



D1 Linux I2S 开发指南

版本号: 1.0
发布日期: 2021.04.14

版本历史

版本号	日期	制/修订人	内容描述
1.0	2021.04.14	AWA1692	1. 添加 1.0 版 I2S/PCM 音频模块使用说明文档



目 录

1 前言	1
1.1 文档简介	1
1.2 目标读者	1
1.3 适用范围	1
1.4 相关术语	1
2 模块介绍	2
2.1 模块功能规格介绍	2
2.2 模块源码结构介绍	2
2.3 模块配置介绍	3
2.3.1 Device Tree 配置介绍	3
2.3.2 board.dts 板级配置介绍	4
3 模块使能说明	9
3.1 board.dts 模块使能	9
3.2 kernel menuconfig 使能	9
4 模块功能使用说明	15
4.1 模块声卡/设备查看说明	15
4.2 模块音频控件及通路配置说明	16
4.2.1 音频控件说明	16
4.2.2 模块音频通路配置说明	16
4.3 模块功能验证说明	16
4.3.1 loopback 回环测试功能使用说明	16
4.3.2 同源输出功能使用说明	18
5 外挂 Codec 使用说明	20
5.1 AW SUNXI 平台 I2S/PCM 外挂总览	20
5.2 外挂 Codec 步骤	20
5.2.1 硬件相关准备	20
5.2.2 软件相关准备	21
5.2.3 上板验证	21
6 FAQ	23

插 图

3-1 Device Driver	10
3-2 Sound	10
3-3 Advanced	11
3-4 ALSA	11
3-5 Allwinner	12
3-6 module	12
3-7 HDMI AUDIO	13
3-8 codecs	13
3-9 AC108	14
4-1 loopback	17



1 前言

1.1 文档简介

本文档编写目的是为了音频系统相关的开发者能够了解清楚 AW SUNXI 平台下 I2S/PCM 接口的具体使用方法，能够更快地基于 AW SUNXI 平台完成对 I2S/PCM 接口的使用及二次开发等。

1.2 目标读者

音频系统相关开发人员。

1.3 适用范围

表 1-1: 适用产品列表

产品名称	内核版本	驱动文件
D1	Linux-5.4	sound/soc/sunxi/*

1.4 相关术语

- I2S/PCM: I2S(Inter-IC Sound) 总线, 又称集成电路内置音频总线, 是飞利浦公司为数字音频设备之间的音频数据传输而制定的一种总线标准;
- 内部回环 loopback 测试: AW SUNXI 平台 IC 内部的一种 TX 与 RX 内联的一种播录回环测试方法;
- Daudio: 数字音频接口, 可配置成 i2s/pcm 格式标准音频接口;
- AC108: 一款音频编解码芯片, 集成具有 108 dB 动态范围的四通道 ADC (A-weighted) @0dB boost gain 和 1 个 I2S 接口带 2 路数据输出;
- TinyALSA: tiny + ALSA 即微型 ALSA 库, 用于实现用户空间与内核空间的交互, 能够实现播放/录音等基本功能, 常用于 AW SUNXI 平台驱动层音频模块功能验证;

2 模块介绍

对 AW SUNXI 平台的 I2S/PCM 接口模块的基础介绍

2.1 模块功能规格介绍

AW SUNXI 平台 I2S/PCM 接口模块功能及规格:

- 支持 playback 播放功能;
- 支持 record 录音功能;
- 支持内部回环 loopback 测试;
- 支持同源输出功能;
- 支持主 (master) /从 (slave) 模式;
- 支持标准 I2S 模式 (standard I2S) /左对齐模式 (left-Justified) /右对齐模式 (Right-Justified) /PCM 模式/TDM 模式;
- 播放支持多种采样率格式 8KHz~384KHz;
- 录音支持多种采样率格式 8KHz~384KHz;
- 最高可支持至 16 通道;
- 支持 8bit/16bit/20bit/24bit/32bit 数据精度;
- 模块时钟最高可支持至 24.576MHz;
- 可用于 HDMI audio 播放 PCM 音频

2.2 模块源码结构介绍

模块驱动的源代码位于内核的/sound/soc/sunxi/目录下:

```
/tina/lichee/linux-5.4/sound/soc/  
├─ sunxi // AW Sunxi平台  
│   └─ sunxi-daudio.c // Sunxi平台Daudio接口代码  
│   └─ sunxi-daudio.h // Sunxi平台Daudio驱动头文件  
│   └─ sunxi-pcm.c // Sunxi平台platform部分dma代码  
│   └─ sunxi-pcm.h // Sunxi平台platform部分头文件  
│   └─ sunxi-simple-card.c // Sunxi平台machine部分代码  
└─ codecs // 解码器存放路径  
    └─ ac108.c // AC108解码器codec驱动, 用于外挂
```

2.3 模块配置介绍

2.3.1 Device Tree 配置介绍

对应内核设备树中存在着每款芯片的所有平台的 DMIC 模块配置，而 AW SUNXI 平台的设备树配置文件的路径为：

```
/tina/lichee/linux-5.4/arch/arm64/boot/dts/sunxi/CHIP.dtsi (64bit平台)
/tina/lichee/linux-5.4/arch/arm/boot/dts/CHIP.dtsi (32bit平台)
/tina/lichee/linux-5.4/arch/riscv/boot/dts/sunxi/CHIP.dtsi (riscv平台)
```

其中 CHIP 为研发代号，如 D1 的研发代号为 sun20iw1p1 等。

举例 D1 的设备树模块配置如下所示：（以 I2S2 为例）

(/tina/lichee/linux-5.4/arch/riscv/boot/dts/sunxi/sun20iw1p1.dtsi)

```
daudio2:daudio@2034000 {
    #sound-dai-cells = <0>;
    compatible = "allwinner,sunxi-dauidio";
    reg = <0x0 0x02034000 0x0 0xa0>;
    clocks = <&ccu CLK_PLL_AUDI00>,
            <&ccu CLK_I2S2>,
            <&ccu CLK_BUS_I2S2>,
            <&ccu CLK_PLL_AUDI00_4X>,
            <&ccu CLK_I2S2_ASRC>;
    resets = <&ccu RST_BUS_I2S2>;
    dmas    = <&dma 5>, <&dma 5>;
    dma-names = "tx", "rx";
    interrupts-extended = <&plic0 44 IRQ_TYPE_LEVEL_HIGH>;
    sign_extend = <0x00>;
    tx_data_mode = <0x00>;
    rx_data_mode = <0x00>;
    msb_lsb_first = <0x00>;
    dauidio_rxsync_en = <0x00>;
    pcm_lrck_period = <0x80>;
    slot_width_select = <0x20>;
    frametype = <0x00>;
    tdm_config = <0x01>;
    tdm_num = <0x02>;
    mclk_div = <0x01>;
    clk_parent = <0x01>;
    capture_cma = <128>;
    playback_cma = <128>;
    tx_num = <4>;
    tx_chmap1 = <0x76543210>;
    tx_chmap0 = <0xFEDCBA98>;
    rx_num = <4>;
    rx_chmap3 = <0x03020100>;
    rx_chmap2 = <0x07060504>;
    rx_chmap1 = <0x0B0A0908>;
    rx_chmap0 = <0x0F0E0D0C>;
    asrc_function_en = <0x00>;
    device_type = "dauidio2";
    status = "disabled";
}
```

```

};

sounddaudio2: sounddaudio2@20340a0 {
    reg = <0x0 0x020340a0 0x0 0x4>;
    compatible = "sunxi,simple-audio-card";
    simple-audio-card,name = "snddaudio2";
    simple-audio-card,format = "i2s";
    status = "disabled";
    simple-audio-card,cpu {
        sound-dai = <&daudio2>;
    };
};

hdmaudio: hdmaudio@20340a4 {
    #sound-dai-cells = <0>;
    reg = <0x0 0x020340a4 0x0 0x4>;
    compatible = "allwinner,sunxi-hdmaudio";
    status = "disabled";
};

