

一、杰理实习第二周培训文档记录

前言：勿骄勿躁，打好基础，以客户需求为出发点

首周会议收获：通过前辈们的讲解，认识到了学习要脚踏实地的重要性。并且认识到对于企业而言，写可阅读性强的技术文档的重要性。

第二周任务：

- 完成 AC695N 的 SDK 文档熟悉，包括编译验证。
-

1. 文档资料指南

工程应用技术应用文档

The screenshot shows a document entry summary page. On the left, there is a sidebar with a red border containing a '目录' (Table of Contents) section. The main content area has a title '工程音频技术应用文档入口汇总' and a sub-section '一、资料共享'. Below this, there is a table with several rows of links:

环境安装	杰理WINDOWS环境安装及其他事项	LINUX安装和编译
最新SDK&补丁情况	音频类补丁汇总表	对外特别发布开发注意点信息汇总
认证资料获取汇总	认证资料获取汇总(最新资料后台会更新)	《Hi-Res申请指南v1.0》
杰理常用工具及软件调试方法文档	杰理工具文档 (开发环境、量产相关工具等)	软件调试方法
	Window下使用Makefile编译代码教程	欢迎使用杰理OTA外接库开发文档(Android) — 杰理OTA外接库开发文档(Android) 1.6.0 documentation
常用工具使用	常规设备使用指导	

截图参考

非常好的调试教程

环境安装	■ 杰理WINDOWS环境安装及其他事项 ■ LINUX安装和编译
最新SDK&补丁情况	S 音频类补丁汇总表 S 对外特别发布开发注意点信息汇总
认证资料获取汇总	■ 认证资料获取汇总(最新资料后台会更新) P 《Hi-Res申请指南v1.0》
杰理常用工具及软件调试方法文档	杰理工具文档 (开发环境、量产相关工具等) ■ 软件调试方法 W Window下使用Makefile编译代码教程 R 欢迎使用杰理OTA外接库开发文档(Android) — 杰理OTA外接库开发文档(Android) 1.6.0 documentation
常用工具使用	■ 常规设备使用指导

1.1. 芯片 doc 文件

1.1.1. SDK 介绍文档：

一般是整个芯片系统的介绍，包括 APP 消息、APP 模式管理、蓝牙模块的使用等等；

功能模块说明文档 >		修改日期	类型	大小
功能详细设计文档		2022/3/1 11:49	文件夹	
外挂flash 相关操作文档		2022/3/1 11:49	文件夹	
无晶振相关说明文档		2022/3/1 11:49	文件夹	
PDF AC695N_app升级说明文档.pdf	2022/2/25 11:48	Microsoft Edge ...	877 KB	
PDF AC695N_soundbox_SDK_介绍V2.0 .pdf	2022/3/1 20:08	Microsoft Edge ...	9,635 KB	
PDF AC695N混响音效在线调试说明文档.pdf	2022/2/25 11:48	Microsoft Edge ...	618 KB	
PDF AC695N新版eq工具在线调试说明.pdf	2022/2/25 11:48	Microsoft Edge ...	1,627 KB	
PDF soundbox天猫精灵使用说明.pdf	2022/2/25 11:48	Microsoft Edge ...	357 KB	
PDF 功耗调试说明书.pdf	2022/2/25 11:48	Microsoft Edge ...	638 KB	

1.1.2 功能详细设计文档：

具体解释每一个细分功能的应用原理，例如运用蓝牙模式、广播模式、TWS 模式等等，适合在使用一个具体模块时进行查阅参考。

名称	修改日期	类型	大小
儿通话调试手册	2022/3/1 11:49	文件夹	
UI布局以及升级UI说明文档	2022/3/1 11:49	文件夹	
UI简易架构 (旧版ui) 工具和说明	2024/8/14 10:38	文件夹	
功能详细设计文档	2022/3/1 11:49	文件夹	
外挂flash 相关操作文档	2022/3/1 11:49	文件夹	
无晶振相关说明文档	2022/3/1 11:49	文件夹	
AC695N_app升级说明文档.pdf	2022/2/25 11:48	Microsoft Edge ...	877 KB
AC695N_soundbox_SDK_介绍V2.0.pdf	2022/3/1 20:08	Microsoft Edge ...	9,635 KB
AC695N混响音效在线调试说明文档.pdf	2022/2/25 11:48	Microsoft Edge ...	618 KB

名称	修改日期	类型	大小
recorder mix接口设计说明文档.pdf	2022/2/25 11:48	Microsoft Edge ...	491 KB
record应用详细设计说明书.pdf	2022/2/25 11:48	Microsoft Edge ...	402 KB
uart_iic_spi接口说明(1).pdf	2022/2/25 11:48	Microsoft Edge ...	266 KB
ui接口设计说明文档.pdf	2022/2/25 11:48	Microsoft Edge ...	501 KB
充电接口设计说明文档.pdf	2022/2/25 11:48	Microsoft Edge ...	254 KB
发射器接口设计说明文档.pdf	2022/2/25 11:48	Microsoft Edge ...	410 KB
混响应用详细设计说明书.pdf	2022/2/25 11:48	Microsoft Edge ...	477 KB
蓝牙应用详细设计说明书.pdf	2022/2/25 11:48	Microsoft Edge ...	647 KB
模式管理接口说明.pdf	2022/2/25 11:48	Microsoft Edge ...	881 KB

