AX 音频调试指南

文档版本：**V1.7**

发布日期：**2024/07/23**

目 录

权利声明

注意

您购买的产品、服务或特性等应受商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非商业合同另有约定，本公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

前 言

适用产品

AX620E 系列产品（AX630C、AX620Q）

适读人群

- 软件开发工程师
- 技术支持工程师

符号与格式定义

符号/格式	说明
`xxx`	表示您可以执行的命令行。
斜体	表示变量。如，“*安装目录*/AX620E_SDK_Vx.x.x/build 目录”中的“安装目录”是一个变量，由您的实际环境决定。
☞ 说明/备注：	表示您在使用产品的过程中，我们向您说明的事项。
！ 注意：	表示您在使用产品的过程中，需要您特别注意的事项。

修订历史

文档版本	发布时间	修订说明
V1.0	2023/9/26	文档初版
V1.1	2023/11/17	增加音量增益调节策略
V1.2	2023/11/24	增加播放 pop 音处理方法
V1.3	2024/01/31	修改播放 pop 音处理方法
V1.4	2024/04/10	增加 ALC 使用注意项
V1.5	2024/05/11	增加外置 ES8156 配置方法，补充改善 pop 音的方法
V1.6	2024/06/28	增加 DMA 内存修改配置
V1.7	2024/07/23	补充 FAQ 内容

1 AUDACITY 的常用操作

本章节主要介绍 AUDACITY 的常用操作

1.1 软件简介以及下载链接

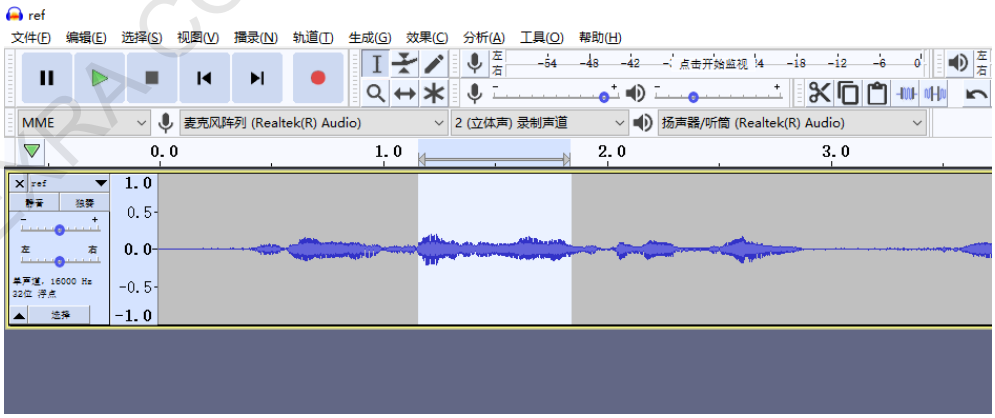
Audacity 是一个跨平台的声音编辑软件，用于录音和编辑音频，是自由、开放源代码的软件。可在 Mac OS X、Microsoft Windows、GNU/Linux 和其它操作系统上运作。

