Linux I2S 挂载 CODEC
开发指南

版本号: 1.3
发布日期: 2025.05.10

版本历史

版本号	日期	制/修订人	内容描述
1.0	2022.11.25	AWA1692	初始版本
1.1	2024.04.29	AWA2136	增加数据延迟说明
1.2	2025.04.07	AWA2077	新增支持的内核版本
1.3	2025.05.10	AWA2136	增加 I2S 协议格式参数获取调试指南

版权所有 © 珠海全志科技股份有限公司。保留一切权利

目　录

版权所有 © 珠海全志科技股份有限公司。保留一切权利

版权所有 © 珠海全志科技股份有限公司。保留一切权利

版权所有 © 珠海全志科技股份有限公司。保留一切权利

插　图

版权所有 © 珠海全志科技股份有限公司。保留一切权利

版权所有 © 珠海全志科技股份有限公司。保留一切权利

表　格

版权所有 © 珠海全志科技股份有限公司。保留一切权利

1 前言

1.1 文档简介

本文档基于 sunxi 平台基础音频框架介绍，能够让使用者在 sunxi 平台开发使用 I2S 音频驱动，并绑定挂载 CODEC，内容分五大部分。

1. I2S 接口介绍；
2. 适配步骤；
3. I2S 格式参数获取与设置；
4. 挂载 CODEC 测试验证；
5. FAQ。

1.2 目标读者

音频系统相关人员。

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
-	Linux-4.9	sound/soc/sunxi_v2/*
-	Linux-5.4	sound/soc/sunxi_v2/*
-	Linux-5.10	bsp/drivers/sound/platform/*
-	Linux-5.15	bsp/drivers/sound/platform/*
-	Linux-6.6	bsp/drivers/sound/platform/*

1.4 相关术语

版权所有 © 珠海全志科技股份有限公司。保留一切权利

表 1-2: 硬件术语

相关术语	解释说明
CODEC	芯片外部音频编解码芯片
I2S/PCM	外置数字音频接口，常用于外接 CODEC 模块。
AHUB	音频集线器，内部集成 I2S 接口及 DAM 混音器，可实现多路输入播放及硬件混音功能。

表 1-3: 软件术语

相关术语	解释说明
ALSA	Advanced Linux Sound Architecture
ASoC	ALSA System on Chip
samplebit	样本精度，记录音频数据最基本的单位，常见的有 16 位。
channel	通道数，该参数为 1 表示单声道，2 表示立体声，大于 2 表示多声道。
rate	采样率，每秒钟采样次数，该次数是针对帧而言。
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。
period size	每次硬件中断处理音频数据的帧数。
period count	处理完一个 buffer 数据所需的硬件中断次数。
buffer size	数据缓冲区大小 (period size * period count)
tinyalsa	在 Linux 内核中与 ALSA 接口对接的库，可用于基本播录。
alsalib	在 Linux 内核中与 ALSA 接口对接的库，可用于基本播录，并可与常见音频算法组合使用。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

2 I2S 接口介绍

2.1 概述

I2S(Inter-IC Sound 或 Integrated Interchip Sound) 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，该总线专门用于音频设备之间的数据传输，广泛应用于各种多媒体系统。

2.2 硬件接口定义

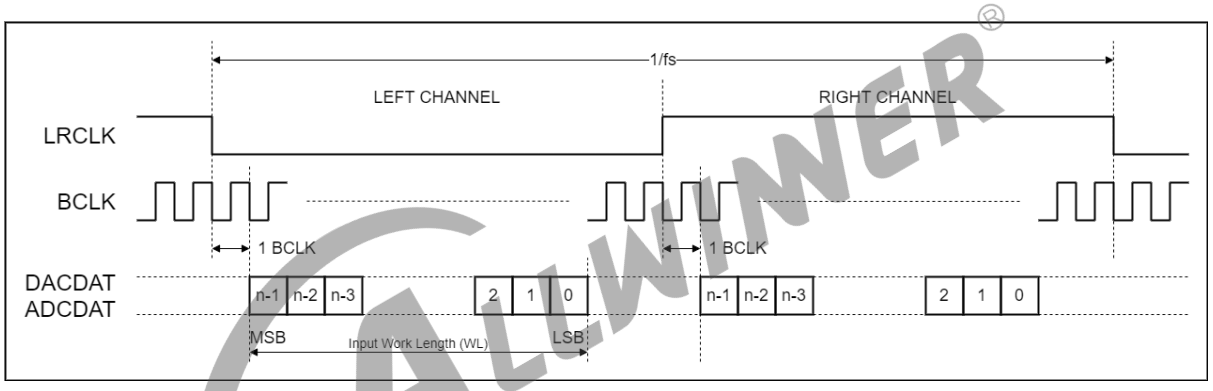


图 2-1: I2S 标准格式

I2S 引脚：

- BCLK : 位同步时钟引脚 (必须)；
- FSYNC : 帧同步时钟引脚，等于音频采样率 (必须)；
- DIN[0-3] : 音频数据接收引脚，正常 1 至 4 线 (非必须)；
- DOUT[0-3]: 音频数据发送引脚，正常 1 至 4 线 (非必须)；
- MCLK : 非 I2S 引脚，为 CPU 提供给 CODEC 的工作时钟 (非必须)。

说明

FSYNC 时钟引脚也可使用 LRCK 名称代替。

I2S 引脚特性：

- MCLK 和 DOUT 为输出引脚，DIN 为输入引脚；
- 音频数据接收和发送引脚分开，即支持全双工工作；
- BCLK & FSYNC 根据所定义的 I2S 主从格式可定义为输入模式或输出模式；
- 一条 I2S 总线至少由 3 条传输线组成，为 FSYNC, BCLK 以及 DIN 或 DOUT。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

2.3 协议参数介绍

I2S 协议具备多种参数，只有 CPU 端和 CODEC 端的 I2S 接口协议设置为一致，才可正常传输音频数据，I2S 协议参数及与常用配套参数内容如下。

2.3.1 MCLK 频率 (非 I2S 协议内容)

参数说明：

为 CPU 提供给 CODEC 的工作时钟的频率，以使模拟/数字转换器的内部操作同步，对于可自行生成工作所需时钟的 CODEC，可无需使用 MCLK。

参数范围：

0 ~ pll_audio (CPU 端最高工作频率)。

常用值：

- 0: 无需 MCLK；
- n * fs: 采样率的整数倍，随采样率变化而变化 (n 常见为 128 等)；
- 11.2896MHz 或 12.288MHz 等较高频率，作为固定频率供 CODEC 工作。

2.3.2 I2S 模式

参数说明：

I2S 协议模式，由标准 I2S 模式拓展，共 5 种。

参数范围：

I2S、I2S_L、I2S_R、DSP_A、DSP_B。

常用值：

标准 I2S。

2.3.3 主从模式

参数说明：

BCLK 和 FSYNC 信号发出方为 CPU 还是 CODEC。

参数范围：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

- CBM_CFM: BCLK CODEC 做主，FSYNC CODEC 做主；
- CBS_CFM: BCLK CODEC 做从，FSYNC CODEC 做主；
- CBM_CFS: BCLK CODEC 做主，FSYNC CODEC 做从；
- CBS_CFS: BCLK CODEC 做从，FSYNC CODEC 做从。

常用值：

CBM_CFM 和 CBS_CFS。

2.3.4 BCLK & FSYNC 时钟翻转

参数说明：

通常情况下，BCLK 和 FSYNC 的电平在不同模式下有着不同的意义，如标准 I2S 模式情况，LRCK 为低电平时代表第一通道 (也称左声道)，高电平则为第二通道，BCLK 电平则在 FSYNC 发生电平变化后从低电平开始，该种电平状态为时钟不翻转 (其它 I2S 模式可见附录)，而翻转状态则电平相反。

参数范围：

- NB_NF: BCLK 正常模式，FSYNC 正常模式；
- NB_IF: BCLK 正常模式，FSYNC 翻转模式；
- IB_NF: BCLK 翻转模式，FSYNC 正常模式；
- IB_IF: BCLK 翻转模式，FSYNC 翻转模式。

常用值：

NB_NF。

2.3.5 短帧 & 长帧

参数说明：

该参数仅对 PCM 模式生效，即 DSP_A、DSP_B 模式。该模式下 FSYNC 时钟不再是 I2S 模式的方波，而是一个脉冲 (FSYNC 波形可见附录)，当脉冲宽度为 1 个 BCLK clock 时为短帧模式，大于 1 个 BCLK clock 时为长帧模式。

参数范围：

- 短帧：1 BCLK clock；
- 长帧：大于 1 BCLK clock，小于 FSYNC 周期。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

常用值：

- 短帧：1 BCLK clock；
- 长帧：2 BCLK clock。

2.3.6 数据传输顺序

参数说明：

I2S 接口传输数据时是按照高比特发生至低比特，或低比特发生至高比特。

参数范围：

- MSB: 高比特发生至低比特；
- LSB: 低比特发生至高比特。

常用值：

MSB.

2.3.7 数据拓展位

参数说明：

通常情况下，若传输的音频数据位数小于 I2S 接口所设置的传输宽度，则需要进行数据拓展，保证有足够的数据供 I2S 接口传输。

参数范围：

0 数据或数据最低位。

常用值：

0 数据。

2.3.8 slot 个数

参数说明：

slot 个数即为 I2S 接口一帧内可传输的最大通道数。

参数范围：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

正常为 2 的倍数，最大值取决于 I2S 的工作时钟、采样率、slot 宽度。

常用值：

无常用值，按需设置。

2.3.9 slot 宽度

参数说明：

slot 宽度即为 I2S 接口传输一通道数据内的最大比特位。

参数范围：

8 ~ 32bit.

常用值：

无常用值，按需设置。

2.3.10 数据延迟

参数说明：

采样点相对于 LRCK 间隔的 BCLK 脉冲个数。

参数范围：

- I2S : 固定为 1;
- I2S_L： 固定为 0;
- I2S_R： 固定为 0;
- DSP_A : 默认为 1，取值范围为 0-3;
- DSP_B : 默认为 0，取值范围为 0-3。

官方对 PCM MODE A(DSP_A) 和 PCM MODE B(DSP_B) 在定义上的主要区别是，DSP_A 的数据延迟固定为 1 bclk，而 DSP_B 的数据延迟固定为 0 bclk。我司对此做了拓展，DSP_A 和 DSP_B 均可根据实际设置数据延迟值，取值范围为 0~3 bclk。即可不区分 DSP_A 与 DSP_B，另行命名为 PCM early mode(data late:0),PCM late1 mode(data late:1),PCM late2 mode(data late:2),PCM late3 mode(data late:3)

2.3.11 通道映射 (非 I2S 协议内容)

参数说明：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

待传输的音频数据通道序号和 slot 序号的映射。

参数范围：

一一映射即可，如将通道 0 映射至 data[0] 数据线的第 0 个 slot，通道 1 映射至 data[1] 数据线的第 3 个 slot。

常用值：

通道序号按照 data 数据线 slot 序号一一对应。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3 适配步骤

I2S 接口挂载 CODEC 可分为 3 种情况，分别如下。

1. 芯片内部连接，已固化芯片内部模块间的引脚连接，并用内部总线控制；
2. 外部引脚连接，外部 CODEC 固化某种 I2S 格式，无需控制接口；
3. 外部引脚连接，外部 CODEC 可变更 I2S 格式，需要控制接口。

针对以上三种情况，本文着重说明情况 3，也是较为常见的场景。

3.1 I2S 接口与 CODEC 硬件连接确认

芯片内部连接：芯片内部将 I2S 引脚固化连接至芯片内部的 CODEC 模块，如通过 I2S 传输数据的内置 CODEC 模块、HDMI 音频播放模块等，该种情况下通常无需关注引脚功能复用，仅需确认所使用的 I2S 引脚有哪些即可。

外部引脚连接 (不带控制)：通过外部引脚连接外部 CODEC 模块，确认所使用的 I2S 引脚和功能复用。

外部引脚连接 (带控制)：通过外部引脚连接外部 CODEC 模块，确认所使用的 I2S 引脚和功能复用，控制接口常用通用接口，如 TWI, SPI 等。

3.1.1 I2S 接口引脚确认

I2S 引脚有 {MCLK, BCLK, FSYNC, DIN[0-3], DOUT[0-3]}，外部引脚连接依据外部 CODEC 需求而定，需求依据如下。

I2S 接口挂载单个外部 CODEC：

1. BCLK 和 FSYNC 是必须使用的引脚；
2. 确认外部 CODEC 是否需要工作时钟，若需要则使用 MCLK 引脚；
3. 确认外部 CODEC 是否有录音功能，需要几线，若需要则使用 DIN 引脚；
4. 确认外部 CODEC 是否有播放功能，需要几线，若需要则使用 DOUT 引脚；

I2S 接口挂载多个外部 CODEC：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

1. BCLK 和 FSYNC 是必须使用的引脚，同时连接至所有 CODEC；
2. 确认外部 CODEC 是否需要工作时钟，若需要则使用 MCLK，同时连接至所有 CODEC；
3. 确认外部 CODEC 是否有录音功能，需要几线，若需要则将 DIN 按序连接至所有 CODEC；
4. 确认外部 CODEC 是否有播放功能，需要几线，若需要则将 DOUT 按序连接至所有 CODEC；

