课程设计报告

一、课设题目

ESP32语音交互系统

二、功能要求

1. 实现录音以及放音功能。
2. 实现WIFI连接功能。
3. 实现与大模型的对话。
4. 实现可充电功能。

三、设计方案

3.1 硬件方案

1. 主控：主控使用ESP32-WROOM-32UE物联网芯片，该芯片支持高达240MHz的时钟频率，可为复杂的音频信号处理提供硬件支持。WIFI功能在802.11n模式下数据速率高达150Mbps，能大大缩短系统响应时间。
2. 麦克风：麦克风采用数字I2S输出麦克风ICS-43434,其64dBA的高信噪比和宽带频率响应，灵敏度容差为正负1dB，无需系统校准即可实现高性能麦克风阵列。
3. 音频功率放大器：音频功率放大器采用MAX98257。其是一款易于使用、低成本的数字脉冲和编码调制（PCM）输入D类放大器。支持I2S数据传输。
4. 电源稳压器：采用RT9013芯片。其是一款高性能的500mALDO稳压器，具有极高的电源抑制比以及超低的压降，非常适合具有苛刻性能和空间要求的便携式射频和无线应用。
5. 锂电池充电模块：采用MCP73831T-2ATI-OT芯片。其内置恒压/恒流充电算法。

下图为原理图：

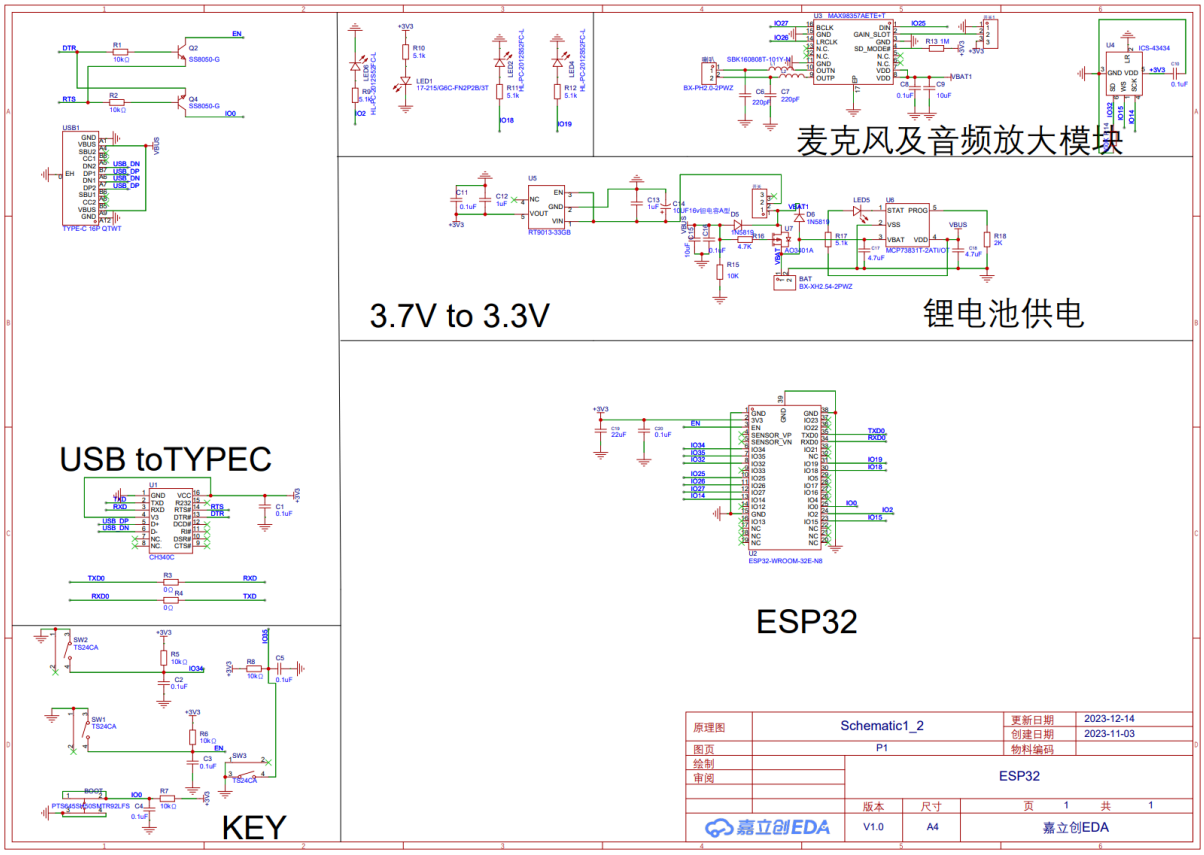

图3-1 原理图

下图为pcb设计图，pcb采用四层板设计:

