**ChatChatty AI聊天机器人**

贺靖1

摘 要：ChatChatty AI聊天机器人的灵感来源于对人工智能技术的热爱和对智能交互体验的追求。随着AI技术的快速发展，我们希望探索AI在日常生活中的应用，特别是在提供陪伴和娱乐方面的潜力ChatChatty AI聊天机器人这是一款基于ESP32平台的智能交互设备，集成了先进的语音识别、处理和合成技术。ChatChatty的创新之处在于其结合了MiniMax AI大模型，为用户提供了具有情感韵律的语音交互体验，使得日常交流更加生动、愉快和智能。项目的核心设计考虑了用户体验，通过网页控制台和表情显示屏增强了人机互动，同时在硬件选择上平衡了成本、兼容性和易用性，确保了系统的高效和稳定。ChatChatty不仅是一款智能设备，更是一个能够提供温馨交流和娱乐互动的伙伴。

关键词：ESP32；AI；MiniMax

# 1. 项目背景与起源

### 1.1 灵感来源

ChatChatty AI聊天机器人的灵感来源于对人工智能技术无限潜能的深刻认识和对提升人类交互体验的持续追求。在当今数字化时代，人工智能正迅速成为推动社会发展的关键力量，其在语音识别、自然语言处理和机器学习等领域的突破，为创造更加智能、个性化的交互体验提供了可能。

我们的灵感还来自于对人类情感交流的深入理解。交流不仅传递信息，更是情感连接的桥梁。ChatChatty旨在通过模拟人类情感反应，使用户与机器的互动更加自然、温暖，仿佛与一个真正的伙伴对话。此外，随着技术的发展，人们对于个性化、智能化产品的需求日益增长。ChatChatty的构想正是在这样的背景下应运而生，它不仅是一款技术产品，更是一种新型的陪伴方式，无论是解答疑问、提供信息，还是简单的闲聊，都能给用户带来愉悦的体验。

# 制作原理

ChatChatty AI聊天机器人的制作原理基于集成先进的语音识别和合成技术，通过MAX9814麦克风模块进行对对语音信号的实时捕捉和处理。利用百度的TTS服务将语言转化为文字，在通过MiniMax AI大模型的语言能力，提供情感丰富的对话体验。

硬件方面，项目采用了TFT圆形屏幕展示动态表情和信息反馈，结合MAX98357音频放大器提供高质量的音频输出，以及物理按键进行交互控制。软件层面，使用Arduino框架，软件与硬件紧密结合，实现了传感器数据的读取、语音信号的精细处理、网络通信的稳定性以及用户界面的直观展示。屏幕驱动和显示的优化，进一步确保了系统的流畅运行和高度交互性。

### 2.1 材料与工具

在构建ChatChatty AI聊天机器人的过程中，我们精心挑选了一系列材料和工具，确保了项目的高效实现和卓越性能。对于ESP32开发板作为项目的大脑，它提供了强大的计算能力和丰富的接口，在这里我们使用性能较强的ESP32S3N16R8型号，而屏幕我们使用圆形TFT屏，与动画表情的展示完美契合。此外，MAX9814麦克风模块和MAX98357音频放大器进行语音输入和输出。使用物理按键进行交互操作，这些和必要的连接线构成了项目的硬件基础。开发环境使用platefromIO作为主要的编程环境，使用Arduino框架C++编程。配合百度TTS服务和MiniMax AI大模型，赋予ChatChatty自然语言处理和语音合成的能力。

以下是ChatChatty AI聊天机器人项目主要材料的表格展示：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 名称 | 型号/规格 | 数量 | 用途描述 | 价格 |
| 1 | ESP32开发板 | 最好使用ESP32S3N16R8 | 1 | 主控器，处理语音和网络通信 | 29元 |
| 2 | MAX9814麦克风模块 | MAX9814 | 1 | 捕捉语音输入 | 8.3元 |
| 3 | MAX98357音频放大器 | MAX98357 | 1 | 放大输出语音信号 | 5.9元 |
| 4 | TFT屏幕 | 圆形屏幕 | 1 | 显示动态表情 | 12元 |
| 5 | 物理按键 | 根据设计选择 | 1 | 用户交互控制 | 0.2元 |
| 6 | 喇叭 | 可自由选择，一般腔体越大，声音越大 | 1 | 语音播放 | 1~10元 |
| 7 | 连接线 | 适当长度和类型 | 按需 | 连接各个模块和开发板 | 3元一捆 |



图2.1材料图片

## **2.2 设计考虑**

在设计ChatChatty AI聊天机器人时，我们深思熟虑地选择了适合的组件和方法来提供最佳的用户体验。我们选用了TFT圆形屏幕来显示丰富多彩的表情和反馈信息，为用户互动增添了视觉上的吸引力。这种设计选择使表情显示更加直观和生动，有助于传达机器人的"情感"状态，从而增强用户的情感参与度。在主控选择上，我们注重性能与稳定性，挑选了适合的 ESP32S3 微控制器和高质量的麦克风和音频放大器组件。

## **2.2.1 主控模块**

ChatChatty AI聊天机器人的主控部分由ESP32S3开发板担纲，它以强大的双核处理能力、内置Wi-Fi和蓝牙模块、丰富的外设接口以及低功耗设计，成为整个系统的核心。ESP32S3不仅高效地执行语音识别和网络任务，同时还通过SPI接口无缝连接TFT屏幕和音频放大器，实现用户界面显示和音频输出。此外，ESP32有IO MUX（输入/输出多路复用）功能，可以选择任意 GPIO管脚作为外设的信号线，控制各种输入输出设备，而ESP32所支持的arduino框架还拥有广阔的开发社区和库支持，进一步简化了开发过程，确保了ChatChatty在提供智能交互体验的同时，也能保持系统的稳定性和可扩展性。

