

AC608X 软件问题整理

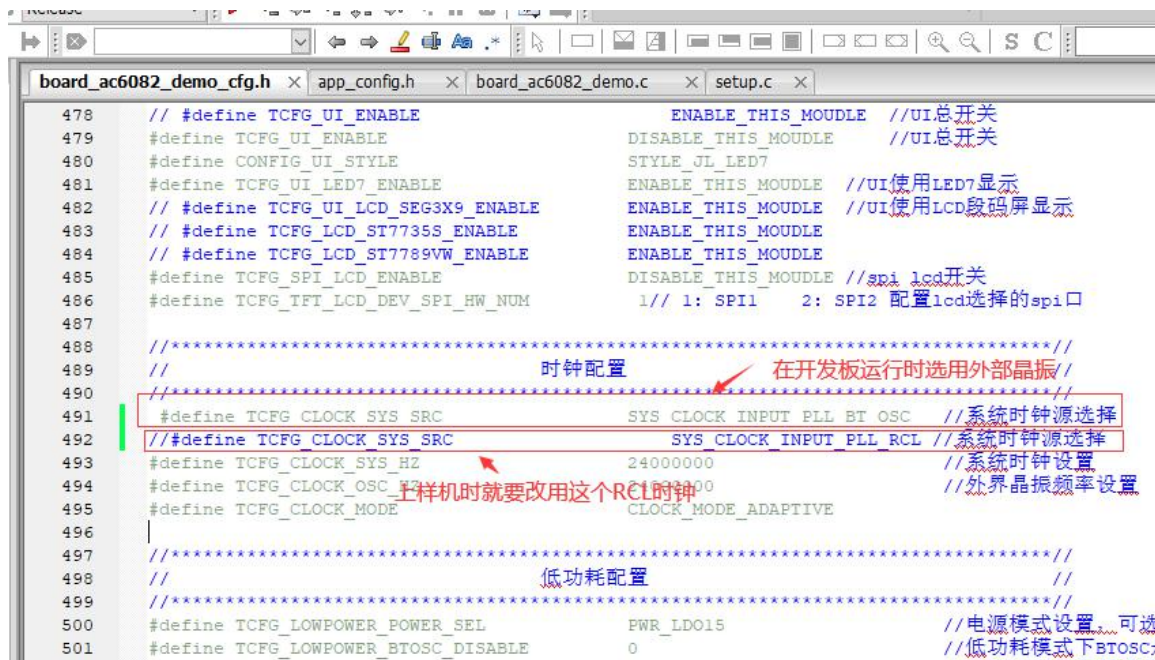
链接: <https://pan.baidu.com/s/1kjhBSPTfegAm3xRpuVv8NA>

提取码: fxq3

1.AC608X 在 AC696X 开发板上运行时若出现打印乱码、工作不正常需修改配置用外挂晶振	WTS	
20201016.....		2
2.AC608 的芯片在进行调式前需要使用一拖二烧写器烧写,校准过 LRC 之后方能使用,升级工具需用 V4.0 版本	WTS	
20201016.....		3
3.AC608X SDK1.0.0 卡和 U 盘同时在线时,需要按两次设备切换按键才能切换设备问题修改	WTS	
20201016.....		4
4.AC608X SDK1.0.0 外挂 Flash 使用无法读取 id 问题注意点	LAQ	
20201020.....		5
5.AC608X SDK1.0.0 设置为单声道后,有的机子有时播放 U 盘或卡音乐声音非常小解决方法	---WTS	
20201022.....		6
6.AC608X SDK1.0.0 无法使用红外按键注意点	LAQ	
20201028.....		6
7.6082A、6082B 共用上层板烧录注意点,芯片封装差异,烧录上层版需要跳电阻选择芯片型号,如下图:		
20201029 LZT.....		8
8.AC608X SDK1.0.0 无法播放隐藏文件修改方法	---WTS	
20201102.....		9
9.AC608X SDK1.0.0 在播放 WAV 音频文件时,卡口无法做复用修改方法	---WTS	
20201106.....		10
10.AC608X SDK1.0.0 录音暂停修改方法修改方法	---WTS	
20201106.....		10
11.AC6082A/B 原理图使用注意	20201110 YWP.....	14
12.AC608N 配置 PWM 注意点和参考资料	20201111 WTS.....	14
13.AC608X SDK1.0.0 空白 U 盘不能录音问题处理	20201111 LAQ.....	16
14.AC608N SDK1.0.0 上下曲切歌有破声问题需要更换的文件	20201111 WTS.....	17
15.AC608N SDK1.0.0 两张含有加密文件的 SD 卡(U 盘)轮流插入,由于簇号相同导致第二张插入的 SD 卡(U 盘)根据上一个设备的断点播放的处理方法	20201116 LJW.....	18
16.AC608N SDK1.0.0 录音模式播放录音文件时插入 U 盘播歌,再回录音模式播放录音文件时无法播放问题	20201119 LAQ.....	19
17.AC608N SDK1.0.0 无缝循环多次播放音乐时,第一遍和第二遍之间出现衔接不好的问题解决方案(包括一些注意点)	20201119 LAQ.....	21
18.AC608N SDK1.0.0 外挂 flash 录音录到 fat,并在 music 模式下播放录音文件夹的方法	20201124 FSW.....	22
19.AC608N SDK1.0.0 没有提示音,开机直接进 FM/LINEIN 模式声音小处理方法	20201126 LZK.....	26
20.AC608N /AC696 LINEIN 没声音或系统音量超过 24 没声音补充	20201130 LHY.....	26

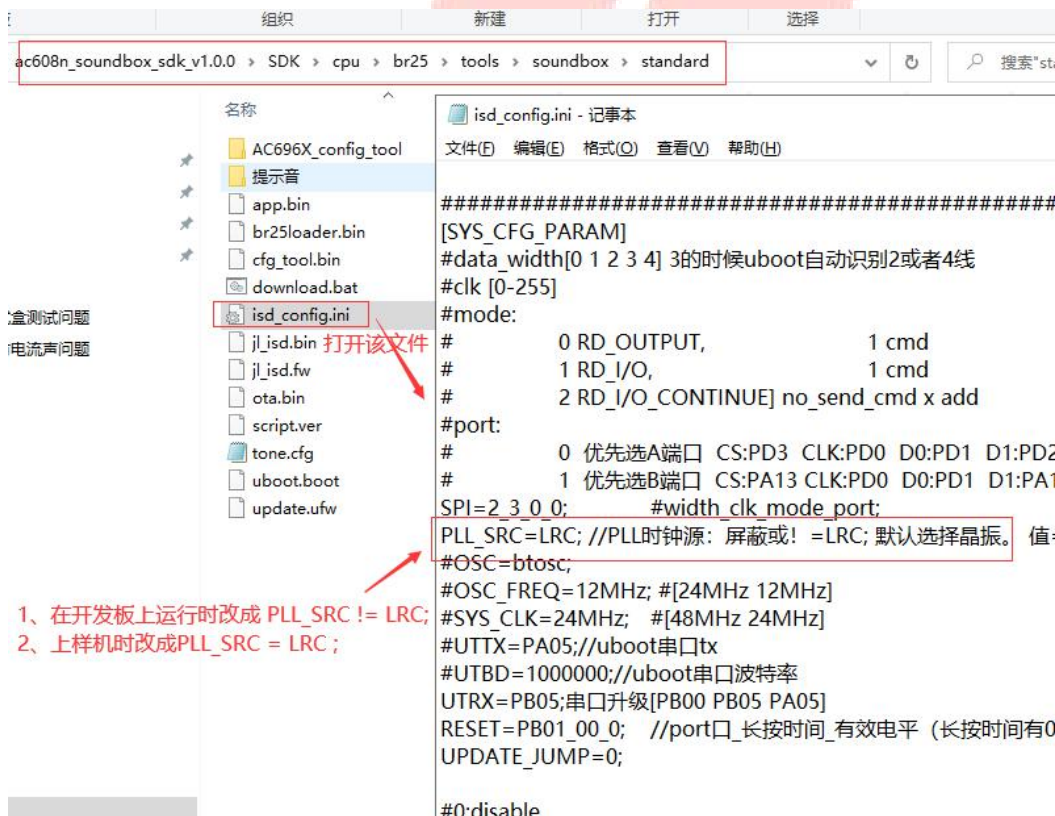
1.AC608X 在 AC696X 开发板上运行时若出现打印乱码、工作不正常需修改配置用外挂晶振 WTS 20201016

