瑞芯微平台安卓音频驱动开发

音频是安卓平台的重要组成部分，在我司，音频对讲也是业务的重要组成单元，为了能够使安卓系统识别音频器件，并且能正常运行，音频驱动是中间的桥梁。主要工作包括从kernel的驱动层开始，逐级调试，直至应用层能够正常采集与播放音频数据，且音频数据正常。

# 音频相关的一些概念

首先，需要熟知音频的几个基本属性。采样率–采样频率，每秒钟采集的声音样本数，我司常用的有8k、16k、32k、44.1k、48k。位宽–每个声音样本的位数，常用的有8bit、16bit、24bit。我司音频相关算法均为16bit。声道数–记录声音时，如果每次生成一个声波数据，称为单声道，每次生成两个声波数据，称为双声道（立体声）。其次，需要指导PCM格式的音频数据格式，PCM(Pulse Code Modulation)需要进行采样和量化。将模拟的音频信号转成数字信号，并可以使用cooledit软件分析和展示所采集到音频数据，如图1.1所示。

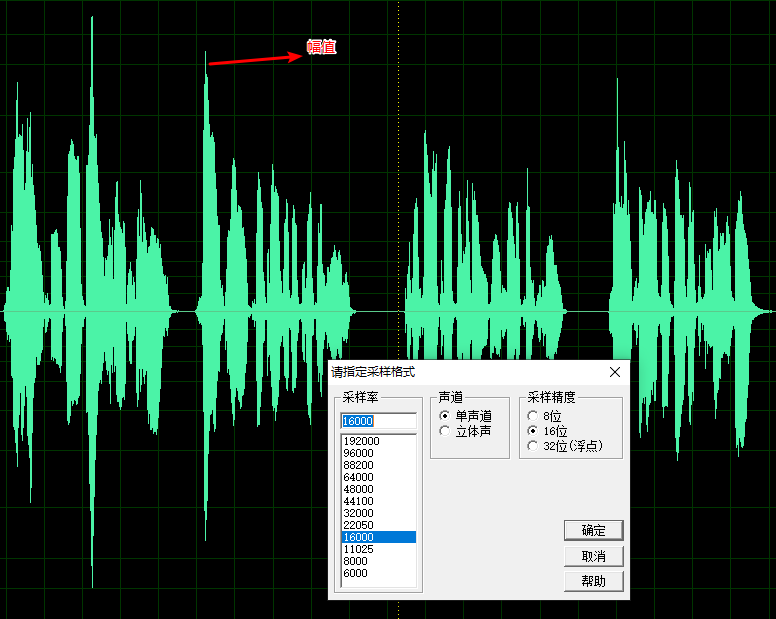
 

图1.1 声音的数字存储形式

接着，需了解音频驱动的一些相关概念。Codec-音频编解码器，负责处理音频信息，包括ADC、DAC、Mixer、DSP、输入输出以及音量控制等功能。Codec与处理器之间通过I2C总线和数字音频接口DAI进行通信，其中I2C总线主要实现对Codec寄存器数据的读写，DAI(digital audio interface)实现音频数据在CPU与Codec间的通信。包括I2S、PCM等。DAI\_LINK:绑定CPU\_DAI和Codec为一个声卡，等同于Machine Driver，本文主要使用simple-card来创建声卡。图1.2为音频数据通过I2S传输的具体形式。

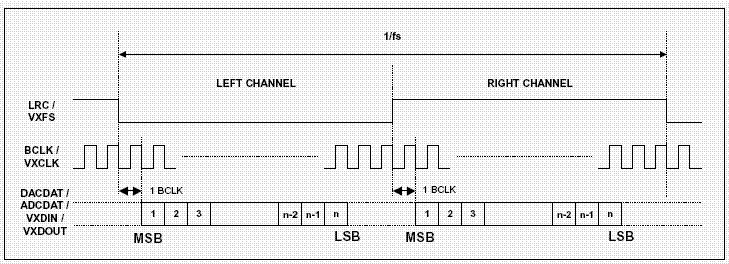


图1.1 I2S音频数据传输

其次，音频数据调试主要方法是采用tinyalsa命令，tinyalsa主要沿用了Linux内核内置的音频子系统ALSA，tinyalsa主要封装了内核的ALSA的接口，简化ALSA的用户空间。Tinyalsa的源码在external/tinyalsa,编译tinyalsa后会生成tinymix、tinyplay、tinycap、tinypcminfo4个小工具，tinymix打印出声卡信息，tinyplay只能播放wav格式的双通道音频，tinycap录音并能保存音频数据，tinypcminfo用于查看pcm通道相关信息。

