H616 AndroidQ

音频模块使用说明书

1.0

2019.12.05

文档履历

版本号	日期	制/修订人	内容描述
1.0	2019.12.05		

目录

1. 前言

1.1 编写目的

本文档目的是让开发者了解 H616 音频系统框架，能够在 H616 平台上开发新的音频方案。

1.2 适用范围

本模块说明适用于 H616 Android Q + Linux4.9 平台。

1.3 相关人员

音频系统开发人员。

1.4 相关术语

- ALSA：Advanced Linux Sound Architecture
- DMA：即直接内存存取, 指数据不经 cpu, 直接在设备和内存, 内存和内存, 设备和设备之间传输.
- OSS：Open Sound System
- 样本长度 (sample)：样本是记录音频数据最基本的单位，常见的有 8 位和 16 位
- 通道数 (channel)：该参数为 1 表示单声道，2 则是立体声。
- 帧 (frame)：帧记录了一个声音单元，其长度为样本长度与通道数的乘积。
- 采样率 (rate)：每秒钟采样次数，该次数是针对帧而言。
- 周期 (period)：音频设备一次处理所需要的帧数，对于音频设备的数据访问以及音频数据的存储，都是以此为单位。
- 交错模式 (interleave)：是一种音频数据的记录模式，在交错模式下，数据以连续帧的形式存放，即首先记录完帧 1 的左声道样本和右声道样本（假设为立体声格式），再开始帧 2 的记录，而在非交错模式下，首先记录的是一个周期内所有帧的左声道样本，再记录右声道样本，数据是以连续通道的方式存储。不过多数情况下，我们只需要使用交错模式就可以了。

- HDMIaudio：内置 hdmi 音频接口
- SPDIF：外置音响音频设备接口，一般使用同轴电缆或光纤接口
- I2S：外置音频通道接口
- AGC：录音自动增益控制
- DRC：音频输出动态范围控制
- daudio：数字音频接口，可配置成 i2s/pcm 格式标准音频接口
- aif：数字音频接口
- xrun：音频流异常状态

2. H616 音频系统框架概述

2.1 H616 原型机音频硬件框架图

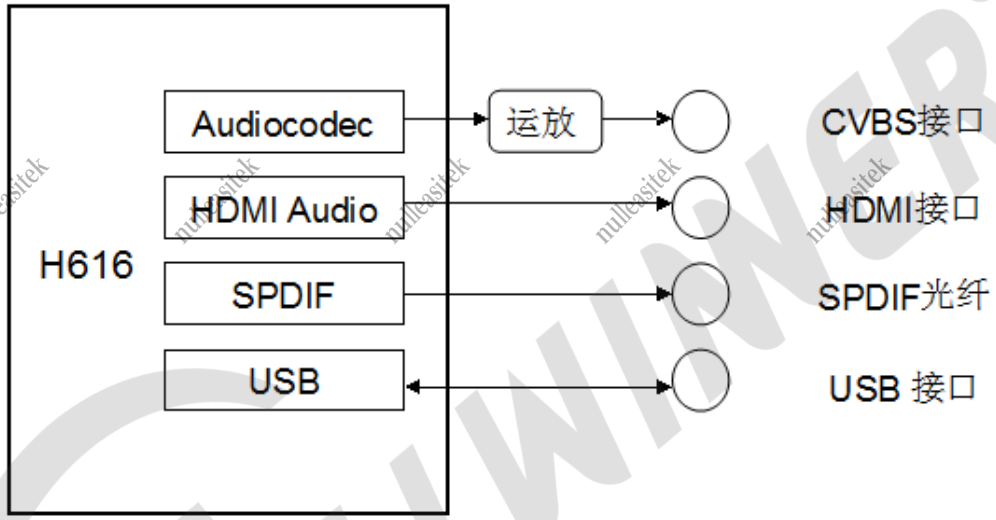

图 1: H616 原型机硬件框图

输入以下命令查看系统当前音频设备节点：

```
console:/ # cat proc/asound/cards
 0 [audiocodec ]: audiocodec - audiocodec
                    audiocodec
 1 [sndspdif ]: sndspdif - sndspdif
                    sndspdif
 2 [sndahub ]: sndahub - sndahub
                    sndahub
 3 [sndhdmi ]: sndhdmi - sndhdmi
                    sndhdmi
 4 [snddaudio0 ]: snddaudio0 - snddaudio0
                    snddaudio0
```

2.2 H616 软件框架图

H616 音频软件框架如图所示，H616 盒子有一套独立的音频输入输出策略，同时具备原生系统所不具备的功能，例如支持 HDMI，USB，CVBS 等接口的热插拔，支持单/多路输出，支持音频透传等。

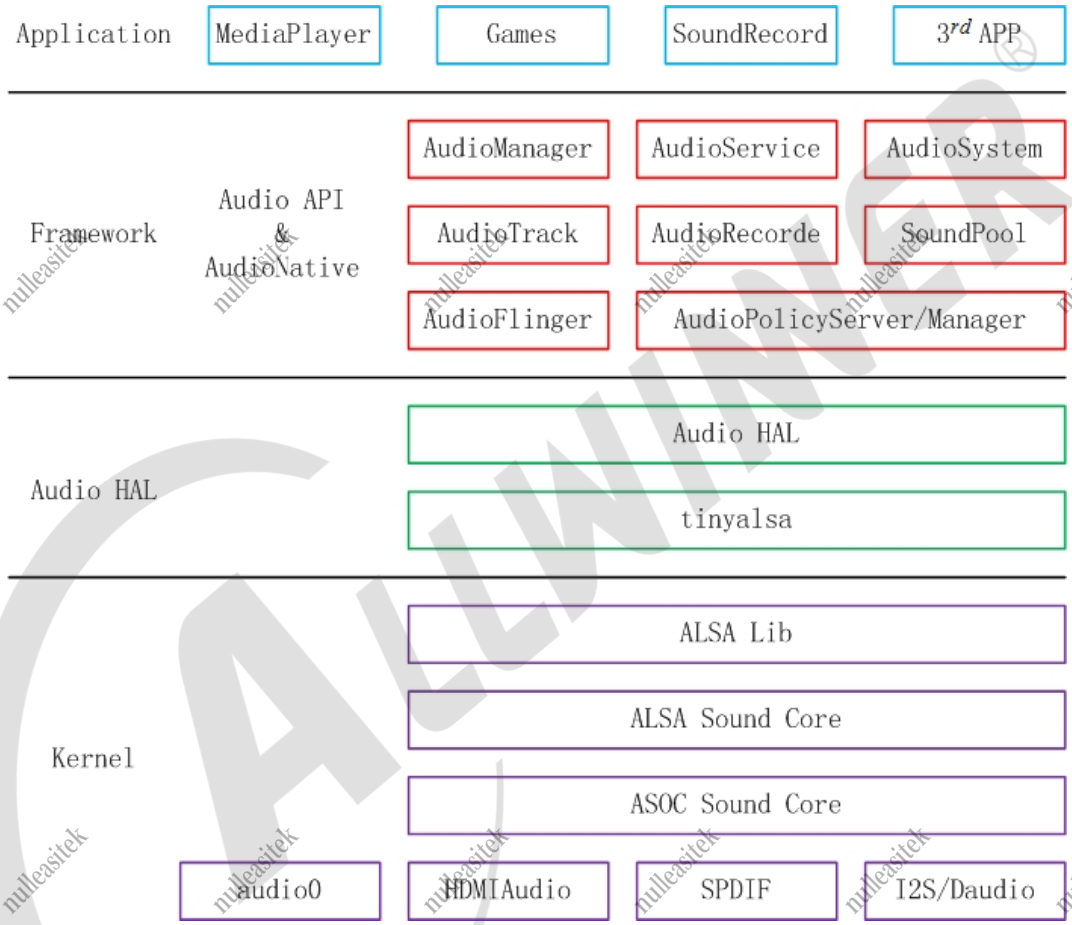

图 2: H616 音频软件框架图

3. H616 音频模块介绍

在 H616 中，存在 7 个音频设备，分别是：

- daudio0
- daudio1（接 HDMI）
- daudio2
- daudio3
- audiocodec（line out）
- DMIC（公版版型未透出）
- SPDIF