下载链接：https://www.audacityteam.org/

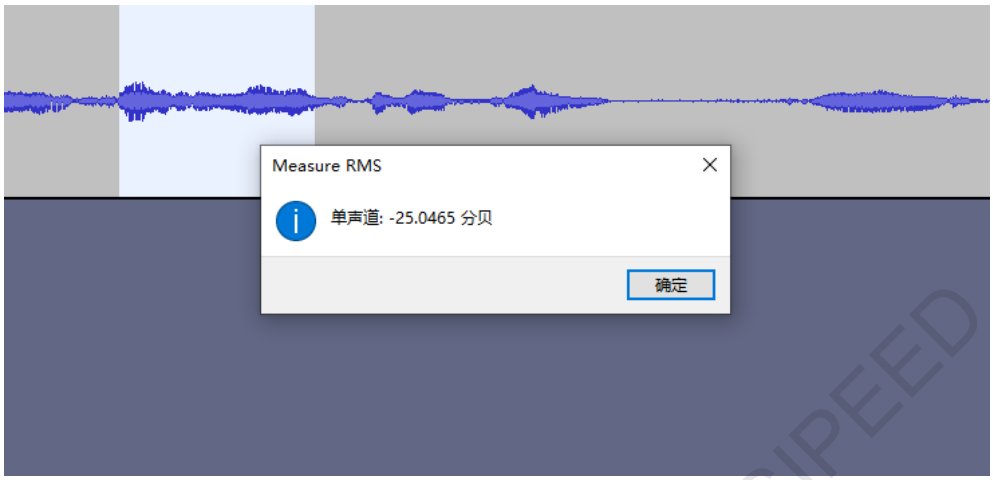
1.2 查看增益

在对讲的调试过程中经常需要查看一段时间内的信号大小。下面以图为例，说明如何通过 Audacity 来查看信号的大小。

1) 选取需要分析的时间段



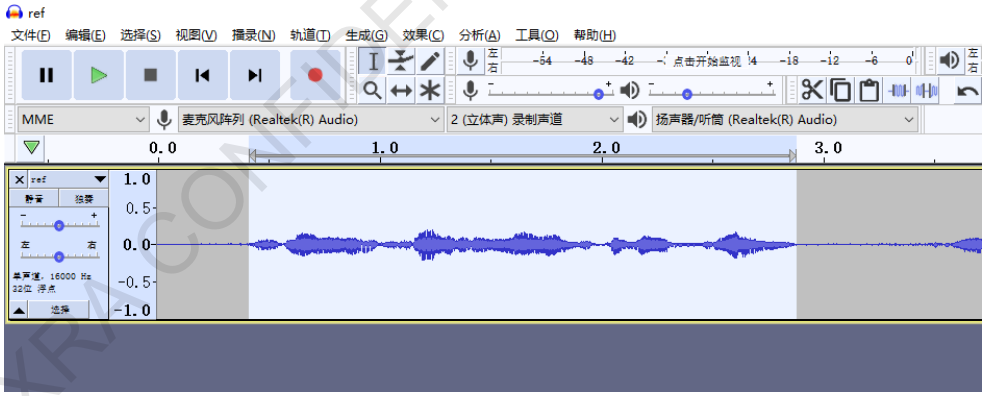
2) 打开“分析”窗口，点击“Measure RMS”，查看选区的平均振幅



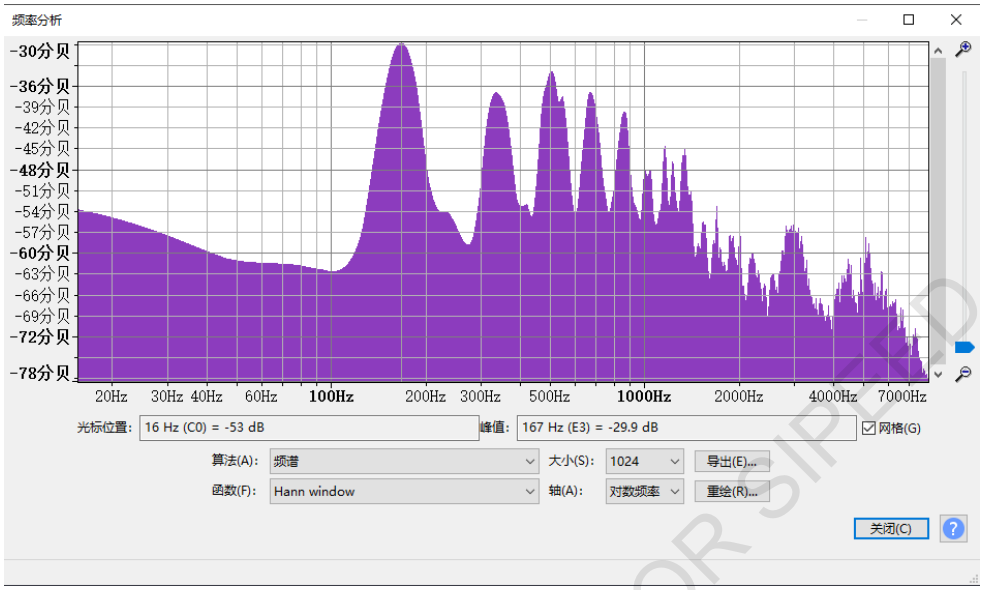
1.3 查看频响特性

在对讲的调试过程中，经常需要对设备的频响特性做调整，下面将以图例讲述如何通过 Audacity 来查看音频的频响特性。

1) 选取需要分析的数据选区



2) 打开“分析”窗口，点击“频谱分析”



2 AUDIO PROCESS 内部流程介绍

本章节主要介绍回声消除环境搭建

2.1 外置 codec

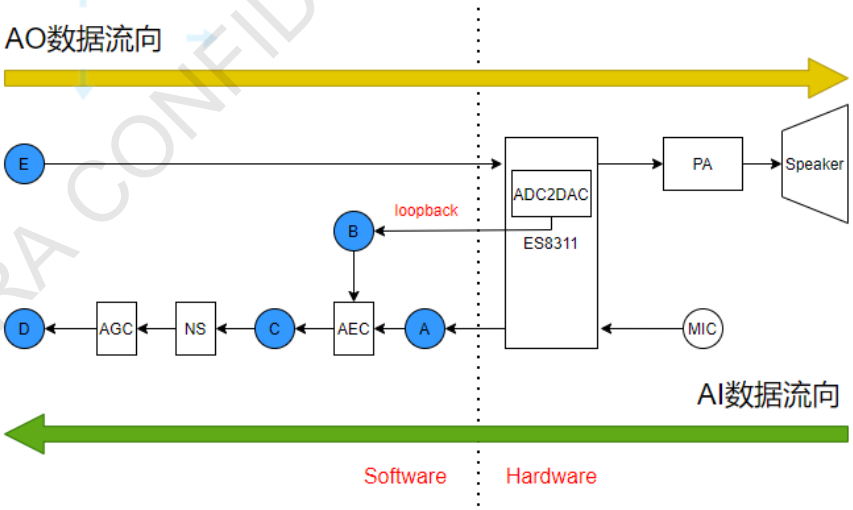
本章节以 ES8311 为例。

2.1.1 ES8311 内部流程

1) AUDIO PROCESS 内部框图

下图为 ES8311 的基本数据流向：

（以下描述为不包含声音检测、重采样、编解码等算法的基本情况）



2) AI 数据流向分析

上图中绿色箭头方向为 AI 数据的流向，数据由麦克风采集，经过芯片内部的增益，到达 AUDIO PROCESS 层（A 点），此处（A 点）的数据为原始的 PCM 数据，接着和 AO 端的数据（B 点）一起送入 AEC 算法进行处理，AEC 算法处理后的数据（C 点）会继续送

到后面的 Audio Process 算法（NS、AGC）进行处理，处理后的数据即为最终的数据（D 点）。

3) AO 数据流向分析

上图中黄色箭头方向为 AO 数据的流向，待播放 PCM 数据（E 点）由应用通过 tinyalsa 接口送入后，经过芯片内部的增益、外部电路的功放芯片，再由喇叭输出。

2.1.2 回声消除算法测试

➢ 环境准备

硬件环境：

- 一块 AX620E DEMO 板

  确认开发板 ES8311 ADC 通路（ES8311-MIC）和 DAC 通路（ES8311-PA-Speaker）连接正常。

- 一台 PC

软件环境：

- 开发板烧写指定 SDK
- PC 上安装 Audacity，下载链接见 1.1 章节
- 测试音频，下载链接：https://echocatzh.github.io/Demo-of-DeepComplexAEC/samples/simu_test/reference/clean/50_lpb.wav