最后，将一些基本概念以表格形式呈现，如表1所示。

表1 音频驱动一些基本概念

|  |  |
| --- | --- |
| 名称 | 释义 |
| 采样率 | 每秒钟采集的声音样本数 |
| 位宽 | 每个声音样本的位数 |
| Codec | 音频编解码器 |
| I2C | 实现对codec寄存器数据的读写 |
| 12S | 传输音频数据，涉及SCLK、MCK、LRCK、SData |
| DAI | 实现音频数据在CPU与Codec间的通信 |
| 设备树 | 配置设备信息的一个树形结构 |
| 驱动 | 一般由厂商提供的器件驱动程序 |
| Tinyalsa | 便于音频调试的工具 |

# 音频驱动开发

安卓音频驱动主要包含以下3部分，注册声卡，tinyalsa调试，通过应用app测试。如图2.1所示。

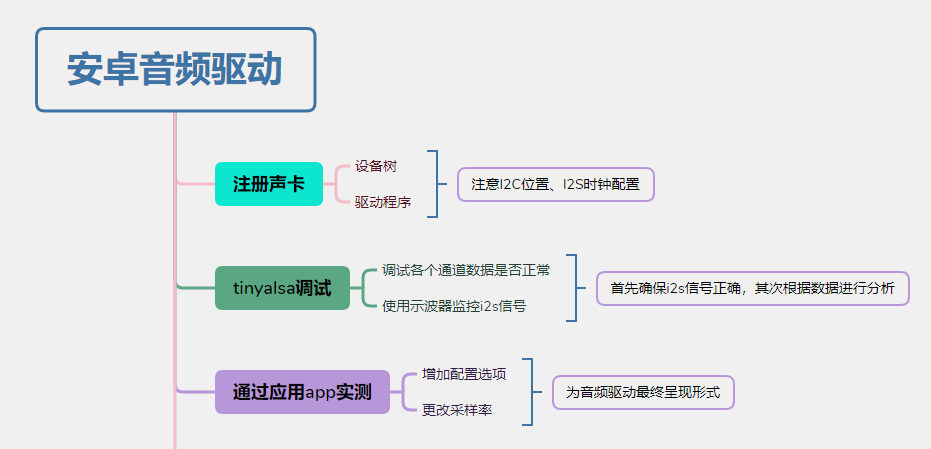


图2.1 安卓音频驱动开发流程图

## 注册声卡

安卓音频驱动首先需注册声卡，可使用cat /proc/asound/cards 和ls /dev/snd等命令查看，其中注册声卡主要包括配置设备树和编译驱动程序。主要涉及到设备树以及kernel方面的修改。

音频驱动移植依据板子上所集成的器件决定，本例音频采集芯片以es7210为例，音频播放以es7149为例。

es7210可以支持4路通道同时输入，有两个SDOUT1/2 数据线可以输出，当ES7210 工作在级联模式的时候可以SDOUT2/TDMIN 作为输入。而需要同时采集4路通道的音频数据需采用2mic+2回采的方案。即可以把4通道的16bit 的数据转成32bit的左右声道输出，此方案造成还需要在hardware层和framework层等为了上述定制而做出一些定制化的修改，es7210芯片的工作流程如图2.2.所示。