硬件框图如图所示：

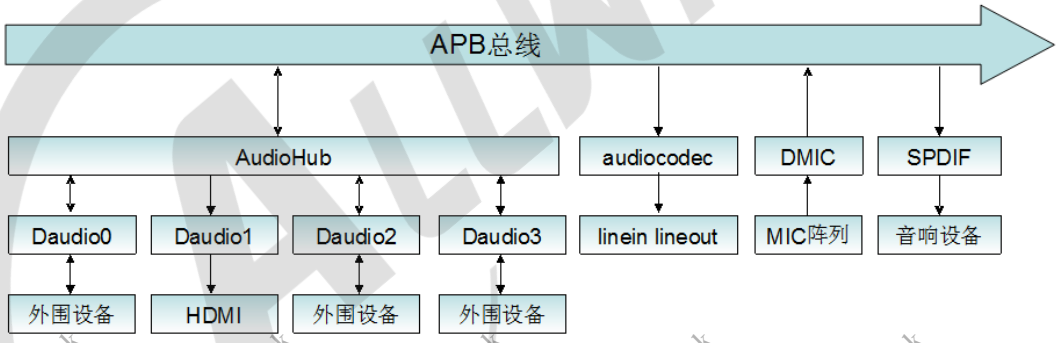

图 3: H616 音频硬件框图

每一个音频设备都采用 asoc 架构实现，asoc 是建立在标准 alsa 驱动层上，为了更好地支持嵌入式处理器和移动设备中的音频 codec 的一套软件体系，asoc 将音频系统分为 3 部分：Machine，Platform 和 Codec。软件框架图如图所示：

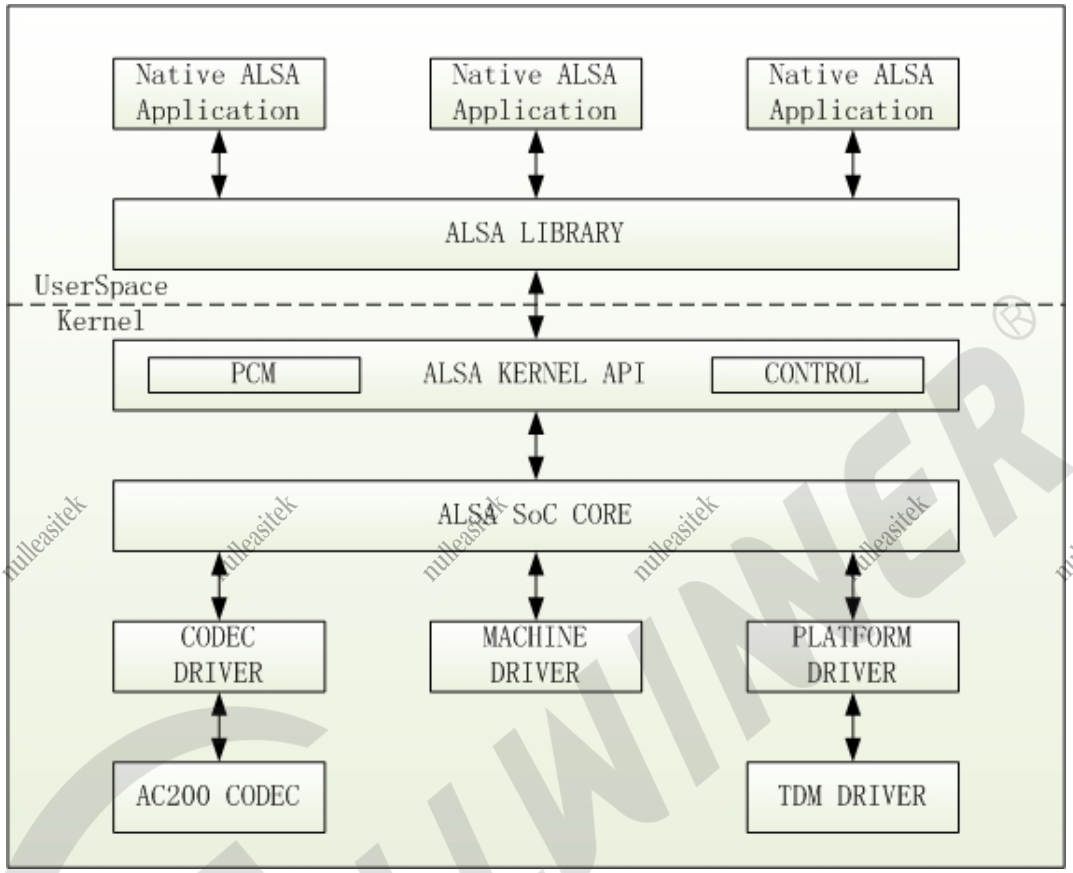

图 4: H616 ALSA 软件框架图

3.1 公共部分

Platform（dma 注册）：

sunxi_dma.c：该文件内处理 dma 部分，主要负责提供注册 platform 设备的公共函数。

3.2 Audio_hub 驱动功能

Audio_hub 是 H616 特有模块，集成了音频的基本输入输出功能，还有硬件混音特殊功能，可应用在卡拉 OK 场景。混音功能具有三个输入端，四路 I2S，即可完成 3 路数据的混音，将混音后的数据通过 HDMI、I2S0、12S2、I2S3 输出。

3.3 audiocodec 驱动功能

audiocodec 是具有数模转换功能的内置模块，可将音频数字信号转换成模拟信号发送出去，通常接 CVBS。

audiocodec 驱动支持以下功能：

- 播放支持多种采样格式（8kHz，11.025kHz，16kHz，22.05kHz，24kHz，32kHz，44.1kHz，48kHz，96kHz，192kHz）
- 支持 mono 和 stereo 模式
- 只支持 playback 模式，不支持 record 模式

sun50iw9-codec.c：目录位于 sound/soc/sunxi 中，负责 audiocodec 音频 codec 的部分，注册为 codec,codec_dai 模型

3.4 Daudio 模块功能

Daudio 驱动具有以下功能：

- 支持多种采样率格式（8kHz，11.025kHz，16kHz，22.05kHz，24kHz，32kHz，44.1kHz，48kHz，88.2kHz，96kHz，176.4kHz，192kHz）
- TDM 模式最多支持 16 个通道
- 支持全双工模式
- 支持 i2s、pcm 配置
- 支持 16-bit、20-bit、24-bit、32-bit 数据精度

sunxi_daudio.c：该文件处理 daudio 部分，在 asoc 中框架中设计为 cpu_dai 模型，其中 platform 也在此注册 sunxi-snddaudio.c：该文件处理 daudio 部分，在 asoc 中框架中设计为 machine 模型