➢ 测试步骤

1) 开发板上电，确认 ES8311 驱动成功加载

使用 dmesg 命令判断 ES8311 驱动是否成功加载

2) 将测试音频放到/soc/scripts/ref_dir

cd /soc/scripts/

mkdir ref_dir

将测试音频放进来，重命名为 audio.wav

3) 设置参考声音通路，如下命令设置左声道为参考声道

/usr/local/bin/tinymix set 33 1

/usr/local/bin/tinymix set 32 4

可用/usr/local/bin/tinymix contents 查看设置是否成功

4) 测试硬件信号和回声通路

cd /soc/scripts/

sh aec_test.sh

对麦克风说话，5 秒后 CTRL+C 停止程序

生成 123.wav，导入电脑用 Audacity 打开，看是否能录制参考声音（信号 1）和 MIC 声音（信号 2）

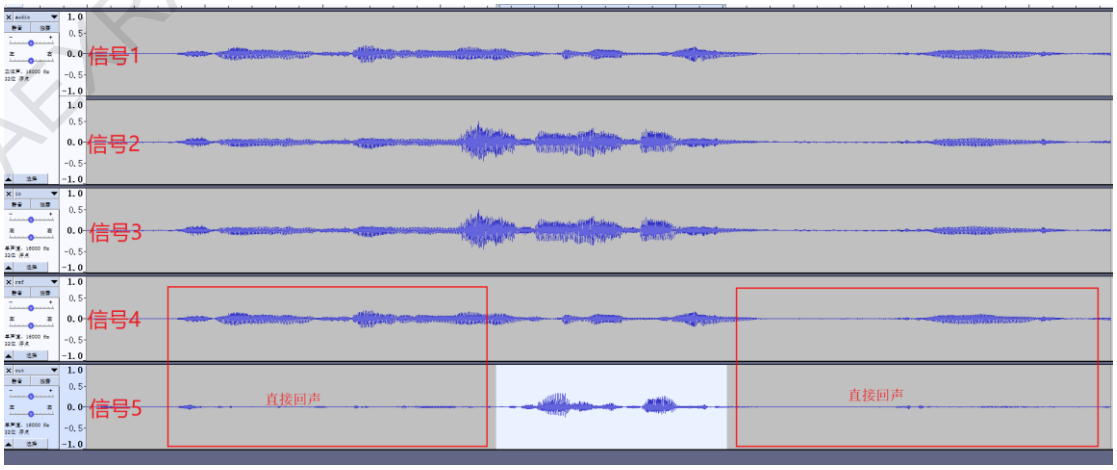
5) 测试算法效果

cd /soc/scripts/

sh aec_test_algo.sh

对麦克风说话，5 秒后 CTRL+C 停止程序

生成 audio.wav、in.wav、ref.waf、out.wav，查看 out.wav 的效果（信号 5）



2.1.3 异常说明以及处理方法

- 问题描述：ref 声音消顶导致的 out 声音异常

  解决方案：尝试降低 audio out 的音量来解决

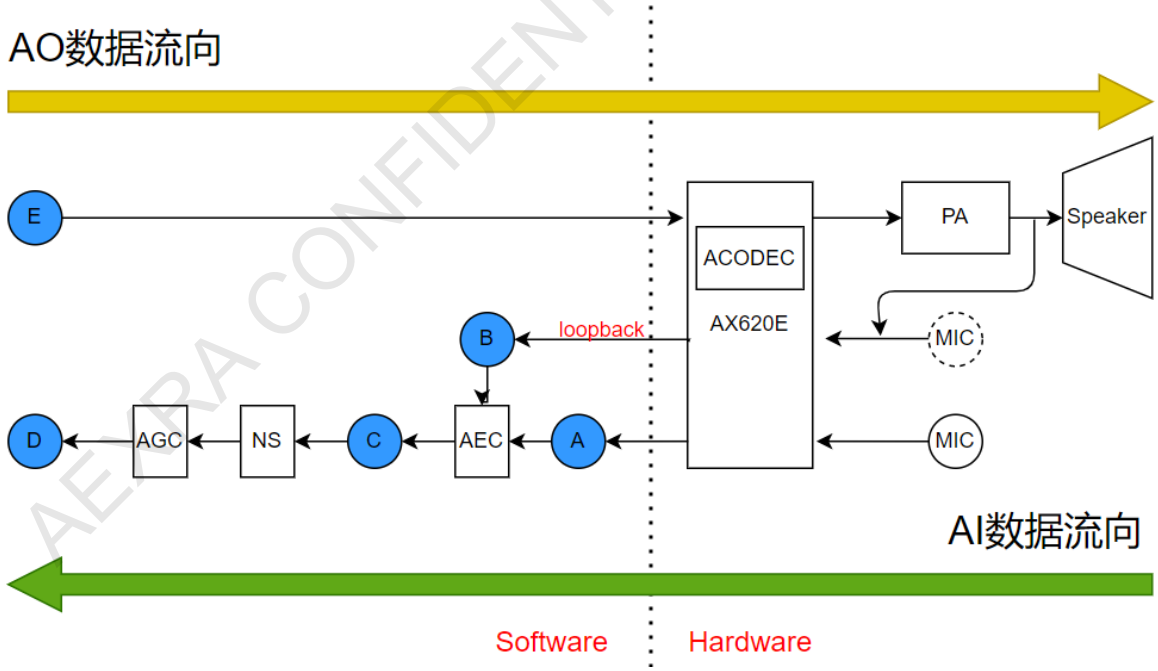
  /usr/local/bin/tinymix set 25 <xxx>

2.2 内置 codec

2.2.1 内置 codec 内部流程

内置 codec 的回声消除在硬件上有两种方式：内部回环和外部回环。

外回环流程具体流程图如下：



- AI 数据流向分析

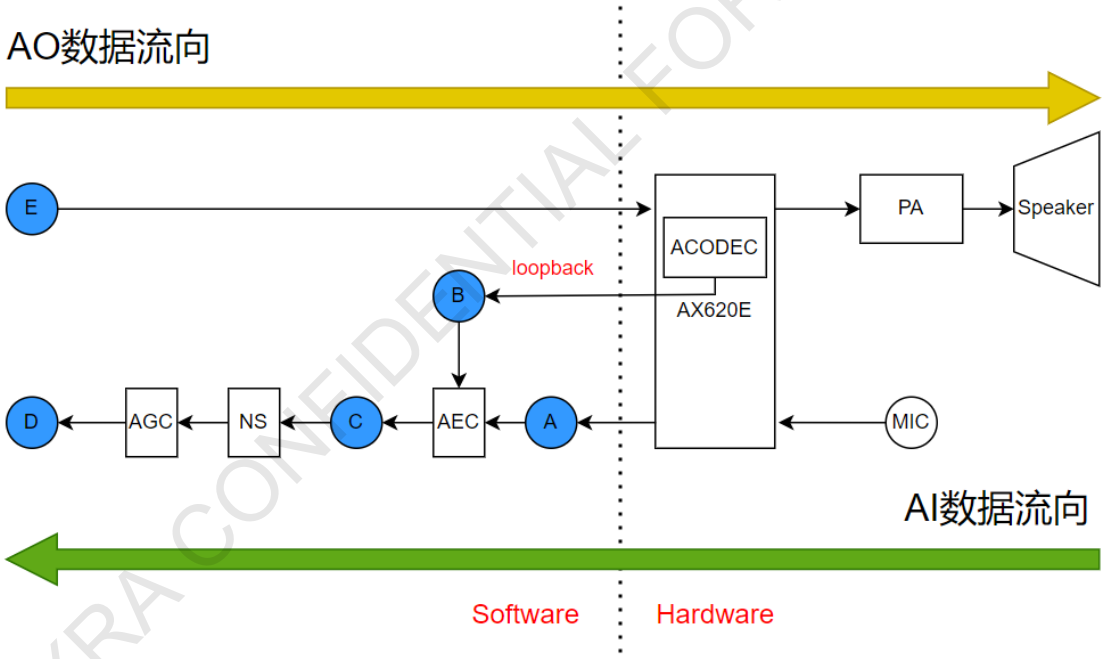
  上图中绿色箭头方向为 AI 数据的流向，一路数据由一路麦克风采集，经过芯片内部的增益，到达 AUDIO PROCESS 层（A 点），此处（A 点）的数据为原始的 PCM 数据，另外

一路 MIC 处的数据是直接由播放端回环过来的数据，同样经过芯片内部增益，到达 B 点，然后 A 点和 B 点的数据一起送入 AEC 算法进行处理，AEC 算法处理后的数据（C 点）会继续送到后面的 Audio Process 算法（NS、AGC）进行处理，处理后的数据即为最终的数据（D 点）。

- AO 数据流向分析

上图中黄色箭头方向为 AO 数据的流向，待播放 PCM 数据（E 点）由应用通过 tinyalsa 接口送入后，经过芯片内部的增益、外部电路的功放芯片，再由喇叭输出。

内回环流程具体流程图如下：



4) AI 数据流向分析

上图中绿色箭头方向为 AI 数据的流向，数据由麦克风采集，经过芯片内部的增益，到达 AUDIO PROCESS 层（A 点），此处（A 点）的数据为原始的 PCM 数据，接着和 AO 端的数据（B 点）一起送入 AEC 算法进行处理，AEC 算法处理后的数据（C 点）会继续送到后面的 Audio Process 算法（NS、AGC）进行处理，处理后的数据即为最终的数据（D 点）。

5) AO 数据流向分析

上图中黄色箭头方向为 AO 数据的流向，待播放 PCM 数据（E 点）由应用通过 tinyalsa 接口送入后，经过芯片内部的增益、外部电路的功放芯片，再由喇叭输出。