```

其中, 各项配置参数及其说明如下所示:

表 2-1: 模块 DTS 节点配置说明

节点配置	解释说明
reg	模块在 IC 中的模块基址及其最大偏移地址
clock	模块使用的时钟, 一般分别为时钟源及模块时钟
status	模块使能/关闭开关, "okay"使能, "disabled"关闭
其他	模块声卡注册、绑定等相关, 一般不应改动

2.3.2 board.dts 板级配置介绍

board.dts 用于保存每一个板级平台的设备信息 (如 demo 板, perf1 板, ver 板等等), 里面的同名配置信息会覆盖上面的 DTS 设备树默认配置信息。

board.dts 板级配置文件路径为:

```
/tina/device/config/chips/IC/configs/BOARD/board.dts
```

举例 D1 的 board.dts 板级配置文件模块配置如下所示:

```
(/longon/device/config/chips/d1/configs/nezha/board.dts)
```

```

daudio2_pins_a: daudio2@0 {
    /* I2S_PIN: MCLK, BCLK, LRCK */
    pins = "PB7", "PB5", "PB6";
    function = "i2s2";
    drive-strength = <20>;
    bias-disable;
};

```

```
daudio2_pins_b: daudio2@1 {
    /* I2S_PIN: DOUT0 */
    pins = "PB4";
    function = "i2s2_dout";
    drive-strength = <20>;
    bias-disable;
};

daudio2_pins_c: daudio2@2 {
    /* I2S_PIN: DIN0 */
    pins = "PB3";
    function = "i2s2_din";
    drive-strength = <20>;
    bias-disable;
};

daudio2_pins_d: daudio2_sleep@0 {
    pins = "PB7", "PB5", "PB6", "PB4", "PB3";
    function = "io_disabled";
    drive-strength = <20>;
    bias-disable;
};

/* 外挂 AC108 所使用的 TWI 接口 */
&twi0 {
    clock-frequency = <400000>;
    pinctrl-0 = <&twi0_pins_a>;
    pinctrl-1 = <&twi0_pins_b>;
    pinctrl-names = "default", "sleep";
    twi_drv_used = <1>;
    dmas = <&dma 43>, <&dma 43>;
    dma-names = "tx", "rx";
    status = "disabled";
    ac108: ac108@3B {
        #sound-dai-cells = <0>;
        compatible = "Allwinner,MicArray_0";
        device_type = "MicArray_0";
        reg = <0x3B>;
        regulator_used = <0x0>;
        power_voltage = <3300000>;
        regulator_name = "vcc-3v3";
        power_gpio_used = <0x0>;
        reset_gpio_used = <0x0>;
        twi_bus = <0x1>;
        pga_gain = <0x1F>;
        slot_width = <0x20>;
        lrck_period = <0x80>;
        ref_pga_used = <0x1>;
        ref_pga_gain = <0x10>;
        ref_channel = <0x3>;
        debug_mode = <0x0>;
        status = "disabled";
    };
};

/*-----
* pcm_lrck_period      16/32/64/128/256
*                      (set 0x20 for HDMI audio out)
* slot_width_select    16bits/20bits/24bits/32bits
```

```

*          (set 0x20 for HDMI audio out)
* frametype      0 --> short frame = 1 clock width;
*               1 --> long frame = 2 clock width;
* tdm_config     0 --> pcm
*               1 --> i2s
*          (set 0x01 for HDMI audio out)
* mclk_div       0 --> not output
*               1/2/4/6/8/12/16/24/32/48/64/96/128/176/192
*          (set mclk as external codec clk source, freq is pll_audio/mclk_div)
* pinctrl_used   0 --> I2S/PCM use for internal (e.g. HDMI)
*               1 --> I2S/PCM use for external audio
* daudio_type:   0 --> external audio type
*               1 --> HDMI audio type
*-----*/
&daudio2 {
    mclk_div      = <0x00>;
    frametype     = <0x00>;
    tdm_config    = <0x01>;
    sign_extend   = <0x00>;
    tx_data_mode  = <0x00>;
    rx_data_mode  = <0x00>;
    msb_lsb_first = <0x00>;
    pcm_lrck_period = <0x20>;
    slot_width_select = <0x20>;
    asrc_function_en = <0x00>;
    pinctrl-names = "default", "sleep";
    pinctrl-0     = <&daudio2_pins_a &daudio2_pins_b &daudio2_pins_c>;
    pinctrl-1     = <&daudio2_pins_d>;
    pinctrl_used  = <0x0>;
    daudio_type   = <0x1>;
    status = "okay";
};

/* if HDMI audio is used, daudio2 should be enable. */
&hdmaudio {
    status = "disable";
};

/*-----*/
* simple-audio-card,name      name of sound card, e.g.
*                             "snddaudio0" --> use for external audio
*                             "sndhdmi" --> use for HDMI audio
* sound-dai                   "snd-soc-dummy" --> use for I2S
*                             "hdmaudio" --> use for HDMI audio
*                             "ac108" --> use for external audio of ac108
* simple-audio-card,format    "i2s" --> 标准模式
*                             "right_j" --> 右对齐模式
*                             "left_j" --> 左对齐模式
*                             "dsp_a" --> pcm 短帧模式
*                             "dsp_b" --> pcm 长帧模式
*-----*/
&sounddaudio2 {
    status = "okay";
    simple-audio-card,name = "sndhdmi";
    daudio2_master: simple-audio-card,codec {
        /* sound-dai = <&ac108>; */
    };
};

```

其中, 各项配置参数及其说明如下所示:

表 2-2: 模块 board.dts 板级配置文件引脚配置说明

节点配置	解释说明
pins	模块需要使用到的引脚组定义
function	模块引脚组复用功能
drive-strength	模块引脚驱动力，默认配置为 20 即可
bias-disable	失能上下拉

表 2-3: 模块 board.dts 板级配置文件配置说明

节点配置	配置可选值	解释说明
mclk_div	0~192	MCLK 分频系数选择，一般来说 $MCLK = PLL_AUDIO / MCLK_DIV$ ，而 PLL_AUDIO 一般为 22.5792MHz/24.576MHz，其中配置为 0 则表示 MCLK 不做输出；
frametype	0/1	长短帧选择，即仅用于 PCM 模式中一个 LRCK=1（配置为 0 时）/2（配置为 1 时）个 BCLK 的宽度，其中配置 0 为短帧，配置 1 则为长帧；
tdm_config	0/1	传输模式选择，其中配置 0 为 PCM 模式，配置 1 为 I2S 模式；
sign_extend	0/1	数据扩展位补充选择，配置为 0 则表示填充数据是 0，配置为 1 则表示填充数据是数据最后一位，一般默认为 0 即可；
tx_data_mode	0/1/2/3	pcm 传输模式中 TX 端选择数据的格式，其中，0: 16bit linear PCM; 1: 8bit linear PCM; 2: 8bit u-law; 3: 8bit a-law，一般默认配置为 0 即可；
rx_data_mode	0/1/2/3	pcm 传输模式中 RX 端选择数据的格式，其中，0: 16bit linear PCM; 1: 8bit linear PCM; 2: 8bit u-law; 3: 8bit a-law，一般默认配置为 0 即可；
msb_lsb_first	0/1	数据传输有效位选择，其中，0: msb first（数据低位开始）；1: lsb first（数据高位开始），一般默认配置为 0 即可；