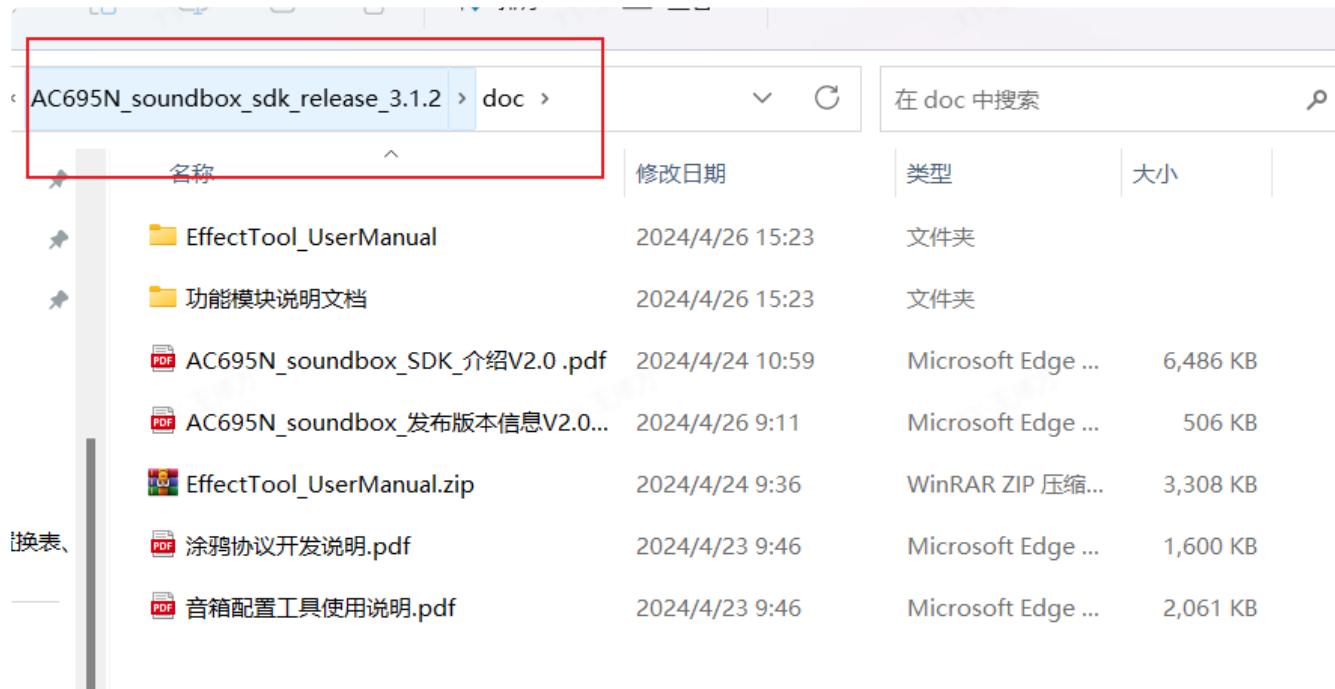
2. AC695N-SDK 资料阅读

前言：

对于每一个文档介绍的函数功能都尽可能地去调用编译到开发板上验证结果。

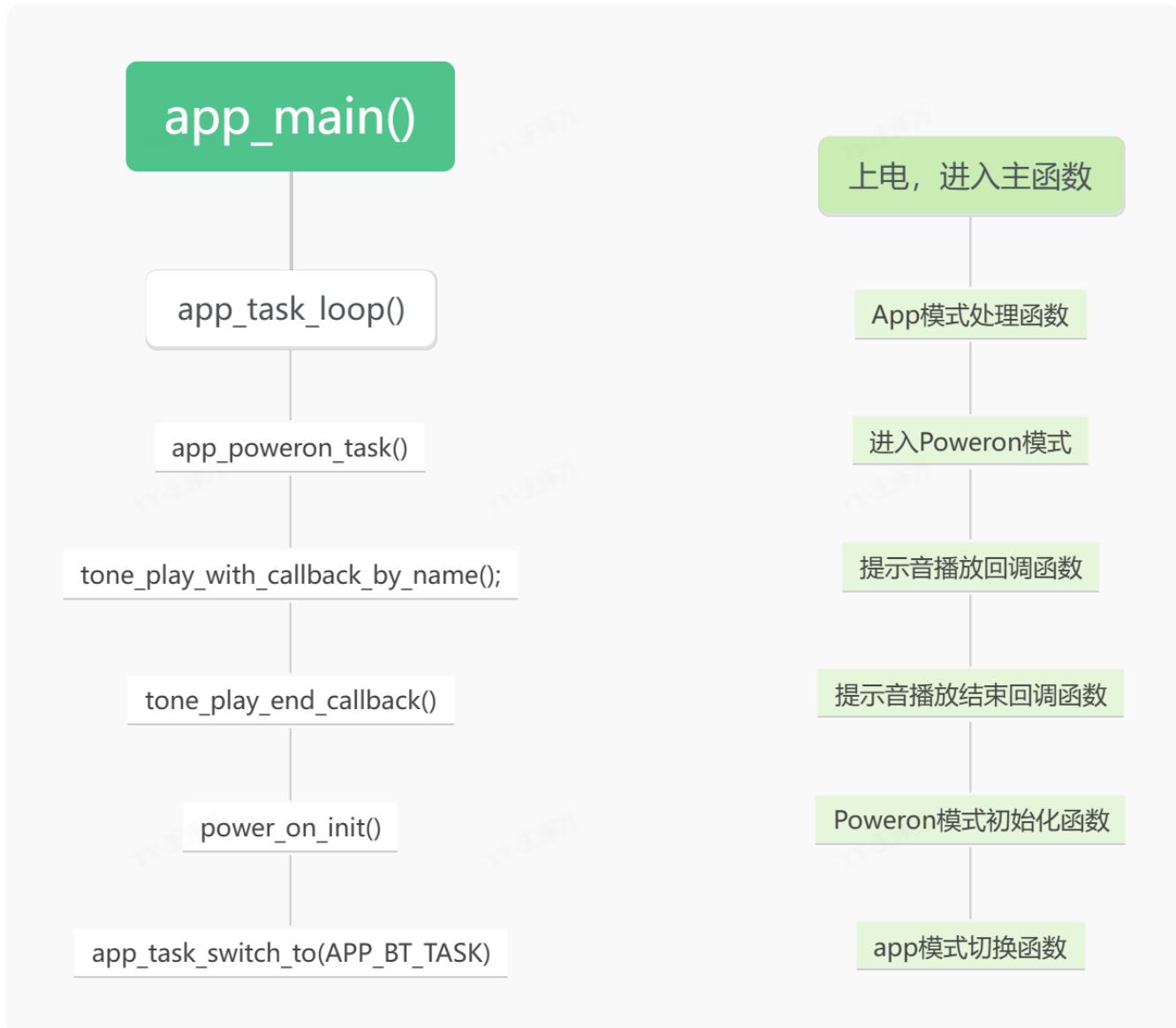
2.1 AC695N-SDK 介绍文档

之所以先使用 AC695N 的 SDK 来进行学习，是因为 AC695N 的 SDK 版本更新地更加完善，都更新到了 V3.1.2 版本了。



名称	修改日期	类型	大小
EffectTool_UserManual	2024/4/26 15:23	文件夹	
功能模块说明文档	2024/4/26 15:23	文件夹	
AC695N_soundbox_SDK_介绍V2.0.pdf	2024/4/24 10:59	Microsoft Edge ...	6,486 KB
AC695N_soundbox_发布版本信息V2.0...	2024/4/26 9:11	Microsoft Edge ...	506 KB
EffectTool_UserManual.zip	2024/4/24 9:36	WinRAR ZIP 压缩...	3,308 KB
涂鸦协议开发说明.pdf	2024/4/23 9:46	Microsoft Edge ...	1,600 KB
音箱配置工具使用说明.pdf	2024/4/23 9:46	Microsoft Edge ...	2,061 KB

2.1.1 模式流程的详细介绍



串口打印验证上述流程：

验证方法：在执行语句前加上打印语句

C/C++

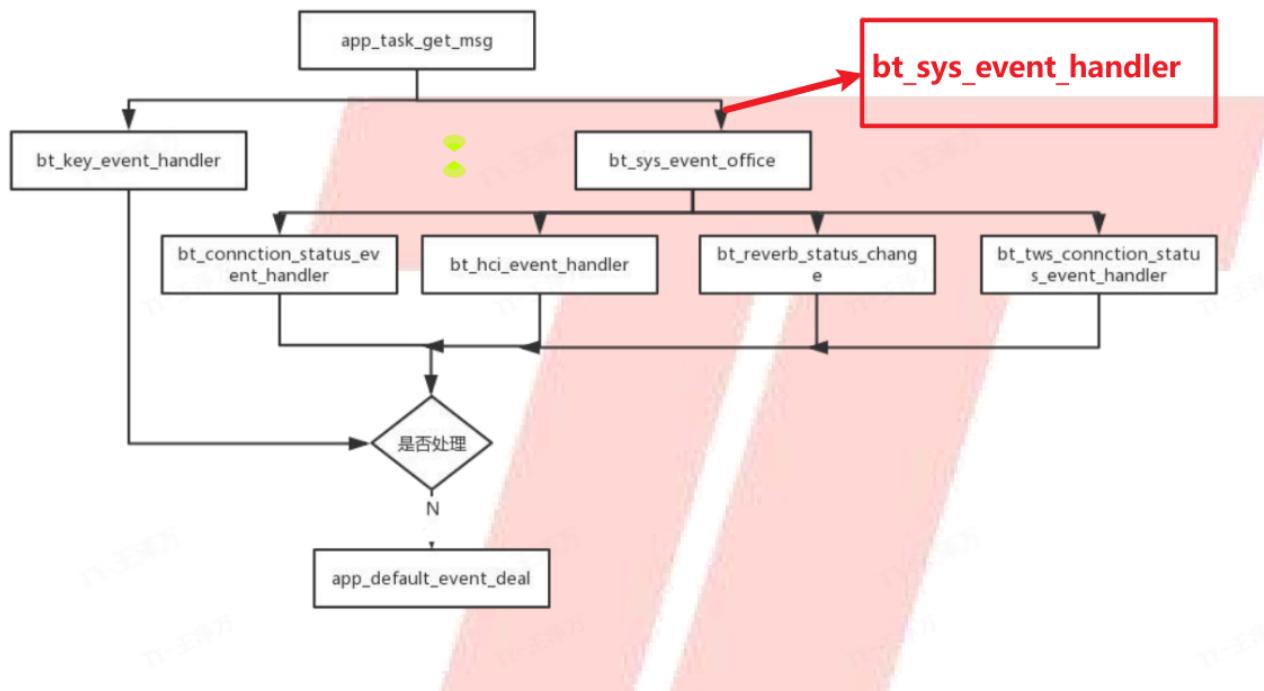
```
1 static void tone_play_end_callback(void *priv, int flag)
2 {
3     int index = (int)priv;
4
5     if (APP_POWERON_TASK != app_get_curr_task()) {
6         log_error("tone callback task out \n");
7         return;
8     }
9
10    switch (index) {
11        case INDEX_TONE_POWER_ON:
12            log_info("tone_play_end_callback()>power_on_init()\n");
13            //在执行语句前加上日志打印;
14            power_on_init();
15            break;
16    }
17 }
```

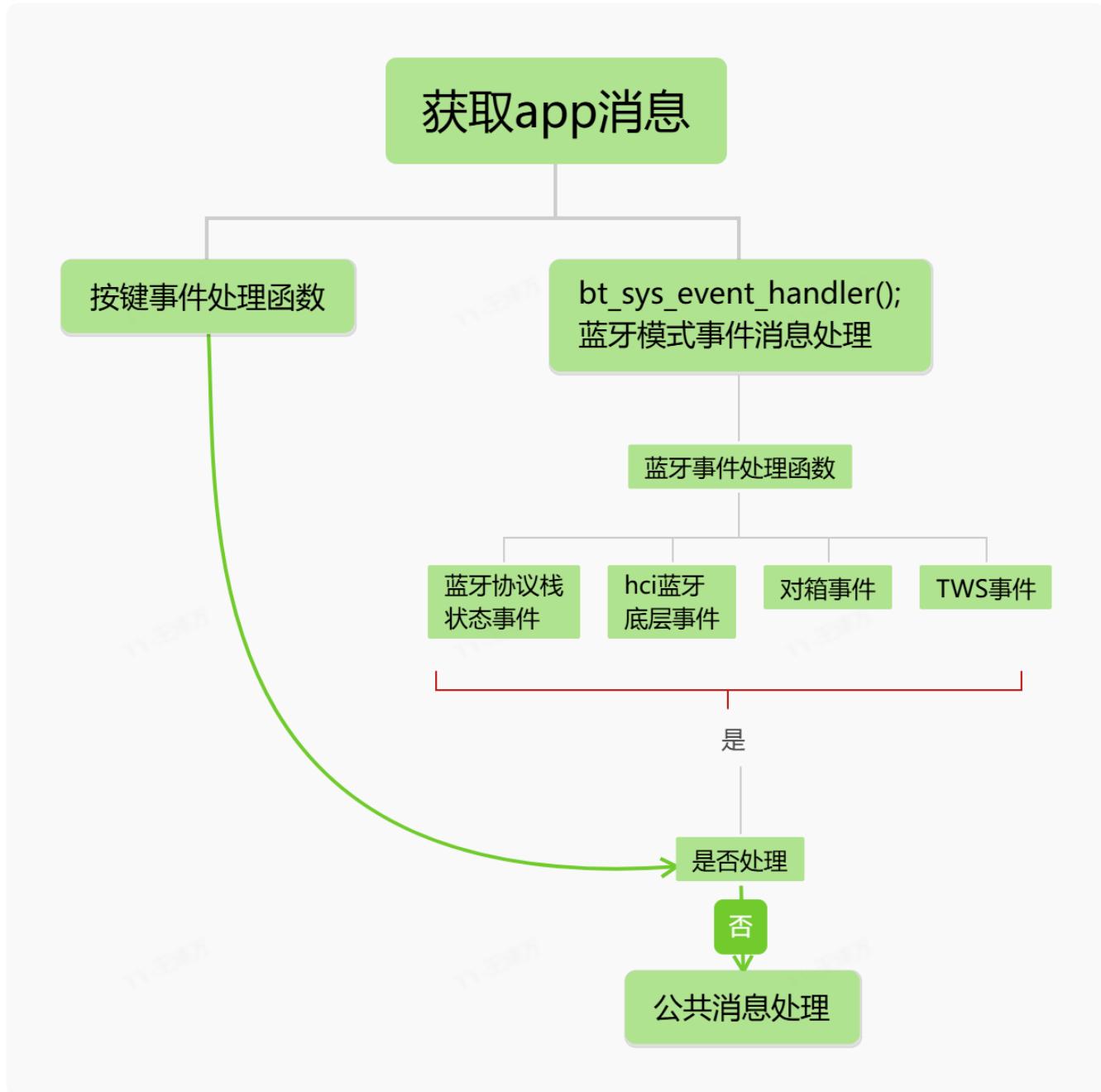
验证成功

```
[Info]: [APP]9:40->app_main  
[Info]: [APP]app_main  
[Info]: [APP]app_main()->app_task_loop()  
[Info]: [APP]APP_POWERON_TASK  
[Info]: [APP_IDLE]app_poweron_task()->tone_play_with_callback_by_name()  
[Info]: [AUDIO-DAC]sample 8000  
[Info]: [AUDIO-DAC]fifo init : 0xa30c 0x800 0x0 0x0 0x7ff  
[Info]: [AUDIO-DAC]DAC LR OUTPUT  
[Info]: [AUDIO-DAC]Audio dac fifo start : 350  
[Info]: [AUDIO-DAC]dac stop  
[Info]: [APP_IDLE]tone_play_end_callback()->power_on_init()  
[Info]: [APP_IDLE]power_on_init()->app_task_switch_to(APP_BT_TASK)
```