下表为ESP32S3的主要参数描述

|  |  |
| --- | --- |
| **特性** | **详细描述** |
| CPU | 双核 Xtena LX7 微处理器，频率最高可达 240 MHz，支持双精度和单精度浮点运算及向量处理器。 |
| 内存 | 512 KB SRAM 和 384 KB ROM，支持外部 SRAM 和 PSRAM 扩展。 |
| 存储 | 集成 16 MB SPI flash 和 8 MB PSRAM（具体型号可能有所不同）。 |
| Wi-Fi | 支持 802.11 b/g/n（2.4 GHz），速度最高 150 Mbps，支持 Station、SoftAP 和 P2P 模式。 |
| GPIO | 多达 45 个通用输入输出引脚，支持多种功能复用。 |
| ADC | 12 位 SAR ADC，最多支持 20 个通道。 |
| I2C | 支持 2 个 I2C 接口，适用于传感器和其他外部设备的连接。 |
| SPI | 提供 4 个 SPI 接口，其中 **2** 个为高速 SPI 接口。 |
| UART | 提供 3 个 UART 接口，支持硬件流控和波特率检测。 |

### 2.2.2TFT屏幕

屏幕显示方面TFT屏幕在ChatChatty AI聊天机器人项目中扮演着至关重要的角色，它不仅是信息的重要组成部分，也是展示机器人"情感"和反馈信息的窗口。以下是TFT屏幕显示部分的详细介绍：

TFT屏幕（Thin Film Transistor Liquid Crystal Display，薄膜晶体管液晶显示器）以下是TFT屏幕的技术规格，屏幕尺寸我们使用1.28寸以匹配机器人的面部设计，确保表情显示既清晰又不过于庞大。分辨率也足够高，以保证文本和图像的清晰度，我们使用240x240像素，足以提供清晰的视觉效果。而对于颜色深度，决定了屏幕能显示的颜色数量我们使用全彩位色显示，即能够展示1677万种颜色（18位色深），这使得表情和动画色彩丰富，生动逼真。详细参数看下表：

|  |  |
| --- | --- |
| **参数** | **值** |
| 尺寸 | 1.28英寸IPS LCD |
| 材料 | TFT LCD |
| 分辨率 | 240(RGB)x240 |
| 通信接口 | 4-SPI |
| 工作电压 | 3.3V |
| 驱动芯片 | GC9A01 |
| 显示颜色 | 全彩 |
| 面板尺寸 | 35.6(H) x 38.1(V) x 1.5(D) |
| 显示区域 | 直径32.4mm |
| 点间距 | 0.135(H) x 0.135(V) |
| 背光 | 2 CHIP White LED |
| 工作电流 | 20mA |
| 管脚数量 | 焊接(12Pin, 0.7mm 隙距 FPC 排线) |

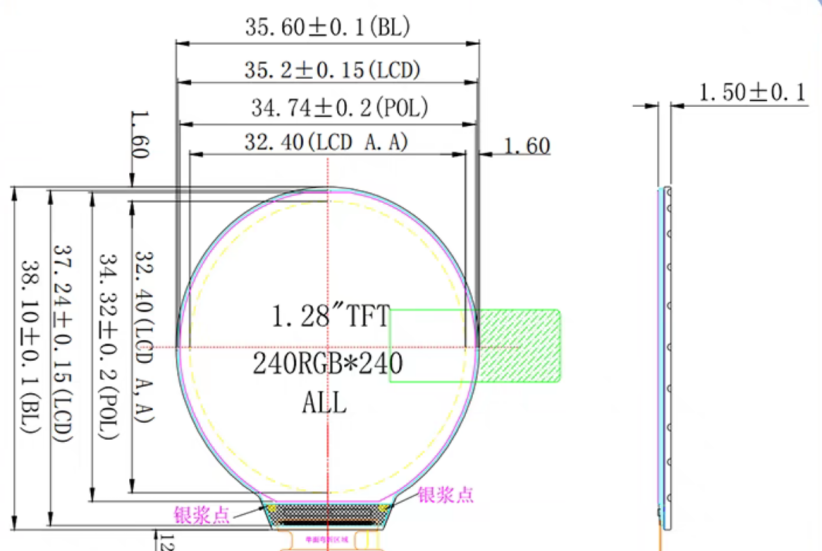


图2.2 TFT圆形屏幕尺寸

同样显示屏驱动的选取和连线配置是确保TFT屏幕顺利集成和操作的关键步骤。我们采用了GC9A01作为显示屏的驱动IC，它以出色的显示性能和广泛的兼容性被选中，为项目提供了稳定可靠的显示控制。针对ESP32-S3这一特定型号的开发板，我们定义了详细的引脚分配方案。

总而言之，圆形TFT屏幕与人类表情的自然形状相契合，使得机器人的"情感"表达更为直观，也利用了屏幕形状与表情之间的和谐关系，增强了情感交流的效果。同时屏幕不仅用于展示动画表情，还可以显示文本信息、系统状态和其他视觉反馈。

### 2.2.3 MAX9814麦克风模块

MAX9814麦克风模块是一款高性能的MEMS（微电子机械系统）麦克风，专为捕获清晰的语音和音频信号而设计。它以其高灵敏度和宽动态范围著称，能够在各种环境条件下提供出色的音频捕捉能力。MAX9814具备集成的噪声抑制功能，有效降低背景噪音，确保语音输入的清晰度。此外，它的宽电源电压范围和多种输出格式选项，使其能够灵活地集成到不同的硬件设计中。在ChatChatty AI聊天机器人项目中，MAX9814作为核心音频输入设备，为实现高质量的语音识别和交互提供了基础。

### 2.2.4 MAX98357音频放大器

MAX98357是一款高度集成的数字输入音频放大器，具备I2S、TDM和PDM输入接口，支持广泛的输入格式，能够提供高保真的音频输出。它以其出色的性能特性，如高信噪比、低失真度、可编程增益控制以及节能特性，成为了ChatChatty AI聊天机器人项目中的首选音频放大器。在设计中，它不仅能驱动外部扬声器以提供清晰的语音输出，还通过I2C接口轻松实现配置和控制，确保了与ESP32开发板的无缝集成和操作的灵活性。