节点配置	配置可选值	解释说明
pcm_lrck_period	16/32/64/128/256	表示一个 LRCK 中有多少个 BCLK, pcm_lrck_period 的计算与配置与传输规格有一定的换算关系即 $\text{pcm_lrck_period} = \text{Max sample_channels} * \text{Max sample_bits};$
slot_width_select	16/20/24/32	数据位宽选择, 必须大于或等于采样精度, 一般等与最高支持采样精度即 $\text{slot_width_select} = \text{Max sample_bits},$ 其中配置分别对应 16bit/20bit/24bit/32bit;
sunxi,snddaudio-codec	根据绑定节点而定	外挂 Codec 的 codec name, 用于与 I2S 完成绑定
sunxi,snddaudio-codec-dai	根据绑定节点而定	外挂 Codec 的 codec_dai name, 用于与 I2S 完成绑定
simple-audio-card,format	"i2s"、"right_j"、"left_j"、"dsp_a"、"dsp_b"	主控端 I2S 接口主从模式选择, 其中, "i2s": 标准模式、"right_j": 右对齐模式、"left_j": 左对齐模式、"dsp_a": pcm 短帧模式、"dsp_b": pcm 长帧模式
simple-audio-card,frame-master	根据绑定节点而定	表示所绑定的节点为 lrck 时钟为主机
simple-audio-card,bitclock-master	根据绑定节点而定	表示所绑定的节点为 bclk 时钟为主机
simple-audio-card,frame-inversion	/	若定义, 则 lrck 时钟反转
simple-audio-card,bitclock-inversion		若定义, 则 bclk 时钟反转
status	"okay"/"disabled"	模块使能/关闭开关, 其中, "okay"使能, "disabled"关闭;

3 模块使能说明

详细介绍模块使能的步骤

3.1 board.dts 模块使能

在相应的板级配置文件 (board.dts) 下，选择将 daudio 节点及 snddaudio 节点下的 “status” 修改为 “okay” 并保存退出即可，具体修改示例如下所示：

```
daudio2:audio@2034000 {
    status = "okay";
};

snddaudio2:sound@20340a0 {
    status = "okay";
};
```

3.2 kernel menuconfig 使能

除了上述模块使能操作外，还需注意的是，需要保证内核配置的模块使能也已选中使能，具体操作步骤如下所示：

- 1、在 /tina/ 目录下执行 “make kernel_menuconfig” 命令进入内核配置界面。(需先选择对应平台)
- 2、选择 Device Drivers 选项进入下一级配置，如下图所示：

