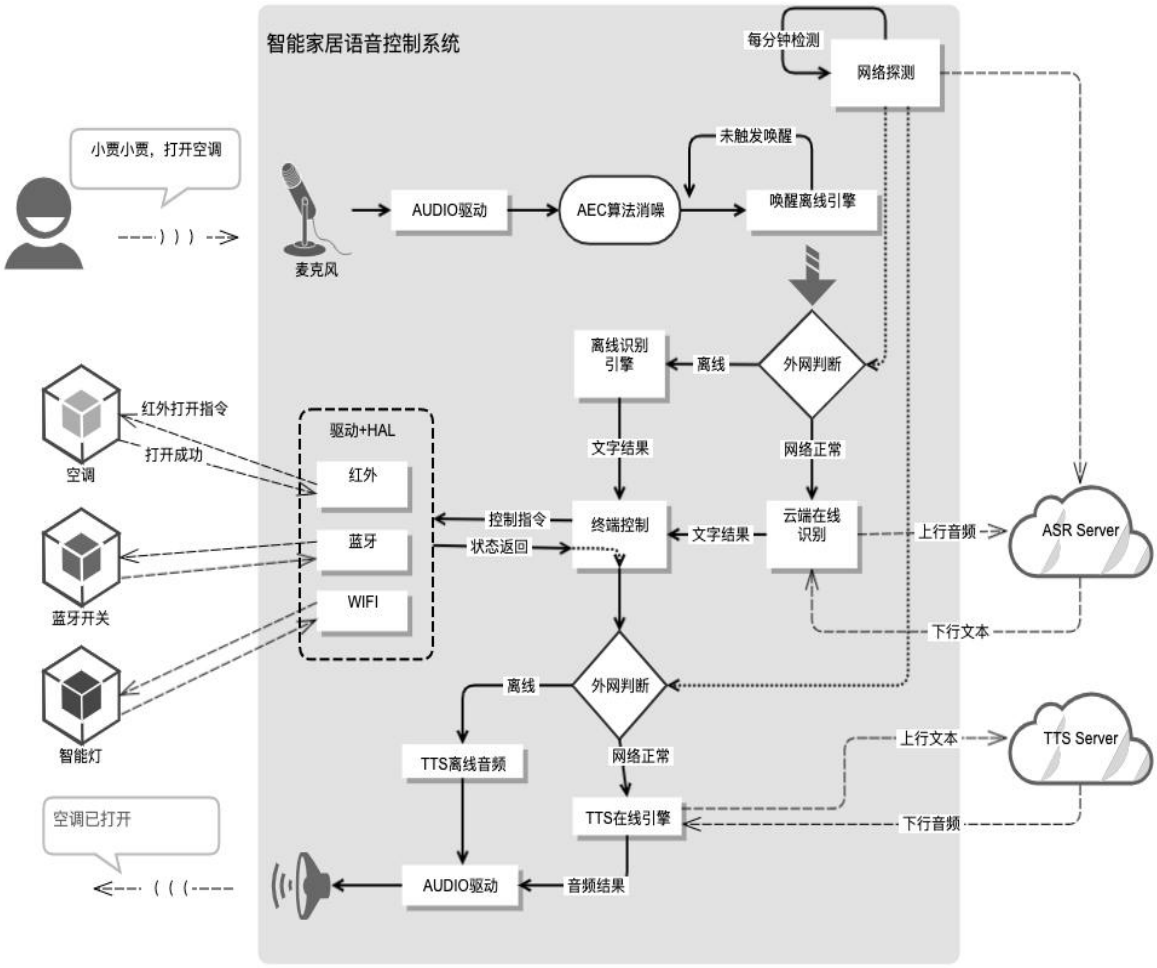

图 6　基于语音的家居控制系统功能示意图

整个基于语音的家居控制系统分为硬件适配层，控制层，算法层，应用层四个层次。每个层次按功能模块又各分成几个功能，层次之间相互通信交互。如图 7 所示的系统架构设计图，该架构设计图比较直观的展示了整个系统的层次结构。

系统及硬件：该层包括操作系统，驱动，硬件及相关硬件外围接口。操作系统使用 Linux，该系统包含了必要的硬件及接口驱动，本系统使用的相关外围接口包括麦克风，扬声器，WiFi 和蓝牙模组，红外发射器等[12]。

硬件适配层：该层主要作用是向上为应用提供统一接口，向下屏蔽硬件和驱动细节。该层集成了包括麦克风，播放器，WiFi 蓝牙，红外等功能模块的驱动接口，提供了统一的读写和控制接口。该层是可移植性的桥梁，简化上层设计，提高开发效率，使本系统结构更加清晰。

传输层：该层包含了录音，播音，控制信号发送，控制信号接收四个功能模块。传输层在硬件适配层的基础上实现了基本的输入输出控制等逻辑，为算法层和应用层提供了必要数据来源和控制通路。

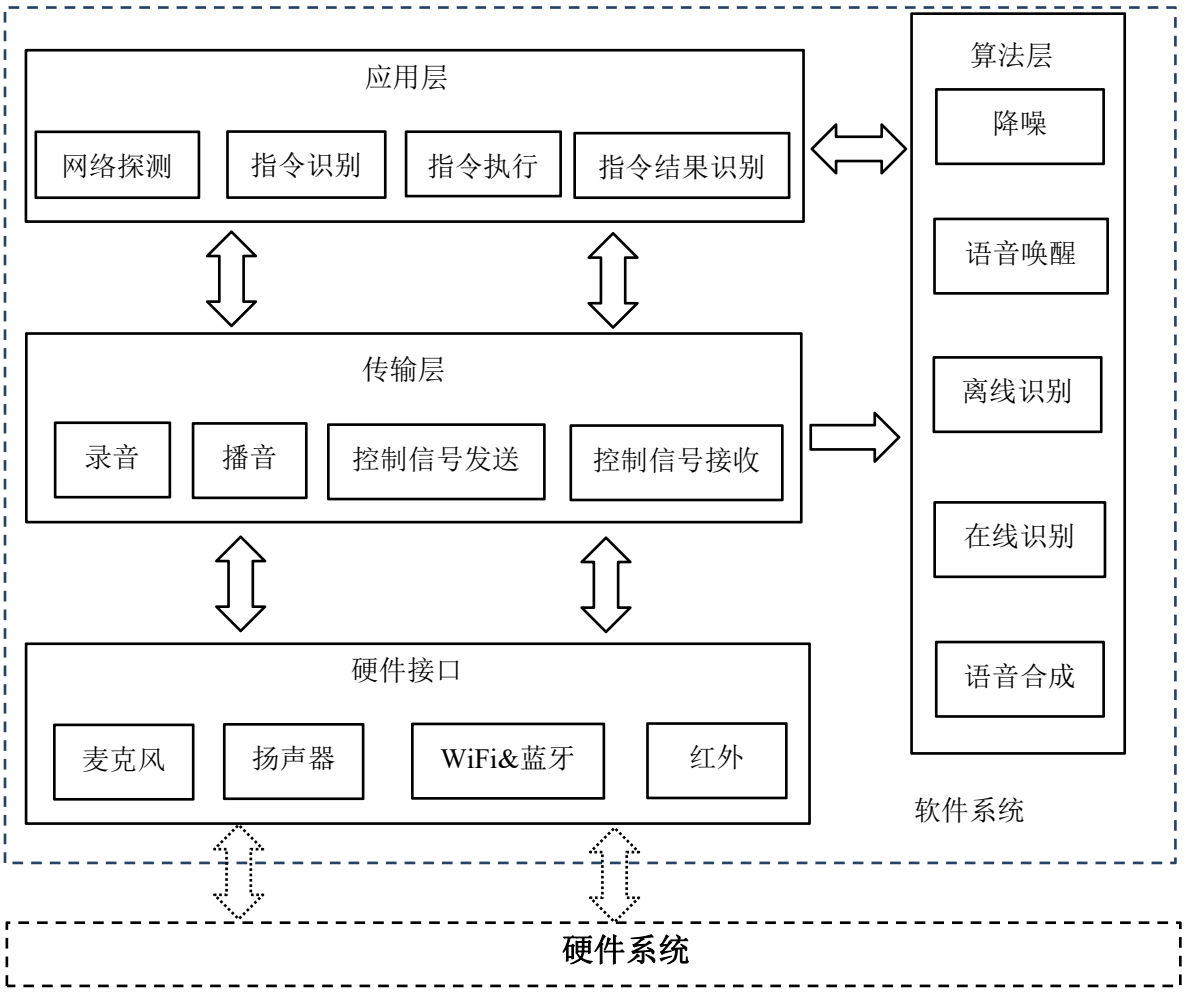

图 7　系统架构图

算法层：该层将系统中相关算法进行了整合，包含语音唤醒，离线识别，在线识别，降噪算法，语音合成等功能模块，是整个系统计算的核心[13]。

应用层：该层承载了主要的业务逻辑，是基于语音的家居控制系统的控制核心。主要包括网络探测，识别结果解析，控制结果解析等功能。

3.3 系统功能结构设计

结合语音控制和智能家居的实际情况，并通过对该系统的需求分析设计出的系统的功能结构如图 8 所示。

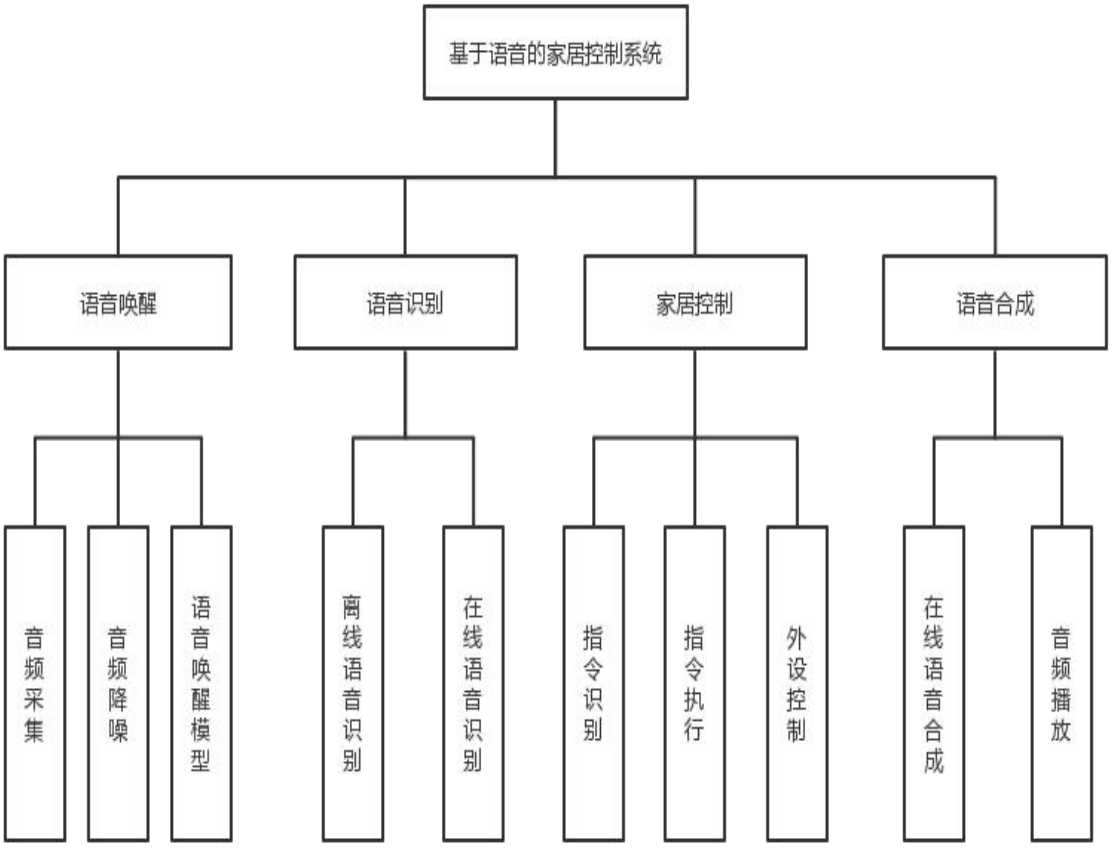

图 8　系统功能结构图

唤醒子系统：该模块包含麦克风数据采集并通过声学算法模块进行处理，最终将数据送入唤醒引擎，唤醒引擎采用 Snowboy，该工具是一款高度可定制的唤醒词检测引擎，具有高度可定制，轻巧，可嵌入的特点，比较符合当前系统需求。唤醒模块流程如图 9 所示。

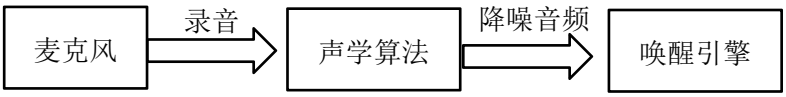

图 9　唤醒模块流程图

语音识别子系统：该子系统主要包含离线识别和在线识别引擎。在线识别引擎主要通过三方接口高质量的完成语音识别转化；离线识别通过深度学习训练的识别模型，当在线识别不可用时采用离线识别，大大提高系统的可用性，系统离线语音识别模块的结构设计如图 10 所示。

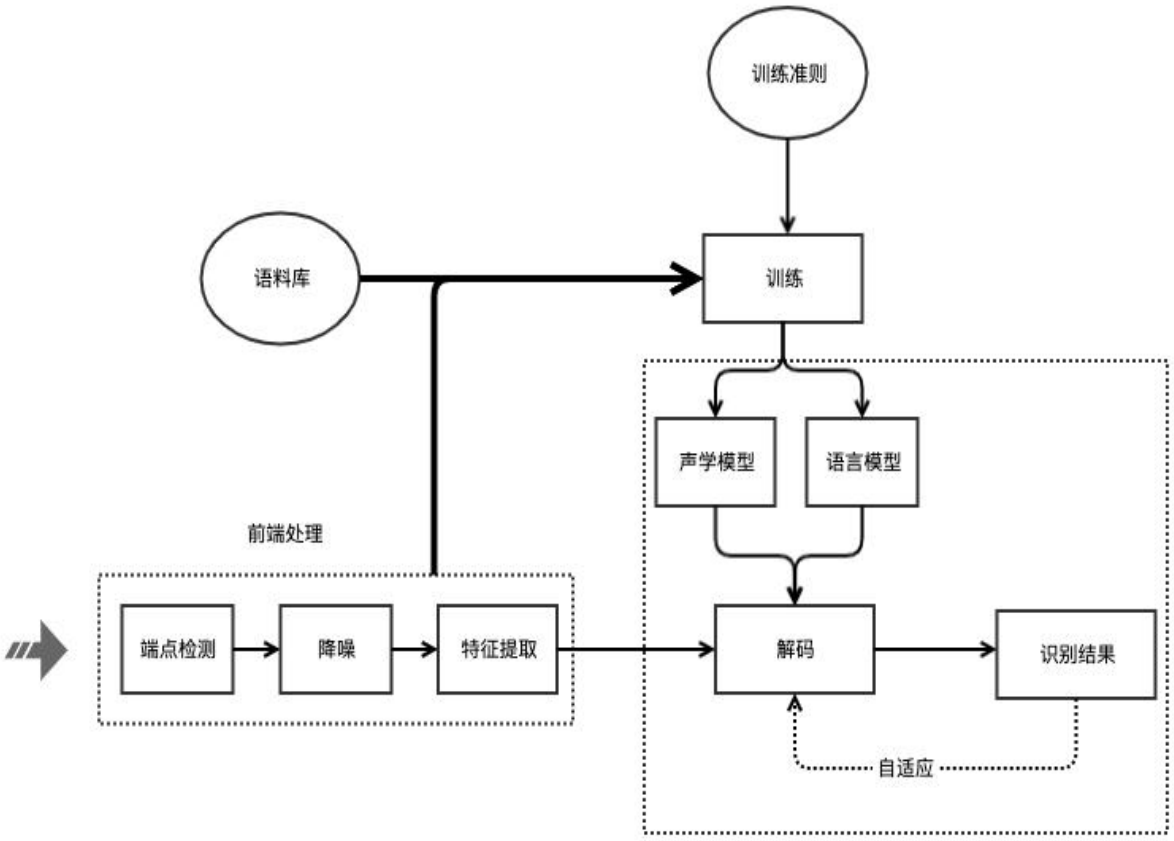

图 10　离线语音识别模块结构图

家居控制子系统：主要包含指令识别和指令执行，对语音识别得到的指令进行匹配，外设的驱动控制和指令发送，另外该子系统还接收外设的指令执行结果，最终用来通知用户指令的执行情况。

语音合成子系统：该子系统包含在线语音合成和音频数据播放两个模块。在线语音合成将执行执行结果的文字通过线上平台提供的语音合成能力转化为音频文件，而音频播放功能则将音频文件通过扬声器播放出来，是人机交互的重要模块[14]。

3.4 系统网络拓扑

整体网络拓扑如图 11 所示，其中家中智能家电包括冰箱，蓝牙开关，空调，手机，智能灯等，这些智能设备均通过无线 WiFi 模组与家中路由器 WiFi 相连，同路由器下的设备组成局域网，本控制系统通过 WiFi 局域网与冰箱连接。本控制系统支持红外发射，可充当空调遥控器开关。

另外本控制系统上有蓝牙模组，可通过蓝牙配对与家中蓝牙智能开光相连。路由器 WiFi 外部与通信厂商局端路由器相连，通过该路由器可访问外网，图中我们访问了两个服务地址，一个是 www.sina.com，一个是 www.ai.baidu.com。www.sina.com 这个服务地址的作用是作为外网探测的地址，我们可以通过这个地址来判断当前网络是否可以