3.1.2 控制接口引脚确认

按控制接口类型在原理图索引对应引脚即可。

3.1.3 I2S 接口与 CODEC 硬件连接示例

> 以单个外部 CODEC AC107 为例。

图 3-1: I2S 外部接口连接 SOC 端

版权所有 © 珠海全志科技股份有限公司。保留一切权利

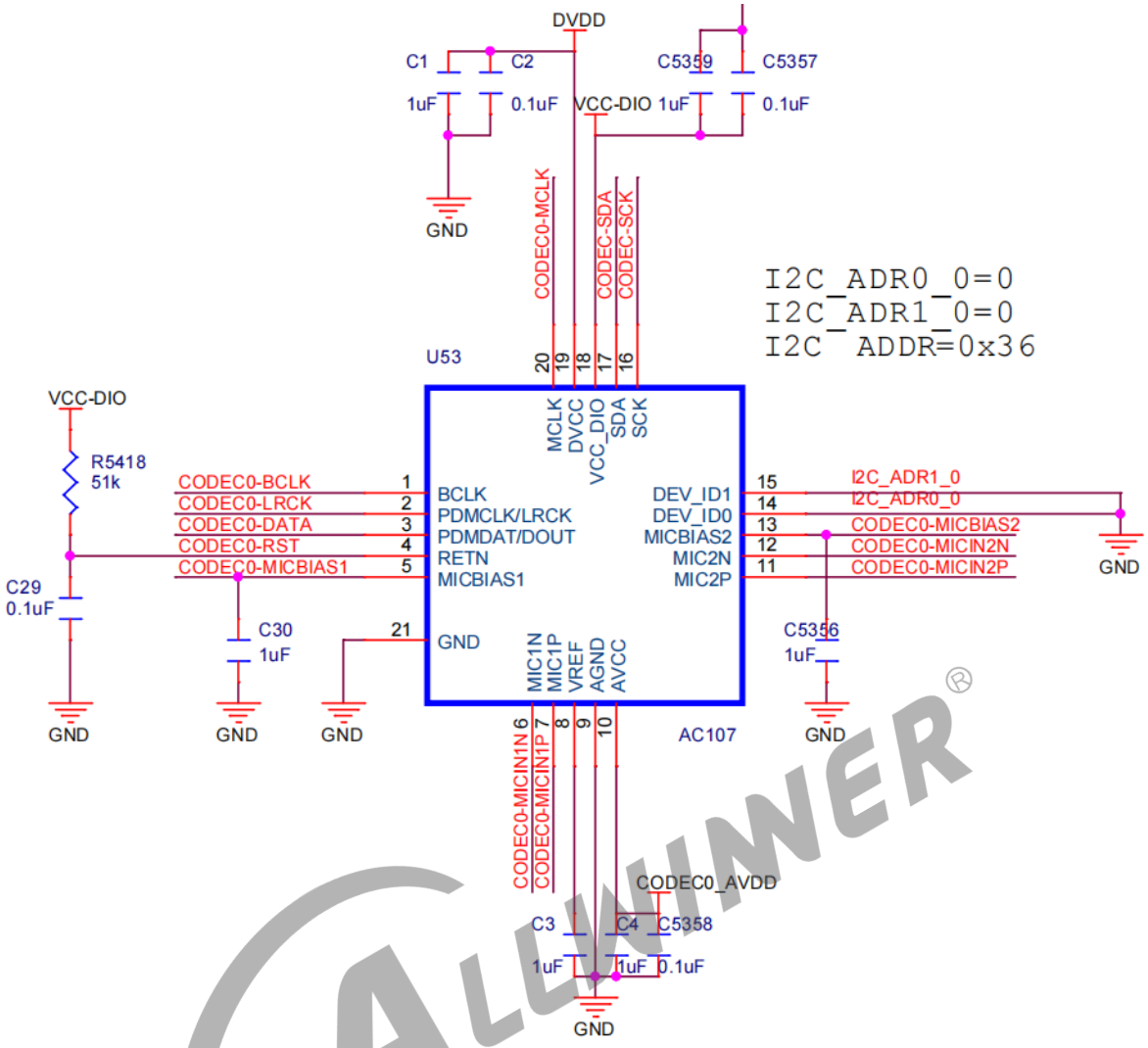


图 3-2: I2S 外部接口连接 CODEC 端

I2S 接口

1. BCLK 对应引脚 PB5，FSYNC 对应引脚 PB6；
2. 需要工作时钟，使用 MCLK，对应引脚 PB4；
3. 需要录音功能，为单线，使用 DIN[0]，对应引脚 PB8；
4. 无需播放功能，不需要使用 DOUT 引脚；

控制接口

需要控制接口，接口为 TWI1，TWI-SCK 对应引脚 PH2，TWI-SDA 对应引脚 PH3。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3.2 配置 CODEC Linux 驱动源码

> 适用于 I2S 接口外部引脚连接 CODEC 且需控制接口，若其它情况，则跳过该步骤。

3.2.1 添加 CODEC 驱动源码

CODEC 驱动源码由 CODEC 原厂提供，获取到驱动源码后将其添加至 Linux 内核源码目录下，不同内核版本目录如下。

内核版本	CODEC 目录
Linux-4.9	sound/soc/codec
Linux-5.4	sound/soc/codec
Linux-5.10	bsp/drivers/sound/codec
Linux-5.15	bsp/drivers/sound/codec
Linux-6.6	bsp/drivers/sound/codec

添加完毕后仿照原有的其它 CODEC 源码，修改其同目录下的 Kconfig 和 Makefile 文件，让其加入编译即可。

3.2.2 添加编译配置

Kconfig 和 Makefile 文件修改示例如下 (以 AC107 为例)。

Kconfig

```
config SND_SOC_AC107
  tristate "Allwinner AC107 CODEC"
  depends on I2C
  default n
  help
    Enable support for the Allwinner AC107 CODEC.
    The device provides a I2S/TDM/PDM interface for audio data
    and a standard I2C interface for control data communication.
    Select this if your sound card has AC107.
```

Makefile

```
snd-soc-ac107-objs := ac107.o

obj-$(CONFIG_SND_SOC_AC107) += snd-soc-ac107.o
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3.3 CODEC 端 I2S 协议参数获取

I2S 接口是 CPU 端和 CODEC 端进行音频传输的接口，其协议具备多种参数，只有 CPU 端和 CODEC 端的 I2S 接口协议一致，才可正常传输音频数据，同时两端的 I2S 接口通常不会支持全规格的 I2S 协议，因此需要 CPU 端和 CODEC 端所支持的协议存在重叠才可使用。

由于 CPU 端的所支持的 I2S 协议格式较多，因此在 I2S 协议参数获取与设置上，优先确认 CODEC 端协议参数，再将 CPU 端协议参数设置为与 CODEC 端一致。

说明

I2S 协议参数介绍参考协议参数介绍 章节。

3.3.1 功能需求

1. 仅播放、仅录音、同时支持播放和录音三种功能需求；
2. 数据线为单线模式、数据线为多线模式 (需确认使用哪些数据线)。

3.3.2 常用协议参数

1. MCLK 频率 (非 I2S 协议内容)
   - 无需 MCLK 提供频率
   - 需要 MCLK 提供 n*fs 频率
   - 需要 MCLK 提供固定频点频率
2. I2S 模式
   - I2S、I2S_L、I2S_R、DSP_A、DSP_B
3. 主从模式
   - CBM_CFM、CBS_CFM、CBM_CFS、CBS_CFS
4. BCLK & FSYNC 时钟翻转
   - NB_NF、NB_IF、IB_NF、IB_IF
5. slot 个数
   - 2 ~ 16(需 2 的倍数)
6. slot 宽度
   - 8 ~ 32bit

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3.3.3 非常用协议参数

1. 数据传输顺序
   - MSB、LSB
2. 数据拓展位
   - 0 数据或数据最低位。
3. 短帧 & 长帧
   - 短帧：1 BCLK clock；
   - 长帧：大于 1 BCLK clock，小于 FSYNC 周期。
4. 通道映射 (非 I2S 协议内容)
   - slot 序号和音频通道的映射关系。

说明

通常仅需更改常用协议参数即可，非常用协议参数保持默认。

3.3.4 CODEC 端 I2S 协议参数示例

以 AC107 为例。

功能需求

仅录音，数据线为单线模式。

常用协议参数

1. MCLK 频率 (非 I2S 协议内容): 固定频点 (11.2896MHz 或 12.288MHz)
2. I2S 模式: I2S
3. 主从模式: CBS_CFS
4. BCLK & FSYNC 时钟翻转: NB_NF
5. slot 个数: 2
6. slot 宽度: 32bit

非常用协议参数

1. 数据传输顺序: MSB
2. 数据拓展位: 0 数据
3. 短帧 & 长帧: I2S 模式无需该参数
4. 通道映射 (非 I2S 协议内容): slot 序号与音频通道序号一致。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3.4 设备树配置

设备树一般分为 chip 级和 board 级共两个配置文件，在芯片端 I2S 驱动已完成开发的情况下，适配 CODEC 仅需更改 board 级设备树。

对于不同挂载情况，设备树配置步骤有所不同，分别如下。

1. 芯片内部连接，配置 I2S 驱动、配置 CODEC 驱动；
2. 外部引脚连接，配置 I2S 驱动、配置 I2S 引脚；
3. 外部引脚连接，配置 I2S 驱动、配置 I2S 引脚、配置控制引脚、配置 CODEC 驱动。

3.4.1 配置引脚

表 3-2: GPIO 功能复用配置项

配置项名称	配置值范围	配置项说明
pinctrl-used	注释为 false, 反之为 ture	是否使用引脚复用功能。
pinctrl-names	“default”，“sleep”	对 pinctrl 属性内容进行名称定义，用于辅助 pinctrl 属性获取。
pinctrl-0	模块 pin 功能复用节点	对应 pinctrl-names 第 0 个属性。
pinctrl-1	模块 pin 功能复用节点	对应 pinctrl-names 第 1 个属性。

表 3-3: 模块引脚组定义说明 (linux-4.9)

节点配置	解释说明
allwinner,pins	模块需要使用到的引脚组定义。
allwinner,function	模块引脚组复用名称。
allwinner,muxsel	模块引脚组复用类型，需和 function 对应。
allwinner,driver	模块引脚驱动力，可选值为 0,1,2,3，默认配置为 1 即可。
allwinner,pull	0: 关闭上下拉，1: 支持上拉，2: 支持下拉（默认配置为 0）。

表 3-4: 模块引脚组定义说明 (linux-5.4, linux-5.10, linux-5.15, linux-6.6)

节点配置	解释说明
pins	模块需要使用到的引脚组定义。
function	模块引脚组复用名称。
drive-strength	模块引脚驱动力，可选值为 10,20,30,40，默认配置为 20 即可。
bias-disable	关闭上下拉（默认选择该项）。
bias-pull-up	支持上拉（默认关闭）。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

节点配置	解释说明
bias-pull-down	支持下拉（默认关闭）。

以 I2S0 接口连接外部单颗 CODEC AC107 为例。

I2S 引脚

```
i2s0_pins_a: i2s0@0 {
    /* MCLK, BCLK, FSYNC, DIN[0] */
    pins = "PB4", "PB5", "PB6", "PB8";
    function = "i2s0";
    drive-strength = <20>;
    bias-disable;
};

i2s0_pins_b: i2s0_sleep@0 {
    pins = "PB4", "PB5", "PB6", "PB8";
    function = "io_disabled";
    drive-strength = <20>;
    bias-disable;
};

&i2s0_plat {
    pinctrl-names   = "default","sleep";
    pinctrl-0       = <&i2s0_pins_a>;
    pinctrl-1       = <&i2s0_pins_b>;
};
```

使用 I2S0 连接，需要 MCLK，无需播放功能，使用单线，为 DIN[0]。

i2s0_pins_a: 将 “PB4”，“PB5”，“PB6”，“PB8” 引脚复用为 i2s0，驱动能力为 20；

i2s0_pins_b: 将 “PB4”，“PB5”，“PB6”，“PB8” 引脚关闭。

控制引脚

```
twi1_pins_a: twi1@0 {
    /* SCK, SDA */
    pins = "PH2", "PH3";
    function = "twi1";
    drive-strength = <10>;
};

twi1_pins_b: twi1@1 {
    pins = "PH2", "PH3";
    function = "io_disabled";
};

&twi1 {
    pinctrl-names = "default", "sleep";
    pinctrl-0 = <&twi1_pins_a>;
    pinctrl-1 = <&twi1_pins_b>;
}
```

使用 twi1 连接，TWI-SCK 对应引脚 PH2，TWI-SDA 对应引脚 PH3。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

twi1_pins_a: 将 “PH2”，“PH3” 引脚复用为 twi1，驱动能力为 10；

twi1_pins_b: 将 “PH2”，“PH3” 引脚关闭。

3.4.2 配置 CODEC 驱动

> 以 AC107 控制接口为 twi1 为例。

```
&twi1 {
    clock-frequency = <400000>;
    pinctrl-names = "default", "sleep";
    pinctrl-0 = <&twi1_pins_a>;
    pinctrl-1 = <&twi1_pins_b>;
    status = "okay";
    ac107: ac107@36 {
        #sound-dai-cells    = <0>;
        compatible          = "allwinner,sunxi-ac107";
        reg                 = <0x36>;
        pllclk-src          = "MCLK";
        sysclk-src          = "MCLK";
        pcm-bit-first       = "MSB";
        frame-sync-width    = <1>;
        rx-chmap            = <0xaaaa>;
        ch1-dig-vol         = <160>;
        ch2-dig-vol         = <160>;
        ch1-pga-gain        = <31>;
        ch2-pga-gain        = <31>;
        status              = "okay";
    };
};
```

因为 AC107 驱动是当作 TWI 设备注册，因此需要在 TWI1 节点下添加 ac107 子节点，同时需将 TWI1 和 AC107 节点的 status 属性均设置为 “okay”。

其中 AC107 节点的各个属性值则依据 TWI 地址、驱动源码特性等按需配置，和 CODEC 原厂获取相关配置使用说明即可。

3.4.3 配置 I2S 驱动

> 以 AC107 I2S 接口为 I2S0 为例。

I2S 设备节点属性说明

表 3-5: I2S/PCM 模块板级配置项

配置项名称	配置值范围	配置项说明
status	“okay”，“disabled”	使能或关闭该节点驱动
tdm-num	0~1	指定 I2S 序号，

版权所有 © 珠海全志科技股份有限公司。保留一切权利

配置项名称	配置值范围	配置项说明
		需和 i2s(n)_plat 的 (n) 对应
tx-pin	0~3	指定 I2S 所使用的 DOUT 引脚序号
rx-pin	0~3	指定 I2S 所使用的 DIN 引脚序号
tx-hub-en	注释为 false, 反之为 ture	选择是否注册 txhub 控件
rx-sync-en	注释为 false, 反之为 ture	选择是否注册 rxsync 控件
format	“i2s”，“right_j”，“left_j”，“dsp_a”，“dsp_b”	选择 tdm 协议格式
frame-master	cpu 子节点，codec 子节点	选择 LRCK 信号主模式
bitclock-master	cpu 子节点，codec 子节点	选择 BCLK 信号主模式
frame-inversion	注释为 false, 反之为 ture	LRCK 信号是否翻转
bitclock-inversion	注释为 false, 反之为 ture	BCLK 信号是否翻转
slot-num	1~16	slot 数量（可简单理解为支持最大通道数）
slot-width	8, 16, 24, 32	单个 slot 宽度（可简单理解为支持最大数据精度）
mclk-fp	mclk-fp[0] 为 44100 的倍数; mclk-fp[1] 为 48000 的倍数； mclk-fp 注释时为 false	配置数值: mclk 以固定频段输出 false: mclk 以采样率倍数输出
mclk-fs	u32	固定频段：mclk = mclk-fs * mclk-fp[0] or mclk-fp[1] 采样率倍数：mclk = mclk-fs * pcm rate

配置 I2S 驱动配置示例