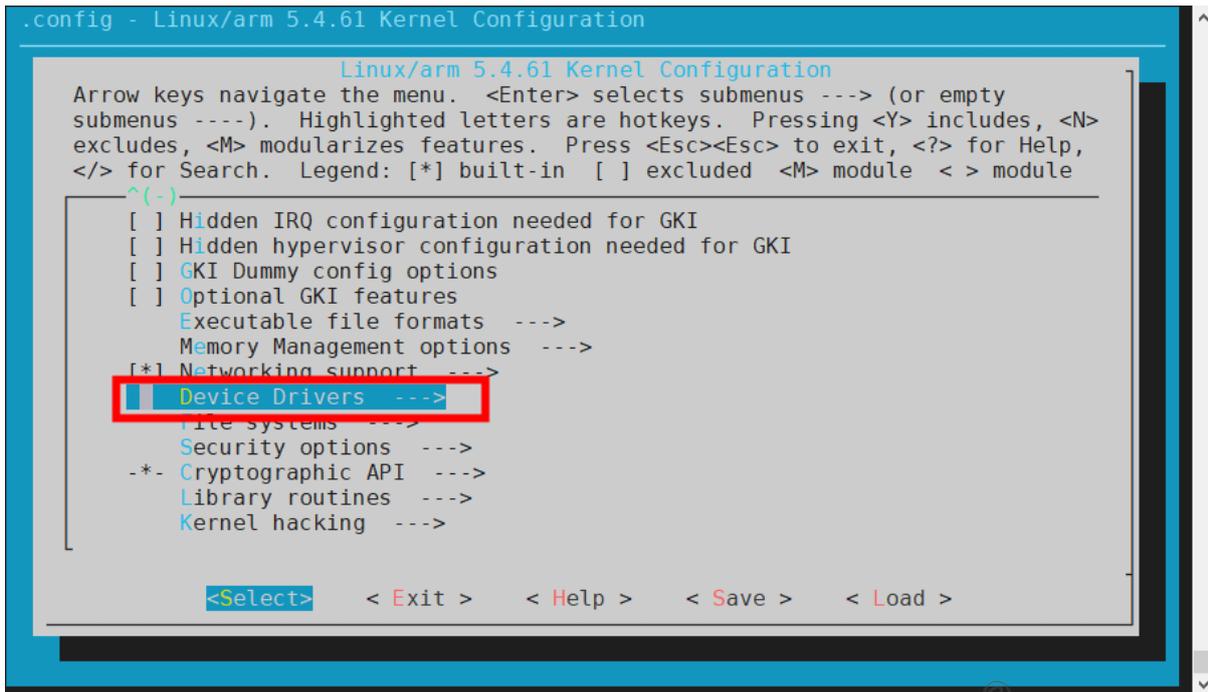


图 3-1: Device Driver

3、选择 Sound card support 选项，进入下一级配置，如下图所示：

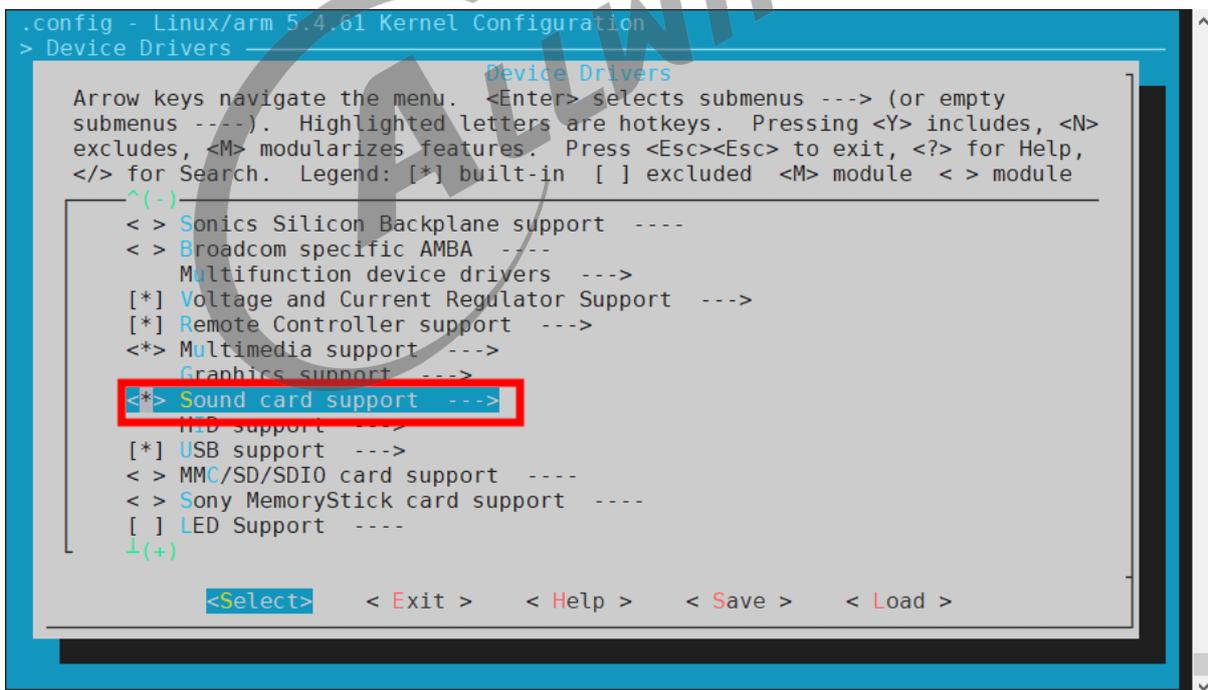


图 3-2: Sound

4、选择 ALSA 框架，即 Advanced Linux Sound Architecture 选项，如下图所示：

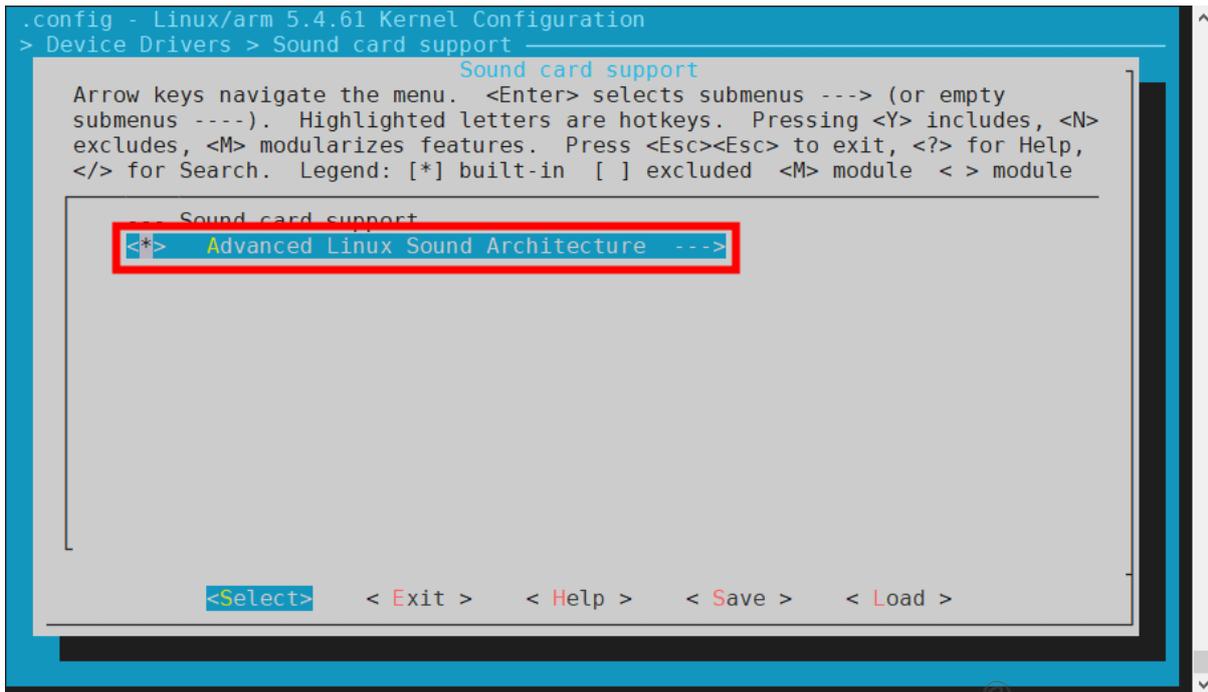


图 3-3: Advanced

5、选择 ALSA for SoC audio support 选项，进入下一级配置，如下图所示：

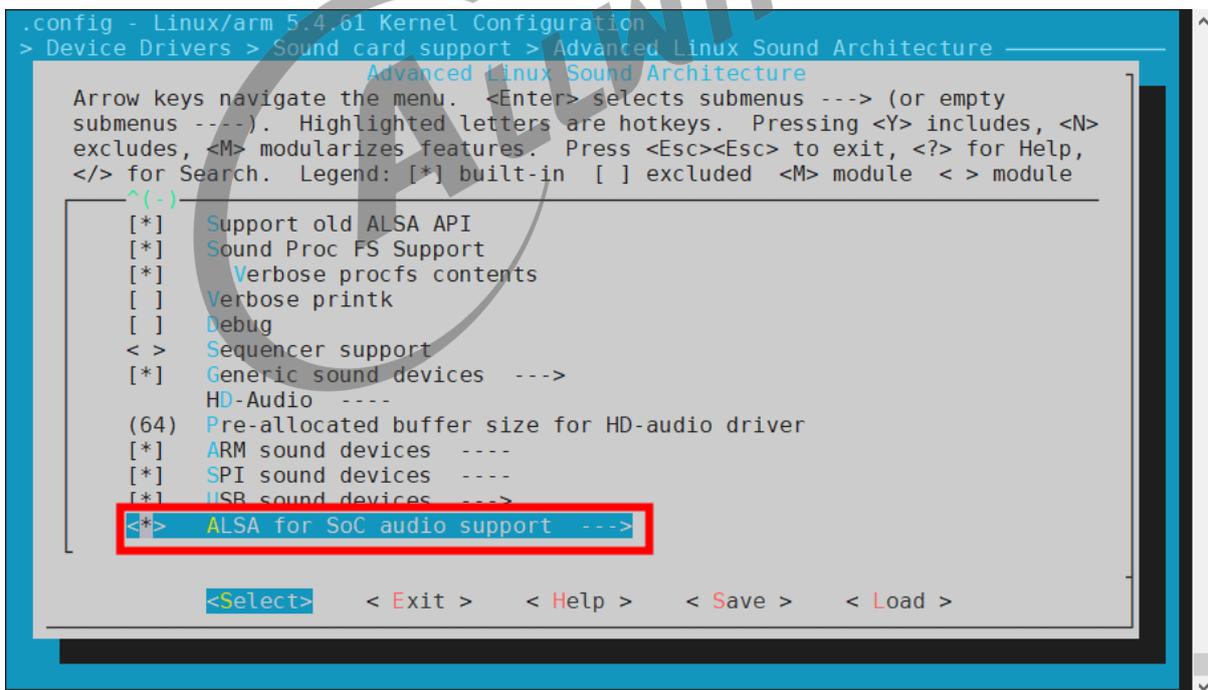


图 3-4: ALSA

6、选择 Allwinner SoC Audio support 选项，如下图所示：

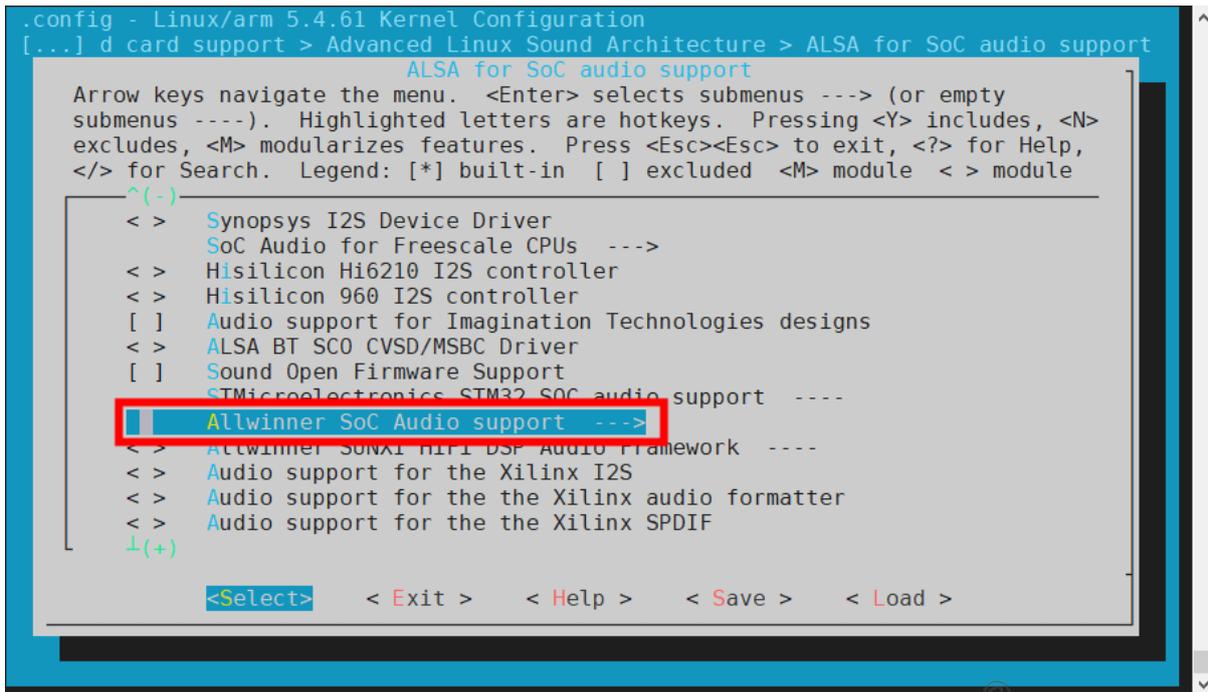


图 3-5: Allwinner

7、选择需要的模块，可选择直接编译进内核，也可编译成模块。如下图所示：

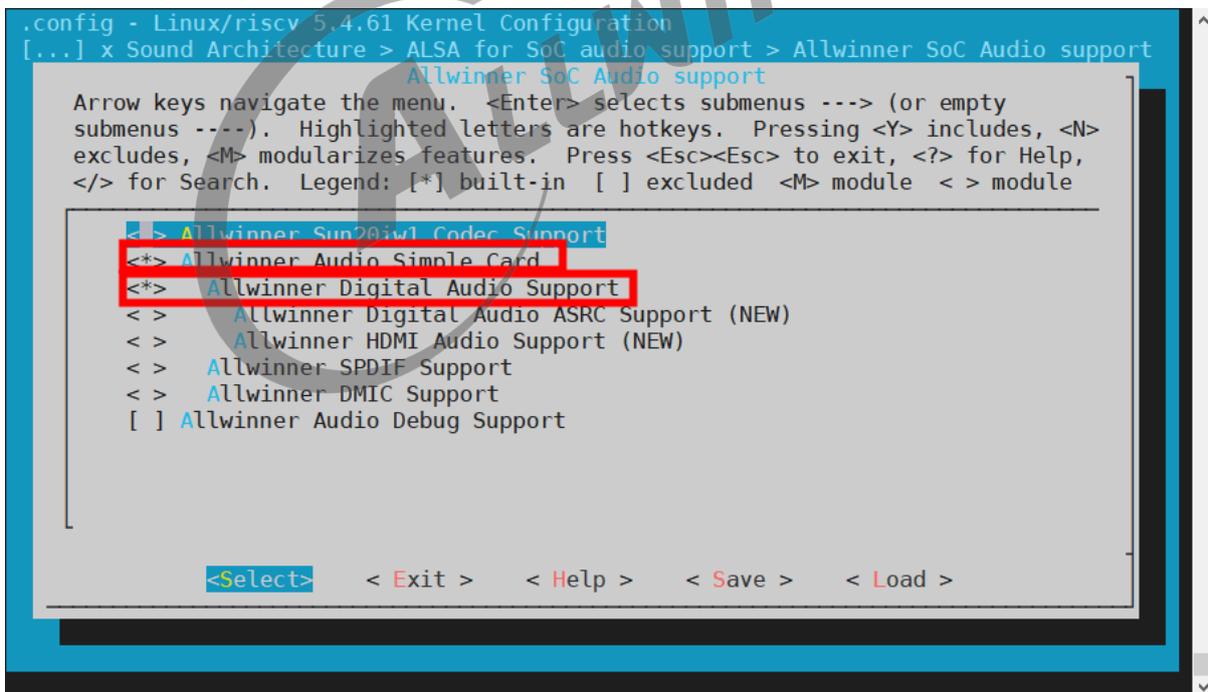


图 3-6: module

8、若选择 HDMI audio 播放功能，可选择直接编译进内核，也可编译成模块。如下图所示：

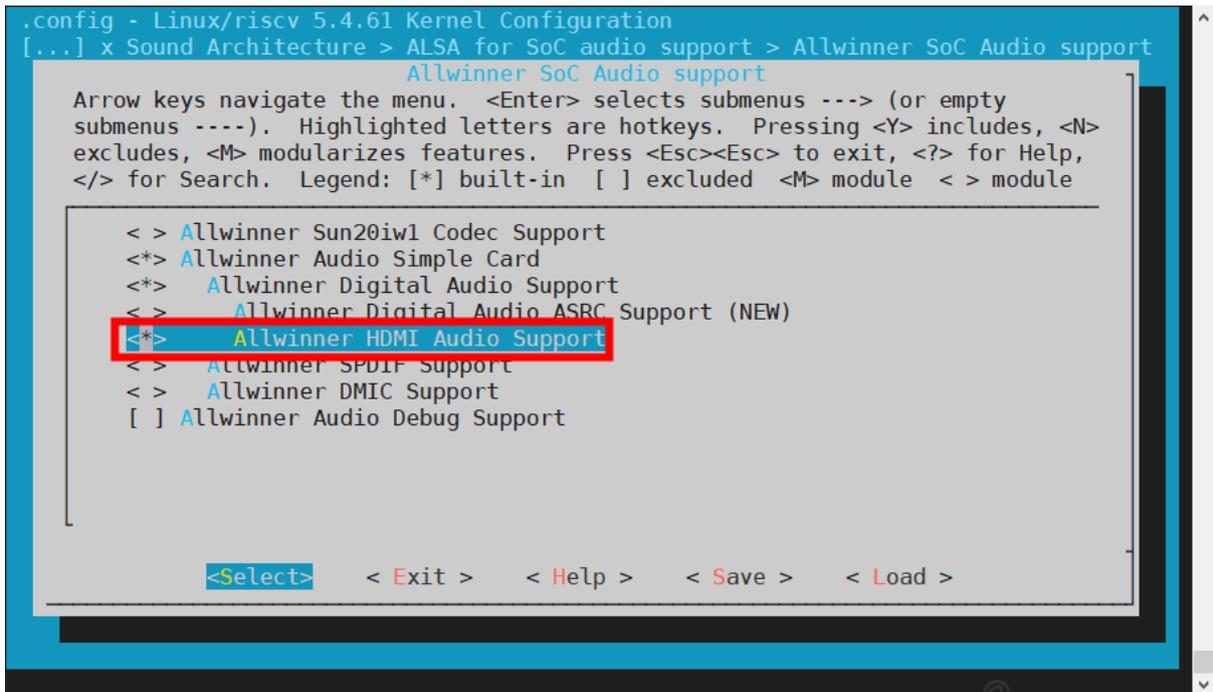


图 3-7: HDMI AUDIO

9、若选择外挂 Codec 模块，可选择直接编译进内核，也可编译成模块。如下图所示：（以 AC108 为例）

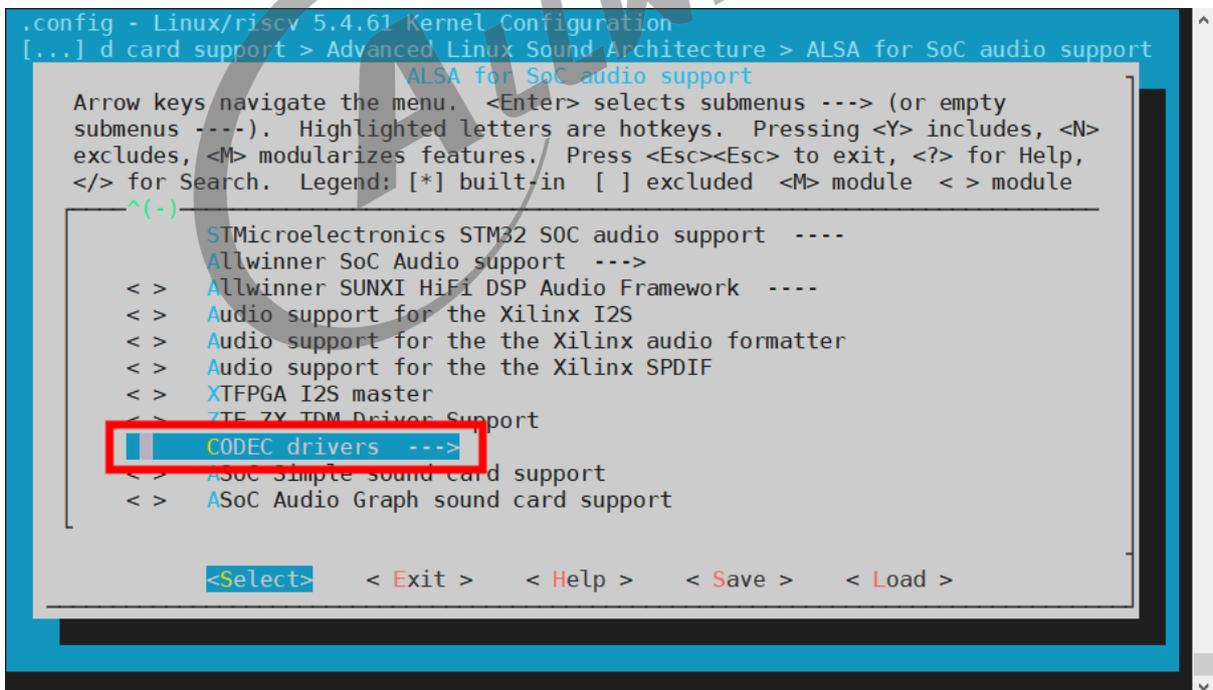


图 3-8: codecs

```
.config - Linux/riscv 5.4.61 Kernel Configuration
[...] Advanced Linux Sound Architecture > ALSA for SoC audio support > CODEC drivers
CODEC drivers
Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty
submenus ----). Highlighted letters are hotkeys. Pressing <Y> includes, <N>
excludes, <M> modularizes features. Press <Esc><Esc> to exit, <?> for Help,
</> for Search. Legend: [*] built-in [ ] excluded <M> module <> module
^(-)
<> ZTE ZX AUD96P22 CODEC
<> Maxim MAX9759 speaker Amplifier
<> MediaTek MT6351 Codec
<> MediaTek MT6358 Codec
<> Nuvoton Technology Corporation NAU85L40 CODEC
<> Nuvoton Technology Corporation NAU88C10 CODEC
<> Nuvoton Technology Corporation NAU88C22 CODEC
<> Nuvoton Technology Corporation NAU88L24 CODEC
<> Texas Instruments TPA6130A2 headphone amplifier
<> Sunxi AC107 Codec
<> Sunxi AC108 Codec
<Select> < Exit > < Help > < Save > < Load >
```

图 3-9: AC108

综上，即可完成该模块的使能配置，重新编译烧录固件即可生成相应的模块声卡及设备（所使用的引脚与其它模块无冲突）。

4 模块功能使用说明

详细介绍模块接口的使用方法

4.1 模块声卡/设备查看说明

外挂 codec (AC108 为例)、注册为 HDMI 声卡方法, 均在 board.dts 板级配置介绍中说明, 以下为虚拟 I2S 声卡具体介绍

当相应的模块使能都打开并编译、烧录固件成功起来后, I2S 模块在没有实际外挂注册绑定 Codec 时, 正常会生成相应的虚拟声卡 snddaudio 及设备, 而具体 I2S2 虚拟声卡查看及确认操作示例如下所示:

```
/ # cat /proc/asound/cards
0 [snddaudio2      ]: snddaudio2 - snddaudio2      // I2S2接口注册的虚拟声卡