### 2.3搭建电路

**下面列出所有使用的模块接线：**

|  |  |  |  |
| --- | --- | --- | --- |
| **模块** | **模块引脚** | **ESP32 引脚** | **功能描述** |
| 按键 | KEY | 3 | 用户交互，触发语音识别执行操作 |
| max9814麦克风 | OUT | 2 | 捕捉模拟声音信号，转换为数字信号供处理 |
| max98357扬声器 | I2S\_DIN | 6 | 传输数字音频信号至音频放大器 |
| I2S\_BCLK | 5 | 同步数据传输，控制音频信号的时钟 |
| I2S\_LRC | 4 | 控制音频数据流的左右声道时钟 |
| TFT 显示屏 | TFT\_SCLK | 18 | SPI 时钟信号，控制显示屏数据传输 |
| TFT\_SDA | 17 | SPI 主设备输出，从设备输入，数据信号 |
| TFT\_MISO | 未使用 | SPI 主设备输入，从设备输出，未使用 |
| TFT\_RST | 16 | 复位显示屏，初始化使用 |
| TFT\_DC | 15 | 数据/命令控制，区分传输的是数据还是命令 |
| TFT\_CS | 14 | 芯片选择控制，激活显示屏 |
| TFT\_BLK | 13 | 背光控制，开启或关闭显示屏背光 |
| LED 指示灯 A | LED\_A的长脚（正极） | 10 | 提供视觉反馈，指示设备状态 |
| LED 指示灯 B | LED\_B的长脚（正极） | 11 | 提供视觉反馈，指示设备状态 |

请注意表格中没有明示标出 "VCC" 和 "GND" ，而各模块的"VCC" 和 "GND"都需要需要连接到 ESP32 的相应电源和地线引脚上。除了max98357扬声器模块的 "VCC" 引脚应连接到 ESP32 的 "VCC"（5V），其他所有模块的 "VCC" 引脚应连接到 ESP32 的 "3V3"（3V） 引脚。

### **连线步骤**

1. 按键连接：将按键的一端连接到 ESP32 的3号引脚，另一端接地。
2. max9814麦克风连接：将麦克风模块的输出端（ ADC\_PIN）连接到 ESP32 的2号引脚。
3. max98357扬声器连接：将扬声器的数据输入端（  I2S\_DIN）连接到 ESP32 的6号引脚，并连接 I2S\_BCLK到5号引脚 以及  I2S\_LRC到4号引脚以同步音频信号。
4. 显示屏连接：将 TFT 显示屏的 SPI 接口连接到 ESP32 的相应 SPI 引脚，并将控制引脚连接到 TFT\_RST、TFT\_DC、TFT\_CS 和 TFT\_BLK。
5. LED 连接（开发板上自带就不需要额外准备）：将两个 LED 的长脚（正极）分别连接到10号引脚 和11号引脚，短脚（负极）通过限流电阻接地。

**根据原理图搭建电路，****整体接线如图2.3：**

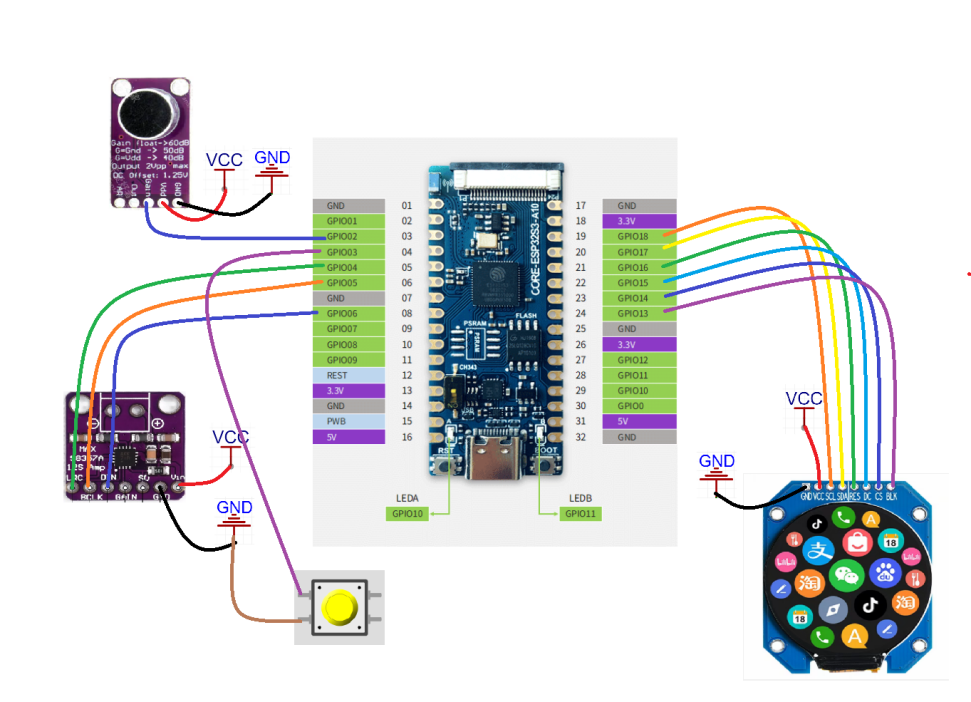


图2.3 整体接线

# 3 总体设计

## **3.1总体设计思路**

ChatChatty的系统框架图包括语音输入、处理、输出和显示四个主要部分。设计思路是实现一个闭环的交互系统，用户可以通过按钮激活机器人，进行语音对话。ChatChatty AI聊天机器人的总体设计就像搭乐高一样，我们采用了模块化的设计方法，这样不仅让整个系统的构建变得灵活，而且也方便了未来的升级和维护。