```
&i2s0_plat {
  tdm-num    = <0>;              /* I2S 接口为 0 */
  tx-pin     = <0>;              /* 使用 DOUT[0] */
  rx-pin     = <0>;              /* 使用 DIN[0] */
  pinctrl-used;
  pinctrl-names  = "default","sleep";  /* 工作时引脚状态为 pinctrl-0 */
  pinctrl-0      = <&i2s0_pins_a>;  /* 工作时 I2S 引脚复用 */
  pinctrl-1      = <&i2s0_pins_b>;  /* 不工作时 I2S 引脚复用 */
  status = "okay";               /* 使能该节点驱动 */
};

&i2s0_mach {
  soundcard-mach,format        = "i2s";   /* I2S 接口格式为 "i2s" */
  soundcard-mach,frame-master    = <&i2s0_cpu>;  /* FSYNC 由 CPU 做主 */
  soundcard-mach,bitclock-master  = <&i2s0_cpu>;  /* BCLK 由 CPU 做主 */
  /* soundcard-mach,frame-inversion; */     /* FSYNC 信号不翻转 */
  /* soundcard-mach,bitclock-inversion; */   /* BCLK 信号不翻转 */
  soundcard-mach,slot-num       = <2>;    /* slot 个数为 2 */
  soundcard-mach,slot-width      = <32>;   /* slot 宽度为 32 */
  status = "okay";                  /* 使能该节点驱动 */
  i2s0_cpu: soundcard-mach,cpu {
    sound-dai = <&i2s0_plat>;     /* I2S CPU 端驱动为 i2s0_plat */
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    soundcard-mach,pll-fs  = <1>;    /* CPU 端驱动父时钟频率系数，无需关注 */
    soundcard-mach,mclk-fp  = <11289600 12288000>; /* CPU 端 MCLK 频率为固定频点 */
    soundcard-mach,mclk-fs  = <1>;    /* CPU 端 MCLK 频率为
                              mclk-fs * mclk-fp[0] or mclk-fp[1] */
  };
  i2s0_codec: soundcard-mach,codec {
    sound-dai = <&ac107>;          /* I2S CODEC 端驱动为 ac107 */
  };
};
```

i2s0_plat 节点: CPU 端 I2S0 的设置，按需设置 I2S 引脚。

i2s0_mach 节点: 根据 CODEC 端所需的协议格式，将 CPU 端 I2S 接口协议格式设置为一致，并引用 CPU 端和 CODEC 端的驱动，将其进行绑定。

3.5 配置 menuconfig

I2S 驱动 menuconfig 配置

Linux-5.4 之前 (包含 Linux-5.4) 的版本。

```
Device Drivers  --->
  <*> Sound card support  --->
    <*> Advanced Linux Sound Architecture  --->
      <*> ALSA for SoC audio support  --->
        Allwinner SoC Audio support V2  --->
          <M> Allwinner DAUDIO support
```

Linux-5.4 之后的版本。

```
Allwinner BSP  --->
  Device Drivers  --->
    SOUND Drivers  --->
      Platform drivers  --->
        <M> Allwinner I2S Support
```

选择需要的模块，可选择直接编译进内核（Y），也可编译成模块（M）。

CODEC 驱动 menuconfig 配置

按照 CODEC 添加编译方式进行相应配置即可。

3.6 声卡确认

上述步骤完成后，即可编译生成固件，等待系统启动后查看是否生成对应声卡。

查看声卡命令。

```
cat /proc/asound/cards
```

如果没有对应声卡，根据 log 查看FAQ 章节。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

说明

若 menuconfig 配置为（M），则系统启动后按顺序加载 ko，再查看是否生成声卡。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

4 挂载 CODEC 测试验证

适用于声卡创建成功的情况。

4.1 软件测试验证

使用 tinyalsa 工具进行播放或录音测试，如挂载 AC107，则使用 tinymix 工具配置相应通路，再用 tinycap 工具进行录音，并查看录音结果是否正确。

4.2 硬件测试验证

适用于外部引脚连接 CODEC 场景。

在播放或录音的过程中，使用示波器查看 I2S 相关引脚 {MCLK, BCLK, FSYNC, DIN[0-3], DOUT[0-3]} 波形是否正确，波形频率、幅度以及各信号之间的关系可查看附录中的I2S 格式协议图 章节。

MCLK 频率验证：根据 MCLK 频率配置，需和所配置一致，否则为验证失败；

BCLK 频率验证：BCLK 频率 = slot * slot_width * fs，若不一致，则验证失败；

LRCK 频率验证：LRCK 频率 = fs，若不一致，则验证失败；

DIN 和 DOUT 波形：与所录或所播音频数据相关，无固定频率，有声音情况下应有波形跳变。

说明

上述中 fs 表示播放或录音时所使用的采样率，如 44.1kHz，48KHz 等。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

5 I2S 协议格式参数获取

协议格式参数的三个获取途径：1. 外挂 CODEC 芯片手册；2. 外挂 CODEC 驱动源码；3. 使用示波器测量的 I2S 时序图。

说明

- **获取外挂 CODEC 完整 I2S 协议格式信息最快的方式是咨询 CODEC 原厂；**
- **带 I2C 或 SPI 控制引脚的 CODEC 可通过寄存器配置 I2S 协议格式，否则通过硬件电路连接或固定格式配置 I2S 协议；**
- **若 CODEC 芯片手册未提供 I2S 协议格式描述，需在 CODEC 为主的前提下，通过示波器测量 I2S 时序图确认协议格式；**
- **确认外挂 CODEC 的完整 I2S 协议格式后，需在 SoC 端进行协议格式对齐配置。**

- I2S 设备节点属性参考I2S 设备节点属性说明 。

5.1 I2S 模式

5.1.1 从芯片手册中获取

搜索关键词 (不区分大小写)：format, mode, left, right, long/short frame, msb/lsb justified, i2s...等，示例如下：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

5.1.1.1 通过硬件方式设置 I2S 模式

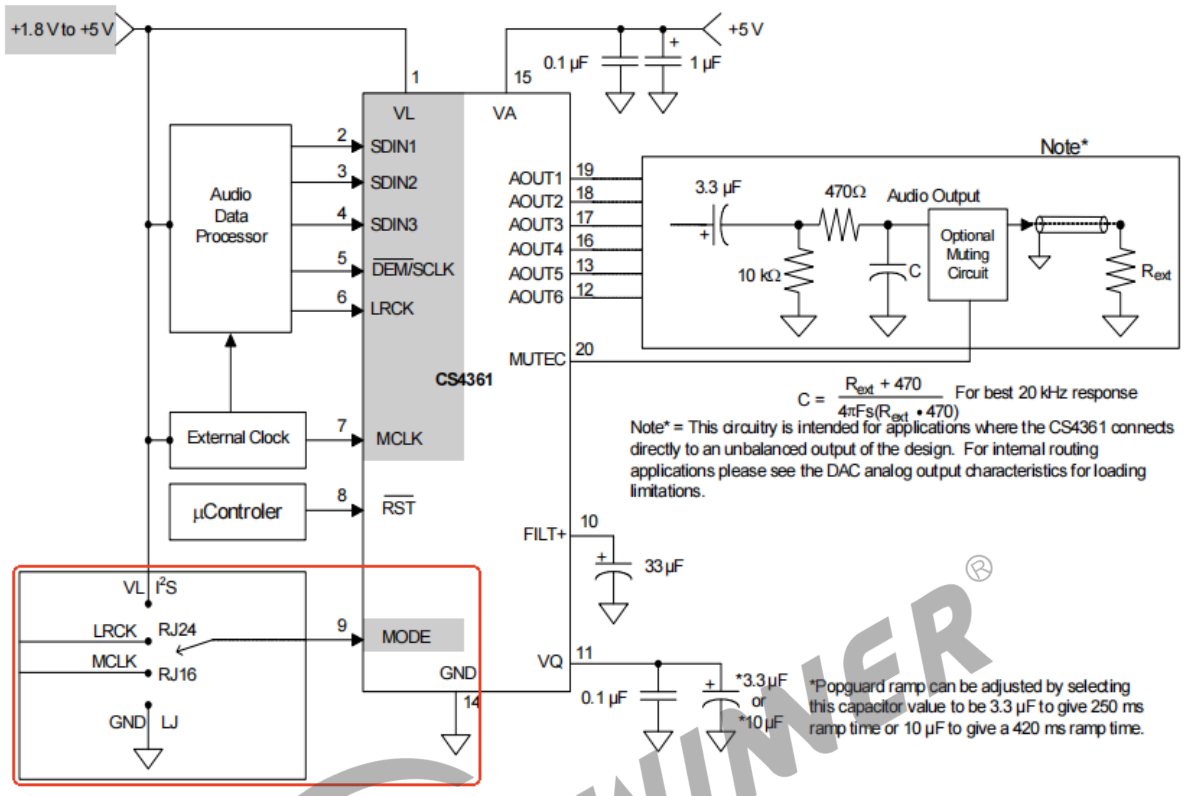


图 5-1: CS4361: 硬件电路

Mode Select

Mode selection is determined by the Mode Select pin. The value of this pin is locked 1024 LRCK cycles after RST is released. This pin requires a specific connection to supply, ground, MCLK, or LRCK as outlined in Table 2.

Mode pin is:	Mode	Figure
Tied to VL	I²S	7
Tied to GND	Left-Justified	8
Tied to LRCK	Right-Justified - 24 bit	9
Tied to MCLK	Right-Justified - 16bit	10

Table 2. Mode Pin Settings

图 5-2: CS4361: 模式配置说明

例如：CS4361 的 MODE 引脚与 VL 引脚连接时，CODEC 端采用标准 I2S 模式。DTS 配置如下示例：

```
&i2s0_mach {
    ...
    /* 对齐SoC端与Codec端的I2S模式 */
    soundcard-mach,format      = "i2s";
    ...
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
};
```

5.1.1.2 通过寄存器方式设置模式

Command	Addr	Register Name	D7	D6	D5	D4	D3	D2	D1	D0
	0x0040	BICK Format Setting1	BCKP1	Reserved	Reserved	Reserved	Reserved	DCF1[2:0]		
W: 0xC0	R/W		R/W	R/W	R/W	R/W	R/W	R/W		
R: 0x40	Default		0	0	0	0	0	0h		

BCKP1: Relationship of LRCK1 and BICK1 Edges
0: LRCK1 starts on a BICK1 falling edge (default)
1: LRCK1 starts on a BICK1 rising edge

DCF1[2:0]: LRCK1/BICK1 Data Format Setting
000: I^{2}S Mode (default)
101: DSP Mode
110: PCM Short Frame
111: PCM Long Frame

图 5-3: AK7739: 模式相关寄存器配置说明

```
/* set format */
setSDMaster(codec, nSDNo, msnbit);
addr = AK7739_C0_040_BICK_FORMAT_SETTING1 + 2 * nSDNo;
snd_soc_update_bits(codec, addr, 0x87, format);
```

图 5-4: AK7739: 驱动设置模式代码段

例如：AK7739 通过 0x0040 寄存器的 DCF1[2:0] 控制 I2S 模式，CODEC 原厂驱动将 DCF1 配置为 0x7, 即采用 PCM 长帧模式，AW 驱动中 dsp_b 默认代表 PCM 长帧模式。DTS 配置如下示例：