                        snddaudio2

/ #
/ # ls -l /proc/asound/snddaudio2/
total 0
-r--r--r--    1 root    root          0 Jan  1 00:48 id      // I2S2模块虚拟声卡ID名称
dr-xr-xr-x    3 root    root          0 Jan  1 00:48 pcm0c  // I2S2模块虚拟声卡录音设备
dr-xr-xr-x    3 root    root          0 Jan  1 00:48 pcm0p  // I2S2模块虚拟声卡播放设备
```

查看模块音频流的相关设置参数操作示例如下:

```
#查看播放参数 (需在播放过程中查看)
/ # tinyplay tmp/test.wav &
/ # cat /proc/asound/snddaudio2/pcm0p/sub0/hw_params
access: RW_INTERLEAVED
format: S16_LE           // 采样精度: 16bit
subformat: STD
channels: 2              // 通道数: 2 channels
rate: 44100 (44100/1)   // 采样率: 44.1KHz
period_size: 1024
buffer_size: 4096
/ # killall tinyplay

#查看录音参数 (需在录音过程中查看)
/ # tinycap tmp/test.wav -c 2 &
/ # cat /proc/asound/snddaudio2/pcm0c/sub0/hw_params
access: RW_INTERLEAVED
format: S16_LE           // 采样精度: 16bit
subformat: STD
channels: 2              // 通道数: 2 channels
rate: 44100 (44100/1)   // 采样率: 44.1KHz
period_size: 1024
buffer_size: 4096
```

```
/ # killall tinycap
```

