|  |
| --- |
| **2025芯原杯**  **全国嵌入式软件开发大赛**  **设计文档** |
| **芯原微电子（上海）股份有限公司** |
| **团队名称: 自动化摸鱼队**  **联系邮箱: shu\_xy714@qq.com**  **联系电话: 17761458707**  **日 期: 2025.07.24** |

**文档修订历史**

本节记录文档版本修订历史。

注:本文档不需要对每个补丁或次要修订进行更新。 本文档中的内容在一系列的修订中趋于稳定。

|  |  |  |  |
| --- | --- | --- | --- |
| **版本** | **日期** | **作者** | **说明** |
|  |  |  |  |
|  |  |  |  |

目录

[文档修订历史 3](#_Toc203984149)

[目录 4](#_Toc203984150)

[1. 概述 5](#_Toc203984151)

[2. 方案介绍 6](#_Toc203984152)

[3. 算法设计 7](#_Toc203984153)

[4. 软件工作流程 8](#_Toc203984154)

[5. 验证及测试结果 9](#_Toc203984155)

[6. 团队介绍及分工 10](#_Toc203984156)

# 概述

本项目在RISC-V嵌入式开发板上实现了人声检测和说话人识别功能，涵盖PDM读取、音频播放、模型运行、蓝牙收发等功能模块，所有功能测试正常~~，无明显BUG~~；

算法部分，使用MFCC进行特征提取；设计、训练并量化部署了int8类型的CNN，分别用于人声检测、说话人识别，人声检测模型的测试集准确率为xxx，说话人识别模型的测试集准确率为xxx。所有模型都经过了量化、裁剪设计，能够在开发板上实时运行，正确识别。

嵌入式部分，使用环形缓冲区实现安全且高效的语音PCM数据内存管理；使用SDK的Event System API实现多任务管理与任务间消息传输；使用NMSIS-DSP库实现并加速MFCC计算，使用NMSIS-NN库提供的API实现在开发板上进行卷积和池化计算，最终在开发板上实现了实时人声检测和说话人识别，算法更新频率为1秒。系统所有功能均按题目要求实现，且进行了10分钟以上的稳定运行测试。

蓝牙部分，定义了蓝牙GATT Service；使用App写入Control和Rwa\_Data的特征值实现开关；创建ble\_task\_app注册监听事件并循环监听；创建ble\_task\_event\_handler实现算法处理结果打印；创建bt\_speech\_set\_raw\_data\_value 去Notify原始音频数据；基于原始数据包格式文件创建包头实现对原始数据封装和CRC-8数据校验。最终在APP上实现了BLE数据传输的完整性、验证了开始/结束音频播放及正确性、实现了算法处理与原始数据发送同时工作。此外，使用iOS手机App持续测试了30分钟原始数据发送，实际采样率始终稳定在8000 SPS附近，准确率一直为100%，验证了蓝牙的长时数据传输的稳定性。在使用新版本的App后，对原始数据开关测试了20次，均可以稳定完整播放数据，蓝牙的循环稳定性非常好。

# 方案介绍

# 算法设计

# 软件工作流程

系统上电并完成初始化后，分别创建多个task，各task各自完成自己负责的功能。

1. 创建audio\_task用来初始化i2s和codec，随后注册Event，等待其它Task发送请求播放音频Event；为避免在audio\_task的回调函数中直接操作i2s和codec，在回调函数中仅设置标志位，实际播放函数在audio\_task的主循环中进行；
2. 创建algo\_task，注册两个Event用来分别监听来自speech\_task的PCM数据就绪消息和来自ble\_task的算法运行开关消息；随后初始化mfcc并malloc相关缓冲区（这些缓冲区会一直使用，故仅在algo\_task中分配一次，后续不再释放），随后在主循环中监听Event；
3. 创建speech\_task用来初始化PDM并注册PDM中断回调函数，在回调函数中将PDM中断数据存入环形缓冲区中，并判断缓冲区中的数据累积是否超过1秒，若超过1秒则向algo\_task发送数据就绪Event；
4. 创建ble\_task，同样注册两个Event用来监听来自algo\_task的打印识别结果消息和发送原始数据消息；编写BLE相关回调函数并与定义的BLE GATT Service绑定；由于原始数据发送功能较为耗时，为避免直接在ble的task回调函数中进行原始数据发送，同样只在回调函数中设置标志位，实际的原始数据发送在ble\_task的主循环中进行；

**高效安全的语音数据内存管理：**

为了实现安全且高效的PCM数据内存管理，使用了环形缓冲区RingBuffer来存储语音数据，RingBuffer由静态分配的全局数组和一个读索引、一个写索引组成，长度为32001byte，可以存储2秒的PCM数据。当RingBuffer数据存满时，会自动覆盖最前面的数据，而不会造成数据溢出；

**系统工作流程：**

系统上电后创建以上task并完成初始化，此时PDM开始工作，系统不断在PDM的中断回调函数中将PCM数据保存到RingBuffer，并在RingBuffer数据存储量大于2000byte（0.125秒）时向algo\_task发数据就绪消息，通知algo\_task处理数据；

