static u8 lc1160\_codec\_reg[LC1160\_CACHEREGNUM] = {

0x00, /\*R0\*/

0xAD, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x20, 0x72, 0xFA, /\*R01~R10\*/

0x3C, 0x72, 0x1F, 0xE7, 0x00, 0x28, 0x00, 0x32, 0x00, 0x3F, /\*R11~R20\*/

0x67, 0x3F, 0xE7, 0x00, 0x28, 0x00, 0x32, 0x00, 0x3F, 0xFF, /\*R21~R30\*/

0xE0, 0x60, 0x67, 0x67, 0x00, 0x07, 0x23, 0x7F, 0x00, 0x7F, /\*R31~R40\*/

0x00, 0x32, 0x00, 0x02, 0x02, 0x02, 0x02, 0x02, 0x33, 0x00, /\*R41~R50\*/

0x07, 0x23, 0xE7, 0x67, 0x7F, 0x00, 0x7F, 0x00, 0x32, 0x00, /\*R51~R60\*/

0x1F, 0x07, 0x26, 0xD5, 0x8F, 0x82, 0x0A, 0x34, 0x2C, 0x00, /\*R61~R70\*/

0x1B, 0x1B, 0x19, 0x19, 0x07, 0x08, 0x27, 0x00, 0x00, 0x00, /\*R71~R80\*/

0x00, /\*R81 Read Only\*/

0x4c, //\*LDOA15 For Codec,addr:0x3b\* /\*\*/

/\* Jack & Hookswitch,addr:0xa7,0xa8,0xa9,0xaa,0xab\*/

0x00,0x04,0x04,0x04,0x00,

0x00, /\*DBB\_PCM\_SWITCH\*/

0x00, /\*PA\_ENABLE\*/

};

static struct snd\_soc\_codec\_driver soc\_codec\_dev\_lc1160 = {

.probe = lc1160\_probe,

.remove = lc1160\_remove,

.read = lc1160\_read\_reg\_cache,

.write = lc1160\_write,

.set\_bias\_level = lc1160\_set\_bias\_level,

.reg\_cache\_size = sizeof(lc1160\_codec\_reg),

.reg\_word\_size = sizeof(u8),

.reg\_cache\_default = lc1160\_codec\_reg,

.ignore\_pmdown\_time = true,

.controls = lc1160\_snd\_controls,

.num\_controls = ARRAY\_SIZE(lc1160\_snd\_controls),

.dapm\_widgets = lc1160\_dapm\_widgets,

.num\_dapm\_widgets = ARRAY\_SIZE(lc1160\_dapm\_widgets),

.dapm\_routes = intercon,

.num\_dapm\_routes = ARRAY\_SIZE(intercon),

};

lc1160\_codec\_probe

--->1:snd\_soc\_register\_codec(&pdev->dev,&soc\_codec\_dev\_lc1160,

lc1160\_dai, ARRAY\_SIZE(lc1160\_dai));

static struct snd\_soc\_dai\_driver lc1160\_dai[] = {

{

.name = "comip\_hifi",

.playback = {

.stream\_name = "Playback",

.channels\_min = 1,

.channels\_max = 2,

.rates = COMIP\_1160\_RATES,

.formats = COMIP\_1160\_FORMATS,

},

.capture = {

.stream\_name = "Capture",

.channels\_min = 1,

.channels\_max = 2,

.rates = COMIP\_1160\_RATES,

.formats = COMIP\_1160\_FORMATS,

},

.ops = &lc1160\_dai\_ops,

},

{

.name = "comip\_voice",

.playback = {

.stream\_name = "VxDL",

.channels\_min = 1,

.channels\_max = 2,

.rates = COMIP\_1160\_RATES,

.formats = COMIP\_1160\_FORMATS,

},

.capture = {

.stream\_name = "VxUL",

.channels\_min = 1,

.channels\_max = 2,

.rates = COMIP\_1160\_RATES,

.formats = COMIP\_1160\_FORMATS,

},

.ops = &lc1160\_dai\_ops,

},

{

.name = "virtual\_codec",

.playback = {

.stream\_name = "Play",

.channels\_min = 1,

.channels\_max = 2,

.rates = COMIP\_1160\_RATES,

.formats = COMIP\_1160\_FORMATS,

},

.capture = {

.stream\_name = "Cap",

.channels\_min = 1,

.channels\_max = 2,

.rates = COMIP\_1160\_RATES,

.formats = COMIP\_1160\_FORMATS,

},

}

};

static struct snd\_soc\_dai\_ops lc1160\_dai\_ops = {

.shutdown = lc1160\_shutdown,

.hw\_params = lc1160\_hw\_params,

.set\_sysclk = lc1160\_set\_dai\_sysclk,

.set\_fmt = lc1160\_set\_dai\_fmt,

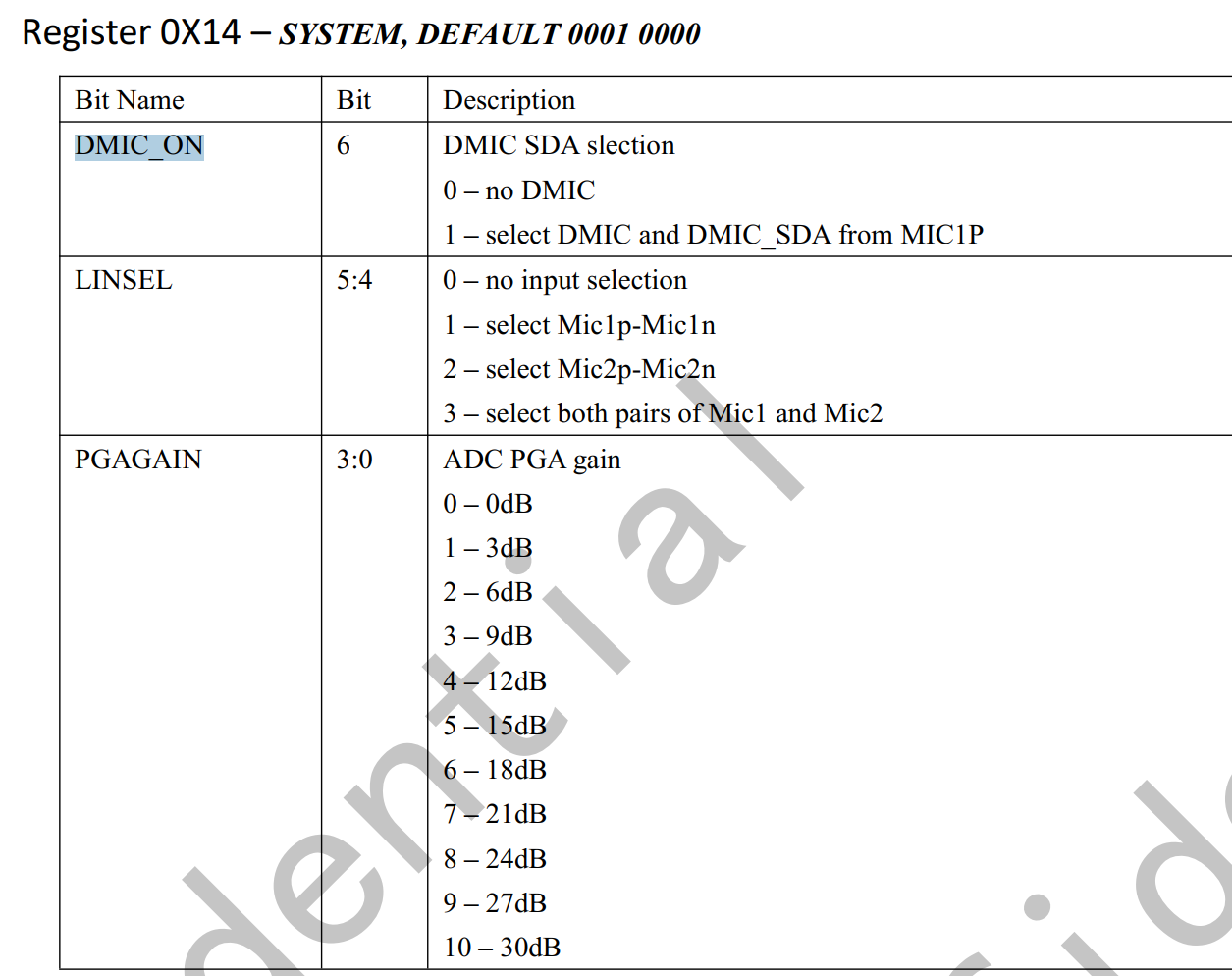
};

tinymix  -D 1  25  150 ==> ok DAC===》 150

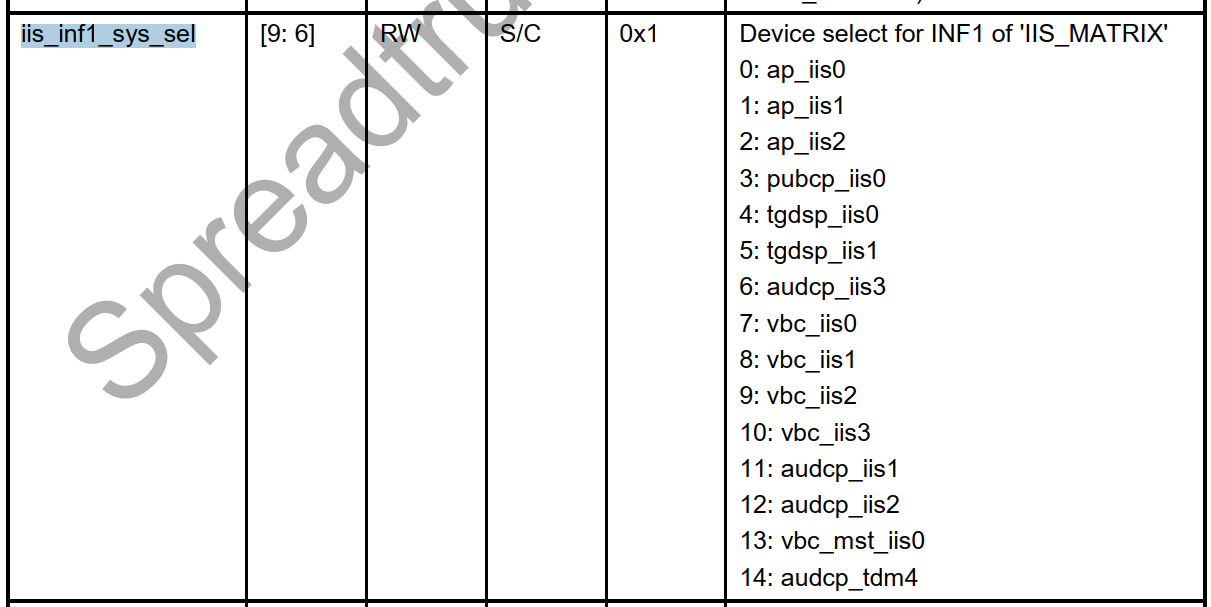
tinymix  -D 1  0 0  ===> ok  adc pga 设置为0

Es8311的回声串绕有改善。

===



====



iis1 默认接通ap iis1

添加iis1:

                                         <&g12\_pll CLK\_TWPLL\_153M6>;

                        };

+

+                       i2s1: i2s@70d00000 {

+                               compatible = "sprd,i2s";

+                               reg = <0 0x70d00000 0 0x1000>;

+                               sprd,dai\_name = "i2s\_bt\_sco1";

+                               sprd,hw\_port = <1>;

+                               sprd,syscon-ap-apb = <&ap\_apb\_regs>;

+                               #sound-dai-cells = <0>;

+                               status = "disable";

+                               clock-names = "clk\_iis1",

+                                             "clk\_twpll\_128m",

+                                             "clk\_twpll\_153m6";

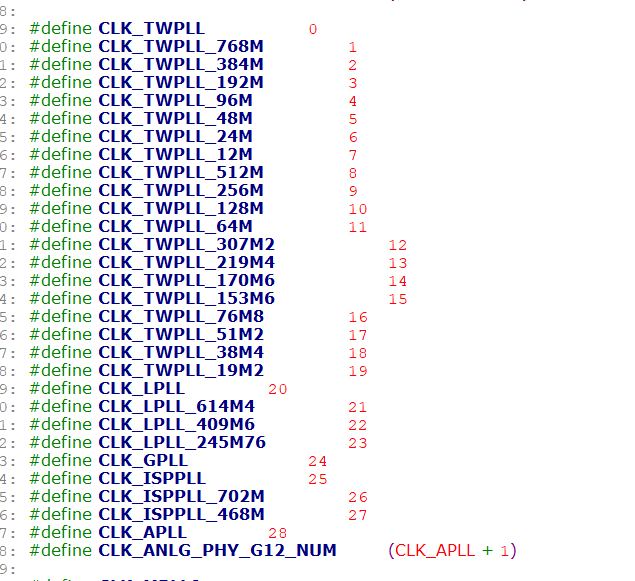
+                               clocks = <&ap\_clk CLK\_AP\_IIS1>,

+                                        <&g12\_pll CLK\_TWPLL\_128M>,

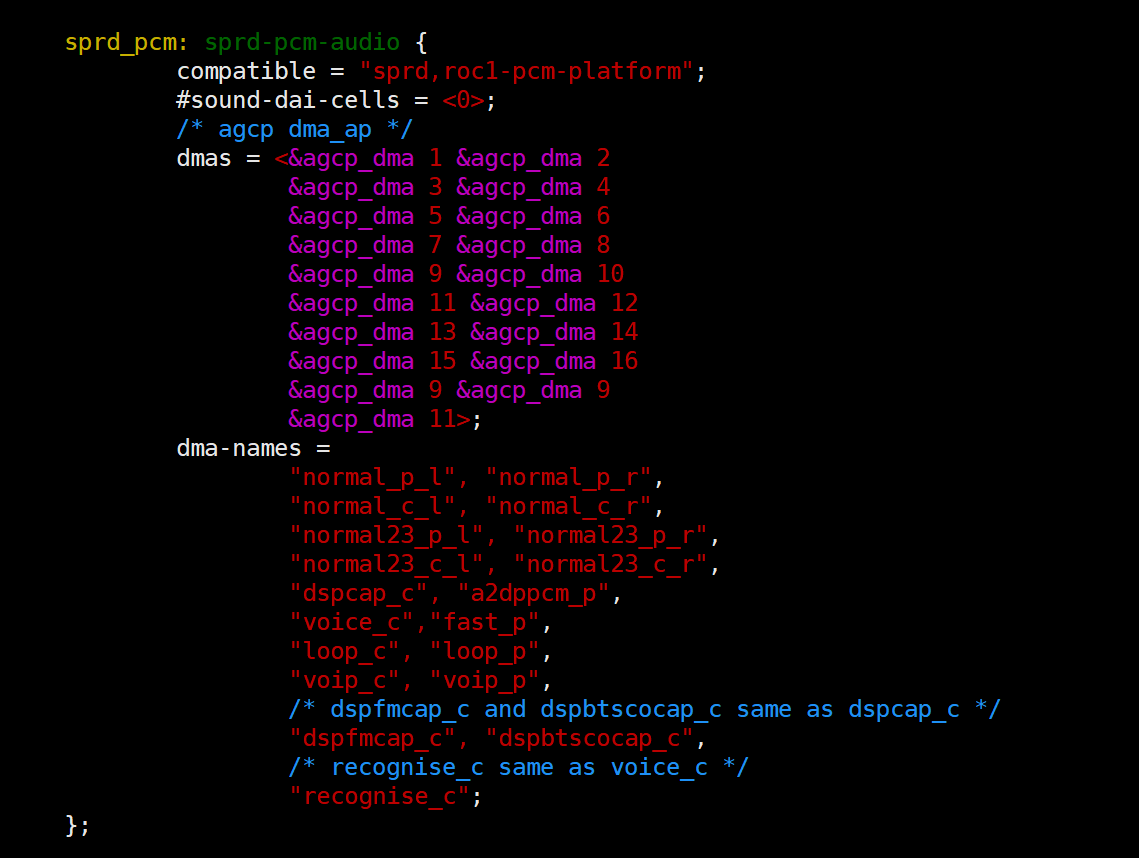
+                                        <&g12\_pll CLK\_TWPLL\_153M6>;

+                       };

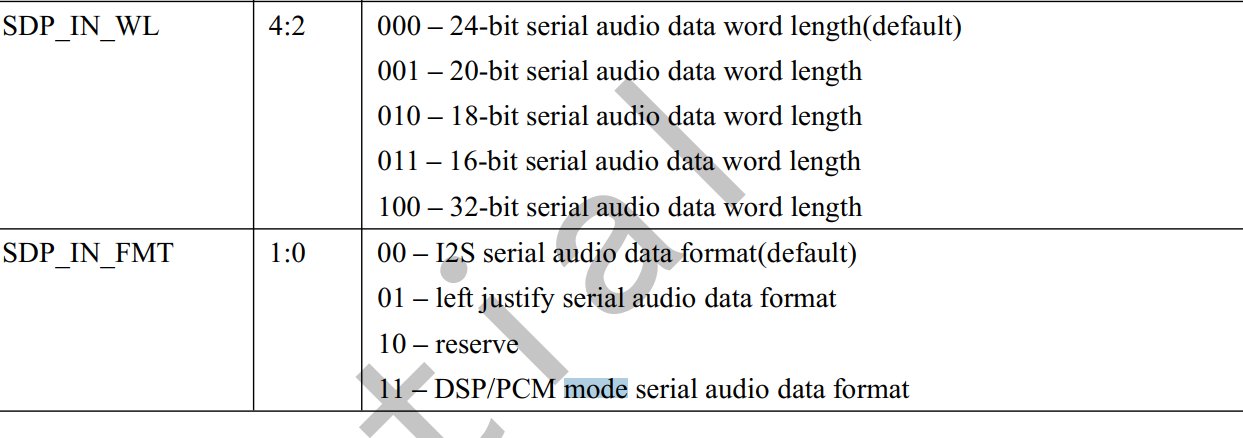
====

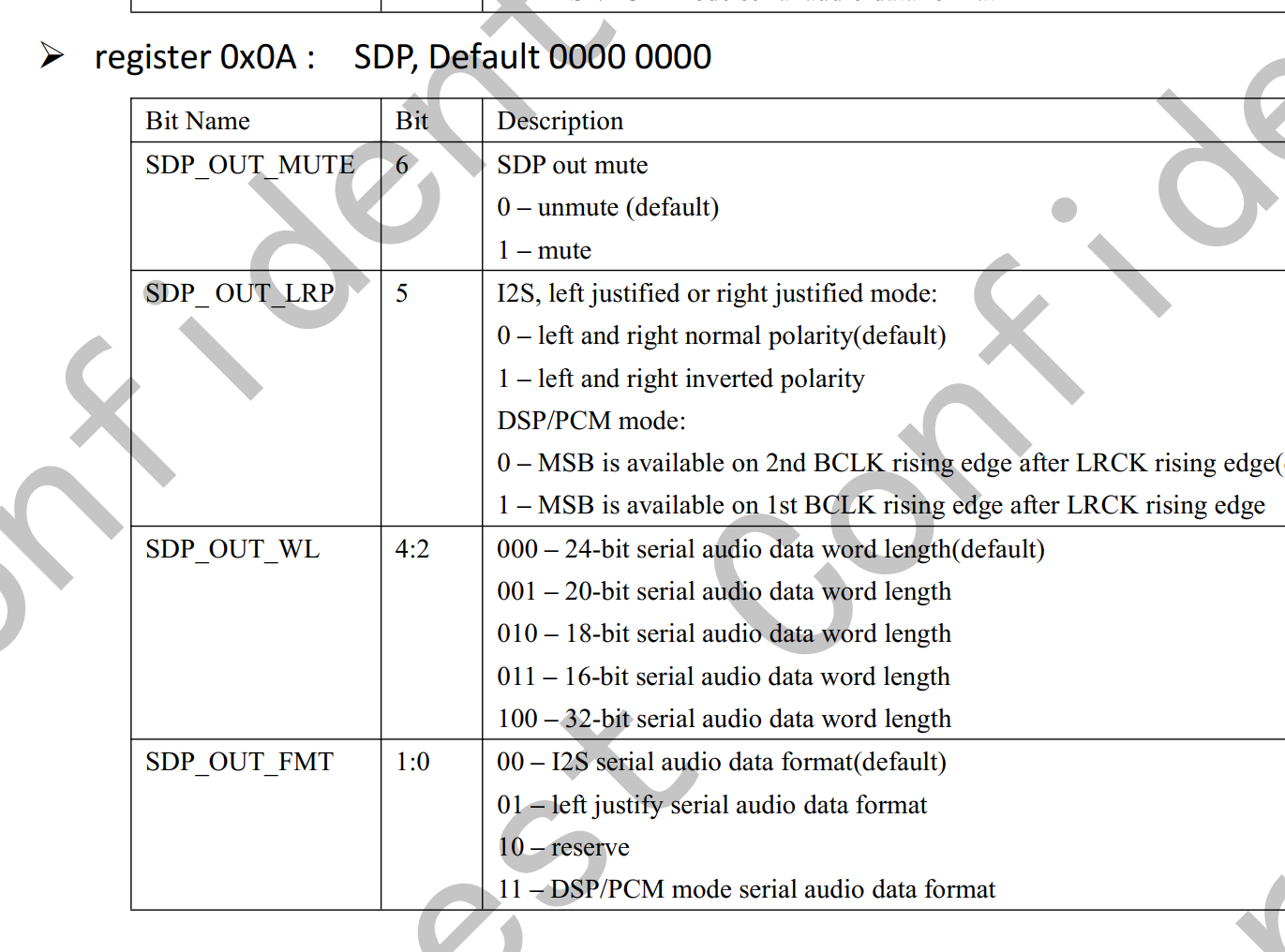


pcm  audio

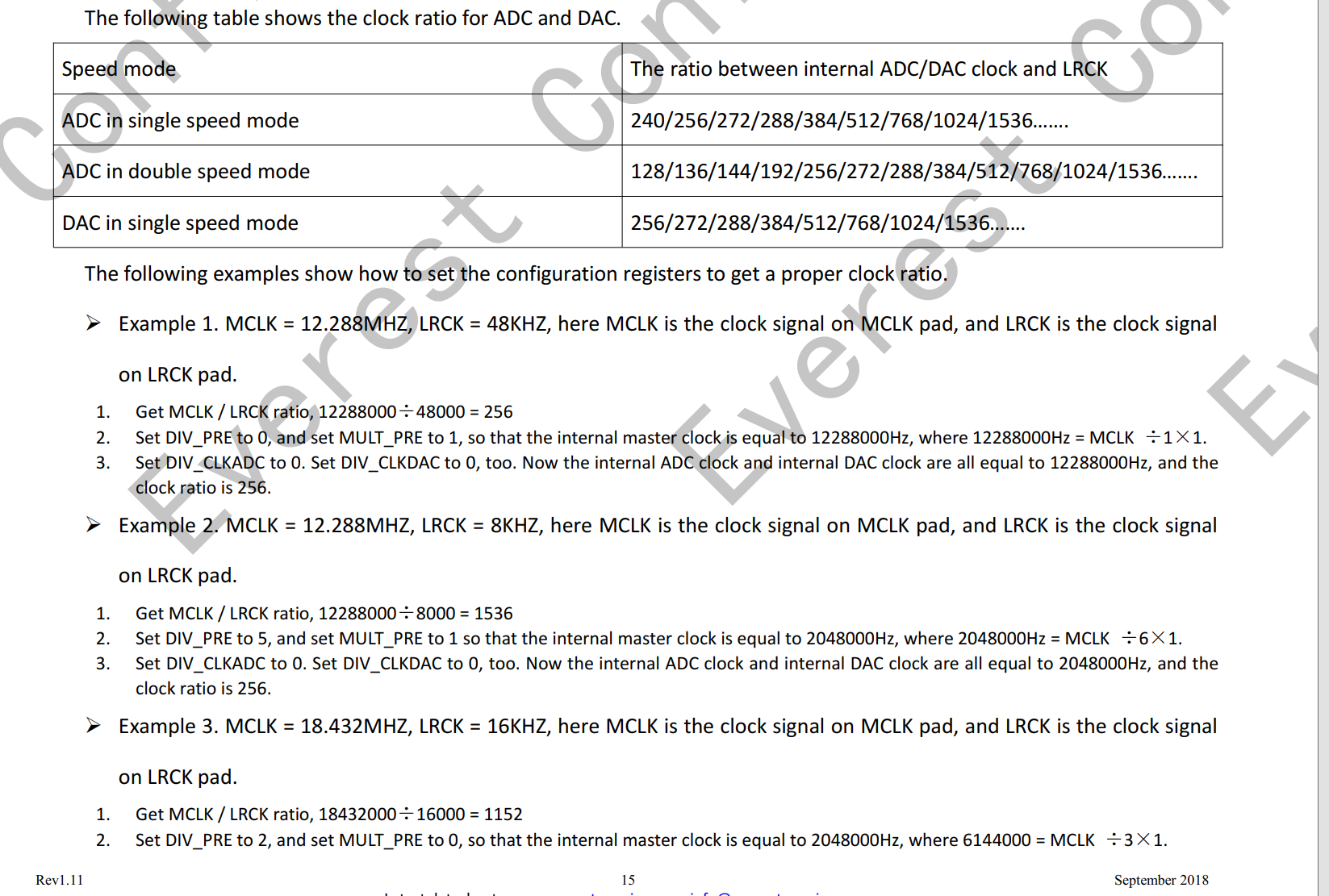


模式选择：





时钟设置：



==

&i2s0 {

        status = "okay";

        sprd,config\_type = "i2s";//i2s和pcm

        sprd,slave\_timeout = <0xf11>;//slave模式使用  超时时间长度

        sprd,\_hw\_port = <0>;// i2s hardware port number

        sprd,fs = <48000>;//采用率

        sprd,bus\_type = <0>;//总线模式 0：i2s 1:pcm

        sprd,rtx\_mode = <3>;//0 idle 1:只接收 2：只发送  3：双工收发

        sprd,byte\_per\_chan = <1>;//每个i2s通道有多少个byte  0 8bit, 1 16 bit, 2 32 bit, 3 32 bit

        sprd,slave\_mode = <0>;//0 master, 1 slave

        sprd,lsb = <0>;//0 :msb is first bit to transmit, 1 lsb is first bit to transmit

        sprd,lrck = <1>; /\*sync\_mode\*/ /0 output lrck, 1 sync. 0:输出的是lrck的信号  1：长帧信号

        sprd,low\_for\_left = <0>; /\*lrck\_inv\*/ //0 low\_for\_left, 1 high\_for\_left 左声道的位置---->默认为1

        sprd,clk\_inv = <1>;//0 not invert, 1 invert 时钟是否反正

        sprd,pcm\_short\_frame = <0>; /\*pcm\_bus\_mode\*/  0:I2S\_LONG\_FRAME     1:I2S\_SHORT\_FRAME

        sprd,pcm\_slot = <0x1>;//1：一个i2s通道  最多3个 pcm\_slot means channel count, support 3 slots. i2s 通道的个数

        sprd,pcm\_cycle = <1>;//slots use how many cycle, for exsample 8bit for each slot, 3 slot, 4 cycle. There will 4 8bit output, 3 8bit is valid, 1 8bit is dymmy.

        sprd,tx\_watermark = <12>;//iis tx watermark for dma request or interrupt, not large than 32

        sprd,rx\_watermark = <20>;//iis rx watermark for dma request or interrupt, not large than 32

        sprd,i2s\_compatible= <1>;//i2s\_compatible the same bit for pcm\_short\_frame. And for i2s bus mode we call this bit as i2s\_compatible.