```
&i2s0_mach {
   ...
   /* 对齐SoC端与Codec端的I2S模式 */
   soundcard-mach,format      = "dsp_b";
   ...
};
```

5.1.2 从示波器时序图中获取

- 在 CODEC 做主的前提下，通过 LRCK 和 BCLK 图像可以快速区分 I2S 协议模式。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

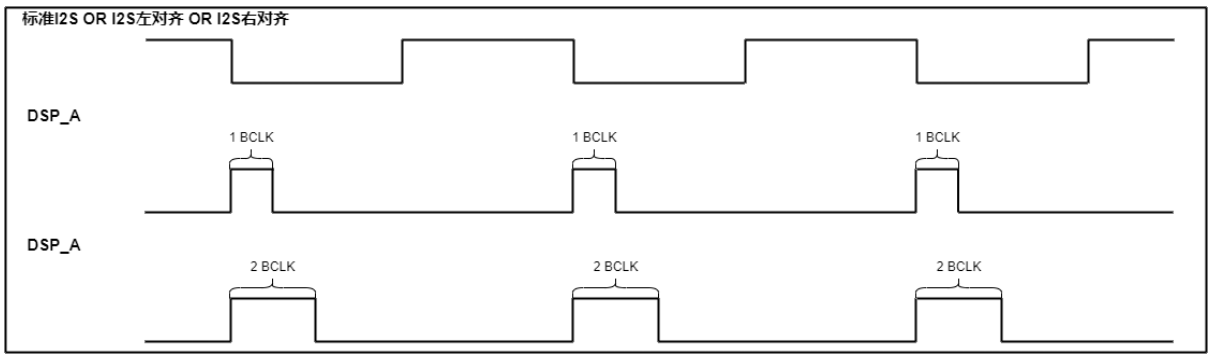


图 5-5: 两类协议的 LRCK 区别

当 CODEC 具备 ADC 时，可继续往下分析模式类型，否则只能通过遍历方式确定模式。

- 设置采样位深小于 slot 宽度（如位深采用 16bit，slot_width 设置为 32bit），通过 ADC DATA 信号图像观察数据开始传输的位置：

若起始采样点总是相对于 LRCK 间隔 1 个 BCLK 脉冲，则为标准 I2S 模式；

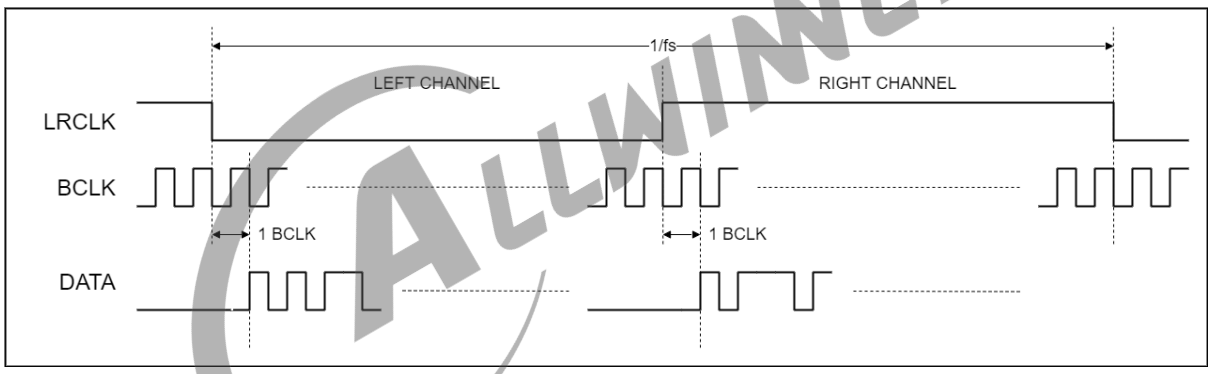


图 5-6: 标准 I2S 模式时序图

若起始采样点总是相对于 LRCK 间隔 0 个 BCLK 脉冲，则为 I2S 左对齐或 I2S 右对齐模式；

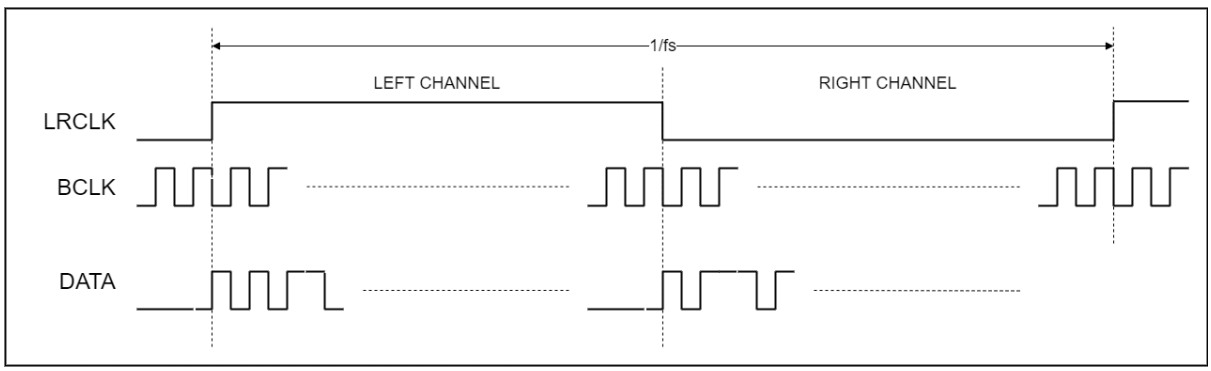


图 5-7: 左对齐或右对齐时序图

版权所有 © 珠海全志科技股份有限公司。保留一切权利

5.2 主从模式

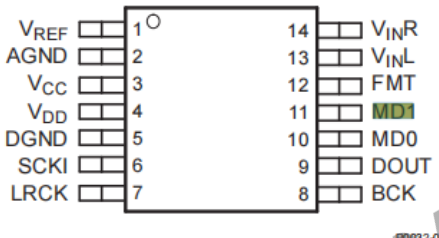
5.2.1 从芯片手册中获取

搜索关键词 (不区分大小写)：internal, external, master, slave, mode…等，示例如下：

5.2.1.1 通过硬件方式设置主从模式

PIN ASSIGNMENTS

PW PACKAGE
(TOP VIEW)



TERMINAL FUNCTIONS

TERMINAL NAME	TERMINAL PIN	I/O	DESCRIPTION
AGND	2	–	Analog GND
BCK	8	I/O	Audio data bit clock input/output (1)
DGND	5	–	Digital GND
DOUT	9	O	Audio data digital output
FMT	12	I	Audio interface format select (2)
LRCK	7	I/O	Audio data latch enable input/output (1)
MD0	10	I	Audio interface mode select 0 (2)
MD1	11	I	Audio interface mode select 1 (2)
SCKI	6	I	System clock input; 256 f_S, 384 f_S or 512 f_S (3)
V_{CC}	3	–	Analog power supply, 5-V
V_{DD}	4	–	Digital power supply, 3.3-V
$V_{IN}L$	13	I	Analog input, L-channel
$V_{IN}R$	14	I	Analog input, R-channel
V_{REF}	1	–	Reference voltage decoupling (= 0.5 V_{CC})

图 5-8: PCM1808: 硬件电路

版权所有 © 珠海全志科技股份有限公司。保留一切权利

INTERFACE MODE

The PCM1808 supports master mode and slave mode as interface modes, which are selected by MD1 (pin 11) and MD0 (pin 10), as shown in Table 2. MD1 and MD0 must be set prior to power on.

In master mode, the PCM1808 provides the timing of serial audio data communications between the PCM1808 and the digital audio processor or external circuit. While in slave mode, the PCM1808 receives the timing for data transfer from an external controller.

Table 2. Interface Modes

MD1 (Pin 11)	MD0 (Pin 10)	INTERFACE MODE
Low	Low	Slave mode (256 f_S, 384 f_S, 512 f_S autodetection)
Low	High	Master mode (512 f_S)
High	Low	Master mode (384 f_S)
High	High	Master mode (256 f_S)

图 5-9: PCM1808: 主从模式配置说明

例如：MOD0 与 MOD1 脚均为低电平时，CLK 接口为 slave 模式。DTS 配置如下示例：

```
&i2s0_mach {
  ...
  /* 由于CODEC为slave模式，故需设置SoC端为master模式 */
  soundcard-mach,frame-master    = <&i2s0_cpu>;
  soundcard-mach,bitclock-master = <&i2s0_cpu>;
  ...
};
```

5.2.1.2 通过寄存器方式设置主从模式

4.3 CONFIGURE ES7243E INTO MASTER OR SLAVE MODE

ES7243E can work either in master clock mode or slave clock mode. In slave mode, LRCK and SCLK are supplied externally, and LRCK and SCLK must be synchronously derived from the system clock with specific rates. In master mode, LRCK and SCLK are derived internally from device master clock.

Bit6 (MSC) of Register 0x00 is used to set ES7243E into either master or slave mode.

Register0x00.bit6(MSC)	Master / Slave
MSC = 0	Slave mode
MSC = 1	Master mode

图 5-10: ES7243: 主从模式相关寄存器配置说明

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
static struct es7243_reg init_mode[] = {
    /* slave mode, software mode */
    {ES7243_MODECFG_REG00, 0x01},
};

int es7243_init_mode(struct i2c_client *client)
{
    int i = 0;
    int err = 0;
    printk("%s\n", __func__);
    for(i = 0; i < ES7243_INIT_NUM; i++) {
        err = es7243_i2c_write(client, init_mode[i].reg_index,
                init_mode[i].reg_value);
        if(err != 0 )
            dev_err(&client->dev, "i2c write 0x%0x failed\n",
                    init_mode[i].reg_index);
    }
    return err;
}
```

图 5-11: ES7243: 驱动设置模式代码段

例如：ES7243 通过 0x00 寄存器的 MSC[bit 6] 控制主从模式，CODEC 原厂驱动将 MSC 配置为 0x01, 即采用 master 模式。DTS 配置如下示例：

```
&i2s0_mach {
  ...
  /* 由于CODEC为master模式，故需设置SoC端为slave模式 */
  soundcard-mach,frame-master    = <&i2s0_codec>;
  soundcard-mach,bitclock-master = <&i2s0_codec>;
  ...
};
```

5.2.2 从示波器时序图中获取

- 将 SoC 设置为 slave 模式，测量 CODEC 在工作过程中的 BCLK 与 LRCK 引脚状态，若有信号输出则 CODEC 为 master，反之为 slave。

5.3 时钟极性

5.3.1 从芯片手册中获取

首先通过芯片手册或从示波器时序图中获取 I2S 模式，再进一步确定时钟极性，一般情况下：

- CODEC 采用 I2S 左对齐与 I2S 右对齐模式时需翻转 LRCK，无需翻转 BCLK；CODEC 采用标准 I2S 模式时无需翻转 LRCK 与 BCLK；

版权所有 © 珠海全志科技股份有限公司。保留一切权利

- CODEC 采用 dsp_a 模式时需要翻转 BCLK，无需翻转 LRCK；CODEC 采用 dsp_b 模式无需翻转 BCLK 与 LRCK。

DTS 配置如下示例：

```
&i2s0_mach {
    ...
    /* I2S模式时钟极性常规配置 */
    /* soundcard-mach,frame-inversion; */
    /* soundcard-mach,bitclock-inversion; */

    /* 左对齐或右对齐模式时钟极性常规配置 */
    soundcard-mach,frame-inversion;
    /* soundcard-mach,bitclock-inversion; */

    /* DSP_A模式时钟极性常规配置 */
    /* soundcard-mach,frame-inversion; */
    soundcard-mach,bitclock-inversion;

    /* DSP_B模式时钟极性常规配置 */
    /* soundcard-mach,frame-inversion; */
    /* soundcard-mach,bitclock-inversion; */
    ...
};
```

5.3.2 从示波器时序图中获取

- 针对 dsp_a 与 dsp_b 模式，在 CODEC 做主的前提下，LRCK 脉冲在 BCLK 的上升沿发生翻转，则 SoC 端需要翻转 BCLK，否则无需翻转;

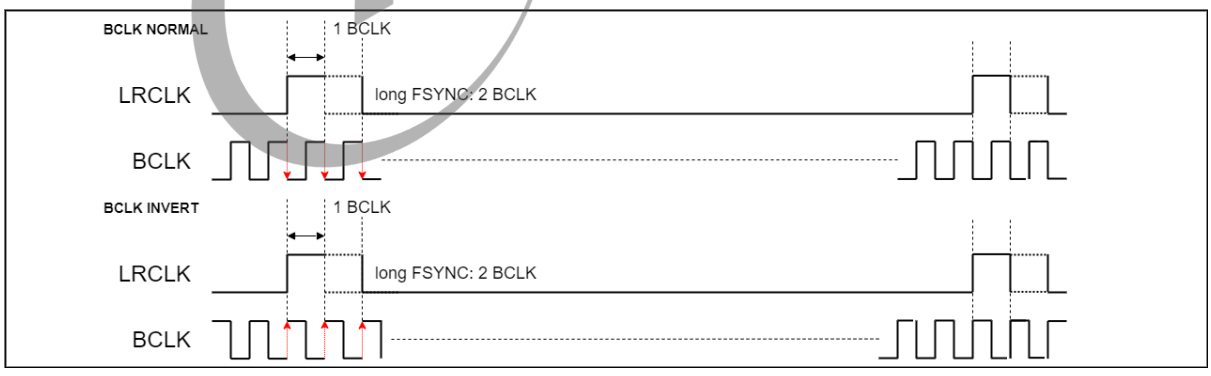


图 5-12: 翻转 BCLK

- 针对 dsp_a 与 dsp_b 模式，在 CODEC 做主的前提下，LRCK 脉冲为低电平，则 SoC 端需要翻转 LRCK，否则无需翻转;