2.2.2 内回环回声消除算法测试

➢ 环境准备

硬件环境：

- 一块 AX620E DEMO 板

  使用内回环的时候确认双 MIC 和 SOC 连接正常。

- 一台 PC

软件环境：

- 开发板烧写指定 SDK
- PC 上安装 Audacity，下载链接见 1.1
- 测试音频，下载链接：https://echocatzh.github.io/Demo-of-DeepComplexAEC/samples/simu_test/reference/clean/50_lpb.wav
- 在对应的 dts 配置文件中将 i2s-m-aec-sclk-sel 配置成

```
&i2s_inner_mst0 {
    status = "okay";
    /*i2s - pad0*/
    i2s-m-aec-cycle-sel = <0>;    /*23-22*/
    i2s-m-aec-sclk-sel = <1>;     /*21*/
    i2s-inner-codec-en = <1>;     /*20*/
    i2s-exter-codec-en = <0>;     /*19*/
```

```
&i2s_inner_slv0 {
    status = "okay";
    /*i2s - pad0*/
    i2s-m-aec-cycle-sel = <0>;    /*23-22*/
    i2s-m-aec-sclk-sel = <1>;     /*21*/
    i2s-inner-codec-en = <1>;     /*20*/
```

然后在驱动文件 sound/soc/axera/ax-actt.c 中增加#define INTERNAL_REF 宏定义。

- 然后确认输入方式，则将 layout 设置成 AX_AI_INTERNAL_MIC_NULL 或者 AX_AI_INTERNAL_NULL_MIC 模式。注意：这里一定要软硬件对应。

➢ 测试步骤

1. 开发板上电，确认驱动成功加载

2. 将测试音频放到测试命令执行的目录下，并命名为 audio_test.wav

3. 测试硬件信号和回声通路

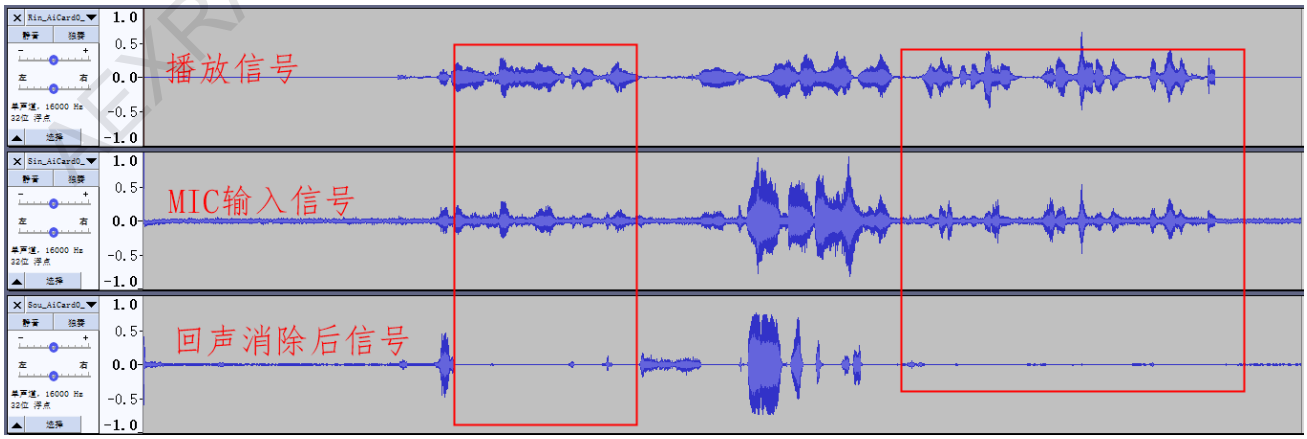
- 执行 sample_audio ai -D 0 -d 0 -r 48000 -p 480 --aec-mode 2 --routing-mode 0 --layout 3 --save-file 1 & sample_audio ao -D 0 -d 1 --resample 1 --resrate 16000 -r 48000 -p 480 -i audio_test.wav
- 调节 AI 输入音量 tinymix -D 0 set 0 0; tinymix -D 0 set 3 45（这一步根据实际情况，尽量保证播放信号和 MIC 接收后的信号幅值相差不大即可）。
- 对麦克风说话，5 秒后 CTRL+C 停止程序
- 生成生成 Sin_AiCard0.wav、Rin_AiCard0.waf、Sou_AiCard0.wav 三个文件，导入电脑用 Audacity 打开，查看 Sin_AiCard0.wav(MIC 输入信号)、Rin_AiCard0.waf(播放信号)是否对应。

6) 测试算法效果

执行 sample_audio ai -D 0 -d 0 -r 48000 -p 480 --aec-mode 2 --routing-mode 0 --layout 3 --save-file 1 --ns 1 -w 1 & sample_audio ao -D 0 -d 1 --resample 1 --resrate 16000 -r 48000 -p 480 -i audio_test.wav

5 秒后 CTRL+C 停止程序

生成 audio.wav、Sin_AiCard0.wav(MIC 输入信号)、Rin_AiCard0.waf(播放信号)、Sou_AiCard0.wav(回声消除后的信号)。



2.2.3 外回环回声消除算法测试

➢ 环境准备

硬件环境：

- 一块 AX620E DEMO 板

  使用外回环的时候确认一路 MIC 和 SOC 连接正常，另外一路 MIC 与对应输出回环起来，且确认此 MIC 不收音。

- 一台 PC

软件环境

- 开发板烧写指定 SDK
- PC 上安装 Audacity，下载链接见 1.1
- 测试音频，下载链接：https://echocatzh.github.io/Demo-of-DeepComplexAEC/samples/simu_test/reference/clean/50_lpb.wav
- 在对应的 dts 配置文件中将 i2s-m-aec-sclk-sel 配置成

```
&i2s_inner_slv0 {
    status = "okay";
    /*i2s - pad0*/
    i2s-m-aec-cycle-sel = <0>;    /*23-22*/
    i2s-m-aec-sclk-sel = <0>;     /*21*/
    i2s-inner-codec-en = <1>;     /*20*/
    i2s-exter-codec-en = <0>;     /*19*/
```

```
};
&i2s_inner_mst0 {
    status = "okay";
    /*i2s - pad0*/
    i2s-m-aec-cycle-sel = <0>;    /*23-22*/
    i2s-m-aec-sclk-sel = <0>;     /*21*/
    i2s-inner-codec-en = <1>;     /*20*/
    i2s-exter-codec-en = <0>;     /*19*/
```