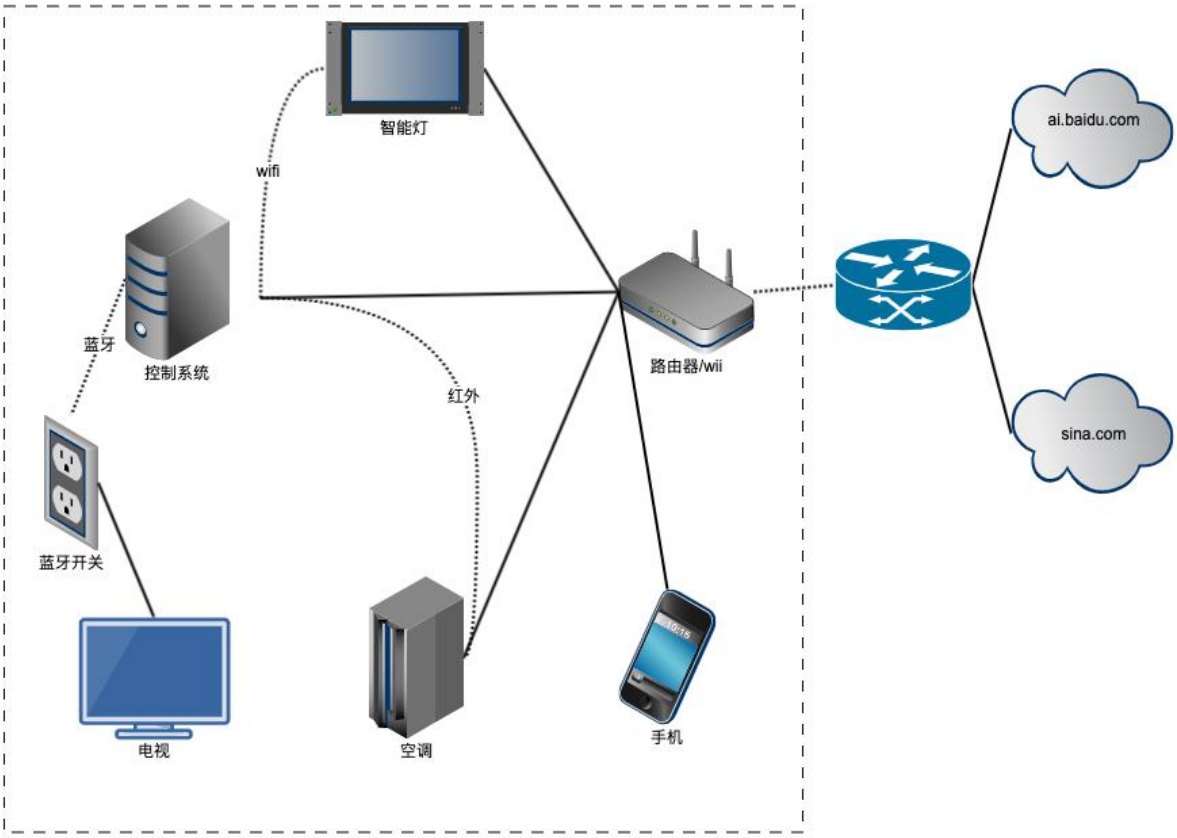

图 11　系统网络拓扑图

访问外网，同时也作为外网是否通畅的测试地址。www.ai.baidu.com 这个服务地址提供了我们在线模式下的语音识别功能，我们通过 API 接口访问这个服务地址并实现在线的语音识别[15]。与 www.sina.com 相同的一个作用是 www.ai.baidu.com 也同样作为外网探测地址。

3.5 接口设计

1) 音频采集接口

void thread_audio_recorder_by_alsa();

接口输入参数为空，返回值为空。接口创建一个独立的线程，通过 Linux ALSA 持续读取音频数据，并将数据存放在循环缓冲区内，当数据存满后会从头进行数据覆盖。

2）降噪接口

void thread_audio_aec_process();

接口输入参数为空，返回值为空。接口会创建一个独立的线程，该线程持续从循环缓冲区内读取数据并调用降噪接口执行 AEC 内噪消除。将内噪消除后的数据存入第二个循环缓冲区内。

3）语音识别接口

离线语音识别请求

void thread_asr_process_offline(void* buffer);

接口输入参数 void*指针，该指针指向待识别的音频，返回值为空。接口将降噪后的音频通过 buffer 指针传递给离线识别引擎，离线识别引擎通过解码器返回识别结果。

在线语音识别请求

void thread_asr_process_online(void* buffer);

接口输入参数 void*指针，该指针指向待识别的音频，返回值为空。接口将降噪后的音频通过 buffer 指针传递给在线识别引擎，在线识别引擎将音频通过网络上传到在线 ASR 识别平台，同时收集识别结果并进行解析。

4）识别结果解析

void* system_instruct_process(void* buffer);

接口输入参数为 void*指针，该指针指向识别文字结果，返回值为 void*指针，该指针为语音结果解析出来的指令函数地址。接口通过比对文字结果和控制指令表进行指令识别。

5）控制结果解析

void* system_instruct_result_process(void* result)

接口输入参数为 void*指针，该指针指向指令执行返回值，返回值为 void*指针，该指针指向返回的文字结果。接口通过指令执行返回值判断指令执行成功与否，并返回文字结果。

6）语音合成接口

void* thread_tts_request_online(void* buffer);

接口输入参数为 void*指针，该指针指向指令执行返回的文字结果，返回值为 void*，该指针指向语音合成的音频地址。接口通过请求百度 TTS 在线语音合成平台，将传入的文字结果转换为音频数据，并将音频地址返回。

3.6 开发板选型

从系统、计算芯片类型、开发环境、生态完善度等多个因素考虑，选择以下方案进行探索并调研[16]。

1）瑞芯微 RK3399 开发板

2）瑞芯微 RK3399PRO 开发板

3）华为 Atlas200DK 开发板

4）Nvida JetSon TX2 开发板

5）Nvida Jetson Xavier NX 开发板

选型调研过程，以人体关键点识别为例，对比如表 1 所示。

表 1　开发板功能对比表

序号	方案	加速芯片类型	环境搭建难度	操作系统	支持接口	主流深度学习框架支持	主流框架的模型支持
1	瑞芯微 RK3399	GPU	中等	Android Linux	WiFi 蓝牙 红外	主流框架无加速	同框架
2	瑞芯微 RK3399PRO	GPU NPU	中等	Android Linux	WiFi 蓝牙 红外	主流框架无NPU加速，只能官方RKNN、Rockx框架	Caffe、TF、TF-Lite、ONNX、Darknet 等模型转换
3	华为 Atlas200DK	AI 芯片	复杂	Linux	WiFi 蓝牙	主流框架无 AI 芯片加速，只能官方框架	Caffe、TF
4	Nvida JetSon TX2	GPU	中等	Linux	WiFi 蓝牙 红外	主流框架都支持但无加速 TensorRT 可加速	同框架
5	Nvida Jetson Xavier NX	GPU	中等	Linux	WiFi 蓝牙 红外	都支持但无加速 TensorRT 可加速	同框架

以上数据是根据实际测量与开发板官网数据进行整合的，另外为了有一个更全面的认识和体验，做了更进一步的探索，开发板功能深度体验如表 2 所示。

表 2 开发板功能深度体验表

序号	方案	深度体验	使用感受
1	瑞芯微 RK3399	a.TF-Lite 框架，posenet 模型，5-10FPS。	性能一般，对性能要求比较低的场景可以使用
2	瑞芯微 RK3399PRO	a. Rockx 框架，pose 模型，40FPS，但精度很差。 b.RKNN 框架，posenet 模型，10FPS。	性能一般，相比RK3399 有 NPU 加速，对性能要求一般的场景可以使用
3	华为 Atlas200DK	a. 验证官方Demo。 b. 加载 Caffe 的 openpose 模型，4FPS。	从搭建环境 Demo 跑通，踩坑最多的方案，文档不明确、模型支持少，使用较为繁琐，生态环境差。宣称性能不错，但不适合快速开发产品。
4	Nvida JetSon TX2	Pytorch 下 yolo+pose_resnet50 模型串联，10FPS。	性能满足大部分使用场景，同 PC 一样，基于 CUDA，生态完善，部署成本低。
5	Nvida Jetson Xavier NX	Pytorch 下 yolo+pose_resnet50 模型串联，10FPS。	性能强于 TX2，满足大部分使场景，同 PC 一样，基于 CUDA 生态完善，部署成本低。

选型调研总结：

1）国产瑞芯微 RK3399 和 RK3399PRO，性能一般，开发难度一般，可以满足模型精度、以及实时性要求不高的场景。优点是支持 Android 系统[17]。

2）华为 Atlas200DK，算力虽然强，但开发部署过于繁琐，且支模型持较少生态不完善，遇到问题不好解决，不太适合快速开发产品。

3）Nvidia JetSon 系列，符合我们对通信接口的基本需求，同时支持 WiFi，蓝牙，红外。基于 CUDA，性能强，迁移部署成本低，生态完善，主流框架都可支持，适合快速开发产品。Xavier NX比TX2 增加了 TensorCore 以及 CUDA 核心数，性能更强，优先选择 Nvidia Xavier NX。

3.7 系统采用的关键技术难点及解决方案

本论文的关键技术难点在于离线语音识别模块和多外设控制模块。

1）离线语音识别

离线语音识别模块是在系统本地通过人工智能语音识别技术将音频信号转变为机器可识别的文字，进一步通过自然语音理解技术转变为机器可以执行的指令的技术。而这一切都在本地系统上执行，不需要通过网络和服务器。目的就是赋予机器人的听觉能力，可以听懂人的语言和用户意图，并作出相应的行为。

离线语音识别难点有两方面，一是噪声的困扰，因为噪声环境有各种各样的声源，人声，汽车声，走路声，音乐等，而各类声源的处理是目前公认的技术难题，在机器的角度根本无法从复杂多样的背景噪音中分辨出人声，对于千差万别的背景噪声，即使是深度学习也无法训练出完全匹配的真实环境[18]。因此，要实现噪音下的语音识别要比在安静的场景下难得多，这也是无法避免的问题。二是模型的有效性，识别系统中包括语言模型、词法模型在短句和少量词汇的情况下效果还可以，但要放在在大词汇量和一些连续语音识别的场景中还不能完全正常的工作，为此，除了基于大数据的训练，还需要有效地结合语言学、心理学及生理学已经一些语言习惯等其他学科的知识[19]。另外，语音识别系统在实验室演示效果虽然不错，但真正要转化到商品的还有许多具体的技术细节等问题需要解决。

2）多外设控制模块

本系统为了能控制所有家居设备，需要相关的硬件接口和通信协议来支持。目前物联网智能家居多种多样，使用的通信方式和协议也非常复杂，因此如何在保证具有基本控制能力的基础上尽量多的控制家居设备是本系统的难点。

家居设备中最常用的通信协议包括 WiFi，蓝牙和红外。因此在开发板的选型时要关注支持的硬件接口和相关的通信协议。

3.8 本章小结

本章主要介绍了系统的总体设计，从系统的整体架构开始，详细设计了各个子系统，又对各个子系统所包含的模块进行了细致的分析和设计，对于硬件系统方面的开发板进行了选型和设计。展示了系统的各个模块和整体运行的俯视图，对于后续的详细设计和实现具有指导作用。

第四章 系统详细设计与实现

4.1 控制系统的底层设计与实现

我们都知道智能系统的智能主要依靠它的心脏，主控芯片提供算力。一个完整的控制系统，其核心为底层芯片和系统组成的平台，该平台为上层应用和算法提供了基本的环境和算力。探索 AI 在嵌入式设备上的使用时方式和性能表现，指定可用于人工评估的智能嵌入式方案，让 AI 赋能评估，使系统的实现和表现更加准确高效。

1）Nvida Jetson Xavier NX 环境搭建

Nvidia Jetson Xavier NX 形状，外接口类似于树莓派的嵌入式主板，搭载了 6 核 NVIDIA CarmelARMv8.264 位 CPU，GPU 则是有 384g 个 NVIDIA CUDA 内核和 48 个 Tensor 内核的 NVIDIA Volta 架构，具有以太网接口，USB 主机从机，两个 SPI 接口，四个 TWI 接口，六个 UART 接口，三个 SD 卡接口，两个 I2S/PCM 数字音频接口。采用八核 cortex-A7 处理器，SGX544 GPU，支持 60 帧的 1080P 视频播放，采用 HawkView ISP，最高支持 800 万像素摄像头。内存为 8GB 的 LPDDR4x@51.2GB/s，支持 DDR3/DDR3L/LPDDR3/LPDDR2 多种规格，满足不同成本定位应用。存储支持 EMMC4.5，采用 FCBGA 封装，345pin，14*14mm 面积。支持 4K 60Hz 视频解码。基本上可以将 Jetson Xavier NX 视为一台适用于深度学习的算力强大的微型电脑。

材料准备，NX 开发板+19V 电源电流 1A，32G 以上 TF 卡，路由器、网线 X2，电脑和开发板组网。

烧录固件，在 Linux 电脑上下载并安装 Nvidia SDK Manager 并运行。下载镜像，使用USB 连接开发板和电脑，照软件提示进行烧录。安装所需软件和环境，与 Ubuntu 类似。安装 Pytorch。使用开发板进行开发测试 。

2）系统网络配置

系统初次启动时，本系统需要用户介入进行网络配置，经过调研选择使用 webApp 开源框架进行实现，webApp 是基于 web 的系统和应用，在系统初次启动和网络连接失败的情况下系统将拉起 webApp 进程启动一个网络配置的 web server，用户通过访问 192.168.2.1 进行网络 ssid 和 psk 的配置，配置完成后系统自动重启 WiFi 模块进行联网。

4.2 音频处理功能设计与实现

4.2.1 音频采集

音频的采集和播放在软件上主要是写音频的驱动程序，同时提供接口给上层调用。

Linux 中跟音频相关的就是大名鼎鼎的 ALSA(Advanced Linux Sound Architecture)了。它是 Linux 上的音频子系统，在 kernel space 和 user space 都有相应的代码。kernel space 里主要是音频的驱动程序，user space 里主要是 alsa-lib,也就是提供接口给上层应用程序调用。User space 和 kernel space 通过字符设备进行交互。

Linux 下本地录音模块流程如图 12 所示。

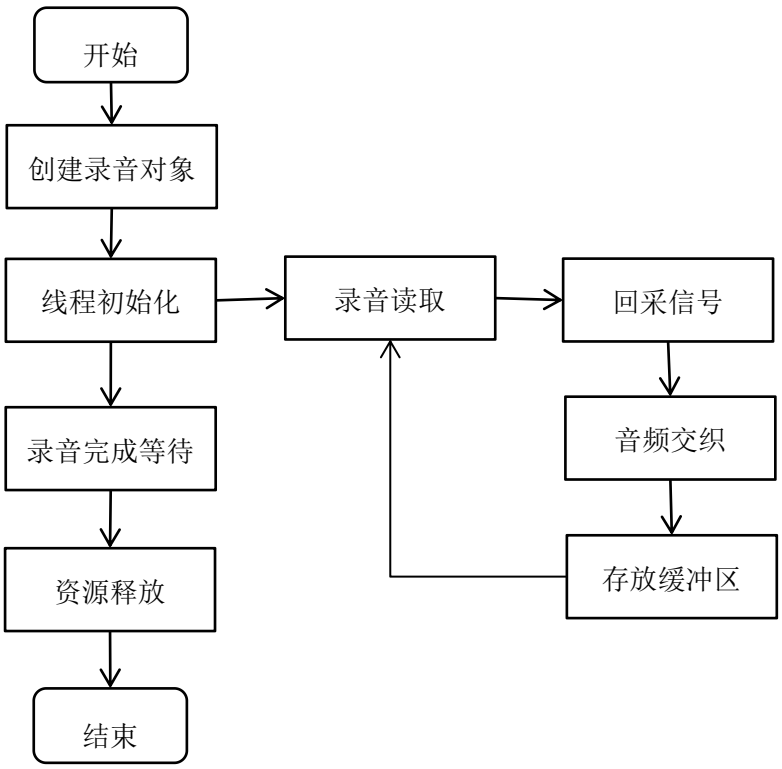

图 12　录音模块流程图

如流程图，程序开始后我们首先创建录音对象，该对象主要功能是初始化变量和分配存储录音的循环缓冲区。创建一个线程，该线程会一直存在于整个系统的生命周期中，主要功能是对录音和回采信号进行读取并进行音频的交织，存储到循环缓冲区中，当缓冲区存满后写指针会从头开始覆盖原来的数据，设计循环缓冲区的目的是防止数据无限增多导致的堆溢出或内存不足的情况。整个录音的核心是录音读取部分，以下是录音数据读取流程。

首先要配置硬件参数，包括设置采样位数、通道数、采样率等，然后向 DMA 缓存区写或者读，实现播放和录音，接口设计如下：

打开 PCM 设备，录制回采信号，调用接口为 snd_pcm_open ()打开 PCM 设备，录音音频 snd_pcm_open ()，申请录音参数对象 snd_pcm_hw_params_alloca ();初始化录音对象和设置录音对象模式，设置 PCM 为通道数和采样率，另外需要设置一个录音周期，也就是需要录制一段音频的时长和数据量，使能配置，将参数设置到驱动上。申请录音存储 buffer，将录音音频流写入文件。

编译时，需要引入 alsa 库，-L 指定具体的 alsa-utils 类库的 asound 等，按如下方式进行编译。gcc -o local_player local_player.c -L ./alsa-utils-1.1.5 -lasound -lm -ldl