};

0011 1111 1000

采样率8000  16000 32000  时钟： devm\_clk\_get(i2s->dev, "clk\_twpll\_128m");

采样率:9600  12000 24000  48000 时钟：devm\_clk\_get(i2s->dev, "clk\_twpll\_153m6");

Example:

i2s configured as 96k 32bit

i2s0: i2s@70c00000  {

    compatible = "sprd,i2s";

    reg = <0x70c00000 0x100000>;

    sprd,dai\_name = "i2s\_bt\_sco0";

    sprd,hw\_port = <0>;

    sprd,syscon-ap-apb = <&ap\_apb\_controller>;

    sprd,config\_type = "i2s";

    sprd,slave\_timeout = <0xF11>;

    sprd,fs = <96000>;

    sprd,bus\_type = <0>;

    sprd,rtx\_mode = <3>;

    sprd,byte\_per\_chan = <2>; //32bit  对应i2s left justified模式

    sprd,slave\_mode = <0>;

    sprd,lsb = <0>;

    sprd,lrck = <1>;

    sprd,low\_for\_left = <0>;

    sprd,clk\_inv = <1>;

    sprd,pcm\_short\_frame = <0>;

    sprd,pcm\_slot = <0x1>;

    sprd,pcm\_cycle = <1>;

    sprd,tx\_watermark = <24>;

    sprd,rx\_watermark = <24>;

    sprd,i2s\_compatible = <0>;

};

默认的i2c和iis设置：

/\* default pcm config \*/

static const struct i2s\_config def\_pcm\_config = {

    .hw\_port = 0,

    .fs = 8000,

    .slave\_timeout = 0xF11,

    .bus\_type = PCM\_BUS,

    .byte\_per\_chan = I2S\_BPCH\_16,

    .mode = I2S\_MASTER,

    .lsb = I2S\_MSB,

    .rtx\_mode = I2S\_RTX\_MODE,

    .sync\_mode = I2S\_SYNC,/\* I2S\_SYNC better! \*/

    .lrck\_inv = I2S\_L\_LEFT,

    .clk\_inv = I2S\_CLK\_N,

    .pcm\_bus\_mode = I2S\_SHORT\_FRAME,

    .pcm\_slot = 0x1,

    .pcm\_cycle = 1,

    .tx\_watermark = 12,

    .rx\_watermark = 20,

};

/\* default i2s config \*/

static const struct i2s\_config def\_i2s\_config = {

    .hw\_port = 0,

    .fs = 32000,

    .slave\_timeout = 0xF11,

    .bus\_type = I2S\_BUS,

    .byte\_per\_chan = I2S\_BPCH\_16,

    .mode = I2S\_SLAVE,

    .lsb = I2S\_MSB,

    .rtx\_mode = I2S\_RX\_MODE,

    .sync\_mode = I2S\_LRCK,

    .lrck\_inv = I2S\_L\_LEFT,

    .clk\_inv = I2S\_CLK\_N,

    .i2s\_bus\_mode = I2S\_MSBJUSTFIED,

    .tx\_watermark = 12,

    .rx\_watermark = 20,

};

=====

es7210  iis设置:

设备树:

&i2s0 {

    status = "okay";

    sprd,config\_type = "i2s";

    sprd,slave\_timeout = <0xf11>;

    sprd,\_hw\_port = <0>;

    sprd,fs = <16000>;

    sprd,bus\_type = <0>;

    sprd,rtx\_mode = <3>;

    sprd,byte\_per\_chan = <1>;

    sprd,slave\_mode = <0>;

    sprd,lsb = <0>;

    sprd,lrck = <0>; /\*sync\_mode\*/

    sprd,low\_for\_left = <0>; /\*lrck\_inv\*/

    sprd,clk\_inv = <1>;//

    sprd,pcm\_short\_frame = <0>; /\*pcm\_bus\_mode\*/

    sprd,pcm\_slot = <1>;

    sprd,pcm\_cycle = <1>;

    sprd,tx\_watermark = <16>;

    sprd,rx\_watermark = <16>;

    sprd,i2s\_compatible= <1>;

};

hal 层:

static const int ext\_codec\_configs[I2S\_CONFIG\_MAX] = {

    16000, /\* fs \*/

    0, /\* hw\_port \*/

    0xf11, /\* slave\_timeout \*/

    0, /\* bus\_type \*/

    1, /\* byte\_per\_chan \*/

    0, /\* mode(master/slave mode) \*/

    0, /\* lsb \*/

    3, /\* rtx\_mode \*/

    0, /\* lrck\_inv(low\_for\_left) \*/

    0, /\* sync\_mode(lrck) \*/

    1, /\* clk\_inv \*/

    1, /\* i2s\_bus\_mode \*/

    0, /\* pcm\_bus\_mode \*/

    0, /\* pcm\_slot \*/

    0, /\* pcm\_cycle \*/

    16, /\* tx\_watermark \*/

    16, /\* rx\_watermark \*/

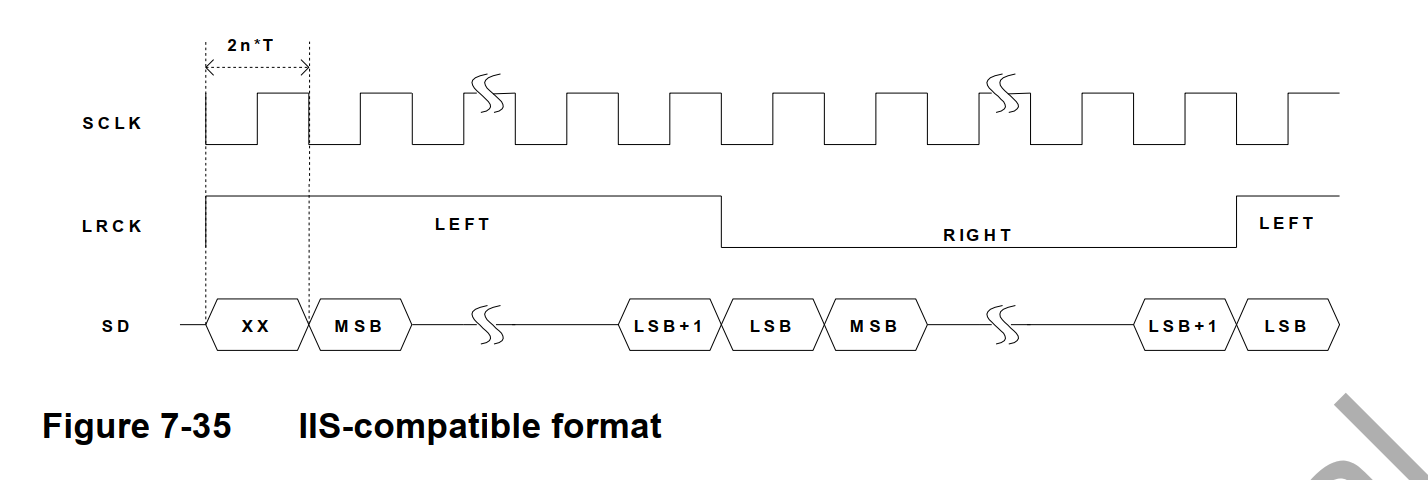
};

====

sync模式

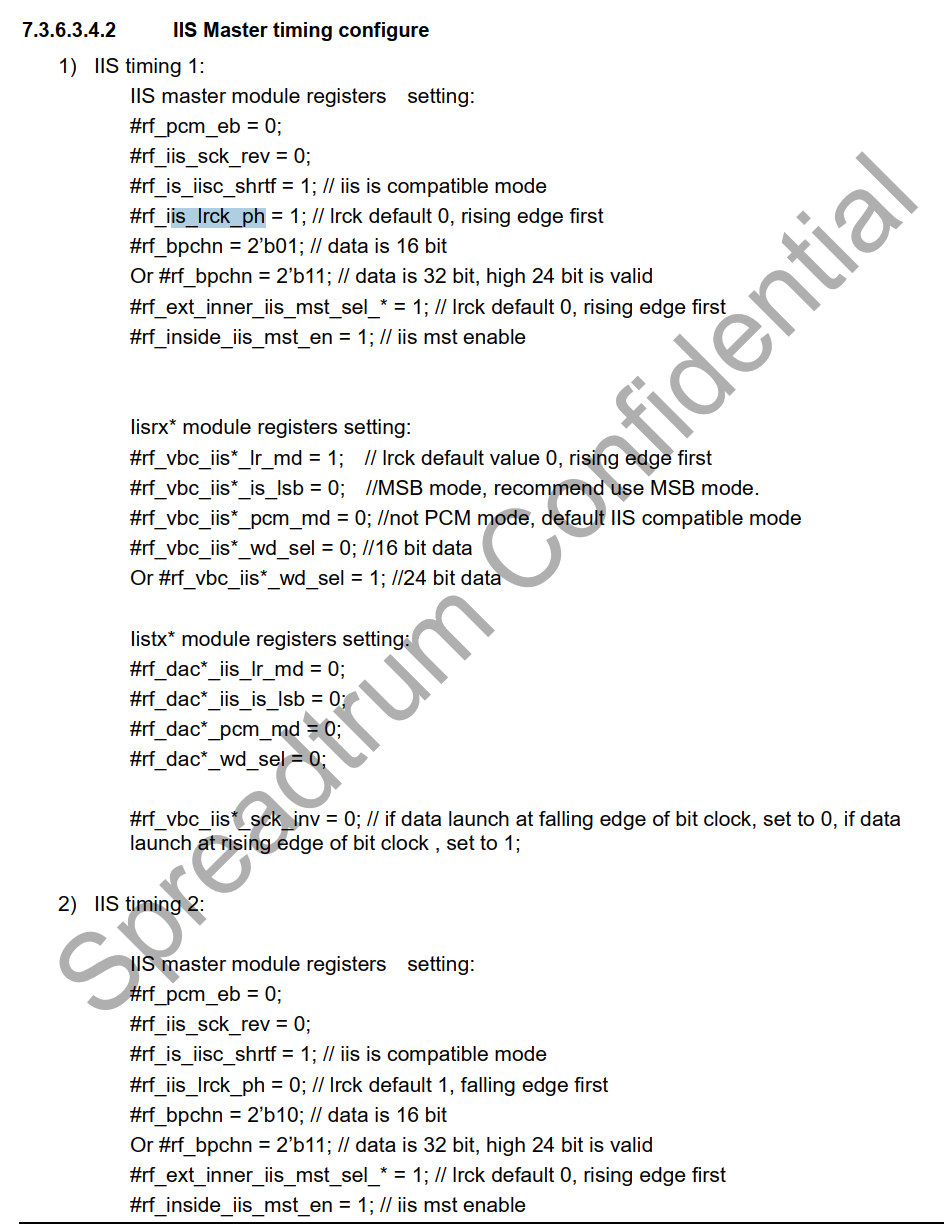


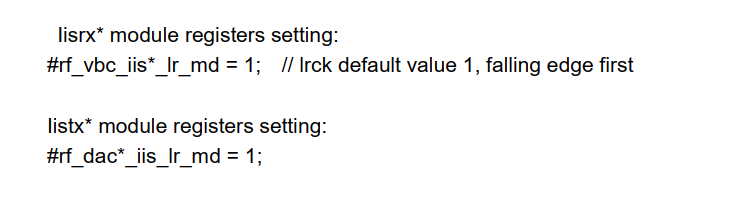




串行时钟 SCLK，也叫做位时钟BCLK，即对应数字音频的每一位数据，SCLK的频率=2×采样频率×采样位数 ,呵呵，现在问题来了，有人会问这些东西到底是什么意思呢？其实，I2S一般是传输立体声，有两个声道channel，采样频率指得是采样数率，多久去采集一个点，每个点是几个bit组成。  
帧时钟LRCK，用于切换左右声道的数据，LRCK为“0”表示正在传输的是左声道的数据，为“1”表示正在传输的是右声道的数据。LRCLK == FS,就是采样频率  
串行数据SDATA，就是用二进制补码表示的音频数据，有时为了使系统间能够更好的同步，还需要另外传输一个信号MCLK，称为主时钟，也叫系统时钟（System Clock），是采样频率的256或384倍

iis timing:

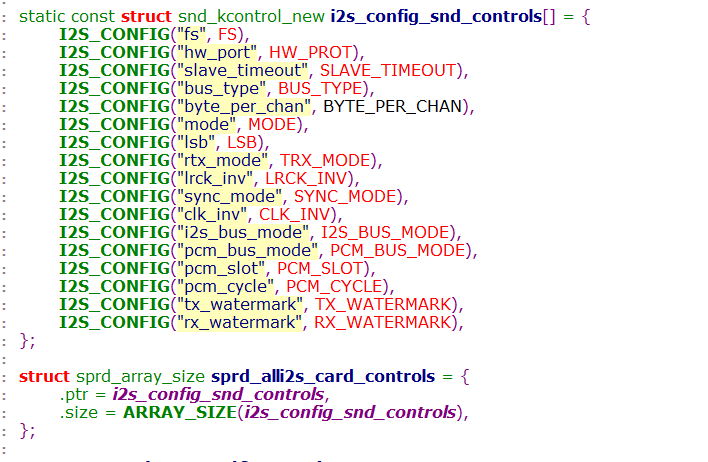




=======

I2s-r0p0-dummy-codec.c (sound\soc\sprd)    3992    2022/2/28

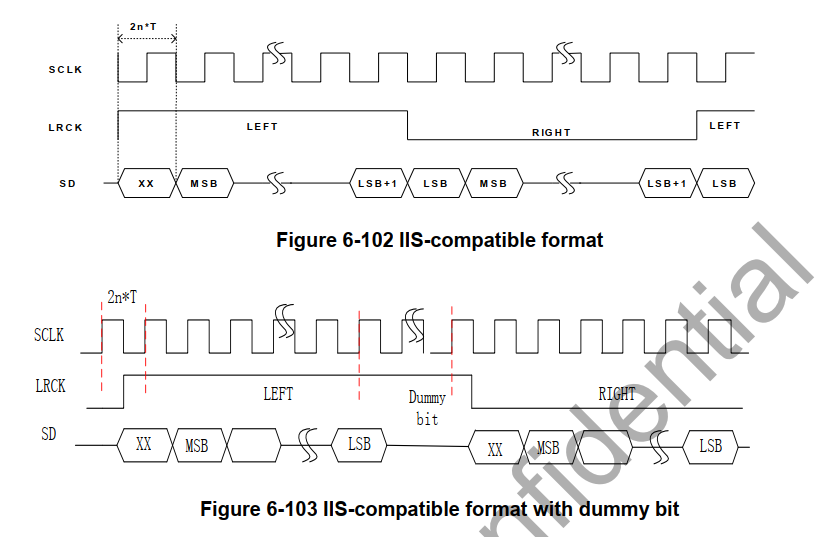
hal层设置i2s 模式的接口:



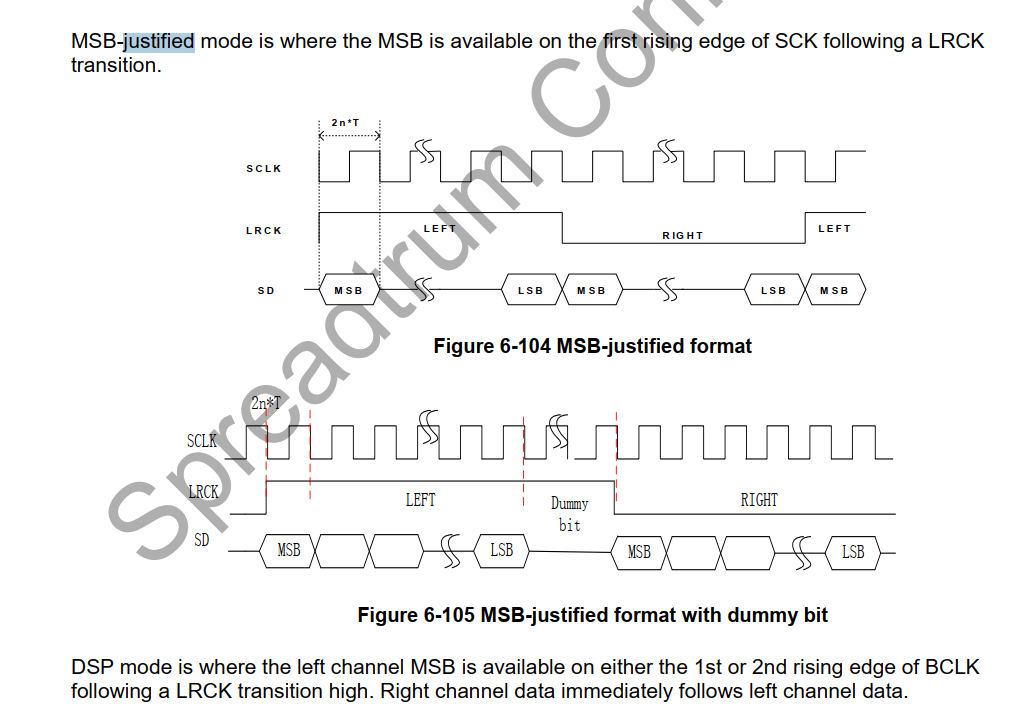
iis 的模式

iis mode 和 MSB-justified mode

IMG_268



MSB-justified mode



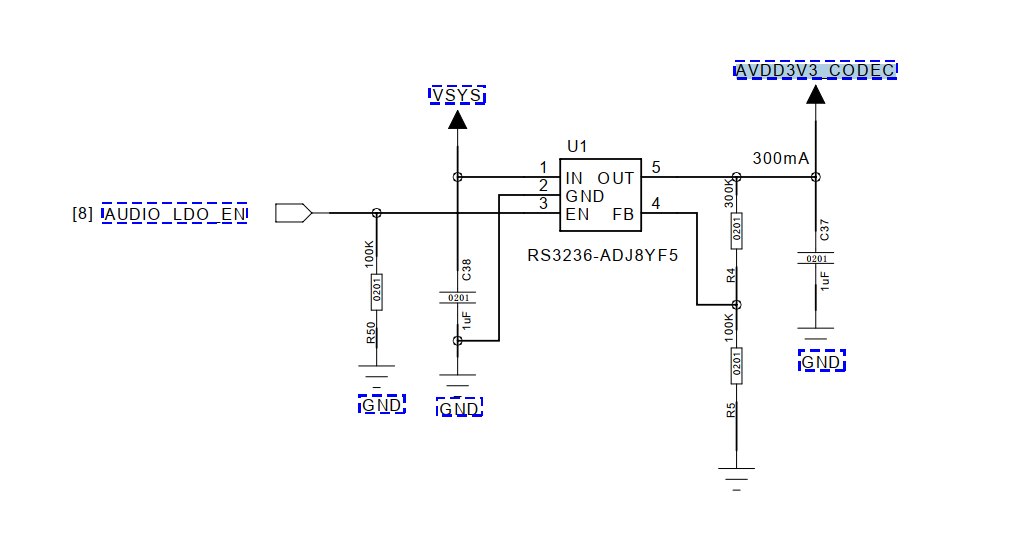
====



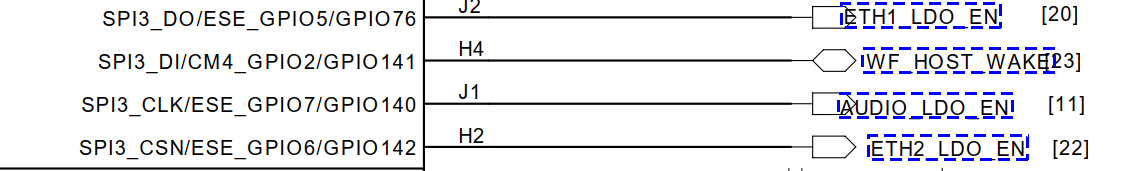
供电:

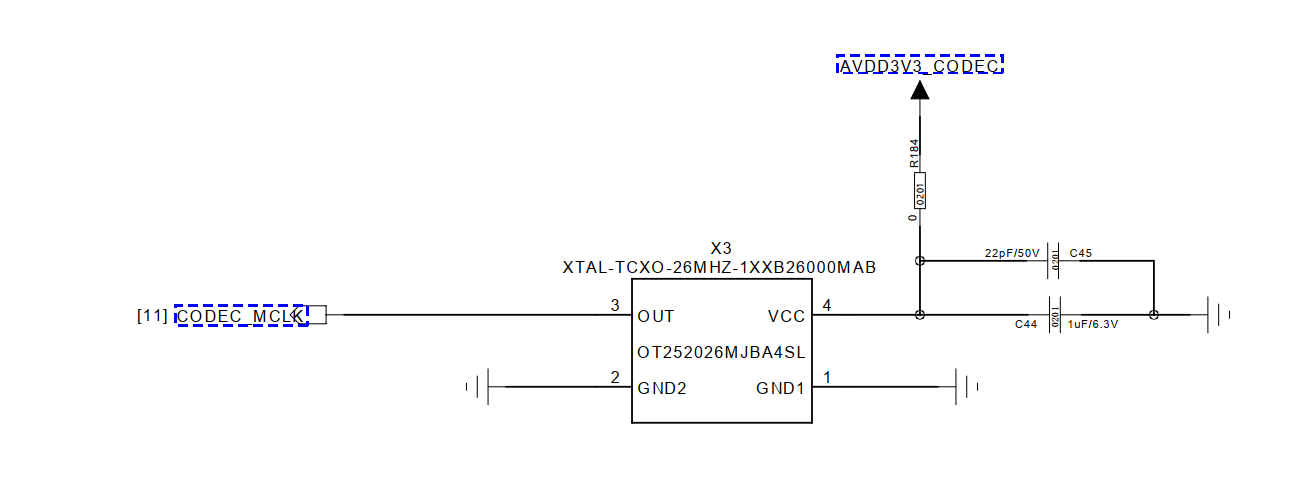
DVDD1V8\_CODEC

AVDD3V3\_CODEC



AUDIO\_LDO\_EN---->gpio140



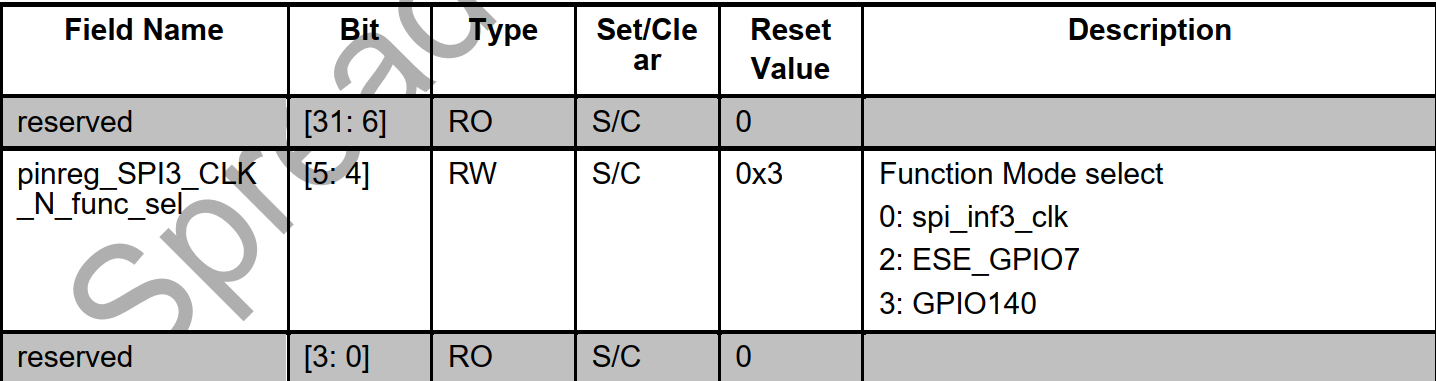


==

gpio140



IMG_276



===

[   15.082789] c6 i2s-null-codec sound@1:     name : i2s\_bt\_sco0-snd-soc-dummy-dai

[   15.089728] c6 i2s-null-codec sound@1:     stream\_name : i2s\_bt\_sco0-snd-soc-dummy-dai

[   15.097424] c6 i2s-null-codec sound@1:     name : i2s\_bt\_sco0-ES8311 HiFi

[   15.103843] c6 i2s-null-codec sound@1:     stream\_name : i2s\_bt\_sco0-ES8311 HiFi

[   15.110941] c6 [ASoC:BOARD] sprd\_asoc\_card\_parse\_hook\_spk hook aw87xx i2c pa

[   15.117990] c6 [Audio:AGDSP\_ACCESS] agdsp\_access\_enable out

[   15.123767] c6 Enter into es8311\_probe()

[   15.213085] c4 modem\_ctrl mdm\_ctrl: crash send to cp.