在设计的过程中，我们汲取了互联网上现有的一些创客项目，同时也不断地自我革新，以确保我们的机器人在功能和体验上都能给大家带来新鲜感。

**下表为此项目的总体设计框架：**

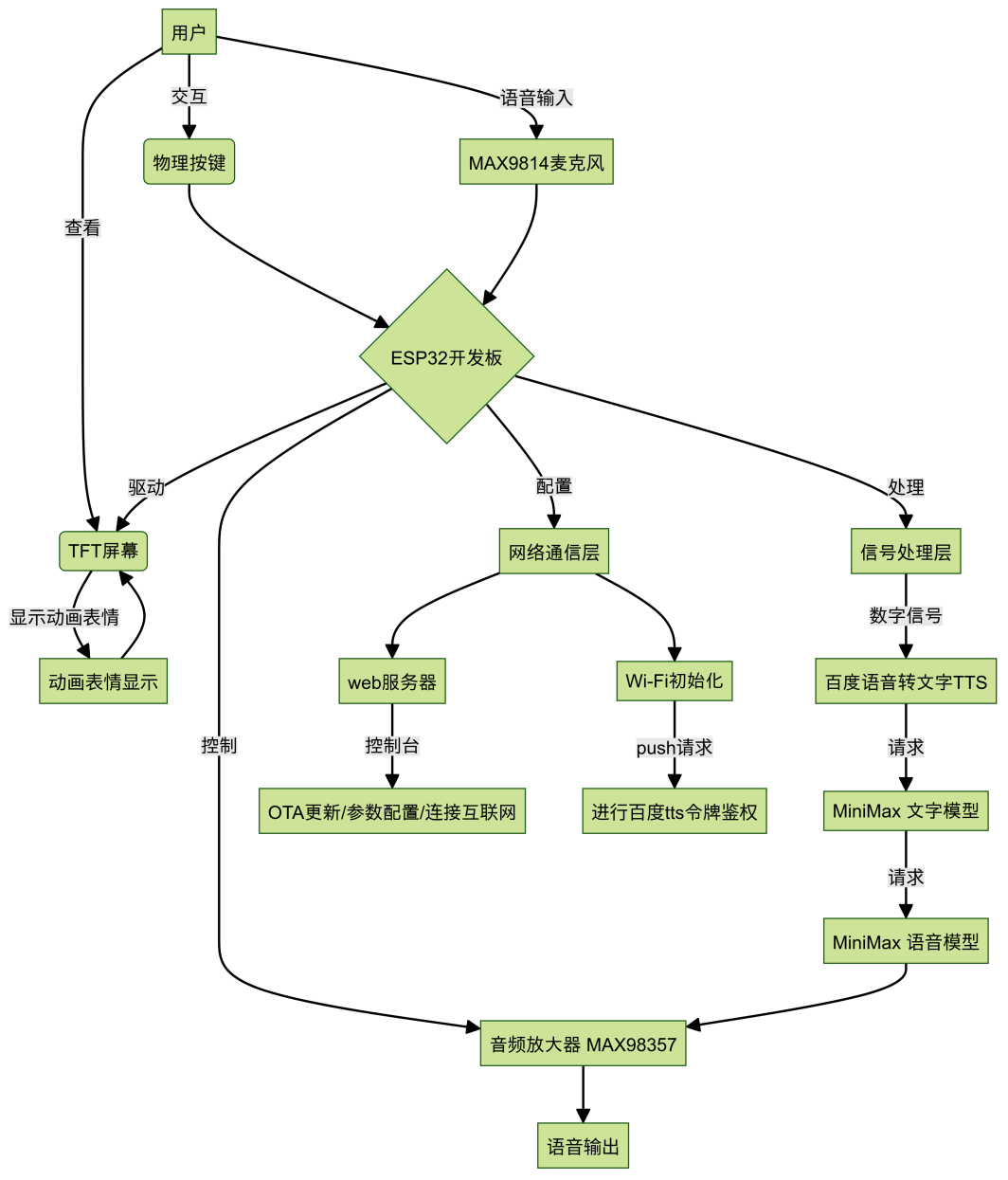


图3.1 ChatChatty 总体设计框架

实物效果

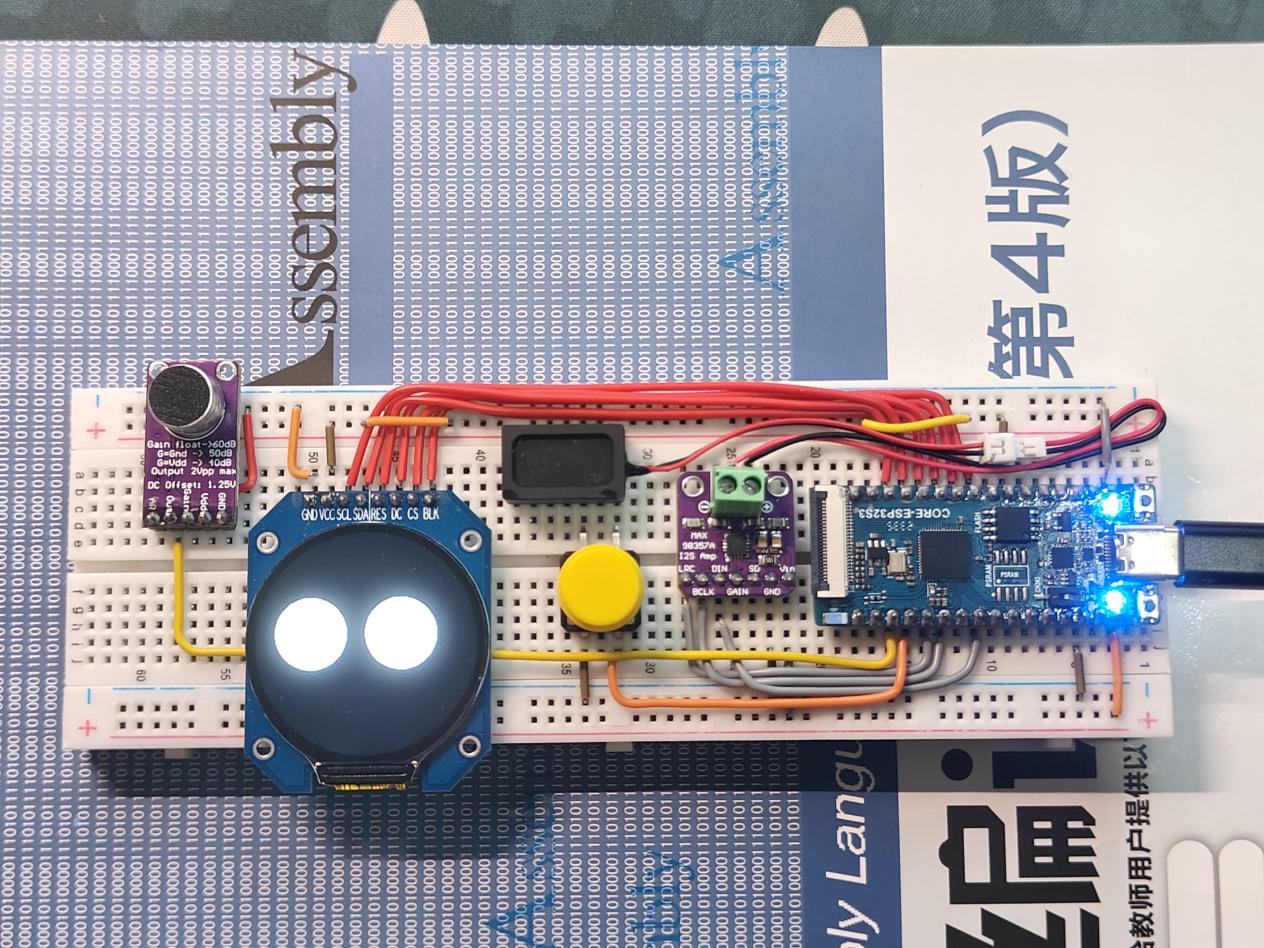


图3.1 ChatChatty 实物效果

### 3.1.2表情显示

ChatChatty AI 聊天机器人的表情显示功能，通过可爱动感的动画表情和经过精心设计的切换逻辑，将机器人的"情感"状态转化为直观的可视语言。我们为它配备了一系列GIF动画，每种动画都承载着不同的情绪，从愤怒到快乐，再到悲伤，这些动画经过提前的加工优化，在保证动画流程的前提下进行抽帧和压缩处理，最后以c数组的格式存储在Flash当中，做到了流畅的播放和内存的有效管理。无论是在待机时的耐心等待，还是在对话中的即时反应，ChatChatty 都能以最合适的表情回应，让每一次的交流都充满生动性和感染力，展现了技术与情感的完美融合。

部分表情状态演示