测试包括播放和录音，通过扬声器是否正常播放音频文件，确认播放成功。通过 dump 出来的文件，用 audiocity 或其他软件打开并播放，确认录音功能正常。

4.2.2 音频降噪

完成对音频的采集后，我们需要对音频进行降噪处理，因为音频质量对于本系统至关重要，不仅影响系统的唤醒率，还对后续的语音识别字准句准影响非常大。降噪方法我们选择使用回声消除算法。对于音频，回声消除是为了消除机器自身发出的声音，不影响外界传递过去的声音。本系统在进行音频播放时，来自扬声器放出来的声音会混叠人声一起被麦克风录进去，严重影响信号质量。

本系统采用了 Speex 高性能语音编解码库，该库提供了声学中回声消除 AEC 算法，通过 API 接口将该库封装到我们系统的语音处理模块中，为了使用方便，本系统将 Speex 中音频处理相关的 API 统一提取出来，形成一个中间的 API 层封装在 libaec.so 中。本系统音频降噪流程如图 13 所示。

流程中采用多线程的方式，一个线程用于音频读取和降噪，创建 Speex 回声消除状态器函数为 speex_echo_state_init();创建 Speex 预处理状态前的函数。音频读取是从音频采集模块的 CaptureCacheBuffer 获取的，每次读取帧大小为 8ms，读取 32ms 3 路麦克风数据和 1 路参考信号各 32ms 数据，也就是 4 帧数据。读取数据成功后进行 Speex 帧预处理，函数为 speex_echo_cancellation();然后进行 Speex 回声消除，处理完的数据存入 AECCacheBuffer，待唤醒模块读取。另一个线程也就是主线程等待数据处理线程完成，

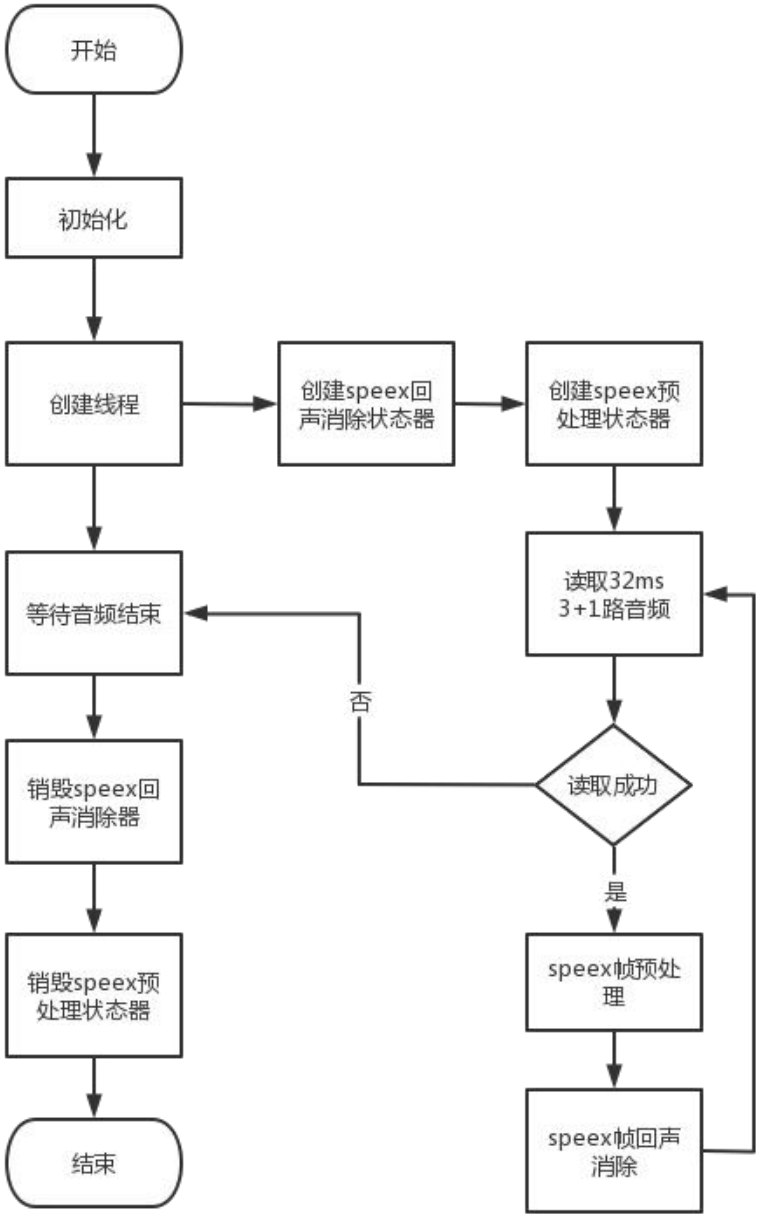

图 13　音频降噪流程图

完成后销毁 Speex 回声消除器，销毁 Speex 预处理状态器至此音频降噪模块实现完成。

4.3 唤醒功能设计与实现

语音唤醒算是语音识别领域里最基础的应用，简单来说就是在后台静默地运行着一个占用较少系统资源的语音识别组件服务，该组件一直处于监视麦克风输入的状态，如果有检测到特定的语音输入，即唤醒词，则激活与之绑定的某个程序“开关”。相当于一个简化版的语音助手，只对某一个特定的词汇进行响应，识别后也只完成某一件指定

的任务。如果说同语音助手的交互是一段持续的交流，那么语音唤醒即可作为这种连续交流的入口。

本系统采用 snowboy，一个开源的，轻量级语音唤醒引擎，来定制我们的系统唤醒词“小贾小贾”。snowboy 唤醒引擎的主要特性如下：

●高度可定制性，可自由创建和训练属于自己的唤醒词。

●始终倾听，可离线使用，无需联网，保护隐私。精确度高，低延迟。

●轻量可嵌入，耗费资源非常低（单核 700MHz 树莓派只占用 10%CPU）。

●开源跨平台，开放源代码，支持多种操作系统和硬件平台，可绑定多种编程语言。

4.3.1 定制唤醒词

1）环境配置

安装 pulseaudio 软件以减少音频配置的步骤：sudo apt-get install pulseaudio，安装 sox 软件测试录音与播放功能：sudo apt-get install sox，安装完成后运行 sox -d -d 命令，对着麦克风说话，确认可以听到自己的声音。安装 PyAudio：sudo apt-get install python3-pyaudio。安装 SWIG（>3.0.10)：$ sudo apt-get install swig。安装 ATLAS：$ sudo apt-get install libatlas-base-dev。

获取源代码：$ git clone https://github.com/Kitt-AI/snowboy.git。使用 python3 编译：$ cd snowboy/swig/Python3 && make

2）唤醒测试

进入官方示例目录 snowboy/examples/Python3 并运行以下命令：

$ python3 demo.py resources/models/snowboy.umdl

命令中的 snowboy.umdl 文件即语音识别模型。修改 snowboy/examples/Python3 目录下的 snowboydecoder.py 文件后运行，然后对着麦克风清晰地讲出“snowboy”，如果可以听到“滴”的声音，则安装配置成功。命令行为 snowboy test。

将包含“小贾小贾”唤醒词的音频文件上传至 snowboy 官网，训练生成本系统需要的语音模型。需要上传 3 个 wav 格式的音频文件，可直接在线录制。训练完成并测试通过后，下载 PMDL 后缀的模型文件，该文件即为“小贾小贾”唤醒词的模型文件。将下载好的 model.pmdl 模型文件复制到自己的项目目录下。另外将 snowboy/swig/Python3

目录下的_snowboydetect.so 库， snowboy/examples/Python3 目录下的 demo.py，snowboydecoder.py，snowboydetect.py 文件和 resources 目录均复制到项目目录下。

在项目目录下执行$ python3 demo.py model.pmdl，唤醒程序已启动，喊出“小贾小贾”，程序被成功唤醒，唤醒结果如下所示。

INFO:Keyword 1 detected at time:2021-08-15 10:11:22 wakeup word 小贾小贾

4.3.2 定制唤醒动作

唤醒模块作为系统触发的第一步，直接影响到用户的使用体验。同时作为开启接下来识别和控制的入口，按照需求，需要定义唤醒后的动作。

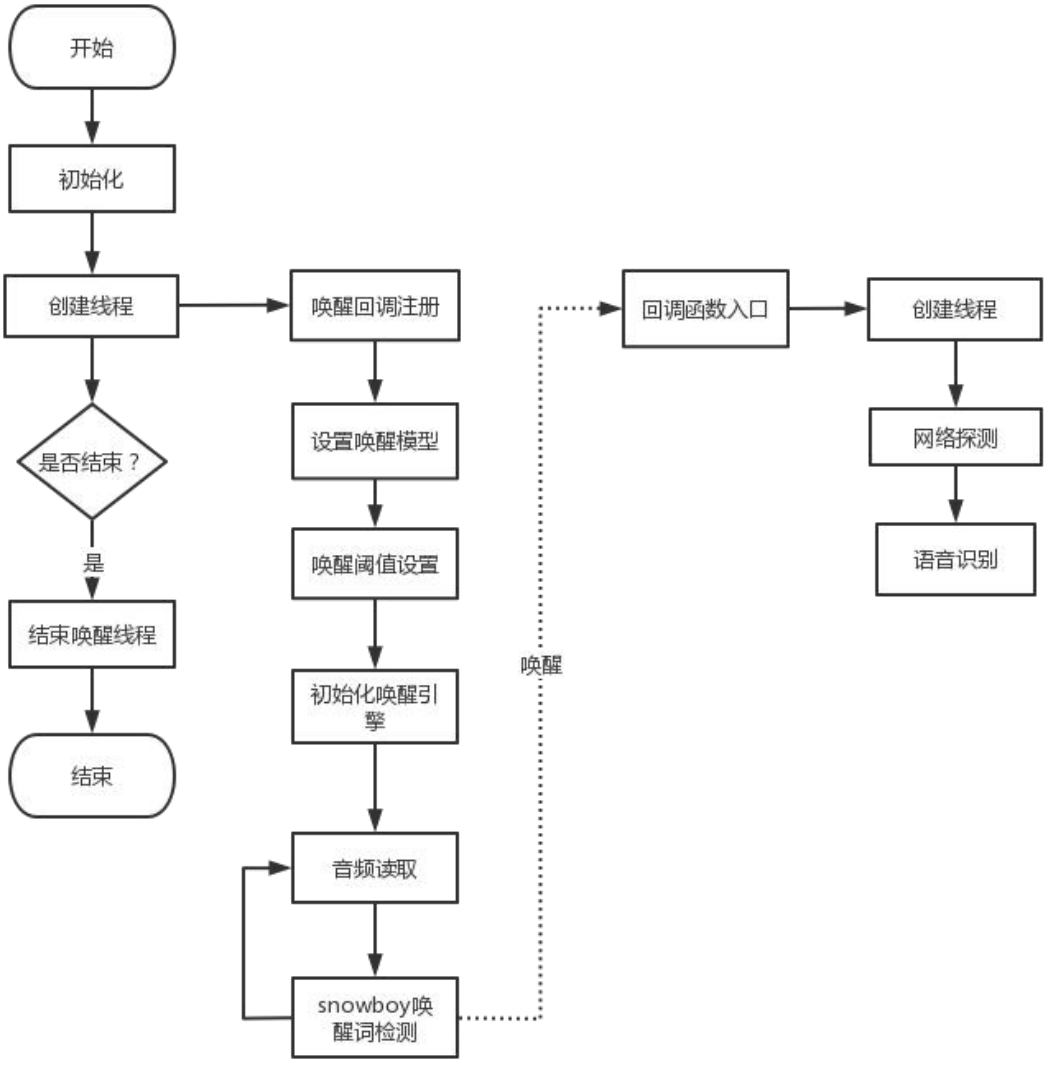

图 14　唤醒模块流程图

唤醒模块分为两个线程任务和一个回调函数。主线程负责创建唤醒任务线程并检查唤醒结束标记，如果用户要关闭唤醒，主线程会结束掉唤醒任务线程，结束函数为detector.terminate()。唤醒任务线程负责唤醒回调注册，设置唤醒模型和唤醒阈值，之后进入主循环，从 AECCacheBuffer 读取降噪后的音频，并调用 snowboy 唤醒词检测接口，接口为 detector.start(aec_audio_buffer,sleep_time=0.03)。唤醒词识别成功以后，程序响应的具体内容由回调函数定义，本系统唤醒的后续操作是创建线程并进行网络检测和语音识别。唤醒模块的执行流程如图 14 所示。

4.4 语音识别功能设计与实现

本系统的核心模块是语音识别，语音识别作为信息技术领域非常重要的科技发展技术之一，近二十年来逐渐从实验室走向实际应用场景。语音识别技术可以将人类发出的声音通过一系列算法转化为计算机可以识别的输入，甚至是人类可以识别的文本信息。本系统支持两种形式的语音识别，一种是在线识别，目前市面上大多数产品采用这种形式，将识别模型和处理算法放到云端，以 C/S 架构的形式提供服务。另一种是离线识别，顾名思义，可以支持在没有网络的情况下进行语音转文字处理，识别模型就在本地系统上，作为在线识别的补充，使系统在网络异常情况下仍能正常工作。

4.4.1 网络探测模块设计与实现

网络探测模块采用守护进程的方式，使用 Linux shell 脚本实现。将进程启动添加到/etc/init.d/initrc 中使其在开机时启动。

判断网络畅通是通过 curl 来访问服务器地址，从而判断服务器网络状态是否畅通，服务器地址选取 www.sina.com 和 www.ai.baidu.com。网络探测流程如图 15 所示。

系统开机后需要进行网络初始化，相关初始化脚本会拉起 wpa_supplicant 进程，该进程会按照/data/wpa_supplicant.conf 配置文件进行 WiFi 连接，WiFi 连接成功后每分钟进行网络探活，探活结果会写入/data/network_status 文件，格式为 wifi:on/off。

关键代码为 local ret_code=`curl -I -s --connect-timeout 1 “www.ai.baidu.com” -w | tail -n1`。

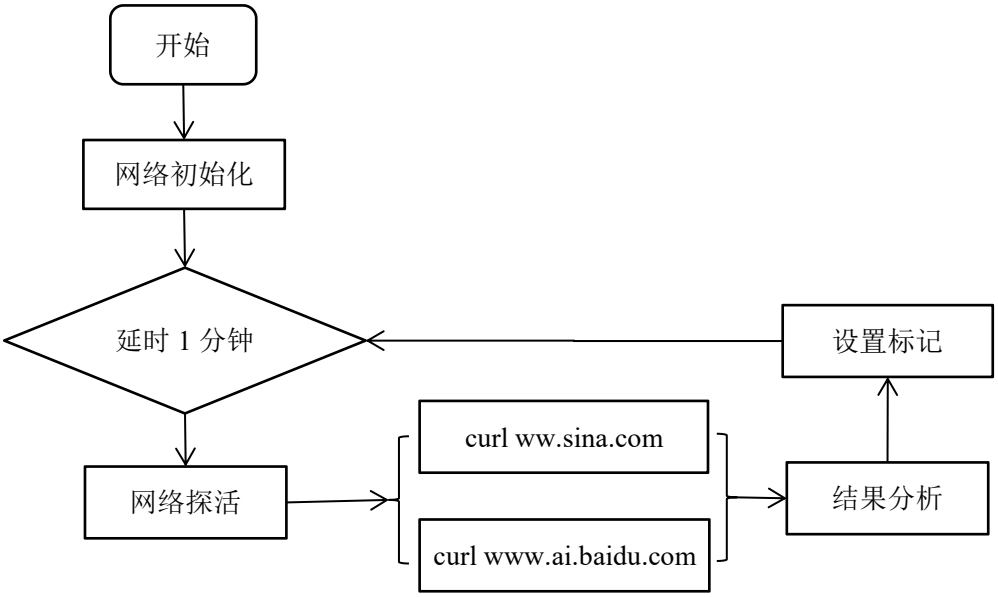

图 15　网络探测流程图

4.4.2 离线识别

通过前期调研对比，选择 Kaldi 作为离线识别的解决方案。Kaldi 是当前最流行的开源语音识别工具，它使用 WFST 来实现解码算法。Kaldi 的主要代码是 C++编写，在此

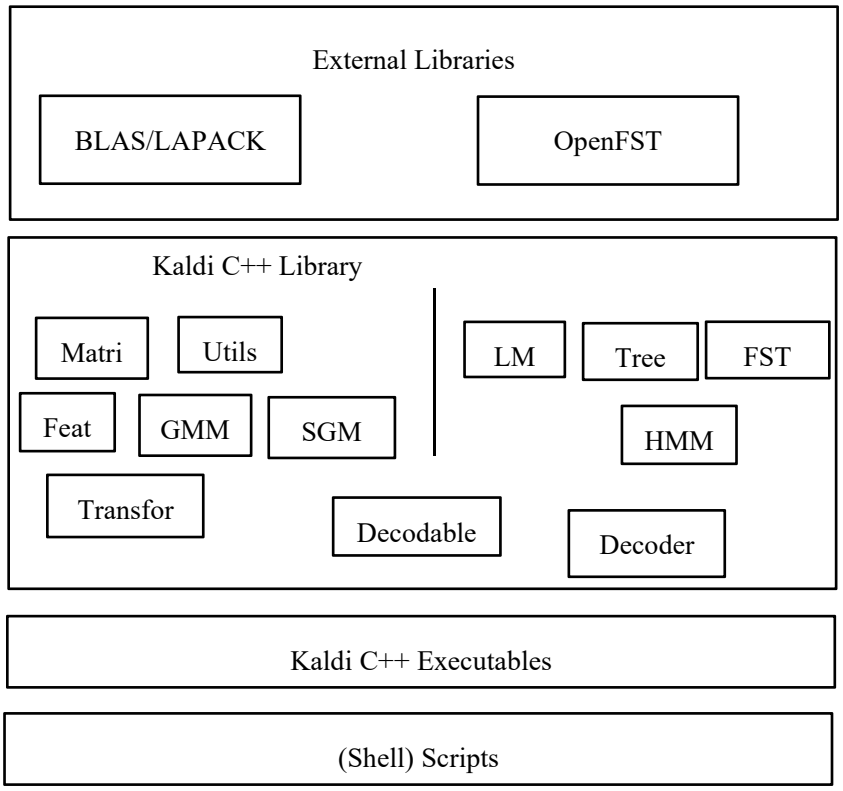

图 16　Kaldi 模块结构图

之上使用 bash 和 python 脚本做了一些工具.语音识别，大体可分为“传统”识别方式与“端到端”识别方式，其主要差异就体现在声学模型上。Kaldi 的整体模块结构如图 16 所示。