[   15.218058] c4 pcie res: wait resource, val=-62.

[   15.222625] c4 mpm: sipc-nr-mpm-1 wait resource, ret=-62, timeout=-1.

[   15.229035] c4 smsg\_ch\_open: channel 1-120 send open msg error = -62!

[   15.235444] c4 sipa\_dele: conn\_thread sipa\_delegator failed to open dst 1 channel 120

[   15.240897] c6 Enter into es8311\_set\_bias\_level(), level = 1

[   15.253718] c4 i2s-null-codec sound@1: snd-soc-dummy-dai <-> i2s\_bt\_sco0 mapping ok

[   15.261266] c4 Enter into es8311\_set\_dai\_fmt()

[   15.267063] c4 ES8311 in Slave mode

[   15.270673] c4 es8311 4-0018: ASoC: Failed to set DAI format: -22

[   15.277240] c4 i2s-null-codec sound@1: ES8311 HiFi <-> i2s\_bt\_sco0 mapping ok

[   15.285773] c0 Enter into es8311\_set\_bias\_level(), level = 1

[   15.299568] c4 ALSA device list:

[   15.302717] c4   #0: sprdphone-sc2730

[   15.306334] c4   #1: all-i2s

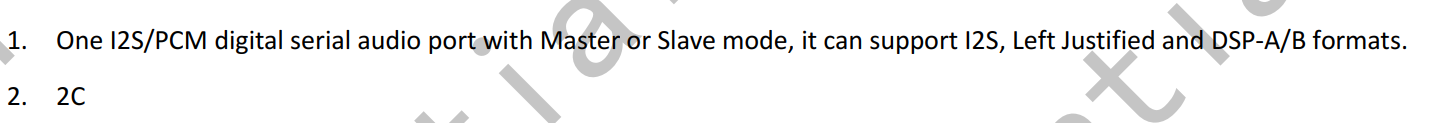
=====

0x00069040

0110    1001    0000    0100    0000--->0x01-->iis1-->ap iis1

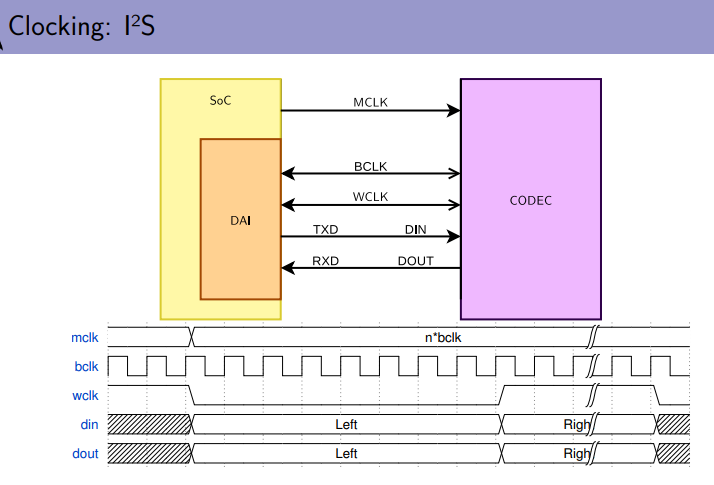
0110    1001    0000    0000    0000--->0x00-->iis1-->ap iis0

====



支持i2s left justified   dsb-a/b 模式

时钟相关



MCLK ---> 给codec的时钟

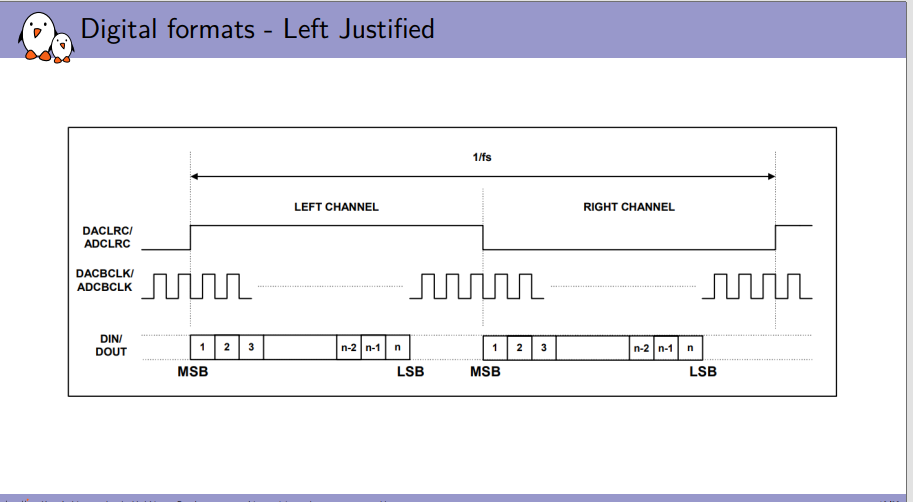
BCLK ---> 数据bit时钟

WCLK--> word 字时钟 LRCLK左右声道时钟 FCLK/FSCLK帧时钟

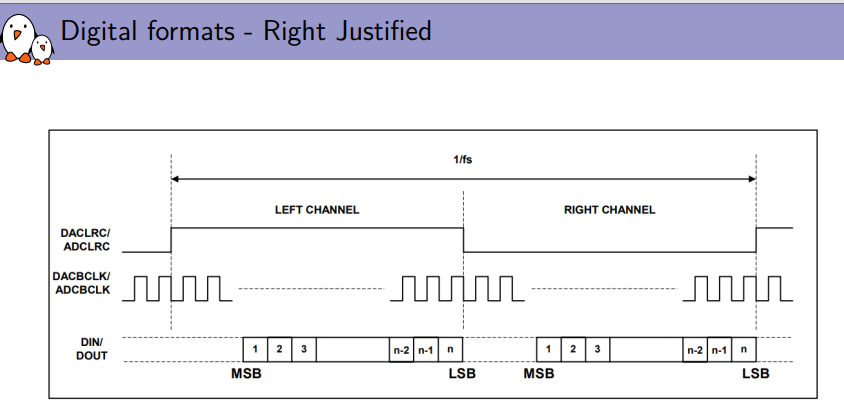
Bclk = Wclk \* Nchannels \* BitDepth

codec的BLCK需要通过MCLK的分频

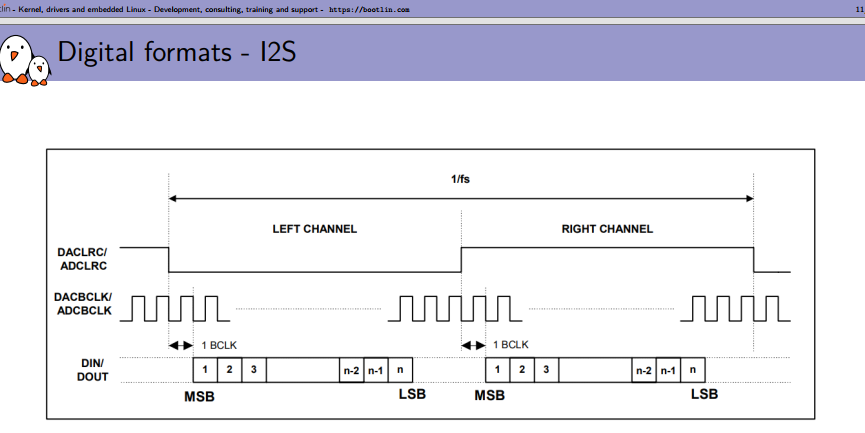
iis的左对齐模式:左声道为高



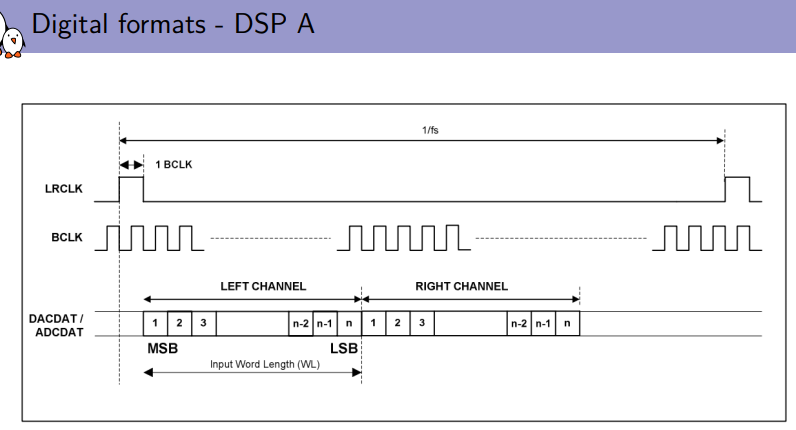
iis 右对齐模式



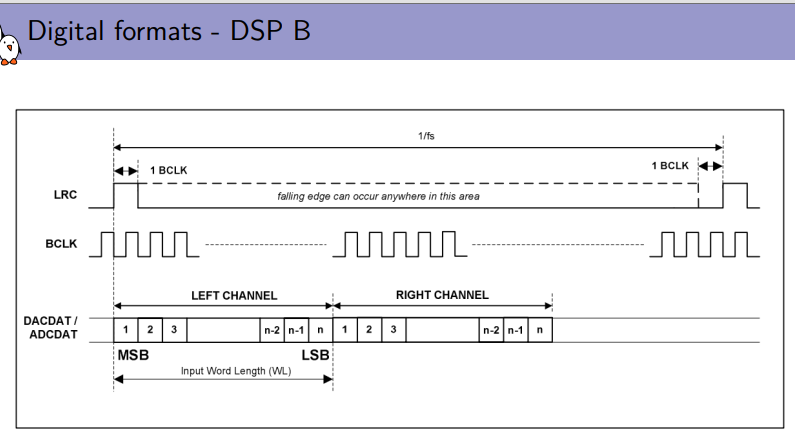
iis的标准模式: 左声道为低



iis  dsp-a 模式

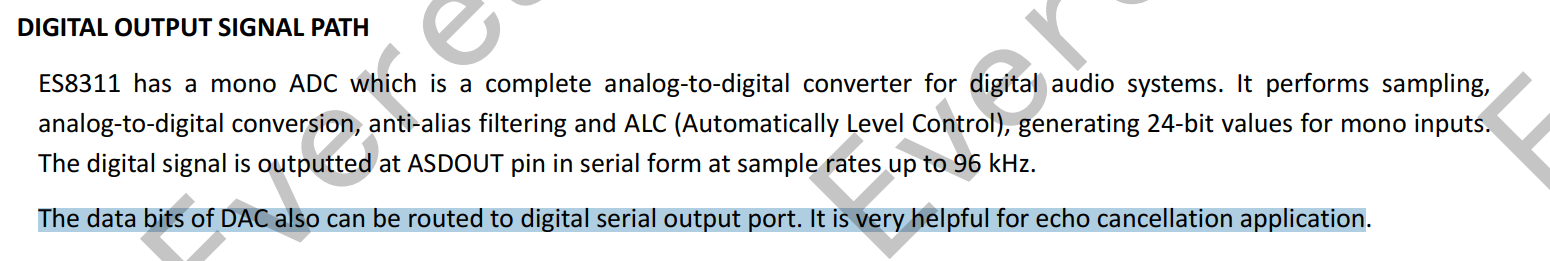


iis dsp-b 模式



===

具有回声消除功能：



==

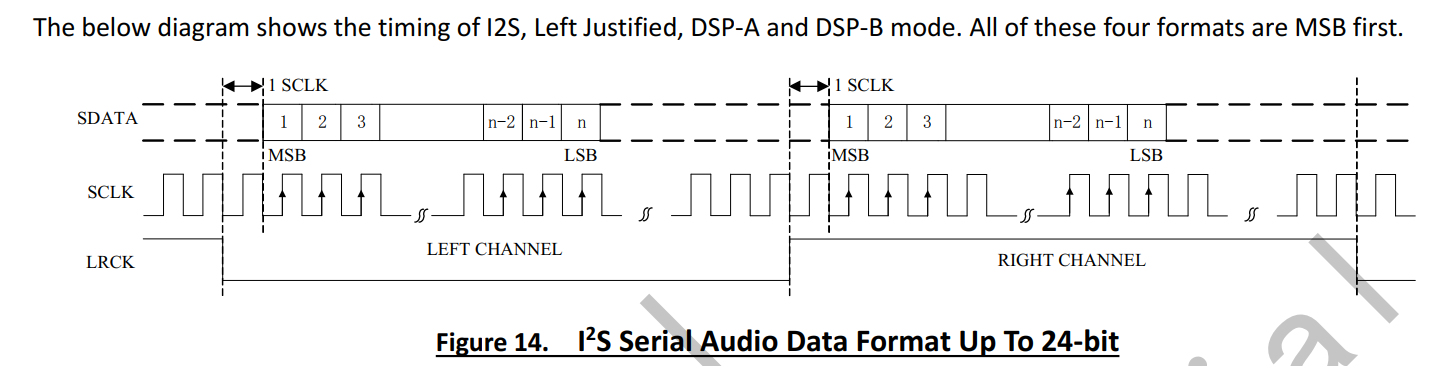
es8311只支持 4种数据格式:

1:标准的I2S格式数据。

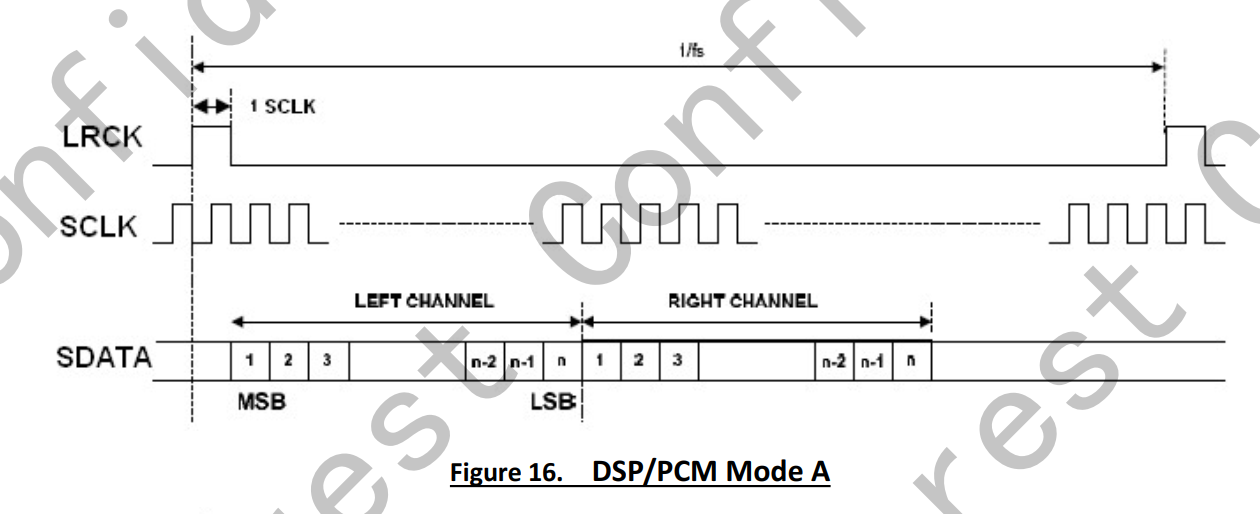
2：左对齐 LEFT Justified

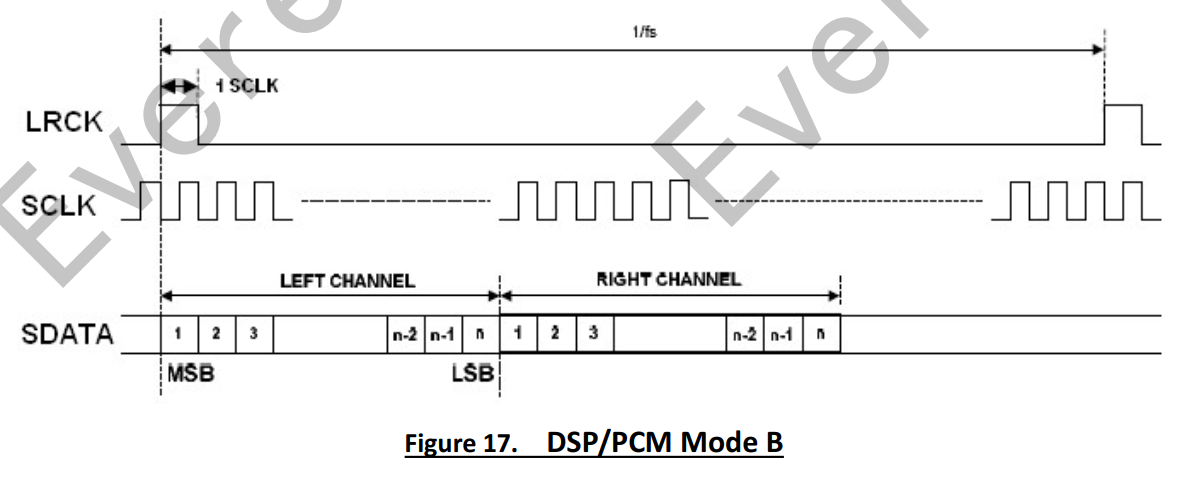
3:DSP mode A

4:DSP mode B.









1：寄存器0x09和寄存器0x0a 选择adc和dac的格式

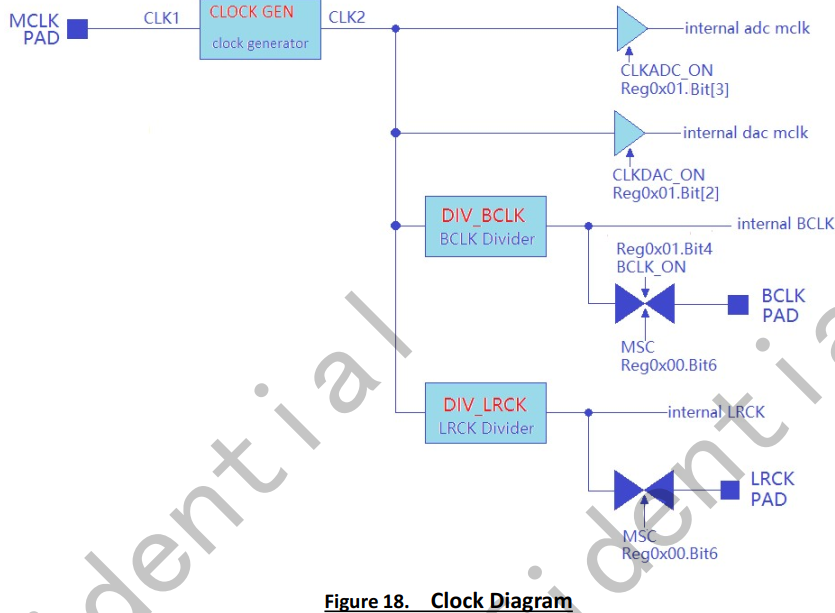
a: DAC的数据通道选择 LEFT和right通道。--->这个左右通道应该是i2s的数据通道

b:    设置极性是否反正

c:     设置数据宽度:    24bit    20bit    18bit    16bit    32bit

d:设置数据格式：i2s left dsp/pcm

时钟架构:



==

寄存器0x00-->MSC  选择Master或者Slave

1:选择master或者slave

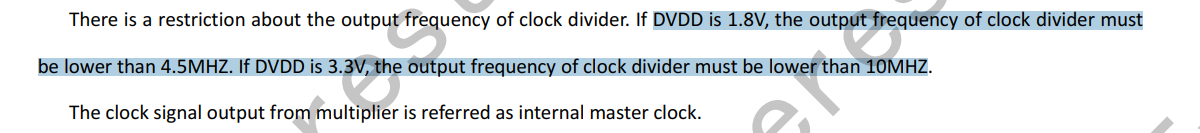
时钟选择 可选择mclk pin  或者sclk

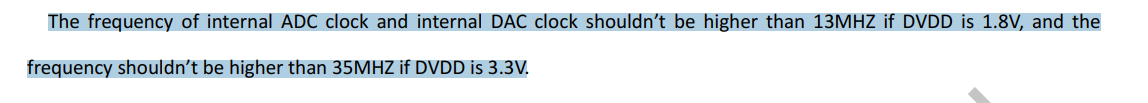
a:通过寄存器的Reg 0x00 Bit7

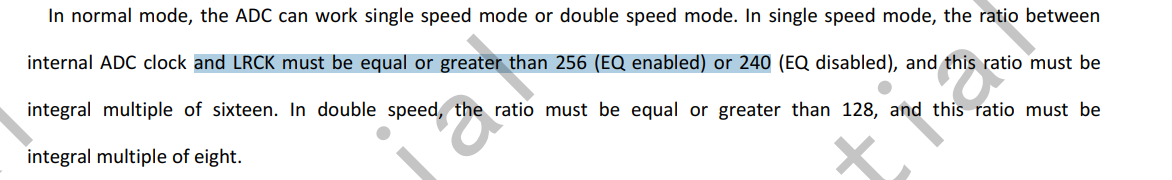
b:寄存器Reg0x01 Bit[5] 设置为1 MCLK ON 和sclk没有关系