3. 蓝牙应用详细设计说明书阅读

3.1 蓝牙消息处理流程





串口打印验证上述流程：

验证方法：在执行语句前加上打印语句

C/C++

```
1 //截取部分代码
2 void app_bt_task()
3 {
4 .....
5 while (1) {
6     //日志打印语句
7     log_info("bt_task_start()->app_task_get_msg()\n");
8     app_task_get_msg(msg, ARRAY_SIZE(msg), 1);
9
10    switch (msg[0]) {
11        case APP_MSG_SYS_EVENT:
12            if (bt_sys_event_handler((struct sys_event *) (msg + 1)) =
= false) {
13                //日志打印语句
14                log_info("bt_sys_event_office()->app_default_event_deal
()\\n");
15                app_default_event_deal((struct sys_event *) (&msg[1]));
16            }
17            break;
18        default:
19            break;
20    }
21
22 .....
23 }
```

C/C++

```
1 //截取部分代码
2 int bt_sys_event_office(struct sys_event *event)
3 {
4     //日志打印语句
5     log_info("bt_sys_event_office()\n");
6     u8 ret = false;
7     if ((u32)event->arg == SYS_BT_EVENT_TYPE_CON_STATUS) {
8         //日志打印语句
9         log_info("bt_sys_event_office()->bt_connction_status_event_handler()\n");
10        bt_connction_status_event_handler(&event->u.bt);
11    } else if ((u32)event->arg == SYS_BT_EVENT_TYPE_HCI_STATUS) {
12        //日志打印语句
13        log_info("bt_sys_event_office()->bt_hci_event_handler()\n");
14        bt_hci_event_handler(&event->u.bt);
15    } else if ((u32)event->arg == SYS_BT_EVENT_FORM_SELF) {
16        bt_reverb_status_change(&event->u.bt);
17    }
18 ...
19 }
20
```

验证成功

```
18:47:21] 2119
18:47:21] 2120    le_support:1 1
18:47:21] 2121    le_config:0 0 0 0
18:47:22] 2122    [Info]: [BT]bt_sys_event_office()
18:47:22] 2123
18:47:22] 2124    [Info]: [BT]bt_sys_event_office()->bt_connction_status_event_handler()
18:47:22] 2125
18:47:22] 2126    [Debug]: [BT]-----bt_connction_status_event_handler 3
18:47:22] 2127    [Info]: [BT]Where is there?
18:47:22] 2128
18:47:22] 2129    [Info]: [BT]BT_STATUS_INIT_OK
18:47:22] 2130
18:47:22] 2131    [Debug]: [BT]connect_switch: 0, 0
18:47:22] 2132
18:47:22] 2133    [Debug]: [BT]is_1t2_connection:0          total_conn_dev:0
18:47:22] 2134
18:47:22] 2135    [Info]: [APP_ACTION]bt_sys_event_office()->app_default_event_deal()
18:47:22] 2136
```

```
[18:47:22] 2136 [Info]: [BT]bt_task_start()->app_task_get_msg()
[18:47:22] 2137
[18:47:22] 2138
[18:47:25] 2139
[18:47:25] 2140
[18:54:27] 2141
[18:54:27] 2142
[18:54:27] 2143
[18:54:27] 2144
[18:54:27] 2145
[18:54:27] 2146
[18:54:27] 2147
[18:54:27] 2148
[18:54:27] 2149
[18:54:27] 2150
[18:54:27] 2151
[18:54:27] 2152
[18:54:27] 2153

[Info]: [APP_AUDIO]VOL_SAVE

[Info]: [BT]bt_sys_event_office()

[Info]: [BT]bt_sys_event_office()->bt_hci_event_handler()

[Debug]: [BT]-----bt_hci_event_handler reason 3 13
[Info]: [BT]bt_hci_event_handler()->HCI_EVENT_CONNECTION_COMPLETE

[Info]: [BT] HCI_EVENT_CONNECTION_COMPLETE

[Debug]: [BT]is_1t2_connection:0          total_conn_dev:0

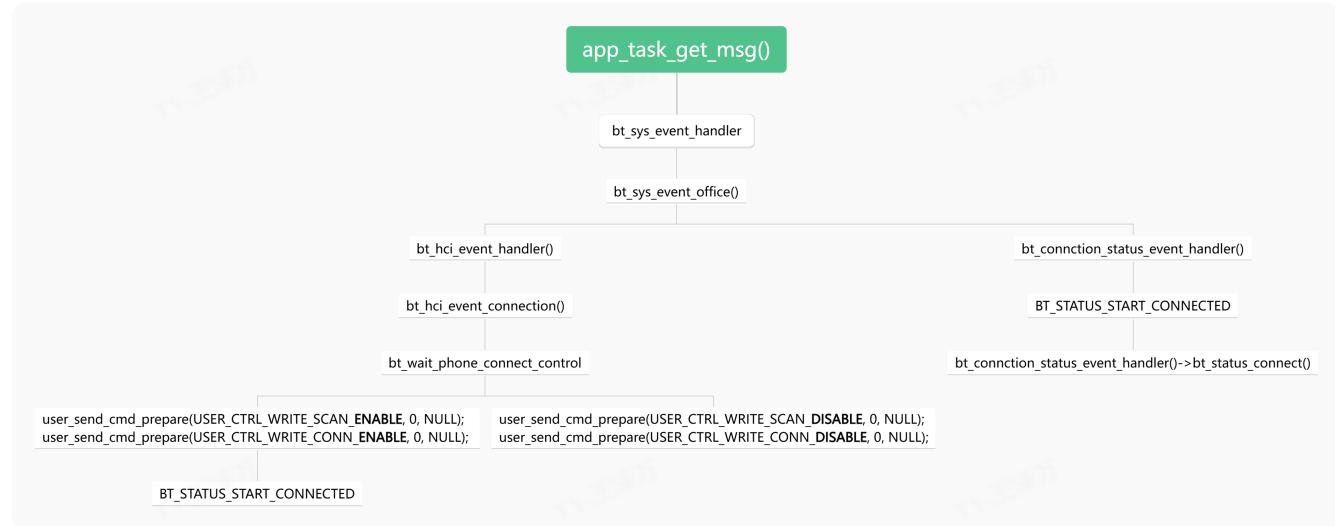
[Info]: [APP_ACTION]bt_sys_event_office()->app_default_event_deal()
```

如果事件在蓝牙模式里面没有处理，消息就会传递到 app_default_event_deal() 函数处理；

蓝牙模式消息处理函数 int bt_sys_event_handler(struct sys_event *event)，函数里面区分两种事件处理，分别为按键事件 SYS_KEY_EVENT，处理函数 bt_key_event_handler，蓝牙事件 SYS_BT_EVENT，处理函数 bt_sys_event_office，然后蓝牙事件下又分蓝牙状态事件、蓝牙协议栈事件、蓝牙对箱事件等，如果事件在蓝牙模式里面没有处理就会返回 false，消息会传到 app_default_event_deal 函数执行。

```
[18:47:22] 2129 [Info]: [BT]BT_STATUS_INIT_OK
[18:47:22] 2130
[18:47:22] 2131 [Debug]: [BT]connect_switch: 0, 0
[18:47:22] 2132
[18:47:22] 2133 [Debug]: [BT]is_1t2_connection:0          total_conn_dev:0
[18:47:22] 2134
[18:47:22] 2135 [Info]: [APP_ACTION]bt_sys_event_office()->app_default_event_deal()
[18:47:22] 2136
[18:47:22] 2137 [Info]: [BT]bt_task_start()->app_task_get_msg()
[18:47:22] 2138
[18:47:25] 2139 [Info]: [APP_AUDIO]VOL_SAVE
[18:47:25] 2140
[18:54:27] 2141 [Info]: [BT]bt_sys_event_office()
```

3.1.1 以手机连接蓝牙为例



app消息获取

蓝牙模式事件消息处理

蓝牙模式状态事件

蓝牙底层事件处理

蓝牙连接状态事件处理

蓝牙链接成功

子主题

C/C++

```
1 //截取部分代码
2 static int bt_hci_event_handler(struct bt_event *bt)
3 {
4     //日志打印
5     //对应原来的蓝牙连接上断开处理函数 ,bt->value=reason
6     log_debug ("-----bt_hci_event_handler reason %x %
7     x", bt->event, bt->value);
8     ...
9     switch (bt->event) {
10         case HCI_EVENT_INQUIRY_COMPLETE:
11             log_info (" HCI_EVENT_INQUIRY_COMPLETE \n");
12             bt_hci_event_inquiry(bt);
13             break;
14         ...
15         case BTSTACK_EVENT_HCI_CONNECTIONS_DELETE:
16         case HCI_EVENT_CONNECTION_COMPLETE:
17             //日志打印
18             log_info ("bt_hci_event_handler()->HCI_EVENT_CONNECTION_COMP
19             LETE\n");
20             log_info (" HCI_EVENT_CONNECTION_COMPLETE \n");
21             switch (bt->value) {
22                 case ERROR_CODE_SUCCESS :
23                     //日志打印
24                     log_info ("ERROR_CODE_SUCCESS \n");
25                     bt_hci_event_connection(bt);
26             break;
27             ...
28 }
```

等待手机连接

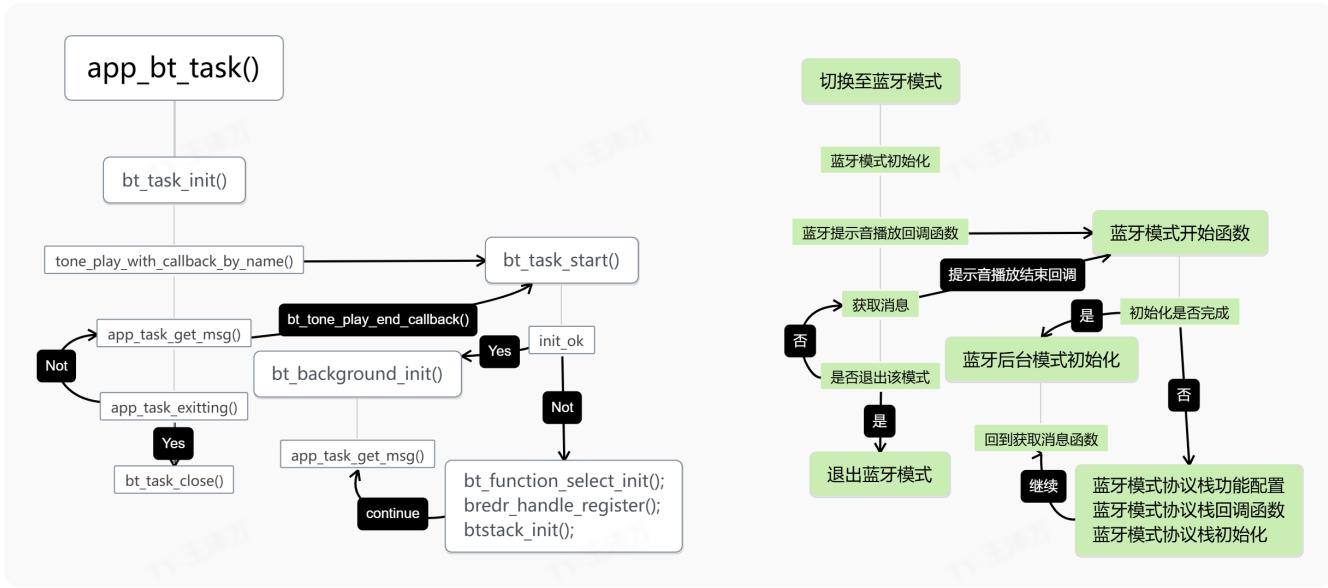
```
[20:50:49] 2527           le_support:1 1
[20:50:49] 2528           le_config:0 0 0 0
[20:50:49] 2529
[20:50:49] 2530           [Info]: [BT]bt_sys_event_office()
[20:50:49] 2531
[20:50:49] 2532           [Info]: [BT]bt_sys_event_office()->bt_connction_status_event_handler()
[20:50:49] 2533
[20:50:49] 2534           [Debug]: [BT]-----bt_connction_status_event_handler 3
[20:50:49] 2535           [Info]: [BT]Where is there?
[20:50:49] 2536
[20:50:49] 2537           [Info]: [BT]BT_STATUS_INIT_OK
[20:50:49] 2538
[20:50:49] 2539           [Debug]: [BT]connect_switch: 0, 0
[20:50:49] 2540
[20:50:49] 2541           [Info]: [BT]bt_hci_event_connection()->bt_wait_phone_connect_control()
[20:50:49] 2542
[20:50:49] 2543           [Debug]: [BT]is_1t2_connection:0          total_conn_dev:0
[20:50:49] 2544           [Info]: [BT]USER_CTRL_WRITE_SCAN_ENABLE
[20:50:49] 2545           USER_CTRL_WRITE_CONN_ENABLE
[20:50:49] 2546
[20:50:49] 2547           [Info]: [APP_ACTION]bt_sys_event_office()->app_default_event_deal()
[20:50:49] 2548
[20:50:49] 2549           [Info]: [BT]bt_task_start()->app_task_get_msg()
[20:50:49] 2550
[20:50:53] 2551           [Info]: [APP_AUDIO]VOL_SAVE
[20:50:53] 2552
[20:50:53] 2553
```