3.5 HDMI 模块功能

HDMI 驱动具有以下功能：

- 支持多种采样率格式（32kHz，44.1kHz, 48kHz，96kHz，192kHz）
- 支持 mono 和 stereo 模式
- 只支持 playback 模式，不支持 record 模式
- 支持 16-bit、20-bit、24-bit、32-bit 数据精度
- 支持 raw 数据输出

sunxi_daudio.c ：该文件处理 daudio 部分，在 asoc 中框架中设计为 cpu_dai 模型，其中 platform 也在此注册 sndhdmi.c：该文件处理 HDMI 解码库接口设置部分，在 asoc 中框架中设计为 codec 模型 sunxi-sndhdmi.c：该文件处理 daudio1 部分，在 asoc 中框架中设计为 machine 模型

3.6 DMIC 模块功能

DMIC 驱动具有以下功能：

- 支持多种采样率格式（8kHz，16kHz，24kHz，32kHz，44.1kHz, 48kHz）
- 支持 16-bit、24-bit 数据精度
- 支持最高 8 通道
- 多个 DMIC 通道同时使用时，采样率必须一致，使能必须同开同关
- 支持过采样率 64OSR 和 128OSR

sunxi_dmic.c ：该文件处理 dmic 部分，在 asoc 中框架中设计为 cpu_dai 模型，其中 platform 也在此注册 sunxi-snddmic：该文件处理 sunxi-snddmic 部分，在 asoc 中框架中设计为 machine 模型

3.7 SPDIF 模块功能

SPDIF 驱动具有以下功能：

- 支持多种采样率格式（22.05kHz，24kHz，32kHz，44.1kHz，48kHz，88.2kHz，96kHz，176.4kHz，192kHz）
- 支持 mono 和 stereo 模式
- 支持 16-bit、20-bit、24-bit 数据精度

- 支持 raw 数据输出

sunxi-sndspdif.c：该文件处理 spdif 部分，在 asoc 中框架中设计为 machine 模型 sunxi_spdif.c：该文件处理 spdif 部分，在 asoc 中框架中设计为 cpu_dai 模型，其中 platform 也在此注册

4. H616 音频配置

4.1 源码结构

代码结构如下所示：

```
`/longan/kernel/llinux-4.9/sound/soc/sunxi$` tree
├── Kconfig
├── Makefile
├── spdif-utils.c
├── sunxi_ahub.c
├── sunxi_ahub_cpudai.c
├── sunxi_ahub_daudio.c
├── sunxi_ahub.h
├── sunxi-cpudai.c
├── sunxi-daudio.c
├── sunxi-daudio.h
├── sun50iw9-codec.c
├── sun50iw9-codec.h
├── sun50iw9-sndcodec.c
├── sunxi-dmic.c
├── sunxi-dmic.h
├── sunxi-sndahub.c
├── sunxi-snddaudio.c
├── sunxi-snddmic.c
├── sunxi-sndhdmi.c
├── sunxi-sndspdif.c
├── sunxi-spdif.c
└── sunxi-spdif.h
```

4.2 内核配置

4.2.1 menuconfig 配置

在编译服务器上，目录为 \longan\kernel\linux-4.9 上，输入命令

make ARCH=arm64 menuconfig

执行结果如图所示：

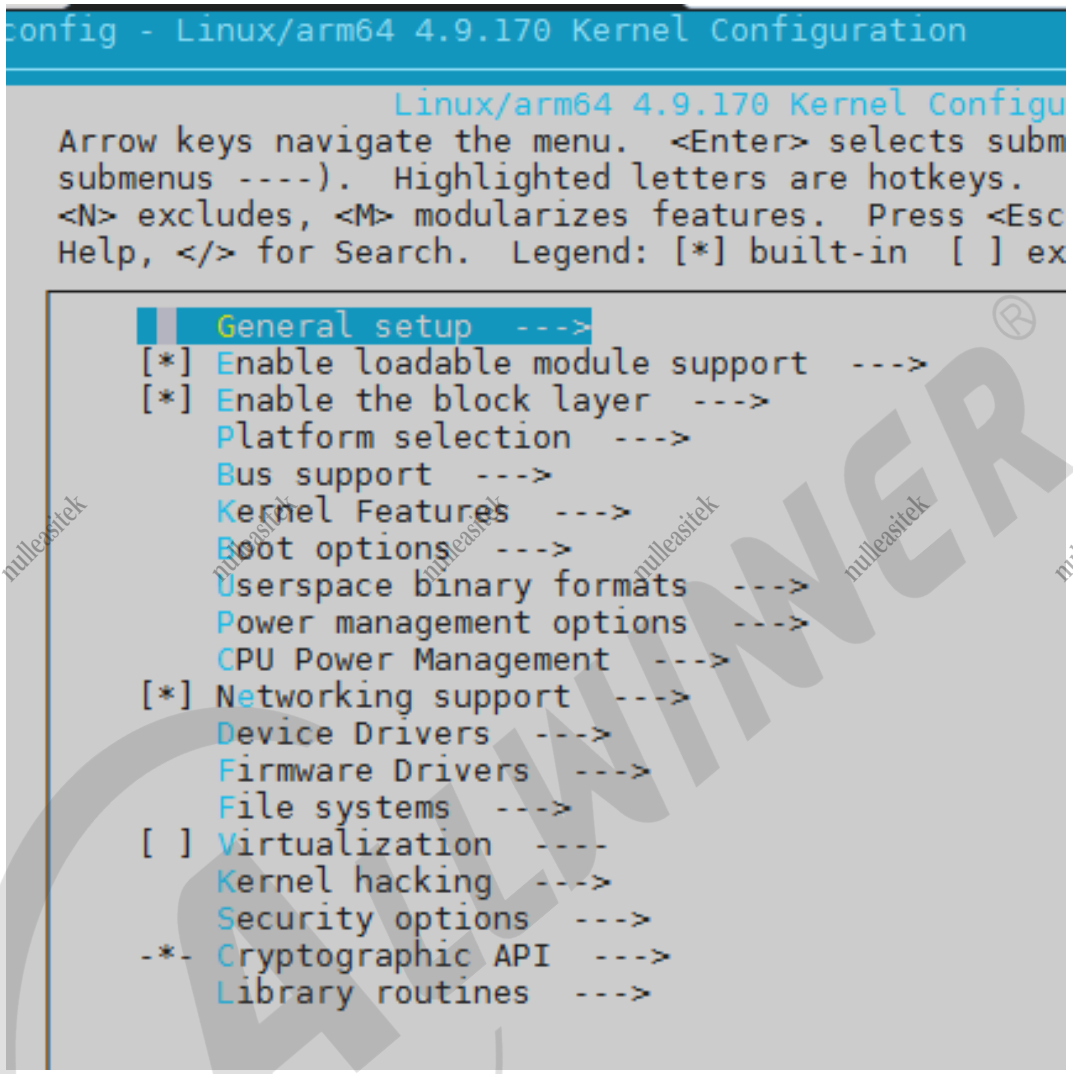

图 5: menuconfig 配置

音频驱动配置：

- Device Drivers -->
- <*> Sound card support -->
- <*> Advanced Linux Sound Architecture -->