==

时钟个电压的关系

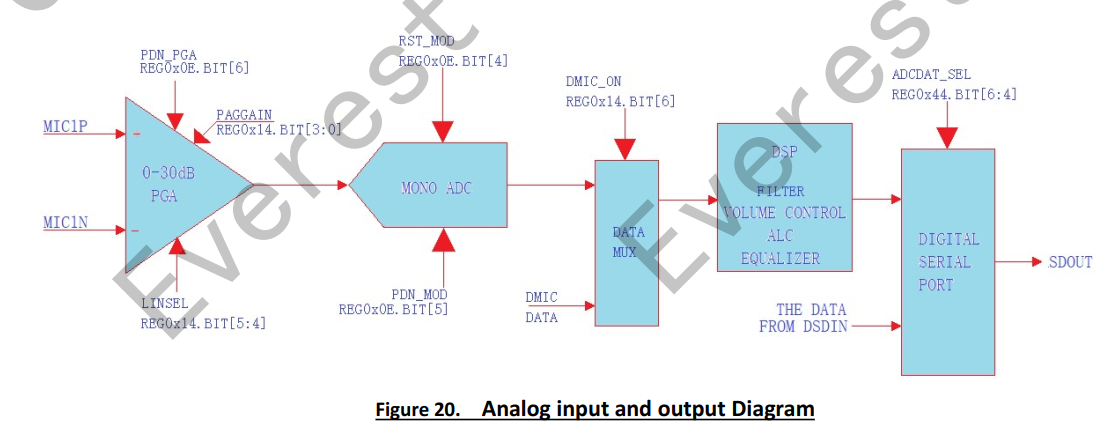


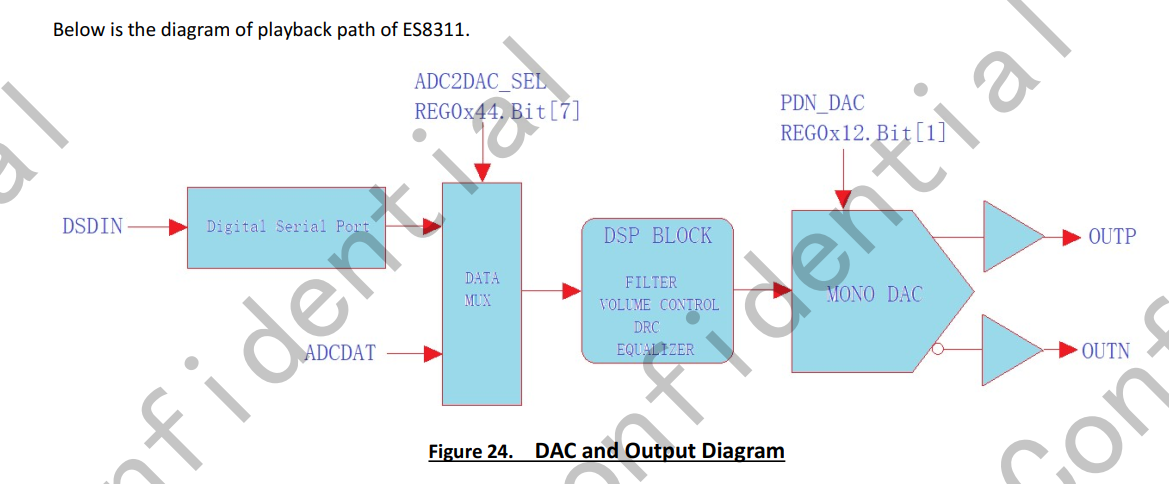




mclk时钟只能做输入

\*\*



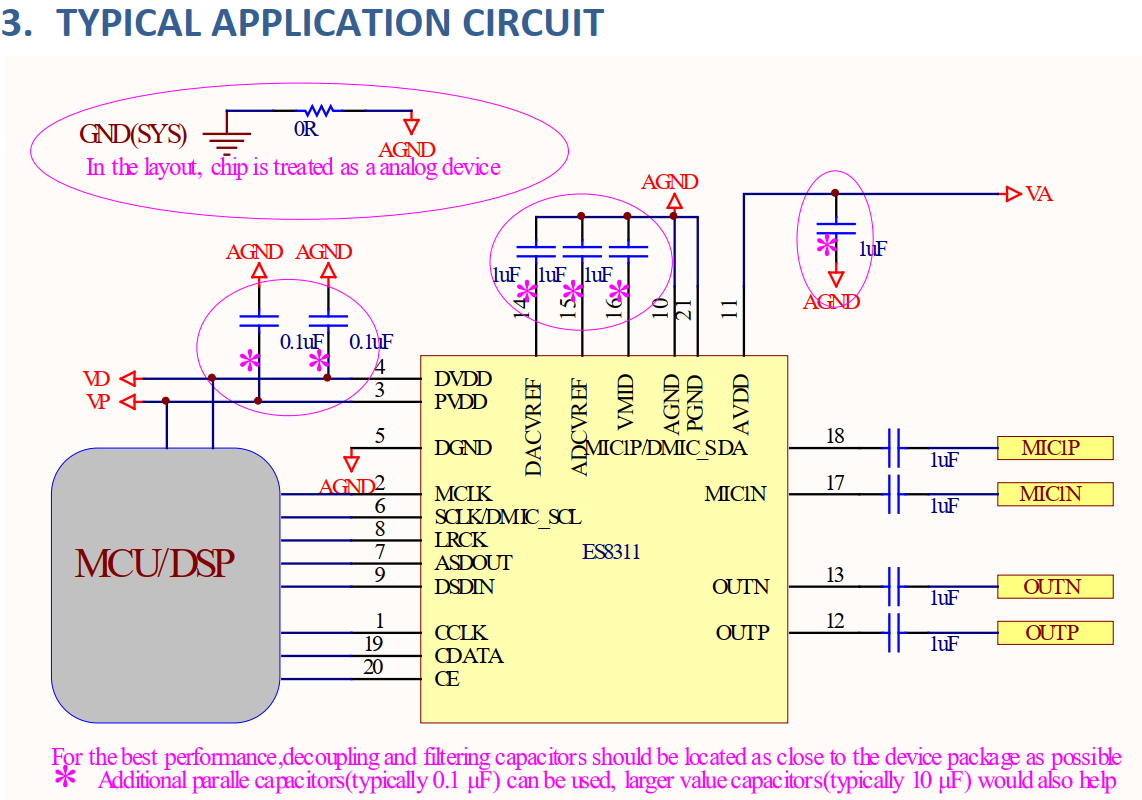


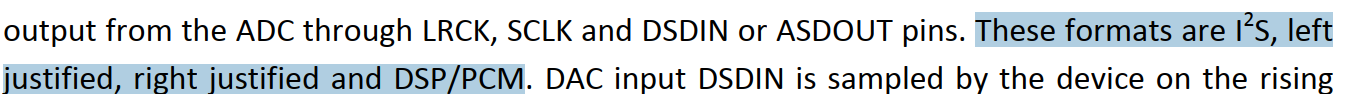
====

引脚设置:



电路





在设备的上升沿接收数据

T10

====

t710端的iis 和pcm的时序

ctl0 ----->  1000 1101 0001

0：bit[15] = 0 i2s模式

0:bit[14] = 0  没使用dma

00:bit[13:12] = 00  没使用

1：bit[11] = 1 sck bit时钟翻转

0:bit10=0 low for left 左声道为低

0：bit9 = 0 输出LRCK

0：bit8  =0 msb justified模式

11:bit[7:6] =11 收发功能

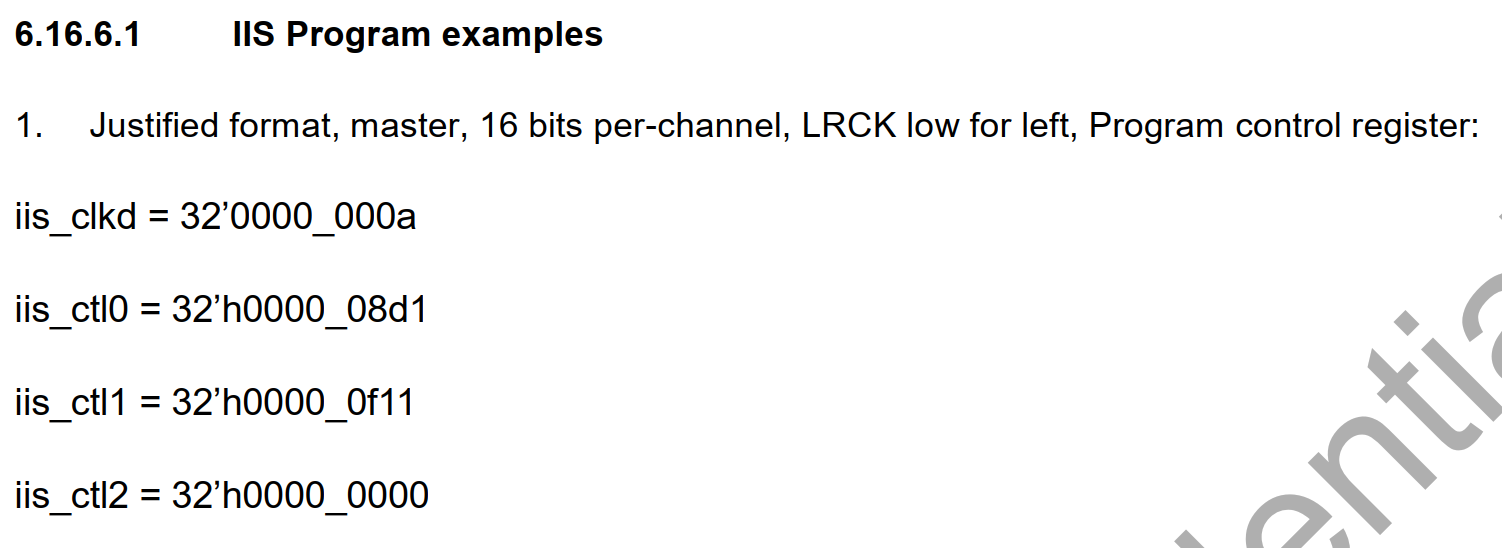
01:bit[5:4] =01 16bit每个channel

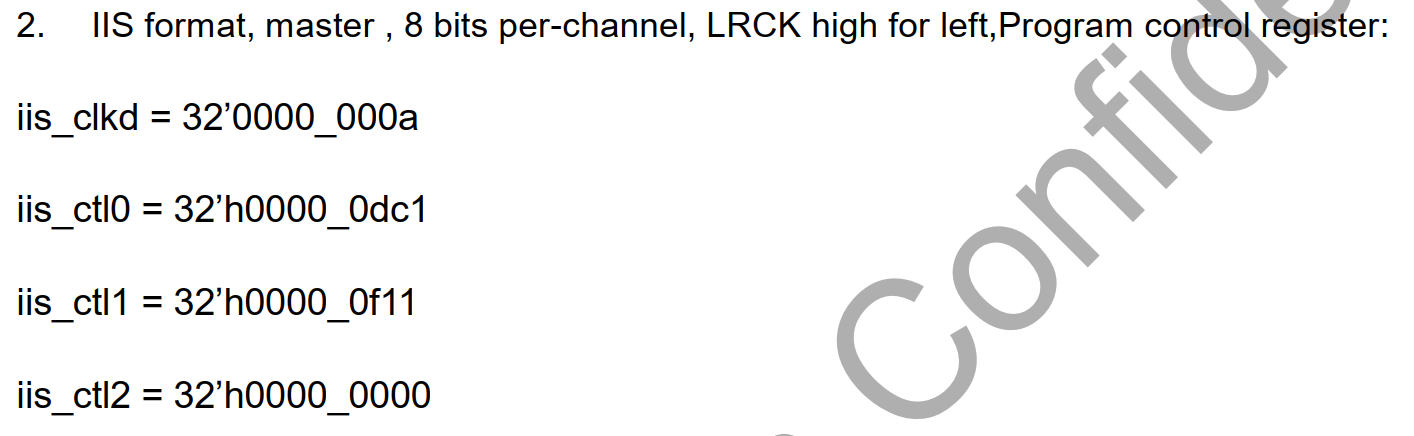
0:bit3:iis master

0: bit2:MSB

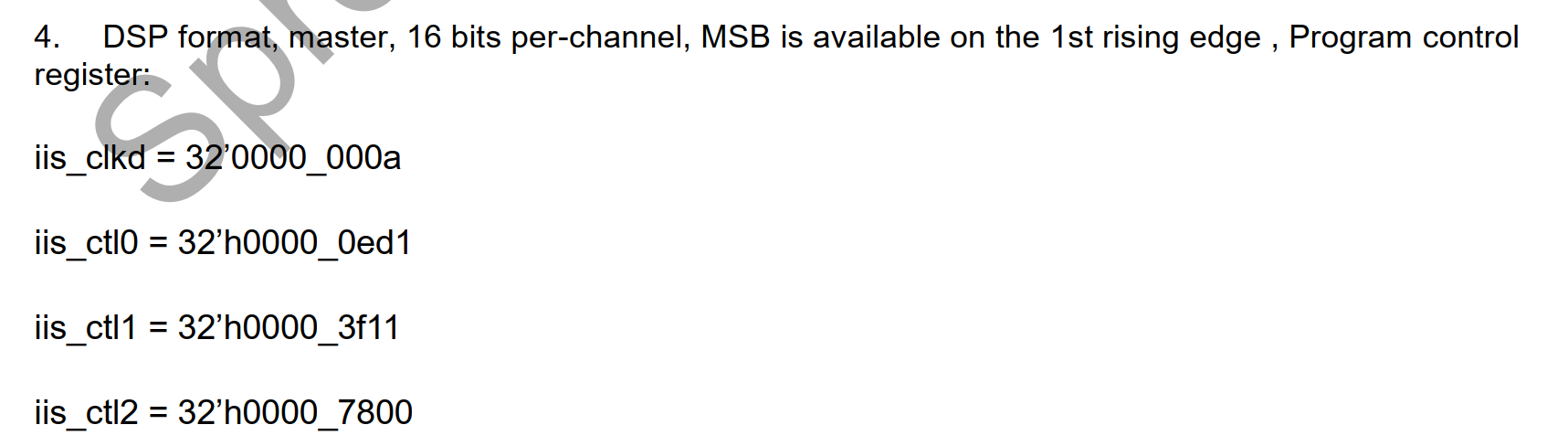
0：bit1

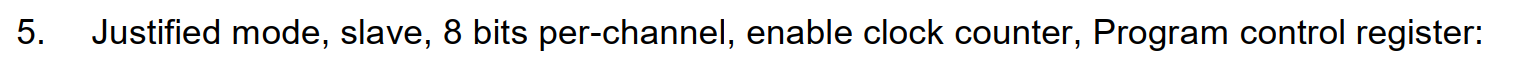
1:bit0 NG\_RX  设置接收数据的移位极性

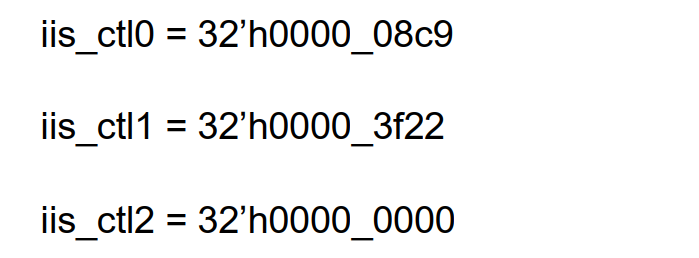




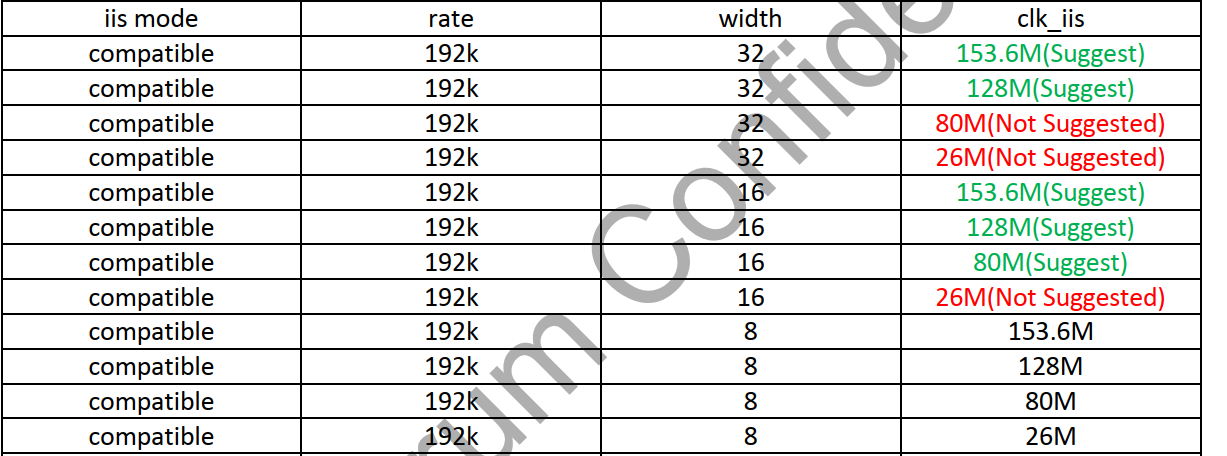


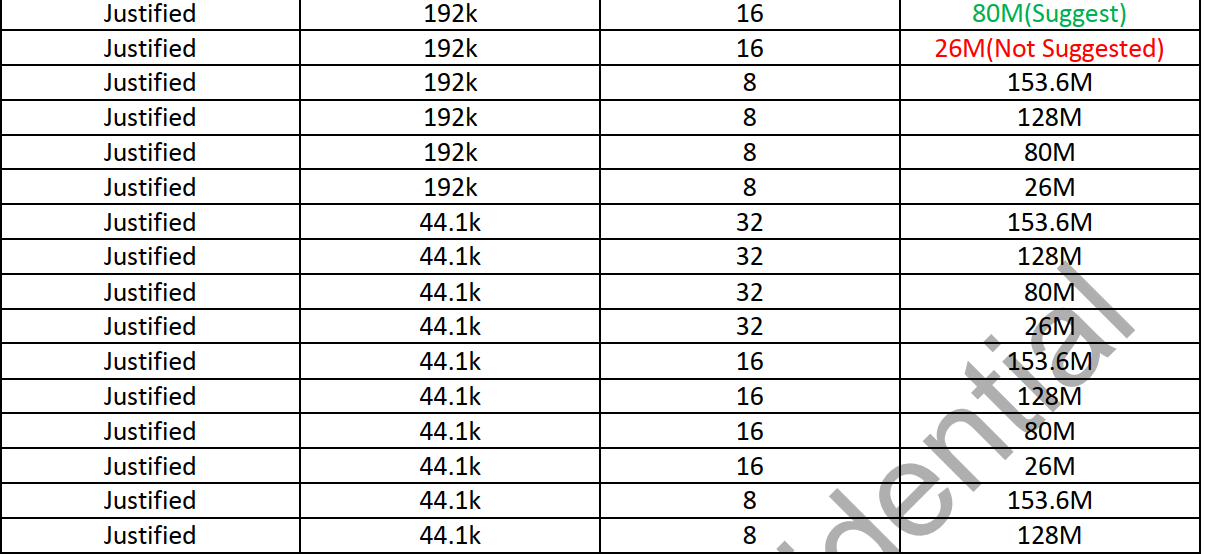


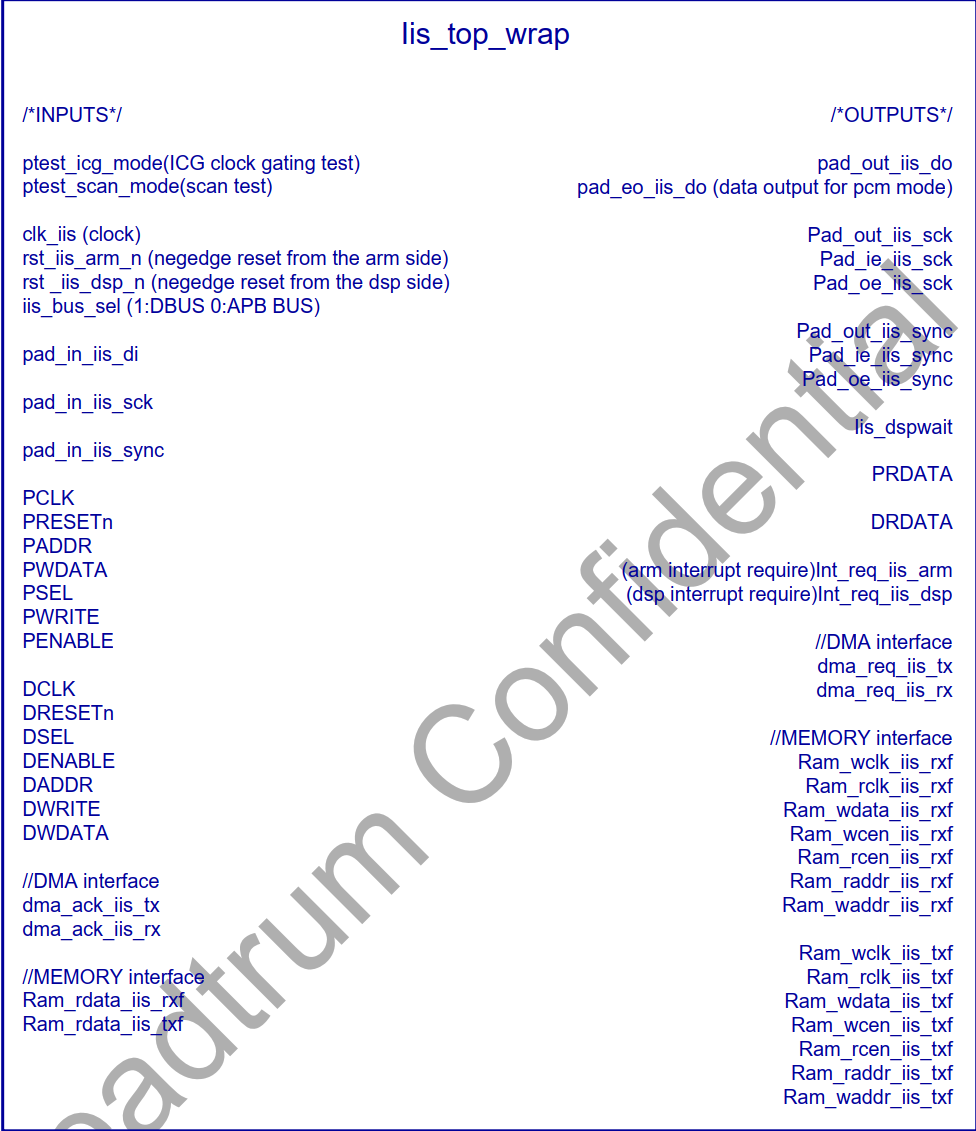




时钟设置：





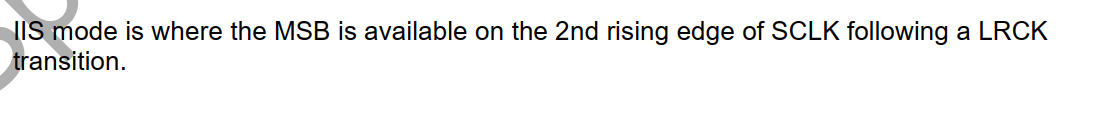


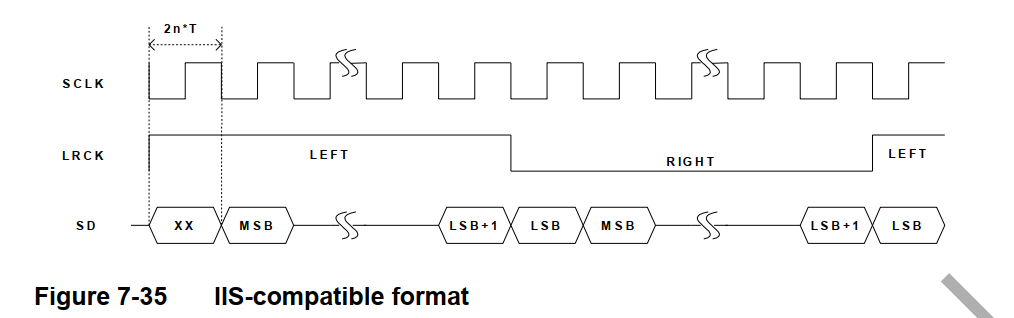
i2s模式选择：

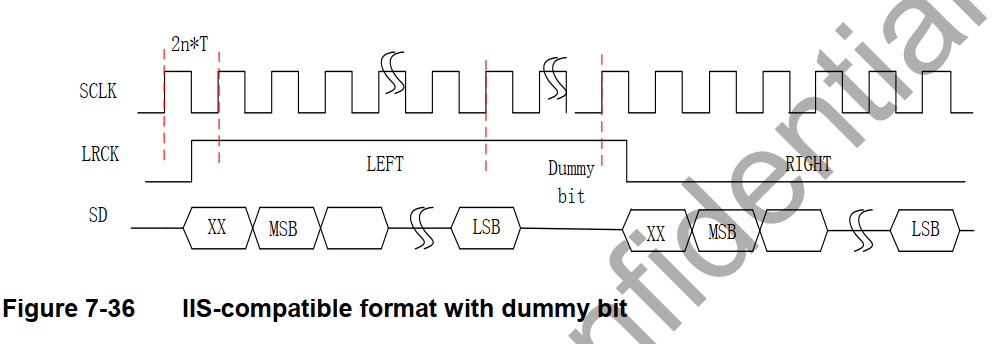
IIS 三种模式：

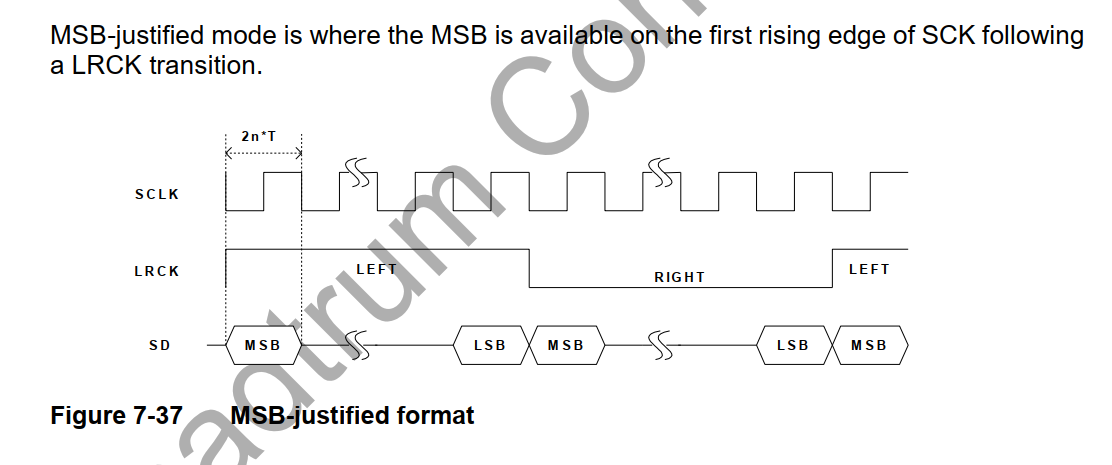
1：iis  2:MSB-Justified 3:DSP

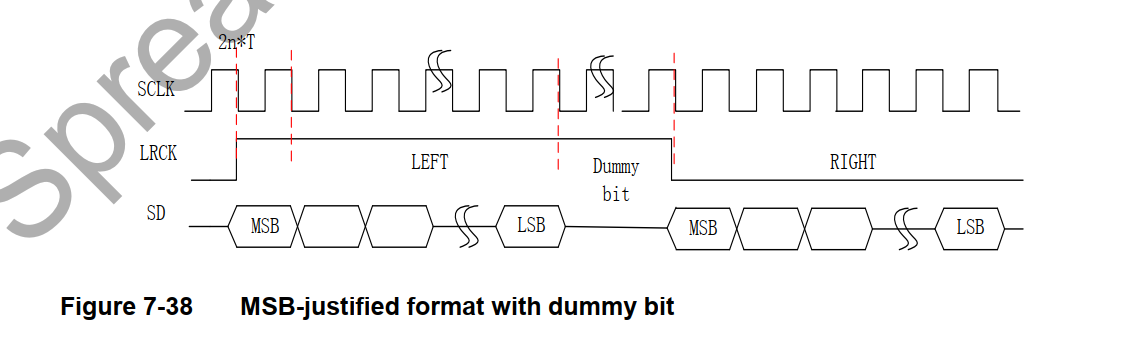
2:字宽： 8 16 32bit

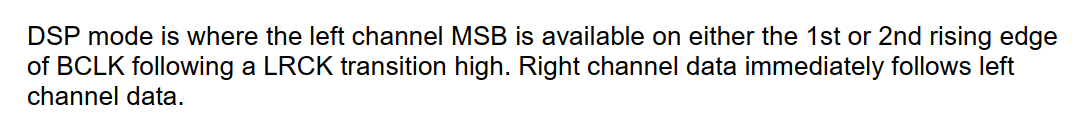


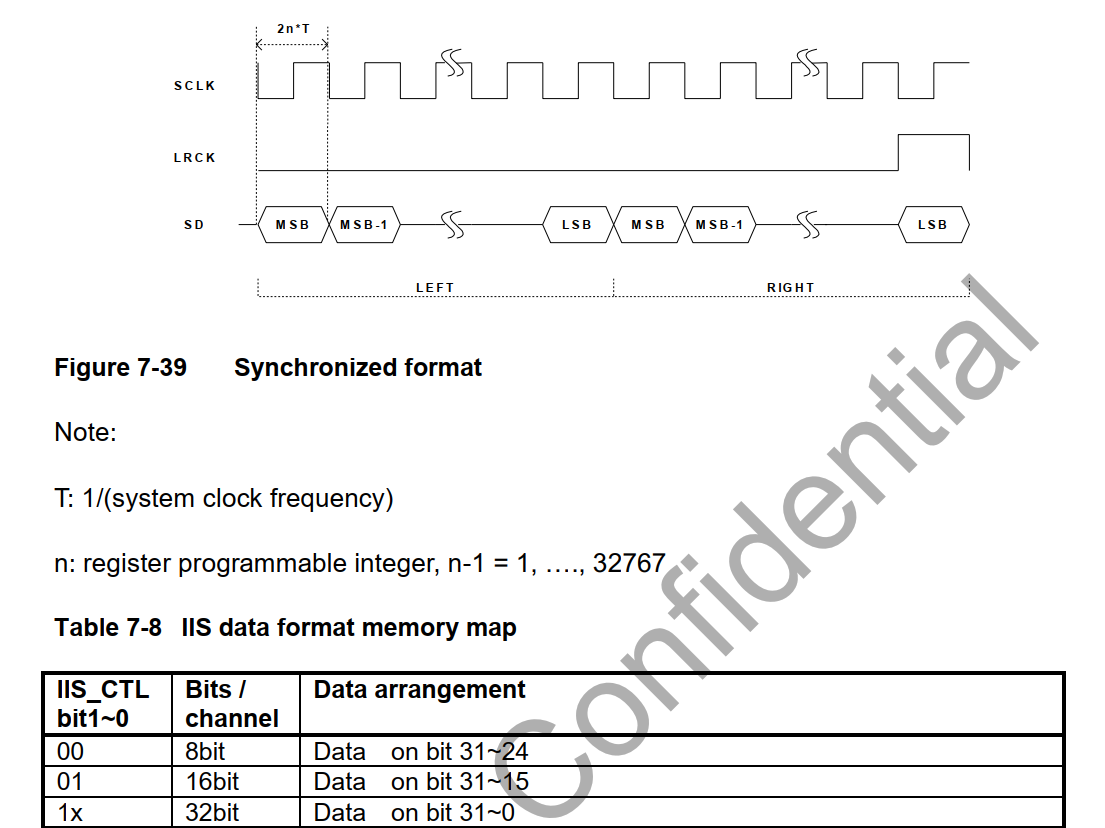






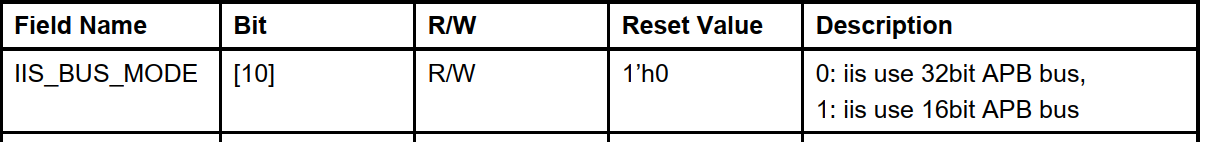






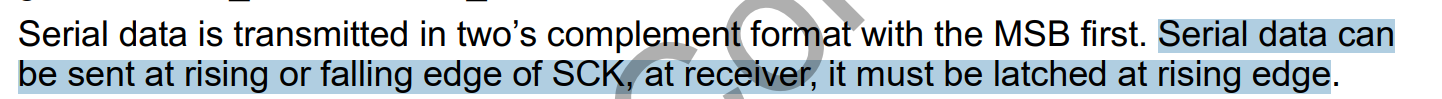
==

iis 总线模式



发送数据可以在上升沿或者下降沿

接收方必须在上升沿采样



## 如果系统的字长比传输的iis 传输字长 要多，则数据进行截断处理

###如果发给接收方的数据比字长要多  lsb部分要忽略掉

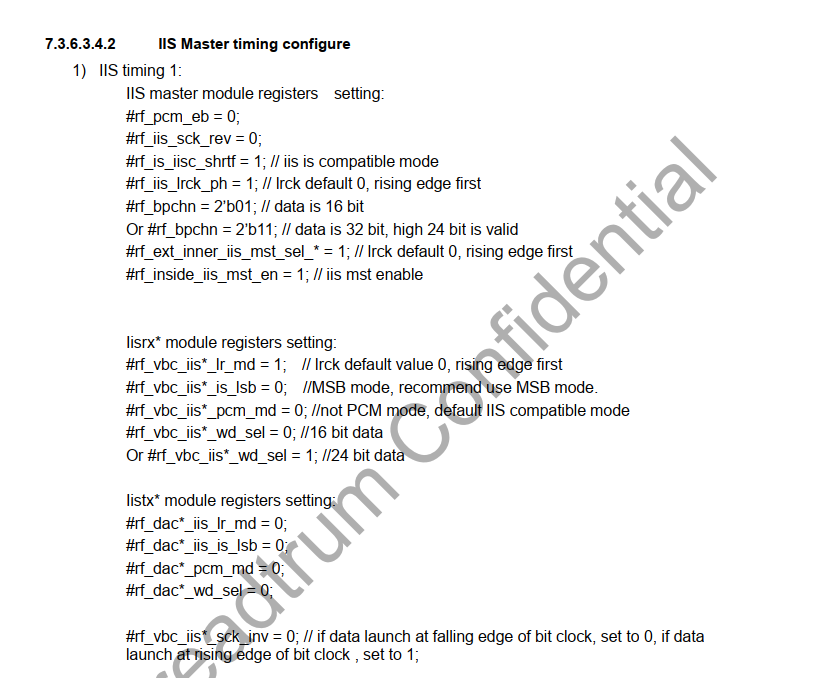
###如果接收方的数据比字长少  少的部分用0补充

###LRCK的极性可以编程

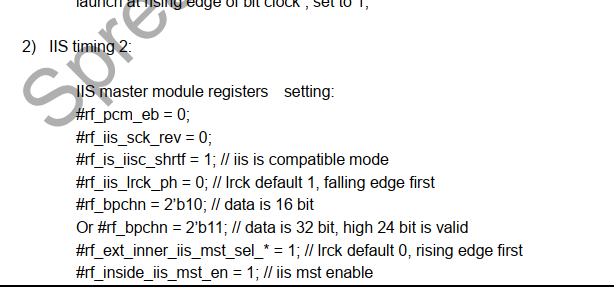
###

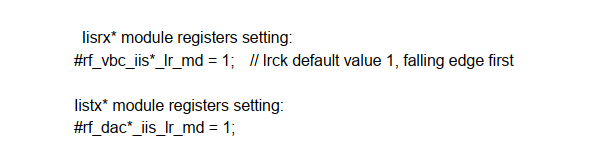
vbc  iis时序设置:

iis 标准格式



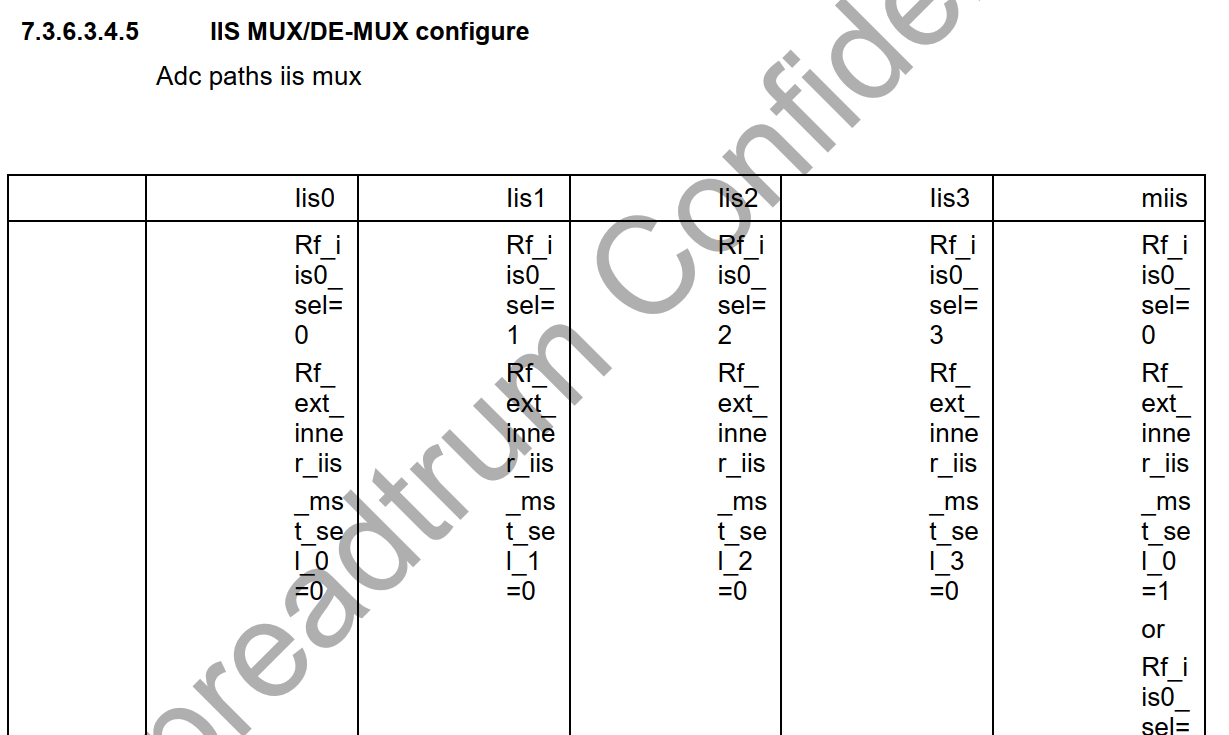
iis  格式：



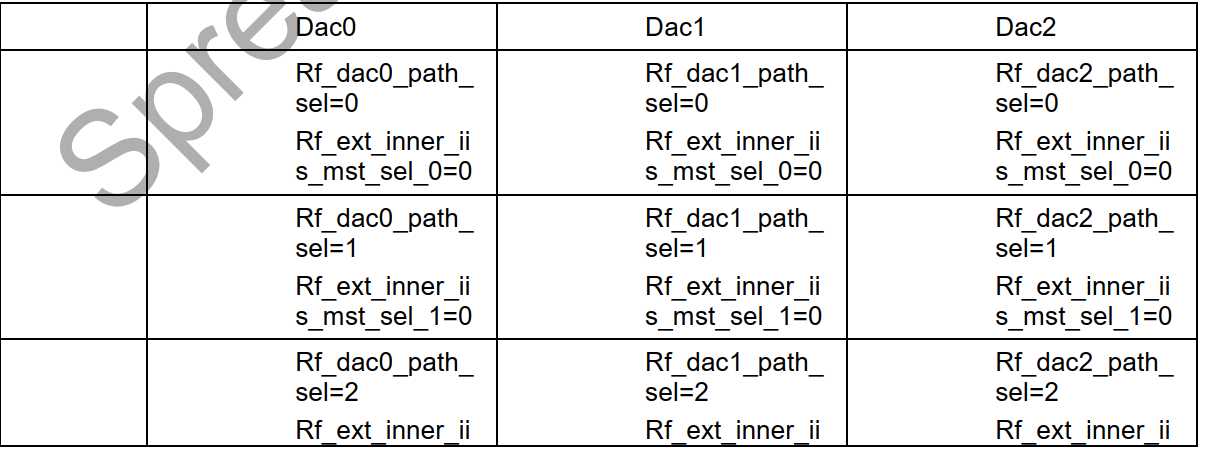


===

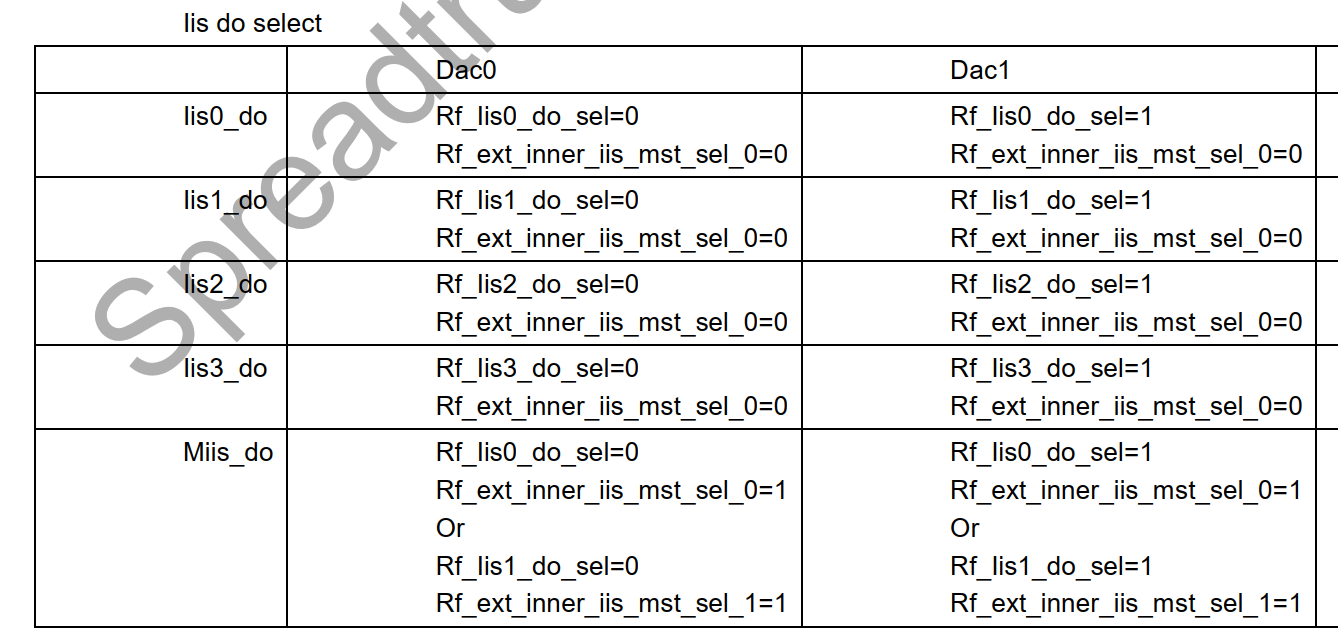
vbc iis 的mux和demux的多路复用和解复用的使用:



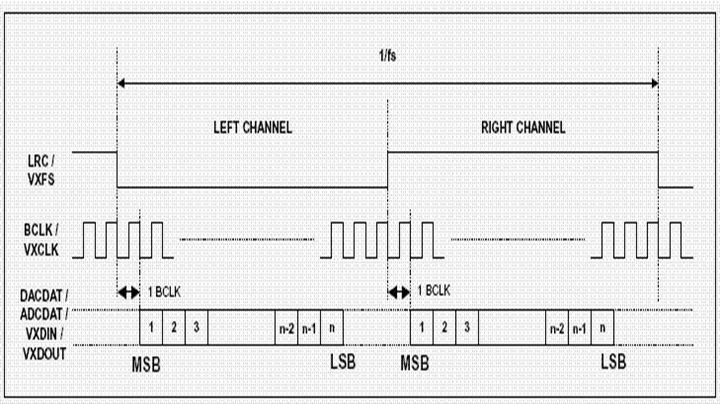
dac 路径 iis clk和 lrck 选择



iis do 选择



===

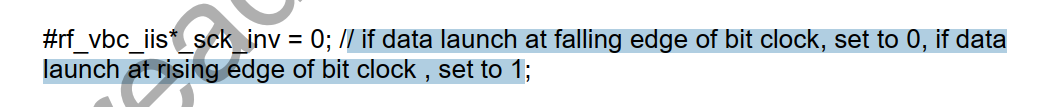


标准的i2s格式 低是左声道    高是右声道

===

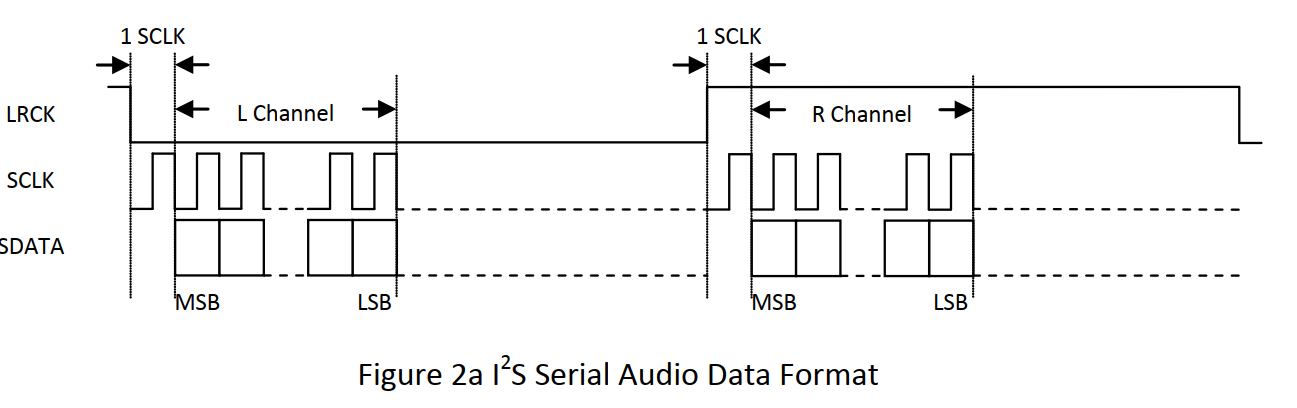
sck  和lrck的关系

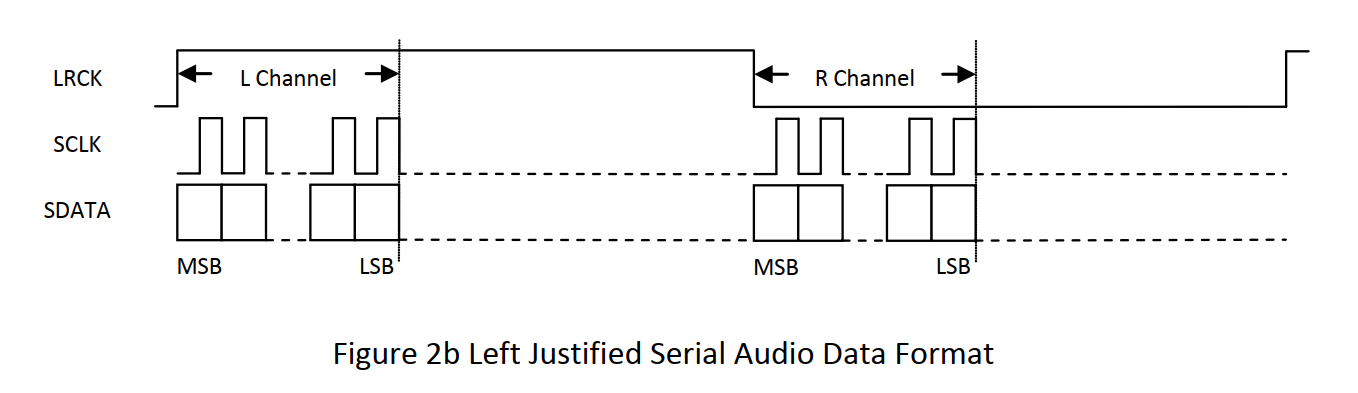
sck\_inv是否要设置为0 Or 1?

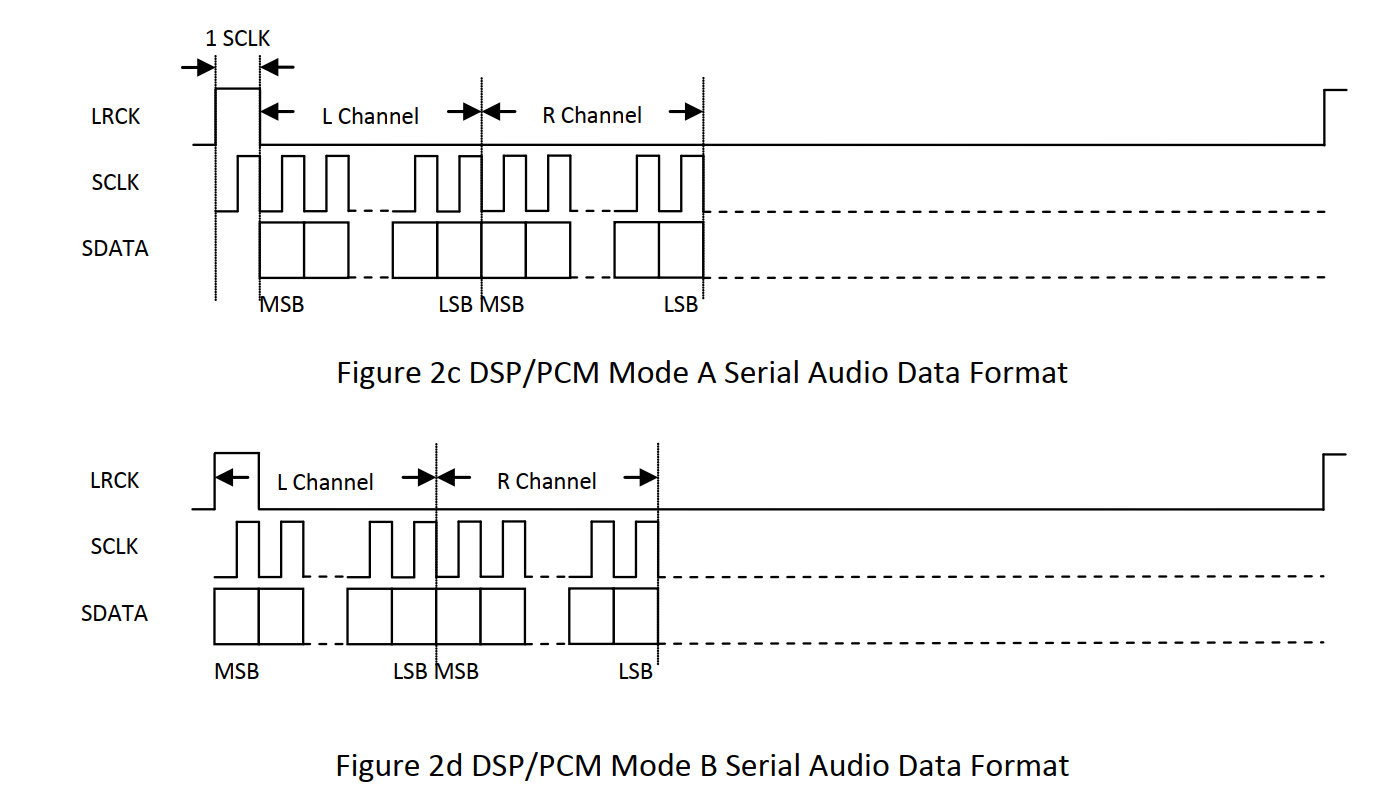


es8311的数据 在接收端于上升沿采样   发送数据在下降沿









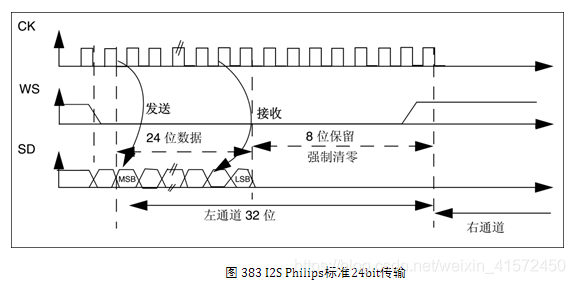
<https://blog.csdn.net/weixin_41572450/article/details/103582662、>

音频数据传输协议标准

使用WS信号来指示当前正在发送的数据所属的通道（即一帧的开始），数据传输从MSB到LSB，发送方在时钟信号（CK）的下降沿改变数据，接收方在上升沿读取数据，WS也在CK的下降沿变化，有3种标准