连接成功

```
[20:53:47] 2553 [Info]: [BT]bt_sys_event_office()
[20:53:47] 2554 [Info]: [BT]bt_sys_event_office()->bt_hci_event_handler()
[20:53:47] 2555 [Debug]: [BT]-----bt_hci_event_handler_reason_3 13
[20:53:47] 2558 [Info]: [BT]bt_hci_event_handler()->HCI_EVENT_CONNECTION_COMPLETE
[20:53:47] 2559 [Info]: [BT] HCI_EVENT_CONNECTION_COMPLETE
[20:53:47] 2560 [Info]: [BT]bt_hci_event_connection()->bt_wait_phone_connect_control()
[20:53:47] 2561 [20:53:47] 2562 [Info]: [BT] HCI_EVENT_CONNECTION_COMPLETE
[20:53:47] 2563 [Info]: [BT]bt_hci_event_connection()->bt_wait_phone_connect_control()
[20:53:47] 2564 [Debug]: [BT]is_1t2_connection:0 total_conn_dev:0
[20:53:47] 2565 [20:53:47] 2566 [Info]: [BT]USER_CTRL_WRITE_SCAN_ENABLE
[20:53:47] 2567 [Info]: [BT]USER_CTRL_WRITE_CONN_ENABLE
[20:53:47] 2568 [Info]: [APP_ACTION]bt_sys_event_office()->app_default_event_deal()
[20:53:47] 2569 [Info]: [BT]bt_task_start()->app_task_get_msg()
[20:53:47] 2570 [Info]: [BT]bt_sys_event_office()
[20:53:47] 2571 [20:53:47] 2572 [Info]: [BT]bt_sys_event_office()->bt_connction_status_event_handler()
[20:53:47] 2573 [20:53:47] 2574 [Info]: [BT]bt_sys_event_office()
[20:53:47] 2575 [20:53:47] 2576 [Info]: [BT]bt_sys_event_office()->bt_connction_status_event_handler()
[20:53:47] 2577 [20:53:47] 2578 [Info]: [BT]-----bt_connction_status_event_handler_5
[20:53:47] 2579 [Info]: [BT] BT_STATUS_START_CONNECTED
[20:53:47] 2580 [Info]: [APP_ACTION]bt_sys_event_office()->app_default_event_deal()
[20:53:47] 2581 [Info]: [BT]bt_task_start()->app_task_get_msg()
[20:53:47] 2582 [Info]: [BT]remote_name : Art Pig Phone
[20:53:47] 2583 [20:53:47] 2584 [20:53:47] 2585 [20:53:47] 2586 [20:53:47] 2587
```

3.2 蓝牙进入退出模式流程

未连接设备



串口打印验证上述流程：

验证方法：在执行语句前加上打印语句

C/C++

```
1 //部分代码
2 void app_bt_task()
3 {
4     int res;
5     int msg[32];
6     ui_update_status(STATUS_EXIT_LOWPOWER);
7     //日志打印语句
8     log_info("app_bt_task()->bt_task_init()\n");
9     bt_task_init(); //初始化变量、时钟、显示(未进行协议栈初始化)
10
11 #if TCFG_TONE2TWS_ENABLE
12     extern void tone2tws_bt_task_start(u8 tone_play);
13     tone2tws_bt_task_start(!__this->cmd_flag);
14 #endif
15
16     extern u8 get_tws_background_connected_flag();
17     if (!__this->cmd_flag && (!get_tws_background_connected_flag())) {
18         //蓝牙后台拉回蓝牙模式不播放提示音
19         //日志打印语句
20         log_info("init_ok->tone_play_with_callback_by_name()\n");
21         tone_play_with_callback_by_name(tone_table[INDEX_TONE_BT_MODE],
22                                         1, bt_tone_play_end_callback, (void *)INDEX_TONE_BT_MODE);
23         //协议栈初始化在提示音结束进行
24     }
25     ....... //省略的代码
26
27     while (1) {
28         //日志打印语句
29         log_info("bt_task_start()->app_task_get_msg()\n");
30         app_task_get_msg(msg, ARRAY_SIZE(msg), 1);
31
32         switch (msg[0]) {
33             case APP_MSG_SYS_EVENT:
34                 if (bt_sys_event_handler((struct sys_event *) (msg + 1)) ==
35                     = false) {
36                     app_default_event_deal((struct sys_event *) (&msg[1]));
37                 }
38                 break;
39         }
40     }
41 }
```

```
36     default:  
37         break;  
38     }  
39 ...  
40 }
```

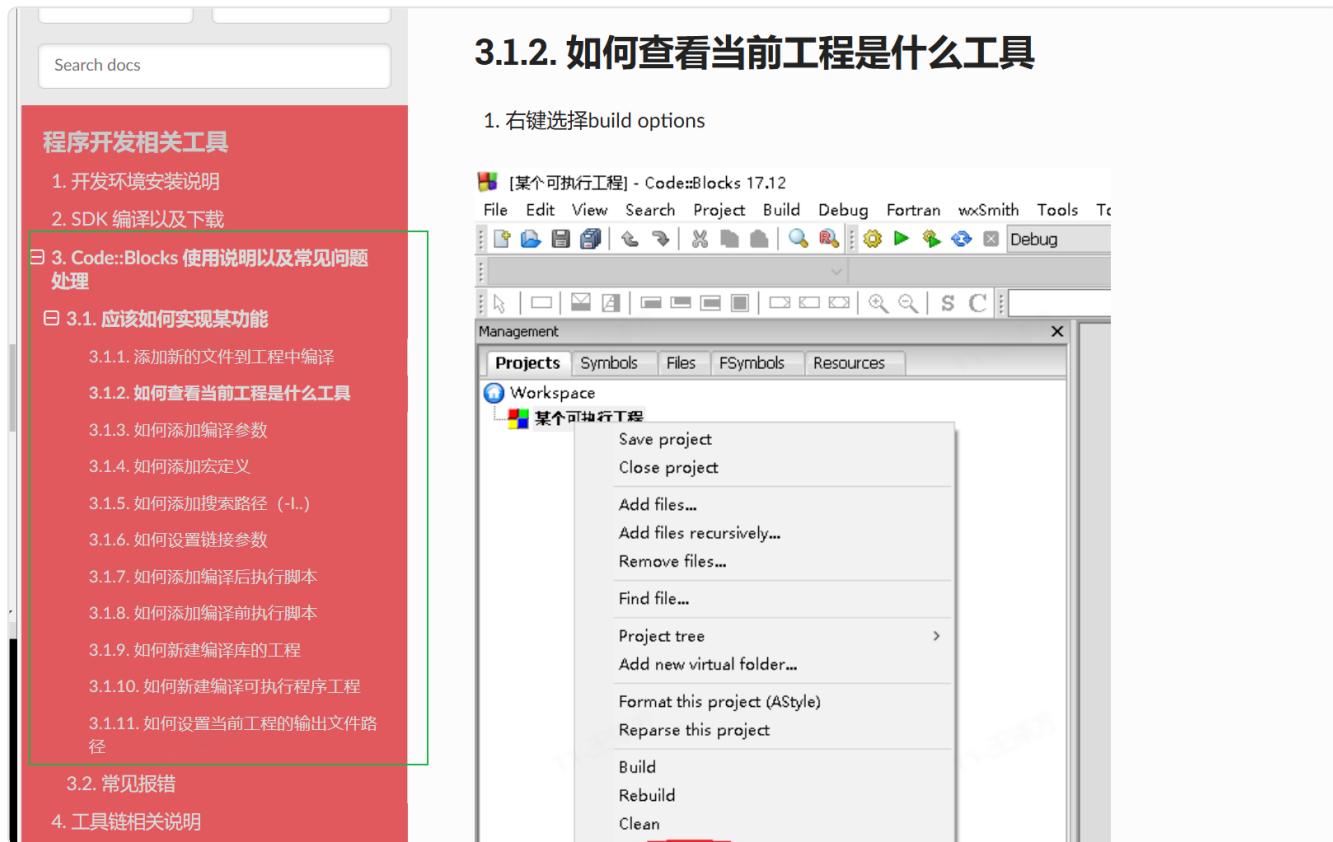
验证成功

```
[Info]: [APP]APP_BT_TASK  
  
[Info]: [BT]app_bt_task()->bt_task_init()  
  
[Info]: [BT]init_ok->tone_play_with_callback_by_name()  
  
[Info]: [AUDIO-DAC]sample 8000  
  
[Info]: [AUDIO-DAC]fifo init : 0xa30c 0x800 0x0 0x0 0x7ff  
  
[Info]: [AUDIO-DAC]DAC LR OUTPUT  
  
[Info]: [AUDIO-DAC]Audio dac fifo start : 350  
  
[Info]: [BT]bt_task_start()->app_task_get_msg()  
  
[Info]: [AUDIO-DAC]dac stop  
  
[Info]: [BT]bt_tone_play_end_callback()->bt_task_start()  
  
[Info]: [BT]__this->init_ok->NOT  
  
[Info]: [BT]bt_task_start()->bt_function_select_init()  
bredr_handle_register()  
btstack_init()  
  
le_support:1 1  
le_config:0 0 0 0  
[Debug]: [BT]-----bt_connction_status_event_handler 3  
[Info]: [BT]BT_STATUS_INIT_OK  
  
[Debug]: [BT]connect_switch: 0, 0  
  
[Debug]: [BT]is_1t2_connection:0          total_conn_dev:0  
  
[Info]: [BT]bt_task_start()->app_task_get_msg()
```

4. 开发环境配置以及出现的问题

技术支持：关 FH& 李 TY

4.1 Code Block 常见问题



The screenshot shows the Code::Blocks IDE interface. On the left, there is a sidebar with a search bar at the top and a red panel titled "程序开发相关工具" (Development Tools). This panel contains several sections: "1. 开发环境安装说明", "2. SDK 编译以及下载", and "3. Code::Blocks 使用说明以及常见问题处理". The third section is expanded, showing a list of 11 sub-topics under "3.1. 应该如何实现某功能", with the second item, "3.1.2. 如何查看当前工程是什么工具", highlighted by a green box. Below this, there are sections for "3.2. 常见报错" and "4. 工具链相关说明". On the right, the main workspace shows the "Management" window with the "Projects" tab selected. A context menu is open over a project node named "某个可执行工程", listing options like "Save project", "Close project", "Add files...", etc.