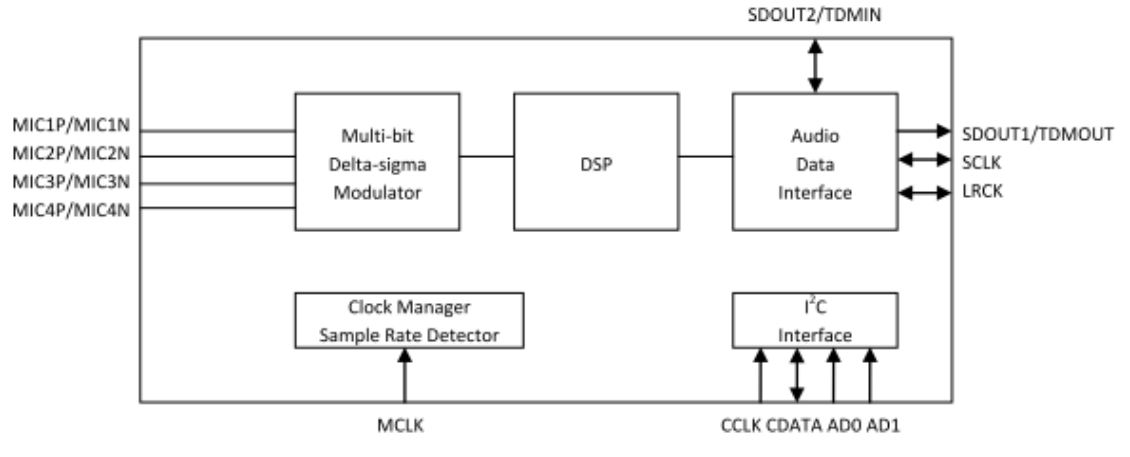


图2.2 es7210芯片工作流程图

### 驱动开发

以RK3568安卓管理机为例，es7210作为安卓管理机的音频采集芯片，负责mic1、mic2、话柄、回采等采集工作，如图2.3所示。

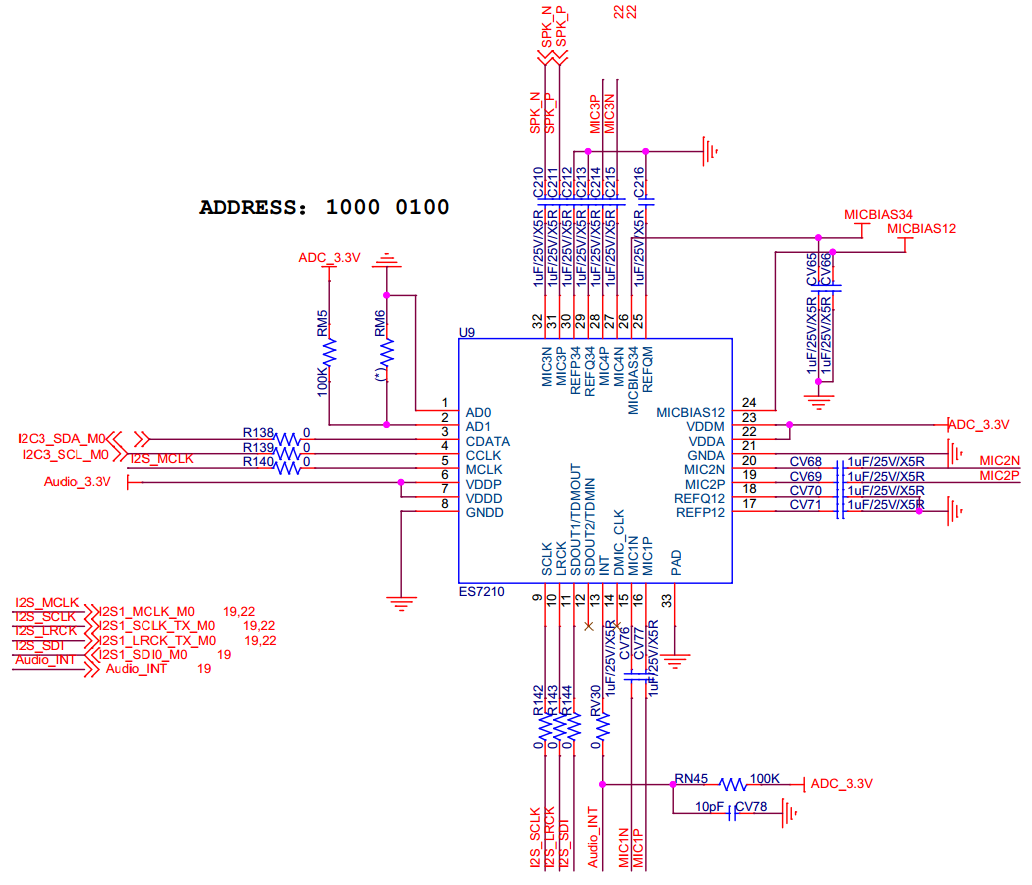


图2.3 安卓管理机音频录音芯片电路图

首先需要在设备树以及codec驱动中增加声卡的配置，安卓管理机codec由es7210采集芯片的codec和es7149播放芯片的codec构成，machine driver采用simple-card framework，CPU\_DAI使用I2S,如图2.4所示。

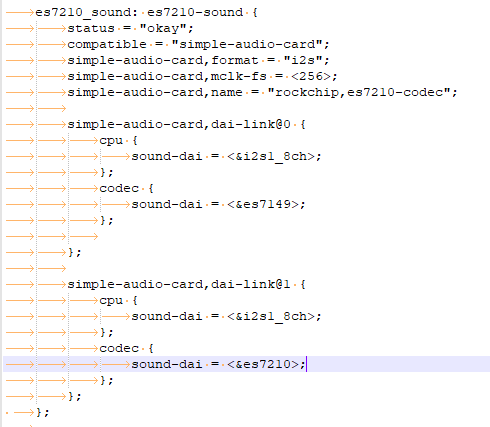


图2.4 安卓管理机声卡配置

es7210 Codec需要首先配置es7210驱动程序,驱动程序主要是由厂家提供，将驱动程序放置对应位置，本例对应的路径为SDK/kernel/sound/soc/codecs/，配置Kconfig与makefile，完成编译，配置如图2.5所示。

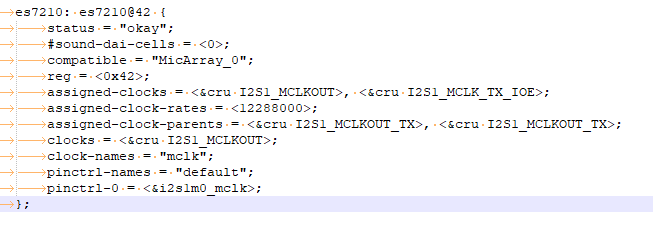


图2.5 es7210 codec 芯片配置

完成上述工作后，可使用命令cat /proc/asound/cards可获取已注册的声卡，如图2.6所示。安卓管理机有两个声卡，分别为rockchip\_hdmi，rochchip\_es7210。