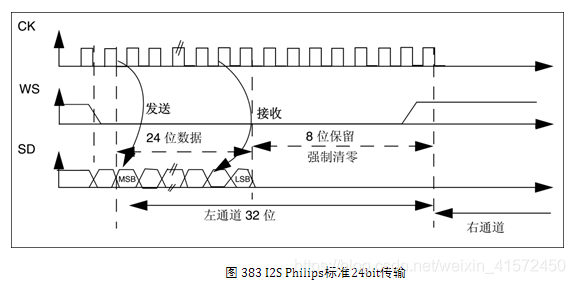
I2S Philips标准

串行数据（SD）在WS变化后的第2个时钟信号（CK）边沿开始发送MSB，下面是为24bit数据封装在32bit帧传输波形。



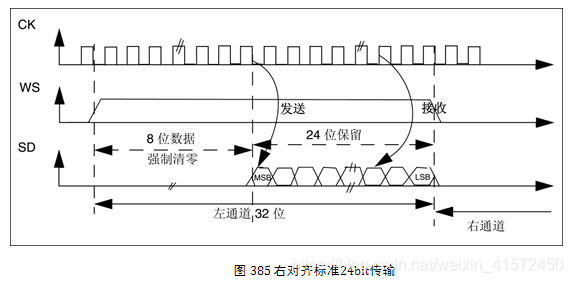
左对齐标准

在WS发生翻转同时开始传输数据，参考图 384，为24bit数据封装在32bit帧传输波形。该标准较少使用。注意此时WS为1时，传输的是左声道数据，这刚好与I2S Philips标准相反。

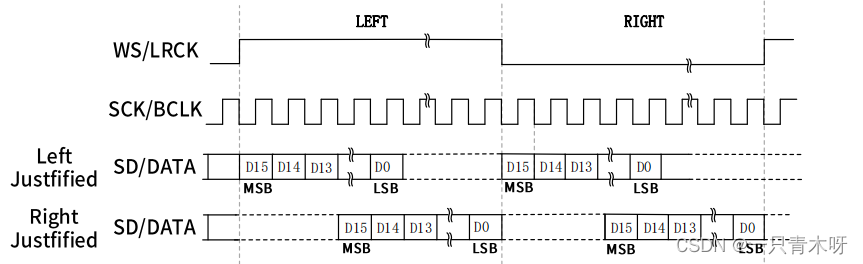


右对齐标准

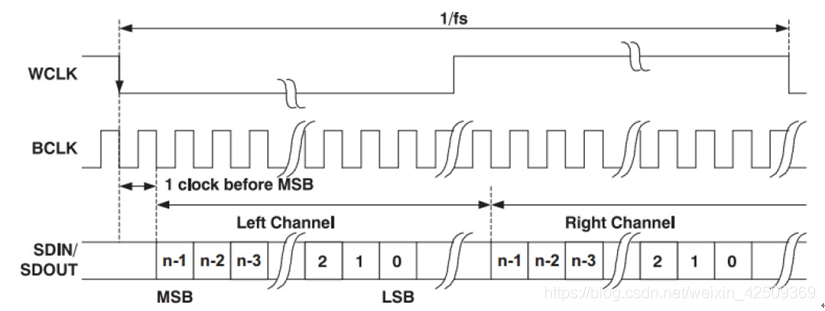
与左对齐标准类似，参考图 385，为24bit数据封装在32bit帧传输波形。



左对齐：数据的MSB在LRCLK边沿起第一个BCLK上升沿，用的比较少。  
右对齐：数据的LSB靠左LRCLK的上升沿，Sony使用这种格式。



IS的操作模式分为三种：标准IIS模式、左对齐模式和右对齐模式。



标准IIS模式   Phillips Standard  
IIS模式是标准左对齐格式再延迟一个时钟位变化来的，时序如下所示：

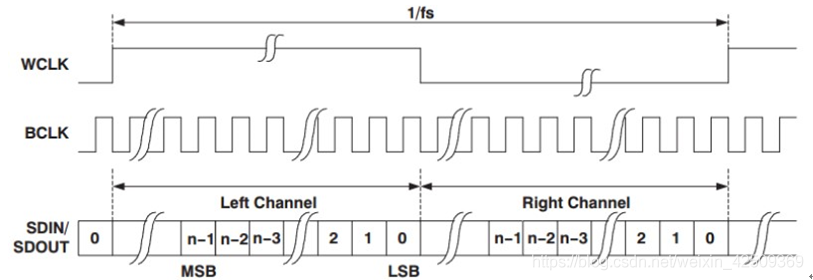
左右通道的数据MSB均是在WS变化后第二个SCK/BCLK上升沿有效。

左对齐模式  Left Justified Standard  
标准左对齐格式的数据的MSB没有相对于BCLK延迟一个时钟。左对齐格式的左右声道数据的MSB在WS边沿变化后SCK/BCLK的第一个上升沿有效。具体如下图所示：



支持16~32bit字长格式；

右边对齐模式  Right Justified Standard     
也叫日本格式，sony格式，具体对齐方式如下图所示：



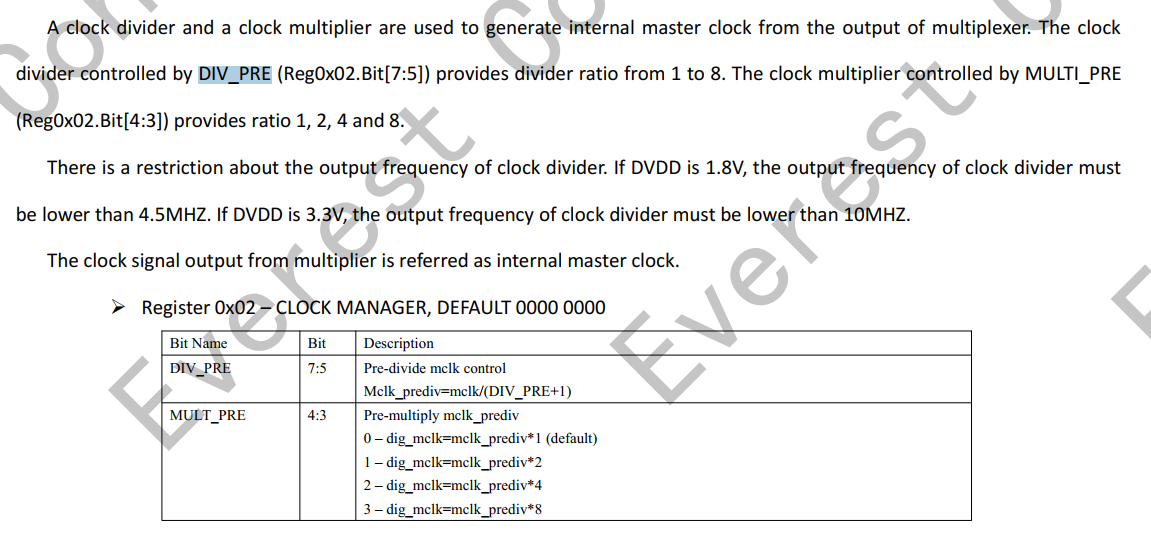
接收设备必须事先知道待传数据的字长。

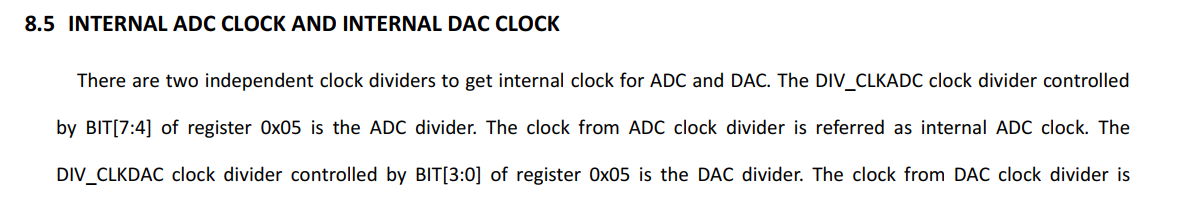
注意左右对齐模式的WS时钟高电平为左声道，低电平为右声道，刚好与标准IIS相反。

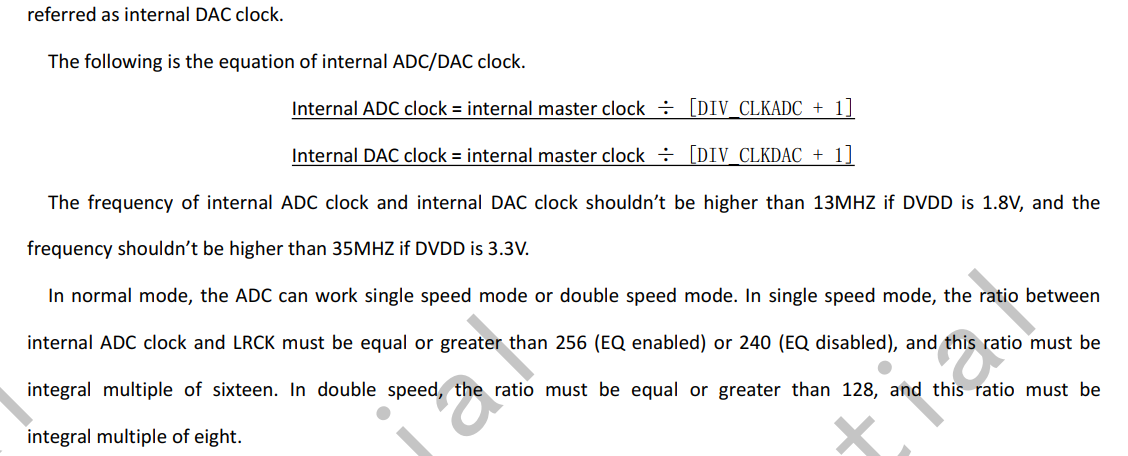
SCK = 采样率（48K、44.1K、16K等） x  字长（16bit、24bit、32bit） x 2（左右两通道）

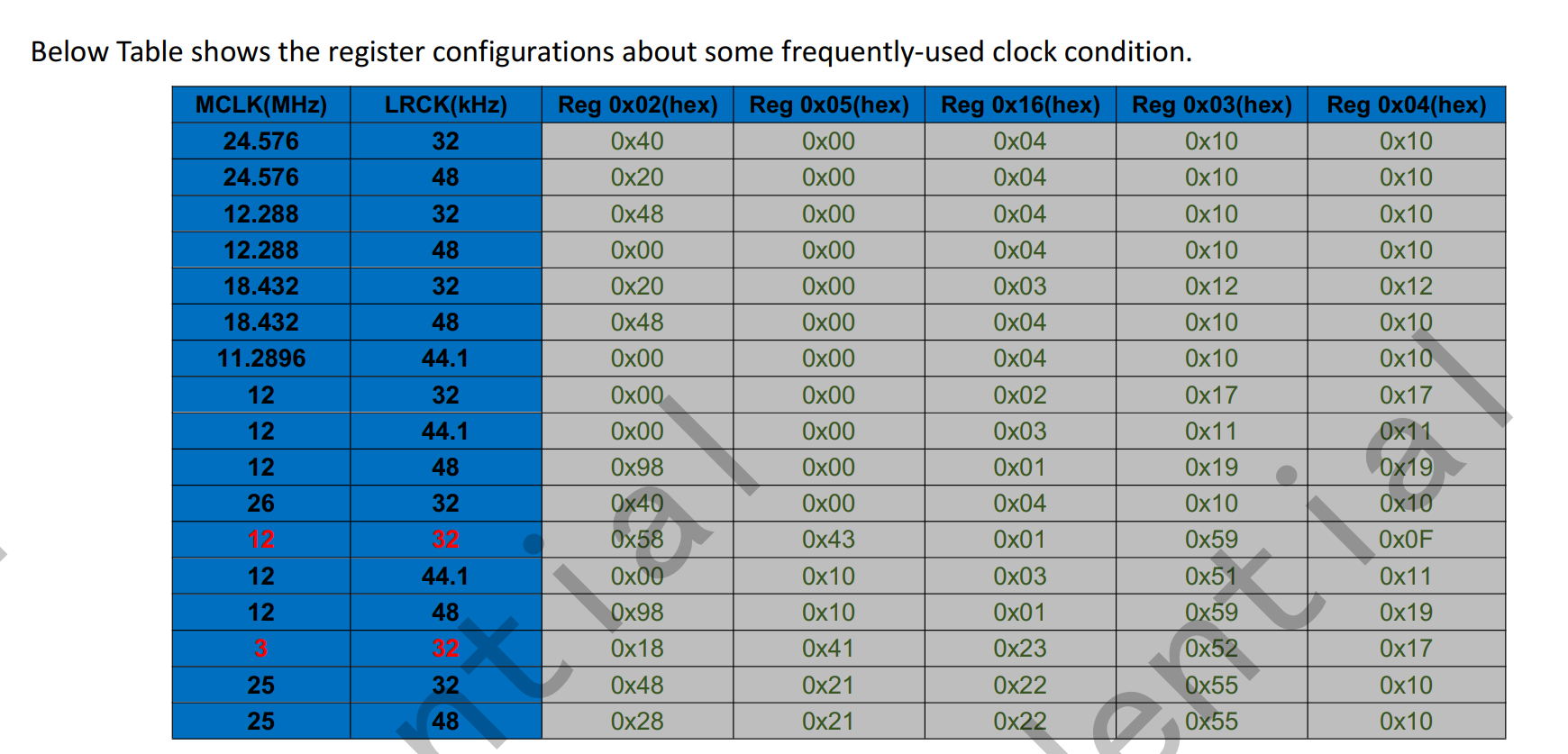
MCLK/SCK =  384 、256 等需要参考手册说明支持哪种；

来源： <https://bbs.huaweicloud.com/blogs/309313>









=====

{12288000, 8000 , 0x06, 0x01, 0x01, 0x01, 0x00, 0x00, 0xff, 0x04, 0x10, 0x10}

12288000, //mclk=12.288M

8000 , //rate  lrck/rate >= 256 --->内部master clk = 8000 \* 256 = 2048000  = mclk / 6 ---> 6 = prediv /premulti -->prediv:0x6   premulti:0x1

0x06, //prediv DIV\_PRE的取值:1 to 8; 寄存器reg02 bit[7:5] bit000--->bit111

0x01,//premulti  MULTI\_PRE的取值:1,2,4,8,寄存器：reg02 bit[4:3] bit00--->bit11

 0x01, //adcdiv    reg05 bit[7:4] 取值1->16 bit0000 --> bit1111 adc\_mclk=dig\_mclk/(DIV\_CLKADC+1)   dig\_mclk=mclk\_prediv\*MULT\_PRE= mclk/(DIV\_PRE+1)\*MULT\_PRE  =

0x01,//dacdiv    reg05 bit[3:0] 取值1->16 bit0000 --> bit1111 dac\_mclk=dig\_mclk/(DIV\_CLKDAC+1)    dac\_mclk/lrck >= 256

 0x00, //fsmode    0:单模式  1:双模式

0x00,//lrck\_h LRCK(master)=dig\_mclk/(LRCK\_DIV+1) ====》 master模式

 0xff, //lrck\_l LRCK(master)=dig\_mclk/(LRCK\_DIV+1)

0x04, //bclkdiv  dig\_mclk/(DIV\_BCLK+1) (default 3)

0x10, //adcosr 积分频率:64fs

0x10//dacosr 积分64fs

===

/\* 64k \*/

{

18432000, // mclk = 18432000

64000,  //rate --->radio = 18432000/64000 = 288

0x03, //--->3

0x04, //4

0x03, //adc mclk =mclk/3 \* 4 /3 =   --->rate = adc\_mclk/64000 = 128

0x01, //dac\_clk = mclk/3 \*4 =24,576,000   ---> rate = dac\_clk/64000 = 384

0x01,

 0x7f,

0x06,

0x10,

0x10

},

==

/\* 44.1k \*/

    {

11289600, //mclk = 11.2896M

 44100, //rate = 44100  ;internal master clk = rate \* 256 = 44100  \* 256 = 11,289,600 = mclk /1 ==> 1 =  prediv /premulti  = 1/1

0x01,  //prediv  =1

0x01,  //premulti   =1

0x01,    //

0x01,

0x00,

0x00,

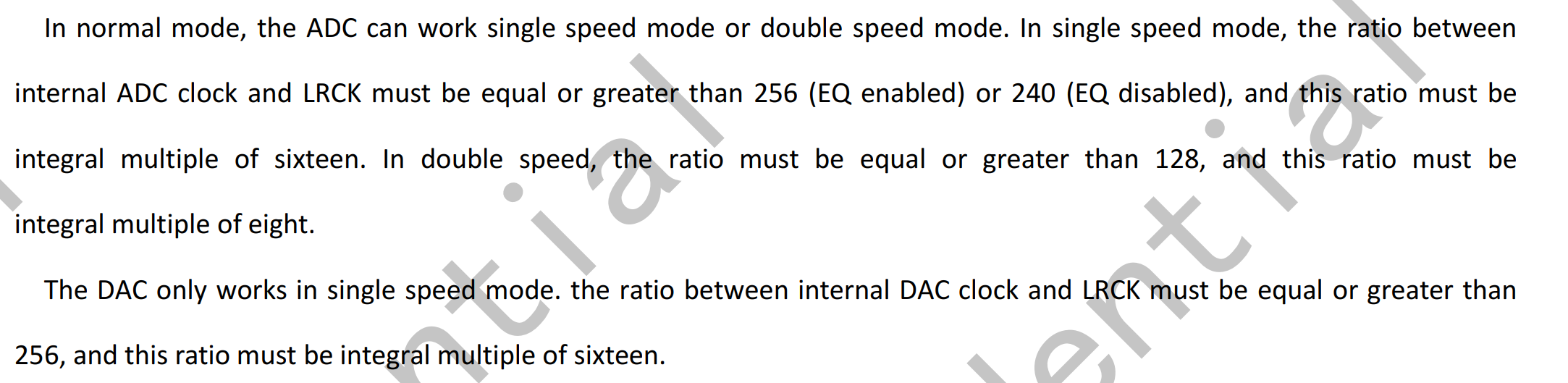
 0xff,

0x04,

0x10,

0x10},

=====





26000000,//mclk = 26M

48000// fs = 48000    internal master clk = 48000 \* 256 = 12288000 ; --> 12288000 = mclk/x == 26000000/x ---> x =

====

reg0x2 0x40--->   MULT\_PRE:1    DIV\_PRE:3

reg0x05 0x00 ----> 设置DIV\_CLKADC和DIV\_CLKDAC为0

reg0x16 0x04 ---> adc 增益 24db

reg0x03 0x10 ADC\_OSR:16-64fs

reg0x04 0x10 DAC\_OSR:16-64fs

26000000,//mclk = 26M

32000 ，// fs = 32000  --->    internal master clk = mclk \*  MULT\_PRE/(DIV\_PRE + 1) = mclk \* 2/(2+1) = mclk \* 2/3 =

3， //prediv bit[7:5] = 0x2  :reg0x2 0x40--->   MULT\_PRE:1    DIV\_PRE:3

1,// ---> 2 premulti bit[4:3] = 0x0

0x1, //adcdiv  reg0x05 0x00 ----> 设置DIV\_CLKADC和DIV\_CLKDAC为0

0x1,//dacdiv

0x00, //fsmode    0:单模式  1:双模式

0x10, //adcosr 积分频率:64fs

0x10//dacosr 积分64fs

{26000000, 32000, 0x03, 0x01, 0x01, 0x01, 0x00, 0x00, 0xff, 0x04, 0x10, 0x10},

{12288000, 32000, 0x03, 0x02, 0x01, 0x01, 0x00, 0x00, 0xff, 0x04, 0x10, 0x10},

{

12288000, //mclk = 12288000

44100, //rate = 44100   mclk/rate=12288000/44100 = 278.639

1,

1,

0x1,

0x1,

0x00,

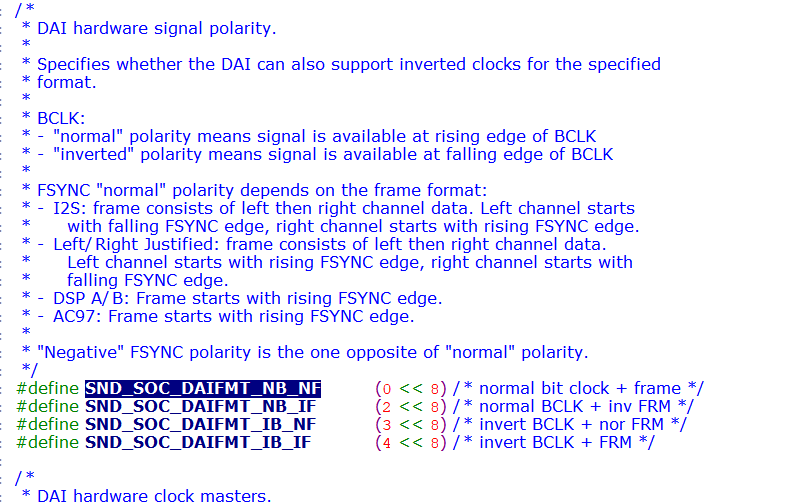
0x10,

0x10

}

===

关于时钟反正



26000000

==

其中的reg字段被设置为SND\_SOC\_NOPM（-1），表明这些widget是没有寄存器控制位来控制widget的电源状态的。麦克风，耳机，扬声器，线路输入接口这几

<https://zhuanlan.zhihu.com/p/537564029>

PRODUCT\_PACKAGES += \

    lights.$(TARGET\_BOARD\_PLATFORM) \

    sensors.$(TARGET\_BOARD\_PLATFORM) \

    tinymix \

PRODUCT\_PACKAGES += \

tinycap \

tinyplay \

tinyhostless \

tinypcminfo

ud710\_2h10:/proc/asound/card1 # tinymix -D 1

Mixer name: 'all-i2s'

Number of controls: 57

ctl type num name value

0 INT 1 MIC PGA GAIN 10

1 INT 1 ADC SCALE 4

2 ENUM 1 DMIC TYPE dmic at high level

3 INT 1 ADC RAMP RATE 0

4 BOOL 1 ADC SDP MUTE On

5 BOOL 1 ADC INVERTED Off

6 BOOL 1 ADC SYNC Off

7 BOOL 1 ADC RAM CLR Off

8 INT 1 ADC VOLUME 191

9 BOOL 1 ALC ENABLE Off

10 ENUM 1 ALC AUTOMUTE TYPE automute disabled

11 INT 1 ALC WIN SIZE 0

12 INT 1 ALC MAX LEVEL 0

13 INT 1 ALC MIN LEVEL 0

14 INT 1 ALC AUTOMUTE WINSIZE 0

15 INT 1 ALC AUTOMUTE GATE THRESHOLD 0

16 INT 1 ALC AUTOMUTE VOLUME 0

17 BOOL 1 ADC FS MODE Off

18 INT 1 ADC OSR 16

19 BOOL 1 DAC SDP MUTE Off

20 BOOL 1 DAC DEM MUTE Off

21 BOOL 1 DAC INVERT Off

22 BOOL 1 DAC RAM CLR Off

23 ENUM 1 DAC DSM MUTE mute to 8

24 INT 1 DAC OFFSET 0

25 INT 1 DAC VOLUME 0

26 BOOL 1 DRC ENABLE Off

27 INT 1 DRC WIN SIZE 0

28 INT 1 DRC MAX LEVEL 0

29 INT 1 DRC MIN LEVEL 0

30 INT 1 DAC RAMP RATE 0

31 INT 1 DAC OSR 16

32 ENUM 1 AEC MODE adc left, adc right

33 ENUM 1 ADC DATA TO DAC TEST MODE disable

34 BOOL 1 MCLK INVERT Off

35 BOOL 1 BCLK INVERT Off

36 ENUM 1 MCLK SOURCE from mclk pin

37 INT 1 fs 32000

38 INT 1 hw\_port 0

39 INT 1 slave\_timeout 3857

40 INT 1 bus\_type 0

41 INT 1 byte\_per\_chan 1

42 INT 1 mode 0

43 INT 1 lsb 0

44 INT 1 rtx\_mode 3

45 INT 1 lrck\_inv 1

46 INT 1 sync\_mode 1

47 INT 1 clk\_inv 0

48 INT 1 i2s\_bus\_mode 1

49 INT 1 pcm\_bus\_mode 0

50 INT 1 pcm\_slot 0

51 INT 1 pcm\_cycle 0

52 INT 1 tx\_watermark 12

53 INT 1 rx\_watermark 20

54 ENUM 1 DMIC MUX DMIC DISABLE

55 ENUM 1 SDP OUT MUX FROM ADC OUT

56 ENUM 1 DAC SDP SRC MUX SELECT SDP LEFT DATA

录音前:

ud710\_2h10:/data # tinymix -D 1

Mixer name: 'all-i2s'