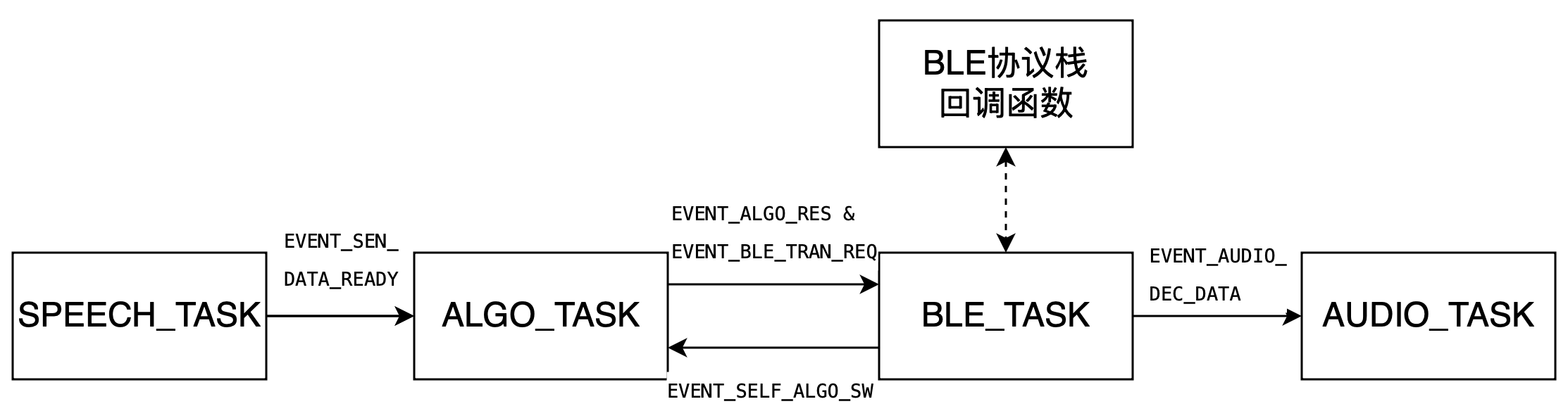
algo\_task收到数据就绪消息后，将RingBuffer中的数据取出并存放在另一个静态数组中，随后判断是否启动了原始数据发送功能和算法识别功能，若启动了原始数据发送功能则发送Event给ble\_task，由ble\_task完成原始数据发送功能；若启动了算法识别功能则分别运行mfcc、人声检测、其他说话人识别、目标说话人识别模型，并将识别结果通过Event发送给ble进行蓝牙发送；

当手机APP上启动原始音频发送功能后，开发板的GATT Service注册的回调函数触发，使能系统的原始数据发送标志位，并在ble\_task的主循环中完成原始音频数据发送；

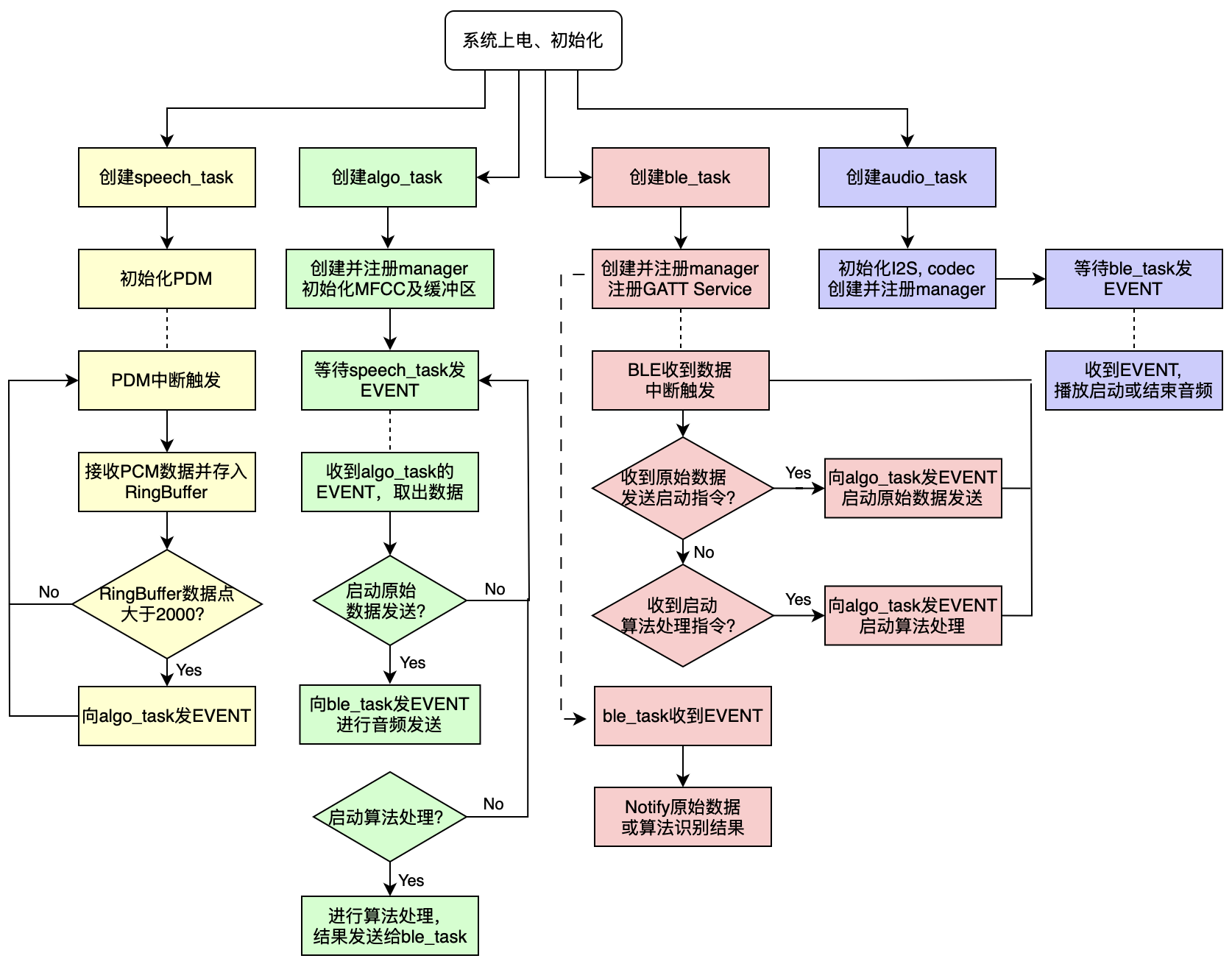
当手机APP上启动算法识别功能后，同样开发板的GATT Service注册的回调函数出发，在回调函数中发送启动算法处理Event，由algo\_task的回调函数接收该Event并启动算法处理；