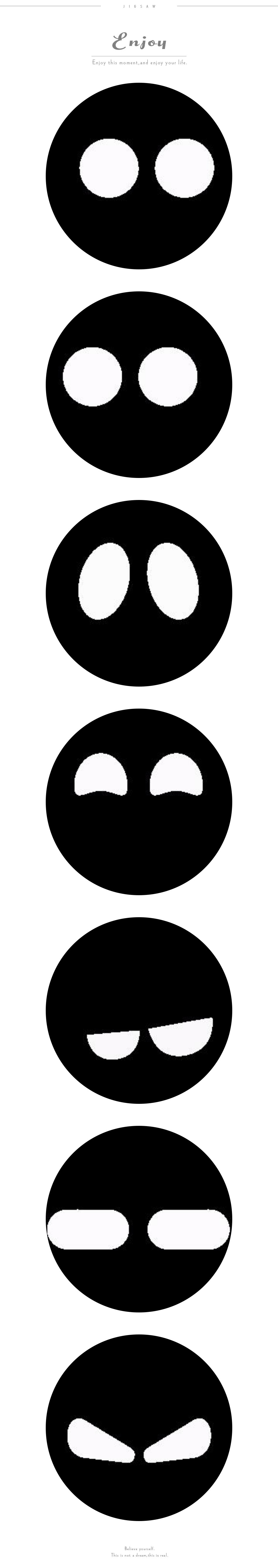
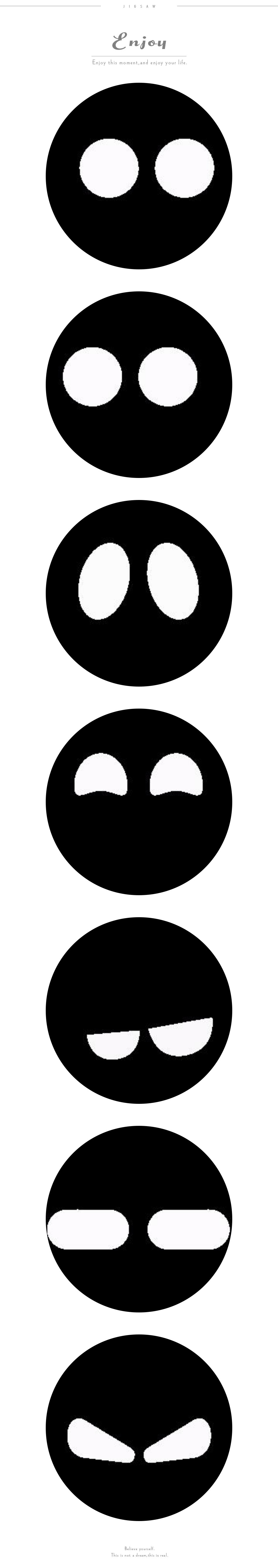
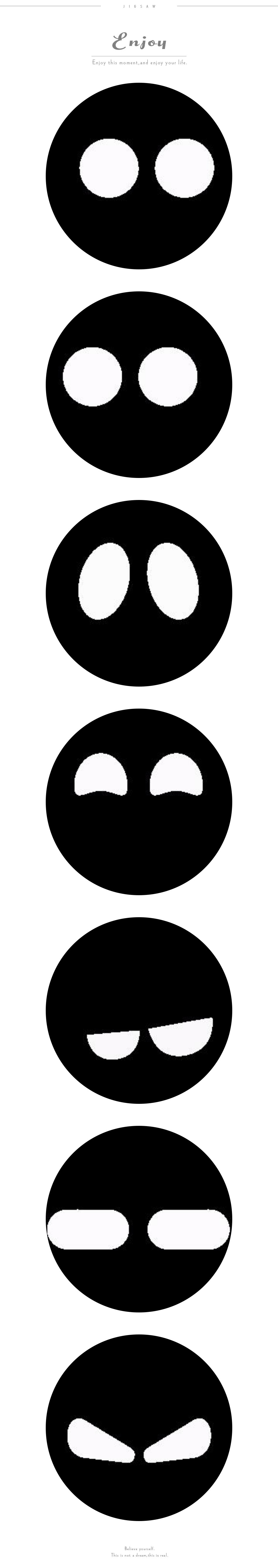


图3.1 表情效果1 图3.1 表情效果2 图3.1表情效果3

### 3.1.3控制交互

控制交互方面，我们精心设计了一个用户友好的网页控制台。这种设计允许用户通过熟悉的网页界面进行操作。主要功能包括，配置页面、OTA更新和音色切换，这三者是构成用户交互体验的核心功能。进入控制界面只需要手机或电脑设备和ChatChatty在同一局域网内，在浏览器输入域名chatchatty.local。或者直接输入ChatChatty的静态IP地址192, 168, 4, 1 ，即可进入控制台的主界面（图3.1.3 控制界面1左侧），如果第一次使用，还没有配置好ChatChatty的wifi信息（此状态下LED\_A和LED\_B会同时闪烁），可以先将手机或电脑设备连接到ChatChatty上的AP热点，热点的名字就叫ChatChatty密码为123456789，然后就可以按照先前的方法进入控制台的主界面了。