Number of controls: 57

ctl type num name value

0 INT 1 MIC PGA GAIN 10

1 INT 1 ADC SCALE 4

2 ENUM 1 DMIC TYPE dmic at high level

3 INT 1 ADC RAMP RATE 0

4 BOOL 1 ADC SDP MUTE On

5 BOOL 1 ADC INVERTED Off

6 BOOL 1 ADC SYNC Off

7 BOOL 1 ADC RAM CLR Off

8 INT 1 ADC VOLUME 191

9 BOOL 1 ALC ENABLE Off

10 ENUM 1 ALC AUTOMUTE TYPE automute disabled

11 INT 1 ALC WIN SIZE 0

12 INT 1 ALC MAX LEVEL 0

13 INT 1 ALC MIN LEVEL 0

14 INT 1 ALC AUTOMUTE WINSIZE 0

15 INT 1 ALC AUTOMUTE GATE THRESHOLD 0

16 INT 1 ALC AUTOMUTE VOLUME 0

17 BOOL 1 ADC FS MODE Off

18 INT 1 ADC OSR 16

19 BOOL 1 DAC SDP MUTE Off

20 BOOL 1 DAC DEM MUTE Off

21 BOOL 1 DAC INVERT Off

22 BOOL 1 DAC RAM CLR Off

23 ENUM 1 DAC DSM MUTE mute to 8

24 INT 1 DAC OFFSET 0

25 INT 1 DAC VOLUME 0

26 BOOL 1 DRC ENABLE Off

27 INT 1 DRC WIN SIZE 0

28 INT 1 DRC MAX LEVEL 0

29 INT 1 DRC MIN LEVEL 0

30 INT 1 DAC RAMP RATE 0

31 INT 1 DAC OSR 16

32 ENUM 1 AEC MODE adc left, adc right

33 ENUM 1 ADC DATA TO DAC TEST MODE disable

34 BOOL 1 MCLK INVERT Off

35 BOOL 1 BCLK INVERT Off

36 ENUM 1 MCLK SOURCE from mclk pin

37 INT 1 fs 32000

38 INT 1 hw\_port 0

39 INT 1 slave\_timeout 3857

40 INT 1 bus\_type 0

41 INT 1 byte\_per\_chan 1

42 INT 1 mode 0

43 INT 1 lsb 0

44 INT 1 rtx\_mode 3

45 INT 1 lrck\_inv 1

46 INT 1 sync\_mode 1

47 INT 1 clk\_inv 0

48 INT 1 i2s\_bus\_mode 1

49 INT 1 pcm\_bus\_mode 0

50 INT 1 pcm\_slot 0

51 INT 1 pcm\_cycle 0

52 INT 1 tx\_watermark 12

53 INT 1 rx\_watermark 20

54 ENUM 1 DMIC MUX DMIC DISABLE

55 ENUM 1 SDP OUT MUX FROM ADC OUT

56 ENUM 1 DAC SDP SRC MUX SELECT SDP LEFT DATA

录音后

ud710\_2h10:/ # tinymix -D 1

Mixer name: 'all-i2s'

Number of controls: 57

ctl type num name value

0 INT 1 MIC PGA GAIN 10

1 INT 1 ADC SCALE 4

2 ENUM 1 DMIC TYPE dmic at high level

3 INT 1 ADC RAMP RATE 4

4 BOOL 1 ADC SDP MUTE On

5 BOOL 1 ADC INVERTED Off

6 BOOL 1 ADC SYNC Off

7 BOOL 1 ADC RAM CLR Off

8 INT 1 ADC VOLUME 191

9 BOOL 1 ALC ENABLE Off

10 ENUM 1 ALC AUTOMUTE TYPE automute disabled

11 INT 1 ALC WIN SIZE 0

12 INT 1 ALC MAX LEVEL 0

13 INT 1 ALC MIN LEVEL 0

14 INT 1 ALC AUTOMUTE WINSIZE 0

15 INT 1 ALC AUTOMUTE GATE THRESHOLD 0

16 INT 1 ALC AUTOMUTE VOLUME 0

17 BOOL 1 ADC FS MODE Off

18 INT 1 ADC OSR 16

19 BOOL 1 DAC SDP MUTE Off

20 BOOL 1 DAC DEM MUTE Off

21 BOOL 1 DAC INVERT Off

22 BOOL 1 DAC RAM CLR Off

23 ENUM 1 DAC DSM MUTE mute to 8

24 INT 1 DAC OFFSET 0

25 INT 1 DAC VOLUME 191

26 BOOL 1 DRC ENABLE Off

27 INT 1 DRC WIN SIZE 0

28 INT 1 DRC MAX LEVEL 0

29 INT 1 DRC MIN LEVEL 0

30 INT 1 DAC RAMP RATE 4

31 INT 1 DAC OSR 16

32 ENUM 1 AEC MODE adc left, adc right

33 ENUM 1 ADC DATA TO DAC TEST MODE disable

34 BOOL 1 MCLK INVERT Off

35 BOOL 1 BCLK INVERT Off

36 ENUM 1 MCLK SOURCE from mclk pin

37 INT 1 fs 32000

38 INT 1 hw\_port 0

39 INT 1 slave\_timeout 3857

40 INT 1 bus\_type 0

41 INT 1 byte\_per\_chan 1

42 INT 1 mode 0

43 INT 1 lsb 0

44 INT 1 rtx\_mode 3

45 INT 1 lrck\_inv 1

46 INT 1 sync\_mode 1

47 INT 1 clk\_inv 0

48 INT 1 i2s\_bus\_mode 1

49 INT 1 pcm\_bus\_mode 0

50 INT 1 pcm\_slot 0

51 INT 1 pcm\_cycle 0

52 INT 1 tx\_watermark 12

53 INT 1 rx\_watermark 20

54 ENUM 1 DMIC MUX DMIC DISABLE

55 ENUM 1 SDP OUT MUX FROM ADC OUT

56 ENUM 1 DAC SDP SRC MUX SELECT SDP LEFT DATA

====

0x69000

iis\_intf1  ----> ap\_iis1

bit[9:6] = 0x1

110 1001 0000 0100 0000

[   77.834979] c5 libprocessgroup: Successfully killed process cgroup uid 1000 pid 5358 in 0ms

[   82.603364] c0 [ASoC: I2S ] i2s config hw\_port 1

[   82.608492] c4 Bad mode in Error handler detected on CPU4, code 0xbe000011 -- SError

[   82.616127] c4 Internal error: Oops - bad mode: 0 [#1] PREEMPT SMP

[   82.622268] c4 dump\_die\_cb in.

[   82.625294] c4 dump\_die\_cb save pregs\_die\_g ok .

[   82.629879] c4 dump\_die\_cb out.

===

iis0:

{REG\_PIN\_IIS0DI,                        BITS\_PIN\_AF(0)},

{REG\_MISC\_PIN\_IIS0DI,                   BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_AP|BIT\_PIN\_SLP\_WPD|BIT\_PIN\_SLP\_IE},//BT\_PCM\_OUT

{REG\_PIN\_IIS0DO,                        BITS\_PIN\_AF(0)},

{REG\_MISC\_PIN\_IIS0DO,                   BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_AP|BIT\_PIN\_SLP\_NUL|BIT\_PIN\_SLP\_OE},//BT\_PCM\_IN

{REG\_PIN\_IIS0CLK,                       BITS\_PIN\_AF(0)},

{REG\_MISC\_PIN\_IIS0CLK,                  BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_AP|BIT\_PIN\_SLP\_WPD|BIT\_PIN\_SLP\_IE},//BT\_PCM\_CLK

{REG\_PIN\_IIS0LRCK,                      BITS\_PIN\_AF(0)},

{REG\_MISC\_PIN\_IIS0LRCK,                 BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_AP|BIT\_PIN\_SLP\_WPD|BIT\_PIN\_SLP\_IE},//BT\_PCM\_SYNC

iis1:

ud710-20c10:

{REG\_PIN\_IIS1DI,                        BITS\_PIN\_AF(0)},

{REG\_MISC\_PIN\_IIS1DI,                   BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_AUDCP|BIT\_PIN\_SLP\_WPD|BIT\_PIN\_SLP\_IE},//SPKR\_I2S\_DIN

{REG\_PIN\_IIS1DO,                        BITS\_PIN\_AF(0)},

{REG\_MISC\_PIN\_IIS1DO,                   BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_AUDCP|BIT\_PIN\_SLP\_NUL|BIT\_PIN\_SLP\_OE},//SPKR\_I2S\_DOUT

{REG\_PIN\_IIS1CLK,                       BITS\_PIN\_AF(0)},

{REG\_MISC\_PIN\_IIS1CLK,                  BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_AUDCP|BIT\_PIN\_SLP\_WPD|BIT\_PIN\_SLP\_IE},//SPKR\_I2S\_BCK

{REG\_PIN\_IIS1LRCK,                      BITS\_PIN\_AF(0)},

{REG\_MISC\_PIN\_IIS1LRCK,                 BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_AUDCP|BIT\_PIN\_SLP\_WPD|BIT\_PIN\_SLP\_IE},//SPKR\_I2S\_WS

iis3:

{REG\_PIN\_RFCTL\_5,                       BITS\_PIN\_AF(1)},

{REG\_MISC\_PIN\_RFCTL\_5,                  BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_AUDCP|BIT\_PIN\_SLP\_WPD|BIT\_PIN\_SLP\_IE},//AP\_I2S3\_SDI

{REG\_PIN\_RFCTL\_6,                       BITS\_PIN\_AF(3)},

{REG\_MISC\_PIN\_RFCTL\_6,                  BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_WPU|BIT\_PIN\_SLP\_AP|BIT\_PIN\_SLP\_WPU|BIT\_PIN\_SLP\_OE},////audio pa(GPIO25)

{REG\_PIN\_RFCTL\_7,                       BITS\_PIN\_AF(1)},

{REG\_MISC\_PIN\_RFCTL\_7,                  BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_AUDCP|BIT\_PIN\_SLP\_NUL|BIT\_PIN\_SLP\_OE},//AP\_I2S3\_LRCK

{REG\_PIN\_RFCTL\_8,                       BITS\_PIN\_AF(1)},

{REG\_MISC\_PIN\_RFCTL\_8,                  BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_AUDCP|BIT\_PIN\_SLP\_WPD|BIT\_PIN\_SLP\_IE},//AP\_I2S3\_SCLK

===

0x69000

110 1001 0000 0000 0000

bit[3:0] = 0000 ---> iis\_inf0 = ap\_iis0

bit4 = 0  intf0--> intf1 loop=0

bit5 = 0  intf0 ---> intf2 loop = 0

bit[9:6] = 0000 ---> iis\_intf1 = ap\_iis0

bit[10] = 0 ---> iis\_inf1 ---> intf2 loop = 0

bit[14:11] = 0010 ---> iis\_inf2 ---> ap\_iis2

bit[18:15] = 1101 --->iis\_intf3-->audcp\_iis0

---->

修改为

0x69001

====================

0x69040-->110 1001 0000 0100 0000

==

主mic录音

tinymix -D 0 "VBC ADCL DG Switch" 1

CONFIG\_SND\_SOC\_SPRD\_AUDIO\_TWO\_STAGE\_DMAENGINE\_SURPPORT

作用？？

===

{REG\_PIN\_IIS\_MATRIX\_MTX\_CFG,0x00069001},//IIS0->AP\_IIS1; IIS1->AP\_IIS0; IIS2->AP\_IIS2; IIS3->AUDCP\_IIS0;

{REG\_PIN\_IIS\_MATRIX\_MTX\_CFG,0x00069001},//IIS0->AP\_IIS1; IIS1->AP\_IIS0; IIS2->AP\_IIS2; IIS3->AUDCP\_IIS0;

==

问题：音频i2s mclk和lrck没有时钟输出

使用iis1(对应gpio130 gpio131 gpio132 gpio133 )接音频codec     t710通过这个音频codec 与天通做语音交互

uboot设置REG\_PIN\_IIS\_MATRIX\_MTX\_CFG的IIS1->AP\_IIS0

{REG\_PIN\_IIS\_MATRIX\_MTX\_CFG,0x00069001},//IIS0->AP\_IIS1; IIS1->AP\_IIS0; IIS2->AP\_IIS2; IIS3->AUDCP\_IIS0;

设置gpio为iis

{REG\_PIN\_IIS1DI,                        BITS\_PIN\_AF(0)},

{REG\_MISC\_PIN\_IIS1DI,                   BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_WPD|BIT\_PIN\_SLP\_ALL|BIT\_PIN\_SLP\_WPD|BIT\_PIN\_SLP\_IE},//FTID\_INT --->  codec iis iis1di

{REG\_PIN\_IIS1DO,                        BITS\_PIN\_AF(0)},

{REG\_MISC\_PIN\_IIS1DO,                   BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_ALL|BIT\_PIN\_SLP\_NUL|BIT\_PIN\_SLP\_OE},//codec iis IIS1DO

{REG\_PIN\_IIS1CLK,                       BITS\_PIN\_AF(0)},

{REG\_MISC\_PIN\_IIS1CLK,                  BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_ALL|BIT\_PIN\_SLP\_NUL|BIT\_PIN\_SLP\_OE},//codec iis IIS1CLK

{REG\_PIN\_IIS1LRCK,                      BITS\_PIN\_AF(0)},

{REG\_MISC\_PIN\_IIS1LRCK,                 BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_ALL|BIT\_PIN\_SLP\_NUL|BIT\_PIN\_SLP\_OE},//codec iis IIS1LRCLK

在内核把codec注册在声卡1上:

&sound\_sprd\_ap\_alliis {

        status = "okay";

        /\* dai-links \*/

        sprd-audio-card,dai-link@1 {

                plat {

                        sound-dai = <&sprd\_pcm\_iis>;

                };

                cpu {

                        sound-dai = <&i2s0>;

                };

                codec {

                        sound-dai = <&es8311 0>;

                };

        };

};

设置了i2s0的格式

&i2s0 {

        status = "okay";

        sprd,config\_type = "i2s";

        sprd,slave\_timeout = <0xf11>;

        sprd,hw\_port = <0>;

        //sprd,fs = <48000>;

        sprd,fs = <32000>;

        sprd,bus\_type = <0>;

        sprd,rtx\_mode = <3>;

        sprd,byte\_per\_chan = <1>;

        sprd,slave\_mode = <0>;

        sprd,lsb = <0>;

        sprd,lrck = <0>; /\*sync\_mode\*/

        sprd,low\_for\_left = <0>; /\*lrck\_inv\*/

        //sprd,clk\_inv = <1>;

        sprd,clk\_inv = <0>;

        sprd,pcm\_short\_frame = <0>; /\*pcm\_bus\_mode\*/

        sprd,pcm\_slot = <0x1>;

        sprd,pcm\_cycle = <1>;

       // sprd,tx\_watermark = <12>;

       // sprd,rx\_watermark = <20>;

        sprd,tx\_watermark = <16>;

        sprd,rx\_watermark = <16>;

        sprd,i2s\_compatible= <1>;

};

内核sprd\_roc1\_defconfig添加

CONFIG\_SND\_SOC\_SPRD\_I2S=y

CONFIG\_SND\_SOC\_SPRD\_I2S\_DUMMY=y

开机后检测到声卡注册成功了：

ud710\_2h10:/proc/asound/card1/pcm1c # cat info

card: 1

device: 1

subdevice: 0

stream: CAPTURE

id: i2s\_bt\_sco0-ES8311 HiFi ES8311 HiFi-1

name:

subname: subdevice #0

class: 0

subclass: 0

subdevices\_count: 1

subdevices\_avail: 1

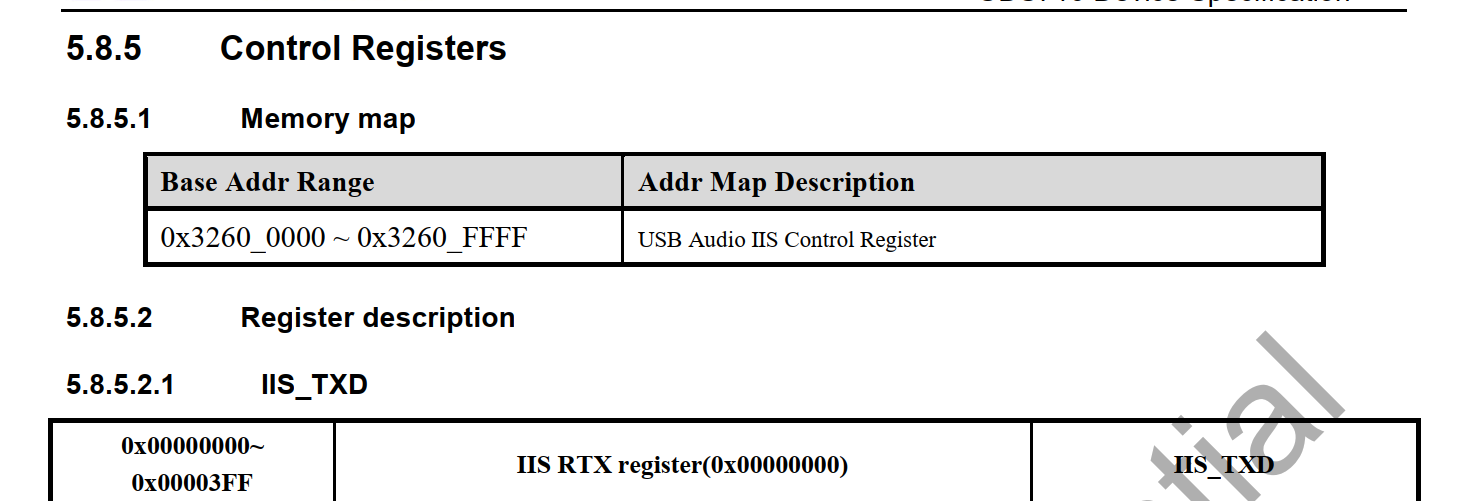
使用tinycap录音 也没报错

tinycap  /data/rec5.wav -D 1 -d 1 -c 2 -r 32000  -T 80

但是录音到的数据 播放不了，同时示波器量 mclk和lrck没有时钟输出。

==

usb 音频控制:



AON 上的MST\_IIS

arch\_audio\_iis\_to\_audio\_top\_enable

/\* AGCP IIS multiplexer setting.

 \* @iis: the iis channel to be set.

 \* @en:

 \*   0: AG\_IIS0\_EXT\_SEL to whale2 top

 \*   1: AG\_IIS0\_EXT\_SEL to audio top

 \*/

static inline int arch\_

===

sprd,syscon-agcp-ahb = <&audcp\_ahb\_regs>;

audcp\_ahb\_regs: syscon@0x335e0000 {

IMG_344

adb  push F:\work\2022-work\t710-pad\tinycap  /system/bin/tinycap

FE\_NORMAL\_AP01

.stream\_name = "FE\_DAI\_NORMAL\_AP01\_P",

            .aif\_name = "FE\_IF\_NORMAL\_AP01\_P",

udx710

udx710\_4h10\_nse

252 {REG\_PIN\_IIS0DI, BITS\_PIN\_AF(0)},

253 {REG\_MISC\_PIN\_IIS0DI, BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_AUDCP|BIT\_PIN\_SLP\_WPD|BIT\_PIN\_SLP\_IE},//I2S0\_SDO(GPIO45)

254 {REG\_PIN\_IIS0DO, BITS\_PIN\_AF(0)},

255 {REG\_MISC\_PIN\_IIS0DO, BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_AUDCP|BIT\_PIN\_SLP\_NUL|BIT\_PIN\_SLP\_OE},//I2S0\_SDI(GPIO44)

256 {REG\_PIN\_IIS0CLK, BITS\_PIN\_AF(3)},

257 {REG\_MISC\_PIN\_IIS0CLK, BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_WPD|BIT\_PIN\_SLP\_AP|BIT\_PIN\_SLP\_WPD|BIT\_PIN\_SLP\_IE},//I2S0\_SCLK(GPIO46) mofiy by yangjia

258 {REG\_PIN\_IIS0LRCK, BITS\_PIN\_AF(0)},

259 {REG\_MISC\_PIN\_IIS0LRCK, BITS\_PIN\_DS(1)|BIT\_PIN\_NULL|BIT\_PIN\_NUL|BIT\_PIN\_SLP\_AUDCP|BIT\_PIN\_SLP\_WPD|BIT\_PIN\_SLP\_OE},//I2S0\_LRCK(GPIO47)

260 {REG\_PIN\_SCL0, BITS\_PIN\_AF(0)},

CONFIG\_SND\_SMARTPA\_AW881XX

CONFIG\_SND\_SOC\_AW87XX\_IIC\_PA

==

tinycap /data/rec.wav -D -0 -d 0 -r 16000 -b 16 -T 10

tinycap /data/rec.wav -D -0 -d 0 -r 16000 -b 16 -T 10

<double-mic device="0x80000084">

<on>

<ctl name="ADC LRCLK Select" val="invert"/>

<ctl name="ADC1 LRCLK Select" val="invert"/>

<ctl name="MIC Boost" val="1"/>

<ctl name="AUXMIC Boost" val="1"/>

<ctl name="ADCR Mixer AuxMICADCR Switch" val="1"/>

<ctl name="ADCL Mixer MainMICADCL Switch" val="1"/>

<ctl name="AUD ADC0L Switch" val="1" />

<ctl name="AUD ADC0R Switch" val="1" />

<ctl name="Aux Mic Function" val="1"/>

<ctl name="Mic Function" val="1"/>

</on>

tinymix "ADC1 LRCLK Select" "invert"

tinymix "ADC LRCLK Select" "invert"

tinymix "ADC1 LRCLK Select" "invert"

tinymix "MIC Boost" "1"

tinymix "AUXMIC Boost" "1"

tinymix "ADCR Mixer AuxMICADCR Switch" "1"

tinymix "ADCL Mixer MainMICADCL Switch" "1"

tinymix "AUD ADC0L Switch" "1"

tinymix "AUD ADC0R Switch" "1"

tinymix "Aux Mic Function" "1"

tinymix "Mic Function" "1"

tinymix "ADC LRCLK Select"                  "invert"

tinymix "ADC1 LRCLK Select"              "invert"

tinymix "MIC Boost"                      "1"

tinymix "AUXMIC Boost"                      "1"

tinymix "ADCR Mixer AuxMICADCR Switch"   "1"

tinymix "ADCL Mixer MainMICADCL Switch"  "1"

tinymix "AUD ADC0L Switch"                  "1"

tinymix "AUD ADC0R Switch"                  "1"

tinymix "Aux Mic Function"                  "1"

tinymix "Mic Function"                      "1"

==

audio\_pcm.xml

<mm\_normal channels="2" rate="48000" period\_size="960" period\_count="4" format="0" card="sprdphone-sc2730"  device="3"/>

format:

audio\_format\_t

AUDIO\_FORMAT\_MP3

AUDIO\_FORMAT\_PCM\_SUB\_16\_BIT

card:声卡

device：pcm

//

====

**声卡测试：**  
① 编译工具：**mmm external/tinyalsa/**  
② 播放：tinyplay file.wav [-D card] [-d device] [-p period\_size] [-n n\_periods]  
　　        **tinyplay /sdcard/test.wav -D 0 -d 0 -p 1024 -n 3**  
③ 录音：**tinycap /sdcard/rec.wav -D 0 -d 0 -c 2 -r 44100 -b 16 -p 1024 -n  3**