算法处理主要包括4大部分，第一部分调用NMSIS-DSP库的API实现float格式的MFCC计算；第二部分调用NMSIS-NN库提供的卷积、池化等函数进行二分类VAD人声检测，若检测结果为非人声，则向ble\_task发送打印非人声识别结果消息并退出算法处理函数，若检测结果为人声，则进行第三部分计算；第三部分同样使用NMSIS-NN库进行二分类其他说话人识别，若识别结果为其他人，则向ble\_task发送打印其他人识别结果消息并退出算法处理函数，若识别结果为目标说话人，则进行第四部分计算；第四部分运行向量嵌入模型，并调用NMSIS-DSP库实现结果向量与目标向量的相似度计算，并将相似度最大的结果作为最终识别结果向ble\_task发送消息。

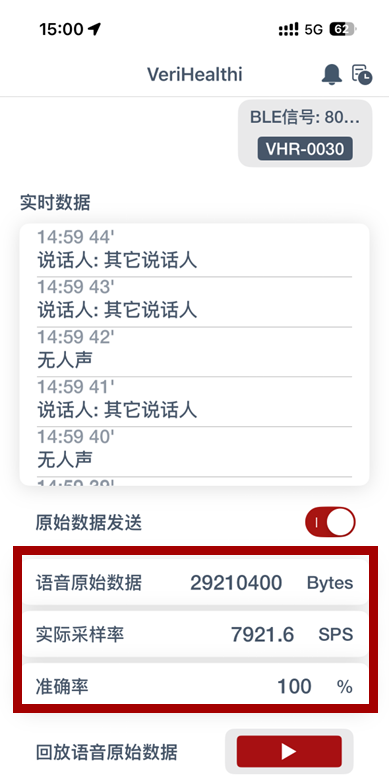
我们绘制了系统创建的所有task及它们之间相互通信的Event流向，如下图所示



我们绘制了系统的整体工作流程图，如下图所示：



# 验证及测试结果 蓝牙部分：验证及测试结果如下，左一及左二为iOS手机VESCApp按下原始数据发送开关后，持续测试了30分钟，实际采样率始终稳定在8000 SPS附近，跳动范围小，准确率一直为100%，验证了蓝牙长时原始数据传输的稳定性。开始/结束音频播放及正确性验证无误，对接收到的原始数据进行了播放，验证了BLE数据传输的完整性。左三图为语音原始数据与算法处理同时开启的截图显示，可以看到功能运行正常。



# 团队介绍及分工

我们是自动化摸鱼队，来自西部赛区-电子科技大学自动化工程学院，年级都是研二，都是同一间教研室的同门，我们的团队分工情况如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 姓名 | 单位 | 年级 | 分工 |
| 舒星研（队长） | 电子科技大学自动化工程学院 | 研二 | 嵌入式软件开发、模型部署 |
| 李东昊 | 电子科技大学自动化工程学院 | 研二 | 算法设计、模型训练、量化和部署 |
| 张含 | 电子科技大学自动化工程学院 | 研二 | BLE蓝牙协议开发 |