版权所有 © 珠海全志科技股份有限公司。保留一切权利

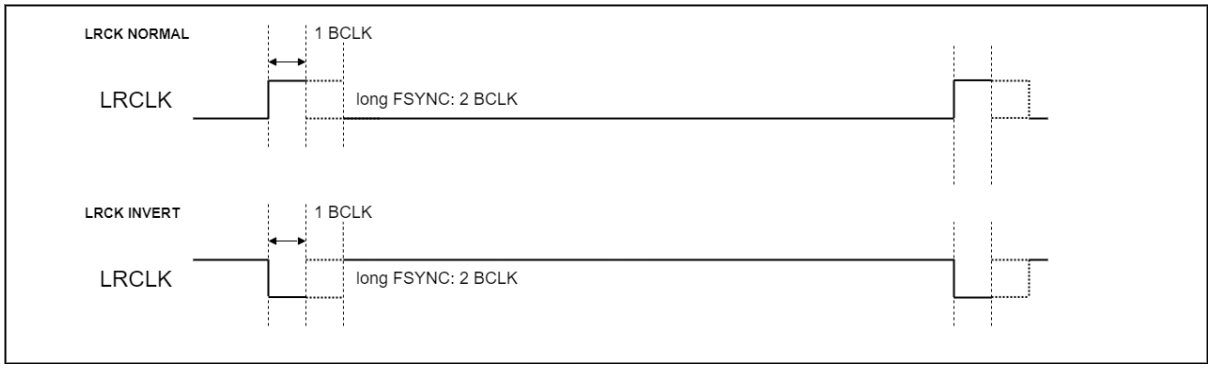


图 5-13: 翻转 LRCK

5.4 slot 宽度

5.4.1 从芯片手册中获取

搜索关键词 (不区分大小写)：bit, slot width, slot length，Data size…等, 示例如下：

5.4.1.1 通过硬件方式设置 slot 宽度

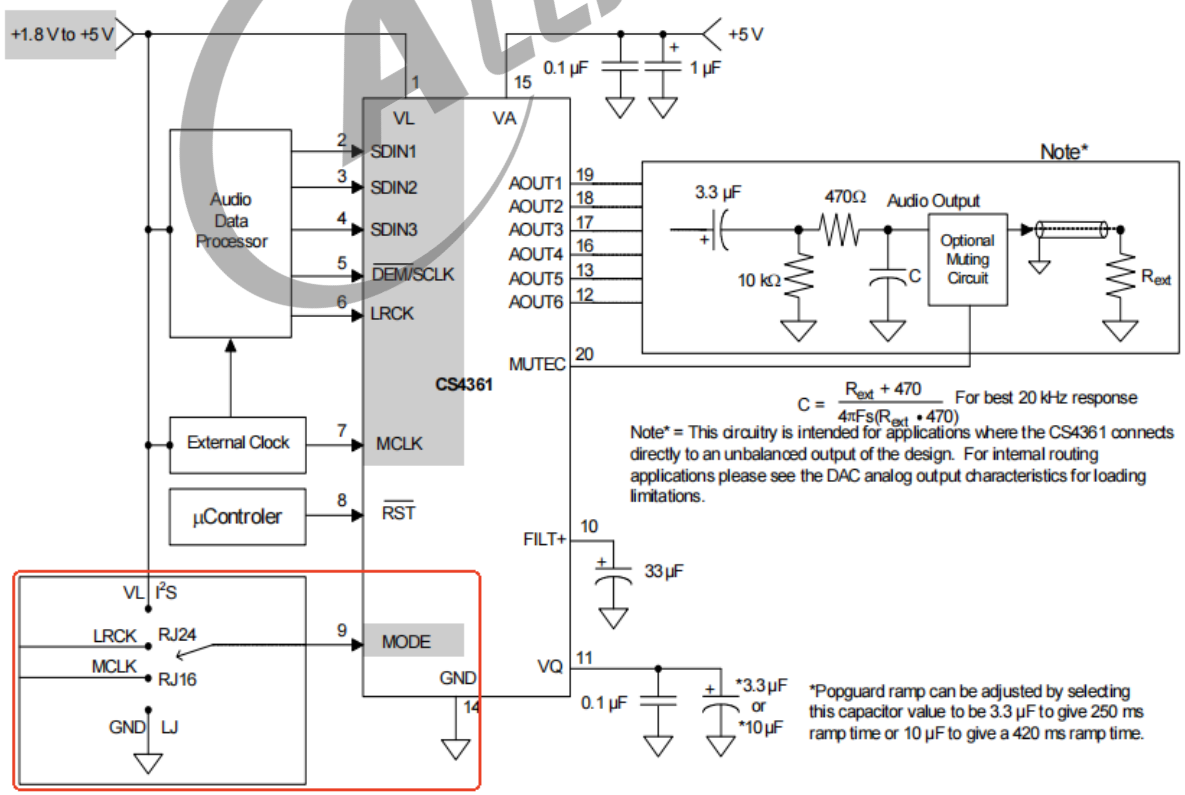


图 5-14: CS4361: 硬件电路

版权所有 © 珠海全志科技股份有限公司。保留一切权利

Mode Select

Mode selection is determined by the Mode Select pin. The value of this pin is locked 1024 LRCK cycles after RST is released. This pin requires a specific connection to supply, ground, MCLK, or LRCK as outlined in Table 2.

Mode pin is:	Mode	Figure
Tied to VL	I²S	7
Tied to GND	Left-Justified	8
Tied to LRCK	Right-Justified - 24 bit	9
Tied to MCLK	Right-Justified - 16bit	10

Table 2. Mode Pin Settings

图 5-15: CS4361: 模式配置说明

例如：CS4361 的 MODE 引脚与 MCLK 引脚连接时，CODEC 端采用 16bit 的 slot 宽度。DTS 配置如下示例：

```
&i2s0_mach {
  ...
  /* 对齐SoC端与Codec端的slot宽度 */
  soundcard-mach,slot-width     = <16>;
  ...
};
```

5.4.1.2 通过寄存器方式设置 slot 宽度

Reg 35h_I2S Format Configure 2 Register

Default: 0x55			Register Name: I2S_FMT_CTRL2
Bit	Read/Write	Default	Description
7	/	/	/
6:4	R/W	0x5	SW Slot Width Select 0: Reserved 1: Reserved 2: Reserved 3: 16-bit 4: 20-bit 5: 24-bit 6: 28-bit 7: 32-bit

图 5-16: TD104:slot width 相关寄存器配置说明

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
/*8/12/16/20/24/28/32bit Slot Width */
td100_update_bits(SUNXI_I2S_FMT_CTRL2, 0x7 << SW,
        (TD100_SLOT_WIDTH / 4 - 1) << SW);
```

图 5-17: TD104: 驱动设置 slot 宽度代码段

例如：TD104 通过 0x55 寄存器的 SW[bit 6:4] 控制 slot 宽度，CODEC 原厂驱动将 SW 配置为’ TD100_SLOT_WIDTH / 4 - 1’ ，slot 宽度为 TD100_SLOT_WIDTH。DTS 配置如下示例：

```
&i2s0_mach {
  ...
  /* 对齐SoC端与Codec端的slot宽度,假设TD100_SLOT_WIDTH为32 */
  soundcard-mach,slot-width      = <32>;
  ...
};
```

Reg 35h_I2S Format Configure 2 Register

Default: 0x55			Register Name: I2S_FMT_CTRL2
Bit	Read/Write	Default	Description
7	/	/	/
6:4	R/W	0x5	SW Slot Width Select 0: Reserved 1: Reserved 2: Reserved 3: 16-bit 4: 20-bit 5: 24-bit 6: 28-bit 7: 32-bit

图 5-18: BP1048B2: 音频 spec

例如：BP1048B2 的音频 spec 中明确说明最大有效位宽为 32bit，整篇手册未提及位宽可调节，说明其 slot 宽度为固定的 32bit。DTS 配置如下示例：

```
&i2s0_mach {
  ...
  /* 对齐SoC端与Codec端的slot宽度 */
  soundcard-mach,slot-width      = <32>;
  ...
};
```

5.4.2 从示波器时序图中获取

在 CODEC 做主的前提下，测量 LRCK 与 BCLK 的频率，并且在已知 slot 个数前提下，可反推 slot 宽度：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

slot 宽度 = BCLK 频率 / LRCK 频率 / slot 个数。

5.5 slot 个数

5.5.1 从芯片手册中获取

搜索关键词 (不区分大小写)：Fs, LRCK, SCLK, slot，LRCK period…等。I2S 协议中 BCLK 频率的计算公式为：

```
BCLK 频率 = LRCK 频率 * slot 宽度 * slot 个数;
LRCK 频率 = 音频采样率(Fs)。
```

可以通过如上公式反推 slot 个数，如：从 LRCK period 或 BCLK 频率反推 slot 个数，示例如下：

5.5.1.1 通过硬件方式设置 slot 个数

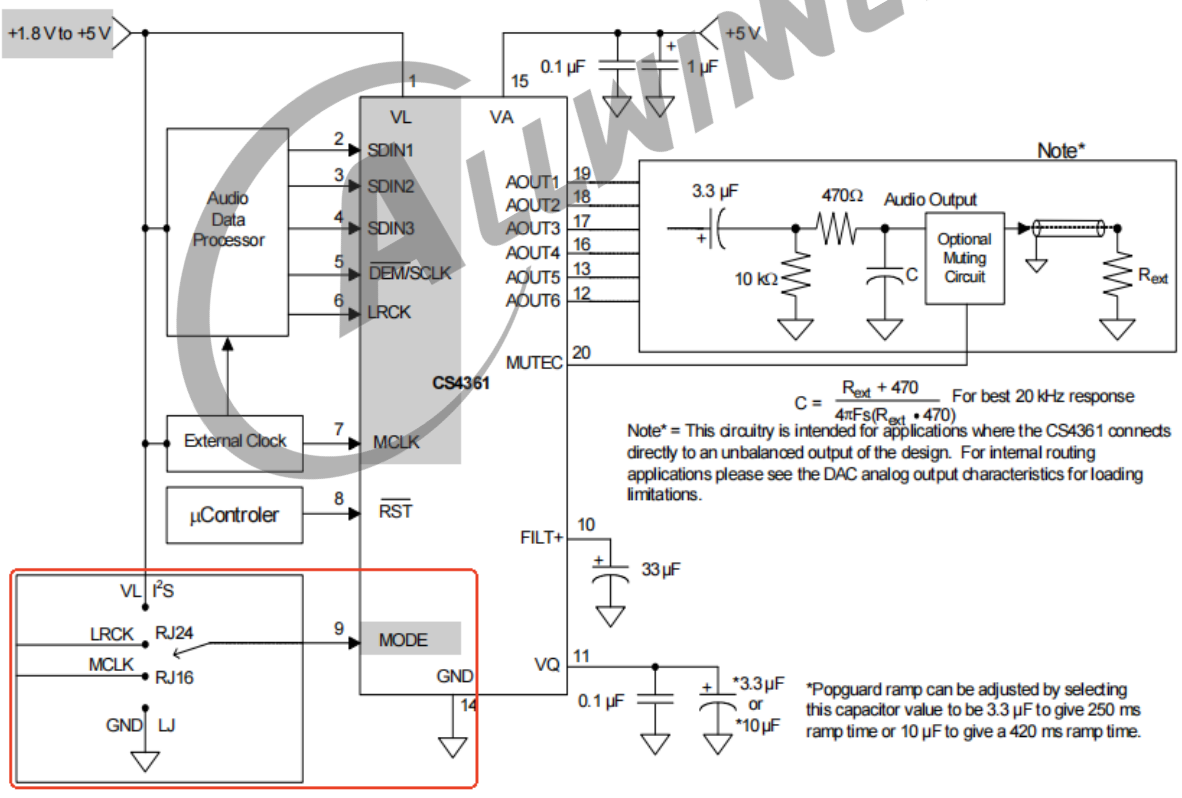
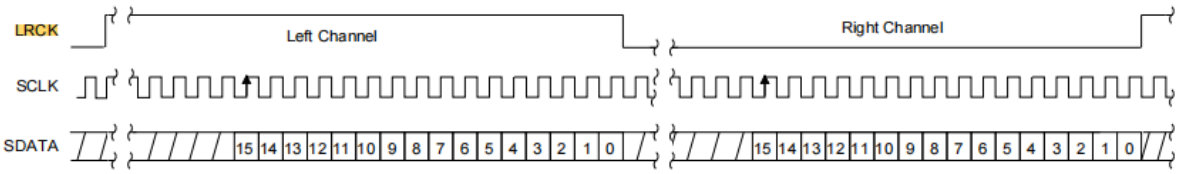


图 5-19: CS4361: 硬件电路

版权所有 © 珠海全志科技股份有限公司。保留一切权利



Internal SCLK Mode	External SCLK Mode
Right-Justified, 16-Bit Data INT SCLK = 32 Fs if MCLK/LRCK = 1024, 512, 256, 128, or 64 INT SCLK = 48 Fs if MCLK/LRCK = 768, 384, 192, or 96 INT SCLK = 72 Fs if MCLK/LRCK = 1152	Right-Justified, 16-Bit Data Data Valid on Rising Edge of SCLK SCLK Must Have at Least 32 Cycles per LRCK Period

Figure 10. CS4361 Data Format (Right-Justified 16)

图 5-20: CS4361:slots 个数配置说明

例如：CS4361 采用内部时钟 (master 模式) 和右对齐模式时，当 MCLK/LRCK=1024，512，256，128，or 64，则 SCLK(又称 BCLK) 为 32 Fs，而此时 slot 宽度为 16，故 slot 个数为 2。DTS 配置如下示例：

```
&i2s0_mach {
    ...
    /* 对齐SoC端与Codec端的slot个数 */
    soundcard-mach,slot-num     = <2>;
    ...
};
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

5.5.1.2 通过寄存器方式设置 slot 个数

1:0	R/W	0x0	LRCK_PERIODH The 2-High bit of LRCK period value. It is used to program the number of BCLKs per channel of sample frame. This value is interpreted as follow: PCM mode: Number of BCLKs within (Left + Right) channel width I2S / Left-Justified / Right-Justified mode: Number of BCLKs within each individual channel width (Left or Right) N+1 For example: n = 7: 8 BCLK width … n = 1023: 1024 BCLKs width This bit is must be configured in master or slave mode

Reg 33h_I2S LRCK Configure 2 Register

Default: 0x00			Register Name: I2S_LRCK_CTRL2
Bit	**Read/Write**	**Default**	**Description**
7:0	R/W	0x0	LRCK_PERIODL The 8-Low bit of LRCK period value. It is used to program the number of BCLKs per channel of sample frame. This value is interpreted as follow: PCM mode: Number of BCLKs within (Left + Right) channel width I2S / Left-Justified / Right-Justified mode: Number of BCLKs within each individual channel width (Left or Right) N+1 For example: n = 7: 8 BCLK width … n = 1023: 1024 BCLKs width This bit is must be configured in master or slave mode

图 5-21: TD104:LRCK period 相关寄存器配置说明

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
/*
 * config LRCK period:
 * 16bit * 8ch = 128,
 * 32bit * 8ch = 256,
 * 32bit *16ch = 512
 */
/*config LRCK period */
td100_update_bits(SUNXI_I2S_LRCK_CTRL1, 0x3 << LRCK_PERIODH,
          ((TD100_LRCK_PERIOD - 1) >> 8) << LRCK_PERIODH);
td100_write(SUNXI_I2S_LRCK_CTRL2, (u8) (TD100_LRCK_PERIOD - 1) & 0xFF);

td100_update_bits(SUNXI_I2S_FMT_CTRL1,
    0x1 << TX_SLOT_HIZ | 0x1 << TX_STATE,
    0x0 << TX_SLOT_HIZ | 0x0 << TX_STATE);

/*8/12/16/20/24/28/32bit Slot Width */
td100_update_bits(SUNXI_I2S_FMT_CTRL2, 0x7 << SW,
        (TD100_SLOT_WIDTH / 4 - 1) << SW);
```

图 5-22: TD104: 驱动设置 lrck period 代码段

例如：TD104 通过 0x1033 寄存器的 LRCK_PERIODL[bit 7:0] 和 0x1032 寄存器的 LRCK_PERIODH[bit 1:0] 控制 lrck period，CODEC 原厂驱动将 lrck period 配置为’TD100_LRCK_PERIOD’，并将 slot_width 配置为’TD100_SLOT_WIDTH’，故 slot 个数为’TD100_LRCK_PERIOD / TD100_SLOT_WIDTH’。DTS 配置如下示例：