默认 SDK 是配置成使用 IC 内部 LRC 时钟的。如果要改成使用外部晶振，按以下修改方法改：1、



```
478 // #define TCFG_UI_ENABLE          ENABLE_THIS_MOUDLE //UI总开关
479 #define TCFG_UI_ENABLE          DISABLE_THIS_MOUDLE //UI总开关
480 #define CONFIG_UI_STYLE          STYLE_JL_LED7
481 #define TCFG_UI_LED7_ENABLE      ENABLE_THIS_MOUDLE //UI使用LED7显示
482 // #define TCFG_UI_LCD_SEG3X9_ENABLE  ENABLE_THIS_MOUDLE //UI使用LCD段码屏显示
483 // #define TCFG_LCD_ST7735S_ENABLE  ENABLE_THIS_MOUDLE
484 // #define TCFG_LCD_ST7789VW_ENABLE  ENABLE_THIS_MOUDLE
485 #define TCFG_SPI_LCD_ENABLE      DISABLE_THIS_MOUDLE //spi lcd开关
486 #define TCFG_TFT_LCD_DEV_SPI_HW_NUM 1// 1: SPI1 2: SPI2 配置lcd选择的spi口
487
488 //***** 时钟配置 *****//
489 //***** 在开发板运行时选用外部晶振 *****//
491 #define TCFG_CLOCK_SYS_SRC        SYS_CLOCK_INPUT_PLL_BT_OSC //系统时钟源选择
492 // #define TCFG_CLOCK_SYS_SRC        SYS_CLOCK_INPUT_PLL_RCL //系统时钟源选择
493 #define TCFG_CLOCK_SYS_HZ          24000000 //系统时钟设置
494 #define TCFG_CLOCK_OSC_HZ          24000000 //外界晶振频率设置
495 #define TCFG_CLOCK_MODE            CLOCK_MODE_ADAPTIVE
496
497 //***** 低功耗配置 *****//
498 //***** 低功耗配置 *****//
499 #define TCFG_LOWPOWER_POWER_SEL    PWR_LD015 //电源模式设置...可选
500 #define TCFG_LOWPOWER_BTOSC_DISABLE 0 //低功耗模式下BTOSC...
```

2、

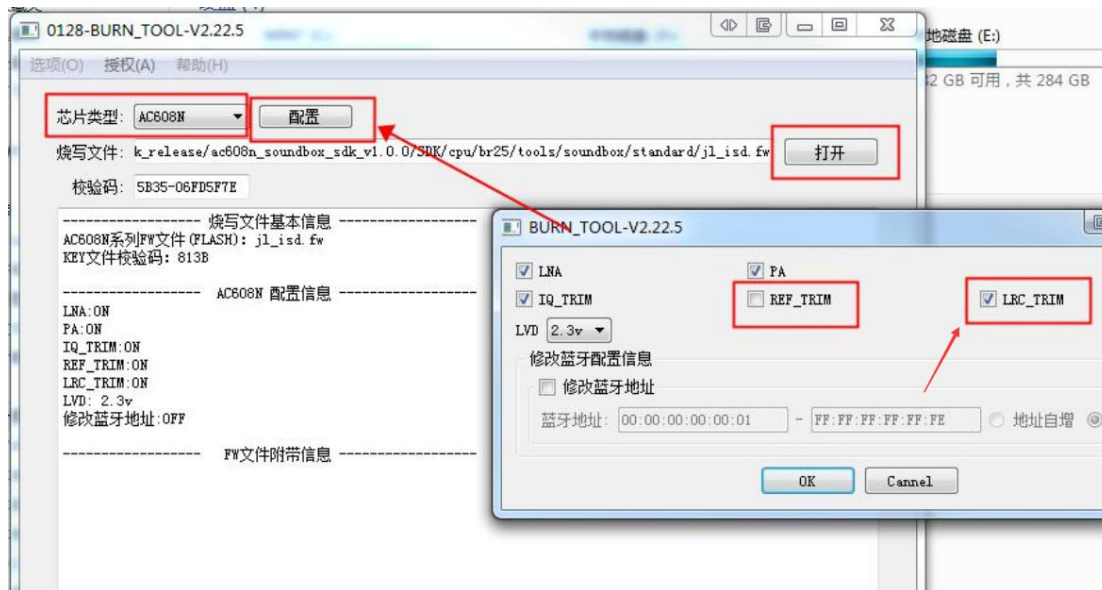


```
ac608n_soundbox_sdk_v1.0.0 > SDK > cpu > br25 > tools > soundbox > standard
名称
AC696X_config_tool
提示音
app.bin
br25loader.bin
cfg_tool.bin
download.bat
isd_config.ini
jl_isd.bin
jl_isd.fw
ota.bin
script.ver
tone.cfg
uboot.boot
update.ufw
isd_config.ini - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
#####
[SYS_CFG_PARAM]
#data_width[0 1 2 3 4] 3的时候uboot自动识别2或者4线
#clk [0-255]
#mode:
# 0 RD_OUTPUT, 1 cmd
# 1 RD_I/O, 1 cmd
# 2 RD_I/O_CONTINUE] no_send_cmd x add
#port:
# 0 优先选A端口 CS:PD3 CLK:PD0 D0:PD1 D1:PD2
# 1 优先选B端口 CS:PA13 CLK:PD0 D0:PD1 D1:PA1
SPI=2_3_0_0; #width_clk_mode_port;
PLL_SRC=LRC; //PLL时钟源: 屏蔽或! =LRC; 默认选择晶振。 值:
#OSC=btosc;
#OSC_FREQ=12MHz; #[24MHz 12MHz]
#SYS_CLK=24MHz; #[48MHz 24MHz]
#UTTX=PA05; //uboot串口tx
#UTBD=1000000; //uboot串口波特率
UTRX=PB05;串口升级[PB00 PB05 PA05]
RESET=PB01_00_0; //port口_长按时间_有效电平 (长按时间有0
UPDATE_JUMP=0;
#0:disable
```

- 1、在开发板上运行时改成 PLL_SRC != LRC;
- 2、上样机时改成PLL_SRC = LRC;

2.AC608 的芯片在进行调式前需要使用一拖二烧写器烧写，校准过 LRC 之后方能使用，升级工具需用 V4.0 版本 WTS 20201016

1、一拖二烧录时要选择 LRC_TRIM ， 不要选择 REF_TRIM:

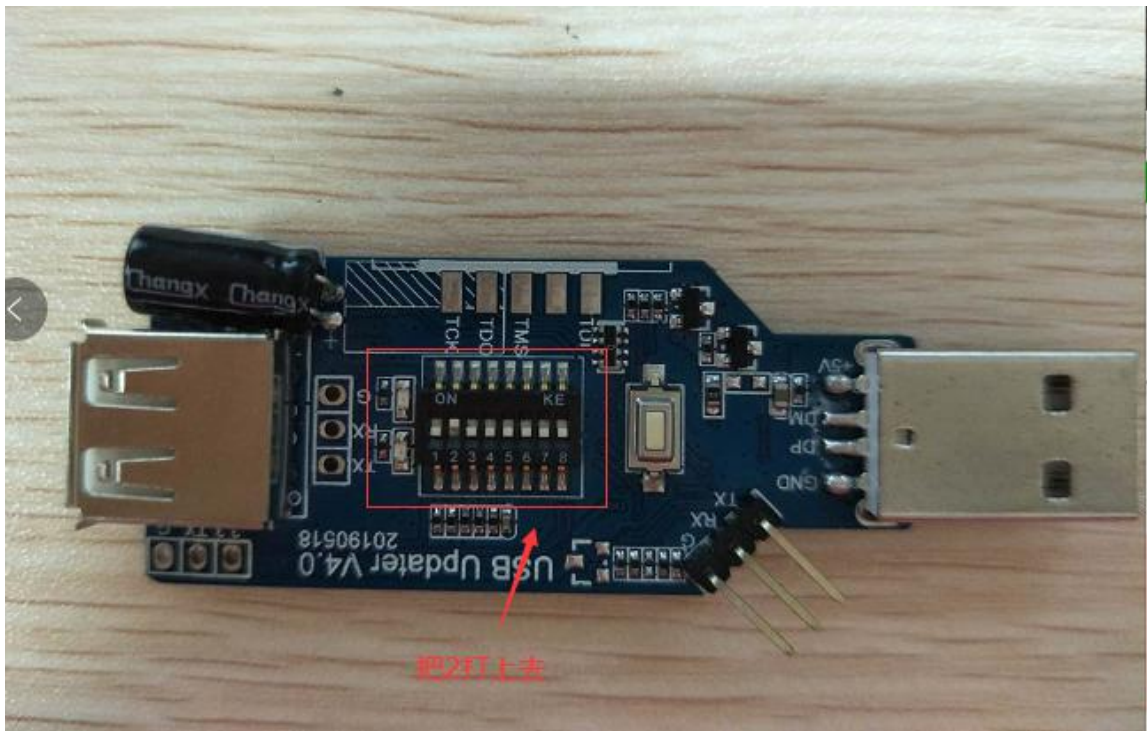


2、强制升级工具需使用 V4.0 版本工具，工具的固件需要升级到最新（可以在线使用 remote_firmware_update-0.0.1.exe 升级）。使用时拨码开关 2 要打上去：