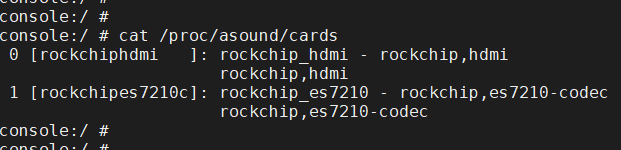


图2.6 安卓声卡

可通过命令ls /dev/snd获取声卡信息，其中controlC0、controlC1为声卡控制设备，pcmC1D1c为录音设备，pcmC0D0p、 pcmC1D0p为播放设备，timer为时序，C0D0c，表示Card 0 Device 0，即编号0的声卡下的编号0设备，c代表capture，表示录音设备；p代表playback，表示播放设备。如图2.7所示。

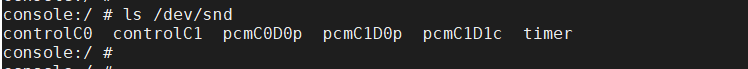


图2.7 声卡信息

### Tinyalsa调试

Tinyalsa调试，可通过tinycap命令和tinyplay命令调通各个通道音频数据是否正确，tinycap可采集音频各个通道数据到本地，结合原理图配置，调通各个通道数据，tinyplay播放一个双通道对应采样率的音频文件，判断左右声道是否有声音以及声音是否失真。主要涉及kernel以及hal层方面的修改，如果上述出现无法采集声音数据，或声音数据异常，可通过示波器检测I2S信号进一步分析判断。

可通过tinycap命令验证所采集到的数据，录音命令：tinycap t.wav –r 16000 –c 2 –b 32。

文件t.wav 包含4通道16k 16bit数据，依次按顺序channel1，channel3，channel2，channel4。对照原理图与器件，验证每个通道是否都有数据，且音频数据播放是否正常。

为此，需要分别在hal层以及codec驱动修改对应的配置。在hal层中，设置采集的pcm\_config的采样率为16k,双通道，格式为32bit,左对齐。在程序中，在获取录音相关信息中，配置为32bit,如图2.8所示。