图中 ExternalLibraries 模块中包含 BLAS/LAPACK 和 OpenFST，OpenFST 的主要作用是构造有限状态机，对于有限的状态进行构造遍历搜索优化等。在计算过程中常用的线性代数相关的计算封装在 ATLAS 库中，提供了 C++的使用接口。IRSTLM 是一个统计语言模型的工具包。sph2pipe 是处理 SPHERE_formatted 数字音频文件的软件，它可以将 LDC 的 sph 格式的文件转换成其它格式。SRILM(SRILM - The SRI Language Modeling Toolkit)是由 SRI International 提出的一套工具集，主要用于创建和使用统计语言模型。传统方式的声学模型一般采用隐马尔可夫模型 HMM，而“端到端”方式一般采用深度神经网络 DNN，Kaldi 主要用脚本来驱动，每个 recipe 下会有很多脚本。local

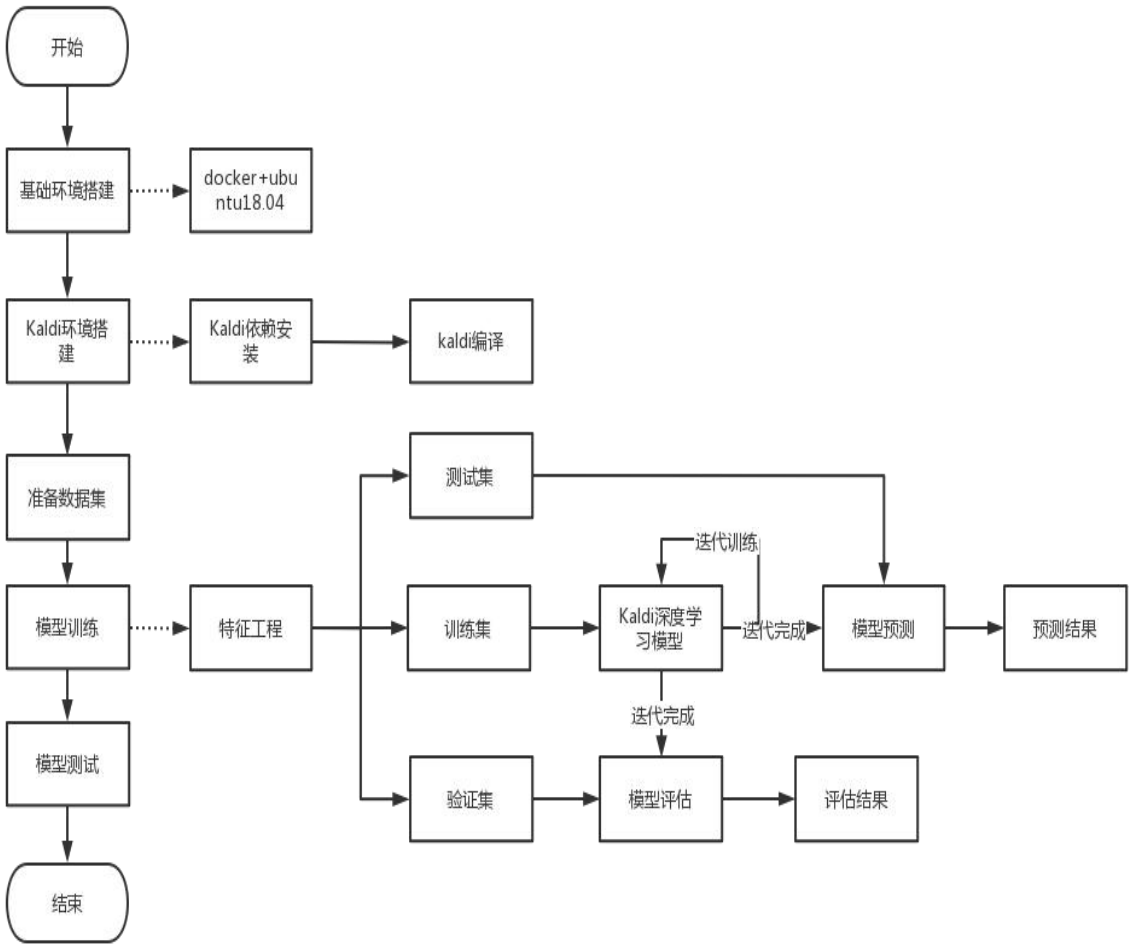

图 17　Kaldi 识别训练过程如图

目录下的脚本通常是与这个 example 相关，不能移植到别的例子，通常是数据处理等一次性的脚本。而 util 下的脚本是通用的一些工具。steps 是训练的步骤，最重要的脚本。

Kaldi 识别训练过程如图 17 所示。基础环境搭建部分主要搭建相关基础开发环境，使用使用 docker 搭建 Ubuntu18.04 的开发环境。docker 启动参数为 docker run -it -d -m 4G fengzhishang/kaldi_env:1 /bin/bash 。

Kaldi 环境搭建部分包含代码下载，依赖检查，依赖软件安装，外部依赖安装，Kaldi 软件配置，编译等模块。因编译过程比较消耗内存，使用 Makefile 4 线程进行编译。

准备数据集部分，Kaldi 中文语音识别公共数据集一共有 4 个，分别是 aishell: AI SHELL 公司开源 178 小时中文语音语料及基本训练脚本，位于 Kaldi-master/egs/aishell。gale_mandarin 是中文新闻广播数据集 LDC2013S08,LDC2013S08。hkust: 中文电话数据集(LDC2005S15, LDC2005T32)。thchs30: 清华大学 30 小时的数据集。本系统选择 thchs30 语料库。首先制作我们需要语料库，语料包含 3 个文件，具体内容如表 3 所示。

表 3 thchs30 数据表

包名	大小	内容
data_thchs30	6.4G	语音数据和训练脚本
test-noise	1.9G	标准无噪音测试数据
resource	24M	补充资源，包括培训数据词典、噪音样本

将数据包解压到 egs/thchs30/s5/thchs30-openslr 下，另外包含训练好的语言模型 word.3gram.lm 和 phone.3gram.lm 以及相应的词典 lexicon.txt。这个数据集包含以下内容如表 4 所示。

表 4 thchs30 数据集内容

数据集	音频时长(h)	句子数	词数
train(训练)	26	20000	386504
dev(开发)	2:15	1611	35486
test(测试)	7:16	4980	98170

训练集 train 包含的数据量最多，主要用于模型的训练。Dev 开发集的作用是为了训练出更好，与 train 训练集进行交差验证。训练的结果分为音素和词，测试也根据音素和词进行目标数据分类。local/thchs-30_data_prep.sh 主要工作是从 thchs/data_thchs30（下

载的数据）三部分分别生成 word.txt（词序列），phone.txt（音素序列），text（与 word.txt 相同），wav.scp（语音），utt2pk（句子与说话人的映射），spk2utt（说话人与句子的映射）。

MFCC features 是提取 MFCC 特征，分为两步，先通过 steps/make_mfcc.sh 提取 MFCC 特征，再通过 steps/compute_cmvn_stats.sh 计算倒谱均值和方差归一化。language stuff 是构建一个包含训练和解码用到的词的词典。而语言模型已经由王东老师处理好了。基于词的语言模型包含 48k 基于三元词的词，从 gigaword 语料库中随机选择文本信息进行训练得到，训练文本包含 772000 个句子，总计 1800 万词，1.15 亿汉字。基于音素的语言模型包含 218 个基于三元音的中文声调，从只有 200 万字的样本训练得到，之所以选择这么小的样本是因为在模型中尽可能少地保留语言信息，可以使得到的性能更直接地反映声学模型的质量。这两个语言模型都是由 SRILM 工具训练得到。

数据准备，monophone 单音素训练，tri1 三因素训练，trib2 进行 lda_mllt 特征变换，trib3 进行 sat 自然语言适应，trib4 做 quick，后面就是 dnn。执行 run.sh 即可开始训练。

查看训练效果 cat tri1/decode_test_word/scoring_kaldi/best_wer：

%WER:36.06[29255/81139,545ins,1059del,27651sub\]exp/tri1/decode_test_word/wer_10_0.0

得到模型后，整体的语音识别设计如图 18 所示，语音输入后先经过信号处理，该信号处理与训练数据经过的处理一致，基于声学模型和语言模型进行解码得到文本输出。

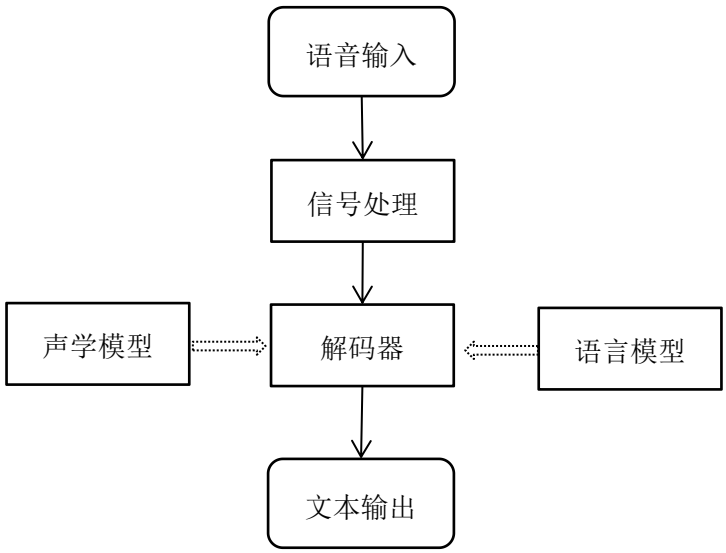

图 18　语音识别需求图

执行语音识别，将声音文件复制到 online-data/audio/ 目录，然后运行 run.sh 执行识别操作，识别效果如下：

online-wav-gmm-decode-faster --verbose=1 --rt-min=0.8 --rt-max=0.85 --max-active=4000 --beam=12.0 --acoustic-scale=0.0769 --left-context=3 --right-context=3 scp:./work/input.scp online-data/models/tri1/final.mdl online-data/models/tri1/HCLG.fst online-data/models/tri1/words.txt 1:2:3:4:5 ark,t:./work/trans.txt ark,t:./work/ali.txt

File: A1_00 “灯光调亮”“灯光调暗”“打开电视”“关闭电视”“打开空调”“关闭空调”

由此验证，该模型可以作为本系统离线识别模型。

4.4.3 在线识别

在线识别选择接入百度 AI 开放平台，下载 BDSSDKMessage 语音识别 SDK 包，接入流程如图 19 所示。

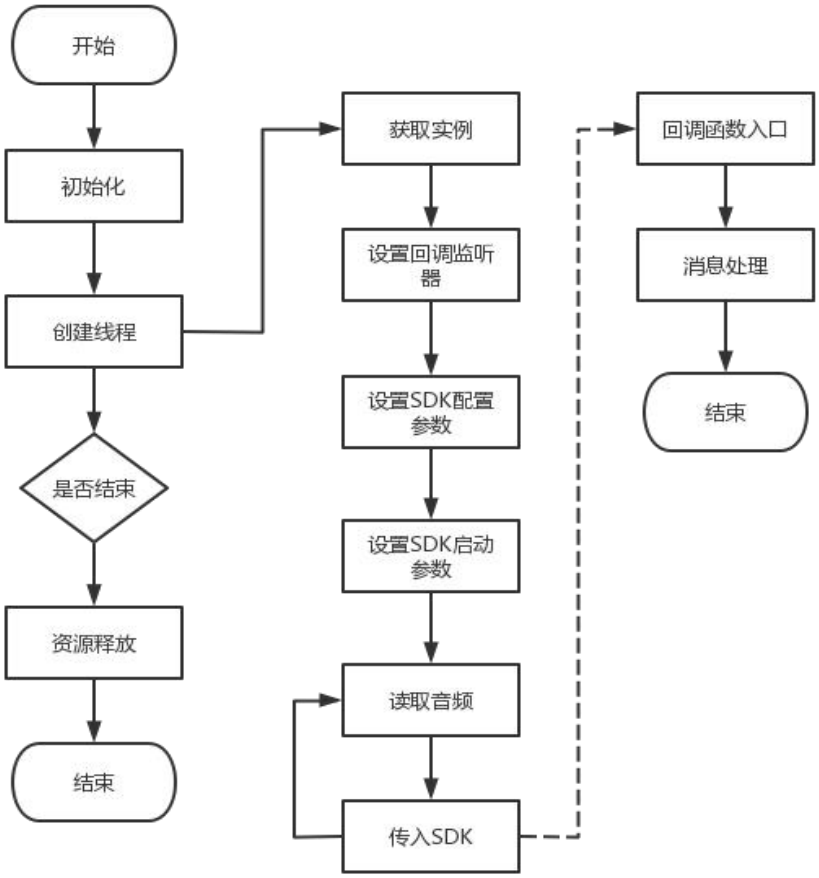

图 19　在线识别流程图

在线识别模块分为两个线程任务和一个回调函数。主线程负责创建识别任务线程并检查识别结束标记，如果识别结束，主线程会结束掉识别任务线程。识别任务主要调用BDSSDKMessage 接口进行识别相关任务。

获取实例接口为 BDSpeechSDK::get_instance(bds::SDK_TYPE_ASR, err_msg);每次识别一个音频流，都需要从获取实例到释放实例完整地执行一遍。即 get_instance 每个音频流获取一次，get_instance 最多可以保持 10 个实例，即最多同时识别 10 个音频。BDSSDK 接口统一使用命令字进行配置，由一个标明意向的 name，及其它参数组成，然后通过 post 函数传递命令。支持的命令字如表 5 所示。

表 5　语音在线识别 BDSSDK 命令字表

命令字	说明
ASR_CMD_CONFIG	设置配置参数
ASR_CMD_START	设置启动参数
ASR_CMD_PUSH_AUDIO	传递音频数据
ASR_CMD_STOP	停止当前当前音频流输入
ASR_CMD_CANCEL	取消当前的整个识别过程

设置回调监听器的任务是当 SDK 有返回结果时调用该函数，设置接口为 sdk->set_event_listener(&asr_output_callback, (void*)& thread_seq);回调产生在 SDK 内部的线程中。SDK 配置参数根据官网提供信息，本系统需要支持普通话和远场识别功能，配置函数为 cfg_params.set_parameter(bds::ASR_CMD_CONFIG, sdk_log_level);选择配置如表 6 所示。

表 6　语音识别输入参数配置表

PID	语言	模型	是否有标点	在线语义
1936	普通话	语音近场识别模型	有标点（逗号）	不支持

设置 SDK 启动参数只需要填写 ASR_PARAM_KEY_APP 参数，填写自定义的应用名称 MyTest，接口为 start_params.set_parameter(bds::ASR_PARAM_KEY_APP, "MyTest")。传递音频数据模块的音频流是从 AECCacheBuffer 获取的，音频流的音频格式是 pcm 输入流，3 声道，16bits，小端序，接口为 push_params.set_parameter(bds::DATA_CHUNK, audio_buf, (int)read_cnt); SDK 音频流已经输入完毕，不再有后续音频。需要调用

push_params.set_parameter()；push_params.set_parameter()。取消识别时需要告诉 SDK 本次识别取消，即用户不再需要识别结果，调用接口为 sdk->post(cancel_params, bds::ASR_CMD_CANCEL)); 在设置的 event_listener 输出回调中，SDK 返回 EvoiceRecognitionClientWorkStatusCancel 事件。识别取消或识别完成需要主动释放 SDK 资源，接口为 bds::BDSpeechSDK::release_instance(sdk);清理所有线程池，所有识别结束，不需要发起新的识别。SDK 空闲时执行清理动作 bds::BDSpeechSDK::do_cleanup()。识别时序如图 20 所示。