**OTA更新**功能允许用户远程升级ChatChatty的固件。用户在网页上选择并上传新的固件文件（图3.1.3 控制界面1右侧），设备接收到文件后会自动进行更新。这一过程无需用户直接连接到设备，简化了升级流程，并确保用户能够及时接收到最新的功能和安全更新。

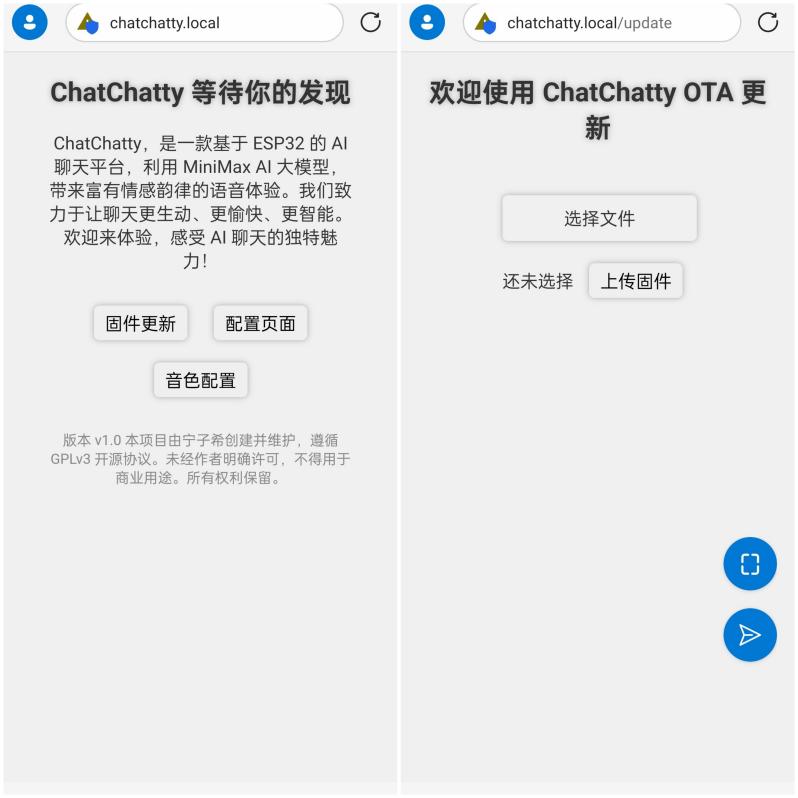


图3.1.3 控制界面1

**配置页面**提供了一个简洁的用户界面（图3.1.3 控制界面2左侧），使用户能够轻松设置ChatChatty的网络和大模型API参数。用户只需在网页表单中输入Wi-Fi的SSID和密码，以及必要的API密钥，然后提交表单。这些信息被永久地存储在ESP32 flash的NVS分区上，设备重启后会自动应用这些设置。

**音色切换**功能则为用户提供了个性化的语音体验。用户可以在网页上选择不同的音色选项（图3.1.3 控制界面2右侧），这些选项对应不同的语音合成模型。选择后，设备会保存用户的偏好，并在随后的交互中使用新音色，让用户享受到更加丰富和个性化的语音反馈。

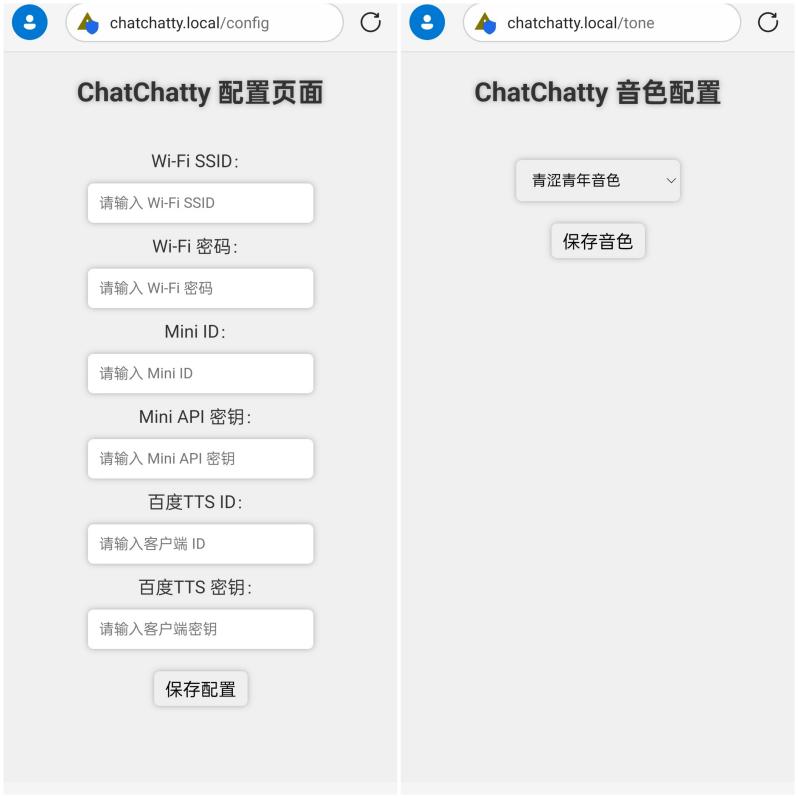


图3.1.3 控制界面2

网页控制台的实现还带来了跨平台的便利性，用户可以在任何带有浏览器的设备上与ChatChatty进行互动，无需下载额外的应用程序。同时这些网页的实现使用了响应式的设计，能够根据用户设备的屏幕尺寸和分辨率动态调整布局。这意味着无论是在手机、平板还是桌面电脑上，用户都能获得优化过后的浏览体验。网页控制台的实现不仅提升了用户体验，也增强了ChatChatty的可访问性和灵活性。