```
&i2s0_mach {
  ...
  /* 对齐SoC端与Codec端的slot个数，假设:TD100_LRCK_PERIOD / TD100_SLOT_WIDTH = 8 */
  soundcard-mach,slot-num    = <8>;
  ...
};
```

5.5.2 从示波器时序图中获取

在 CODEC 做主的前提下，测量 LRCK 频率与 BCLK 频率，并且在已知 slot 宽度前提下，可反推 slot 个数：

> slot 个数 = BCLK 频率 / LRCK 频率 / slot 宽度。

5.6 MCLK

- 部分 CODEC 在某些场景下需要 SoC 提供 MCLK(syncclk or master clock)，此时需要使能 MCLK 输出。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

5.6.1 从芯片手册中获取

搜索关键词 (不区分大小写)：mclk, sysclk, internal, external…等。

7.3.3 Clock

The system clock(SYSCLK) of AC107 must be 128*fs(fs=48KHz or 44.1KHz). So the system should arrange the divider to generate 12.288MHz for audio clock series of 48KHz or 11.2896MHz for series of 44.1KHz.

SYSCLK can be selected from MCLK or I2S_BCLK pin which always provided externally or internal PLL while the PLL reference clock can be select from MCLK or I2S_BCLK or PDMCLK. SYSCLK is the clock reference of ADC, DVC, MIXER, HPF and I2S module except TWI. If MCLK is not 12.288MHz for ADC 48KHz series sample rate or 11.2896MHz for ADC 44.1KHz series sample rate, SYSCLK must be set by PLL. SYSCLK need always to be configured in these cases.

图 5-23: AC107:SYSCLK 时钟介绍

版权所有 © 珠海全志科技股份有限公司。保留一切权利

Reg 20h_System Clock Control Register

Default: 0x00	Register Name: SYSCLK_CTRL

Revision 1.0 Copyright © 2018 X-Powers Limited. All Rights Reserved. 23

Bit	Read/Write	Default	Description
7	R/W	0x0	PLLCLK_EN PLLCLK Enable 0: Disable 1: Enable
6	/	/	/
5:4	R/W	0x0	PLLCLK_SRC PLL Clock Source Select 0: MCLK 1: BCLK 2: PDMCLK 3: Reserved
3:2	R/W	0x0	SYSCLK_SRC System Clock Source Select 0: MCLK 1: BCLK 2: PLL 3: Reserved
1	/	/	/
0	R/W	0x0	SYSCLK_EN SYSCLK Enable 0: Disable 1: Enable

图 5-24: AC107:MCLK 相关寄存器配置说明

```
switch (pdata->sysclk_src) {
case SYSCLK_SRC_MCLK:
    regmap_update_bits(regmap, SYSCLK_CTRL, 0x3 << SYSCLK_SRC, 0x0 << SYSCLK_SRC);
    break;
case SYSCLK_SRC_BCLK:
    regmap_update_bits(regmap, SYSCLK_CTRL, 0x3 << SYSCLK_SRC, 0x1 << SYSCLK_SRC);
    break;
case SYSCLK_SRC_PLL:
    regmap_update_bits(regmap, SYSCLK_CTRL, 0x3 << SYSCLK_SRC, 0x2 << SYSCLK_SRC);
    break;
default:
    dev_err(dai->dev, "ac107 sysclk source config error: %d\n", pdata->sysclk_src);
    return -EINVAL;
}
```

图 5-25: AC107: 驱动设置 MCLK 代码段

版权所有 © 珠海全志科技股份有限公司。保留一切权利

例如：

1. 从手册介绍上，AC107 的系统时钟（SYSCLK）必须是 128 fs（fs=48KHz 或 44.1KHz）。因此在采用 48kHz 系列采样率时，sysclk 频率应为 12.288MHz，在采用 44.1kHz 系列采样率时，sysclk 频率应为 11.2896MHz；

2. 根据此信息，MCLK 设备树配置如下所示，AW I2S 驱动会根据不同采样率将 MCLK 频率分别设置为 ‘(12288000 * 1)Hz’ 或 ‘(11289600 * 1)Hz’；

3. 从寄存器配置上看，AC107 的系统时钟源可选择为 MCLK, BCLK, or PLL, 当使用 MCLK 作为 AC107 SYSCLK 时钟源时，需要将 0x20 寄存器的 SYSCLK_SRC[bit 3:2] 设置为 0。

DTS 配置如下示例：

```
&i2s0_mach {
  ...
  i2s0_cpu: soundcard-mach,cpu {
    ...
    /* 提供CODEC所需指定频率的MCLK */
    soundcard-mach,mclk-fs  = <1>;
    soundcard-mach,mclk-fp  = <11289600 12288000>;
  }
  ...
};
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

6 FAQ

6.1 无任何相关 log 打印

解决方法

1. 是否有加载模块；
2. 是否有正确编译音频相关驱动；
3. 设备树是否将相关节点的 status 属性设置未 “okay”；
4. 音频驱动依赖模块是否已成功加载（如 dma 驱动等）。

6.2 引脚冲突

关键 log 示例

```
pinctrl: pin PA0 already requested by 5002000.twi; cannot claim for 5002c00.twi
pinctrl: pin-0 (5002c00.twi) status -22
pinctrl: could not request pin 0 (PA0) from group PA0  on device 300b000.pinctrl
```

解决方法

将其它模块的冲突引脚注释。

6.3 TWI 通信失败

关键 log 示例

```
sunxi-i2c sunxi-i2c1: SLA+W has been transmitted; ACK not received
sunxi-i2c sunxi-i2c1: engine mode: I2C BUS error state is 0x20
sunxi-i2c sunxi-i2c1: engine-mode: xfer failed(dev addr:0x36)
```

解决方法

1. 外部 codec 是否正常供电；
2. twi 信号是否有上拉；
3. twi 地址是否正确配置。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

6.4 录音数据为 0 数据

解决方法

1. 是否正确焊接录音器件，且正常供电；
2. 是否开启 CODEC 调试模式；
3. I2S 数据线是否正确配置。

6.5 播放或录音速度异常

解决方法

确认 I2S 模块父时钟是否不准确。

6.6 播放无声音

解决方法

1. 是否正确接播放设备，且正常供电；
2. I2S 数据线是否正确配置。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