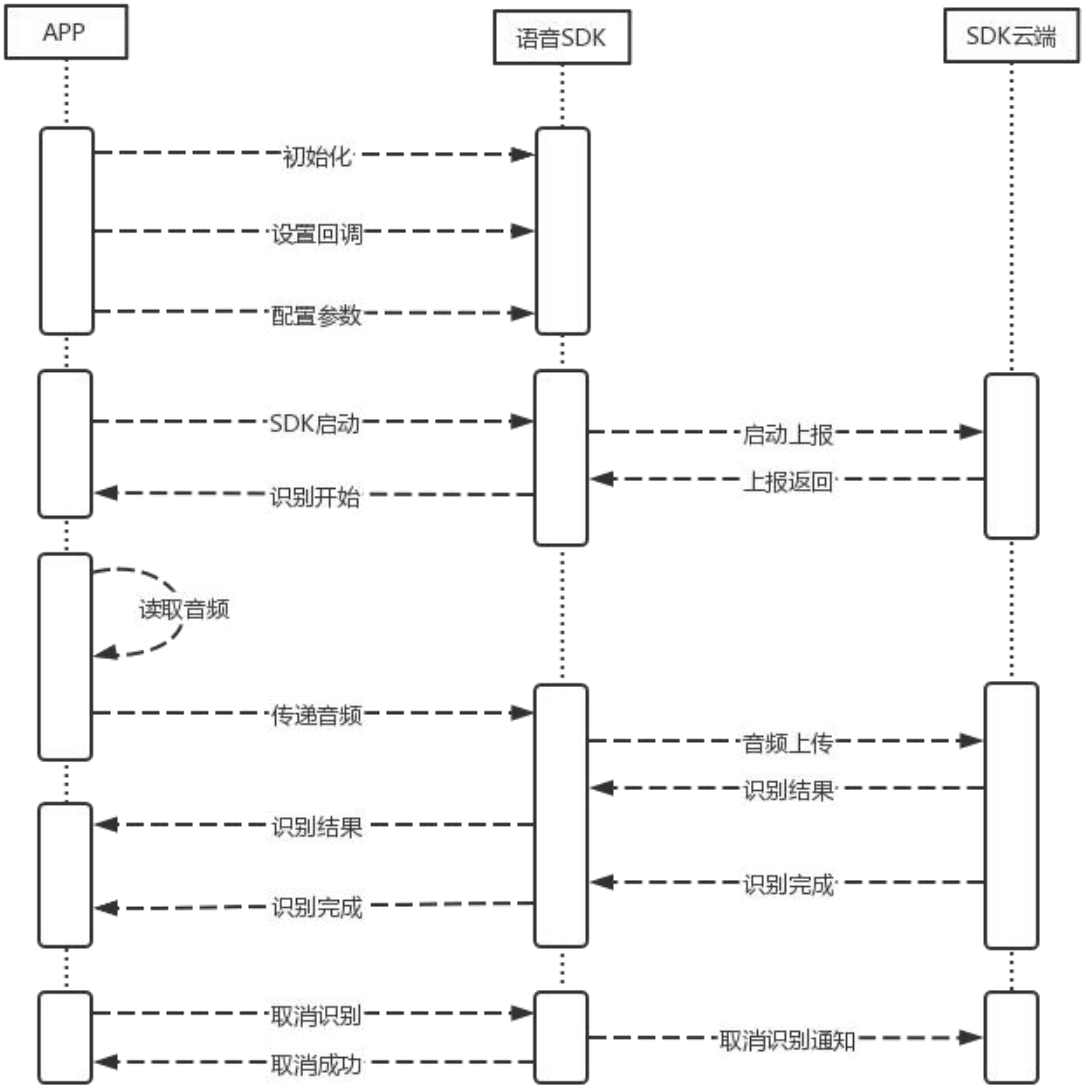

图 20　在线语音识别时序图

4.5 控制功能设计与实现

4.5.1 指令识别

从流程上，语音识别结果出来后需要对识别结果进行解析，解析出用户所发出语音指令的含义。本系统设计支持对蓝牙开关，红外空调，智能电灯的控制，所有支持指令如表 7 所示。

表 7　控制指令表

设备	支持指令
蓝牙开关	打开电视
	关闭电视
红外空调	打开空调
	关闭空调
智能电灯	打开电灯
	关闭电灯
	灯光调亮
	灯光调暗

由于蓝牙开关连接的是非智能电视，本系统将蓝牙开关的指令主体设置为电视。系统需要具有一定灵活性。因此需要将指令进行泛化，如用户的语音指令为“空调关掉”

表 8　控制指令泛化支持表

标准指令	泛化指令
打开电视	电视打开
关闭电视	电视关闭，电视关掉，关掉电视
打开空调	空调打开
关闭空调	空调关闭，空调关掉，关掉空调
打开电灯	电灯打开，开灯
关闭电灯	电灯关闭，关灯
灯光调亮	调亮灯，灯调亮
灯光调暗	调暗灯，灯调暗

指令，系统应具有一定的识别和纠错能力，可以匹配到“关闭空调”指令的动作。控制指令泛化支持如表 8 所示。

识别结果与指令的匹配中，本系统采用两次定向查找加一次不定向查找的方式进行匹配。第一次，在识别结果中遍历标准指令，如果匹配到，则执行指令动作，如果匹配失败，则重新在识别结果中遍历泛化指令，如果匹配到，则执行指令动作，如果匹配失败，则进行不定向指令词匹配。指令匹配流程如图 21 所示。

当标准指令匹配和泛化指令匹配均失败时，系统进行不定向指令匹配，不定向指令匹配是最小化主体和动作的匹配，如“我要打开客厅的灯”，系统根据关键字主体“灯”

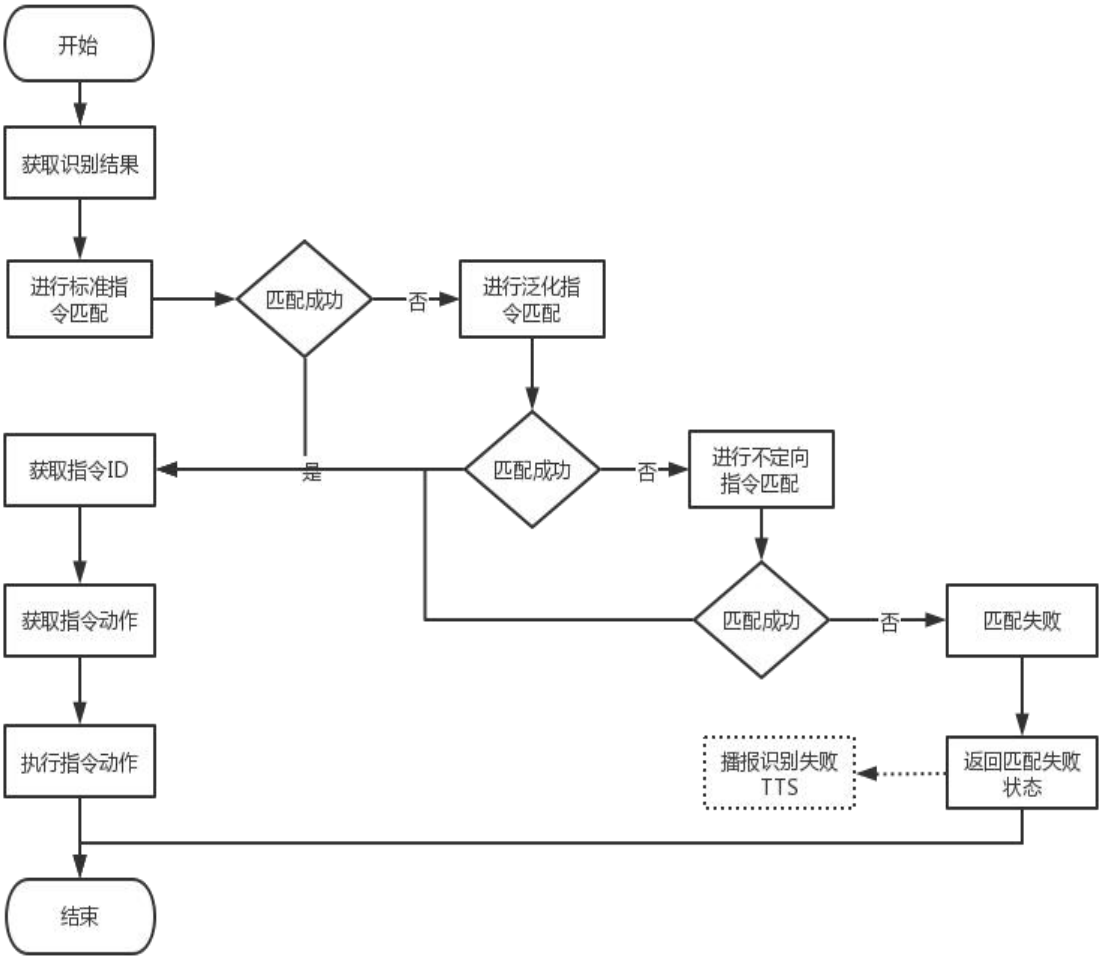

图 21　指令匹配流程图

和关键字动作“开”进行模糊匹配，最小化关键字仍然是以数组的形式存储，控制指令最小化主体动作如表 9 所示。

表 9　控制指令最小化主体动作表

标准指令	最小化主体	最小化动作
打开电视	电视	开
关闭电视	电视	关
打开空调	空调	开
关闭空调	空调	关
打开电灯	电灯	开
关闭电灯	电灯	关
灯光调亮	灯	亮
灯光调暗	灯	暗

4.5.2 指令执行

所有指令包括泛化指令和非定向指令匹配成功后最终返回的结果都对应到标准指令上，每个标准指令对一个控制指令，该控制指令用来执行指令动作，比如“打开灯”指令会向 WiFi 外设控制模块发送打开的指令。标准指令注册了一个指令动作处理函数，当标准指令匹配到后会执行回调函数，控制指令动作对照如表 10 所示。

表 10　控制指令动作对照表

标准指令	外设类型	指令动作
打开电视	蓝牙	command_bt_tv_on
关闭电视	蓝牙	command_bt_tv_off
打开空调	红外	command_ir_airconditioner_on
关闭空调	红外	command_ir_airconditioner_off
打开电灯	WiFi	command_wifi_led_on
关闭电灯	WiFi	command_wifi_led_off
灯光调亮	WiFi	command_wifi_led_lighten
灯光调暗	WiFi	command_wifi_led_dark

4.5.3 蓝牙外设控制

蓝牙可以分为经典蓝牙和低功耗蓝牙，本系统采用低功耗蓝牙 BLE(Bluetooh Low Energy)。BLE 协议栈组成如图 22 所示。

控制器层 Controller 包含了物理层，链路层，HCI 层。物理层是 BLE 协议栈最底层，规定了 BLE 通信的基础射频参数，包括信号频率、调制方案等。BLE4 的物理层是 1Mbps 的 GFSK 调制。LL 层用于控制设备的射频状态，可工作于 advertising、sacnning、Initiating、connecting 四中状态。advertising 和 sacnning 状态是配对出现的，一个设备处于 advertising 状态，就会向外部设备发送广播包，处于 scanning 状态的设备可以接收广播包。设备之间的连接流程为处于 sacnning 状态的设备收到 advertising 设备发来的广播包，然后发送一个连接请求给到这个广播设备，处于 advertising 的设备如果接受了连接请求报文，则两个设备就会进入连接状态，也就是成功建立了连接。HCI 层是一种通信接口，它为主机内报文的通信提供了标准协议，HCI 不仅可以属于硬件接口如 USB，SPI，IIC 等，还可以是软件定义的接口[20]。L2CAP 层对 LL 进行了一次简单封装。

GATT 负责主从设备之间的应用数据交换。GATT 作为使用的 ATT 的子流程的一个服务型框架。为主从设备交互数据提供 Profile、Service、Characteristic 等概念的抽象、

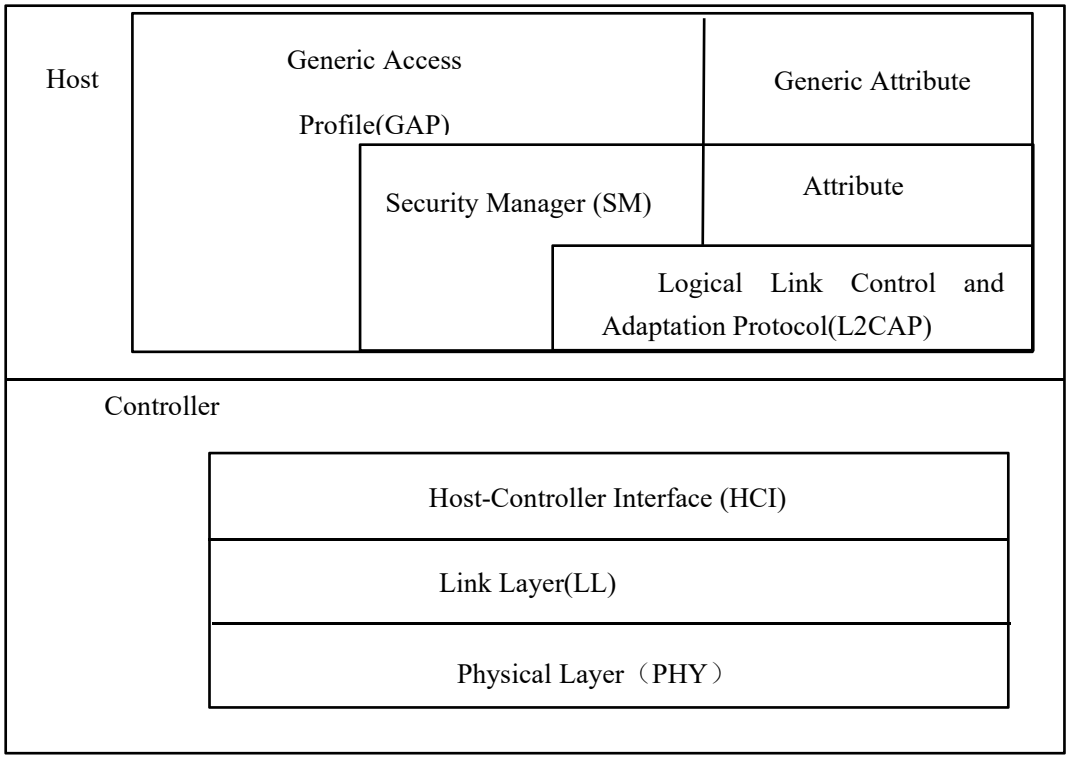

图 22　BLE 协议栈

管理。当两个设备建立连接后，就处于 GATT 服务器或者 GATT 客户端的角色。GAP 是对 LL 层的一部分进行封装，主要用来广播，扫描和发起连接等[21]。Security Manager 属于一种安全方式，通过加密方式进行秘钥分配，该秘钥在配对过程中使用。GAP 层的角色有广播者，观察者，外设和集中器四种。处于 advertising 状态的设备称为广播者，等待 scanning 设备称为观察者，当 scanning 设备发送连接请求，连接建立后，advertising 状态的设备作为从机，scanning 设备作为主机。GATT 层分为客户端和服务器，服务器顾名思义是为客户端提供服务，客户端则是通过服务器提供的服务来获取需要的数据，服务器与客户端相互配合进行通信。本系统的蓝牙插座采用 BLE 进行控制。插座的控制时序如图 23 所示。

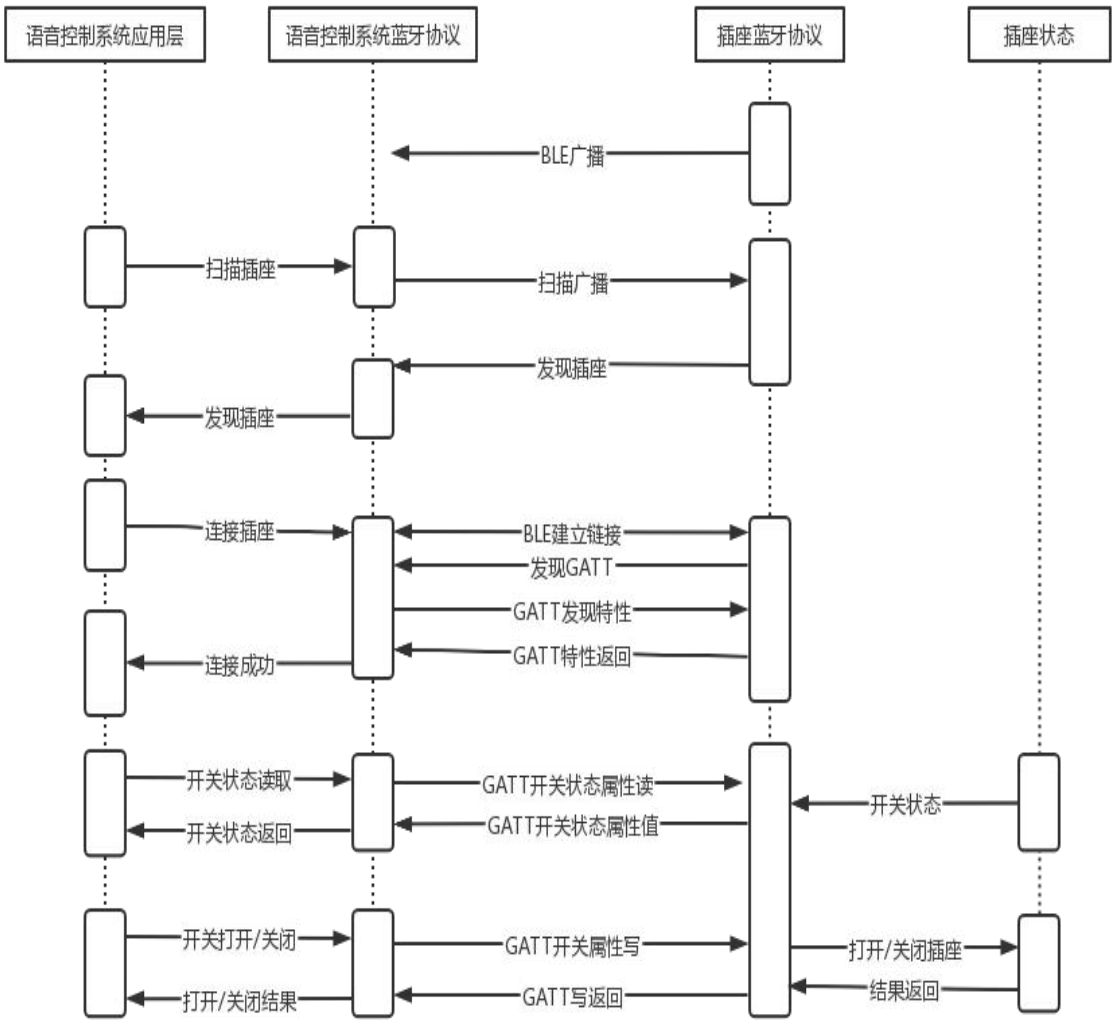

图 23　BLE 插座控制时序图

蓝牙插座是长期带电并且处于广播状态，针对 BLE5.0 协议栈，广播设备利用 37、38、39 主信道和其他的信道作为第二信道向外部发送广播数据，此时并没有指定接收者，所有的处于 scanning 状态的设备都可以接收到该广播数据。广播状态的数据包 pdu 结构如图 24 所示。