A、在线升级 V4.0 工具固件：



B、使用拨码开关 2 要打上(不需要安装虚拟串口驱动和绑定虚拟串口)，操作方法跟之前的 V3.0 工具都是一样的：



3.AC608X SDK1.0.0 卡和 U 盘同时在线时，需要按两次设备切换按键才能切换设备问题修改 WTS 20201016

请按照网盘上资料“AC608N SDK1.0.0 无录音切换设备无效的问题.rar”处理。资料在以下百度网盘：

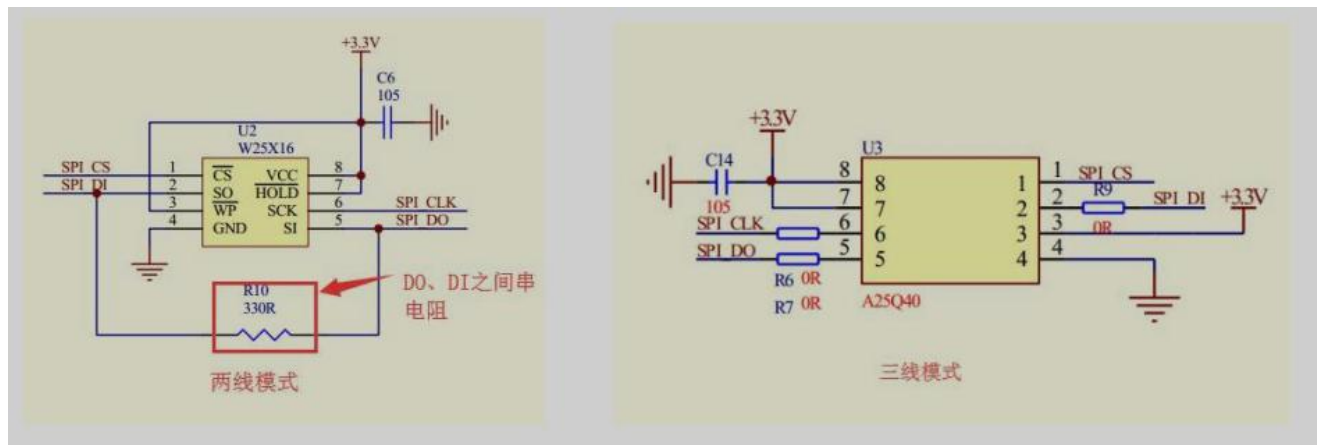
链接：<https://pan.baidu.com/s/1uJulUwywAwDgYHv0Z3vnrg>
提取码：jiei

4.AC608X SDK1.0.0 外挂 Flash 使用无法读取 id 问题注意点 LAQ 20201020

例如:

使用 SPI1 的 B 组 IO 时需要确认以下几点

- (1) 使用两线还是三线模式



对应下图修改， 例如使用两线模式

```
74 //*****
75 #define TCFG_HW_SPI1_ENABLE    ENABLE_THIS_MOUDLE
76 //A组IO:  DI: PB2    DO: PB1    CLK: PB0
77 //B组IO:  DI: PC3    DO: PC5    CLK: PC4
78 #define TCFG_HW_SPI1_PORT      'B'
79 #define TCFG_HW_SPI1_BAUD      4000000L
80 #define TCFG_HW_SPI1_MODE      SPI_MODE_UNIDIR_1BIT
81 #define TCFG_HW_SPI1_ROLE      SPI_ROLE_MASTER
82
83 #define TCFG_HW_SPI2_ENABLE      0//ENABLE_THIS_MOUDLE
84 //A组IO:  DI: PB8    DO: PB10   CLK: PB9
```

修改成两线模式
1bit data模式

- (2) 需要将 CS 引脚修改成 PC3

```
//*****
//                                FLASH 配置                                //
//*****
#define TCFG_NORFLASH_DEV_ENABLE    ENABLE_THIS_MOUDLE //需要关闭SD0
#define TCFG_FLASH_DEV_SPI_HW_NUM    1// 1: SPI1    2: SPI2
#define TCFG_FLASH_DEV_SPI_CS_PORT    IO_PORTC_03
```

备注: 如果出现复位请参考“AC696 软硬件问题整理”文档第 67 点

5.AC608X SDK1.0.0 设置为单声道后，有的机子有时播放 U 盘或卡音乐声音非常小解决方法 ---WTS 20201022

AC608X SDK1.0.0 设置为单声道后，有的机子有时播放 U 盘或卡里的音乐时，声音非常小。调音量也没有任何变化问题。解决方法是用以下网盘的 media.a 库替换后 rebuilt 解决：

链接：<https://pan.baidu.com/s/1gTfIRbUVtrWZfBs0fKr2xA>

提取码：jiei

6.AC608X SDK1.0.0 无法使用红外按键注意点 LAQ 20201028

(1) 红外相关修改点（按下图进行修改）：

```
194     sys_s_hi_timer_add(NULL, ir_timeout, 2); //2ms
195 }
196
197 void irflt_config()
198 {
199     JL_IR->RFLT_CON = 0;
200     JL_IR->RFLT_CON |= BIT(7); //256 div
201     JL_IR->RFLT_CON |= BIT(4);
202     JL_IR->RFLT_CON |= BIT(3); //osc
203     JL_IR->RFLT_CON |= BIT(2);
204     JL_IR->RFLT_CON |= BIT(0); //irflt enable
205
206     set_ir_clk();
207 }
208
```

加上这两句话

```
//红外定时器定义
#define IR_TIMER
#define IR_IRQ_TIME_IDX
#define IR_TIME_REG

TIMER5 // TIMER0
IRQ_TIME5_IDX //IRQ_TIME5_IDX
JL_TIMER5 //JL_TIMER5
```

将定时器修改成timer5


```
97  @note void set_ir_clk(void)
98
99  ((cnt - 1)* 分频数)/lsb_clk = 1ms
100 */
101 /*-----*/
102 #define APP_TIMER_CLK clk_get("timer")
103 #define TIMER_UNIT_MS 1
104 #define MAX_TIME_CNT 0x07ff //分频准确范围, 更具实际情况调整
105 #define MIN_TIME_CNT 0x0030
106 void set_ir_clk(void)
107 {
108     u32 clk;
109     u32 prd_cnt;
110     u8 index;
111
112     for (index = 0; index < (sizeof(timer_div) / sizeof(timer_div[0])); index++) {
113         prd_cnt = TIMER_UNIT_MS * (APP_TIMER_CLK / 1000) / timer_div[index];
114         if (prd_cnt > MIN_TIME_CNT && prd_cnt < MAX_TIME_CNT) {
115             break;
116         }
117     }
118
119     ir_code.timer_pad = prd_cnt;
120     IR_TIME_REG->CON = ((index << 4) | BIT(3) | BIT(1) | BIT(0));
121 }
122
123 /*-----*/
124 /**@brief 获取ir按键值
125 @param void
126 @param void
127 */
```

改成12000000L

修改成BIT (2)

添加一句 JL_IOMAP->CON0 |= BIT(21);

- (2) 注意红外按键是否有其它地方复用, 例如使用 PB7 时候就需要关闭 LINEIN 功能和 SPI2 的 A 组口, 如果关闭 LINEIN 功能, 需要注意的是将默认开机进入 LINEIN 改成 MUSIC 模式或者其他, 如下图

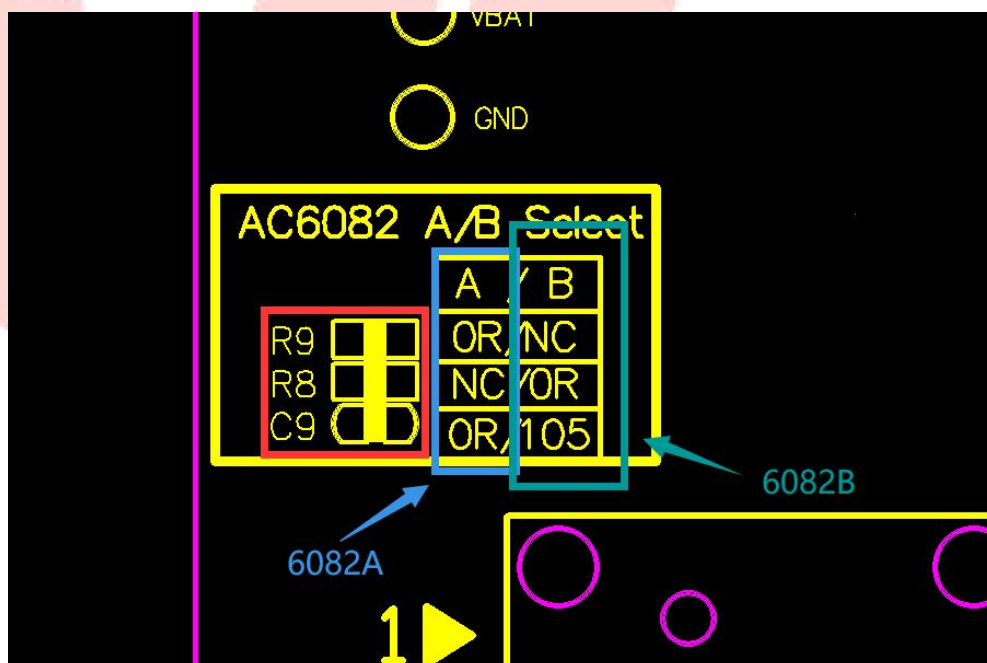
```
46
47 static void lcd_ui_power_on()
48 {
49     #if (TCFG_SPI_LCD_ENABLE)
50         int logo_time = 0;
51         UI_SHOW_WINDOW(ID_WINDOW_POWER_ON);
52         sys_timeout_add(NULL, lcd_ui_power_on_timeout, 1000);
53     #endif
54 }
55
56 static int power_on_init(void)
57 {
58     ///有些需要在开机提示完成之后再初始化的东西, 可以在这里初始化
59     #if (TCFG_SPI_LCD_ENABLE)
60         lcd_ui_power_on();//由ui决定切换的模式
61         return 0;
62     #endif
63
64     #if TCFG_APP_BT_EN
65         app_task_switch_to(APP_BT_TASK);
66     #else
67         /* app_task_switch_to(APP_MUSIC_TASK); */
68         app_task_switch_to(APP_LINEIN_TASK); //如果带检测, 设备不在线, 则不跳转
69     #endif
70
71     return 0;
72 }
73
```