  在驱动文件 sound/soc/axera/ax-actt.c 中删除#define INTERNAL_REF 宏定义

- 然后确认输入方式，则将 layout 设置成 MIC_REF 或者 REF_MIC 对应模式。注意：这里一定要软硬件对应。

➢ 测试步骤

1. 开发板上电，确认驱动成功加载
2. 将测试音频放到测试命令执行的目录下，并命名为 audio_test.wav
3. 测试硬件信号和回声通路

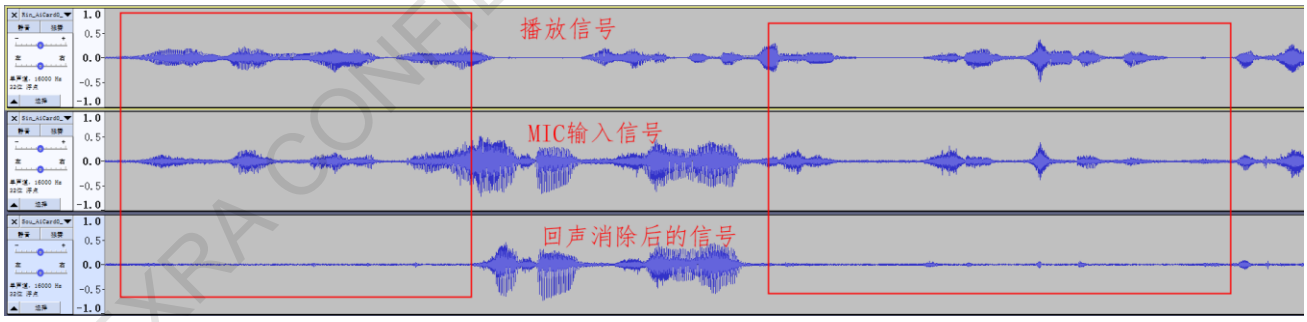
- 执行 sample_audio ai -D 0 -d 0 --aec-mode 2 --routing-mode 0 --layout 1 --save-file 1 & sample_audio ao -D 0 -d 1 -i audio_test.wav
- 调节 AI 输入音量 tinymix -D 0 set 0 0; tinymix -D 0 set 3 60（这一步根据实际情况，尽量保证播放信号和 MIC 接收后的信号幅值相差不大即可）
- 对麦克风说话，5 秒后 CTRL+C 停止程序
- 生成生成 Sin_AiCard0.wav、Rin_AiCard0.waf、Sou_AiCard0.wav 三个文件，导入电脑用 Audacity 打开，查看 Sin_AiCard0.wav(MIC 输入信号)、Rin_AiCard0.waf(播放信号)是否对应。

7) 测试算法效果

执行 sample_audio ai -D 0 -d 0 --aec-mode 2 --routing-mode 0 --layout 1 --save-file 1 -w 1 & sample_audio ao -D 0 -d 1 -i audio_test.wav

5 秒后 CTRL+C 停止程序

生成 audio.wav、Sin_AiCard0.wav(MIC 输入信号)、Rin_AiCard0.waf(播放信号)、Sou_AiCard0.wav(回声消除后的信号)。



2.2.4 异常说明以及处理方法

➢ 问题描述：ref 声音消顶导致的 out 声音异常

解决方案：尝试降低 audio out 的音量来解决

/usr/local/bin/tinymix -D 0 set 11 <xxx> //输出左声道模拟增益

/usr/local/bin/tinymix -D 0 set 12 <xxx> //输出右声道模拟增益

- 问题描述：MIC 声音太小和播放声音相差太大，影响回声消除效果

  解决方案：尝试增大输入音量或者降低输出音量来解决

  /usr/local/bin/tinymix -D 0 set 0 0 //关闭 ALC

  /usr/local/bin/tinymix -D 0 set 3 <xxx> //输入左声道数字增益

  /usr/local/bin/tinymix -D 0 set 4 <xxx> //输入右声道数字增益

  /usr/local/bin/tinymix -D 0 set 11 <xxx> //输出左声道模拟增益

  /usr/local/bin/tinymix -D 0 set 12 <xxx> //输出右声道模拟增益

2.3 音量增益调节策略

2.3.1 音量增益组成

音量增益调节是可以大致分为两个部分：codec 部分和软件部分。

2.3.2 codec 增益

Codec 部分的增益包括模拟增益和数字增益（不使用 ALC 时，可调节这两个增益。）。

对于数字增益和模拟增益之间的调节建议优先调试模拟增益，然后再调节数字增益。但是数字增益不适合调节特别大，因为数字增益是连同噪声一起放大的。

2.3.3 软件增益

软件部分的增益包括 VqeVolume 和 AGCl 两部分。

VqeVolume 范围时 0-10，默认是 1.0。0~1.0 之间是减小，1.0~10.0 之间是增大，所以音量可增加可减小，但是不能防止削顶。

AGC 可以根据自己的需求设置指定增益，让声音达到指定大小，只可增大，但是可以防止削顶。

2.3.4 增益调试建议

首先调试 codec 的模拟增益，然后 codec 数字增益稍微调大一点(比如 15 左右)，然后再根据实际需求去调试软件部分的增益，如果软件部分增益不满足需求，最终再去尝试将 codec 的数字增益放大。此外，建议软件部分降噪常开，能更好的保证数据的质量。

2.3.5 ALC 使用注意项

1. RX 的 ALC 功能（自动增益调节）和手动设置的模拟增益、数字增益是互斥的。也就是说 ALC 打开的情况下，手动配置的模拟增益、数字增益是无效的。ALC 使用期间会自动将模拟增益、数字增益均设置成 0DB。关闭 ALC 之后，需要客户重新配置模拟增益、数字增益成目标值，否则模拟增益、数字增益仍以 0DB 进行后续的采集。
2. ALC 功能现在是需要操作 tinymix –D 0 set 0 1 打开的。ALC 要在开启录音功能之前打开，并且在 ALC 使用期间不要手动修改模拟增益、数字增益，否则会影响效果的。

2.4 播放 pop 音处理方法

2.4.1 控制 PA

内置 codec 播放开始和结束的时候，会有啪的一声，为改善这一问题我们对 PA 进行控制。比如我们 630CDemo 板子，PA 对应的 GPIO 分别是 2、15，那么则需要在 AX630C_emmc_arm64_k419.dts(对应工程的 dts)中对节点 gpio-pa-speaker、gpio-pa-lineout 进行如下配置即可：

```
&audio_codec {
    gpio-mic-rp = <&ax_gpio1 7 0>; /* GPIO1_A7 */
    gpio-mic-rn = <&ax_gpio1 6 0>; /* GPIO1_A6 */
    gpio-mic-ln = <&ax_gpio1 5 0>; /* GPIO1_A5 */
    gpio-mic-lp = <&ax_gpio1 4 0>; /* GPIO1_A4 */
    gpio-pa-speaker = <&ax_gpio_ext 2 0>;
    gpio-pa-lineout = <&ax_gpio_ext 15 0>;
    status = "okay";
};
```

1. 将 status 的状态设置成 okay。

2. gpio-pa-speaker、gpio-pa-lineout 这两个节点是输出端 speaker、lineout 对应运放器的 GPIO 管脚（这两个管脚主要用于使能或者关闭运放器的），客户可以根据自己板子情况修改管脚信息。

2.4.2 客户端做淡入淡出处理

客户端可以尝试做淡入淡出，就是在播放数据开始的一段数据和即将结束的一段数据处做淡入淡出处理。如果淡入淡出不好做的话，可以尝试在播放数据的开始或结尾增加一段静音数据，大概 200ms 左右，具体时长以实际测试效果为准。