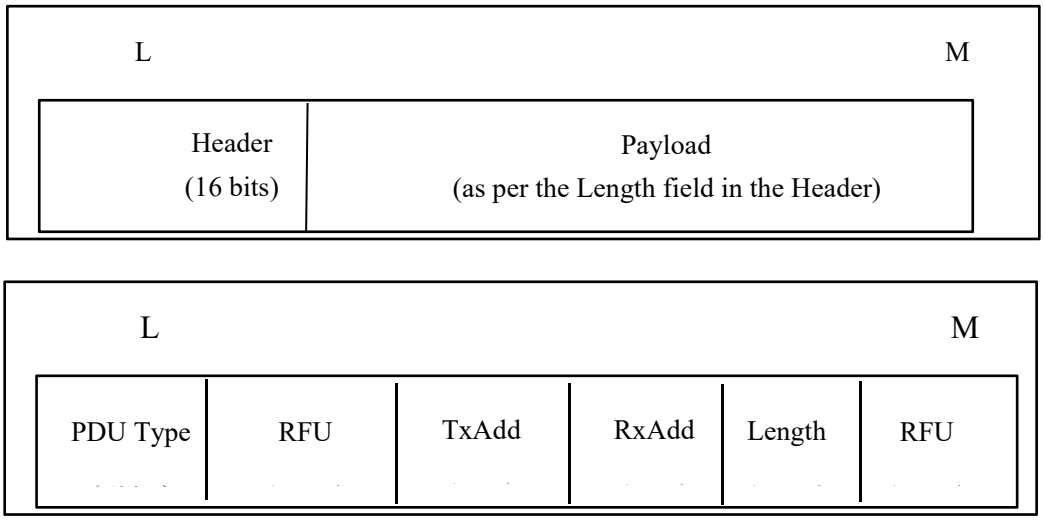

图 24　BLE 广播信道数据表 PDU 组成图

由 16bit Header+payload 组成。4bits PDU Type 决定了广播状态。

基于语音的家居控制系统上电后应用程序调用扫描插座接口，该接口扫描所有 BLE 广播，发现广播中包含蓝牙插座 UUID 的设备，得到返回状态“发现插座”。应用层开始连接插座，通过协议栈建立与插座的 BLE 连接，当两个设备建立连接后，GATT 开始发挥作用，一个设备作为服务器，另一设备作为客户端，连接建立后应用层主要任务就是通过 GATT 服务进行数据的获取和设置[22]。GATT 服务可以包含在一个 GATT 的服务器中，包括的服务有设备开关，电流增大等。插座的开关实际上是 GATT profile 写特性值，控制系统向插座特定的服务地址请求写入要设置的特性值，插座获取设置结果并将数据是否写入成功的信息反馈给控制系统，控制系统即可完成对插座的开关控制。

4.5.4 红外外设控制

本系统使用了红外控制功能，红外控制功能可以控制带有红外接收的家电。一般的

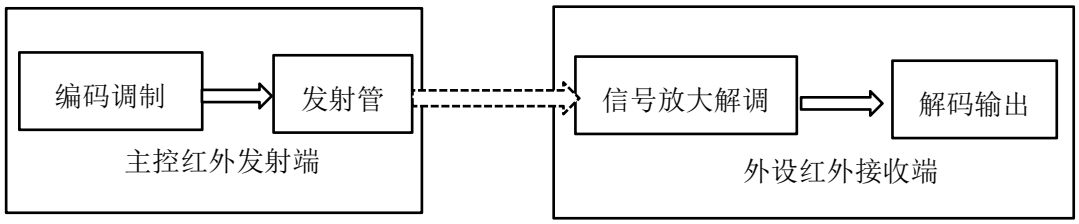

图 25　红外系统结构图

红外遥控系统包含两个部分，分别为红外发射端和红外接收端，通用红外系统的结构图如图 25 所示。

主控的角色就是负责红外信号的发射，红外发射的部分主要包括红外编码调制和发射晶体管。外设的角色是红外信号的接收，主要包括光电信号的转换和解调解码放大等电路。我们平时用的遥控器的信号就是将我们摁下的按键对应的红外码进行编码调制，编码的信号一般调制在 32KHz 到 56kHz 之间的载波上，放大电路将信号进行放大，最后经过驱动电路通过红外晶体管发出，家电设备负责接收并解码识别，执行相应的功能[23]。红外码发射时序如图 26 所示。

红外发射采用了 NEC 编码 nec 编码格式

头节码：8ms 高电平+5.5ms 低电平

节码 0：0.56ms 高电平+0.56ms 低电平

节码 1：0.56ms 高电平+2.86ms 低电平

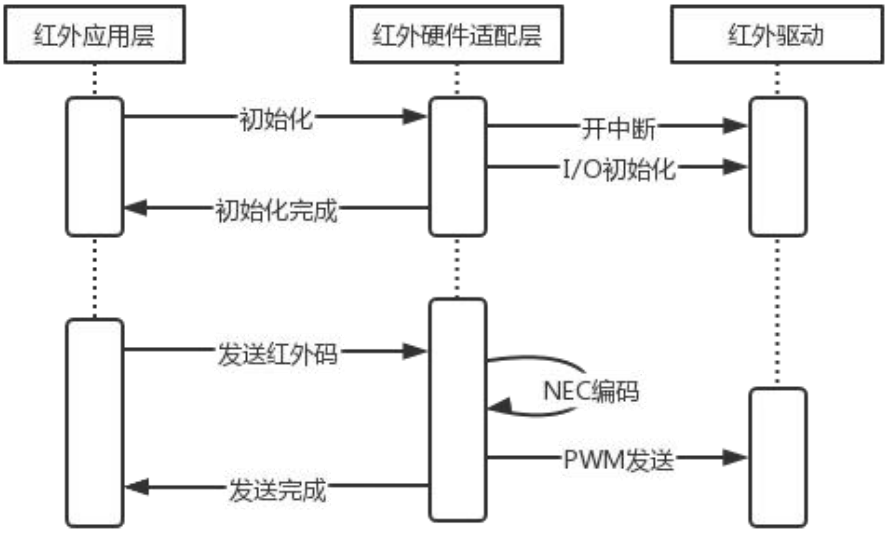

图 26　红外码发射时序图

红外发码顺序为高位最后，先发低位

单键子码：头子码+16 位系统子码+8 位数据子码+8 位数据子码+6 反码

连续键子码：8ms 低电平+1.15ms 高电平+结束子位

简码发送周期：112ms

引导子码+系统子码（16 位）+数据子码（8 位）+数据子码反子码（8 位）+结束码

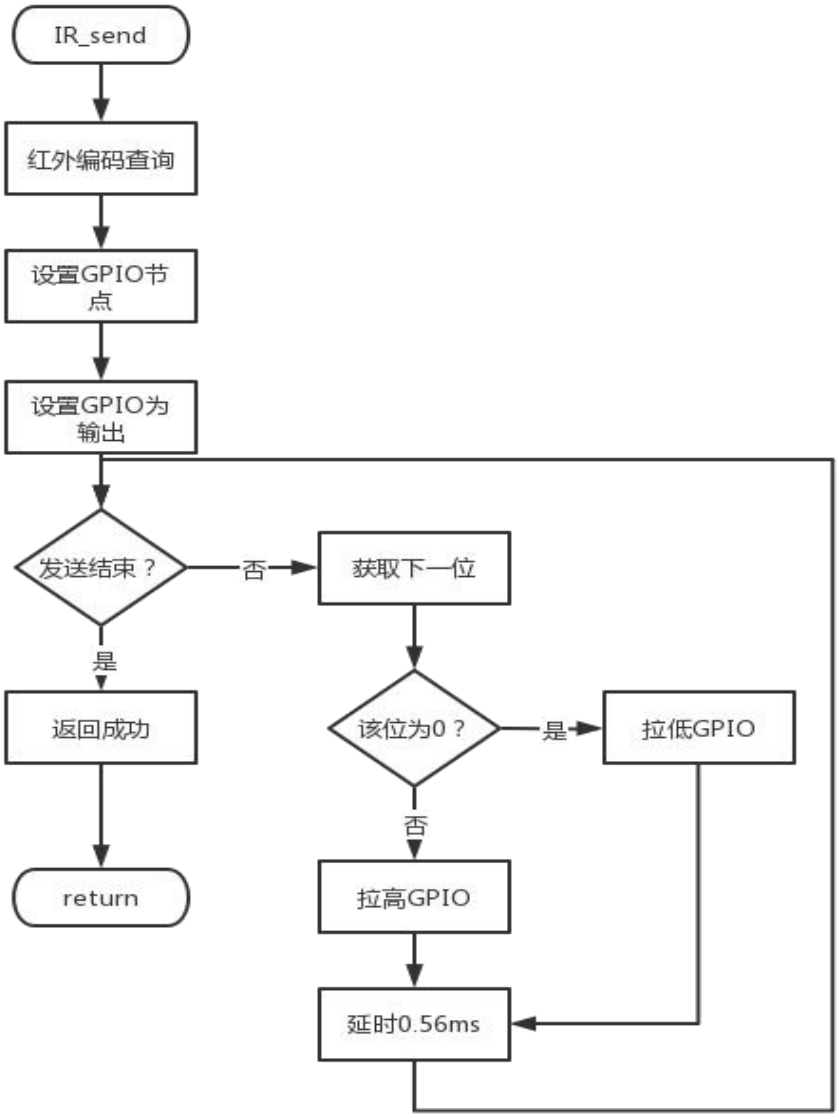

图 27　红外发射接口流程图

全码发送：引导子码+系统子码（16 位）+数据子码（8 位）+数据子码反码（8 位）+结束码。

简码发送：9ms 高电平+2.25ms 低电平+结束位。编码时高低电平以 1，0 表示，红外发送接口流程如图 27 所示。

红外发射接口启动时需要先查询操作对应的红外编码，如“关闭空调”，则需要空调关机的红外编码，本系统控制的美的空调红外指令编码如表 11 所示。

表 11　美的空调红外指令编码

命令	主编码	子编码
开机编码	L,A,A',B,B',C,C'	
关机	S,L,A,A',B,B',C,C'	
风速	B7 B6 B5	自动 101 低风 100 中风 010 高风 001
模式	C3 C2	制冷 00 制热 01 抽湿 10 送风 11
温度	C7 C6 C5 C4	17℃ 0000 18℃ 0001 19℃ 0011 20℃ 0010 ...

GPIO 节点需要通过 Linux 系统 GPIO 驱动接口进行设置，设置命令为 echo 12 > /sys/class/gpio/export，命令成功后生成/sys/class/gpio/gpio12 目录，即为该 GPIO 节点。设置 GPIO 为输出命令为 echo out > /sys/class/gpio/gpio12/direction，只需将 out 字符串写入 direction 即可[24]。

红外发射部分当前系统采用的是 NEC Protocol 的 PWM(脉冲宽度调制)标准。遥控载波的频率为 29kHz，占空比为 1:3，一般的简码重复中的延时为 101 毫秒，也就是两个上升沿中间需要有 98 毫秒的延时。本系统采用的 NTC 编码，是由指令高位编码，指

令字码引导，指令编码键数据，指令反码键数据组成，指令字码引导是 5ms 的间隔关断和 8ms 的波形载波统一时序组成的。红外码的发射会有一个引导码，接收端在接收的时候会先处理接收到的引导码，在后续的处理中能更好的控制引导码与指令编码的数据，控制其检测和各项时序之间的关系。主控发射端编码通过脉冲位置调制的方式，脉冲高点即为 1，脉冲低点即为 0，中间间隔 0.55ms，为了防止传输过程中的误码，传送 8 位数据后会传送一个反码。连续发送两包，第二包总是抓到，两包红外发送不能间隔太短，否则处理不过来，如果第二包不识别，可以加大发送间隔来解决。

4.5.5 WiFi 外设控制

本系统通过 WiFi 通信协议控制外部的智能家电，WiFi 已经在我们生活生产中广泛应用而且具有显著的优点，如覆盖范围广，半径可达 100 米，速度快且可靠性高，无需布线非常适合移动环境使用。WiFi 使用了 802.11 标准，可以在一定范围内将手机，电脑，pad 等移动设备连接起来，是物联网的基础功能。一般的 WiFi 客户端会内置一个无线 WiFi 模组，通过该模组与主控制系统连接通信，内置 TCP/IP 协议簇。当主控系统启动后系统内协议栈会配合开机流程对 WiFi 进行配置，在 Linux 环境下一般通过 wpa_supplicant 软件进行配置，WiFi 模组通过电信号接收和发送信息，接收到的数据会通过内部通信转给主控进入协议栈进行处理。控制协议方面通过调研当前智能家居相关协议，确定选择 MQTT 作为本系统的外设控制协议。MQTT 协议实现方式需要客户端和服务器端通讯完成，在通讯过程中，MQTT 协议中有三种身份：发布者（Publish）、代理（Broker）、订阅者（Subscribe）。其中，消息的发布者和订阅者都是客户端，消息代理是服务器，消息发布者可以同时是订阅者。

MQTT 传输的消息分为：主题（Topic）和负载（payload）两部分。Topic 可以理解为消息的类型，订阅者订阅（Subscribe）后，就会收到该主题的消息内容（payload）；payload 为消息的内容，是指订阅者具体要使用的内容。MQTT 会构建底层网络传输：它将建立客户端到服务器的连接，提供两者之间的一个有序的、无损的、基于字节流的双向传输。当应用数据通过 MQTT 网络发送时，MQTT 会把与之相关的服务质量（QoS）和主题名相关连。MQTT 客户端为使用 MQTT 协议的应用程序或者设备，它总是建立到服务器的网络连接。客户端可以

1）发布其他客户端可能会订阅的信息；

2）订阅其它客户端发布的消息；

3）退订或删除应用程序的消息；

4）断开与服务器连接。作为本系统设计通常这些智能开关可以通过手机 APP 进行控制，本系统正是利用了 APP 的控制接口，模拟 APP 的开关控制。

本系统的外设选择米家智能吸顶灯 Yeelight 作为 WiFi 通信方式的家居外设，程序设计的时序如图 28 所示。

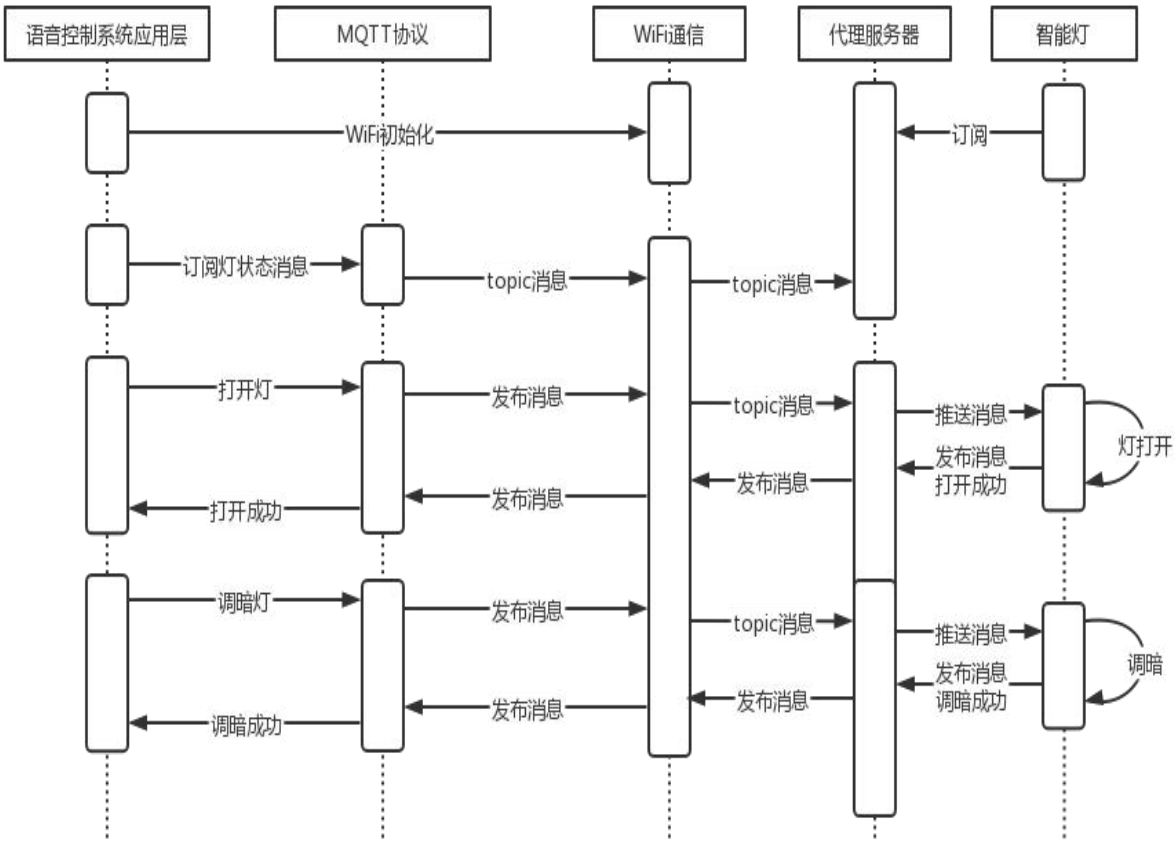

图 28　WiFi 智能灯控制时序图

4.6 语音合成功能设计与实现

本系统语音合成模块使用百度 AI 开发平台提供的语音合成能力，简要步骤如下：

1) 注册成为百度 AI 开放平台的开发者

要调用百度 AI 开放平台的语音合成能力先要成为百度 AI 开放平台的开发者，注册后新建一个百度语音合成应用。然后就能看到创建完的应用和 API KEY 以及 Secret KEY 了。