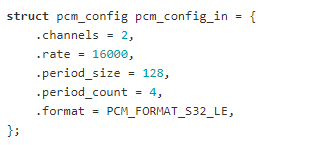


图2.8 hal层配置音频输入格式

为了兼容音频输入格式以32bit形式呈现，还需在hal层中的audio\_hw.c中做相应的修改，主要是更改bufferSize和音频格式，如图2.9所示。

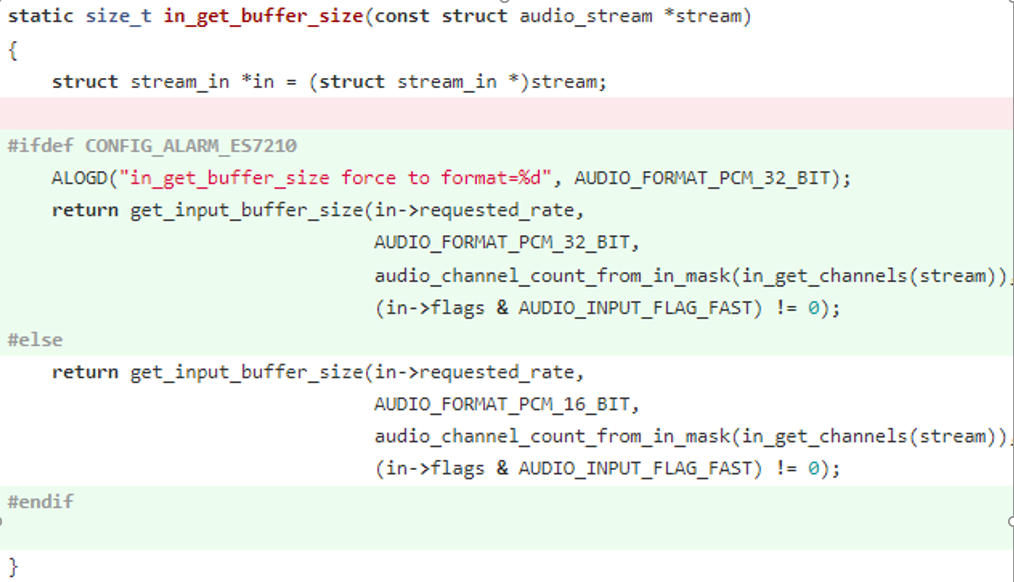


图2.9 配置hal层录音格式

在驱动中，需要配置各个通道采集的数据为16bit，各个通道才能以16bit采集，最后组成32bit双通道传输，如图2.10所示。

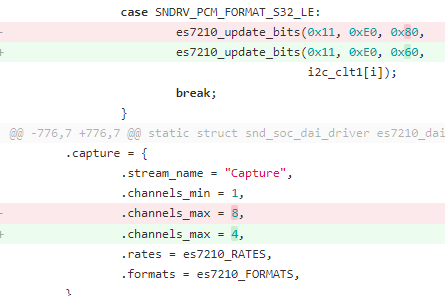


图2.10 驱动配置4通道16bit

### 应用程序测试

应用app调试，可根据Java应用层提供或c++framework层提供的AudioRecord类和AudioTrack类实现音频采集和播放，对音频模块进一步适配，主要涉及到hal层和framework层的修改。

因es7210的2mic+2回采的方案，导致hal层中原本的重采样算法失效，可将固定采样率的音频采集到后经framework层进行重采样，可在AudioRecord类中加载研究院的重采样算法库，并在read函数中获取各个通道的音频数据，分别重采样，再进行整合，最终上传到应用层。从而既能实现各个采样率的音频采集又能判断各个通道数据是否正常。

## Es7149播放芯片

安卓管理机的音频播放芯片采用的是es7149,es7149支持两路DAC数据播放，分别对应左右声道，如图2.11所示。

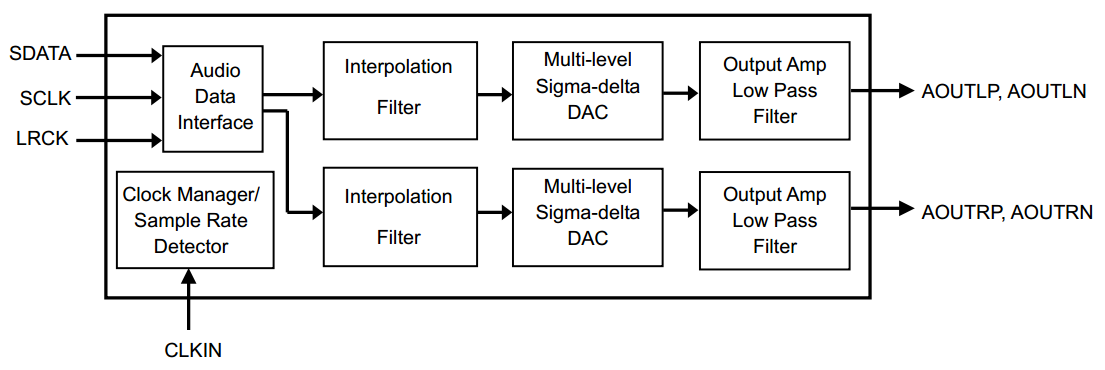


图2.11 es7149原理图

在RK3568项目中，左声道对应lineout口，与三段式耳机相连，右声道对应于speaker,如图2.12所示。

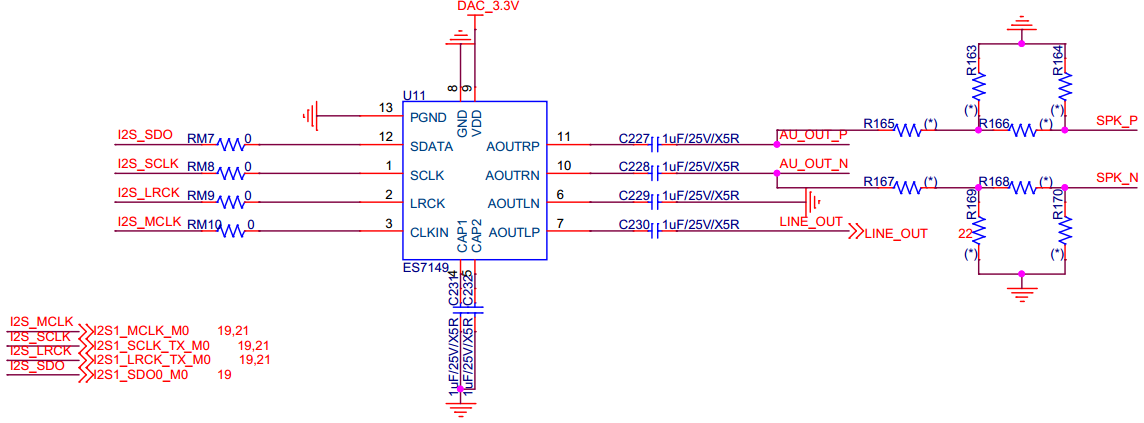


图2.12 RK3568中es7149芯片原理图

采用simple-card framework注册到声卡中，将驱动程序放置对应位置，本例对应的路径为SDK/kernel/sound/soc/codecs/，配置Kconfig与makefile，完成编译，codec配置如图2.13所示。