[官方文档入口](#)：遇到问题可以去官方文档查看

4.1.1 Code Block 编译无法生成 tools 下载文件

缺少 tools 文件夹

Branch_c015e055 > 20240828161628		▼	C	在 20240828161628 中
名称		修改日期	类型	
readme.txt	2024/8/28 16:16		文本文档	
YTS[Branch_c015e055]-主控AC7003D...	2024/8/28 16:16		WinRAR 压缩文件	

将文件名里的'-master'删掉，并且移动至 C 盘根目录

名称	修改日期	类型	大小
pack-tools-master	2024/7/16 16:31	文件夹	
usertool-master	2024/6/17 17:50	文件夹	
pack-tools-master.zip	2024/8/28 15:53	WinRAR ZIP 压缩...	21,386 KB
usertool-master.zip	2024/8/28 15:53	WinRAR ZIP 压缩...	21,386 KB

电脑 > 本地磁盘 (C:)

名称	修改日期	类型	大小
JL	2024/8/13 16:22	文件夹	
JL_Studio	2024/8/22 21:14	文件夹	
jltools	2024/8/24 9:43	文件夹	
NVIDIA	2024/3/22 20:04	文件夹	
pack-tools	2024/8/28 20:19	文件夹	
PerfLogs	2022/5/7 13:24	文件夹	
Program Files	2024/8/21 9:28	文件夹	
Program Files (x86)	2024/8/12 15:11	文件夹	
ProgramData	2024/8/22 16:26	文件夹	
Recovery	2024/3/22 20:16	文件夹	
SolidWorks_Flexnet_Server	2024/8/28 12:21	文件夹	
usertool	2024/8/28 19:20	文件夹	
Windows	2024/8/22 19:30	文件夹	

成功编译生成 tools 文件夹

Branch_c015e055 > 20240828201929 > ▼ C 在 20240828201929 中搜索

名称	修改日期	类型	大小
tools	2024/8/28 20:19	文件夹	
readme.txt	2024/8/28 20:19	文本文档	
YTS[Branch_c015e055][主左]-主控AC7...	2024/8/28 20:19	WinRAR 压缩文件	16,

4.1.2 克隆的文件无法在 Code Blocks 上跳出日志

解决方法：

直接使用 Beyond Compare 文件对比工具，对比能正常打印出日志的文件，可以发现 Release_file.bat 文件的最后一行不同，直接修改成一样的；

The screenshot shows two windows of the Beyond Compare file comparison tool. Both windows display the same file content: Release_file.bat. The left window shows the original code, and the right window shows the modified code. A red arrow points from the original 'call C:\pack-tools\run.bat' line in the left window to the modified 'call C:\pack-tools\\$run.bat' line in the right window. The text '原本这里是不一样的，改成一样的' is overlaid in red at the bottom center of the comparison area.

```
@REM 发布文件目录
set "RELEASE_PATH=..\..\release"
@REM 打包源 目录或者文件
set "PACKAGE_SOURCE=cpu\br36\tools"
@REM Lod bat
set "LOD_BAT=cpu\br36\tools\download.bat"
@REM Fw文件
set "FW_FILE=cpu\br36\tools\download\earphone\jl_isd.fw"
@REM Log 文件
set "LOG_FILE=cpu\br36\tools\download\earphoneinfo.log"
@REM 是否开启git 远程仓库检测
set "GET_CHECK_EN=Y"
@REM 备份sdk源码路径 针对以前692的项目没有使用最新的打包流程
@REM set "SDK_BACK_PATH=Source_code\AC692x_SDK_release_V2.4"
@REM 不需要打包的文件类型 只能设置一个类型更多类型可以在C:\usertool\7z\7z.bat中设置
@REM set "EXCLUDE_TYPE=*.key"

@REM 运行打包处理
call C:\pack-tools\run.bat

pause
exit
```

```
@REM 发布文件目录
set "RELEASE_PATH=..\..\release"
@REM 打包源 目录或者文件
set "PACKAGE_SOURCE=cpu\br36\tools"
@REM Lod bat
set "LOD_BAT=cpu\br36\tools\download.bat"
@REM Fw文件
set "FW_FILE=cpu\br36\tools\download\earphone\jl_isd.fw"
@REM Log 文件
set "LOG_FILE=cpu\br36\tools\download\earphoneinfo.log"
@REM 是否开启git 远程仓库检测
set "GET_CHECK_EN=Y"
@REM 备份sdk源码路径 针对以前692的项目没有使用最新的打包流程
@REM set "SDK_BACK_PATH=Source_code\AC692x_SDK_release_V2.4"
@REM 不需要打包的文件类型 只能设置一个类型更多类型可以在C:\usertool\7z\7z.bat中设置
@REM set "EXCLUDE_TYPE=*.key"

@REM 运行打包处理
call C:\pack-tools\$run.bat

pause
exit
```

