es8311\_codec\_config(AUDIO\_HAL\_44K\_SAMPLES);//AUDIO\_HAL\_44K\_SAMPLES /\*!< set to 44.1k samples per second \*/

audio\_hal\_codec\_config\_t cfg = {

.adc\_input = AUDIO\_HAL\_ADC\_INPUT\_LINE1, //adc 输入选择/\*!< mic input to adc channel 1 \*/

.dac\_output = AUDIO\_HAL\_DAC\_OUTPUT\_LINE1, //dac输出选择: /\*!< dac output signal to channel 1 \*/

.codec\_mode = AUDIO\_HAL\_CODEC\_MODE\_BOTH,//codec模式:/\*!< select both adc and dac \*/

.i2s\_iface = {

.mode = AUDIO\_HAL\_MODE\_SLAVE,//es8311 slave 模式

.fmt = AUDIO\_HAL\_I2S\_NORMAL,//标准i2s格式

.samples = sample\_rate, //采样率:44k

.bits = AUDIO\_HAL\_BIT\_LENGTH\_16BITS,//通道数据宽:16bit

},

};

2:es8311\_codec\_init(&cfg);

|es8311\_write\_reg(ES8311\_CLK\_MANAGER\_REG01, 0x30);//时钟设置mclk bclk adc/dac clk Mclk:从mclk pin mclk bclk开启

|es8311\_write\_reg(ES8311\_CLK\_MANAGER\_REG02, 0x00);//设置时钟clk的分频 设置DIV\_PRE MULT\_PRE

|es8311\_write\_reg(ES8311\_CLK\_MANAGER\_REG03, 0x10);//设置采样模式：单速度 和双速度 以及采样率:64fs(ss) 32fs(ds)

|es8311\_write\_reg(ES8311\_ADC\_REG16, 0x24);//设置为标准的音频adc的同步时钟 audio clk

标准的 音频时钟:64fs,128fs,256fs,384fs,512fs. 非标准时钟16Mhz 25Mhz 26Mhz

|es8311\_write\_reg(ES8311\_CLK\_MANAGER\_REG04, 0x10);//设置DAC的采样率

|es8311\_write\_reg(ES8311\_CLK\_MANAGER\_REG05, 0x00);//设置dac和adc的时钟分频

|es8311\_write\_reg(ES8311\_SYSTEM\_REG0B, 0x00);//pwrup stage a b

|es8311\_write\_reg(ES8311\_SYSTEM\_REG0C, 0x00);//pwrup stage b c

|es8311\_write\_reg(ES8311\_SYSTEM\_REG10, 0x1F);//设置sync模式和偏置电压

|es8311\_write\_reg(ES8311\_SYSTEM\_REG11, 0x7F);

|es8311\_write\_reg(ES8311\_RESET\_REG00, 0x80);//上电 并设置为slave

3：设置模式slave

|regv = es8311\_read\_reg(ES8311\_RESET\_REG00);

|regv &= 0xBF;

|es8311\_write\_reg(ES8311\_RESET\_REG00, regv);

设置时钟

|es8311\_write\_reg(ES8311\_CLK\_MANAGER\_REG01, 0x3F);//时钟设置 mclk

mclk时钟源从mclk pin

|regv = es8311\_read\_reg(ES8311\_CLK\_MANAGER\_REG01);

|regv &= 0x7F;

|ret |= es8311\_write\_reg(ES8311\_CLK\_MANAGER\_REG01, regv);

mclk时钟源从sclk pin

|regv = es8311\_read\_reg(ES8311\_CLK\_MANAGER\_REG01);

|regv |= 0x80;

|ret |= es8311\_write\_reg(ES8311\_CLK\_MANAGER\_REG01, regv);

时钟设置

//AUDIO\_HAL\_44K\_SAMPLES #define MCLK\_DIV\_FRE 256

|sample\_fre = 44100;

|mclk\_fre = sample\_fre \* MCLK\_DIV\_FRE; = 44100 \* 256 = 1289600

|coeff = get\_coeff(mclk\_fre, sample\_fre); 查表获取参数

|设置时钟

|es8311\_write\_reg(ES8311\_SYSTEM\_REG13, 0x10);

|es8311\_write\_reg(ES8311\_ADC\_REG1B, 0x0A);

|es8311\_write\_reg(ES8311\_ADC\_REG1C, 0x6A);

es8311\_set\_bits\_per\_sample(cfg.i2s\_iface.bits);

.bits = AUDIO\_HAL\_BIT\_LENGTH\_16BITS,//通道数据宽:16bit

|dac\_iface |= 0x0c;

|adc\_iface |= 0x0c;

|es8311\_write\_reg(ES8311\_SDPIN\_REG09, dac\_iface)

|es8311\_write\_reg(ES8311\_SDPOUT\_REG0A, adc\_iface)

es8311\_codec\_set\_voice\_volume(60) //设置音量60

|int vol = (volume) \* 2550 / 1000;

|es8311\_write\_reg(ES8311\_DAC\_REG32, vol);

es8311\_set\_mic\_gain(ES8311\_MIC\_GAIN\_30DB); //设置mic增益

|es8311\_write\_reg(ES8311\_ADC\_REG16, gain\_db); // MIC gain scale

es8311\_codec\_ctrl\_state(cfg.codec\_mode, AUDIO\_HAL\_CTRL\_START);