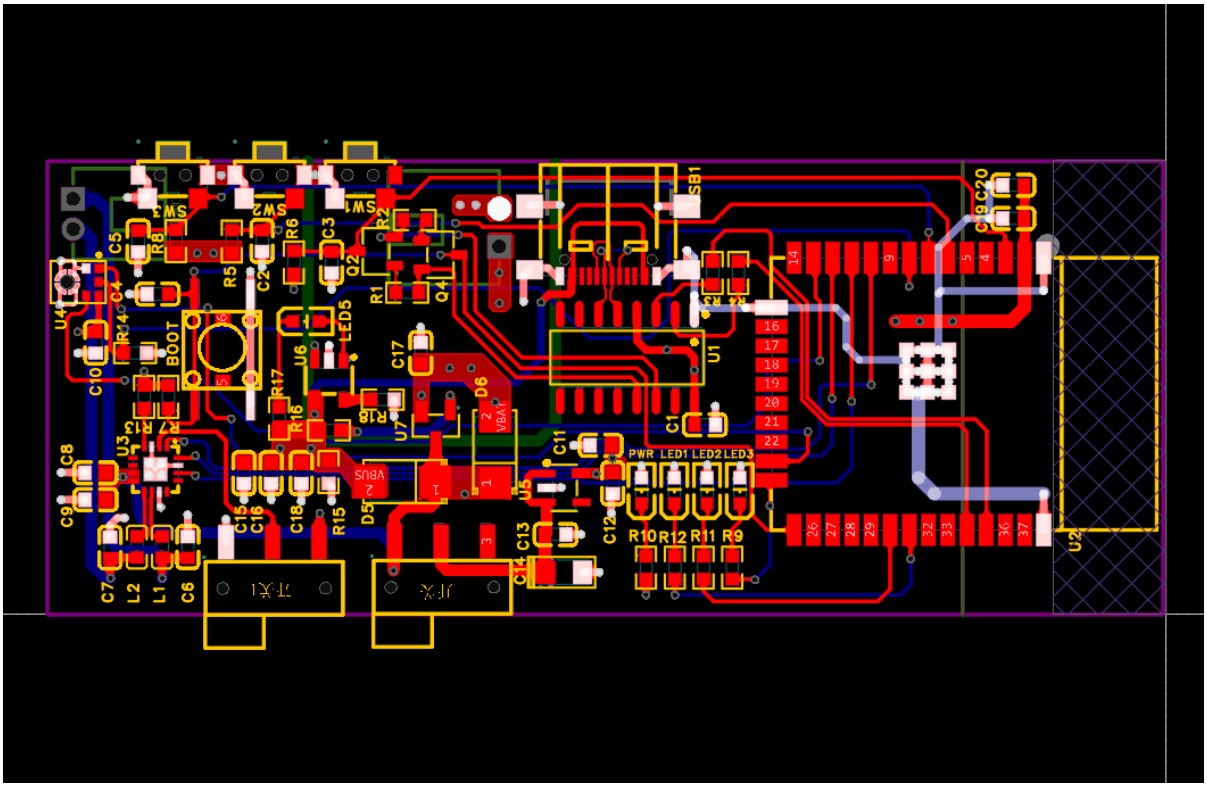

图3-2 PCB图

下图为硬件系统总框图：

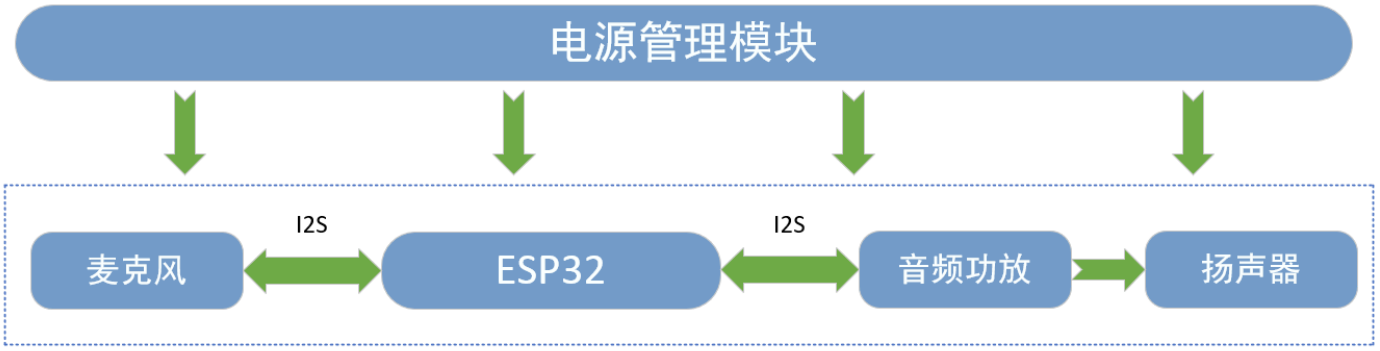

图3-3 硬件系统框图

3.2 软件方案

3.2.1 编程语言

系统开发采用了Arduino编程语言，Arduino是一种基于C++的开源硬件编程语言，广泛应用于微控制器编程。Arduino语言的简洁性和丰富的库支持，使得系统开发过程更加高效。

3.2.2 通讯协议

1. WebSocket：WebSocket是一种网络通信协议，它提供了一个全双工通信渠道，客户端和服务器可以随时向对方发送数据，不需要等待对方的请求。一旦建立了WebSocket连接，它会保持开放状态，直到客户端或服务器决定关闭连接。这种协议特别适合需要实时数据传输的应用，如在线游戏、实时聊天和推送通知等。

   由于esp32内部集成flash非常小，最多只能存储2-3s的音频信号。利用WebSocket技术，可实现麦克风采集的音频数据实时发送到语音识别服务，解决了只能录2-3s的问题，大大提升了用户体验感。