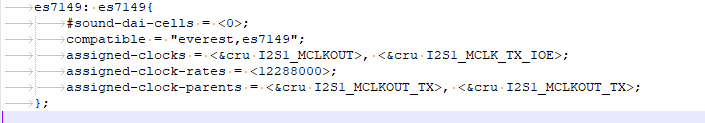


图2.13 es7149 codec 芯片配置

在hal层配置audio的输出格式，采样率为16k,输出格式为16bit左对齐，如图2.14所示。

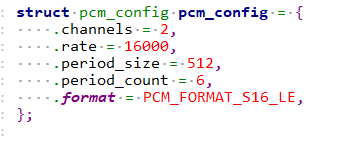


图2.14 es7149 codec输出配置

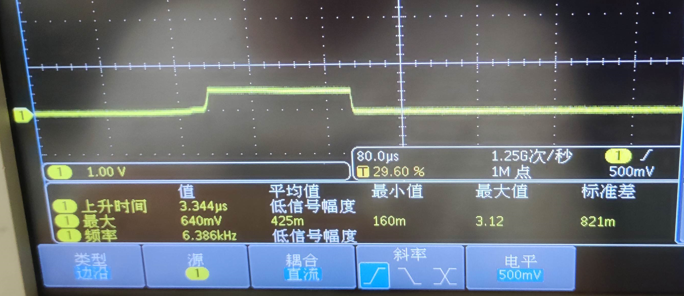
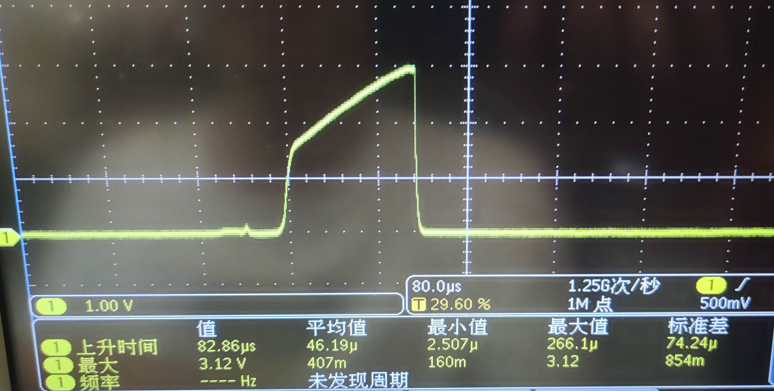
最后可以由tinyplay进行测试，播放命令：tinyplay t.wav。其中t.wav是以wav格式的音频播放文件，播放文件的采样率为16k,双通道。

## 问题分析以及注意事项

安卓音频驱动主要调试重点为I2S信号的正确性以及音频各个通道数据的正确性，其中，I2S信号主要可以通过示波器采集来显示定位问题。音频各个通道数据的正确性主要通过tinyalsa命令以及在hal层获取对应数据来分析各个通道音频的数据。

### 配置安卓管理机音频时钟

RK3568平台需要指定父时钟和时钟频率，父时钟和时钟频率可由assigned-clocks、assigned-clock-rates、assigned-clock-rates指定。如未指定，会导致lrck和mclk失效，从而导致所采集到的声音数据异常，如图2.15所示。

（a）lrck (b)mclk

图2.15 错误的I2S信号以及mclk信号

### ES7210采集到的两通道数据一致

ES7210所采集的数据以双通道32bit形式呈现，但发现各个通道32bit中的两通道数据是一致的，经过调试发现在Hal层中由SPEEX\_DENOISE\_ENABLE宏，音频数据会进行了抑制噪声处理，而抑制噪声处理主要是取两个相邻声音数据取平均，从而导致两通道数据一致，所以只要注释掉噪声处理宏即可，噪声处理过程如图2.16所示。

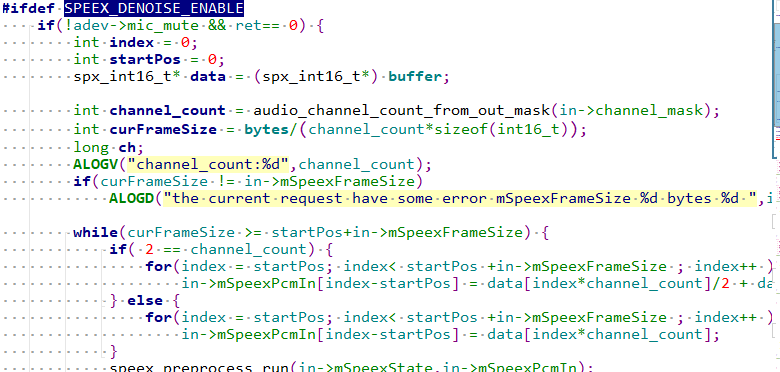


图2.16 抑制噪声处理

### Hal层调试方法

在调试过程中，可在hal层中打开ALSA\_IN\_DEBUG这个宏定义来获取录音数据，开机过程中在/data/目录中创建debug.pcm文件，并赋普通用户的读写权限，在调用tinyplay指令时，在debug.pcm文件可保存录音数据，以进一步分析。