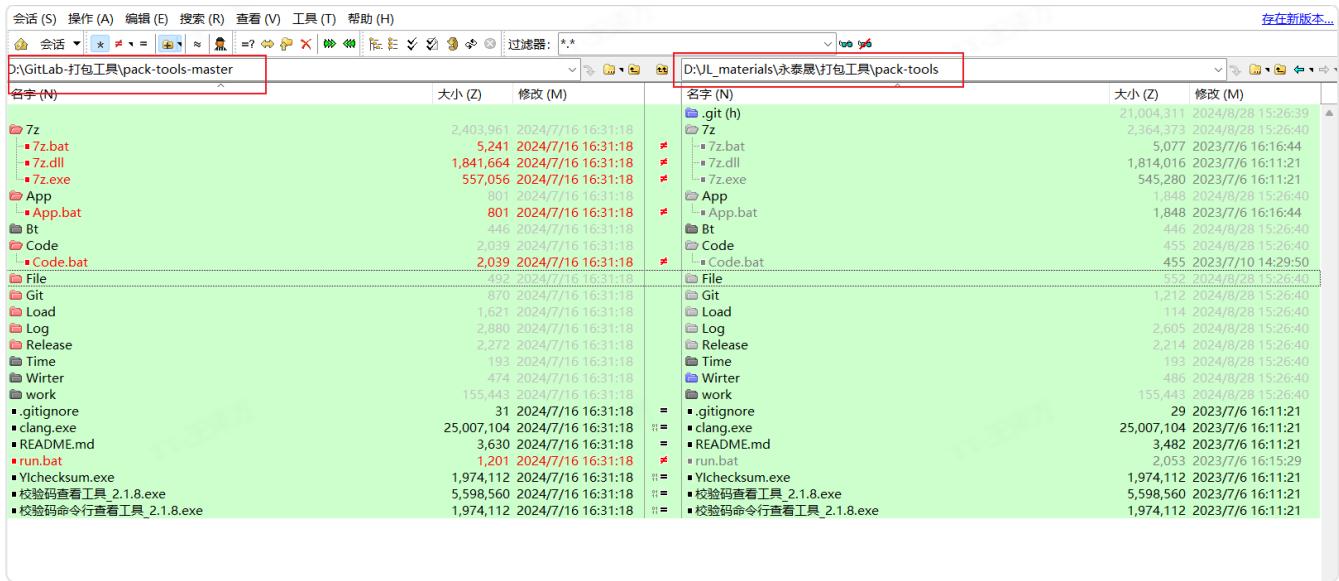
原本这里是不一样的，改成一样的

修改前与修改后的文件详细可以到 GitLab 的推送信息查看，红色为修改前的部分，绿色为修改后的部分；

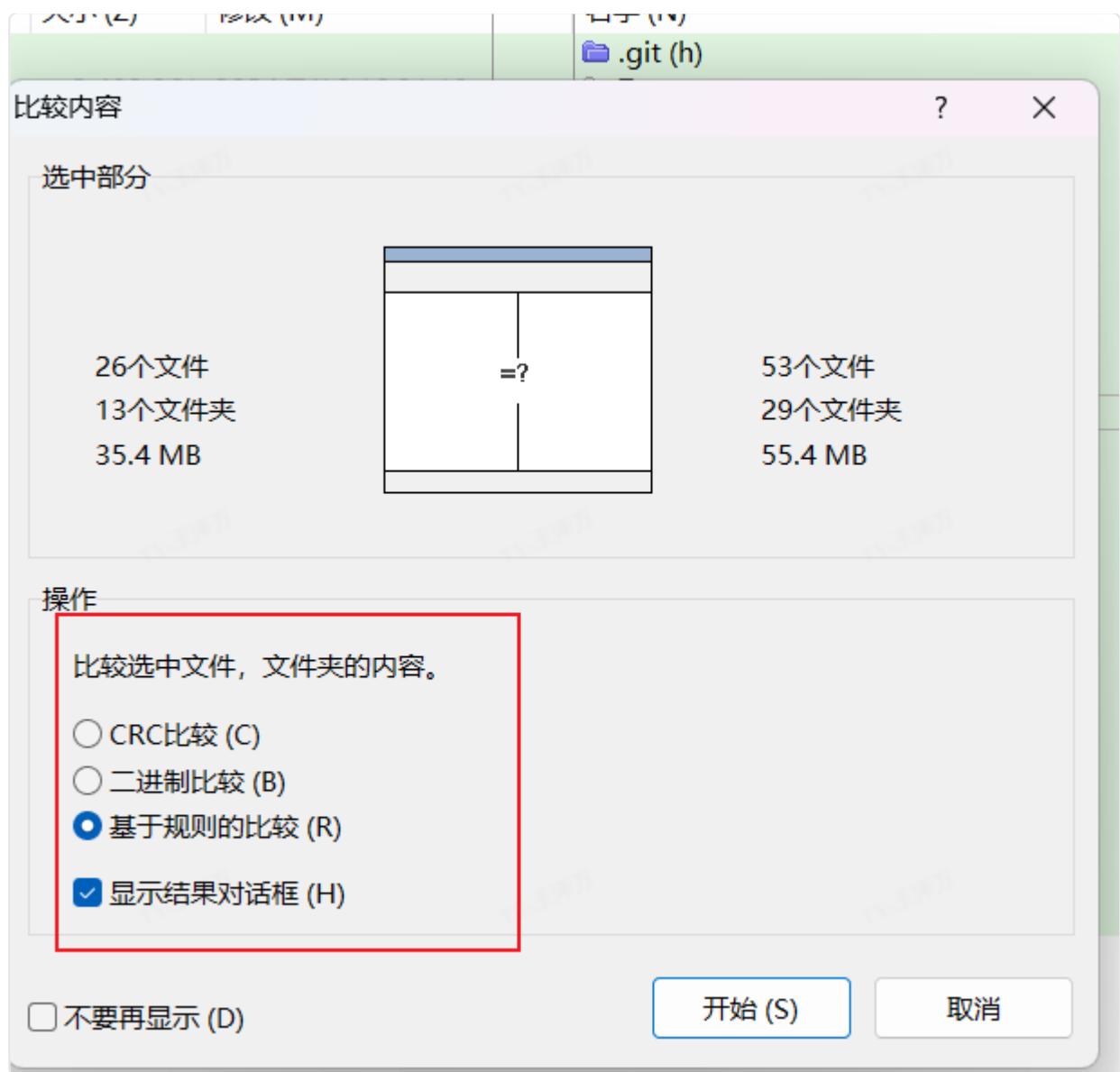
The screenshot shows a commit diff in a GitLab interface. The file Release_file.bat is shown with a red box around the modified line. The original code 'call C:\pack-tools\run.bat' is in red, and the modified code 'call C:\pack-tools\\$run.bat' is in green. The commit message 'View file @7c10cf58' is visible at the top right.

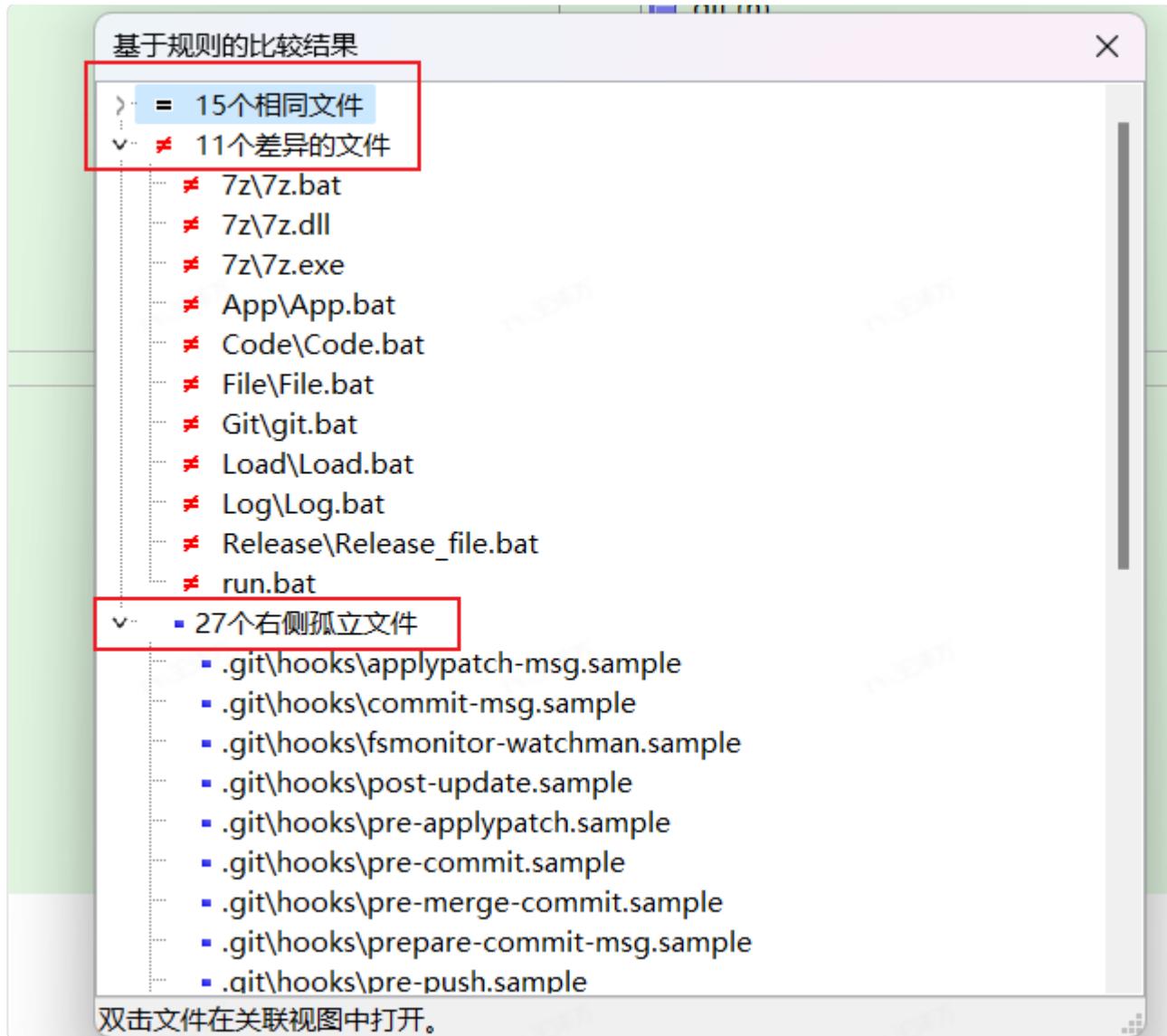
```
...    ... @@ -30,7 +30,7 @@ set "GET_CHECK_EN=Y"
30      30     @REM set "EXCLUDE_TYPE=*.key"
31      31
32      32     @REM 运行打包处理
33      33     - call C:\usertool\run.bat
34      34     + call C:\pack-tools\$run.bat
35      35
36      36
...
```

4.1.3 Beyond Compare 文件对比工具



通过文件对比工具，可以了解到官网打包工具与耳机所用的打包工具的不同





比较结果为两个打包工具的版本号不同，官网的为最新的

```
D:\GitLab-打包工具\pack-tools-master\App\App.bat 2024/7/16 16:31:18 801 字节 <默认> ▾ UTF-8 ▾ PC
@echo off
chcp 65001 1>nul
@REM chcp 936 1>nul
cd /d %~dp0

if exist "%Package_Path%\%APPLICATION_PATH%\Precompiled\run.bat" (
    call %Package_Path%\%APPLICATION_PATH%\Precompiled\run.bat Y
)

@REM if "%1" EQU "app" (
@REM     exit
@REM )

@REM if "%APPLICATION_PATH%" NEQ "" ( echo %APPLICATION_PATH% ) else ( goto exit_
@REM echo %Package_Path%\%APPLICATION_PATH%\Precompiled\run.bat
@REM call %Package_Path%\%APPLICATION_PATH%\Precompiled\run.bat

@REM set "INF_PATH=%Package_Path%\%APPLICATION_PATH%\info"
@REM set "CP_TAR=%Package_Path%\%APP_BIN_PATH%"

@REM echo %INF_PATH%
@REM echo %CP_TAR%

@REM 复制执行子模块预编译需要的文件
@REM copy %INF_PATH%\Precompiled.* %CP_TAR% -Y 1>nul
```

```
D:\JL_materials\永泰晨\打包工具\pack-tools\App\App.bat 2023/7/6 16:16:44 1,848 字节 <默认> ▾ UTF-8 ▾ PC
@echo off
chcp 65001 1>nul
@REM chcp 936 1>nul
cd /d %~dp0

if "%APPLICATION_PATH%" NEQ "" ( echo %APPLICATION_PATH% ) else ( goto exit_flag

set "INF_PATH=%Package_Path%\%APPLICATION_PATH%\info"
set "CP_TAR=%Package_Path%\%APP_BIN_PATH%"

echo %INF_PATH%
echo %CP_TAR%

@REM 复制执行子模块预编译需要的文件
@REM copy %INF_PATH%\Precompiled.* %CP_TAR% -Y 1>nul
```

5. 永**客户无线耳机调试

注意事项：一般客户都是用*Code Blocks*编译的，为了保证不出错，因此*Vs Code*建议仅作为代码修改，查阅，工程师与客户同步使用*Code Blocks*编译。

5.1 调节耳机音量断连问题

问题出现现象

5.1.1 测试手动控制音量断连问题

测试时间：11: 17-11: 43 (26min) , 测试手机：华为mate 30 pro, 蓝牙版本：v5.1 系统：鸿蒙（安卓）

测试结果：未出现异常；

测试时间：13: 49-14: 00 (11min) ,

13: 57，播放音乐时，控制左耳降低音量时出现一次异常，声音像暂停了1s;

14:00, 播放音乐时，控制左耳降低音量时出现一次异常，声音卡顿；

测试时间：14: 16-14:23 (7min)

14: 18，播放音乐时，手动控制右耳提高音量时出现一次异常，右边耳机声音卡顿；

14: 23，播放音乐时，手动控制左耳降低音量时出现一次异常，左边耳机声音卡顿；

测试时间：14: 43-15:03 (20min) , 测试手机：真我GT, 蓝牙版本：v5.2 系统：安卓

14: 55，播放音乐时，手动控制左耳降低音量时出现一次异常，左边耳机声音卡顿；

15: 03，播放音乐时，手动控制右耳提高音量时出现一次异常，右边耳机声音卡顿；

验证结果：

存在客户所测试到的问题，并且与客户所测试的问题现象一致；

5.1.2 测试自动控制音量断连问题

8.29: 耳机自动调节音量测试

测试时间: 16: 56-17:40 (44min) , 测试手机: 华为mate 30 pro, 蓝牙版本: v5.1 系统: 鸿蒙
(安卓)

测试结果: 无异常

8.29: 耳机自动调节音量测试

测试时间: 18: 31-18:52 (21min) , 测试手机: 荣耀x50, 蓝牙版本: v5.1 系统: 安卓

测试结果: 无异常

8.29: 耳机手动调节音量测试

测试时间: 18: 57-19:57 (1h) , 测试手机: 荣耀x50, 蓝牙版本: v5.1 系统: 安卓

19: 28, 播放音乐时, 手动控制左耳降低音量时出现一次异常, 左边耳机声音卡顿;

8.29: 耳机手动调节音量测试

测试时间: 20: 02-20:23 (21min) , 测试手机: 华为mate 30 pro, 蓝牙版本: v5.1 系统: 鸿蒙
(安卓)