2. HTTP: HTTP（超文本传输协议）是互联网上应用最广泛的协议之一，基于TCP/IP，用于分布式、协作式、超媒体信息系统。它定义了客户端和服务器之间的请求/响应模型。可用于大部分简单的应用。

3.2.3 数据交换

系统在数据交换过程中采用了JSON（JavaScript Object Notation）格式。JSON是一种轻量级的数据交换格式，易于人阅读和编写，同时也易于机器解析和生成。系统通过ArduinoJson库处理JSON数据，确保了数据的准确性和可靠性。

3.2.4 服务器

本系统共连接了三个服务器，分别为讯飞语音识别服务器，讯飞星火大模型以及百度语音合成服务器。

1. 讯飞语音识别服务器（stt）：语音识别技术实际上是stt（speak to text）技术。即将音频信号转化为文本。讯飞的语音识别准确率在业界领先，尤其是在普通话识别上。且还支持英文识别。
2. 星火大模型： 星火大模型是科大讯飞推出的一款认知智能大模型，它集成了自然语言处理等多种AI能力，能够提供更加智能的语音交互体验。
3. 百度语音合成服务器（tts）:语音合成技术实际上是tts（text to speak）技术。即将文本转化为音频信号。百度的TTS技术支持多种语言和方言，以及多种角色声音选择。可通过简单的http协议进行连接。为考虑时间和成本，本系统选择百度语音合成技术。