### 3.1.4待机模式

当ChatChatty在一段时间内未检测到用户交互时，将自动转入节能的待机模式。在这种模式下，设备会从NTP（Network Time Protocol）服务器获取并显示当前的准确时间，这不仅保持了设备的实用性，也展示了其智能化特点。通过这种方式，ChatChatty在非交互期间也能为用户提供有价值的信息，同时保持了设备的能源效率。



图3.1.4 ChatChatty待机模式

## **3.2 程序设计**

ChatChatty AI聊天机器人的程序设计是一个全面的过程，它从硬件接口的初始化开始，配置网络连接，集成百度TTS和MiniMax AI服务以实现语音识别和智能对话。通过按键与用户交互，处理录音，将声音转换成文字，再将AI生成的回答转换回语音，最后通过扬声器播放。同时，程序还涉及错误处理、日志记录、代码优化和详尽的测试。用户界面友好，通过TFT屏幕展示动画表情和信息，通过网页表单配置参数，而整个系统的固件支持OTA更新，确保了软件的可维护性和可升级性。

**程序设计分为几个主要步骤：**

* 初始化硬件接口。
* HTML页表注册
* 网络连接和配置。
* 语音识别和合成API的集成。
* 百度TTS鉴权
* OTA空降更新
* 用户界面的实现。
* LVGL界面绘制。

程序的难点包括语音数据处理、百度TTS鉴权和OTA空降更新。

### 3.2.1 百度TTS鉴权

1. ***//----------------------百度TTS本地鉴权---------------------------------***
2. **String getAccessToken( String \_Baidu\_TTS\_ID, String \_Baidu\_TTS\_key) {**
3. ***// 动态生成获取访问令牌的 URL***
4. **String token\_url = "https://aip.baidubce.com/oauth/2.0/token?grant\_type=client\_credentials&client\_id=" + \_Baidu\_TTS\_ID + "&client\_secret=" + \_Baidu\_TTS\_key;**
5. **Serial.println("----------------百度TTS本地鉴权开始------------");**
6. ***// 创建 HTTP 客户端***
7. **HTTPClient http;**
8. **http.begin(token\_url);**
9. **int httpResponseCode = http.POST("");**
10. **String accessToken = "";**
11. **if (httpResponseCode > 0) {**
12. **String response = http.getString();**
13. ***// Serial.println("访问令牌 JSON:");***
14. ***// Serial.println(response);***
15. ***// 解析 JSON 响应以获取访问令牌***
16. **DynamicJsonDocument doc(1024);**
17. **deserializeJson(doc, response);**
18. **accessToken = doc["access\_token"].as<String>();**
19. **} else {**
20. **Serial.print("发送POST错误: ");**
21. **Serial.println(httpResponseCode);**
22. **}**
23. **http.end();**
24. **return accessToken;**
25. **}**

**getAccessToken** 函数负责向百度TTS服务请求访问令牌，这是使用百度文本到语音服务的关键一步。函数通过组合客户端ID和密钥构建OAuth2.0令牌URL，然后发起POST请求。成功响应后，它解析返回的JSON数据以提取access\_token。如果请求失败，则通过串口打印错误代码。最后，函数返回获取的访问令牌供程序其他部分使用，确保了与百度TTS API的安全和有效通信。

### 3.2.2 OTA空降更新

1. ***// -------------------------------OTA方法--------------------------------------***
2. **void performOTAUpdate(){**
3. ***// 设置服务器处理函数***
4. ***// server.on("/", HTTP\_GET, handleRoot); // 根路由重定向到 OTA 页面***
5. **server.on("/update", HTTP\_GET, []() {**
6. **server.sendHeader("Connection", "close");**
7. **server.send(200, "text/html", updateIndex);**
8. **});**
9. **server.on("/update", HTTP\_POST, []() {**
10. **server.sendHeader("Connection", "close");**
11. ***//动态显示结果***
12. **String message = Update.hasError() ? "更新失败" : "更新成功。重新启动…";**
13. **server.sendHeader("Content-Type", "text/html; charset=utf-8");**
14. **server.send(200, "text/html", "<span style='font-size: 36px;'>" + message + "</span>");**
15. **delay(1000);**
16. **ESP.restart();**
17. **}, []() {**
18. **HTTPUpload& upload = server.upload(); *//用于处理上传的文件数据***
19. **if (upload.status == UPLOAD\_FILE\_START) {**
20. **Serial.printf("Update: %s\n", upload.filename.c\_str());**
21. **if (!Update.begin(UPDATE\_SIZE\_UNKNOWN)) { *// 以最大可用大小开始***
22. **Update.printError(Serial);**
23. **}**
24. **} else if (upload.status == UPLOAD\_FILE\_WRITE) {**
25. ***// 将接收到的数据写入Update对象***
26. **if (Update.write(upload.buf, upload.currentSize) != upload.currentSize) {**
27. **Update.printError(Serial);**
28. **}**
29. **} else if (upload.status == UPLOAD\_FILE\_END) {**
30. **if (Update.end(true)) { *// 设置大小为当前大小***
31. **Serial.printf("更新成功 : %u bytes\n", upload.totalSize);**
32. **} else {**
33. **Update.printError(Serial);**
34. **}**
35. **}**
36. **});**
37. **}**

OTA（Over-The-Air）更新是一种无线固件更新技术，允许设备通过无线网络接收并安装新的软件版本。在嵌入式系统和物联网设备中，OTA更新可以远程进行，无需物理接触设备，从而方便地分发安全补丁、新功能和性能改进。在ChatChatty 上，可以通过我们的网页控制台进行OTA更新，用户只需通过浏览器访问我们的URL即可触发更新过程。我们通过设置HTTP服务器的路由来处理更新请求，包括根路由重定向、更新页面的显示、上传文件的处理以及固件的写入和验证。当上传完成后，设备将自动重启以应用更新。如果在更新过程中出现错误，会显示相应的错误信息。