7 附录

7.1 I2S 格式协议图 (普通模式)

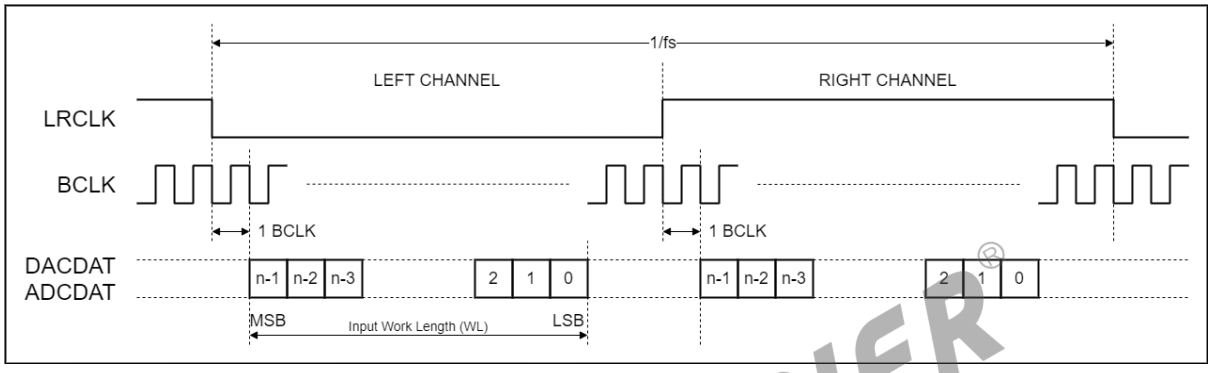


图 7-1: I2S 标准格式

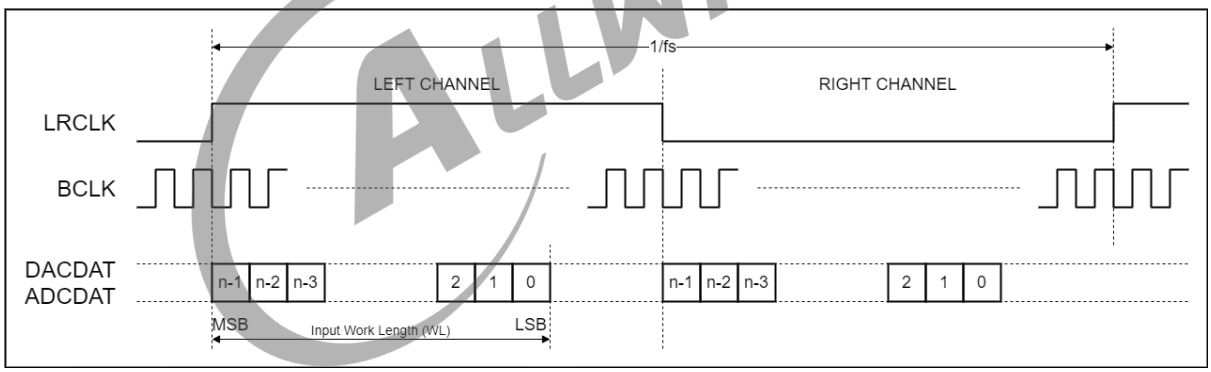


图 7-2: I2S 左对齐格式

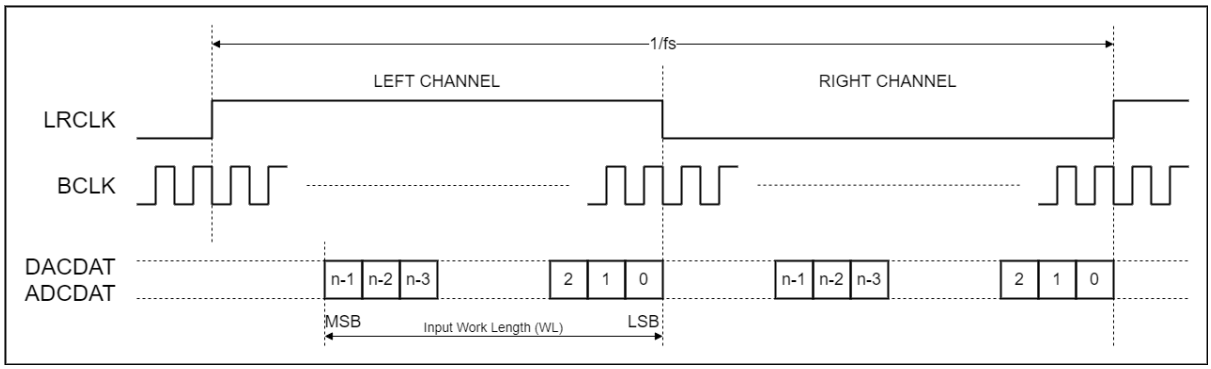


图 7-3: I2S 右对齐格式

版权所有 © 珠海全志科技股份有限公司。保留一切权利

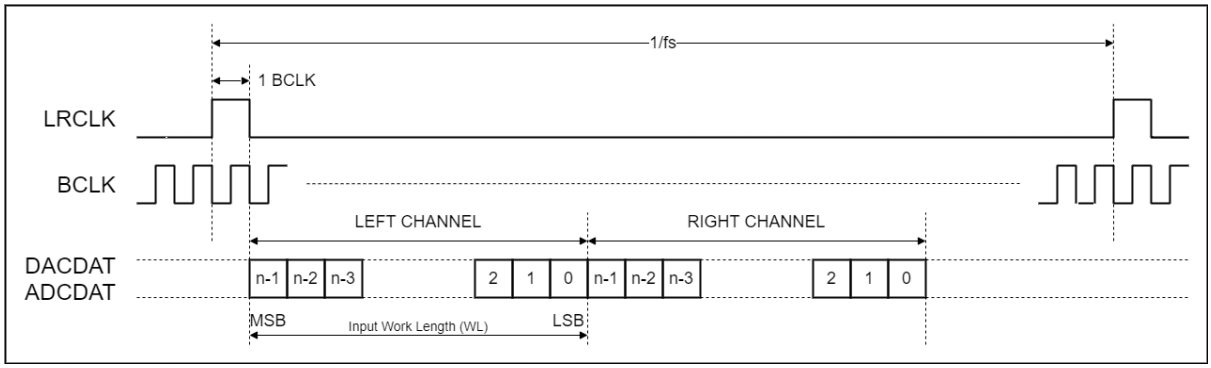


图 7-4: I2S DSP_A/PCM LATE1 SHORT FRAME

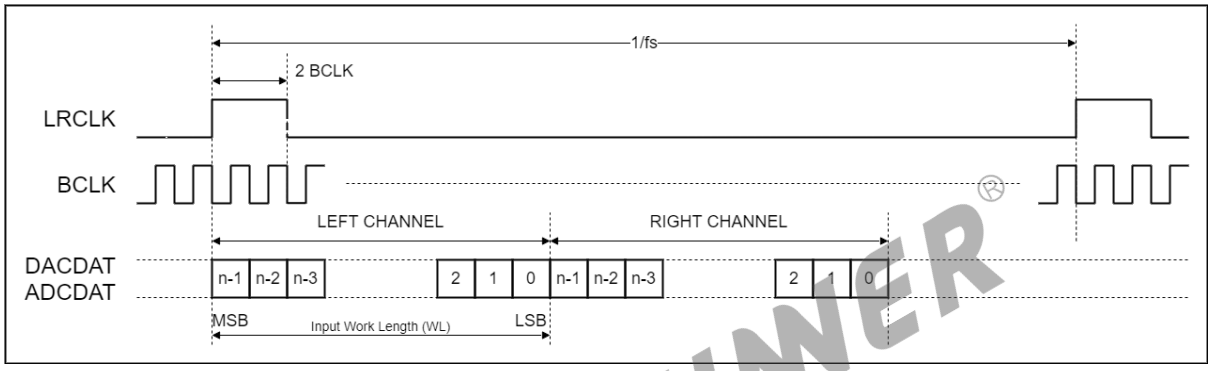


图 7-5: I2S DSP_B/PCM EARLY LONG FRAME

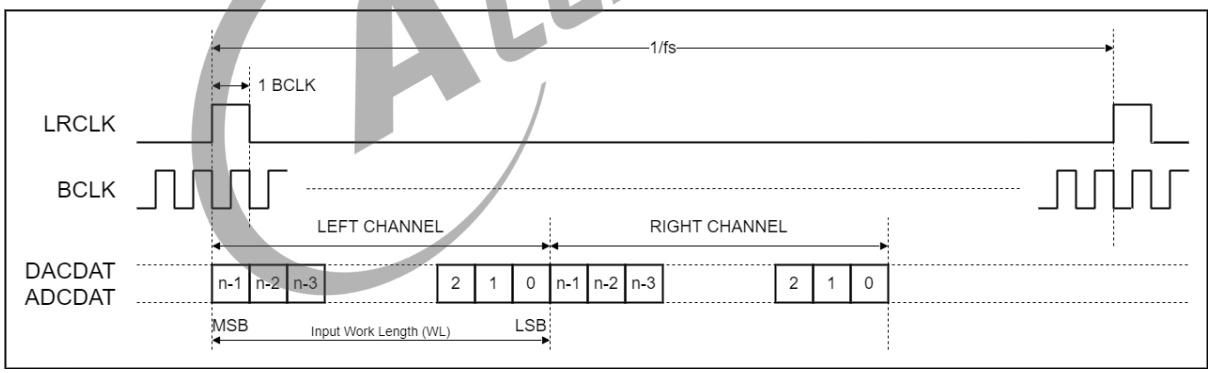


图 7-6: I2S DSP_A_EARLY/PCM EARLY SHORT FRAME

版权所有 © 珠海全志科技股份有限公司。保留一切权利

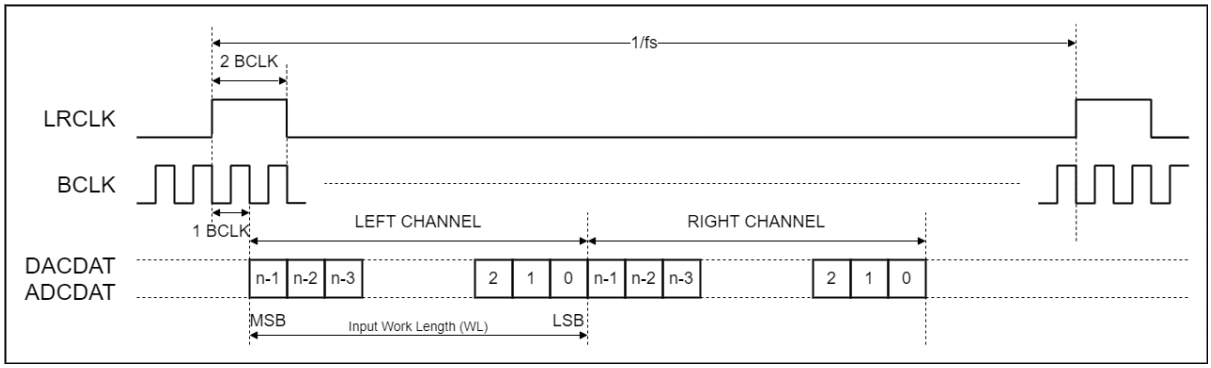


图 7-7: I2S DSP_B_LATE1/PCM LATE1 LONG FRAME

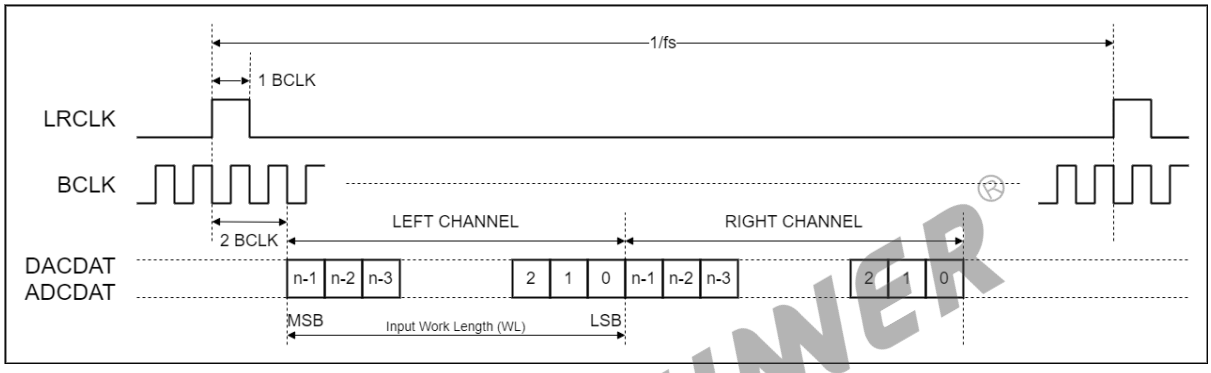


图 7-8: I2S DSP_A_LATE2/I2S PCM LATE2 SHORT FRAME

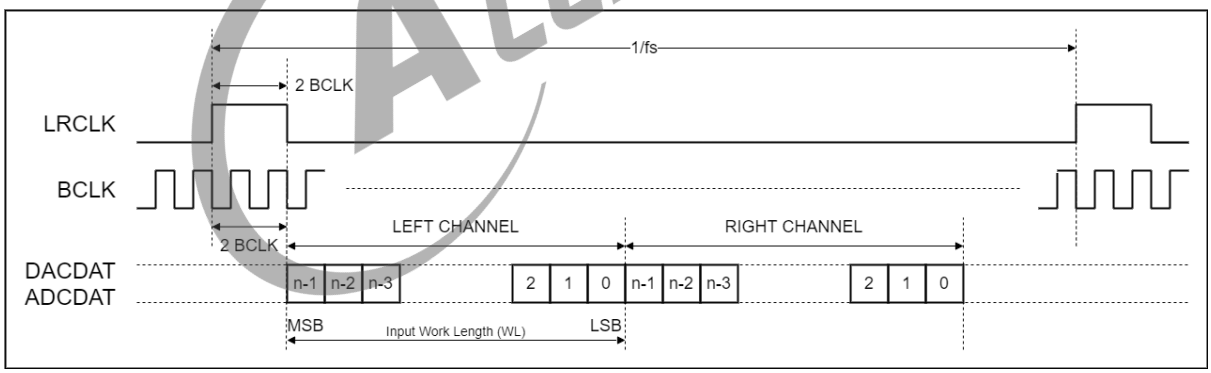


图 7-9: I2S DSP_B_LATE2/PCM LATE2 LONG FRAME

版权所有 © 珠海全志科技股份有限公司。保留一切权利

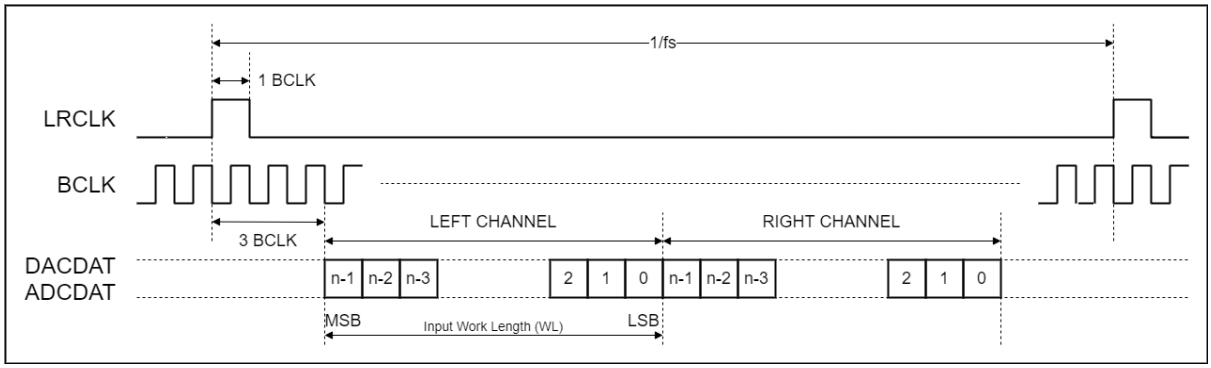


图 7-10: I2S DSP_A_LATE3/I2S PCM LATE3 SHORT FRAME

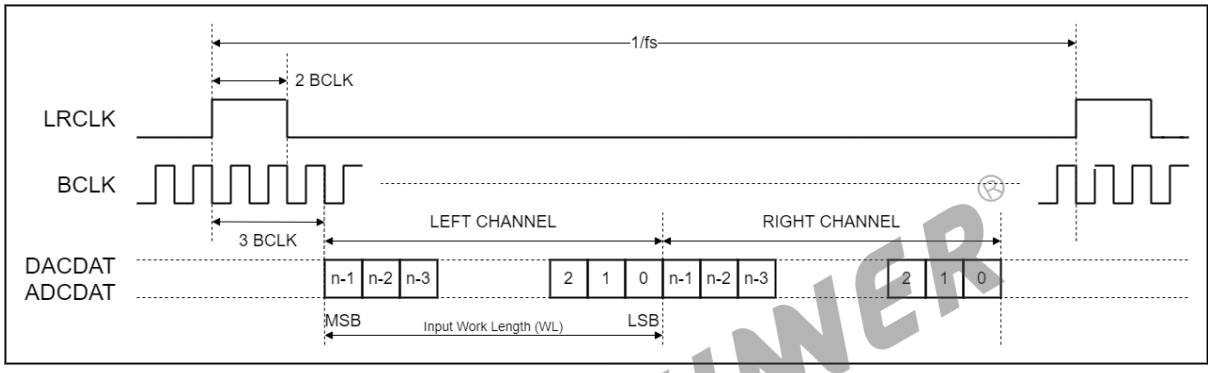


图 7-11: I2S DSP_B_LATE3/PCM LATE3 LONG FRAME

7.2 I2S 格式协议图 (TDM 模式)

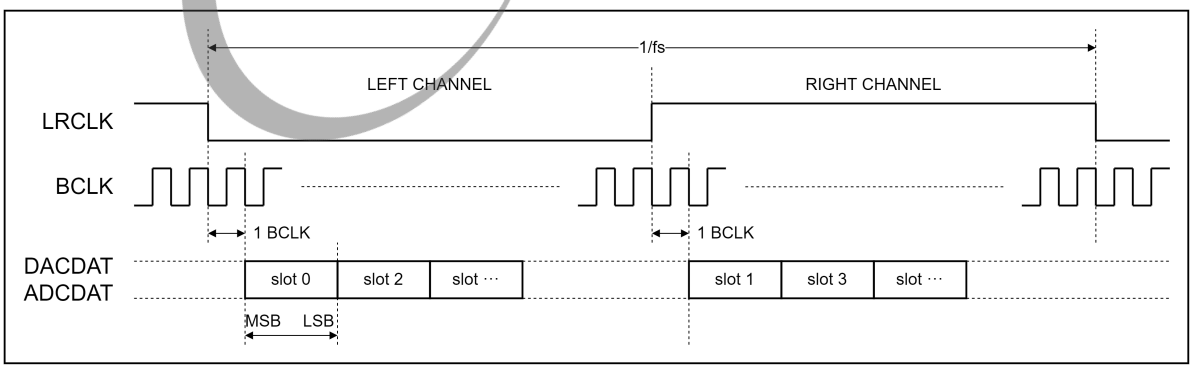


图 7-12: I2S 标准格式-TDM 模式

版权所有 © 珠海全志科技股份有限公司。保留一切权利