4.2 模块音频控件及通路配置说明

本章说明将会基于 TinyALSA 工具的使用上进行说明

查看 daudio 模块声卡音频控件列表及音频路由：

```
/ # cat /proc/asound/cards
0 [snddaudio2      ]: snddaudio2 - snddaudio2
                        snddaudio2

/ #
/ # tinymix -D 0
Mixer name: 'snddaudio2'
Number of controls: 2
ctl      type      num      name                               value
0        ENUM      1        sunxi daudio audio hub mode       Disable
1        BOOL      1        sunxi daudio loopback debug       Off
```

4.2.1 音频控件说明

模块音频控件使用说明如下所示：

控件序号	控件名称	配置可选值	控件说明
0	sunxi daudio audio hub mode	0 (关闭) /1 (使能)	模块同源输出功能开关
1	sunxi daudio loopback debug	0 (Off) /1 (On)	模块 loopback 回环测试使能开关

4.2.2 模块音频通路配置说明

由于 I2S 模块声卡在未实际外挂绑定 Codec 前，注册的都是虚拟声卡，通路上也是输入输出直通型，故目前不需要配置相关音频通路，只有实际外挂 Codec 后再根据外挂 Codec 的音频通路配置外挂 Codec 的输入输出通路即可。

4.3 模块功能验证说明

4.3.1 loopback 回环测试功能使用说明

loopback 回环实际是 IC 内部 TXFIFO 与 RXFIFO 之间的直通测试通路，不需要任何外部引脚的干涉，loopback 使能开关后，往 I2S TX 播放写数据，就能直接从 I2S RX 录音读回数据，常

可用于一些数据回录的功能场景中，而其具体操作验证示例如下所示：

```

/ # cat /proc/asound/cards
0 [snddaudio2      ]: snddaudio2 - snddaudio2
                          snddaudio2

/ #
/ # tinymix -D 0 1 1          // 使能loopback回环测试开关
/ #
/ # tinymix
Mixer name: 'snddaudio0'
Number of controls: 2
ctl      type      num      name                               value
0        ENUM      1        sunxi daudio audio hub mode       hub_disable
1        B00L      1        sunxi daudio loopback debug       0n

/ #
/ # tinyplay /tmp/play_test.wav -D 0 -d 0 &          // 后台用snddaudio2声卡播放音频
/ # [ 2239.622263] snddaudio snddaudio2: codec_dai set sysclk failed
[ 2239.628779] snddaudio snddaudio2: codec_dai set set_pll failed.
[ 2239.635475] snddaudio snddaudio2: codec dai set fmt failed
[ 2239.641679] snddaudio snddaudio2: codec_dai set clkdiv failed
Playing sample: 2 ch, 44100 hz, 16 bit          // 开始播放

/ #
/ # tinycap /tmp/cap_test.wav -D 0 -d 0          // 用snddaudio2声卡录制默认同样格式的音频文件
[ 2253.030564] snddaudio snddaudio2: codec_dai set sysclk failed
[ 2253.037064] snddaudio snddaudio2: codec_dai set set_pll failed.
[ 2253.043752] snddaudio snddaudio2: codec dai set fmt failed
[ 2253.049985] snddaudio snddaudio2: codec_dai set clkdiv failed
Capturing sample: 2 ch, 44100 hz, 16 bit          // 开始回录

^CCaptured 987136 frames          // Ctrl + C退出程序结束回录
/ # killall tinyplay

```

回录音频可以通过 adb 工具（adb pull）将刚回录的音频文件拉出来用音频解析软件（Audition/ocenaudio）播放查看，具体回录音频文件解析实例如下所示：

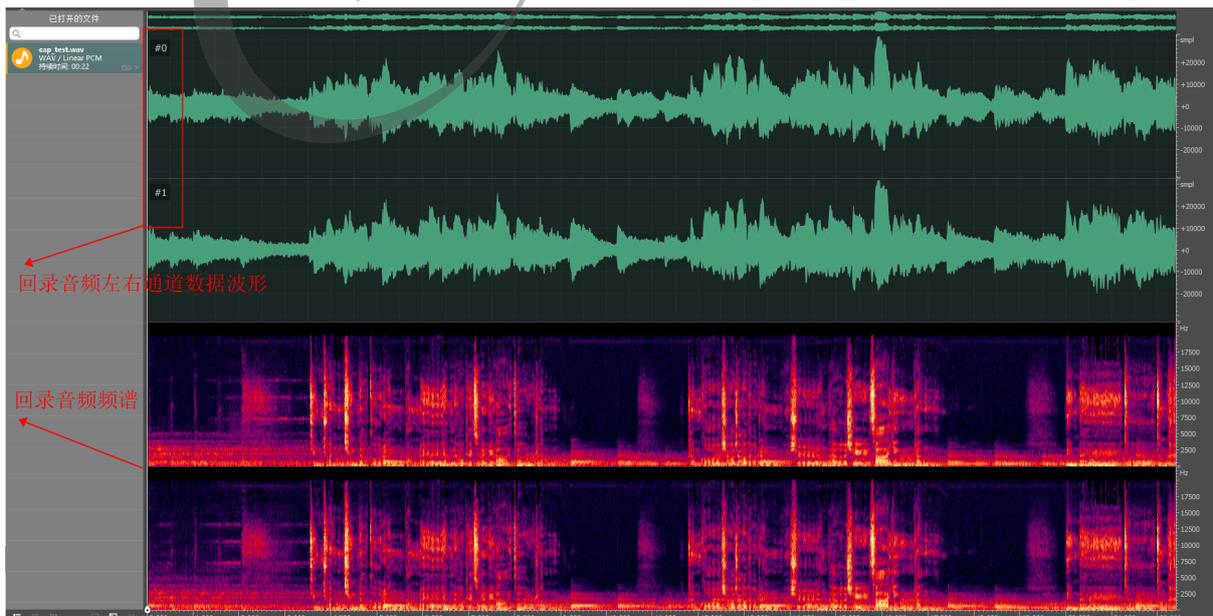


图 4-1: loopback

4.3.2 同源输出功能使用说明

同源输出功能：该功能是 AW SUNXI 平台通过硬件方法实现的一种能够让不同音频接口同时输出播放同一份音频数据的一个功能。

使用方法：

- 将需要进行同源输出的两个或多个声卡的同源输出控件“hub mode”一并使能打开
- 打开需要进行同源输出的两个或多个声卡（pcm_open）；
- 配置相关播放参数等（pcm_config）；
- 往其中的一个音频声卡开始写入数据（pcm_write）即可；
- 最后关闭相应的已打开的音频声卡；