开机切入音乐模式
换: APP_MUSIC_TASK

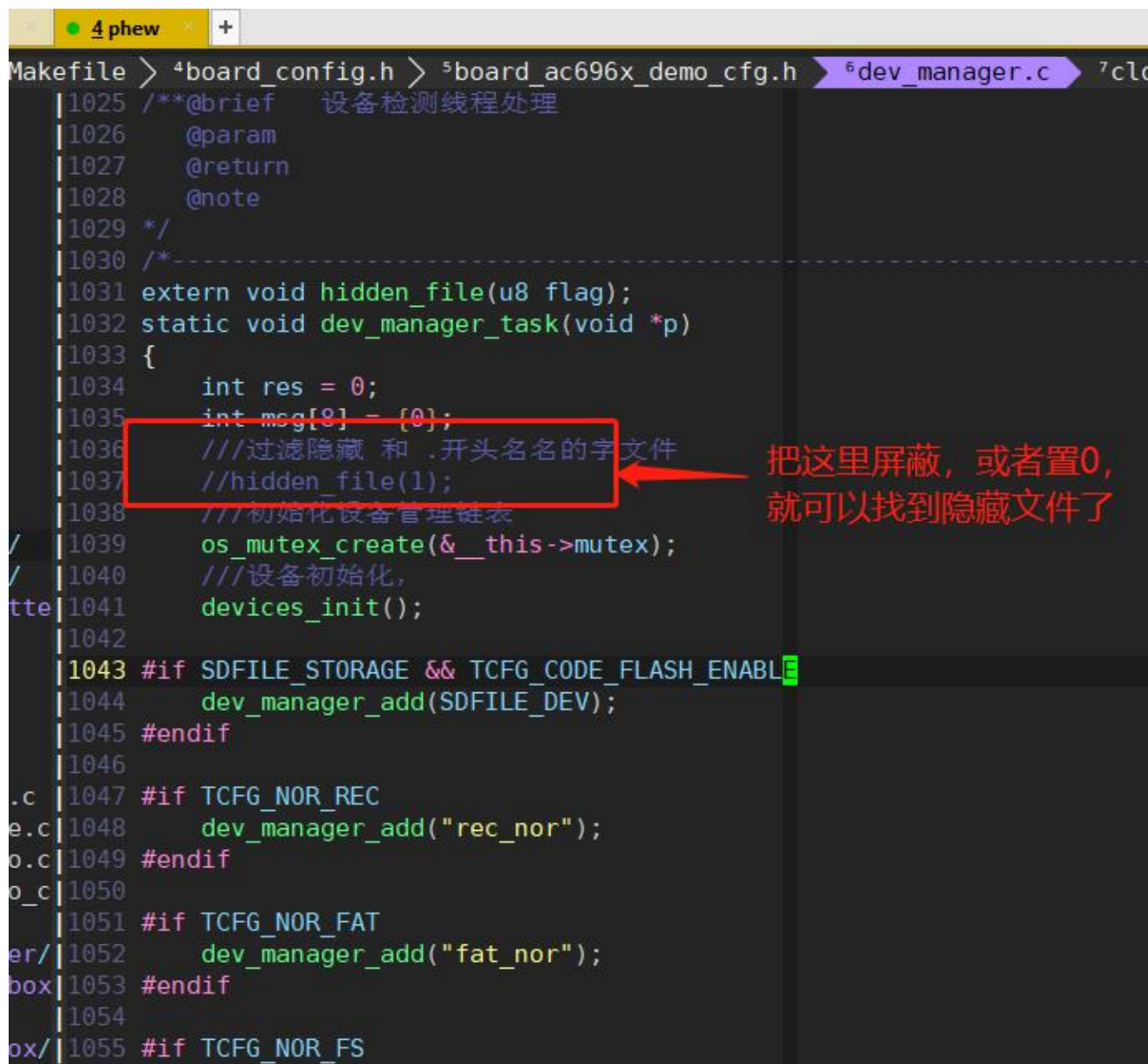
(3) 如果使用其它 IO 口的时候，需要留意是否是双邦，如果是的话需要将另一个 IO 口设为高阻，如果出现红外功能不正常的情况下，可以使用示波器抓取 IO 口获得的波形，如果出现 IO 口获取的波形正确，但是电压值不够的情况下，可以将 IO 口设置成上拉输入。如下图

```
}  
  
void |ir_input_io_sel(u8 port)  
{  
    //选择input channel1输入  
    // IOMC2[13 : 8]  
    // 0 ~ 15:      PA0 ~ PA15  
    //16 ~ 31:      PB0 ~ PB15  
    //32 ~ 47:      PC0 ~ PC15  
    //48 ~ 55:      PD0 ~ PD7  
    gpio_irflt_in(port);  
    gpio_set_direction(port, 1);  
    gpio_set_pull_up(port, 1);  
    gpio_set_die(port, 1);  
}
```

7.6082A、6082B 共用上层板烧录注意点，芯片封装差异，烧录上层版需要跳电阻选择芯片型号，如下图： 20201029 LZT



按以下截图修改:



```
Makefile > ^4board_config.h > ^5board_ac696x_demo_cfg.h > ^6dev_manager.c > ^7clo
1025 /**@brief 设备检测线程处理
1026 @param
1027 @return
1028 @note
1029 */
1030 /*-----
1031 extern void hidden_file(u8 flag);
1032 static void dev_manager_task(void *p)
1033 {
1034     int res = 0;
1035     int msg[8] = {0};
1036     ///过滤隐藏 和 .开头名名的字文件
1037     ///hidden_file(1);
1038     ///初始化设备管理链表
1039     os_mutex_create(&__this->mutex);
1040     ///设备初始化,
1041     devices_init();
1042
1043 #if SDFILE_STORAGE && TCFG_CODE_FLASH_ENABLE
1044     dev_manager_add(SDFILE_DEV);
1045 #endif
1046
1047 #if TCFG_NOR_REC
1048     dev_manager_add("rec_nor");
1049 #endif
1050
1051 #if TCFG_NOR_FAT
1052     dev_manager_add("fat_nor");
1053 #endif
1054
1055 #if TCFG_NOR_FS
```

9.AC608X SDK1.0.0 在播放 WAV 音频文件时，卡口无法做复用修改方法 20201106

----WTS

该问题是由于在播放 WAV 音频时，卡无法进入空闲引起。请使用以下网盘链接上的 cpu.a 库，替换后，rebuild 处理：

链接：<https://pan.baidu.com/s/1SnMy59Vtv2OK-VoGqvMDWA>

提取码：jjel

10.AC608X SDK1.0.0 录音暂停修改方法修改方法

----WTS

20201106

1、在 ac608n_soundbox_sdk_v1.0.0\SDK\cpu\br25\audio_enc\audio_enc_file.c 文件中添加：

```
audio_encoder_resume(&enc->encoder);
}

static void pcm2file_wfile_resume(struct pcm2file_enc_hdl *enc)
{
    os_sem_set(&enc->sem_wfile, 0);
    os_sem_post(&enc->sem_wfile);
    enc_write_file_resume(enc->whdl);
}

void pcm2file_enc_pp(void *hdl)    ///实现暂停
{
    struct pcm2file_enc_hdl *enc = (struct pcm2file_enc_hdl *)hdl;
    if(enc)
    {
        enc->enc_pause = !enc->enc_pause;
    }
}

u32 pcm2file_enc_get_pause_status(void *hdl)    /// 获取暂停状态
{
    struct pcm2file_enc_hdl *enc = (struct pcm2file_enc_hdl *)hdl;
    if(enc)
    {
        return enc->enc_pause;
    }
    return 0;
}

// 与pcm数据
int pcm2file_enc_write_pcm(void *priv, s16 *data, int len)
{
    struct pcm2file_enc_hdl *enc = (struct pcm2file_enc_hdl *)priv;
    if (!enc || !enc->status || enc->enc_err) {
        return 0;
    }
    if(enc->enc_pause) ///
    {
        return len;
    }

    enc->encoding = 1;
    u16 wlen = cbuf_write(&enc->pcm_cbuf, data, len);
    if (!wlen) {
```