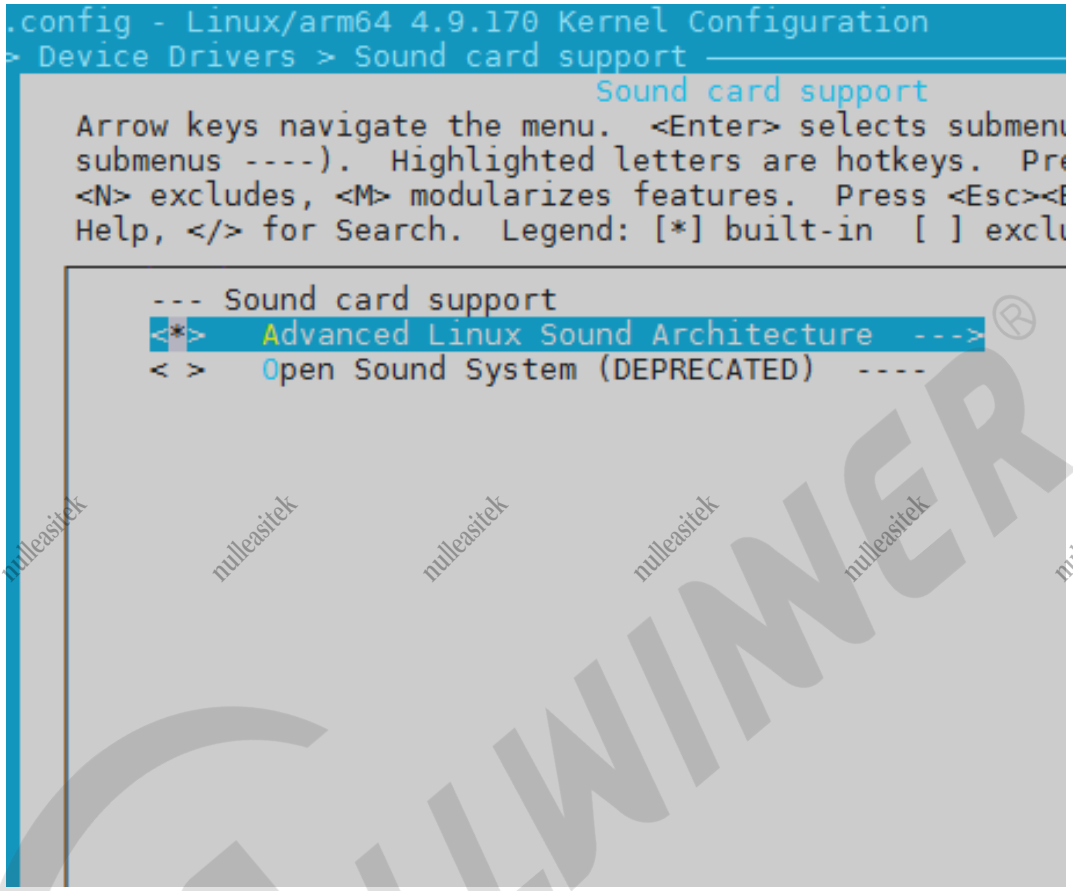

图 6: menuconfig 音频驱动配置

- <*> ALSA for SoC audio support -->
- <*> Allwinner SoC Audio support -->

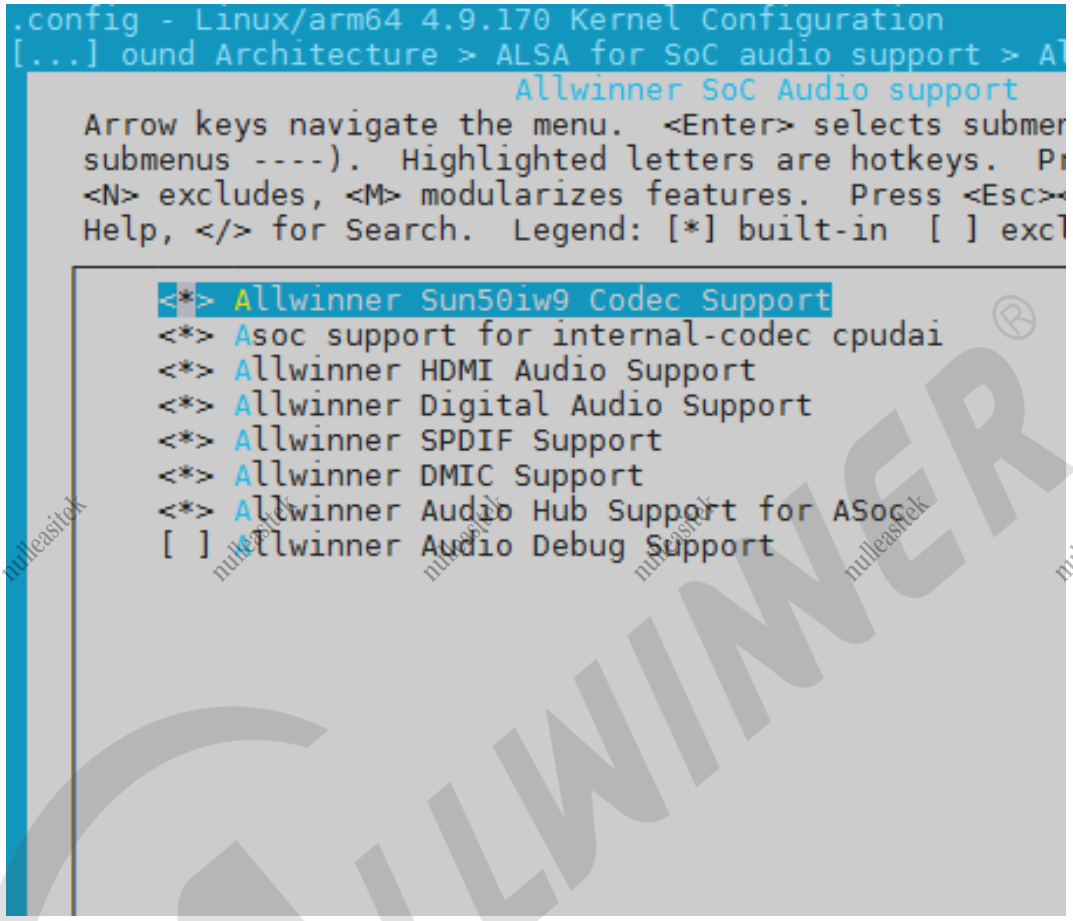

图 7: menuconfig Allwinner 音频驱动配置

H616 所有音频模块都直接编入内核。

4.3 board.dts 配置

配置文件目录：longan/device/config/chips/h616/configs/p2/board.dts 以 daudio0 为例，配置说明如下：

配置项	配置项含义
status	是否开启ahub_daudio0/snddaudio0，okay：开启，disabled：不开启
frametype	长帧或短帧 0: long frame = 2 clock width; 1: short frame
pcm_lrckr_period	未使用
slot_width_select	数据word的宽度，对i2s模式，pcm模式都有效。16bits/20bits/24bits/32bits
daudio_master	Master/slave模式：1:daudio0 slave; 4:daudio0 master
audio_format	1 SND_SOC_DAIFMT_I2S(standard i2s format). use 表示标准i2s格式； 2 SND_SOC_DAIFMT_RIGHT_J(right justfied format). 表示右对齐格式； 3 SND_SOC_DAIFMT_LEFT_J(left justfied format) 表示左对齐格式； 4 SND_SOC_DAIFMT_DSP_A 短帧模式 并设置 frame_width 为0.短帧； 5 SND_SOC_DAIFMT_DSP_B 长帧模式 并设置frame_width 为1.长帧；
signal_inversion	信号的翻转，比如标准的I2S模式，如果lrck翻转是模式，那么用示波器测量，左右声道是跟标准i2s模式相反的。 如果bclk是翻转模式，那么用示波器测量，BCLK信号是翻转的。 1 SND_SOC_DAIFMT_NB_NF(normal bit clock + frame) use 表示bclk采用正常模式，lrck也正常模式 2 SND_SOC_DAIFMT_NB_IF(normal BCLK + inv FRM) 表示bclk采用正常模式，lrck采用翻转模式 3 SND_SOC_DAIFMT_IB_NF(invert BCLK + nor FRM) use 表示bclk采用翻转模式，lrck采用正常模式 4 SND_SOC_DAIFMT_IB_IF(invert BCLK + FRM) 表示bclk采用翻转模式，lrck采用翻转模式
tdm_config	I2S或PCM选择 0:pcm 1:i2s
mclk_div	时钟分频，默认0x00