//codec\_mode = AUDIO\_HAL\_CODEC\_MODE\_BOTH

//AUDIO\_HAL\_CTRL\_START /\*!< set start mode \*/

|es\_mode = ES\_MODULE\_ADC\_DAC;

|es8311\_start(es\_mode);

|adc\_iface &= ~(BIT(6));

|dac\_iface &= ~(BIT(6));

|es8311\_write\_reg(ES8311\_SDPIN\_REG09, dac\_iface);

|es8311\_write\_reg(ES8311\_SDPOUT\_REG0A, adc\_iface);

|es8311\_write\_reg(ES8311\_ADC\_REG17, 0xBF);

|es8311\_write\_reg(ES8311\_SYSTEM\_REG0E, 0x02);

|es8311\_write\_reg(ES8311\_SYSTEM\_REG12, 0x00);

|es8311\_write\_reg(ES8311\_SYSTEM\_REG14, 0x1A);

非DMIC

|regv = es8311\_read\_reg(ES8311\_SYSTEM\_REG14);

|regv &= ~(0x40);

|ret |= es8311\_write\_reg(ES8311\_SYSTEM\_REG14, regv);

|es8311\_write\_reg(ES8311\_SYSTEM\_REG0D, 0x01);

|es8311\_write\_reg(ES8311\_ADC\_REG15, 0x40);

|es8311\_write\_reg(ES8311\_DAC\_REG37, 0x48);

|es8311\_write\_reg(ES8311\_GP\_REG45, 0x00);

T710 es8311

es8311\_i2c\_probe

|es8311->dmic\_enable = false; //dmic 关闭

/\* the edge of lrck is always at the falling edge of mclk \*/

|es8311->mclkinv = false;

|es8311->sclkinv = false;

|es8311->mclk = 26000000; //add stephen

soc\_codec\_dev\_es8311

1:es8311\_probe

|snd\_soc\_codec\_set\_drvdata(codec, es8311);

|codec->control\_data = es8311->regmap;

|es8311\_codec = codec;

|es8311->codec = codec;

|snd\_soc\_write(codec, ES8311\_GP\_REG45, 0x00);//上拉 BCLK和lrck

|snd\_soc\_write(codec, ES8311\_CLK\_MANAGER\_REG01, 0x30);//BCLK 和MCLK 时钟打开 其他时钟关闭

|snd\_soc\_write(codec, ES8311\_CLK\_MANAGER\_REG02, 0x00);

|snd\_soc\_write(codec, ES8311\_CLK\_MANAGER\_REG03, 0x10);

|snd\_soc\_write(codec, ES8311\_ADC\_REG16, 0x24);//设置为标准时钟源 adc 24db增益

|snd\_soc\_write(codec, ES8311\_CLK\_MANAGER\_REG04, 0x10);//设置DAC\_OSR

|snd\_soc\_write(codec, ES8311\_CLK\_MANAGER\_REG05, 0x00);

|snd\_soc\_write(codec, ES8311\_SYSTEM\_REG0B, 0x00);

|snd\_soc\_write(codec, ES8311\_SYSTEM\_REG0C, 0x00);

|snd\_soc\_write(codec, ES8311\_SYSTEM\_REG10, 0x1F);

|snd\_soc\_write(codec, ES8311\_SYSTEM\_REG11, 0x7F);

|snd\_soc\_write(codec, ES8311\_RESET\_REG00, 0x80);

|snd\_soc\_write(codec, ES8311\_SYSTEM\_REG0D, 0x01);

|snd\_soc\_write(codec, ES8311\_CLK\_MANAGER\_REG01, 0x3F);

|snd\_soc\_write(codec, ES8311\_SYSTEM\_REG14, 0x1A);

|snd\_soc\_write(codec, ES8311\_SYSTEM\_REG12, 0x00);

|snd\_soc\_write(codec, ES8311\_SYSTEM\_REG13, 0x10);

snd\_soc\_write(codec, ES8311\_SDPIN\_REG09, 0x00);

snd\_soc\_write(codec, ES8311\_SDPOUT\_REG0A, 0x00);

snd\_soc\_write(codec, ES8311\_SYSTEM\_REG0E, 0x02);

snd\_soc\_write(codec, ES8311\_ADC\_REG15, 0x40);

snd\_soc\_write(codec, ES8311\_ADC\_REG1B, 0x0A);

snd\_soc\_write(codec, ES8311\_ADC\_REG1C, 0x6A);

snd\_soc\_write(codec, ES8311\_DAC\_REG37, 0x48);

snd\_soc\_write(codec, ES8311\_GPIO\_REG44, 0x08);

snd\_soc\_write(codec, ES8311\_ADC\_REG17, 0xBF);

snd\_soc\_write(codec, ES8311\_DAC\_REG32, 0xBF);

|es8311\_set\_bias\_level(codec, SND\_SOC\_BIAS\_STANDBY);