2）领取免费额度

创建完应用后，可以到概览页领取语音合成的免费额度。

3）准备数据

语音合成是将文本转换为可以播放的音频文件的服务，本系统语音合成文本设计如表 12 所示。

表 12　语音合成文本设计表

设备	支持指令
蓝牙开关	电视已打开
	电视已关闭
红外空调	空调已打开
	空调已关闭
智能电灯	电灯已打开
	电灯已关闭
	电灯已调亮
	电灯已调暗

4）编写程序

编写语音合成应用程序，在线语音合成程序流程如图 29 所示。

安装语音合成 C++ SDK，最低支持 C++11+，使用开发包简要步骤如下

1.在官方网站下载识别、合成 RESTful API C++ SDK 压缩包。

2.将下载的 aip-cpp-sdk-version.zip 解压, 其中文件为包含实现代码的头文件。

3.安装依赖库 libcurl（需要支持 https） openssl jsoncpp(>1.6.2 版本，0.x 版本将不被支持)。

4.编译工程时添加 C++11 支持 (gcc/clang 添加编译参数 -std=c++11), 添加第三方库链接参数 lcurl, lcrypto, ljsoncpp。

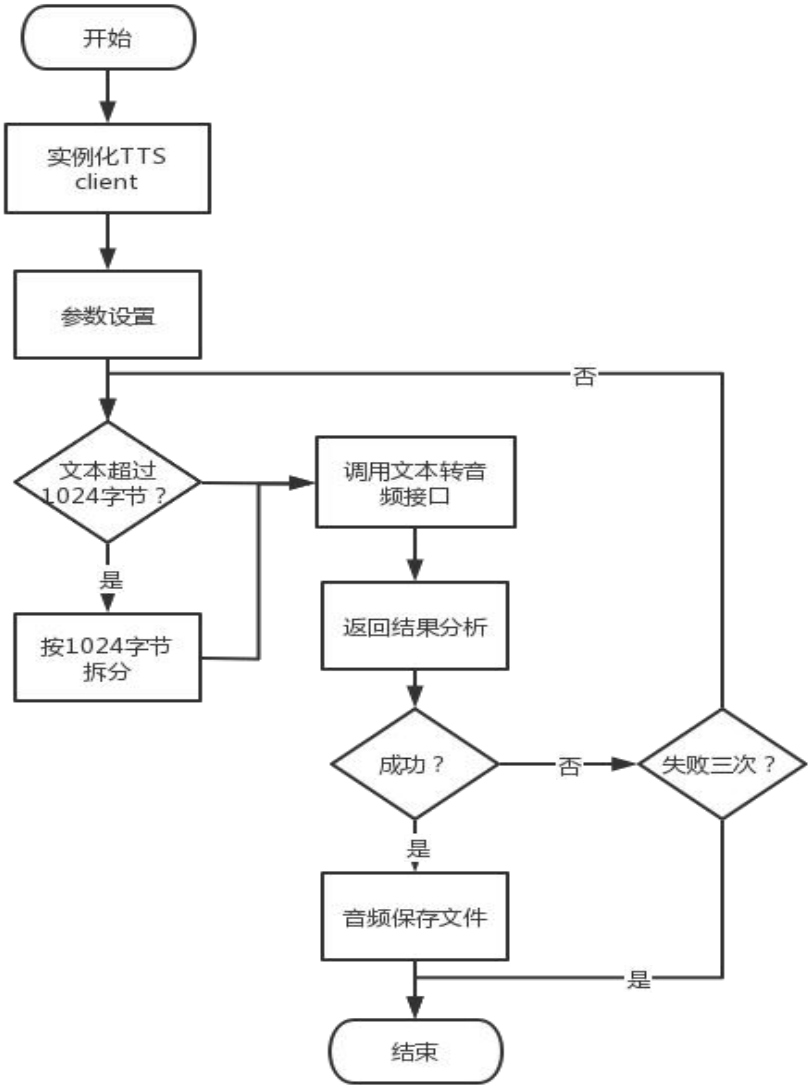

图 29　在线语音合成程序流程图

5.在源码中 include speech.h ，引入压缩包中的头文件以使用 aip 命名空间下的类和方法。

实例化 client 是语音合成的的 C++客户端，申请 APPID/AK/SK 后，使用接口 aip::Speech client(app_id, api_key, secret_key)新建 TTS client，常量 APP_ID 在百度云控制台中创建，接口中需要两个参数 api_key 和 secret_key，这两个参数是开放平台创建语音合成应用后自动分配的字符串，用于授权和校验。参数设置部分可以设置音频的语速，音调，音量，发音人等。在线语音合成参数对照如表 13 所示。

合成文本长度必须小于 1024 字节，如果本文长度较长，可以采用多次请求的方式。把系统所需文字合成为语音文件，成功返回后将数据写入文件进行保存。

表 13　在线语音合成参数对照表

参数	类型	描述	是否必须
text	字符串	待合成的文本数据，字段长度需小于 512 字节	是
cuid	字符串	实例号，是用来区分用户的，系统内唯一，一般使用 mac 地址，字符串长度小于 64 字节	否
speed	字符串	语音发音速度，一般选择 6 作为正常中速	否
pitch	字符串	语音音调，一般选择 6 作为正常中文语调	否
volum	字符串	发音音量，一般选择 6 作为中等音量	否
persion	字符串	系统发音人类型，分为 0 女声，1 男声。一般选择 0 为默认的普通女声	否

返回结果分析模块根据错误码进行分析，如果失败则重新执行，最多尝试 3 次，语音合成错误码及含义如表 14 所示。

表 14　在线语音合成错误码表

错误码	含义
301	不能识别的输入参数
302	输入的参数验证错误
303	合法性验证失败
304	音频合成过程中失败

合成后的音频为 PCM 格式，直接将音频流写入 DMA 缓存区，实现语音播放。

4.7 关键技术和解决方案

本系统的关键技术点有两个，一是做好离线语音识别过程中的降噪和模型训练等部分，二是做好外设控制部分涉及到的通信协议和控制协议。

1）离线语音识别从降噪到算法需要很高的技术门槛，独自解决不太现实，因此选择站在巨人的肩膀上采用相关的开源框架作为解决方案。目前开源世界里提供了多种不同的语音识别工具包，为开发者构建应用提供了很大帮助。但这些工具各有优劣，需要根据具体情况选择使用。在调研了多种语音识别的解决方案，包括 DeepSpeech，Kaldi，

Wav2Letter++，Julius 等开源项目后，通过对比决定采用 Kaldi 作为离线语音识别的核心解决方案，Kaldi 具有可扩展和模块化的特点且包含了丰富的音频降噪算法。

本系统采用的离线语音识别系统需要语言模型和底层的声学模型，两个模型都需要通过训练得到。需要语音样本库进行特征提取，然后通过声学模型训练。语言模型同样需要语音样本库进行语言模型训练。执行流程是语音输入经过特征提取后进行语音解码和搜索再加上词汇模块得到最后的文本输出，一个连续语音识别系统的结构如图 30 所示。

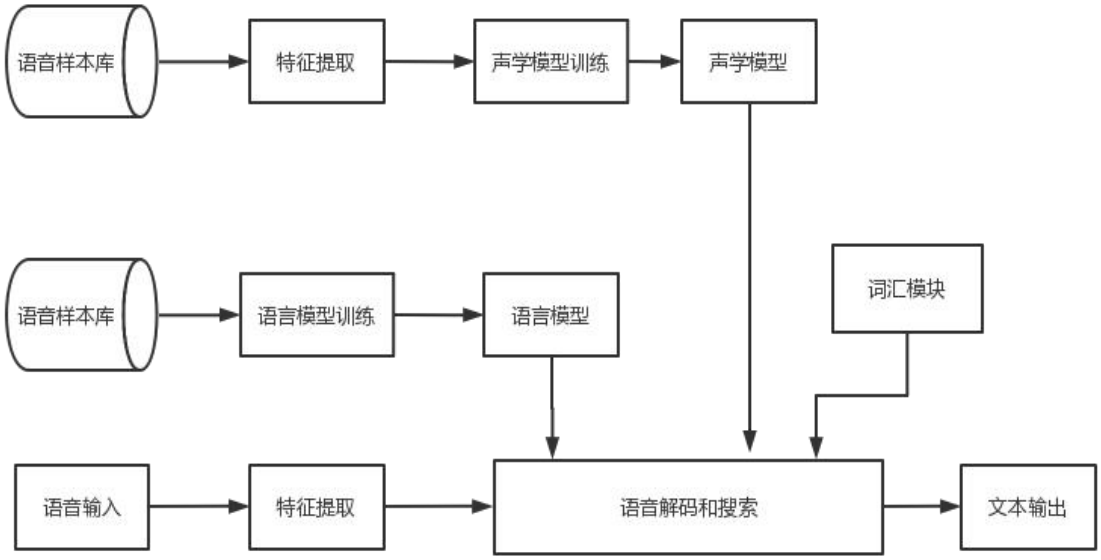

图 30　语音识别系统结构图

3）多外设控制部分除了选择同时支持 WiFi，蓝牙，红外的开发板外，还需要对相关的通信协议和控制协议进行调研开发。物联网家居中使用 WiFi 通信协议的设备最多，在控制协议中应用最普遍的协议是 MQTT。MQTT 使用 C/S 架构和发布/订阅模式，订阅者可以订阅指定主题的消息，代理服务器将指定消息推送给订阅者。发布者可以发布指定消息给代理服务器，服务器接收后进行转发。MQTT 发布/订阅模式结构如图 31 所示。

4）通过 WiFi 的通信协议可以传输 MQTT 控制协议，客户端通过主题消息 topic 进行订阅，订阅后就会收到服务器发送来的消息内容 payload。当系统进行对外设的控制

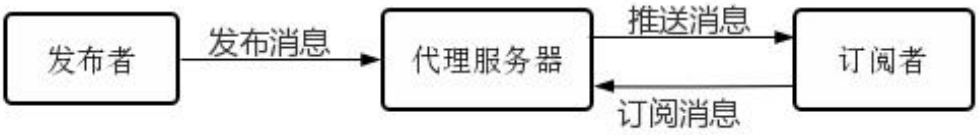

图 31　MQTT 发布/订阅模式结构图

操作时作为发布者进行控制消息的发送，当系统接收外设的消息返回时作为订阅者进行接收，系统控制模块解析 payload 内容，确定外设的准确状态。

4.8 本章小结

本章对整个系统的各个层次和功能模块进行了详细的设计和实现，对控制系统底层进行了环境搭建。对控制层的蓝牙，红外，WiFi 外设进行了设计和实现。音频处理模块详细设计了音频采集和音频 AEC 降噪。唤醒部分采用 snowboy 进行定制并完整应用到系统的唤醒模块中。语音识别模块离线部分采用了 kaldi 开源解决方案，并且集成了百度 AI 开发平台的语音识别功能。应用层对指令识别和指令执行进行了比较详细的设计，并且设计了网络探测模块。最后实现了在线语音合成功能的接入。本章是整个系统设计和实现的核心部分，整个系统的实现也是依托本章内容。

第五章 系统测试分析

5.1 测试概述

本系统的测试是通过将基于语音的家居控制系统组装完整后，进行一些列的功能和性能的测试，目的是为了检验系统的设计成果与需求设计是否一致，进而对系统进行修正，起到一个促进作用。将麦克风，主板，扬声器及相关模组进行整合，烧录制作好的软件，结合成一个整体后，首先进行功能相关的测试，对比需求方案找出所开发过程中系统设计与需求不符合的地方或者矛盾冲突的地方，从而继续完善系统的设计方案。系统测试作为开发的一部分，是为保证基于语音的家居控制系统的正常运行，并在实际操作和运行环境中对基于语音的家居控制系统的严格而有效的测试。而进行系统测试的最终目的也是是为了我们设计的软件系统能够满足需求设计。当前系统测试的主要内容包括：

1）系统功能测试。即测试当前软硬件系统的功能是否按照需求设计正常执行，根据第二章需求设计文档对需求中的相关功能进行一一测试验证。系统本身就是要按需求进行设计的，是软件最重要的质量因素，因此功能测试必不可少。

2）系统稳健性测试。即测试我们的基于语音的家居控制系统能否在异常环境情况下正常运行，软件系统在异常情况下正常运行的能力。健壮性有两个含义：一是容错能力，二是恢复能力。系统测试分为功能测试，对于我们的基于语音的家居控制系统，软件部分除了能实现基本的语音识别和家居控制外未必能够满足相关的性能要求，在每一步的测试中都有对性能相关的点进行考量和测试，以最终达到系统的稳定可靠运行。

5.2 测试工具及测试环境

测试工具：Linux 下录音工具 arecord，程序异常定位工具 gdb 等 gnu 开发套件，网络性能测试工具 netperf，unix 系统性能工具 unixbench，噪音检测手机 APP，WiFi 路由器，蓝牙音响，个人电脑等。

测试环境：唤醒识别测试会从如下几个场景进行测试评估，如表 15 所示。

表 15　唤醒识别测试场景

	一米	三米	五米
噪音类型	安静 音乐噪音 电视噪音	安静 音乐噪音 电视噪音	安静 音乐噪音 电视噪音

由于我们主要的应用场景为家居控制，因此需要基本的家庭环境和一些测试用的必须家居设备，如家用红外空调，蓝牙开关，家用 WiFi 冰箱等。

需要说明的是，对于我们的系统测试场景，需要有噪音检测工具，精度较高的工具价格相对较高，因此采取比较折中的办法，使用低精度的手机噪音检测 APP，这是一款噪音分贝检测应用，通过对环境中的声音进行计算得出具体的实时噪音变化分贝，以提供我们噪音场景下的测试。

本系统对计算和网络能力要求较高，需要有一定的评估手段，因此借用开源工具 Unixbench 对系统运行环境进行计算方面的测试，unixbench 是一个在类 unix 系统上运行的系统性能方面的工具，通过一系列的计算生成一个 benchmark 分数，通过分数可以衡量系统在计算方面的表现。该测试可以对基于语音的家居控制系统提供一个基本的性能指标，这些指标包括如下几点。

- 系统单任务执行方面的性能指标
- 系统在多任务交差运行时的性能指标
- 系统在处理并行任务时的性能指标

本系统的另一个性能关键指标是网络性能，我们借助 Netperf 工具来进行测试。这个工具分为 server 端和 client 端，可以对网络中丢包错报重试等影响网络的因素进行分析和统计，另外它是基于 TCP 协议和 UDP 协议进行的传输可以对协议层进行测试。Server 端作为被测试方，部署在基于语音的家居控制系统上，使用个人电脑作为 client 端，通过 client 端发起请求对 server 所在的家居控制系统进行网络性能测试。

5.3 测试方法及流程

本系统的测试主要包括基本的功能测试和性能测试，还要对可靠性安全性等方面进行全面的测试，有些测试过程可同时包含几个方面，但大部分需要单独设计测试用例。

1）基本的系统功能测试：测试本系统各个子系统和子系统下的功能模块是否能够执行基本的业务逻辑，如唤醒是否正常，识别结果是否正确，指令解析是否配对，外设控制是否生效，外设结果返回是否正确等。

测试方法

●用该系统的路由接口进行录音（安静和噪音的 1、3、5 米）

●将录制的音频通过工具灌入进行识别测试

●根据识别结果利用 wer 工具计算字准和句准

测试流程

音频播放测试也就是播放一条唤醒词之后播放识别 query,使用人声校准音调节调节播放 query 高保真音量为 70dbc。噪音源为电视剧“人民的名义”。将噪音源放在距待测设备 2M150 度并调节音量使得在待测设备出测得噪音音量为 60dbc。记录音箱识别结果计算字准句准。播测用 query：按远场标准对内部识别 100 句。WER（Word Error Rate）是字错误率，是一个衡量语音识别系统的准确程度的度量。其计算公式是 WER=(I+D+S)/NWER=(I+D+S)/N，其中 I 代表被插入的单词个数，D 代表被删除的单词个数，S 代表被替换的单词个数。也就是说把识别出来的结果中，多认的少认的，认错的全都加起来，除以总单词数。

2）系统的性能测试：测试本系统整体的性能，主要包括几个方面，计算性能，网络性能，根据测试数据进行评估，系统的计算及网络的能力是否满足顺畅可用，可支持全部功能的系统的依赖等。