### Framework调用无权限问题

本项目中DSP主要是在framework层中开展工作，调用到音频、视频接口时会遇到因无权限导致中途退出，导致无法获取音视频数据问题。目前采用的方法是修改framework层的相关代码，在无权限处打印无权限信息，但不退出。

# 音频外围开发

音频外围开发主要包括es7210芯片录音增益接口，各个音频采集器件、播放器件切换、音频接入时检测、t1测试程序等。

## ES7210录音增益调节接口

对于一个音频定制化的安卓产品，安卓系统需适配音频采集与播放的各个音频器件，为了达到一个合适的音频效果，需要调节音频采集和播放的增益，音频采集增益一般由音频codec驱动提供，需要提供接口供DSP调用。

Es7210的增益主要包括ADC增益以及ALC增益，有对应的寄存器地址以及调制增益方式，增益调节主要通过I2C完成。应用层与内核层的通信主要通过内核层注册一个miscdevice的节点，通过ioctl指令完成，如图3.1所示。

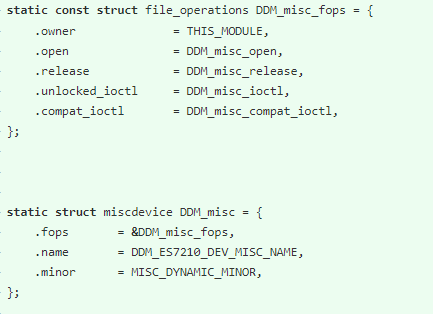


图3.1 通过ioctl完成增益调节

在注册es7210声卡时，注册/dev/es7210这个节点，实现file\_operations中各个函数，最终操作es7210\_read、es7210\_write接口实现对寄存器的操作。其中es7210\_read、es7210\_write函数均由驱动提供，主要的功能是读取指定寄存器的值和将值写入指定的寄存器中。所以应用程序中，只要传入指定寄存器地址，即可完成指定寄存器值的读取，传入指定寄存器地址以及对应的值，即可将值写入到指定寄存器中。

## 音频器件的切换

安卓管理机的录音器件主要包含mic1、mic2、三段式耳机输入、话柄输入、咪杆等。ES7210音频采集芯片只支持4路同时采集，所以需要通过切换开关实现各个器件之间的配合工作，如表2所示。

表2 音频采集器件配置表

|  |  |  |
| --- | --- | --- |
| Es7210采集通道 | 音频采集器件 | GPIO切换 |
| MIC1 | MIC1 | 无 |
| MIC2 | MIC2 | GPIO1A4 |
| 三段式耳机输入 |
| MIC3 | 回采 | 无 |
| MIC4 | 话柄输入 | GPIO1B0 |
| 咪杆 |

安卓管理机的播放器件主要包含喇叭输出、话柄输出、三段式耳机输出等，ES7149音频播放芯片只支持2路同时播放，所以也需要通过切换开关实现各个器件之间的配合工作，如表3所示。

表3 音频采集器件配置表

|  |  |  |
| --- | --- | --- |
| Es7149播放通道 | 音频播放器件 | GPIO切换 |
| 左声道 | 三段式耳机 | 无 |
| 右声道 | 喇叭输出 | GPIO1B2 |
| 话柄输出 |

音频器件的切换主要通过控制gpio口的高低电平来实现，通常会在kernel中与其余gpio控制一起新增一个alarm\_misc驱动，应用通过传入gpio寄存器通道号以及高低电平值以达到上述效果。