新增这两函数

增加这部分代码

```
void pcm2file_enc_write_file_set_limit(void *hdl, u32 cut_size, u32 limit_size)
{
    struct pcm2file_enc_hdl *pcm2file = hdl;
    enc_write_file_set_limit(pcm2file->whdl, cut_size, limit_size);
}

void pcm2file_enc_set_evt_handler(void *hdl, void (*handler)(struct audio_encoder *, int, int *), u32 ma)
{
    struct pcm2file_enc_hdl *pcm2file = hdl;
    audio_encoder_set_event_handler(&pcm2file->encoder, handler, ma);
}

void pcm2file_enc_start(void *hdl)
{
    struct pcm2file_enc_hdl *pcm2file = hdl;
    pcm2file->status = 1;
    pcm2file->enc_pause = 0; ///  
    audio_encoder_start(&pcm2file->encoder);
    enc_write_file_start(pcm2file->whdl);

    //记录录音启动时的tick，后面定时更新头部信息用到
    pcm2file->head_update_tick = timer_get_ms();
}

int pcm2file_enc_get_time(void *hdl)
{
    struct pcm2file_enc_hdl *pcm2file = hdl;
    if (!pcm2file) {
        return 0;
    }

    int ret = audio_encoder_ioctl(&pcm2file->encoder, 1, AUDIO_ENCODER_IOCTL_CMD_GET_TIME);
    return ret;
}

void *pcm2file_enc_open(struct audio_fmt *pfmt, char *logo, char *folder, char *filename)
{
    int err;
    struct pcm2file_enc_hdl *pcm2file = NULL;
    //...
```

```
#define PCM_ENC2FILE_FILE_LEN_OVERLAY    (4 * 1024)
#define WAV_FILE_HEAD_LEN                90
#define PCM2FILE_ENC_BUF_COUNT            1
//wav录音头部信息定时更新时间设置，单位ms
#define UPDATA_WAV_HEAD_TIME              (5000)

extern struct audio_encoder_task *encode_task;

struct pcm2file_enc_hdl {
    struct audio_encoder encoder;
    s16 output_frame[1152 / 2]; //align 4Bytes

    int pcm_frame[64]; //align 4Bytes
    u8 file_head_frame[128];
    u8 file_head_len;
    /* u8 pcm_buf[PCM_ENC2FILE_PCM_LEN]; */
    u8 *pcm_buf;
    cbuffer_t pcm_cbuf;

    int out_file_frame[512 / 4];
    /* u8 out_file_buf[PCM_ENC2FILE_FILE_LEN]; */
    u8 *out_file_buf;
    cbuffer_t out_file_cbuf;

    void *whdl;
    OS_SEM sem_wfile;

    volatile u32 status : 1;
    volatile u32 enc_err : 1;
    volatile u32 encoding : 1;
    volatile u32 enc_pause : 1; ///  
    u32 lost;

    #if PCM2FILE_ENC_BUF_COUNT
        u16 pcm_buf_max;
        u16 out_file_max;
    #endif

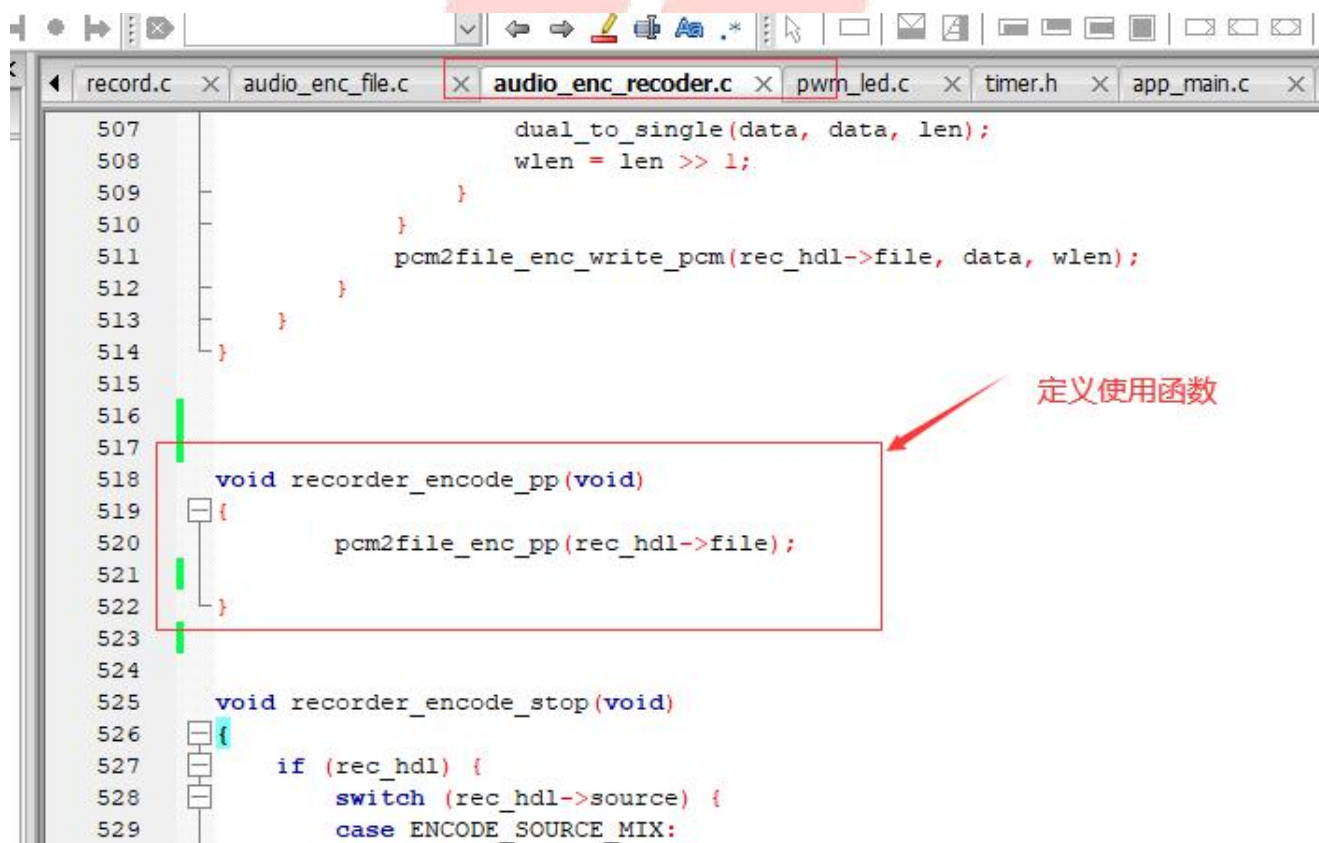
    u32 head_update_tick;
};
```


以上的 `void pcm2file_enc_pp(void *hdl)` `u32 pcm2file_enc_get_pause_status(void *hdl)` 函数的代码内容:

```
void pcm2file_enc_pp(void *hdl)    ///实现暂停
{
    struct pcm2file_enc_hdl *enc = (struct pcm2file_enc_hdl *)hdl;
    if(enc)
    {
        enc->enc_pause = !enc->enc_pause;
    }
}

u32 pcm2file_enc_get_pause_status(void *hdl) /// 获取暂停状态
{
    struct pcm2file_enc_hdl *enc = (struct pcm2file_enc_hdl *)hdl;
    if(enc)
    {
        return enc->enc_pause;
    }
    return 0;
}
```

2、调用案例:



```

1
2 #ifndef _AUDIO_ENC_RECORDER_H_
3 #define _AUDIO_ENC_RECORDER_H_
4
5 #include "media/audio_encoder.h"
6
7
8 int audio_adc_mic_init(u16 sr);
9 void audio_adc_mic_exit(void);
10
11 void linein_sample_set_resume_handler(void *priv, void (*resume)(void));
12 void fm_inside_output_handler(void *priv, s16 *data, int len);
13 int linein_sample_read(void *hdl, void *data, int len);
14 int linein_sample_size(void *hdl);
15 int linein_sample_total(void *hdl);
16 void *linein_sample_open(u8 source, u16 sample_rate);
17 void linein_sample_close(void *hdl);
18 void *fm_sample_open(u8 source, u16 sample_rate);
19 void fm_sample_close(void *hdl, u8 source);
20
21 //record player api 录音接口
22 void recorder_pcm2file_write_pcm_ex(s16 *data, int len);
23 void recorder_encode_stop(void);
24 u32 recorder_get_encoding_time();
25 //检查录音是否正在进行
26 int recorder_is_encoding(void);
27 void recorder_device_offline_check(char *logo);
28 void recorder_encode_pp(void);
29
30 #endif
31

```

```

179 int key_value = event->u.key.value;
180
181 log_i("key_event:%d \n", key_event);
182 switch (key_event) {
183     case KEY_ENC_START:
184         printf("KEY_ENC_START_REC_MODE\n");
185         log_i(" KEY_ENC_START \n");
186         record_key_pp();
187         return true;
188
189     #if (TCFG_MIC_EFFECT_ENABLE)
190     case KEY_REVERB_OPEN:
191         log_i("record mode ignore mic_effect!!\n");
192         ret = true;
193         break;
194     #endif
195
196     case KEY_MUSIC_PP:
197         printf("KEY_MUSIC_PP\n");
198         if(recorder_is_encoding())
199         {
200             printf("KEY MUSIC_PP_OK\n");
201             recorder_encode_pp();
202             return true;
203         }
204         break;
205
206     default:
207         break;
208 }
209
210 return ret;
211 }

```

11.AC6082A/B 原理图使用注意 20201110 YWP

AC6082A 音箱方案标准原理图 V2.0.pdf 和 AC6082B 音箱方案标准原理图 V2.0.pdf 是带有晶振的原理图，统一作废。AC6082A/B 不需要外部晶振，用内部 LRC 作为起振即可，不过烧录时请查看本文档第 2 点。请使用 **AC6082A 音箱方案标准原理图 V1.0.pdf** 和 **AC6082B 音箱方案标准原理图 V1.0.pdf** 为准。

12.AC608N 配置 PWM 注意点和参考资料 20201111 WTS

1、AC608N 配置跟 AC696X 音箱是一样的，仅需要修改时钟源为 RC 时钟即可（因为 AC608N 是省晶振，使用的是内部 LRC 时钟）。因此请参考网盘资料中 AC696X 音箱配 PWM 的资料。