20: 23, 播放音乐时, 手动控制右耳降低音量时出现一次异常, 右边耳机声音卡顿;

测试结果:

自动调节音量中未遇到蓝牙断连情况, 断连可能是因为手动控制音量时挡到了蓝牙天线;

5.2 客户需求：取消耳机拔出自动开机启动功能

找到板级配置文件 board_config.h

```
Terminal Help ← → ac700n_earphone [Administrator]
C board_config.h C board_ac700n_demo_cfg.h M Makefile
apps > earphone > board > br36 > C board_ac700n_demo_cfg.h > ...
546 #define TCFG_CHARGE_CALIBRATION_ENABLE DISABLE_THIS_MOUDLE //是否支持充电
547 #if TCFG_AUDIO_ANC_ENABLE
548 #define TCFG_ANC_BOX_ENABLE 1 //是否支持ANC测试盒
549 #else
550 #define TCFG_ANC_BOX_ENABLE 0 //是否支持ANC测试盒
551 #endif/*TCFG_AUDIO_ANC_ENABLE*/
552 #define TCFG_CHARGESTORE_PORT IO_PORTP_00 //耳机和充点仓通讯的IO口
553 #define TCFG_CHARGESTORE_UART_ID IRQ_UART1_IDX //通讯使用的串口号
554
555 //*****充电参数配置*****
556 // 充电参数配置
557 //*****充电参数配置*****
558 //是否支持芯片内置充电
559 #define TCFG_CHARGE_ENABLE ENABLE_THIS_MOUDLE
560 //是否支持开机充电
561 #define TCFG_CHARGE_POWERON_ENABLE DISABLE
562 //是否支持自动开机功能
563 #define TCFG_CHARGE_OFF_POWERON_NE DISABLE //enable->disable, 关闭拔出启动蓝牙
564 /*充电截止电压可选配置*/
565 #define TCFG_CHARGE_FULL_V CHARGE_FULL_V_4199
566 /*充电截止电流可选配置*/
567 #define TCFG_CHARGE_FULL_MA CHARGE_FULL_MA_10
568 /*充电电流可选配置*/
569 #define TCFG_CHARGE_MA CHARGE_MA_50
570 #if (TCFG_CHARGE_ENABLE == DISABLE_THIS_MOUDLE) || (TCFG_CHARGE_POWERON_ENABLE == ENABLE)
571 #undef TCFG_CHARGE_CALIBRATION_ENABLE
572 #define TCFG_CHARGE_CALIBRATION_ENABLE DISABLE_THIS_MOUDLE
573 #endif
```

PROBLEMS 103 OUTPUT DEBUG CONSOLE TERMINAL

Ctrl+F 搜索 “” 拔出 “”，将拔出自动开机功能的宏定义由“ENABLE”改成“DISABLE”；

Ctrl + S 保存，退出 Vs Code，到 Code Blocks 进行编译；

5.2.1 升级文件提交给客户

注意事项：

查看项目名是否与客户的设备名称一致，修改负责人名字缩写，填上更新内容；

删除为更新内容的日志；

文件 编辑 查看

时间:20240828201929

作者:GFH

项目名:Branch_c015e055

主控:AC7003D4

蓝牙名:"G5-HEAR"

SDK版本:"1.3.6"

声道配置方式:"主左"

GIT分支:"Branch_c015e055"

GIT远端地址:"http://192.168.1.245/freebrand/yts/ac700n_earphone.git"

本地地址:"D:\JL_materials\永泰晟\永泰晟\ac700n_earphone"

更新内容:

1.

时间:20240828192135

作者:GFH

项目名:Branch_c015e055

主控:AC7003D4

蓝牙名:"G5-HEAR"

SDK版本:"1.3.6"

声道配置方式:"主左"

GIT分支:"Branch_c015e055"

GIT远端地址:"http://192.168.1.245/freebrand/yts/ac700n_earphone.git"

本地地址:"D:\JL_materials\永泰晟\永泰晟\ac700n_earphone"

更新内容:

1.

时间:20240828191342

作者:WZW

项目名:Branch_c015e055

主控:AC7003D4

蓝牙名:"G5-HEAR"

SDK版本:"1.3.6"

GIT分支:"Branch_c015e055"

GIT远端地址:"http://192.168.1.245/freebrand/yts/ac700n_earphone.git"

本地地址:"D:\JL_materials\永泰晟\永泰晟\ac700n_earphone"

更新内容:

1.

YTS[G5-AC7003D4-ANCG5-V136-G5-HEAR]

[主左]-主控AC7003D4-[设备名G5-HEAR](校验)

码DF68-EC16DE32)(负责人

GFH)-20240611111434

@关富华 以上软件帮忙改一版
去掉拿起开机功能，保留手动开机功能

修改之后

将文件压缩打包给客户

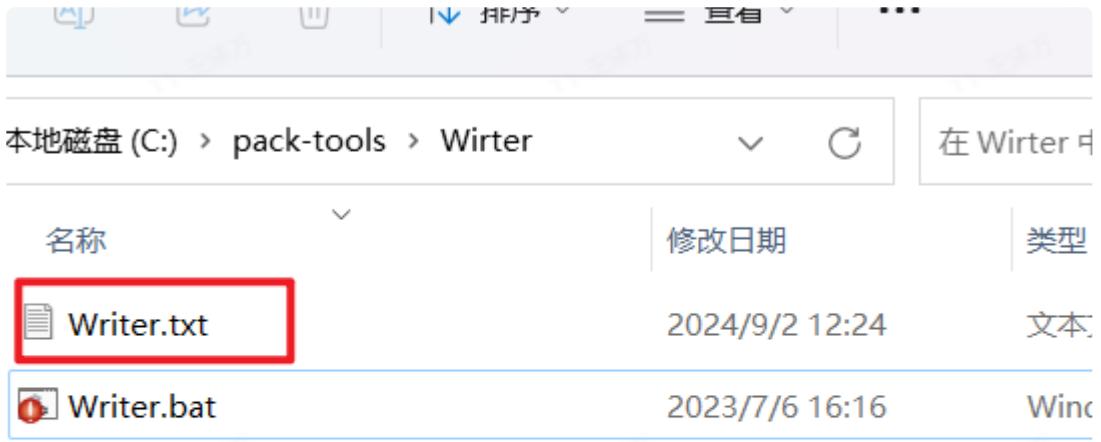
时间:20240828201929
作者:WZW
项目名:Branch_c015e055
主控:AC7003D4
蓝牙名:"G5-HEAR"
SDK版本:"1.3.6"
声道配置方式:"主左"
GIT分支:"Branch_c015e055"
GIT远端地址:"http://192.168.1.245/freebrand/yts/ac700n_earphone.git"
本地地址:"D:\JL materials\永泰晟\永泰晟\ac700n_earphone"
更新内容:
1.去掉拿起开机功能，保留手动开机功能

YTS[G5-AC7003D4-ANCG5-V136-G5-HEAR]
[主左]-主控AC7003D4-[设备名G5-HEAR]校验
码DF68-EC16DE32}{负责人
GFH}-20240611111434
@以上软件帮忙改一版
去掉拿起开机功能，保留手动开机功能

时间:20240611111434
作者:GFH
项目名:G5-AC7003D4-ANCG5-V136-G5-HEAR
主控:AC7003D4
蓝牙名:G5-HEAR
SDK版本:1.3.6
声道配置方式:主左
GIT分支:G5-AC7003D4-ANCG5-V136-G5-HEAR
GIT远端地址:http://192.168.1.245/freebrand/yts/ac700n_earphone.git
本地地址:D:\永泰晟\G5-AC7003D4-ANCG5-V136-G5-HEAR\ac700n_earphone
更新内容:
1.替换EQ文件[G5-助听EQ-0611]
2.修复首次开机和首次切换辅听都是设置最高辅听等级问题

时间:20240603155253
作者:GFH
项目名:G5-AC7003D4-ANCG5-V136-G5-HEAR
主控:AC7003D4
蓝牙名:G5-HEAR
SDK版本:1.3.6
声道配置方式:主左
GIT分支:G5-AC7003D4-ANCG5-V136-G5-HEAR

将 writer.txt 文件内容改为自己的名字大写缩写



5.3 客户需求：开机状态灯改为蓝色呼吸灯模式

需求判断：蓝色呼吸灯属于 UI 范畴，因此先到 user_cfg.c 文件是否使能了配置工具

```
#define USE_CONFIG_BIN_FILE 1

#define USE_CONFIG_STATUS_SETTING 0 //状态设置，包括灯状态和提示音
#define USE_CONFIG_AUDIO_SETTING 0//USE_CONFIG_BIN_FILE //音频设置
#define USE_CONFIG_CHARGE_SETTING USE_CONFIG_BIN_FILE //充电设置
#define USE_CONFIG_KEY_SETTING 0//USE_CONFIG_BIN_FILE //按键消息设置
#define USE_CONFIG_MIC_TYPE_SETTING 0//USE_CONFIG_BIN_FILE //MIC类型设置
#define USE_CONFIG_LOWPOWER_V_SETTING USE_CONFIG_BIN_FILE //低电提示设置
#define USE_CONFIG_AUTO_OFF_SETTING USE_CONFIG_BIN_FILE //自动关机时间设置
#define USE_CONFIG_COMBINE_VOL_SETTING 1 //联合音量读配置

#define USE_CONFIG_DEBUG_INFO 0 //打印配置信息用于调试和测试，打开该宏编译后会多出一些调试信息

#if USE_CONFIG_DEBUG_INFO
#if USE_CONFIG_STATUS_SETTING
static char *status_c[] = {
    "charge_start",
    "charge_full",
}
```