## 音频接入检测

音频的接入检测主要分为3种，为查询gpio口高低电平的变化、查询逐次逼近型的模数转换器、安卓系统提供的以广播形式检测3段式耳机的拔插检测。

以循环检测话柄接入为例，在原理图中，通过检测SARADC\_VIN4这个逐次逼近型的管脚的电压值来检测话柄是否接入，如图3.2所示。

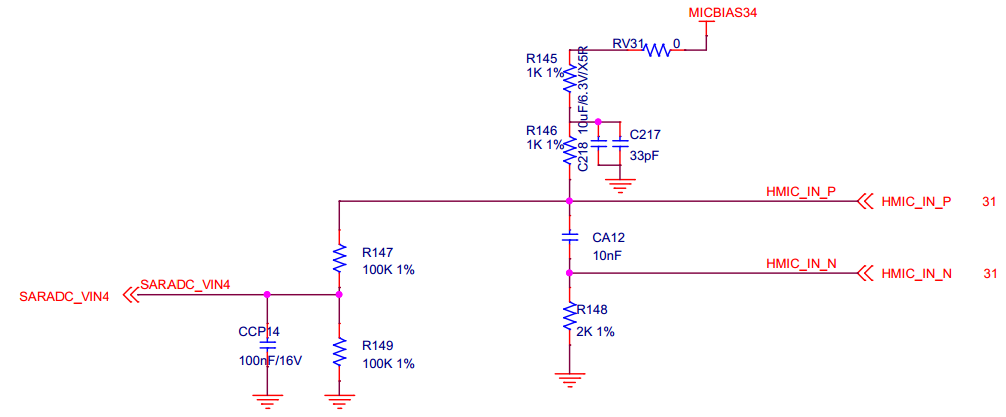


图3.2 话柄插入检测原理图

项目中，主要以input事件形式提供给应用层调用，所以需要在内核中增加input事件驱动。首先在设备树中配置相关信息，如图3.3所示。

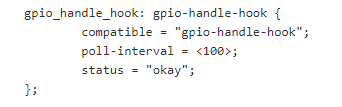


图3.3 话柄插入检测设备树配置

在驱动程序中，使用input\_polled\_dev查询话柄插入的状态，查询时间间隔为100ms,最终查询结果以input事件形式上报，如图3.4所示。

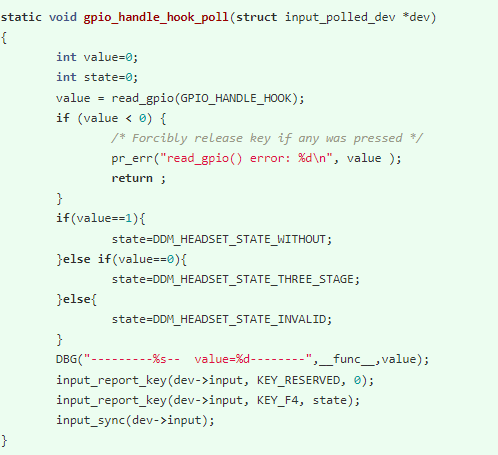


图3.4 input形式上报话柄插入检测

3段式耳机的检测主要通过Android内部提供的广播机制来实现，主要是构造一个类，这个类继承BroadcastReceiver，IntentFilter过滤android.intent.action.HEADSET\_PLUG事件，将这个广播注册成一个UI线程，当这个过滤事件发生会调用onReceive方法，如图3.5所示。



图3.5 3段式耳机拔插检测

## t1测试程序

t1测试程序主要测试器件的功能是否出现异常，针对音频器件，录音采用的是通过tinycap指令获取录音数据文件，再读取录音数据文件，将各个通道进行分离，得到各个器件的录音数据，播放是将一段双通道的音频数据文件通过tinyplay进行播放。为了提高产线的生产效率，采用ald算法用于检测，ald主要通过比较两端音频的相似度来实现，即在播放指定文件的同时采集播放的声音，将两段声音提取相同的部分放入算法，算法返回值为0或者1表示成功，表示播放和采集的器件所采集到的音频为同一段音频，即能表示音频器件采集和播放都正常。

在10寸床头机项目中，播放ald音频文件，同时将接入的话柄放置在喇叭边上，同时采集话柄的声音数据和mic的声音数据，所得的数据与播放的音频数据进行比较，结果如图3.5所示。

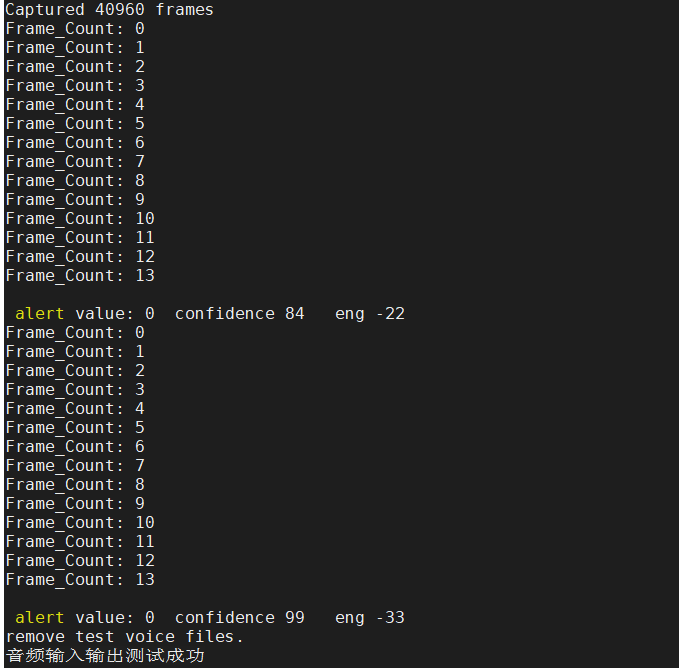


图3.5 音频t1测试程序结果图