### 3.2.3 MiniMax文字转语音

1. ***//-----------------------------tts语音播报------------------------------------***
2. **String getvAnswer(String ouputText) {**
3. **HTTPClient http2;**
4. **String tts\_url = String(base\_url) + String(Mini\_ID);**
5. **http2.begin(tts\_url);**
6. **http2.addHeader("Content-Type", "application/json");**
7. **http2.addHeader("Authorization", Mini\_Token\_key);**
8. ***// 创建 JSON 对象并填充数据***
9. **StaticJsonDocument<200> doc;**
10. **doc["text"] = ouputText;**
11. **doc["model"] = "speech-01";**
12. **doc["audio\_sample\_rate"] = 32000;**
13. **doc["bitrate"] = 128000;**
14. **doc["voice\_id"] = voice\_id;**
15. ***// 序列化 JSON 对象到字符串***
16. **String jsonString;**
17. **serializeJson(doc, jsonString);**
18. ***// 发送 HTTP POST 请求***
19. **int httpResponseCode = http2.POST(jsonString);**
20. **if (httpResponseCode == 200) {  *// 如果响应成功***
21. **String response = http2.getString();**
22. **http2.end();**
23. ***// 清除之前的 JSON 文档数据***
24. **jsonDoc.clear();**
25. ***// 反序列化 JSON 响应***
26. **DeserializationError error = deserializeJson(jsonDoc, response);**
27. **if (error) {  *// 检查反序列化是否成功***
28. **Serial.printf("deserializeJson() 失败: %s\n", error.c\_str());**
29. **return "error";**
30. **}**
31. ***// 获取音频文件 URL***
32. **const char\* tempURL = jsonDoc["audio\_file"];**
33. **if (tempURL) {**
34. **return String(tempURL);**
35. **} else {**
36. **Serial.println("JSON 响应中没有音频文件 URL");**
37. **return "error";**
38. **}**
39. **} else {  *// 如果响应失败***
40. **Serial.printf("tts %i \n", httpResponseCode);**
41. **String response = http2.getString();  *// 获取响应内容***
42. **http2.end();**
43. **Serial.println(response);  *// 打印错误响应***
44. **return "error";**
45. **}**
46. **}**

使用HTTP客户端向一个文字转语音服务发送请求，将输入的文本转换成语音。函数首先构建请求URL和头部信息，然后创建并序列化一个包含文本和配置参数的JSON对象，通过POST方法发送到服务端。如果请求成功，它将解析返回的JSON响应以获取音频文件的URL，并返回这个URL。如果请求失败或解析出错，函数将返回"error"。这个过程允许我们将文本通过API转换成语音，以供后面的播放。

## **3.3 调试和遇到的问题**

在调试过程中，遇到了音频同步和网络稳定性以及屏幕显示等一些问题。好在通过优化代码和调整硬件连接，最终解决了这些问题。例如在开发过程中，调试**getvAnswer**函数时，就遇到令牌请求失败的现象。刚开始，我怀疑是网络连接导致请求失败的，检查网络连接并进行API的调试之后，排除了这一点。其次，授权问题也可能导致请求被拒绝，果然是因为获取访问令牌的 URL无效。另一个常见问题是JSON格式错误，这是因为序列化或反序列化过程中出现了问题，还有就是对服务端返回的JSON响应的格式理解不对，导致后面的处理有问题。在调试时，我习惯使用串行监视器输出来跟踪请求和响应的状态，这有助于识别问题所在。例如，如果HTTP响应码不是200，我们通过打印出的错误信息和响应内容来了解具体的错误原因。如果JSON解析失败，我们通过检查**DeserializationError**来获取更多细节。

同时我还遇到了一些屏幕显示问题，包括内存分配失败、GIF动画加载问题、待机模式下时间显示不准确，以及性能问题导致的刷新缓慢。为了解决这些问题，我首先审查了初始化代码，确保LVGL显示缓冲区的内存分配充足。我还检查了GIF资源的加载，保证动画能够平滑播放。针对待机模式下的时间显示问题，我优化了时间更新逻辑，确保显示的时间准确。为了提高性能，同时优化了表情切换那块代码，使用函数指针代替原来回调的机制，减少了屏幕刷新的延迟。通过这些步骤，屏幕显示的稳定性和响应速度得到了提升，显示效果也有所改善。

# 4. 改进和心得

## **4.1 未来改进**

在开发ChatChatty AI聊天机器人项目的过程中，我深刻认识到了项目当前状态与理想目标之间的差距，并从中发现了一些关键的改进领域。首先，屏幕刷新的流畅性在高负载或特定操作下并不理想，这可能会影响用户的视觉体验，尤其是在动画播放和快速屏幕切换时。为了解决这一问题，我计划深入分析当前的屏幕刷新机制，并探索更高效的图形处理和缓冲策略，以确保在各种条件下都能提供稳定流畅的显示效果。同时目前项目中的表情动画种类和自定义程度尚不能完全满足用户的个性化需求。为了丰富用户的情感体验，我计划扩展表情动画库，加入更多情感表达的动画，并提供更多的自定义选项，使用户能够根据个人喜好和场景需求进行选择，从而增强用户的互动体验。

通过这些综合性的改进，我相信ChatChatty项目将能够更好地满足用户的期望，并在AI聊天机器人领域中提供更加丰富、精准和个性化的交互体验。这些改进不仅将提升项目的技术成熟度，也将加强其市场竞争力，为用户带来更加完善的产品。

## **4.2 心得体会**

通过着手ChatChatty AI聊天机器人项目的整个开发过程，我获得了宝贵的经验和深刻的认识。这个项目不仅是一个技术挑战，也是一个深入了解用户需求和提升产品可用性的机会。在技术层面，我学习了如何将AI模型和语音识别的API调用集成到单片机上，这涉及到了跨学科的知识和技能，包括编程、电子工程、用户界面设计等。

在项目的早期阶段，我花费了大量时间研究和选择合适的硬件平台和软件框架，以确保项目的可行性和扩展性。随着项目的推进，我逐渐意识到了良好设计的重要性，这不仅关乎产品的外观，更关乎用户与产品交互的直观性和便捷性。在开发过程中，我不断迭代设计，测试不同的用户界面布局和功能实现，以期达到最佳的效果。我也认识到了项目管理和时间管理的重要性。在项目的不同阶段，我学会了如何设置合理的里程碑，优先处理关键任务，并灵活应对突发的问题和变更。ChatChatty AI聊天机器人项目是一个充满挑战和学习的经历，它不仅提升了我的技术能力，也锻炼了我的创新思维和问题解决能力。我期待着将这些经验应用到未来的工作中，继续在人工智能领域探索和成长。