图 8: board.dtsdaudio 配置说明

```
ahub_daudio0:ahub_daudio0@0x05097000{
    /* for choose the corresponding pins */
    pinctrl-0 = <&ahub_daudio0_pins_c>;
    pinctrl-1 = <&ahub_daudio0_pins_d>;
    pinconfig = <0x01>;
    frametype = <0x00>;
    pcm_lrck_period = <0x20>;
    slot_width_select = <0x20>;
    daudio_master = <0x04>;
    audio_format = <0x01>;
    signal_inversion = <0x01>;
    tdm_config = <0x01>;
    mclk_div = <0x00>;
    status = "okay";
};

snddaudio0:sound@0{
    status = "okay";
};
```

4.3.1 蓝牙 SCO 驱动挂载

蓝牙 SCO 设备挂在在 daudio2，配置请参照 board.dts 配置对 daudio2 声卡进行配置。

4.4 audiocodec 通路配置说明

H616 audiocodec 仅支持 Lineout 输出。

```
cupid-p2:/ # tinymix
Mixer name: 'audiocodec'
Number of controls: 16
ctl     type    num     name                                    value

0       ENUM    1       codec hub mode                          hub_disable
1       INT     1       digital volume                          0
2       INT     1       LINEIN to output mixer gain control     3
3       INT     1       FMIN to output mixer gain control       3
4       INT     1       LINEOUT volume                          31
5       BOOL    1       LINEOUT Switch                          On
6       BOOL    1       Left Output Mixer DACL Switch           On
7       BOOL    1       Left Output Mixer DACR Switch           Off
8       BOOL    1       Left Output Mixer FMINL Switch          Off
9       BOOL    1       Left Output Mixer LINEINL Switch        On
10      BOOL    1       Right Output Mixer DACL Switch          Off
11      BOOL    1       Right Output Mixer DACR Switch          On
12      BOOL    1       Right Output Mixer FMINR Switch         Off
13      BOOL    1       Right Output Mixer LINEINR Switch       On
14      ENUM    1       Left LINEOUT Mux                        LOMixer
15      ENUM    1       Right LINEOUT Mux                       ROMixer
```

图 9: audiocodec 路由通路

4.4.1 系统音频场景

ctrl 配置：

下列配置默认以左对左，右对右的方式进行配置，左右相关可以自行调节。

4.4.1.1 系统 Lineout 输出

number	ctl_name	value
1	LINEOUT Volume	0-31(0 为 mute)
2	LINEOUT Switch	1
3	Left Output Mixer DACL Switch	1
4	Right Output Mixer DACR Switch	1

4.5 Audio_hub 配置说明

4.5.1 Audio_hub 音频路径配置

```
Mixer name: 'sndahub'
Number of controls: 21
ctl     type    num     name                                    value
0       ENUM    1       DAM1Chan2 Src Select                    NONE
1       ENUM    1       DAM1Chan1 Src Select                    NONE
2       ENUM    1       DAM1Chan0 Src Select                    NONE
3       ENUM    1       DAM0Chan2 Src Select                    NONE
4       ENUM    1       DAM0Chan1 Src Select                    NONE
5       ENUM    1       DAM0Chan0 Src Select                    NONE
6       ENUM    1       I2S3 Src Select                         NONE
7       ENUM    1       I2S2 Src Select                         NONE
8       ENUM    1       I2S1 Src Select                         NONE
9       ENUM    1       I2S0 Src Select                         NONE
10      ENUM    1       APBIF2 Src Select                       NONE
11      ENUM    1       APBIF1 Src Select                       NONE
12      ENUM    1       APBIF0 Src Select                       NONE
13      BOOL    1       I2S0IN Switch                           off
14      BOOL    1       I2S0OUT Switch                          off
15      BOOL    1       I2S1IN Switch                           off
16      BOOL    1       I2S1OUT Switch                          off
17      BOOL    1       I2S2IN Switch                           off
18      BOOL    1       I2S2OUT Switch                          off
19      BOOL    1       I2S3IN Switch                           off
20      BOOL    1       I2S3OUT Switch                          off
```

图 10: Audio_hub 路由通路

上图控件可以分成两部分：

- 控件 0-12 用于表示对应的 rxif 所连接的 txif，可以配置的值如下所示
  - NONE
  - APBIF_TXDIF0
  - APBIF_TXDIF1
  - APBIF_TXDIF2
  - I2S0_TXDIF
  - I2S1_TXDIF
  - I2S2_TXDIF
  - I2S3_TXDIF
  - DAM0_TXDIF
  - DAM1_TXDIF

- 控件 13-18 为是为了 DAPM 机制所需要而是使用的虚拟控件，是用时需要打开所需要使用 PIN 的 IN/OUT switch

以下通过几个实例说明 Audio_hub 的路径配置

- APB0->I2S0 播放

number	ctl_name	value
9	I2S0 Src Select	APBIF_TXDIF0
14	I2S0OUT Switch	On

- APB0 ->DAM1 Chan 1-> I2S2 播放

number	ctl_name	value
1	DAM1Chan1 Src Select	APBIF_TXDIF0
7	I2S2 Src Select	DAM0_TXDIF
18	I2S2OUT Switch	On

- I2S3->DAM0 Chan1->APB2 & APB2-->DAM0 Chan2->APB2 混音录制

number	ctl_name	value
1	DAM1Chan1 Src Select	I2S3_TXDIF
7	DAM1Chan2 Src Select	APBIF_TXDIF2
18	APBIF2 Src Select	DAM1_TXDIF
19	I2S3IN Switch	On

4.6 Audio_hub 操作流程

Audio_hub 驱动设计框图如下，共设计成五个声卡设备，其中音频路径配置通过 Sndahub 声卡配置，APB0、APB1、APB2 分别设计成 Sndahub 声卡下三个设备，Sndaudio0/2/3 为 Daudio 声卡设备，Sndhdmi 与内部 HDMI 相连。