3.2.5 系统工作流程

系统的工作流程如下：

1. 初始化：系统启动后，初始化各个接口。
2. WiFi连接：系统尝试连接到预设的WiFi网络，只有成功连接才能进行下一步。
3. 获取服务器时间：系统通过HTTP请求获取服务器时间，用于后续的鉴权和时间同步。
4. 生成WebSocket URL：根据获取的时间和其他参数，生成用于WebSocket连接的URL。
5. 主循环：系统进入主循环，轮询和处理WebSocket消息，处理播放音频、按键检测等任务。
6. 语音识别：当检测到按键按下时，系统通过WebSocket连接到语音识别服务器，开始录音并发送音频数据。最后将语音识别服务器将解析后的文本结果存储在askquestion中。
7. 对话大模型：系统将askquestion发送至星火大模型进行语义理解，并将理解后的文本回答存储在Answer中。
8. 语音合成：系统将Answer发送至百度语音合成服务器，最终通过扬声器播放合成的语音。

图3-4为系统数据流图，3-5为程序流程图：

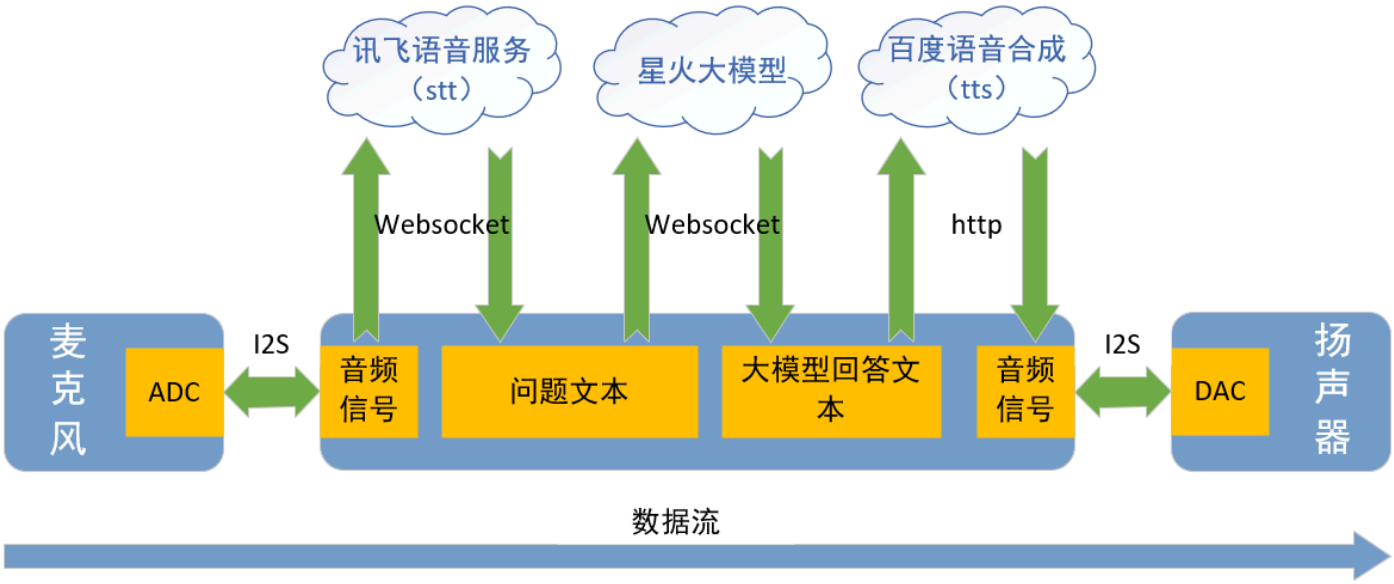