2.5 ES8156 配置方法

1. 首先我们要把 es8156 驱动加载到内核中：

```
CONFIG_SND_SOC_ES8156=y
```

这个是要在对应工程中的 defconfig 中配置。

2. 修改 dts

按照下面的图片内容修改 sound 里面的内容，然后把其他打开的 I2S 都关闭掉，打开 I2S_MST.

```
simple-audio-card,widgets =
    "Headphone", "Headphone Jack";
simple-audio-card,routing =
    "Headphone Jack", "LOUT",
    "Headphone Jack", "ROUT";
```

```
        simple-audio-card,dai-link@0 {
            format = "i2s";
            cpu {
                sound-dai = <&i2s_mst0>;
            };
            codec {
                sound-dai = <&es8156>;
                system-clock-frequency = <12288000>;
            };
        };
```

```
    &audio_codec {
        gpio-mic-rp = <&ax_gpio1 7 0>; /* GPIO1_A7 */
        gpio-mic-rn = <&ax_gpio1 6 0>; /* GPIO1_A6 */
        gpio-mic-ln = <&ax_gpio1 5 0>; /* GPIO1_A5 */
        gpio-mic-lp = <&ax_gpio1 4 0>; /* GPIO1_A4 */
        gpio-pa-speaker = <&ax_gpio_ext 2 0>;
        gpio-pa-lineout = <&ax_gpio_ext 15 0>;
        status = "disabled";
    };
```

```
    &i2s_inner_mst0 {
        status = "disabled";
        /*i2s - pad0*/
        i2s-m-aec-cycle-sel = <0>;      /*23-22*/
        i2s-m-aec-sclk-sel = <0>;       /*21*/
        i2s-inner-codec-en = <1>;       /*20*/
        i2s-exter-codec-en = <0>;       /*19*/
        i2s-m-exter-codec-en = <0>;     /*18*/
```

```
&i2s_inner_slv0 {
    status = "disabled";
    /*i2s - pad0*/
    i2s-m-aec-cycle-sel = <0>;      /*23-22*/
    i2s-m-aec-sclk-sel = <0>;       /*21*/
    i2s-inner-codec-en = <1>;       /*20*/
    i2s-exter-codec-en = <0>;       /*19*/
```

```
&i2s_mst0 {
    status = "okay";
    /*i2s - pad0*/
    i2s-m-aec-cycle-sel = <0>;     /*23-22*/
    i2s-m-aec-sclk-sel = <0>;      /*21*/
    i2s-inner-codec-en = <0>;      /*20*/
    i2s-exter-codec-en = <1>;      /*19*/
    i2s-m-exter-codec-en = <0>;    /*18*/
    i2s-exter-codec-mst = <1>;     /*17*/
    i2s-m-exter-codec-mst = <0>;   /*16*/
    iis-out-tdm-en = <0>;          /*15*/
    iis-m-out-tdm-en = <0>;        /*14*/
    i2s-m-rx0-sel = <0>;           /*13*/
    i2s-m-rx1-sel = <2>;           /*12-11*/
    i2s-s-rx0-sel = <0>;           /*10-9*/
    i2s-s-rx1-sel = <0>;           /*8-7*/
    i2s-s-sclk-sel = <0>;          /*6-5*/
    tdm-m-rx-sel = <0>;            /*4-3*/
    tdm-s-rx-sel = <0>;            /*2-1*/
    tdm-s-sclk-sel = <0>;          /*0*/
};
```

3. 挂载

将 8156 挂载到对应的 I2C 下面，注意 8156 挂载的 I2C 一定要和硬件对应起来。这部分内容也是在对应的 dts 里面修改。

```
&i2c3 {
    #address-cells = <0x1>;
    #size-cells = <0x0>;
    status = "okay";
    es8156: es8156@9 {
        compatible = "everest,es8156";
        reg = <0x9>;
        #sound-dai-cells = <0>;
    };
```

2.6 双 MIC 使用注意事项

当客户使用双 MIC 方案时，两 MIC 位置同一水平线上，朝向保持一致。

2.7 音频 DMA 内存修改配置

当音频使用 DMA 的时候，DMA 使用的内存大小是可以配置的。

配置的路径：在对应型号的 dts 中修改使用 I2S 节点中的 dma-buffer 信息。

比如：linux-4.19.125/arch/arm/boot/dts/axera/AX620Q_nor_arm32_k419.dts 中 I2S 使用的是 i2s_inner_mst0, i2s_inner_slv0，想设置内存为 256k, 具体修改如下：

```
&i2s_inner_mst0 {
    status = "okay";
    dma-buffer = <256>;
    /*i2s - pad0*/
    i2s-m-aec-cycle-sel = <0>;      /*23-22*/
    i2s-m-aec-sclk-sel = <0>;       /*21*/
    i2s-inner-codec-en = <1>;       /*20*/
    i2s-exter-codec-en = <0>;       /*19*/
    i2s-m-exter-codec-en = <0>;     /*18*/
    i2s-exter-codec-mst = <0>;      /*17*/
    i2s-m-exter-codec-mst = <0>;    /*16*/
    iis-out-tdm-en = <0>;           /*15*/
    iis-m-out-tdm-en = <0>;         /*14*/
    i2s-m-rx0-sel = <0>;            /*13*/
    i2s-m-rx1-sel = <0>;            /*12-11*/
    i2s-s-rx0-sel = <0>;            /*10-9*/
    i2s-s-rx1-sel = <2>;            /*8-7*/
    i2s-s-sclk-sel = <0>;           /*6-5*/
    tdm-m-rx-sel = <0>;             /*4-3*/
    tdm-s-rx-sel = <0>;             /*2-1*/
    tdm-s-sclk-sel = <0>;           /*0*/
};

&i2s_inner_slv0 {
    status = "okay";
    dma-buffer = <256>;
    /*i2s - pad0*/
    i2s-m-aec-cycle-sel = <0>;      /*23-22*/
    i2s-m-aec-sclk-sel = <0>;       /*21*/
    i2s-inner-codec-en = <1>;       /*20*/
```

2.8 FAQ

1. Q：接口调用流程不熟悉，使用起来困难怎么办？

   A: 针对客户使用方便角度，我方提供了相关的使用 sample(sample_audio_lin.c)。该文件中包括了 AI、AO、AENC、ADEC 各模块相关功能使用方法以及参数配置等。客户可以参照使用。

2. Q: 调用 API 接口失败，怎么办？

   A: 排查如下:

   a) 保存日志，查看错误码，针对错误码明确原因；

   b) 详细查看 API 文档，确定是否有违背了相关注意事项；

   c) 还是不明确原因可以联系我方技术支持人员进一步排查。

3. Q: 在音频流程启动后，发现录音、播放或者某些功能异常，怎么办？

   A: 排查如下:

   a) 首先将我方法提供的 sample 编译导入到板端，然后运行对应的功能，查看运行结果是否符合预期；

   b) 如果相同的功能，sample 运行结果正常，客户程序异常，则应排查两者之间配置的参数是否存在差异，排查手段 cat /proc/ax_proc/xxx。

   c) 如果 sample 运行结果也异常，请排查硬件（MIC、SPEAKER 等对应器件）是否正常，硬件的连接方式是否与 sample 软件配置一致（比如 layout 等）。

   d) 经过以上排查还是有问题，请联系我方技术支持人员进一步排查。

4. Q: 音频 3A 效果中，什么时候打开 AEC(回声消除)功能？

   A: 回声消除的使用场景一般是对讲场景下。但是有些其他场景也可以根据需求打开。比如播放端（非对讲模式下）在播放警戒音或者一些录制内容，如果 web 想听取这段播放声则关闭 AEC，如果 web 端不想听取播放声则打开 AEC。