3）系统的可靠性测试：根据当前系统硬件和软件方面的结构和能力，针对基于语音的家居控制系统的压力测试，首先对各个组件包括网络探活，录音，唤醒，离线语音识别，在线语音识别，外设控制，语音合成等模块进行单元化的压力测试，可通过自动化脚本进行辅助以节省人力。集中压力测试，对基于语音的家居控制系统进行整体的压力测试，可通过录制的音频进行播放测试，提前录制好唤醒词加语音控制的音频，在本阶段通过蓝牙音箱进行播放，对系统进行整体的压力测试。真实环境测试环节，在播测等压力测试进行过后，要进行真实的人力环境测试，人声唤醒加语音控制，控制端也由 log 辅助判断结果改为真实家居设备。随机破坏测试环节可适当对网络限速遮挡设备的红外发射模块等。

4）系统的安全性测试：

系统的安全性方面，借助 OpenVAS 漏洞扫描工具进行扫描,因本系统采用了 Linux 系统，基本的安全补丁已经比较完善，针对我们开发的功能进行针对性的模拟攻击,比如 ssh，telnet 等方式登录系统，获取录音文件，获取识别的文字结果，获取指令解析字典等。本文将针对上述的测试方式制定和设计测试用例，更换测试场景，并进行有效的性能和压力测试等手段来进行系统测试工作，以发现更多的系统问题促使系统更加安全完整。

5.4 系统功能测试

针对基于语音的家居控制系统，由于测试场景比较多样，我们先固定下一个最常用的场景，硬件方面需要准备：

●支持红外控制的家居空调

●支持蓝牙控制的智能插座

●支持 WiFi 控制的智能灯

软件方面对相关的唤醒，识别等参数进行预设，根据该环境对系统进行全面测试，本系统测试用软件环境如表 16 所示。

表 16　系统软件环境

参数类型	参数值
voice_pid	385
voice_hi_package	com.hi.intell
voice_hi_server_url	https://hi.baidu.com/intell
debug_hi_switch	1
sdk_hi_version	intell_speech
clicklog_hi_version	2.0.5
event_hi_server_url	https://hi.baidu.com:443/saiya/ws "
outputlog_hi_console	0
clicklog_hi_url	https://hi.baidu.com/saiya/log?
asr_hi_two_in_one	1

表 16 系统软件环境（续）

参数类型	参数值
ais_hi_switch	1
wakeup_hi_dnn	0
voice_hi_debug_log_on	0
voice_hi_speech_sdk_version	3.0.6.15c6ba8-20170714-181721
asr_hi_confirm_request_url	https://hi.baidu.com/ws
request_hi_login_token_url	https://hi.baidu.com:443/saiya/device/token

其中关键参数为voice_pid该参数为产品pid，voice_hi_server_url为识别服务器地址，clicklog_hi_url为打点日志上传地址，出现问题时可通过日志内容进行定位。

1）唤醒功能测试

唤醒功能测试统计如表17所示，其中安静一米距离测试结果中，在安静电视剧场景下唤醒率达到100%，音乐场景下略低但也达到了99%的唤醒率，一米距离各场景下唤醒效果非常好。三米距离各场景下唤醒率均达到98%，效果也较好。五米距离安静场景下唤醒率可以达到95%，电视剧和音乐噪音下也能达到90%以上，可见五米唤醒效果也达到了预期。

表 17 唤醒功能测试统计表

距离	角度	安静	电视剧	音乐
1m	90	100%	100%	99%
3m	90	98%	98%	98%
5m	90	95%	90%	93%

2）识别功能测试

语音识别测试安静场景数据如表18所示。其中一米的字准可以达到98%以上句准在90%以上，三米和五米数据略差但字准也都达到了95%以上，句准也在85%以上，误识别个数也只有五米出现了一次，因此安静场景下符合需求。

语音识别测试电视噪音场景数据如表19所示。其中一米的字准可以达到95%以上句准在86%以上，三米和五米数据差一些，三米字准为92%，句准76%，五米字准69%，

表 18　语音识别播放测试数据-安静场景

	安静			
距离	字准	句准	asr_err	asr_err 个数
1m	98.37%	92.00%	0.00%	0
1m	98.72%	93.00%	0.00%	0
3m	96.97%	87.00%	0.00%	0
5m	95.92%	86.00%	1.00%	1

句准在 24%，误识别个数也只有五米出现了一次。从数据看音乐噪音场景下对识别的字准和句准都有影响，其中影响最大的是五米场景下，误识别的数量也有所增加，但总体还是符合噪音场景的需求。

表 19　语音识别播放测试数据-电视噪音场景

	音乐噪音			
距离	字准	句准	asr_err	asr_err 个数
1m	95.45%	86.20%	0.00%	0
1m	97.55%	89.00%	0.00%	0
3m	92.18%	76.00%	1.00%	1
5m	69.20%	42.00%	1.00%	1

由以上数据可知，语音识别数据满足系统需求，在噪音场景下语音识别字准和句准都有下降，在五米的场景下最低，因为我们系统最常用的距离是 1 米和 3 米，所以五米场景可以后续继续进行优化。

5）外设控制功能测试

外设控制功能测试结果如表 20 所示，分别对应红外空调蓝牙插座和支持 WiFi 的智能灯进行了测试统计，每项功能测试次数均为 20 次，其中红外空调控制部分失败一次，原因是识别错误无法执行相关指令。蓝牙插座有一次打开电视失败，原因是蓝牙出现了

断连，重连后恢复。WiFi 智能灯有一次灯光调暗失败，原因是没有识别到请求语音。综上外设控制功能符合需求。

表 20 外设控制功能测试结果统计表

设备	请求功能	成功次数	失败原因
红外空调	打开空调 20 次	20 次	1 次识别错误
	自动风 20 次	20 次	
	空调制热 20 次	20 次	
	温度 20 度 20 次	19 次	
	关闭空调 20 次	20 次	
蓝牙插座	打开电视 20 次	19 次	1 次蓝牙断连
	关闭电视 20 次	20 次	
WiFi 智能灯	开灯 20 次	20 次	1 次无识别结果
	关灯 20 次	20 次	
	灯光调亮 20 次	19 次	
	灯光调暗 20 次	20 次	

4）性能测试

在本系统上执行 Unixbench，对 Unixbench 数据进行统计，总共执行三次取平均值，收集到的数据如表 21 所示。

表 21 性能测试-Unixbench 数据统计表

序号	score	1x_score	Double	FileCp_1024	FileCp_256	PipeCtxSw	SysCall
1	2998.4	1317.1	2242.0	1962.3	1213.7	2219.5	3500.9
2	3270.3	1525.5	2795.9	2164.9	1371.6	2652.3	3672.9
3	3356.2	1578.8	2805.2	2254.0	1442.0	2681.4	3650.6
平均值	3208.3	1473.8	2614.3	2127.0	1342.4	2517.7	3608.1

Score 是整体跑分评估，这是一个比较综合的指标，直接标识该系统的整体计算性能。由平均值 3208.3 可知系统整体跑分较高满足系统的功能计算需求。Double 数值表示系统在浮点运算方面的性能。这两个指标是系统在内存，计算，编译优化等方面的能力，硬件的设计和软件的架构都会对这两个指标产生影响。FileCp_1024 FileCp_256 这

两个指标是指在文件复制拷贝过程中对于缓存和硬盘存储方面的读写速度的衡量，从平均值 2127.0 和 1342.4 来看，文件处理速度较好，满足系统的执行需求。PipeCtxSw 这个指标是进程间通信中数据交换和执行的效率。SysCall 这个指标是进程执行系统调用进入内核态的效率，系统调用从用户空间到内核空间执行相关的操作后重新返回用户空间，中间有大量的数据和标志进行了更新和交换。

综上，该系统整体性能较高，可满足基本数据和计算需求，多次取平均值后可见，系统性能比较稳定，在均值上下小幅波动，该项满足需求。

Netperf 测试数据统计结果如表 22 所示。

表 22　Netperf 数据统计表

序号	tcp_crr	tcp_band	tcp_rr	udp_band	udp_stream	rtt_avg
1	99622.52	1430.41	483787.06	1478.17	798367.5	0.1
2	120313.95	1448.31	607314.38	1477.33	798171.06	0.1
3	120880.99	1447.9	599643.75	1477.53	798004.38	0.09
平均值	113605.65	1445.82	578234.23	1477.64	798201.44	0.096

tcp_crr 表示在一次 TCP 链接中只进行一组 Request/Response 通信即断开，并不断新建 TCP 链接时的响应效率。tcp_crr 在 Web 服务器访问中较为普遍。tcp_band 表示 TCP 进行批量数据传输时的数据传输吞吐量。tcp_rr 表示在同一次 TCP 长链接中进行多次 Request/Response 通信时的响应效率。tcp_rr 在数据库访问链接中较为普遍。udp_band 表示 UDP 进行批量数据传输时的数据传输吞吐量。udp_stream 表示 UDP 进行数据传输时整体数据流速。rtt_avg 网络 ping 时延。

综上，系统的网络性能较好，由于受网络本身波动影响，测试过程中会有一定的数据波动，但整体波动不大，该系统的网络性能符合需求。

5.5 系统运行效果评估

本系统硬件包括主控板腔体扬声器等部分组成，主控板硬件如图 32 所示。

图 32　系统主控板硬件展示图

系统整体结构如图 33 所示。

图 33　系统整体结构展示图

通过唤醒词“小贾小贾”唤醒，如表 23 是智能灯，蓝牙插座，美的空调的控制日志。

表 23　系统运行日志表

外设	唤醒日志	语音指令	识别日志	控制日志
智能灯	INFO:Keyword 1 detected at time:2021-11-11 12:11:22 wakeup word 小贾小贾	调亮灯	asr finish content={"**调亮灯**":"err_no":0}	"Result":"control successfully"
蓝牙插座	INFO:Keyword 1 detected at time:2021-11-11 12:21:31 wakeup word 小贾小贾	打开电视	asr finish content={"**打开电视**":"err_no":0}	"Result":"control successfully"
美的空调	INFO:Keyword 1 detected at time:2021-11-11 12:32:20 wakeup word 小贾小贾	打开空调	asr finish content={"**打开空调**":"err_no":0}	"Result":"control successfully"

用户通过 web 对系统进行网络配置界面如图 34 所示。

图 34　系统网络配置界面图

本系统从各个方面的测试情况来看，整体实现度和表现都比较高，其中功能测试运行稳定，各功能模块运行符合需求和预期。性能方面从测试结果和人为体验上效果均达预期。

5.6 本章小结

本章对系统测试进行了整体概述，对测试工具及测试环境进行了研究和部署，进一步的对系统的测试方法和流程进行了设计，在系统功能测试部分，对系统的功能进行了详细的测试，并对系统性能进行了测试和分析。通过对测试结果分析，可以得到系统的整体运行情况，并且可以满足用户的需求。

总结与展望

1.总结

前面章节对基于语音的家居控制系统进行了比较详细的设计和实现，通过最后的测试和分析后我们实现了预期的效果。在系统的设计过程中除了要解决几个关键问题，同时要从更高的视角把握整个系统，在系统需求分析时对整个系统进行了功能和非功能的需求划分，简单明了的设立了我们的目标。总体设计部分构建了整个系统的设计蓝图，将整个系统拆解为四个子系统，又将子系统拆解为功能模块，在后续的详细设计中对于关键功能分别采用了流程图和时序图进行表现。在最后的测试章节，本文论述了系统测试的详细过程，对应功能需求进行测试并展示了测试报表，比较直观的展现了测试的结果得到了有效的结论。

基于语音的家居控制系统的开发过程中主要以 Linux 系统为基础，加以语音唤醒和离在线语音识别和外设控制等技术形成了一个完整的控制系统。语音唤醒和离线语音识别都依赖于人工智能技术，保证了系统的精准唤醒和高识别率等非功能需求。控制部分支持红外 WiFi 蓝牙通信接口几户涵盖了所有家居设备，外设控制协议我们使用了通用的 MQTT，蓝牙 BLE 等使得开发更加高效，保证了系统的可维护性和安全性。

2.展望

基于语音的家居控制系统在实现后可以充分体验到在无网络的情况下仍然稳定可靠，并且可控制绝大部分家居设备，对于一些非标准设备暂时还无法做到完全控制，需要厂家配合或开放相关 API 接口，这也是后续商业化最重要的部分。在后续的工作中可以搭配一个移动端控制软件，这样可以扩大使用范围，通过 APP 随时随地的控制该系统对家居设备进行操控。

参考文献

[1]钟浩, 鲍鸿, 张晶. 一种改进的语音动态组合特征参数提取方法[J]. 电脑与信息技术. 2017(03)

[2]田阳, 王泽宇. 无线终端设备低功耗唤醒方法的研究范[J]. 中文信息学报, 2018, 16(5):51-66.

[3]李文凤, 徐及, 张鹏远. 基于状态后验概率的语音唤醒识别系统[J].计算机工程, 2017, 31(10): 177-179.

[4]张庆芳. 基于语音识别的灯光控制系统[J]. 计算机应用, 2014, 24(1):14-16 天大学出版社, 2015.

[5]Rabiner L R. Control Method for Smart Home System[J]. Proceedings of the IEEE, 2016, 77(2): 257-286.

[6]冼进, 许振山, 刘峥嵘等.基于 DTW 和 HMM 算法的语音识别系统对比研究[M]. 北京:电子工业出版社, 2017.

[7]周立功. 基于汉语语素对 RNN 语音识别系统的改进与研究[J]. 淮海工学院学报, 2012,21(3):16-19.

[8]易克初, 田斌, 付强. 基于 DNN-HMM 模型的语音识别的语音导航系统[M]. 北京:国防工业出版社, 2012.

[9]蔡莲红, 黄德智, 蔡锐. 一种 GPU 及深度置信网络的语音识别加速算法研究[M]. 北京: 清华大学出版社, 2013.

[10]赵力. 连续音素的改进深信度网络的识别算法[M]. 北京: 机械工业出版社, 2019.

[11]张晓丹,黄丽霞,张雪英.关于在噪声环境下语音识别优化研究[J]..计算机仿真..2016(08)

[12]何英, 何强. 一种基于层次结构深度信念网络的音素识别方法[M]. 北京: 清华大学出版社, 2012.

[13]杨行峻, 迟惠生. 基于 TTS 技术的外挂式语音报警软件开发与应用[M]. 北京:电子工业出版社, 2015.

[14]李虎生. 利用 TTS 技术开发放射科语音播放系统[D]. 北京:清华大学, 2012.

[15]俞栋, 邓力, 俞凯. 减少驾驶人语音唤醒指令词误触发的方法及装置[M]. 计算机工程,2014,30(9): 94-96.

[16]王志国. 基于 MQTT 的物联网系统文件传输方法的实现究[D]. 中国科学技术学, 2018, 31(4):524-529

[17]田国会, 许亚雄. 基于飞思卡尔 MCU 的 AEC 算法实现述[J]. 山东大学学报 (工学版), 2018, 44(6): 47-54.

[18]薛胜尧. 基于 ROS 的智能语音交互系统设计与实现[J]. 电子设计工程, 2015 (4):78-81

[19]王帅. 智慧家庭环境下语音交互系统的设计与实现[D]. 华中科技大学. 2019

[20]Mohamed A, Dahl G, Hinton G.Development and analysis of Punjabi ASR system for mobile phones under different acoustic models. [J]. IEEE Transactions on audio, speech, and language processing, 2012, 20(1): 30-42.

[21]基于 HMM 模型语音识别系统中声学模型的建立[J]. 胡石, 章毅, 陈芳, 陈心怡. 通讯世界. 2017(08)

[22]范兴隆. ESP8266 在智能家居监控系统中的应用[J]. 单片机与嵌入式系统应用. 2016(09)

[23]刘军, 钟毅, 刘天成. 一种基于蓝牙的智能家居控制系统的硬件设计[J]. 网络安全技术与应用. 2020(05)

[24]Myers C, Rabiner L, Rosenberg A. Voice Control Smart Home System [J]. IEEE Transactions on Acoustics, Speech, and Signal Processing, 2018, 28(6): 623-635.

致 谢

在即将完成我的研究生论文的时候，我发现北航三年的研究生学习生涯就要结束了。离开熟悉的校园，亲爱的老师和同学，重新开始崭新的人生旅途，心里有许许多多的舍不得，也有许许多多的感谢要说。

首先感谢我的母校，北京航空航天大学，自从我踏入学校的第一步，我就对学校产生了深深的敬意，磅礴的新主楼，时尚的图书馆，还有浓厚的学习氛围，非常荣幸能够考入这所大学，感谢学校给我了在这里继续学习的机会，感谢学校提供的环境和平台，让我结识这么多认真负责且知识渊博的老师和一群优秀可爱的同学，让我看到了更广阔的世界。

其次，我要衷心感谢我敬爱的导师王丽华老师！是您一直以来对我的学习和研究的悉心指导，才使得我能够克服种种困难完成毕设课题。您严谨治学的作风和诲人不倦的态度是我不断学习的榜样。王老师非常注重学生在实际工程能力方面的培养，在您的指导下，我在各方面的能力都得到了提高，使我在论文完成过程中更加得心应手。在我论文写作最困难的时候，是您的专业分析和悉心帮助使我克服难关，不断前进，终于守得云开见月明。在此对您致以最诚挚的敬意!

最后，感谢我的父母家人。每当我遇到困难的时候，父母总是第一个给我鼓励的人。回顾走过的路，每一个脚印都浸满着他们无私的关爱和谆谆教诲，研究生期间，我一直处于半工半读的状态，工作上巨大的压力和学习任务分去了我大部分的时间，家人不但没有抱怨而且作为我强大的后盾支持我克服各种困难。他们在精神上和物质上的无私支持，坚定了我追求人生理想的信念。家人的爱是天下最无私的爱，惟有以永无止境的奋斗和一生的行动来回报。