图3-4 数据流图

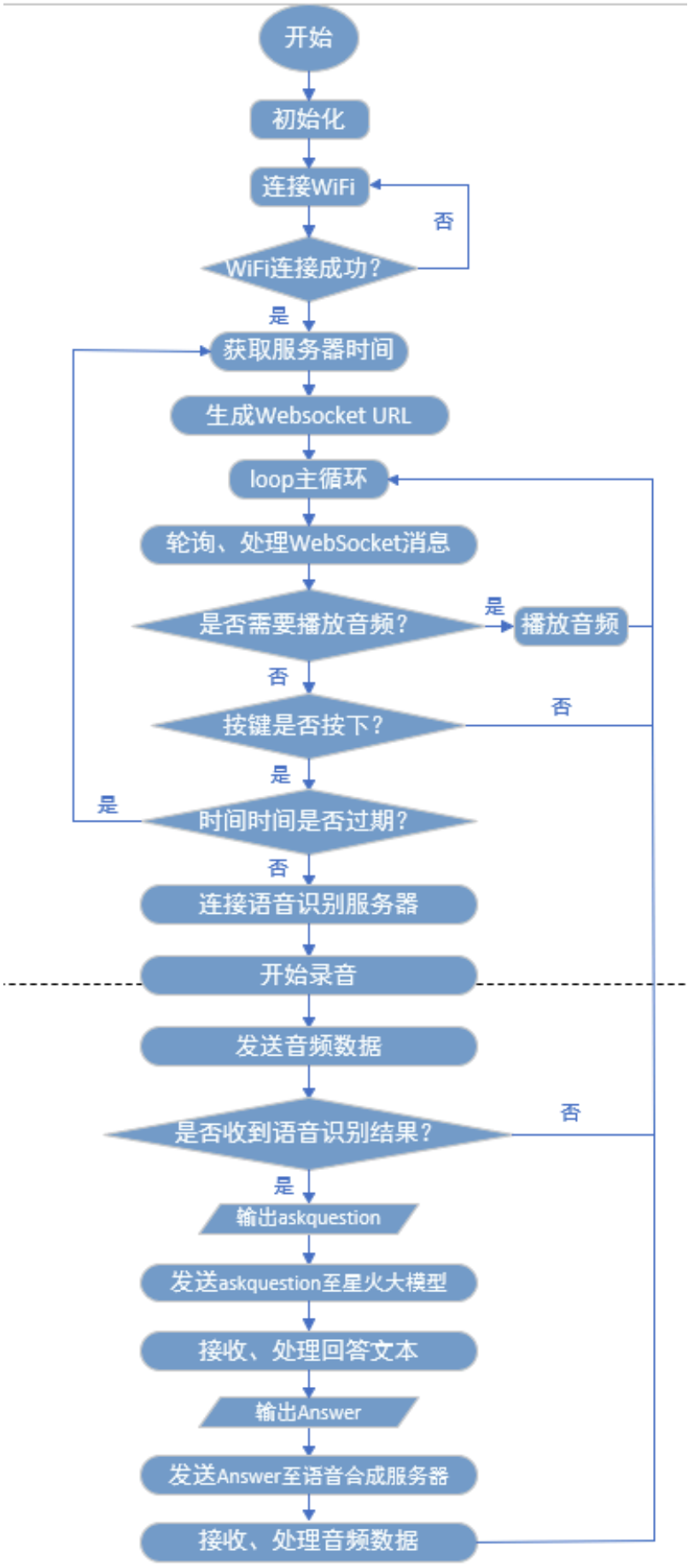

图3-5 流程图

3.2.6 语音识别程序设计

1. 检查连接状态：系统首先检查与语音识别服务的连接是否成功。因为Websocket建立的是持续的连接，系统必须要处于连接状态才可正常工作。
2. 创建JSON文件：系统创建一个JSON文件，用于封装语音数据和相关参数，以便发送给语音识别服务器。
3. 录音：系统开始录音，采集音频数据。
4. 计算平均音量值：对录音数据进行均方根值计算，便以判断是否处于说话状态。
5. 数据帧处理：将计算出的平均音量值与预设的阈值进行比较。如果平均音量值大于阈值，则判断是否为语音第一帧，是则标为首帧，否则标记为中间帧；如果平均音量值小于阈值，系统进行大约0.6s的等待，若期间没有检测到说话的声音，则判定为讲话完毕。将刚刚的语音数据标记为尾帧。以上三种帧状态的处理流程均为一个个的数据帧，每处理完一种状态则发送一帧数据至服务器。
6. 尾帧数据发送完毕后，系统退出录音和语音识别流程。