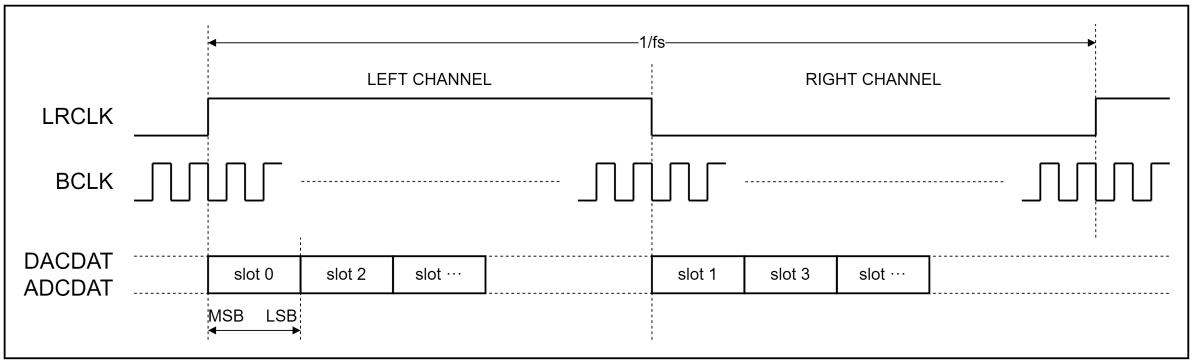


图 7-13: I2S 左对齐格式-TDM 模式

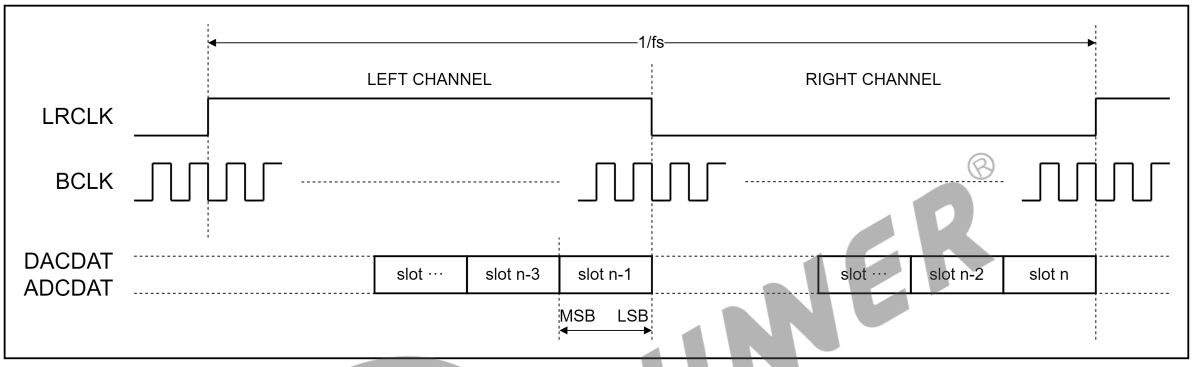


图 7-14: I2S 右对齐格式-TDM 模式

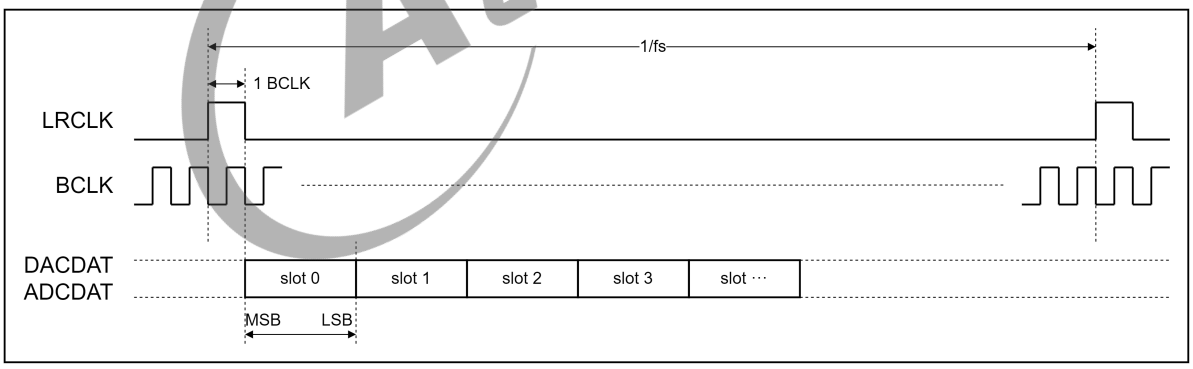


图 7-15: I2S DSP_A-TDM 模式/PCM LATE1 SHORT FRAME-TDM 模式

版权所有 © 珠海全志科技股份有限公司。保留一切权利

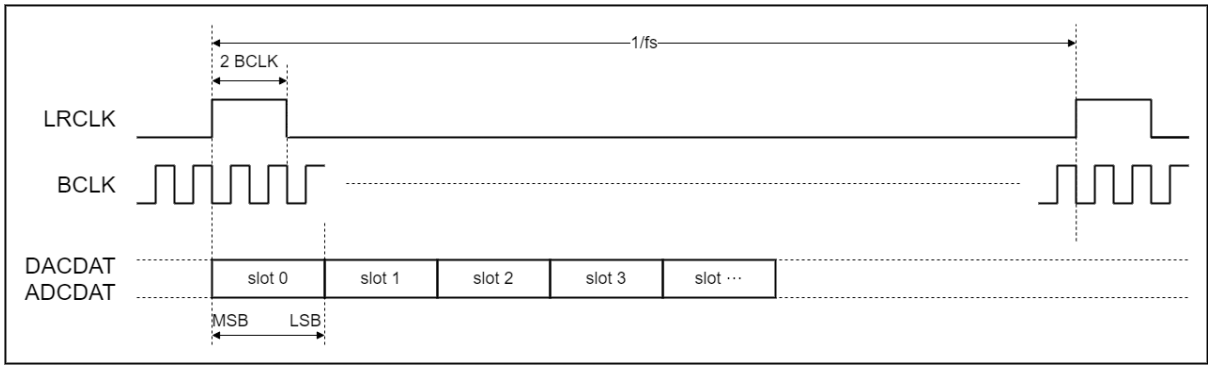


图 7-16: I2S DSP_B-TDM 模式/PCM EARLY LONG FRAME-TDM 模式

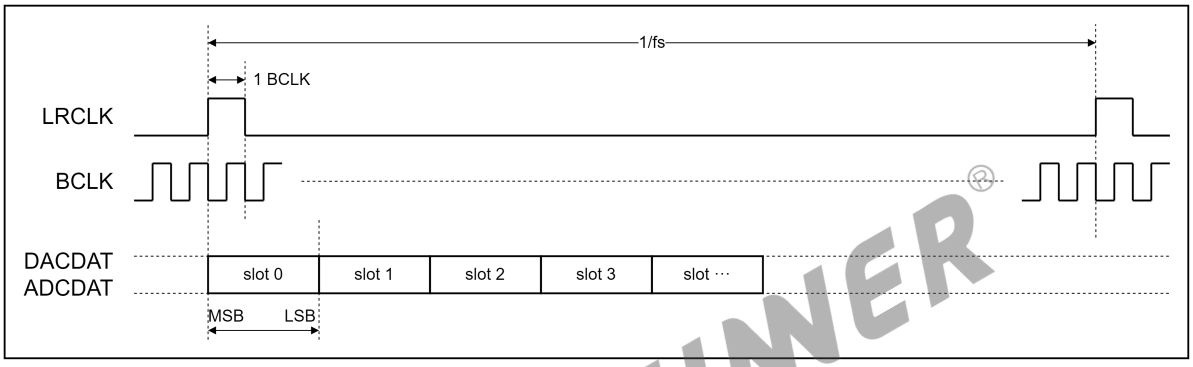


图 7-17: I2S DSP_A_EARLY-TDM 模式/PCM EARLY SHORT FRAME-TDM 模式

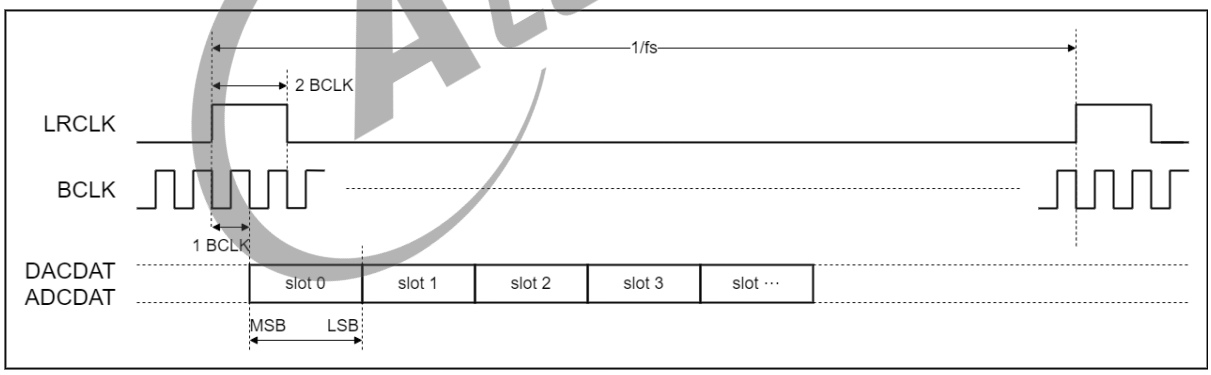


图 7-18: I2S DSP_B_LATE1-TDM 模式/PCM LATE1 LONG FRAME-TDM 模式

版权所有 © 珠海全志科技股份有限公司。保留一切权利

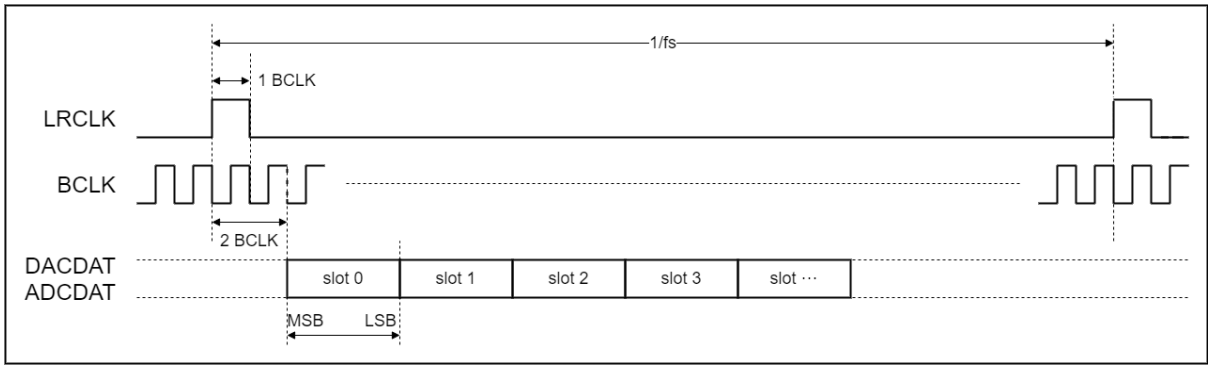


图 7-19: I2S DSP_A_LATE2-TDM 模式/I2S PCM LATE2 SHORT FRAME-TDM 模式

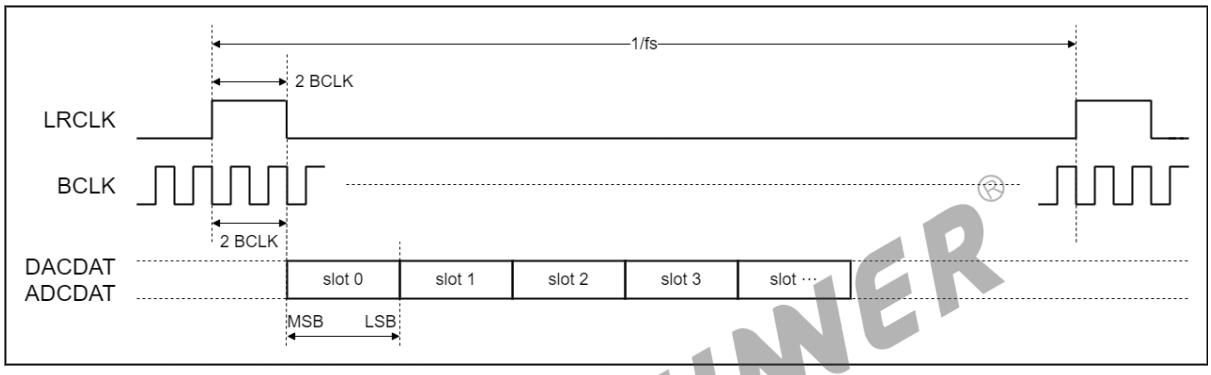


图 7-20: I2S DSP_B_LATE2-TDM 模式/PCM LATE2 LONG FRAME-TDM 模式

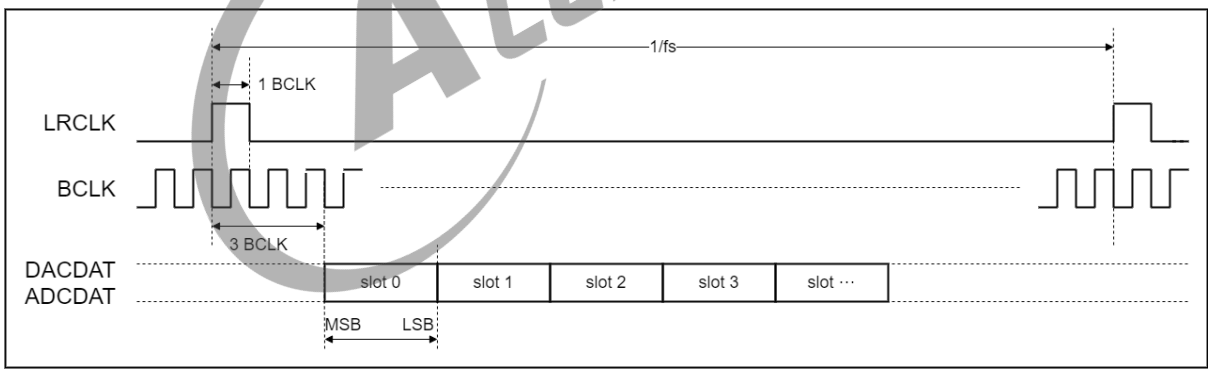


图 7-21: I2S DSP_A_LATE3-TDM 模式/I2S PCM SHORT FRAME-TDM 模式

版权所有 © 珠海全志科技股份有限公司。保留一切权利

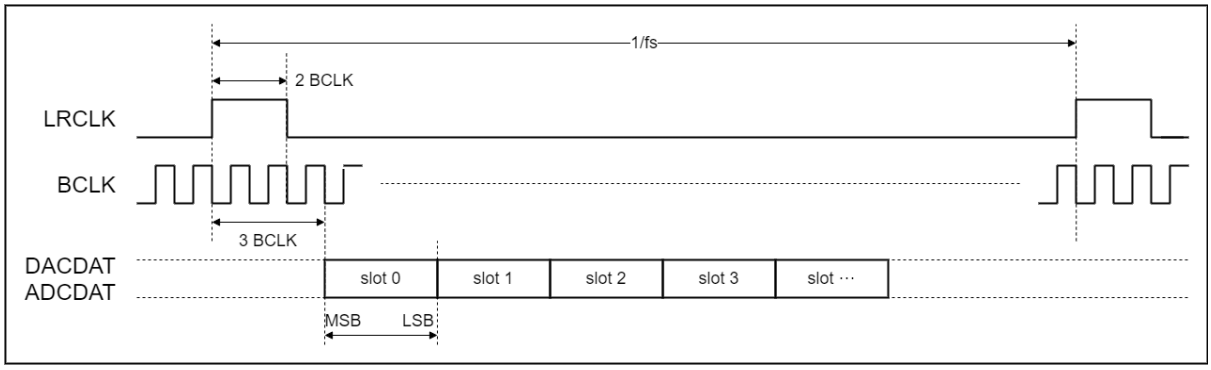


图 7-22: I2S DSP_B_LATE3-TDM 模式/PCM LATE3 LONG FRAME-TDM 模式

版权所有 © 珠海全志科技股份有限公司。保留一切权利

著作权声明

版权所有 ©2025 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。