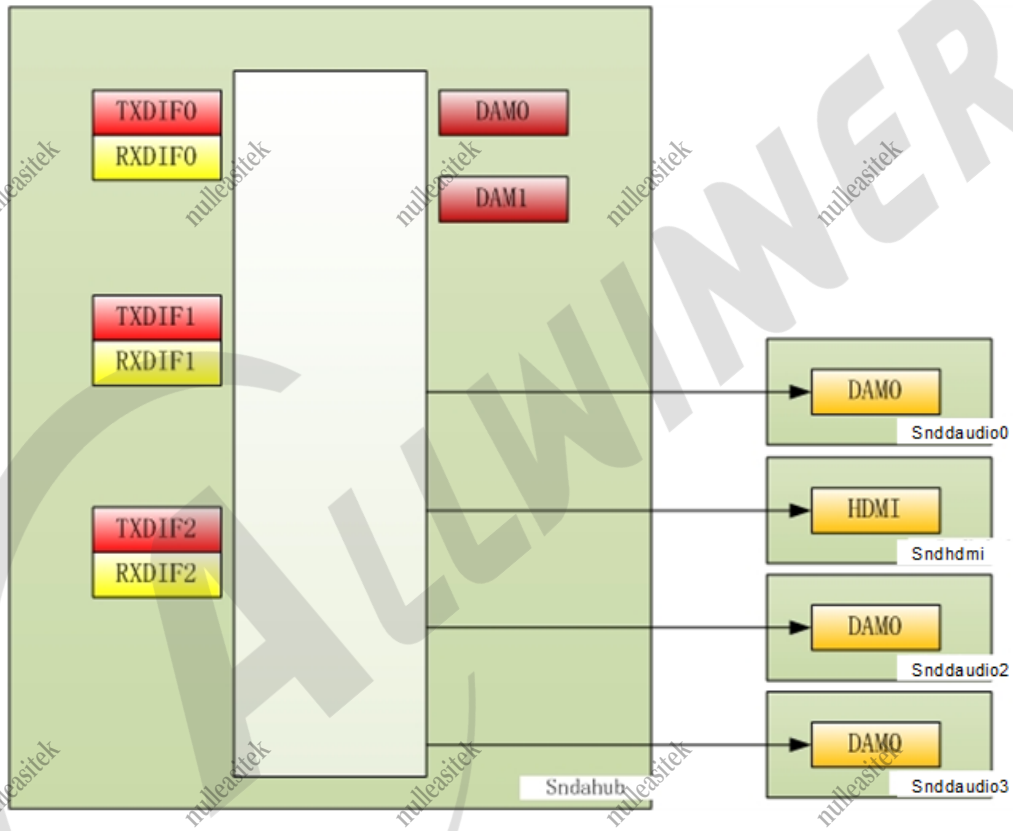

图 11: Audio_hub 驱动设计框图

HDMI 播放操作流程如图所示：

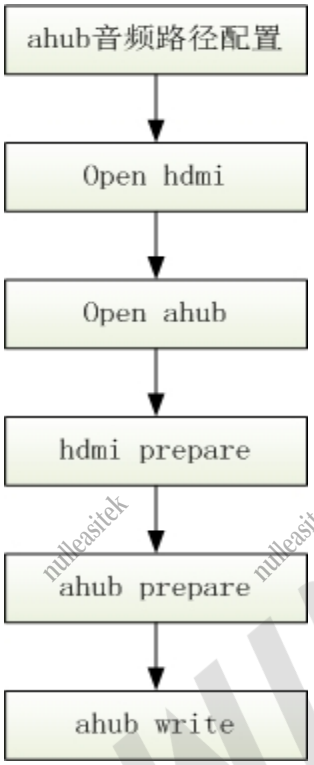

图 12: HDMI 播放操作流程

Daudio 录制流程如图所示：

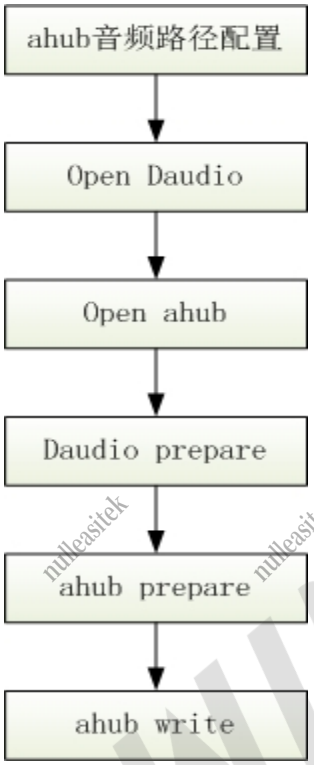

图 13: Daudio 录制流程

5. 音频输入输出切换策略

5.1 音频输出策略

H616 AndroidQ 音频输出策略如图所示。

当前状态	音频输出	具体操作	操作后显示状态	音频输出
只接HDMI	HDMI输出或用户手动更改	拔出HDMI线	输出为NONE	切换到CODEC
		接入CVBS线	输出为CVBS	CODEC
		接入USB音频输出设备	输出为CVBS	选择后切换到USB
		待机唤醒	输出为CVBS	USB
		关机开机	输出为CVBS	USB
只接CVBS	CODEC输出或用户手动更改	拔出CVBS线	输出为NONE	输出为CODEC
		接入HDMI线	输出为HDMI	添加HDMI
		接入USB音频输出设备	输出为HDMI	选择后切换到USB
		待机唤醒	输出为HDMI	USB
		关机开机	输出为HDMI	USB
同时接入HDMI和CVBS	默认HDMI和CVBS或用户手动更改	拔出CVBS线	输出为HDMI	HDMI和CODEC
		拔出HDMI线	输出为NONE	CODEC
		接入USB音频输出设备	输出为NONE	选择后切换到USB
		待机唤醒	输出为NONE	USB
		关机开机	输出为NONE	USB
无接入任何设备	默认CODEC输出	接入HDMI线	输出为HDMI	添加HDMI
		接入CVBS线	添加CVBS	CODEC
		接入USB音频输出设备	输出CVBS、HDMI	选择后切换到USB
		待机唤醒	输出CVBS、HDMI	USB
		关机开机	输出CVBS、HDMI	USB
USB音频输出	仅USB音频输出	插拔HDMI、CVBS，待机唤醒，关机开机	相应输出	输出为USB
		拔出USB音频设备	不变	切换到当前显示对应音频输出
		手动选择其他音频	不变	切换对应输出
开启透传模式	SPDIF透传	插拔HDMI、CVBS，待机唤醒，关机开机	相应输出	输出为SPDIF
	HDMI透传	插拔HDMI、CVBS，待机唤醒，关机开机	不变	输出为HDMI
		接入USB音频输出设备	不变	选择后切换到USB

图 14: H616 音频输出策略

5.2 音频输入策略

H616 音频输入策略如图所示：

当前设备	音频输入	具体操作	音频输入
CODEC	内置MIC或LINE IN	接入USB音频输入设备	切换到USB音频输入
USB音频设备	USB音频输入	拔出USB音频设备	切换到CODEC音频输入

图 15: H616 音频输入策略

6. TV_BOX 音频特性

TV_BOX 的音频系统具有以下特性：

- 支持单/多路音频输出
- 支持 HDMI USB 音频设备的热插拔
- 支持音频透传

音频策略定制相关代码如下：

1. Audio 设备插拔事件广播管理：/android/vendor/aw/homlet/framework/audio/java/AudioDeviceManagerObserver.java
2. Audio 切换策略：/android/vendor/aw/homlet/framework/audio/java/AudioManagerPolicy.java
3. Audio 输入输出接口扩展：/android/vendor/aw/homlet/framework/audio/java/AudioManagerEx.java
4. Audio 服务启动：/android/frameworks/base/services/java/com/android/server/SystemServer.java
5. Audio 系统设置:/android/packages/apps/TvSettings/Settings/src/com/android/tv/settings/device/sound/AudioChannelsSelect.java