下图为语音识别程序设计图：

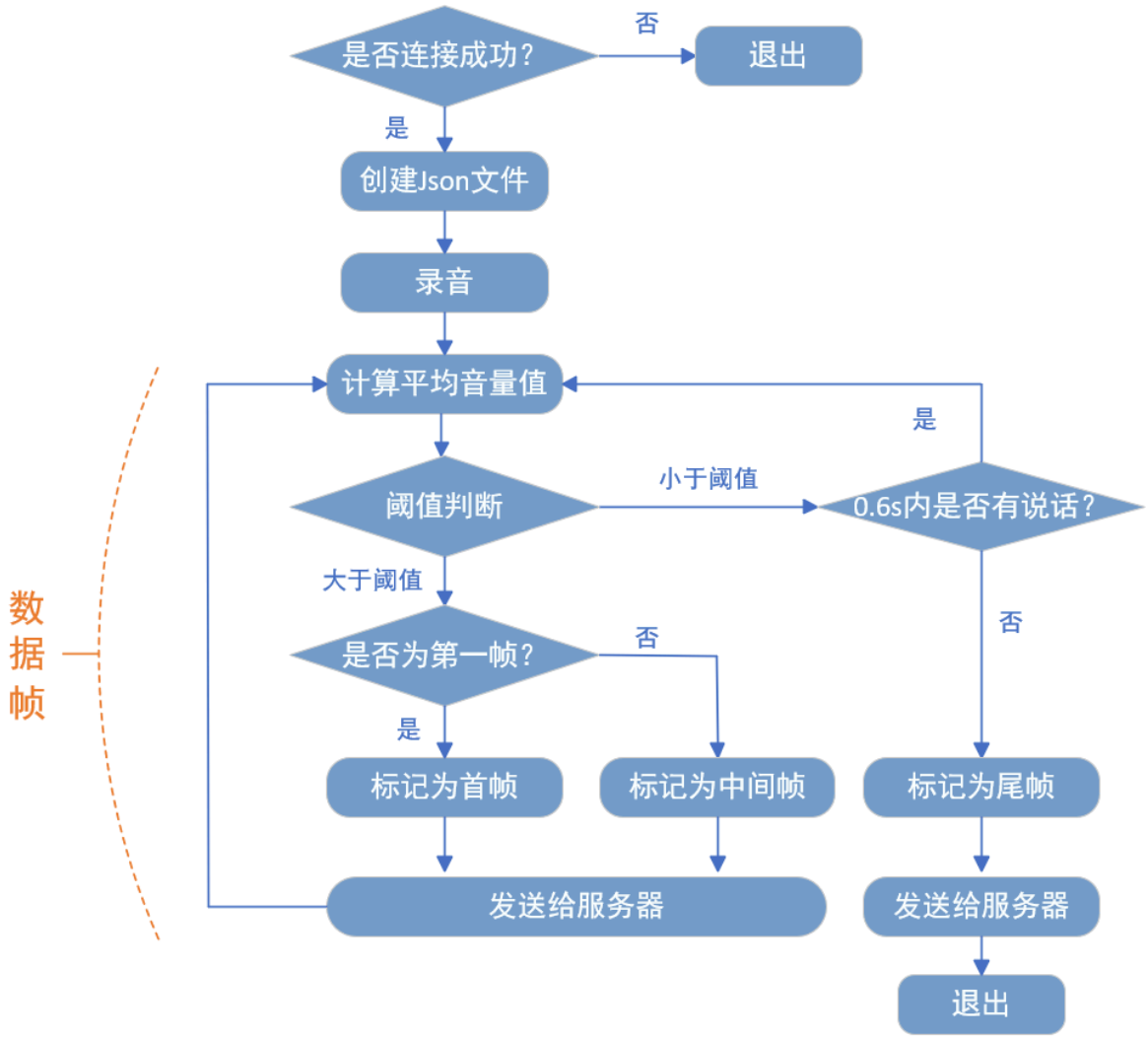

图3-6 语音识别程序设计图