时钟源修改方法：JL_TIMERx->CON |= (0b11 << 2); //选择 RC 时钟

```
74     gpio_set_pull_down(PWM_2_PORT,1);    ///映射io口位并上拉，手册说明
75     gpio_write(PWM_2_PORT,0);            ///映射io输出低，手册说明
76     gpio_set_dir(PWM_2_PORT,1);          ///映射io 数字模拟口设置，手册说明
77
78     #endif //PWM_2_USE_OUTPUT_CHANNEL
79
80 #endif // PWM_TIMER_2_REMAP_EN
81
82
83 _normal_init:
84     ///初始化timer
85     JL_TIMERx->CON = 0;
86     JL_TIMERx->CON |= (0b11 << 2);        ///选择晶振时钟，24M
87     JL_TIMERx->CON |= (0b0001 << 4);      ///时钟源4分频
88     JL_TIMERx->PRD = PWM_SYS_OSC_HZ / (freq * 4); ///设置周期
89     JL_TIMERx->CNT = 0;                    ///清零计数
90     JL_TIMERx->CON |= (0b01 << 0);        ///计数模式
91     JL_TIMERx->CON |= BIT(8);              ///使能PWM
92     JL_TIMERx->PWM = (JL_TIMERx->PRD * duty) / PWM_MAX_DUTY_VALUE; ///0~10000对应0~
93     // wdt_close();
94     // while(1);
95 }
96
```

改成这样

3-2	SSEL	RW	timer 驱动源选择 00: 使用系统时钟作为 timer 的驱动源; 01: 使用 IO 口信号作为 timer 的驱动源; 选择RC时钟 10: 使用 OSC 时钟作为 timer 的驱动源; ✓ 11: 使用 RC 时钟作为 timer 的驱动源。
-----	------	----	--

2、若在开发板上使用外部 24M 晶振调式 PWM 的，则不需要修改为 RC 时钟源。

3、对程序中寄存配置有疑问的，请注意看程序配注及 AC696X 用户手册。

参考资料在以下网盘路径：

链接：<https://pan.baidu.com/s/12Frh1mjPnudisx54RdE7mQ>
提取码：jiei

使用空白 U 盘出现不能录音的问题，可以按照下面图片进行修改。

修改前的程序：

```
case MUSIC_PLAYER_ERR_DEV_READ:
case MUSIC_PLAYER_ERR_DEV_OFFLINE:
    log_e("MUSIC_PLAYER_ERR_DEV_OFFLINE \n");
    logo = music_player_get_dev_cur();
    if (dev_manager_online_check_by_logo(logo, 1)) {
        ///如果错误失败在线， 并且是播放过程中产生的，先记录下断点
        if (music_player_get_playing_breakpoint(breakpoint, 1) == true) {
            breakpoint_vm_write(breakpoint, logo);
        }
        /* dev_manager_set_valid_by_logo(logo, 0);///将设备设置为无效设备 */
        //针对读错误， 因为时间推到应用层有延时导致下一个模式判断不正常， 此处需要将设备卸载
        dev_manager_unmount(logo);
    }
    if (dev_manager_get_total(1) == 0) {
        msg = KEY_MUSIC_PLAYER_QUIT;///没有设备在线， 退出音乐模式
    } else {
```

修改后的程序：

```
///此处不做任何处理， 打断的事件已经重发， 由重发事件执行后续处理
break;
}
case MUSIC_PLAYER_ERR_DEV_READ:
case MUSIC_PLAYER_ERR_DEV_OFFLINE:
    log_e("MUSIC_PLAYER_ERR_DEV_OFFLINE \n");
    logo = music_player_get_dev_cur();
    if (dev_manager_online_check_by_logo(logo, 1)) {
        ///如果错误失败在线， 并且是播放过程中产生的，先记录下断点
        if (music_player_get_playing_breakpoint(breakpoint, 1) == true) {
            breakpoint_vm_write(breakpoint, logo);
        }
        /* dev_manager_set_valid_by_logo(logo, 0);///将设备设置为无效设备 */
        if(err == MUSIC_PLAYER_ERR_FSCAN){
            dev_manager_set_valid_by_logo(logo , 0);
        }else{
            //针对读错误， 因为时间推到应用层有延时导致下一个模式判断不正常， 此处需要将设备卸载
            dev_manager_unmount(logo);
        }
    }
    if (dev_manager_get_total(1) == 0) {
        msg = KEY_MUSIC_PLAYER_QUIT;///没有设备在线， 退出音乐模式
    } else {
```

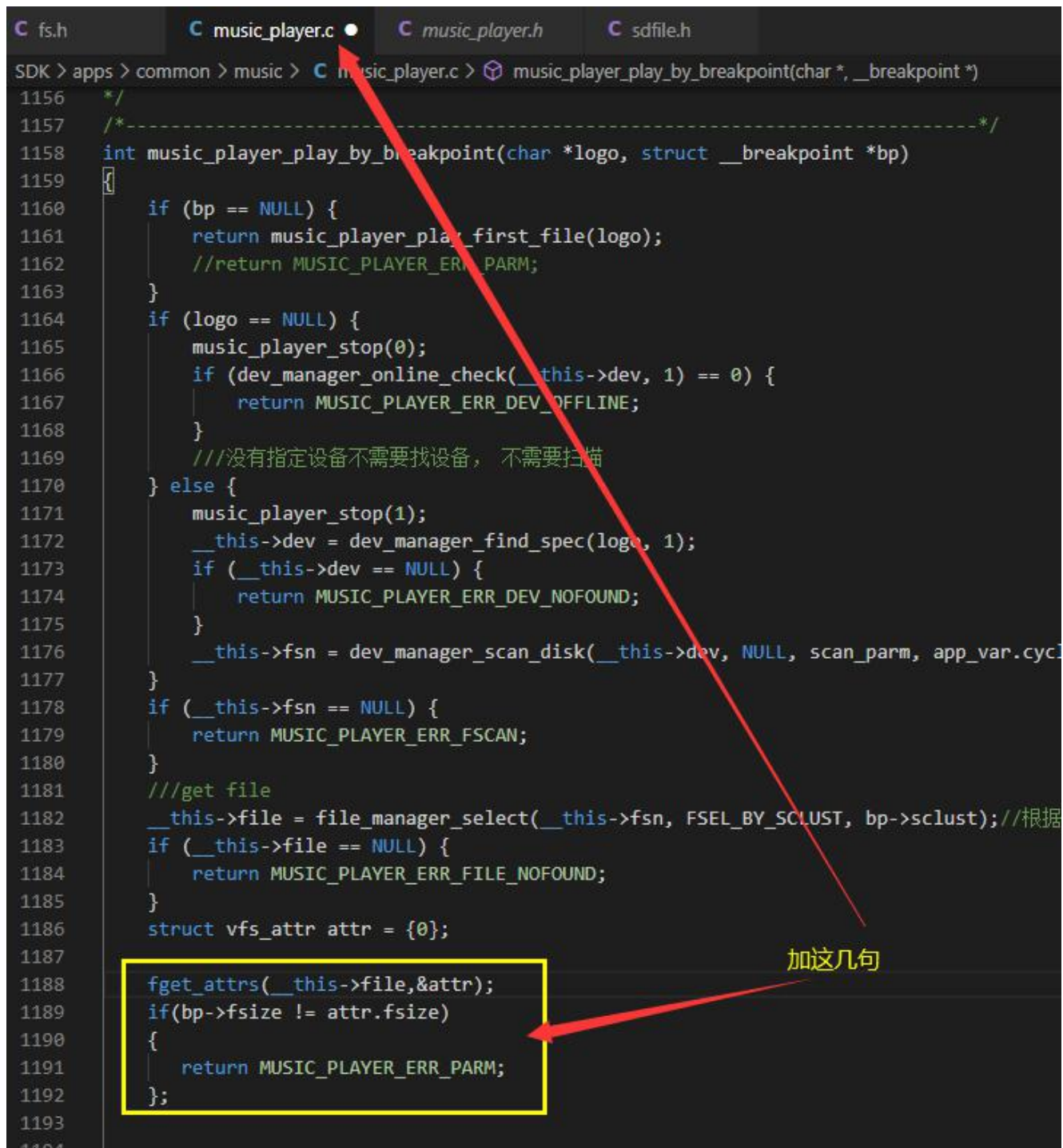
AC608N SDK1.0.0 上下曲切歌有啞声问题需要更换的文件:

链接: <https://pan.baidu.com/s/1rFehVpq4GU656rRF0YCPmw>

提取码: jiel



15.AC608N SDK1.0.0 两张含有加密文件的 SD 卡（U 盘）轮流插入，由于簇号相同导致第二张插入的 SD 卡（U 盘）根据上一个设备的断点播放的处理方法 20201116 LJW



```
SDK > apps > common > music > C music_player.c > music_player_play_by_breakpoint(char *, __breakpoint *)
1156  */
1157  /*-----*/
1158  int music_player_play_by_breakpoint(char *logo, struct __breakpoint *bp)
1159  {
1160      if (bp == NULL) {
1161          return music_player_play_first_file(logo);
1162          //return MUSIC_PLAYER_ERR_PARM;
1163      }
1164      if (logo == NULL) {
1165          music_player_stop(0);
1166          if (dev_manager_online_check(__this->dev, 1) == 0) {
1167              return MUSIC_PLAYER_ERR_DEV_OFFLINE;
1168          }
1169          ///没有指定设备不需要找设备， 不需要扫描
1170      } else {
1171          music_player_stop(1);
1172          __this->dev = dev_manager_find_spec(logo, 1);
1173          if (__this->dev == NULL) {
1174              return MUSIC_PLAYER_ERR_DEV_NOFOUND;
1175          }
1176          __this->fsn = dev_manager_scan_disk(__this->dev, NULL, scan_parm, app_var.cyc);
1177      }
1178      if (__this->fsn == NULL) {
1179          return MUSIC_PLAYER_ERR_FSCAN;
1180      }
1181      ///get file
1182      __this->file = file_manager_select(__this->fsn, FSEL_BY_SCLUST, bp->sclust);///根据
1183      if (__this->file == NULL) {
1184          return MUSIC_PLAYER_ERR_FILE_NOFOUND;
1185      }
1186      struct vfs_attr attr = {0};
1187
1188      fget_attr(__this->file, &attr);
1189      if (bp->fsize != attr.fsize)
1190      {
1191          return MUSIC_PLAYER_ERR_PARM;
1192      };
1193
1194
```