6.1 单/多通路音频输出的使用

打开设置 -> 设备偏好设置 -> 声音 -> 音频输出模式，显示如图所示：

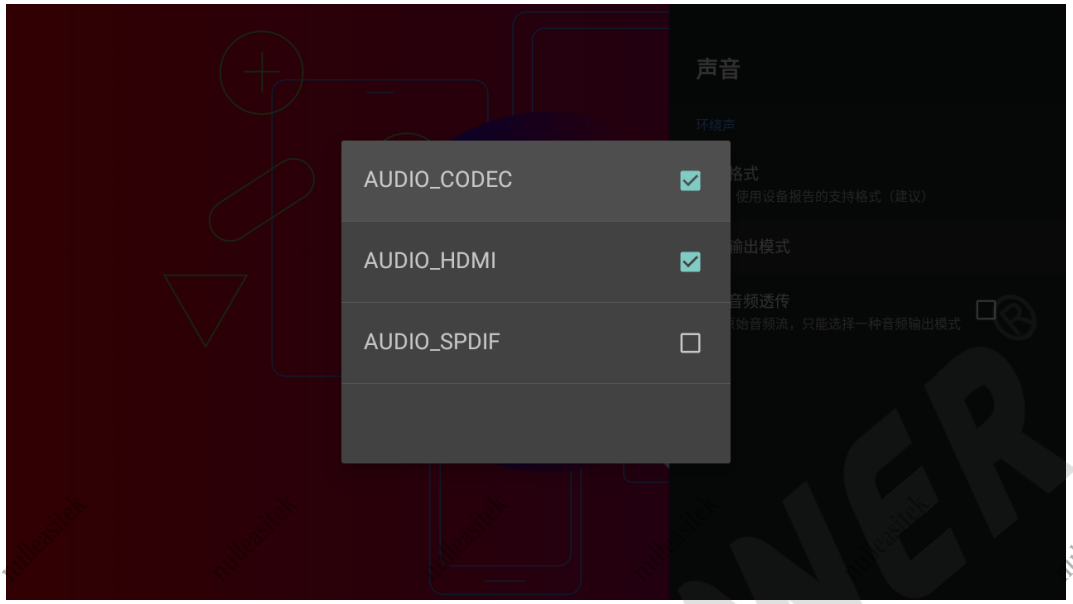

图 16: 音频输出模式

- AUDIO_CODEC 对应 CVBS，电视信号 AV
- AUDIO_HDMI 对应 HDMI，电视信号 HDMI
- AUDIO_SPDIF 对应 SPDIF，光纤接口，外接音箱

若让声音单路输出，只勾选一个选项，例如控制声音只从 HDMI 输出，只勾选 AUDIO_HDMI。将盒子和电视机用 HDMI 线连接后，电视机信号切换到 HDMI，播放音乐或视频就可听到声音。

若让声音多路输出，可勾选多个选项，例如控制声音从 HDMI 和 CVBS 同时输出，则勾选 AUDIO_CODEC 和 AUDIO_HDMI 选项。将盒子和电视机用 HDMI 和 CVBS 线连接后，播放音乐或视频，分别切换电视机信号到 AV 或者 HDMI，都可以听到声音。

在盒子第一次启动时，默认输出是 AUDIO_CODEC 和 AUDIO_HDMI。用户也可根据需求更改默认输出，例如修改为 AUDIO_CODEC,AUDIO_HDMI,AUDIO_SPDIF。

代码目录在 android/device/softwinner/cupid-common/cupid-common.mk

```
PRODUCT_PROPERTY_OVERRIDES += \
vendor.audio.output.active=AUDIO_CODEC,AUDIO_HDMI \
vendor.audio.input.active=AUDIO_CODEC
```

6.2 音频设备热插拔

1. HDMI 线热插拔，将 HDMI 线和 CVBS 线与电视机连接
    - 拔出 HDMI 线，设置 -> 设备偏好设置 -> 声音 -> 音频输出模式 AUDIO_HDMI 未选中，音频数据不可从 HDMI 输出
    - 连接 HDMI 线，设置 -> 设备偏好设置 -> 声音 -> 音频输出模式 AUDIO_HDMI 选中，音频数据可从 HDMI 输出
2. USB 音频输入设备热插拔，将 HDMI 线和 CVBS 线与电视机连接
    - 连接 USB 音频输入设备（一般为 USB 带 MIC 摄像头），音频输入设备切换到此设备
    - 拔出 USB 音频输入设备，音频输入设备切回到 CODEC，默认的音频输入设备是 CODEC
3. USB 音频输出设备热插拔，将 HDMI 线和 CVBS 线与电视机连接
    - 连接 USB 音频输出设备（一般为 USB 音箱），弹出对话框如图所示，显示是否选中 USB 作为输出设备
    - 若选中，音频输出模式仅为 USB 音频输出设备一路，音频数据只可从 USB 音频设备输出
    - 若未选中，音频输出模式不变
    - 断开 USB 音频输出设备，音频输出模式切换到 HDMI