**来源：**<https://www.cnblogs.com/blogs-of-lxl/p/6538769.html>

enum {

AUDIO\_DEVICE\_NONE = 0x0,

/\* reserved bits \*/

AUDIO\_DEVICE\_BIT\_IN = 0x80000000,

AUDIO\_DEVICE\_BIT\_DEFAULT = 0x40000000,

/\* output devices \*/

AUDIO\_DEVICE\_OUT\_EARPIECE = 0x1, // 听筒

AUDIO\_DEVICE\_OUT\_SPEAKER = 0x2, // 扬声器

AUDIO\_DEVICE\_OUT\_WIRED\_HEADSET = 0x4, // 线控耳机，可以通过耳机控制远端播放、暂停、音量调节等功能的耳机

AUDIO\_DEVICE\_OUT\_WIRED\_HEADPHONE = 0x8, // 普通耳机，只能听，不能操控播放

AUDIO\_DEVICE\_OUT\_BLUETOOTH\_SCO = 0x10, // 单声道蓝牙耳机，十进制32

AUDIO\_DEVICE\_OUT\_BLUETOOTH\_SCO\_HEADSET = 0x20, // 车载免提蓝牙设备，十进制64

AUDIO\_DEVICE\_OUT\_BLUETOOTH\_SCO\_CARKIT = 0x40, // 立体声蓝牙耳机

AUDIO\_DEVICE\_OUT\_BLUETOOTH\_A2DP = 0x80, // 十进制128

AUDIO\_DEVICE\_OUT\_BLUETOOTH\_A2DP\_HEADPHONES = 0x100, // 十进制256

AUDIO\_DEVICE\_OUT\_BLUETOOTH\_A2DP\_SPEAKER = 0x200, // 十进制512

AUDIO\_DEVICE\_OUT\_AUX\_DIGITAL = 0x400, // 十进制1024

AUDIO\_DEVICE\_OUT\_ANLG\_DOCK\_HEADSET = 0x800, // 十进制2048

AUDIO\_DEVICE\_OUT\_DGTL\_DOCK\_HEADSET = 0x1000, // 十进制4096

AUDIO\_DEVICE\_OUT\_USB\_ACCESSORY = 0x2000,

AUDIO\_DEVICE\_OUT\_USB\_DEVICE = 0x4000,

AUDIO\_DEVICE\_OUT\_REMOTE\_SUBMIX = 0x8000,

AUDIO\_DEVICE\_OUT\_DEFAULT = AUDIO\_DEVICE\_BIT\_DEFAULT,

AUDIO\_DEVICE\_OUT\_ALL = (AUDIO\_DEVICE\_OUT\_EARPIECE |

AUDIO\_DEVICE\_OUT\_SPEAKER |

AUDIO\_DEVICE\_OUT\_WIRED\_HEADSET |

AUDIO\_DEVICE\_OUT\_WIRED\_HEADPHONE |

AUDIO\_DEVICE\_OUT\_BLUETOOTH\_SCO |

AUDIO\_DEVICE\_OUT\_BLUETOOTH\_SCO\_HEADSET |

AUDIO\_DEVICE\_OUT\_BLUETOOTH\_SCO\_CARKIT |

AUDIO\_DEVICE\_OUT\_BLUETOOTH\_A2DP |

AUDIO\_DEVICE\_OUT\_BLUETOOTH\_A2DP\_HEADPHONES |

AUDIO\_DEVICE\_OUT\_BLUETOOTH\_A2DP\_SPEAKER |

AUDIO\_DEVICE\_OUT\_AUX\_DIGITAL |

AUDIO\_DEVICE\_OUT\_ANLG\_DOCK\_HEADSET |

AUDIO\_DEVICE\_OUT\_DGTL\_DOCK\_HEADSET |

AUDIO\_DEVICE\_OUT\_USB\_ACCESSORY |

AUDIO\_DEVICE\_OUT\_USB\_DEVICE |

AUDIO\_DEVICE\_OUT\_REMOTE\_SUBMIX |

AUDIO\_DEVICE\_OUT\_DEFAULT),

AUDIO\_DEVICE\_OUT\_ALL\_A2DP = (AUDIO\_DEVICE\_OUT\_BLUETOOTH\_A2DP |

AUDIO\_DEVICE\_OUT\_BLUETOOTH\_A2DP\_HEADPHONES |

AUDIO\_DEVICE\_OUT\_BLUETOOTH\_A2DP\_SPEAKER),

AUDIO\_DEVICE\_OUT\_ALL\_SCO = (AUDIO\_DEVICE\_OUT\_BLUETOOTH\_SCO |

AUDIO\_DEVICE\_OUT\_BLUETOOTH\_SCO\_HEADSET |

AUDIO\_DEVICE\_OUT\_BLUETOOTH\_SCO\_CARKIT),

AUDIO\_DEVICE\_OUT\_ALL\_USB = (AUDIO\_DEVICE\_OUT\_USB\_ACCESSORY |

AUDIO\_DEVICE\_OUT\_USB\_DEVICE),

/\* input devices \*/

AUDIO\_DEVICE\_IN\_COMMUNICATION = AUDIO\_DEVICE\_BIT\_IN | 0x1,

AUDIO\_DEVICE\_IN\_AMBIENT = AUDIO\_DEVICE\_BIT\_IN | 0x2,

AUDIO\_DEVICE\_IN\_BUILTIN\_MIC = AUDIO\_DEVICE\_BIT\_IN | 0x4,　　//手机自带MIC

AUDIO\_DEVICE\_IN\_BLUETOOTH\_SCO\_HEADSET = AUDIO\_DEVICE\_BIT\_IN | 0x8,

AUDIO\_DEVICE\_IN\_WIRED\_HEADSET = AUDIO\_DEVICE\_BIT\_IN | 0x10,　　//耳机

AUDIO\_DEVICE\_IN\_AUX\_DIGITAL = AUDIO\_DEVICE\_BIT\_IN | 0x20,

AUDIO\_DEVICE\_IN\_VOICE\_CALL = AUDIO\_DEVICE\_BIT\_IN | 0x40,

AUDIO\_DEVICE\_IN\_BACK\_MIC = AUDIO\_DEVICE\_BIT\_IN | 0x80,

AUDIO\_DEVICE\_IN\_REMOTE\_SUBMIX = AUDIO\_DEVICE\_BIT\_IN | 0x100,

AUDIO\_DEVICE\_IN\_ANLG\_DOCK\_HEADSET = AUDIO\_DEVICE\_BIT\_IN | 0x200,

AUDIO\_DEVICE\_IN\_DGTL\_DOCK\_HEADSET = AUDIO\_DEVICE\_BIT\_IN | 0x400,

AUDIO\_DEVICE\_IN\_USB\_ACCESSORY = AUDIO\_DEVICE\_BIT\_IN | 0x800,

AUDIO\_DEVICE\_IN\_USB\_DEVICE = AUDIO\_DEVICE\_BIT\_IN | 0x1000,

AUDIO\_DEVICE\_IN\_DEFAULT = AUDIO\_DEVICE\_BIT\_IN | AUDIO\_DEVICE\_BIT\_DEFAULT,

AUDIO\_DEVICE\_IN\_ALL = (AUDIO\_DEVICE\_IN\_COMMUNICATION |

AUDIO\_DEVICE\_IN\_AMBIENT |

AUDIO\_DEVICE\_IN\_BUILTIN\_MIC |

AUDIO\_DEVICE\_IN\_BLUETOOTH\_SCO\_HEADSET |

AUDIO\_DEVICE\_IN\_WIRED\_HEADSET |

AUDIO\_DEVICE\_IN\_AUX\_DIGITAL |

AUDIO\_DEVICE\_IN\_VOICE\_CALL |

AUDIO\_DEVICE\_IN\_BACK\_MIC |

AUDIO\_DEVICE\_IN\_REMOTE\_SUBMIX |

AUDIO\_DEVICE\_IN\_ANLG\_DOCK\_HEADSET |

AUDIO\_DEVICE\_IN\_DGTL\_DOCK\_HEADSET |

AUDIO\_DEVICE\_IN\_USB\_ACCESSORY |

AUDIO\_DEVICE\_IN\_USB\_DEVICE |

AUDIO\_DEVICE\_IN\_DEFAULT),

AUDIO\_DEVICE\_IN\_ALL\_SCO = AUDIO\_DEVICE\_IN\_BLUETOOTH\_SCO\_HEADSET,

};

==========

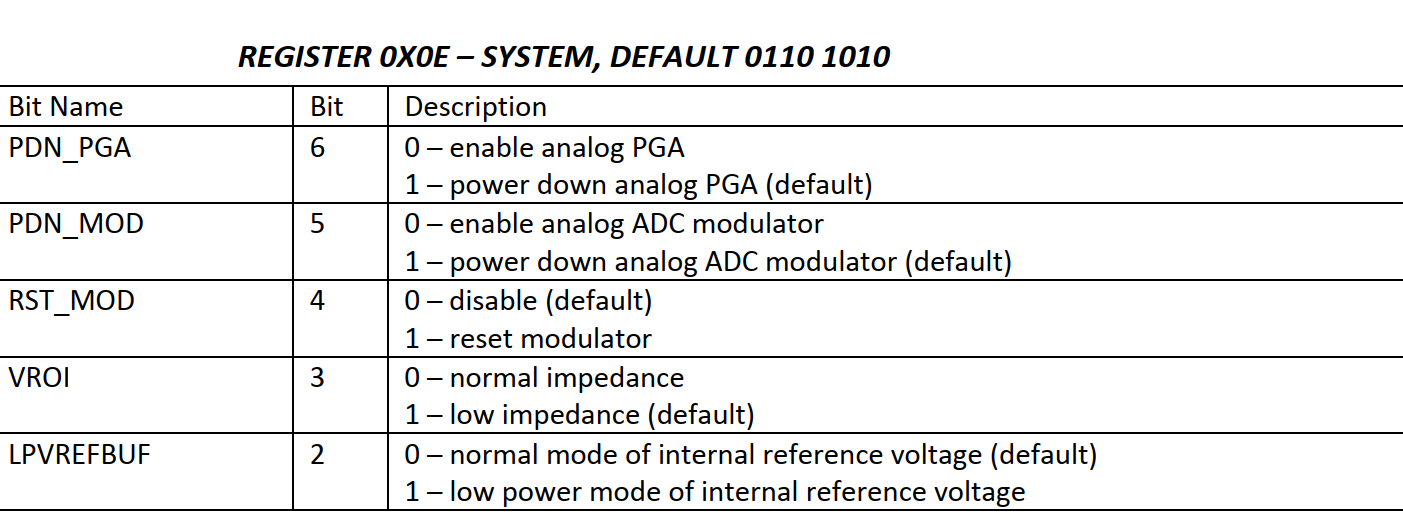
es8311的adc 增益相关

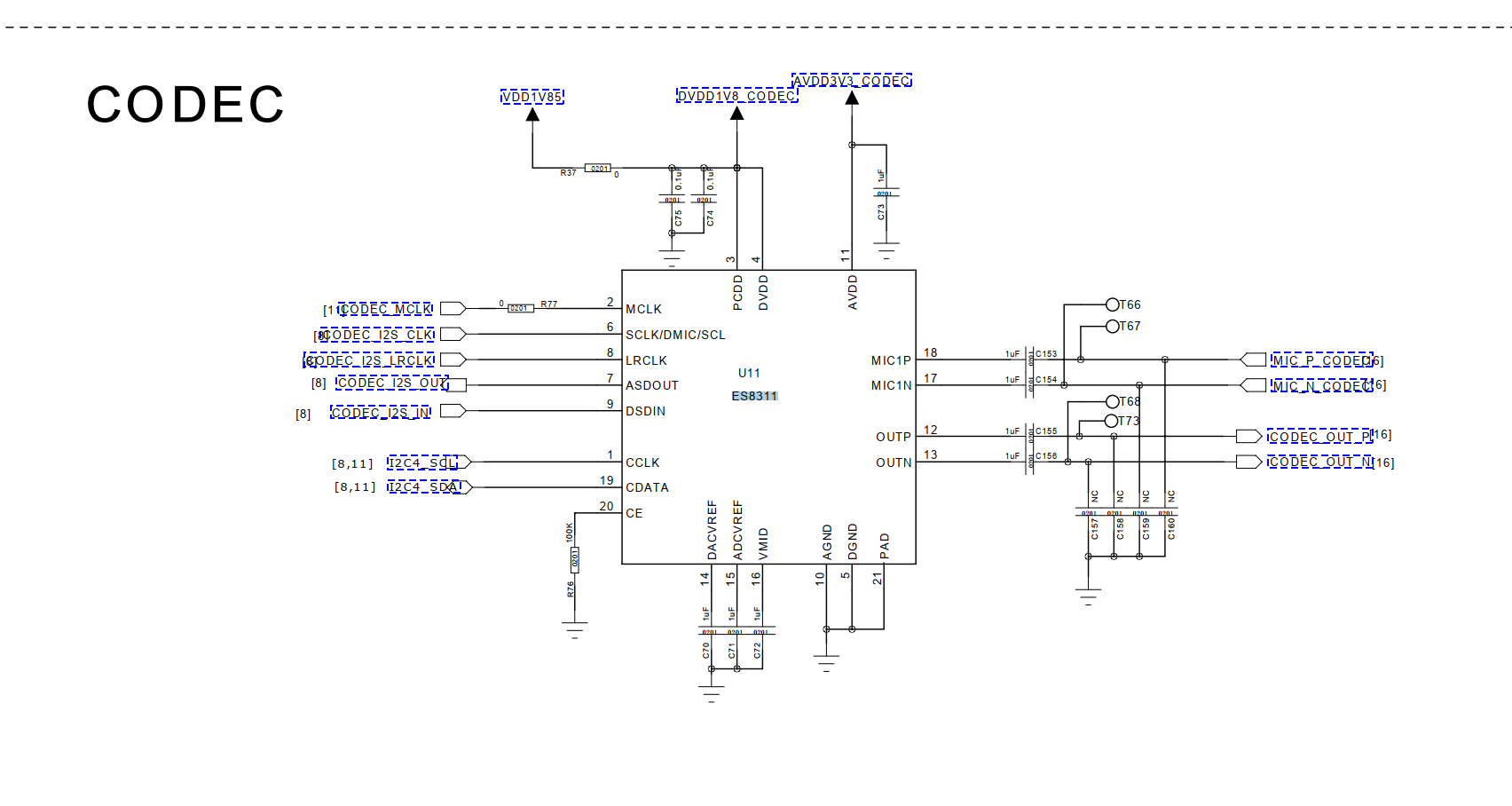
====

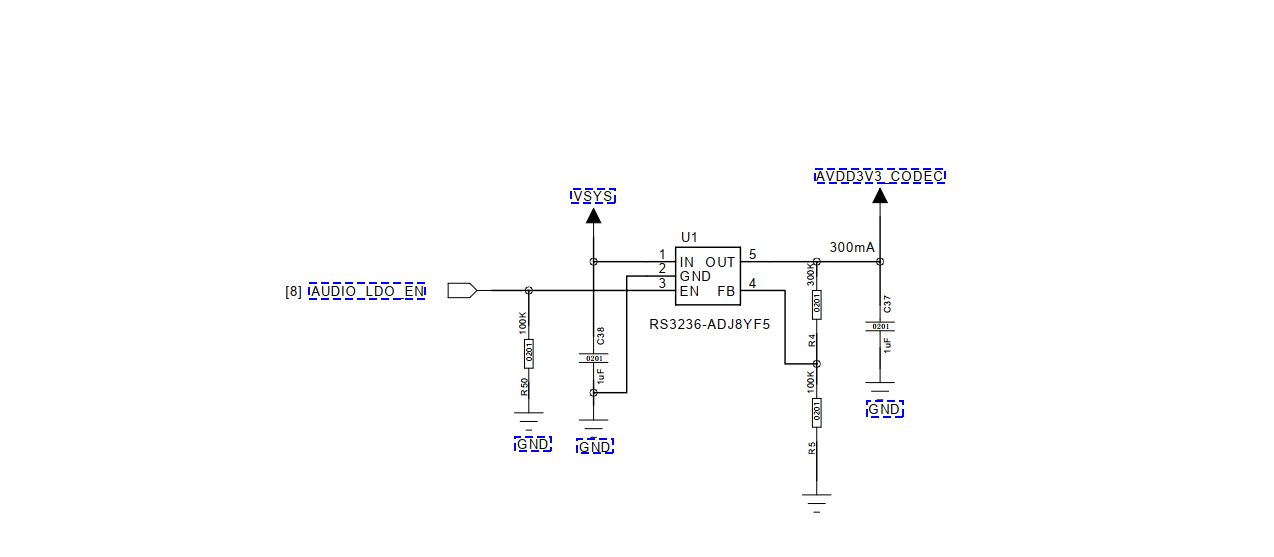
不能播放的原因：

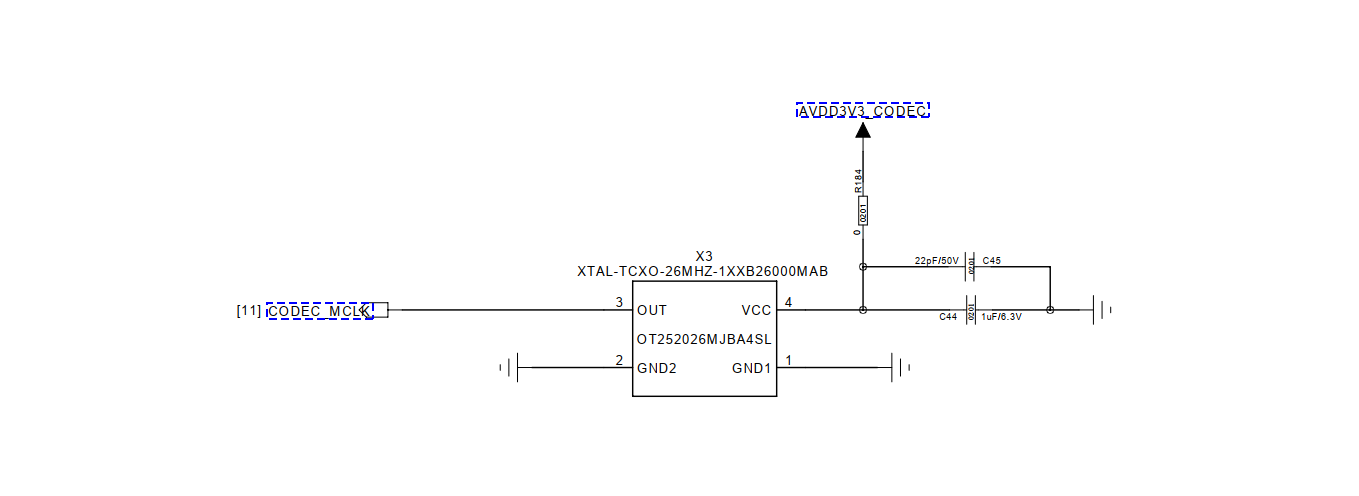
正确的设置为02









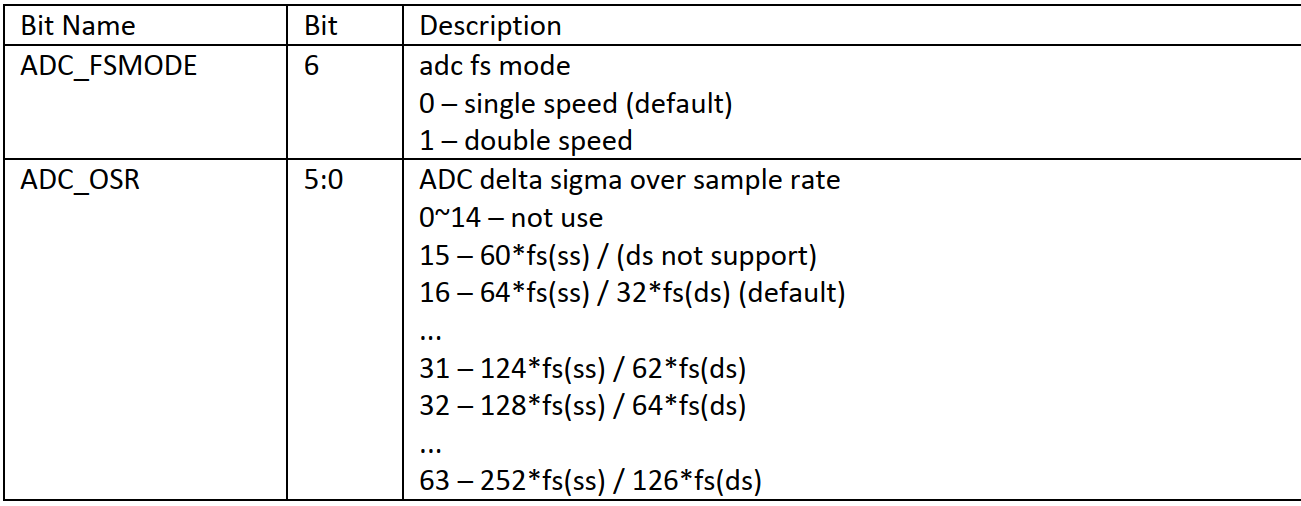


====

[ 2751.435173] c4 Enter into es8311\_pcm\_hw\_params()

[ 2751.439692] c4 stephen es8311\_pcm\_hw\_params es8311->mclk:12288000,rate:16000

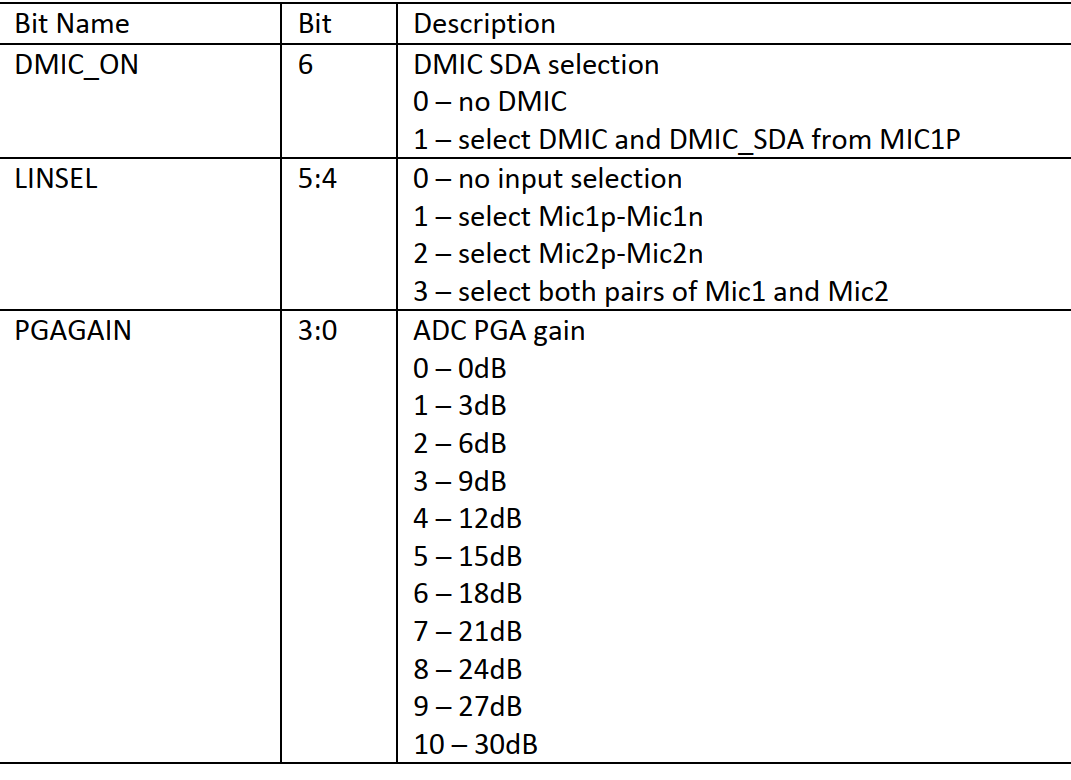
==\



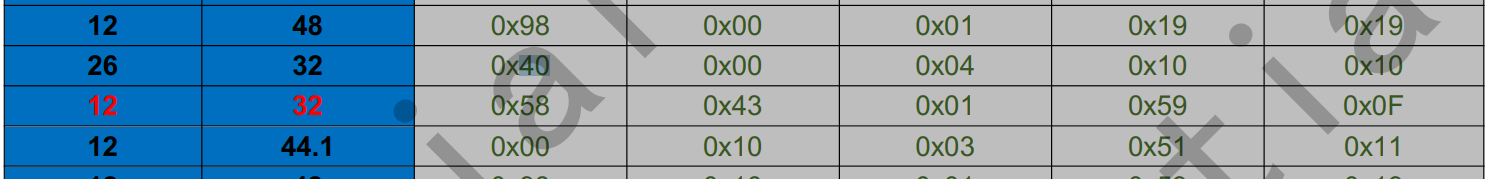
echo 1033f > es8311  ===>没有效果

03: 3f

===



echo 1143a > es8311



====

adc ---> dac设置

你把0x44 改为0x88

从8311的输出端听听

===

把BCLK当MCLK用

这几个寄存器改一下

0X00---0X80

0x01----0x9F

0X02---0X10

echo 1019f > es8311

echo 10210 > es8311

echo 10080 > es8311

===

./tinycap cap.wav -D 1 -d 1 -c 2 -r 48000 -b 16

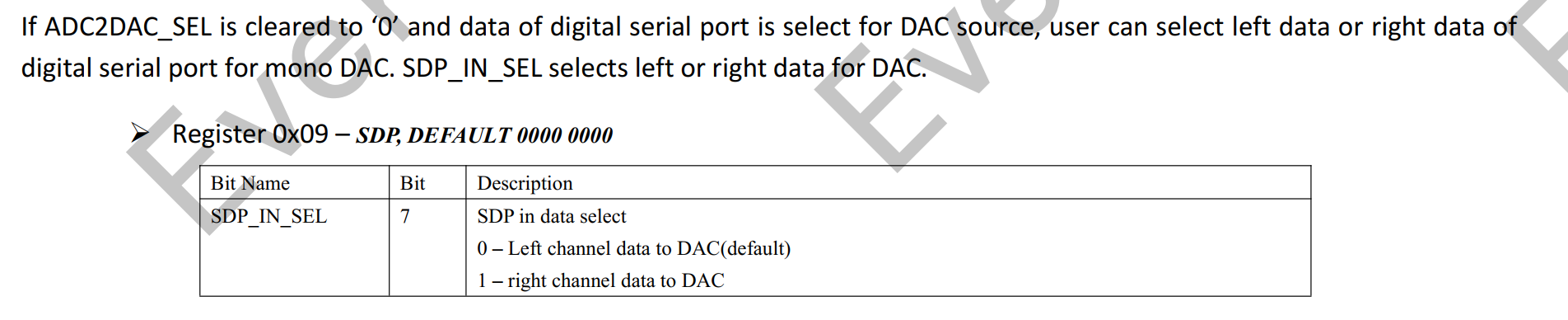
./tinyplay music-48000.wav  -D 1 -d 1

====

0x0d -->0x1

echo 10d01 > es8311

===



==

1100

1000 1100

echo 1098c > es8311

1：写

2:09寄存器

3:8c寄存器的值

====

es8311操作:

ud710\_2h10:/sys # find  ./  -name  \*es8311\*

./kernel/debug/asoc/all-i2s/codec:es8311.4-0018

./devices/platform/soc/soc:ap-apb/70700000.i2c/i2c-4/4-0018/es8311\_debug

./devices/platform/soc/soc:ap-apb/70700000.i2c/i2c-4/4-0018/es8311\_debug/es8311

./bus/i2c/drivers/es8311

./firmware/devicetree/base/soc/ap-apb/i2c@70700000/es8311@0x18

./firmware/devicetree/base/\_\_symbols\_\_/es8311

===