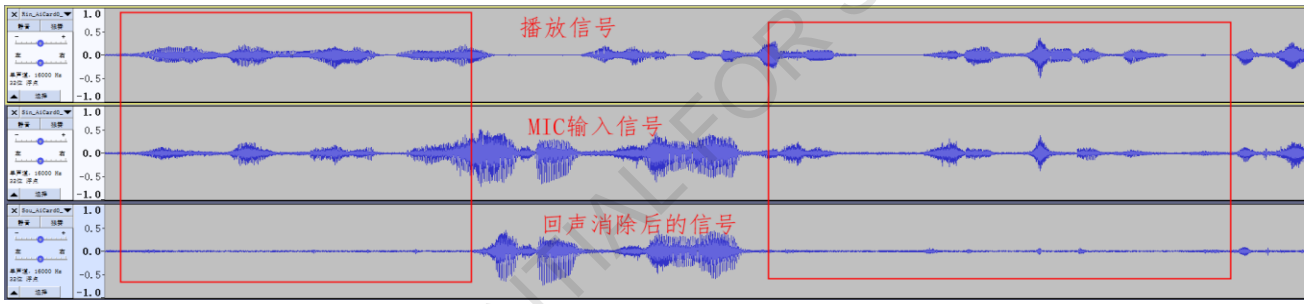
5. Q: 音频 3A 使用中，AEC(回声消除)效果怎么调试？

   A: 调试如下：

   a) 回声消除有两种不同的方式，内回环和外回环。针对两种不同的方式的 AEC 效果调试步骤，本文档在第 2 章节中有详细介绍，供客户参考。

   b) 其中 codec 的增益很重要，无论客户使用哪种 codec，请尽量保持 MIC 采集数据与参考数据幅值尽量接近。

   c) 排查手段可以使用 dump 工具，将保存下来的 MIC 数据和参考数据在 PC 端使用软件进行分析，查看当前情况。



   d) 推荐使用定点模式的。如果客户认为默认回声抑制效果不满足要求，可以考虑提升回声抑制等级 AX_ROUTING_MODE_E（定点模式）。和 ANS、AGC 一起搭配使用效果更佳。

6. Q: 音频播放时有 POP 音，怎么办？

   A: POP 音的产生有不同的原因，详细的规避方案可以参考 2.4 章节。

7. Q: 使用 EQ 时，发现配置后的效果不明显怎么办？

   A: 排查如下：

   a) 检查 EQ 使用的参数是否在允许范围内。

   b) 当 EQ 使用的中心频率非常接近或者等于数据采样率的一半时，效果确实是不明显的，这时可以提升数据采样率来改善效果。

8. Q: AI 或者 AO 使能时，提示 pcm_open 卡住，怎么办？

   A: 排查如下：

   a) 当前音频声卡的打开方式是阻塞模式，即 open 声卡之前需要将声卡 close。

b) 客户从流程上排查是否在 open 前 close，另外还要确定声卡的 fd 确实是被关闭掉了。

9. Q: 数据异常问题，怎么办？

A: 排查如下：

a) 针对某个环节突然数据异常了，直接使用 dump 工具（使用请参考 AUDIO API 文档 5.5 章节）。根据保存下来的数据逐步排查是哪个环节出现的问题。

b) 然后抓取 proc 信息，确认出现问题环节参数是否异常。

10. Q: 硬件电路对音效的影响，怎么调试？

A: 操作如下：

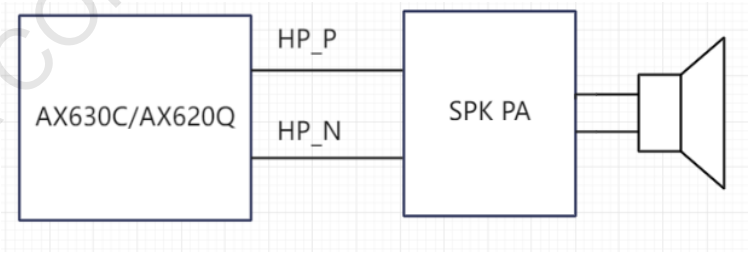
a) 620E 内置 codec 可以通过调节 TX gain 来确保最大不失真输出，这部分可以参考《AX630C_AX620Q 音频硬件设计和调试指南_V1.0》

b) 至于外部喇叭 PA 部分，由于每位客户使用的 PA 不一样，并且 PA 输入电压不同，所以喇叭处的音量和噪声要根据实际使用的 PA spec 去调节输入的阻值和容值，这部分客户一般都是自己解决。

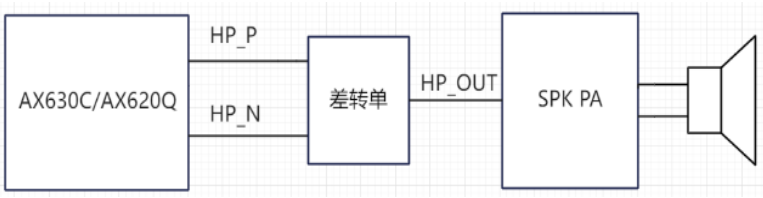
11. Q: SPEAKER 设计有哪些方式？

A: AX630C/AX620Q 内置 DAC 可外接功放 PA。根据 PA 输入方式推荐以下两种接法：

a) 差分输入接法：



b) 单端输入接法:

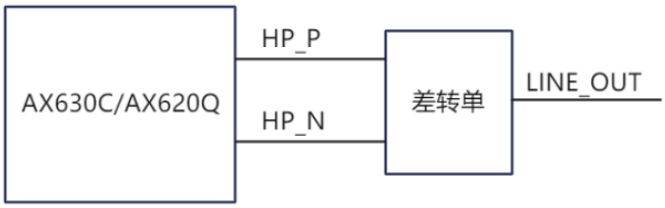


c) 如果 PA 支持差分接法，建议使用差分输入。若只支持单端则需要增加一颗差转单芯片。

12. Q: LINEOUT 设计有哪些方式？

A: AX630C/AX620Q 的内置 DAC 支持 LINE_OUT 应用，推荐如下接法：

a)



b) 为了获得更好的 SNR 和低 noise，推荐在 LINE_OUT 应用中增加一颗差转单芯片。

c) 差转单芯片 AVL List：

AVL List

Part Number	Package	Description
SGM8903	TSSOP-14	2.7V~5.5V，支持差分输入
TPF632A/C	TSSOP-14	2.7V~5.5V，支持差分输入
DIO2133	TSSOP-14	2.7V~5.5V，支持差分输入