具体验证操作示例如下所示：

```
/ # cat /proc/asound/cards // 查看当前声卡序号
0 [audiocodec ]: audiocodec - audiocodec
audiocodec
1 [snddmic ]: snddmic - snddmic
snddmic
2 [snddauido2 ]: snddauido2 - snddauido2
snddauido2
3 [sndspdif ]: sndspdif - sndspdif
sndspdif

/ # tinymix -D 0 31 1 // 播放通路配置
/ # tinymix -D 0 0 1 // hub功能打开
/ # tinymix -D 0
Mixer name: 'audiocodec'
Number of controls: 32
ctl type num name value
0 ENUM 1 codec hub mode hub_enable
...
15 INT 1 LINEOUT volume 26
...
31 BOOL 1 LINEOUT Switch 0n

/ #
/ # tinymix -D 2 0 1 // hub功能使能
/ # tinymix -D 2 1 1 // loopback 回录功能使能
/ # tinymix -D 2
Mixer name: 'snddauido2'
Number of controls: 2
ctl type num name value
0 ENUM 1 sunxi daudio audio hub mode Enable
1 BOOL 1 sunxi daudio loopback debug 0n

/ #
/ # tinyplay_ahub /Panama_48K.wav -aD 0 -ad 0 -D 2 -d 0 & // 打开 audiocodec 与
snddauido2 声卡并往 audiocodec 声卡写数据
/ # Playing sample: 2 ch, 48000 hz, 16 bit, 1024 period, 4 count
Playing sample: 2 ch, 48000 hz, 16 bit
<----->
start --> loop_minutes = 0; loop_num = 1

/ # tinycap
Usage: tinycap file.wav [-D card] [-d device] [-c channels] [-r rate] [-b bits] [-p
period_size] [-n n_periods] [-t capture time]
```

```
/ # tinycap /tmp/test.wav -D 3 -d 0 -c 2 -r 48000 -b 16 -t 20 & // 由于打开了 snddauio2  
    的loopback回录, 此时对 snddauio2 录音可录到同源输出的音频数据  
/ # Capturing sample: 2 ch, 48000 hz, 16 bit  
  
/ # Captured 962560 frames // loopback回录结束, 可将录音音频拉出通过音频解析  
    软件查看确认
```



5 外挂 Codec 使用说明

详细介绍如何使用 AW SUNXI 平台的 I2S/PCM 接口实现外挂 Codec 并验证外挂 Codec 的输入/输出

5.1 AW SUNXI 平台 I2S/PCM 外挂总览

AW SUNXI 平台的 I2S 模块最高可支持有 4 组 I2S/PCM 接口能够独立工作，分别为 I2S0/I2S1/I2S2/I2S3，分别对应 daudio0/daudio1/daudio2/daudio3。

5.2 外挂 Codec 步骤

5.2.1 硬件相关准备

1、确保硬件通路 OK

(1) 确保硬件板子相关连接都已经准备好，例如包括使用到的 I2C_SDA、I2C_SCK 以及 I2S_MCLK、I2S_BCLK、I2S_LRCK、I2S_DIN、I2S_DOUT 等是否都已连接好，硬件上时钟及数据脚通路是否都以确定 OK；

(2) 主控端 I2S 模块及外挂模块供电正常；

2、硬件实现原理图

(1) 通过相应原理图确认使用的哪组 I2S 及其相应的引脚、复用；

(2) 通过相应原理图确认为实现外挂 Codec 与主控间的通讯而使用的哪组 I2C；

3、外挂 Codec 相关 datasheet

(1) 确认其使用的主/从模式？（master/slave?）

(2) 确认其正常工作的模块时钟频率？

(3) 确认其使用的数据传输模式及时钟信号翻转情况？（I2S/PCM?）

(4) 确认其使用的数据格式配置？（最大位宽？pcm_lrck_period 周期等?）

(5) 外挂 I2C 的 I2C 地址？

5.2.2 软件相关准备

1、I2S 驱动及外挂驱动

- (1) 主控端 I2S 接口模块实现驱动确认支持；
- (2) 外挂 Codec 实现驱动确认支持；

2、软件相关配置修改

- (1) 主控端 I2S 模块 daudio 相关数据格式配置项参数确认及配置；
- (2) 主控端 I2S 模块 snddaudio 节点用于与外挂 Codec 绑定用的节点配置确认（外挂 Codec 的 codec name 与 codec_dai name 可通过外挂 Codec 驱动源码等方式确认）；
- (3) 外挂 Codec 的相关节点配置添加确认；

3、驱动模块使能

- (1) 主控端 I2S 模块使能，包括 board.dts 配置文件模块使能及内核配置 menuconfig 模块使能；
- (2) 外挂 Codec 驱动模块使能，包括 board.dts 配置文件模块使能及内核配置 menuconfig 模块使能；

4、编译通过并打包

- (1) 板型编译通过 make；
- (2) 打包 pack；

5.2.3 上板验证

1、确认模块声卡注册并绑定成功；

- (1) 通过指令：`cat /proc/asound/cards` 查看当前上机启动后是否有成功注册并绑定生成相应的外挂 Codec 名称的声卡；
- (2) 通过 `tinymix` 查看当前注册声卡的音频控件列表是否正常；

2、确认 I2C 通讯正常；

- (1) 可通过指令：`dmesg | grep I2C` 来查看当前是否有 I2C 相关的错误打印，如超时 `xfer` 等；
- (2) 可通过外挂 Codec 提供的模块寄存器调试节点，实时操作节点来读写外挂 Codec 寄存器来确认 I2C 通讯是否正常；

3、外挂 Codec 模块音频通路配置

- (1) 若有需要，通过 tinymix 指令工具配置相应外挂 Codec 的播放输出通路；
- (2) 若有需要，通过 tinymix 指令工具配置相应外挂 Codec 的录音输入通路；

4、播放/录音功能验证

- (1) 通过 tinyplay 指令工具播放指定 WAV 音频文件进行播放验证；
- (2) 通过 tinycap 指令工具进行录制指定路径及名称的 WAV 音频文件，并在录音结束后通过 adb 工具 (adb pull) 将录音文件拉出并通过音频解析软件进行播放、查看确认等；



6 FAQ

- 按要求进行 menuconfig 配置，并且在 board.dts 将 I2S 打开，但无声卡生成。
- 查看 I2S 所使用的引脚是否被其它模块占用。



著作权声明

版权所有 © 2021 珠海全志科技股份有限公司。保留一切权利。

本文档及内容受著作权法保护，其著作权由珠海全志科技股份有限公司（“全志”）拥有并保留一切权利。

本文档是全志的原创作品和版权财产，未经全志书面许可，任何单位和个人不得擅自摘抄、复制、修改、发表或传播本文档内容的部分或全部，且不得以任何形式传播。

商标声明

、 **全志科技** （不完全列举）均为珠海全志科技股份有限公司的商标或者注册商标。在本文档描述的产品中出现的其它商标，产品名称，和服务名称，均由其各自所有人拥有。

免责声明

您购买的产品、服务或特性应受您与珠海全志科技股份有限公司（“全志”）之间签署的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所购买或使用的范围内。使用前请认真阅读合同条款和相关说明，并严格遵循本文档的使用说明。您将自行承担任何不当使用行为（包括但不限于如超压，超频，超温使用）造成的不利后果，全志概不负责。

本文档作为使用指导仅供参考。由于产品版本升级或其他原因，本文档内容有可能修改，如有变更，恕不另行通知。全志尽全力在本文档中提供准确的信息，但并不确保内容完全没有错误，因使用本文档而发生损害（包括但不限于间接的、偶然的、特殊的损失）或发生侵犯第三方权利事件，全志概不负责。本文档中的所有陈述、信息和建议并不构成任何明示或暗示的保证或承诺。

本文档未以明示或暗示或其他方式授予全志的任何专利或知识产权。在您实施方案或使用产品的过程中，可能需要获得第三方的权利许可。请您自行向第三方权利人获取相关的许可。全志不承担也不代为支付任何关于获取第三方许可的许可费或版税（专利税）。全志不对您所使用的第三方许可技术做出任何保证、赔偿或承担其他义务。