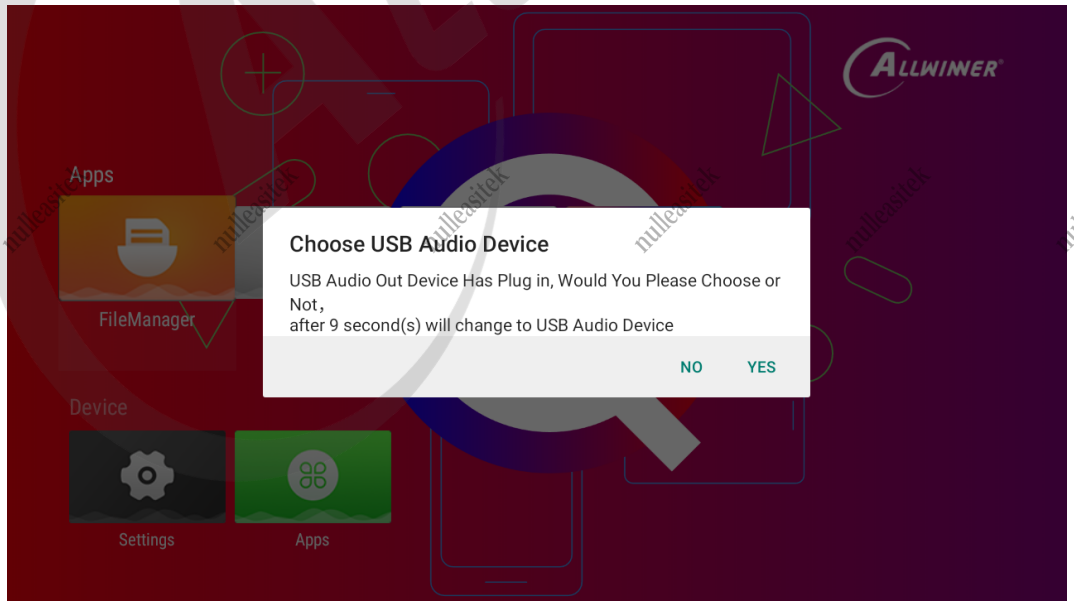

图 17: USB 音频输出设备弹框

6.3 音频透传的使用

打开设置 -> 设备偏好设置 -> 声音 -> 启用音频透传，勾选音频输出模式，如图所示。

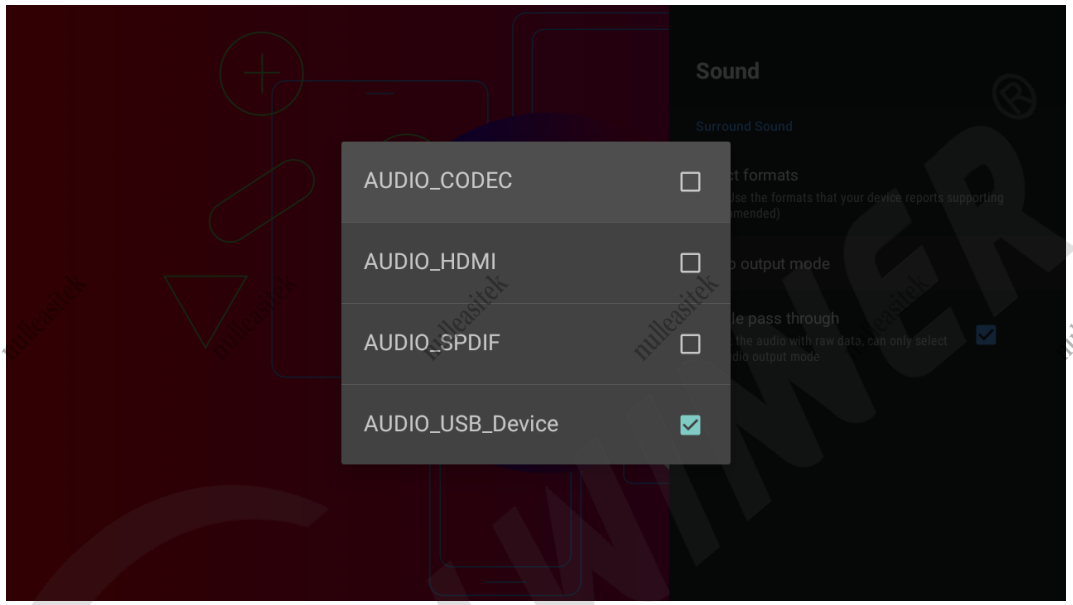

图 18: 音频透传模式

说明：

- 透传模式下，音频输出模式只能勾选一种，即单路输出
- 透传模式下，只有 HDMI 和 SPDIF 才可实现透传
- 透传模式下，选择 CODEC 或 USB，无透传效果，但不影响正常音频输出
- 透传模式下，无法使用遥控器控制音量和静音

6.4 tinyalsa 工具的使用

android/external/tinyalsa 目录下编译，生成以下四个命令，调试音频通路常常使用这四个命令：

- tinypcminfo 用于查看声卡设备参数范围
- tinyplay 用于直接调用 tinyalsa 接口，不走 HAL 层，播放音频

- tinycap 用于直接调用 tinyalsa 接口，不走 HAL 层，录音
- tinymix 用于查看声卡设备节点的通路

生成的四个命令均在/android/out/target/product/版本号/system/bin 下，以 tinymix 举例，安装命令方法：

1. adb remount
2. adb push tinymix /system/bin/
3. adb shell
4. cd system/bin/
5. chmod 777 tinymix

查看设备节点号方法：执行 cat/proc/asound/cards，如图所示，从 0-4 分别是 codec/spdif/ahub/hdmi/i2s0

```
cupid-p2:/ # cat proc/asound/cards
 0 [audiocodec      ]: audiocodec - audiocodec
                       audiocodec
 1 [sndspdif        ]: sndspdif - sndspdif
                       sndspdif
 2 [sndahub         ]: sndahub - sndahub
                       sndahub
 3 [sndhdmi         ]: sndhdmi - sndhdmi
                       sndhdmi
 4 [snddaudio0      ]: snddaudio0 - snddaudio0
                       snddaudio0
```

图 19: H616 音频设备节点

回放录音操作说明：

1. 若操作 spdif、usb、codec（按照 audiocodec 配置 Lineout 通路），直接操作他们设备节点
2. 若操作 hdmi/i2s0/i2s2/i2s3，需要进行以下操作
   - 打开 ahub 设备节点（card num, device APBIF_TXDIF/RX1/2/3)

- 打开 hdmi/i2s0/i2s2/i2s3 设备节点
- 回放连接 hdmi/i2s0/i2s2/i2s3 RX 和 APBIF_TXDIF1/2/3
- 录音连接 APBIF0 RX 和 hdmi/i2s0/i2s2/i2s3 TX
- 向 ahub 设备中写/录数据

6.4.1 tinycap

6.4.1.1 验证 usb mic（usb mic 已连接）

1. 执行 tinycap /sdcard/usb.wav -D usb 设备节点号

6.4.1.2 验证 i2s0 in 功能（i2s0 已挂载）

1. 查看设备节点，i2s0 为 4
2. 执行 tinymix -D 2 13 4 将 i2s0 TX 与 APBIF0 RX（ahub device 0）连接，执行 tinymix -D 2 14 1 将 i2s0 in 打开
3. 修改 tinycap，添加打开 i2s0 设备节点号，打开 i2s0 设备节点 4
4. 执行 tinycap /sdcard/i2s0.wav -D 2 -d 0
5. 示波器测量 i2s0 bclk/mclk/lrclk/data 信号

6.4.2 tinyplay

6.4.2.1 验证 spdif out 功能

执行 tinyplay /sdcard/spdif.wav -D 1

6.4.2.2 验证 usb out 功能（usb out 已连接）

执行 tinyplay /sdcard/usb.wav -D （usb out 设备节点号）

6.4.2.3 验证 codec out 功能

1. 查看设备节点，codec 为 0
2. 执行以下命令打开 codec out 控件（按照 audiocodec 配置 Lineout 通路）tinymix -D 0 4 31 tinymix -D 0 5 1 tinymix -D 0 6 1 tinymix -D 0 11 1
3. 执行 tinyplay /sdcard/codec.wav -D 0

6.4.2.4 验证 i2s0 out 功能（i2s0 已挂载）

1. 查看设备节点，i2s0 为 3
2. 执行 tinymix -D 2 10 3 将 I2S0 RX 与 APBIF_TXDIF2（ahub device 2）连接，执行 tinymix -D 2 15 1 将 i2s0 out 打开
3. 修改 tinyplay 代码，打开 i2s0 设备节点（3）
4. 执行 tinyplay /sdcard/i2s0.wav -D 2 -d 2
5. 示波器测量 i2s0 mclk/bclk/lrclk/data 是否有信号

6.4.2.5 验证 hdmi out 功能

参照验证 i2s0 out 功能

7. FAQ

1. 透传播放 DTS 格式的音频文件，无声音输出

   - 排查问题

     1. 查看设置 -> 声音 -> 启动透传有没有被勾选
     2. 查看设置 -> 声音 -> 音频输出模式有没有勾选对应的音频输出
     3. 如仍有问题请与 FAE 联系

2. 播放音视频，无声音输出

   - 排查问题

     1. 查看设置 -> 声音 -> 音频输出模式有没有勾选对应的音频输出
     2. 查看设备设备节点是否存在，执行命令 cat /proc/asound/cards，执行结果如下所示

```
console:/ # cat proc/asound/cards
 0 [sndspdif ]: sndspdif - sndspdif
                           sndspdif
 1 [sndahub ]: sndahub - sndahub
                           sndahub
 2 [sndhdmi ]: sndhdmi - sndhdmi
                           sndhdmi
 3 [snddaudio2 ]: snddaudio2 - snddaudio2
                           snddaudio2
 4 [sndacx00codec ]: sndacx00-codec - sndacx00-codec
                           sndacx00-codec
```

3. 使用 CODEC 录音，回放有电流声

   - 排查问题

     1. 不接入麦克风，使用 tinycap 录音
     2. 使用音频软件 Audition 查看波形，若波形有小毛刺，说明是底噪

4. 用户 apk 播放音频，有抖音

   - 排查问题

     1. 检查 apk 是否没有按照获取到的 buffer 大小向下送音频数据

5. 蓝牙语音通话无声音或声音错误

- 排查问题
  1. 查看 daudio2 设备节点是否存在
  2. 查看 menuconfig 配置，查看 daudio 是否已经配置成 y
  3. 查看 board.dts 配置，查看 daudio2 是否配置正确

8. Declaration

This document is the original work and copyrighted property of Allwinner Technology (“Allwinner”). Reproduction in whole or in part must obtain the written approval of Allwinner and give clear acknowledgement to the copyright owner. The information furnished by Allwinner is believed to be accurate and reliable. Allwinner reserves the right to make changes in circuit design and/or specifications at any time without notice. Allwinner does not assume any responsibility and liability for its use. Nor for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of Allwinner. This datasheet neither states nor implies warranty of any kind, including fitness for any particular application.tates nor implies warranty of any kind, including fitness for any particular application.