13. Q: PA 最大不失真输出，怎么调试？

A: 操作可包括如下内容：

a) 检查此时设置的 TX A-gain 和 D-gain 的值：

```
/customer/glibc # ./tinymix contents
Number of controls: 45
ctl     type    num     name                            value
0       BOOL    1       ALC ENABLE                      Off
1       INT     1       RX LEFT ANA GAIN                54 (range 0→54)
2       INT     1       RX RIGHT ANA GAIN               54 (range 0→54)
3       INT     1       RX LEFT DIG GAIN                15 (range 0→120)
4       INT     1       RX RIGHT DIG GAIN               15 (range 0→120)
5       INT     1       ALC TARGET LEVEL                18 (range 0→20)
6       INT     1       ALC RMS MAX                     20 (range 0→20)
7       INT     1       ALC GATE THRESHOLD              140 (range 0→200)
8       INT     1       ALC GAIN MAX                    140 (range 0→200)
9       BOOL    1       ADC MUTE                        Off
10      BOOL    1       DAC MUTE                        Off
11      INT     1       TX LEFT ANA GAIN                49 (range 0→49)
12      INT     1       TX RIGHT ANA GAIN               49 (range 0→49)
13      INT     1       TX LEFT DIG GAIN                15 (range 0→120)
14      INT     1       TX RIGHT DIG GAIN               15 (range 0→120)
15      BOOL    1       TX LR SWAP                      Off
```

b) gain 值 49 代表模拟增益为 0 dB，D-gain 值 0 代表数字增益为 0dB。在此条件下 DAC 可以确保输出波形不失真。如果在芯片输出端 HP_P/N 测量出来波形失真，建议优先检查软件 gain 设置。

c) DAC 在 24K/48K/96K 采样率下输入推荐使用 0.95Vrms -0.5db 1KHZ 信号测试，

8K/16K/32K 采样率下输入推荐使用 1Vrms 0db 1KHZ 信号测试。

d) 当播放人声音源时，声音小的情况下可以调节增益，确保输出不失真破音情况下，首先调试 codec 的模拟增益，然后 codec 数字增益稍微调大一点(比如 15 左右)，然后再根据实际需求去调试软件部分的增益，如果软件部分增益不满足需求，最终再去尝试将 codec 的数字增益放大。

e) 软件部分的增益包括 VqeVolume 和 AGC 两部分。VqeVolume 范围时 0-10，默认是 1.0。0~1.0 之间是减小，1.0~10.0 之间是增大，所以音量可增加可减小，但是不能防止削顶。AGC 可以根据自己的需求设置指定增益，让声音达到指定大小，只可增大，但是可以防止削顶

f) 确保 DAC 不失真后根据 PA 输出端的波形再调整 PA 的放大系数，PA 的类型如果是 Class AB 类的可以直接在输出端测量差分输出波形，如果是 Class D/K 类则需要在 PA 输出端加上一个低通滤波电路，将开关调制频率滤除，然后测量滤波器后端的信号即是有效信号。 如下图是 A-gain 和 D-gain 0dB 下由于 PA 放大系数太大导致的输出波形削顶失真。

g) 减小 PA 放大系数后的最大输出不失真波形。

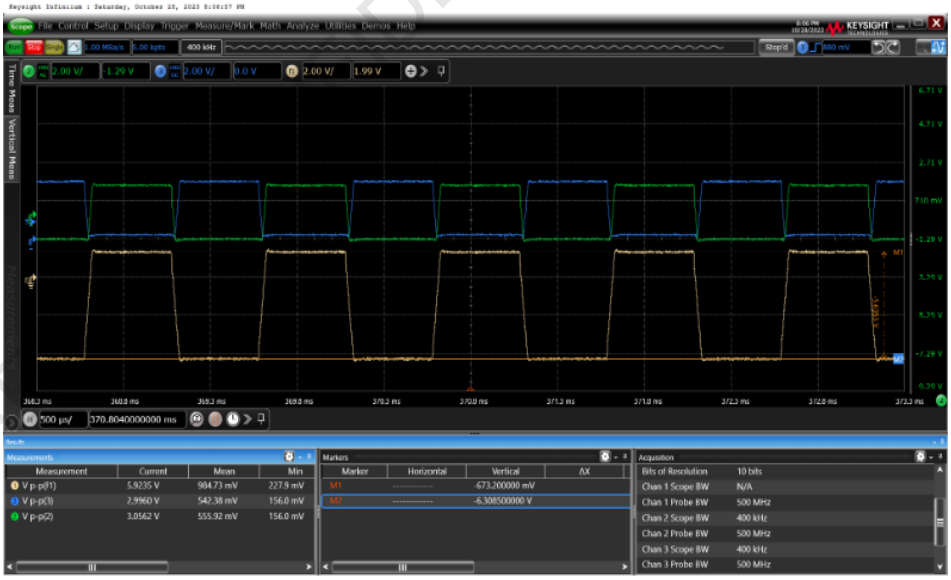


图1-8 Class D VDD=3V3的 PA削顶失真

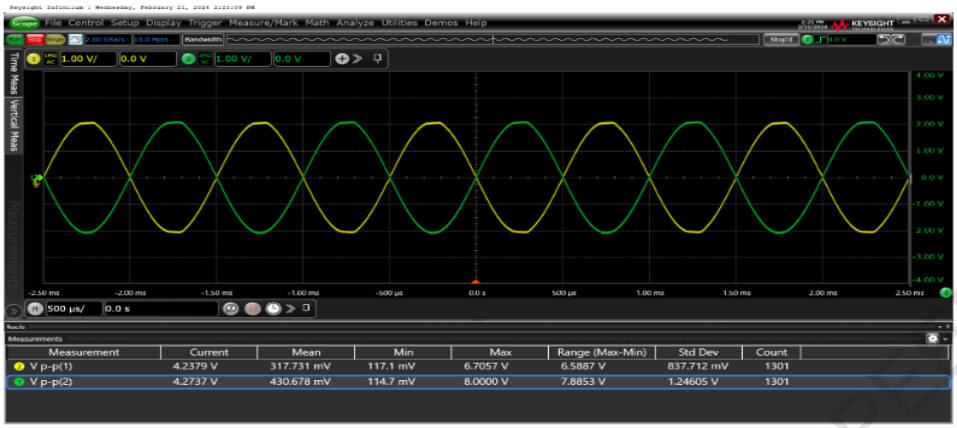


图1-9 Class AB VDD=5V的 PA削顶失真

h） 减小 PA 放大系数后的最大输出不失真波形

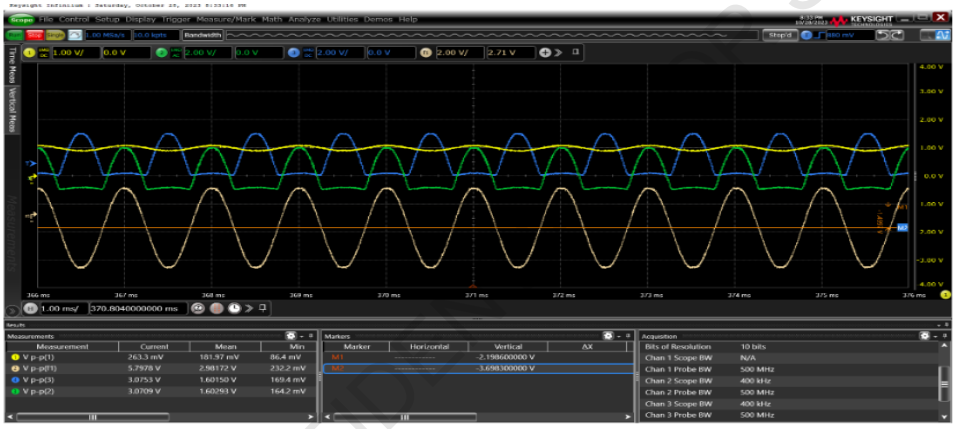


图1-10 Class D VDD=3V3的 PA不失真下最大输出波形