16.AC608N SDK1.0.0 录音模式播放录音文件时插入 U 盘播歌，再回录音模式播放录音文件时无法播放问题 20201119 LAQ

(1) 在 nor_fs.c 中定义图中变量

```
static int is_read_only = 0;
```

(2) 在 nor_fs.c 中添加下面代码

```
/*-----*/
/**@brief 关闭一个文件
 * @param pfile:文件的句柄
 * @return u32:建立文件索引
 * @author liujie
 * @note u32 close_recfile(REC_FILE *pfile)
 */
/*-----*/
static int is_read_only = 0;

u32 close_recfile(REC_FILE *pfile)
{
    RECFILEHEAD file_h;

    /******添加下面这份代码******/
    if(is_read_only) {
        pfile->pfs->index.index = pfile->index.index;
        pfile->pfs->index.sector = pfile->index.sector;
        is_read_only = 0;
        return file_h.len;
    }

    memset(&file_h, 0xff, sizeof(RECFILEHEAD));
    //file_h->crc;
    pfile->w_len = pfile->rw_p;

    pfile->len = pfile->w_len;
    pfile->rw_p = 0; //sizeof(RECFILEHEAD);
    file_h.len = pfile->w_len;
    file_h.index = pfile->index.index;
    file_h.addr = pfile->addr;
    memcpy(file_h.name, pfile->priv_data, NORFS_DATA_LEN);
    r_printf(">>>[test]:file_h.priv_data = %s, len = %d\n", file_h.name, NORFS_DATA_LEN);

    file_h.head_crc = CRC16(&file_h.data_crc, sizeof(RECFILEHEAD) - 2);
    r_printf(">>>[test]:pfile->index.sector = %d\n", pfile->index.sector);
    /* recf_seek(pfile, NOR_FS_SEEK_SET, 0); */
    /* recf_seek(pfile, NOR_FS_SEEK_SET, 0); */
}
```

添加这部分代码

(3) 在 nor_fs.c 的 nor_fs_open 函数中添加下图代码

```
index = create_recfile(&recfs, &recfile);
get_filename_by_path((char *)filename, (char *) path);
nor_rec_set_priv_info((u8 *)filename);
r_printf(">>>[test]:path = %s, ext = %s, filename = %s\n", path, ext, filename);

nor_fs_index = index;
y_printf(">>>[test]: create_recfile nor_fs_hdl->index= %d\n", nor_fs_index);
return 0;
} else if (mode[0] == 'r') {
y_printf(">>>[test]: open_recfile nor_fs_hdl->index= %d\n", nor_fs_index);
if (nor_fs_index) {
index = nor_fs_index;
} else {
log_e("error!!!!!!!!!!");
index = 0;
return 1;
}
reg = open_recfile(index, &recfs, &recfile);
is_read_only = 1;
return 0;
}
return 0;
}

int music_flash_file_set_index(u8 file_sel, u32 index)
```

添加这部分代码

17.AC608N SDK1.0.0 无缝循环多次播放音乐时，第一遍和第二遍之间出现衔接不好的问题解决方案（包括一些注意点） 20201119 LAQ

（1）无缝循环播放注意点：

- a)选取首尾连接较好的音源
- b)将音源的前后的静音区全部去掉

（2）如果想做无限循环播放，可以按下图修改

```
#if 1
/*-----*/
/**@brief    循环播放回调接口
 * @param    *priv: 私有参数
 * @return    0: 循环播放
 * @return    非0: 结束循环
 * @note
 */
/*-----*/
static int file_dec_repeat_cb(void *priv)
{
    struct file_dec_hdl *dec = priv;
    y_printf("file_dec_repeat_cb\n");
    g_printf("sys_time is %d" , clk_get("sys"));
    if (dec->repeat_num) {
        dec->repeat_num--;
    } else {
        y_printf("file_dec_repeat_cb end\n");
        return -1;
    }
    return 0;
}

/*-----*/
```

无限循环时候可以注释掉

（3）出现第一遍和第二遍之间衔接不够好，后面播放没有问题的情况，可以更新下面的库，更新完库后要记得 Rebuild 一下

链接: https://pan.baidu.com/s/1IetJSWo0TA_dHFMVcceCfw

提取码: JLKJ

18.AC608N SDK1.0.0 外挂 flash 录音录到 fat，并在 music 模式下播放录音文件夹的方法 20201124 FSW

默认 SDK 的外挂 flash 录音是录到一个独立的分区，只能通过特定接口单曲播放。如果需要录成文件查看或者在 music 模式播放录音文件夹，可按下面的方法修改。

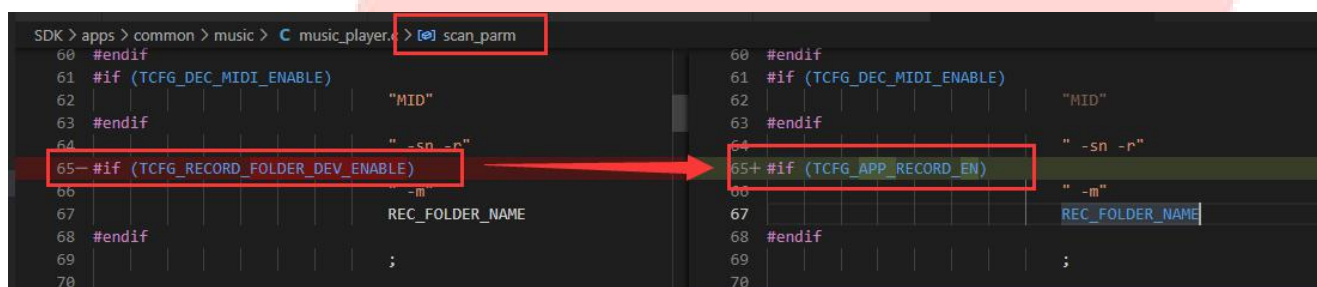
(1) 关闭宏 TCFG_NOR_REC

```
#define TCFG_NOR_REC DISABLE
```

(2) 关闭宏 TCFG_RECORD_FOLDER_DEV_ENABLE

```
#define TCFG_RECORD_FOLDER_DEV_ENABLE DISABLE//音乐播放录音区分使能
```

(3) 在 music_player.c 修改 music 模式下扫描文件用的参数 scan_parm，”-m”参数用于过滤文件夹（这里扫描歌曲时过滤录音文件夹）



(4) 在 music_player.c 创建一个专门用于录音扫描的 rec_scan_parm，不过滤录音文件夹。

```
//播放参数，扫描录音文件时用，文件后缀等
static const char rec_scan_parm[] = "-t"
#if (TCFG_DEC_MP3_ENABLE)
    "MP1MP2MP3"
#endif
#if (TCFG_DEC_WMA_ENABLE)
    "WMA"
#endif
#if (TCFG_DEC_WAV_ENABLE || TCFG_DEC_DTS_ENABLE)
    "WAV"
#endif
#if (TCFG_DEC_FLAC_ENABLE)
    "FLA"
#endif
#if (TCFG_DEC_APE_ENABLE)
    "APE"
#endif
#if (TCFG_DEC_M4A_ENABLE)
    "M4AAC"
#endif
#if (TCFG_DEC_M4A_ENABLE || TCFG_DEC_ALAC_ENABLE)
    "MP4"
#endif
#if (TCFG_DEC_AMR_ENABLE)
    "AMR"
#endif
#if (TCFG_DEC_DECRYPT_ENABLE)
    "SMP"
#endif
#if (TCFG_DEC_MIDI_ENABLE)
    "MID"
#endif
    "-sn -r";
```

(5) 在 music_player.c 新建以下函数。

```
static u8 rec_play_state = 0;
void music_player_set_rec_play_state(u8 state)
{
    rec_play_state = state;
}

u8 music_player_get_rec_play_state(void)
{
    return rec_play_state;
}

//播放 fat 中的录音文件夹
int music_player_play_fat_record_folder(char *logo, const char *path)
{
    if (music_player_get_rec_play_state()){
        return MUSIC_PLAYER_SUCC;
    }
    if (path == NULL) {
        return MUSIC_PLAYER_ERR_POINT;
    }
    if (logo == NULL) {
        music_player_stop(0);
        if (dev_manager_online_check(__this->dev, 1) == 0) {
            return MUSIC_PLAYER_ERR_DEV_OFFLINE;
        }
        ///没有指定设备不需要找设备， 不需要扫描
    } else {
        music_player_stop(1);
        __this->dev = dev_manager_find_spec(logo, 1);
        if (__this->dev == NULL) {
            return MUSIC_PLAYER_ERR_DEV_NOFOUND;
        }
        __this->fsn = dev_manager_scan_disk(__this->dev, NULL, rec_scan_parm, FCYCLE_FOLDER/*cycle_m
ode*/, __this->parm.cb->fsn_break);
    }
    if (__this->fsn == NULL) {
        return MUSIC_PLAYER_ERR_FSCAN;
    }
    ///get file
    __this->file = file_manager_select(__this->fsn, FSEL_BY_PATH, (int)path);///根据簇号查找文件
    if (__this->file == NULL) {
        return MUSIC_PLAYER_ERR_FILE_NOFOUND;
    }
    ///start decoder
    music_player_set_rec_play_state(1);
    int err = music_player_decode_start(__this->file, 0);
    if (err == MUSIC_PLAYER_SUCC) {
        ///选定新设备播放成功后，需要激活当前设备
        dev_manager_set_active(__this->dev);
        log_i("[%s %d] ok\n", __FUNCTION__, __LINE__);
    }
    else{
        music_player_set_rec_play_state(0);
    }
    return err;
}
```

```

C music_player.h (HEAD) X  C board_ac6082_demo.c  C pwm_led_api.c  C ui_manage.h
134 //music_player录音区分切换播放
135 int music_player_play_record_folder(char *logo, struct __breakpoint *bp);
136+ //获取当前设备的物理设备
137 const char *music_player_get_phy_dev(int *len);
138+ //播放fat中的录音文件夹
139+ int music_player_play_fat_record_folder(char *logo, const char *path);
140+ //获取录音播放标志
141+ u8 music_player_get_rec_play_state(void);
142
143 #endif// __MUSIC_PLAYER_H__
144
145

```

(6) 在 music.c 中修改 case KEY_MUSIC_PLAYE_REC_FOLDER_SWITCH:中的代码

```

430     case KEY_MUSIC_PLAYE_REC_FOLDER_SWITCH:
431         log_i("KEY_MUSIC_PLAYE_REC_FOLDER_SWITCH\n");
432     #if (TCFG_RECORD_FOLDER_DEV_ENABLE)
433         ///尝试保存断点
434         if (music_player_get_playing_breakpoint(breakpoint, 1) == true) {
435             breakpoint_vm_write(breakpoint, music_player_get_dev_cur());
436         }
437         if (true == breakpoint_vm_read(breakpoint, music_player_get_cur_music_dev())) {
438             err = music_player_play_record_folder(NULL, breakpoint);
439         } else {
440             err = music_player_play_record_folder(NULL, NULL);
441         }
442+ #else
443+     logo = dev_manager_get_logo(dev_manager_find_active(1));
444+     if (music_player_get_rec_play_state()){
445+         if (true == breakpoint_vm_read(breakpoint, logo)) {
446+             err = music_player_play_by_breakpoint(logo, breakpoint);
447+         } else {
448+             err = music_player_play_first_file(logo);
449+         }
450+     }
451+     else{
452+         if (music_player_get_playing_breakpoint(breakpoint, 1) == true) {
453+             breakpoint_vm_write(breakpoint, music_player_get_dev_cur());
454+         }
455+         err = music_player_play_fat_record_folder(logo, "/JL_REC/");///this is a demo
456+     }
457 #endif//TCFG_RECORD_FOLDER_DEV_ENABLE
458     break;
459     case KEY_MUSIC_PLAYE_BY_DEV_FILENUM:

```

```

logo = dev_manager_get_logo(dev_manager_find_active(1));
if (music_player_get_rec_play_state()){
    if (true == breakpoint_vm_read(breakpoint, logo)) {
        err = music_player_play_by_breakpoint(logo, breakpoint);
    } else {
        err = music_player_play_first_file(logo);
    }
}
else{
    if (music_player_get_playing_breakpoint(breakpoint, 1) == true) {
        breakpoint_vm_write(breakpoint, music_player_get_dev_cur());

```



```
    }  
    err = music_player_play_fat_record_folder(logos, "/JL_REC/");///this is a demo  
}
```

(7) 在 breakpoint.c 中修改存取断点的代码。

声明

```
extern void music_player_set_rec_play_state(u8 state);  
extern u8 music_player_get_rec_play_state(void);
```

需要读取断点时，认为已经退出了录音文件夹播放模式（人为认定，也可以用其他条件认定，看需求）

```
37 bool breakpoint_vm_read(struct __breakpoint *bp, char *logo)  
38 {  
39     if (bp == NULL || logo == NULL) {  
40         return false;  
41     }  
42+ music_player_set_rec_play_state(0);  
43     u16 vm_id;  
44     if (!strcmp(logo, "sd0")) {  
45         vm_id = CFG_SD0_BREAKPOINT0;  
46     } else if (!strcmp(logo, "sd1")) {  
47         vm_id = CFG_SD1_BREAKPOINT0;  
48     } else if (!strcmp(logo, "udisk0")) {
```

当前处于录音文件夹播放模式，不允许存断点

```
62  
63 void breakpoint_vm_write(struct __breakpoint *bp, char *logo)  
64 {  
65+ if (bp == NULL || logo == NULL || music_player_get_rec_play_state()) {  
66     return ;  
67 }  
68     u16 vm_id;  
69     if (!strcmp(logo, "sd0")) {  
70         vm_id = CFG_SD0_BREAKPOINT0;  
71     } else if (!strcmp(logo, "sd1")) {  
72         vm_id = CFG_SD1_BREAKPOINT0;  
73     } else if (!strcmp(logo, "udisk0")) {  
74         vm_id = CFG_USB_BREAKPOINT0;
```

(8) 可选修改

如果录音时会变音卡顿或者直接复位，需要修改录音码率（改小）

```
SDK > cpu > br25 > audio_enc > C audio_enc_recorder.c > recorder_encode_start(record_file_fmt *)  
416 #if 1  
417  
418 /* #define RECORD_PLAYER_DEFAULT_SAMPLERATE (44100L) */  
419 #define RECORD_PLAYER_DEFAULT_SAMPLERATE (16000L)  
420 #define RECORD_PLAYER_DEFAULT_BITRATE (64L)  
421 #define RECORD_PLAYER_DEFAULT_ADPCM_BLOCKSIZE (1024L) //256/512/1024/2048  
422 #define RECORD_PLAYER_MIXER_CHANNELS 1 //mixer record single channel  
423  
424  
425 struct record_hdl {
```

如果只想录一个文件，直接固定录音文件的名字（默认是 char filename[] = {"AC69****"};文件名会自加，也就是多次录音会出现多个文件）。

```
SDK > apps > soundbox > task_manager > record > C record.c > record_mic_start(void)
60  /*-----*/
61  static void record_mic_start(void)
62  {
63      struct record_file_fmt fmt = {0};
64      /* char logo[] = {"sd0"}; */ //可以指定设备
65      char folder[] = {REC_FOLDER_NAME}; //录音文件夹名称
66      char filename[] = {"AC690000"}; //录音文件名，不需要加后缀，录音接口会根据编码格式添加后缀
67
68      #if (TCFG_NOR_REC)
```

19.AC608N SDK1.0.0 没有提示音,开机直接进 FM/LINEIN 模式声音小处理方法 20201126 LZK

【具体现象】：开机出现声音小/几乎没有声音，VCOM 没有电压。插 U 盘/TF 卡后，DAC 被打开，声音恢复正常。

【修改方法】：开机手动打开 DAC 函数，参考如下

```
void app_poweron_task()
{
    int msg[32];

    HIT_SHOW_MENU(MENU_POWER_UP, 0, 0, NULL);
    app_audio_output_start(); //手动调用该函数
    int err = tone_play_with_callback_by_name(tone_table[INDEX_TONE_POWER_ON], 1, tone_play_end_callback);
    /* if (err) { //提示音没有,播放失败,直接init流程 */
    /* power_on_init(); */
    /* } */

    while (1) {
        app_task_get_msg(msg, ARRAY_SIZE(msg), 1);
        switch (msg[0]) {
            case APP_MSG_SYS_EVENT:
                if (poweron_sys_event_handler((struct sys_event *) (msg + 1)) == false) {
                    app_default_event_deal((struct sys_event *) (&msg[1])); //由common统一处理
                }
            break;
        }
    }
}
```

20.AC608N /AC696 LINEIN 没声音或系统音量超过 24 没声音补充 20201130 LHY

优先先看 20 点处理问题，未处理再看这里

- 1、第一个先检测 LINEIN 有没有隔电容，由于音频源设备有可能偏置电压和样机不同有压差，导致在调大增益时候会把 DAC 偏置电压拉高导致没声音。
- 2、第二可能是把 LINEIN 输入引脚配置为唤醒口，唤醒口会将引脚设置为上下拉也会导致 DAC 偏置电压拉高导致没声音。

```
722  /****** PWR config *****/
723  struct port_wakeup port0 = {
724      .pullup_down_enable = 0, //ENABLE, //配置I/O 内部上下拉是否使能
725      .edge = FALLING_EDGE, //唤醒方式选择,可选: 上升沿\下降沿
726      .attribute = BLUETOOTH_RESUME, //保留参数
727      .iomap = IO_PORTB_06, //唤醒口选择
728  };
```