发现灯的状态未被配置，因此到板级配置文件来修改灯的状态；

Terminal Help ← → 002-PWM_Blue_LED [Administrator]

```
... app_config.h M app_config.c earphone.c app_main.c board_ac700n_demo.c 9+ audio_adc.h
ac700n_earphone > apps > earphone > board > br36 > board_ac700n_demo.c > status_config
● 26 #define LOG_TAG_CONST BOARD
27 #define LOG_TAG "[BOARD]"
28 #define LOG_ERROR_ENABLE
29 #define LOG_DEBUG_ENABLE
30 #define LOG_INFO_ENABLE
31 /* #define LOG_DUMP_ENABLE */
32 #define LOG_CLI_ENABLE
33 #include "debug.h"
34
35 void board_power_init(void);
36
37 /*各个状态下默认的闪灯方式和提示音设置，如果USER_CFG中设置了USE_CONFIG_STATUS_SETTING为1，则会从配置文件读取对应
38 STATUS_CONFIG status_config = {
39     //灯状态设置
40     .led = [
41         .charge_start = PWM_LED1_ON,
42         .charge_full = PWM_LED1_ON,
43         .power_on = PWM_LED0_BREATHE, //PWM_LED_ALL_OFF->PWM_LED0_BREATHE,
44         .power_off = PWM_LED_ALL_OFF,
45         .lowpower = PWM_LED_ALL_OFF,
46         .max_vol = PWM_LED_ALL_OFF,
47         .phone_in = PWM_LED_ALL_OFF,
48         .phone_out = PWM_LED_ALL_OFF,
49         .phone_activ = PWM_LED_ALL_OFF,
50         .bt_init_ok = PWM_LED_ALL_OFF,
51         .bt_connect_ok = PWM_LED_ALL_OFF,
52         .bt_disconnect = PWM_LED_ALL_OFF,
53         .tws_connect_ok = PWM_LED_ALL_OFF,
54         .tws_disconnect = PWM_LED_ALL_OFF,
55     ],
56     //提示音设置
57 }
```

```
3 enum pwm_led_mode [
4     PWM_LED_MODE_START,
5
6     PWM_LED_ALL_OFF,           //mode1: 全灭
7     PWM_LED_ALL_ON,            //mode2: 全亮
8
9     PWM_LED0_ON,               //mode3: 蓝亮
10    PWM_LED0_OFF,              //mode4: 蓝灭
11    PWM_LED0_SLOW_FLASH,       //mode5: 蓝慢闪
12    PWM_LED0_FAST_FLASH,       //mode6: 蓝快闪
13    PWM_LED0_DOUBLE_FLASH_5S,   //mode7: 蓝灯5秒连闪两下
14    PWM_LED0_ONE_FLASH_5S,      //mode8: 蓝灯5秒连闪1下
15
16    PWM_LED1_ON,               //mode9: 红亮
17    PWM_LED1_OFF,              //mode10: 红灭
18    PWM_LED1_SLOW_FLASH,        //mode11: 红慢闪
19    PWM_LED1_FAST_FLASH,        //mode12: 红快闪
20    PWM_LED1_DOUBLE_FLASH_5S,   //mode13: 红灯5秒连闪两下
21    PWM_LED1_ONE_FLASH_5S,      //mode14: 红灯5秒闪1下
22
23    PWM_LED0_LED1_FAST_FLASH,   //mode15: 红蓝交替闪 (快闪)
24    PWM_LED0_LED1_SLOW_FLASH,   //mode16: 红蓝交替闪 (慢闪)
25
26    PWM_LED0_BREATHE,          //mode17: 蓝灯呼吸灯模式
27    PWM_LED1_BREATHE,          //mode18: 红灯呼吸灯模式
28    PWM_LED0_LED1_BREATHE,      //mode19: 红蓝交替呼吸灯模式
29
30    PWM_LED_MODE_END,
```

5.3.1 客户反馈灯未亮起

首先查看代码，直接搜索 `pwm_led_mode()`，看在哪被调用了；

查询到时在 `ui_manage.c` UI 管理文件被调用了，可以看出，呼吸灯应该被调用了；

The screenshot shows a code editor interface with several tabs at the top: `ent.c`, `pwm_led.h`, `update.c`, `ui_manage.c` (which is the active tab), and `app_chargestore.c`, `user_cfg.h`. In the search bar at the top left, 'pwm_led_mode' is highlighted. Below the search bar, it says '67 results in 9 files - Open in editor'. The main pane displays the `ui_manage.c` file with the following code snippet:

```
57 void ui_manage_scan(void *priv)
125     sys_ui_var.power_status = STATUS_NORMAL_POWER;
126     break;
127 }
128
129 switch (sys_ui_var.other_status) {
130 case STATUS_POWERON:
131     log_info("[STATUS_POWERON]\n");
132
133     if (p_led->power_on != PWM_LED0_FLASH_THREE) { //判断出能通过pwm_led_mode_set()函数去设置,按理来讲应该没错
134         pwm_led_mode_set(p_led->power_on);           //状态也是设的呼吸灯
135     } else {
136         if (sys_ui_var.ui_flash_cnt) {
137             if (!get_bt_tws_connect_status()) {
138                 sys_ui_var.ui_flash_cnt = 0;
139                 ui_manage_scan(NULL);
140                 return;
141             }
142             if (sys_ui_var.ui_flash_cnt % 2) {
143                 pwm_led_mode_set(PWM_LED0_OFF);
144             } else {
145                 pwm_led_mode_set(PWM_LED0_ON);
146             }
147         }
148     }
149 }
150 break;
```

排查故障：

- 因为客户的上一个需求是关闭拔出耳机开机功能，会不会客户未开机，所以没看到灯闪；
- 我的编译出了问题；

需要验证上述问题，便需要把程序升级到样机上，并开启串口打印；

但是在使用一拖二蓝牙测试盒时遇到了问题

5.3.2 灯没有让其持续做呼吸状态

指导：FH bro

```
Terminal Help ← → 002-PWM_Blue_LED [Administrator]
...
... app_config.h M app_config.c earphone.c app_main.c board_ac700n_demo.c 9+ audio_adc.h
ac700n_earphone > apps > earphone > board > br36 > board_ac700n_demo.c > status_config
● 26 #define LOG_TAG_CONST BOARD
27 #define LOG_TAG "[BOARD]"
28 #define LOG_ERROR_ENABLE
29 #define LOG_DEBUG_ENABLE
30 #define LOG_INFO_ENABLE
31 /* #define LOG_DUMP_ENABLE */
32 #define LOG_CLI_ENABLE
33 #include "debug.h"
34
35 void board_power_init(void);
36
37 /*各个状态下默认的闪灯方式和提示音设置,如果USER_CFG中设置了USE_CONFIG_STATUS_SETTING为1,则会从配置文件读取对应
38 STATUS_CONFIG status_config = {
39     //灯状态设置
40     .led = [
41         .charge_start = PWM_LED1_ON,
42         .charge_full = PWM_LED1_ON,
43         .power_on = PWM_LED0_BREATHE, //PWM_LED_ALL_OFF->PWM_LED0_BREATHE,
44         .power_off = PWM_LED_ALL_OFF,
45         .lowpower = PWM_LED_ALL_OFF,
46         .max_vol = PWM_LED_ALL_OFF,
47         .phone_in = PWM_LED_ALL_OFF,
48         .phone_out = PWM_LED_ALL_OFF,
49         .phone_activ = PWM_LED_ALL_OFF,
50         .bt_init_ok = PWM_LED_ALL_OFF,
51         .bt_connect_ok = PWM_LED_ALL_OFF,
52         .bt_disconnect = PWM_LED_ALL_OFF,
53         .tws_connect_ok = PWM_LED_ALL_OFF,
54         .tws_disconnect = PWM_LED_ALL_OFF,
55     ],
56     //提示音设置
}
```

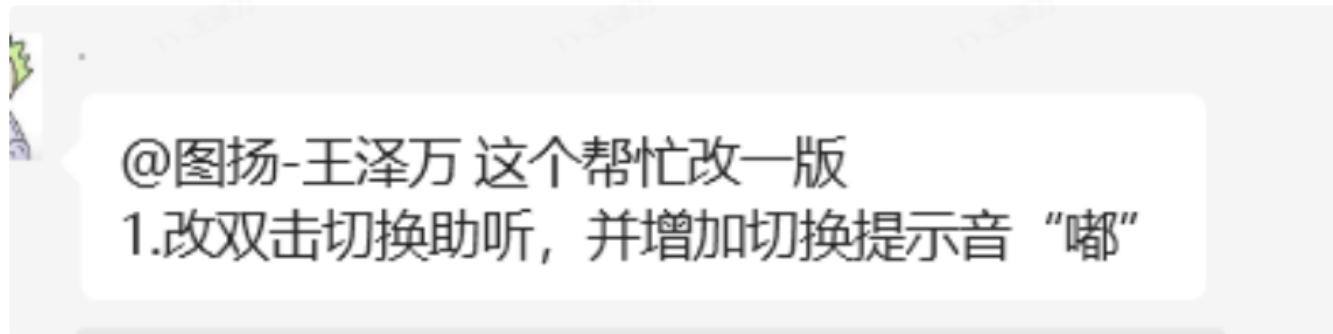
可以看到我只在 power.on 的任务下开启了呼吸灯模式，而这个效果仅在短时间内就被下面的模式给关闭了；

因此需要在除关机模式都把呼吸灯开启；

```
STATUS_CONFIG status_config = {
    //灯状态设置
    .led = [
        .charge_start = PWM_LED1_ON,
        .charge_full = PWM_LED1_ON,
        .power_on = PWM_LED0_BREATHE, //客户需求:开机蓝色呼吸灯, 在关机之前都改为蓝色呼吸灯模式
        .power_off = PWM_LED_ALL_OFF,
        .lowpower = PWM_LED0_BREATHE,
        .max_vol = PWM_LED0_BREATHE,
        .phone_in = PWM_LED0_BREATHE,
        .phone_out = PWM_LED0_BREATHE,
        .phone_activ = PWM_LED0_BREATHE,
        .bt_init_ok = PWM_LED0_BREATHE,
        .bt_connect_ok = PWM_LED0_BREATHE,
        .bt_disconnect = PWM_LED0_BREATHE,
        .tws_connect_ok = PWM_LED0_BREATHE,
        .tws_disconnect = PWM_LED0_BREATHE,
```

5.4 客户需求：将开机默认打开辅听改为双击切换辅听，并增加切换提示音“嘟”；

5.4.1 问题分析：



由之前更新日志可知，助听模式其实为辅听，之前的是默认开机打开辅听功能；

```
49  
50 时间:20231122111540  
51 作者:GFH  
52 项目名:G5-AC7003D4-ANCG5-V136  
53 主控:AC7003D4  
54 蓝牙名:ANCG5  
55 SDK版本:1.3.6  
56 声道配置方式:固定左  
57 GIT分支:G5-AC7003D4-ANCG5-V136  
58 GIT远端地址:http://192.168.1.245/freebrand/yts/ac700n\_earphone.git  
59 本地地址:D:\永泰晟\G5-AC7003D4-ANCG5-V136\ac700n_earphone  
60 更新内容:  
61 1.开机默认开辅听,三击关闭
```

解决思路：

- 从日志来看，之前是有修改过辅听功能；
- 能够看出客户提的需求并不够详细，因此需要进一步确定客户的需求；



@图扬-王泽万 这个帮忙改一版
1.改双击切换助听，并增加切换提示音“嘟”

图扬-王泽万 : YTS[Branch_c015e055][
主左]-主控AC7003D4-{设备名G5-HEA...



嘟.mp3

2.6K



微信电脑版

收到



10.33
@. 原本耳机的功能是开机默认打开辅听功能，现在你是想关闭开机默认打开辅听功能，改为双击切换辅听功能是吧？





@图扬-王泽万 是改成双击切换助听等级，现在是三击切换，加入切换嘟提示音

图扬-王泽万：@. 原本耳机的功能是开机默认打开辅听功能，现在你是想关闭开机默认打开辅听功能，...



嘟(2).mp3

2.6K



微信电脑版

反正就是将之前三击实现的功能改为双击来实现，并且加上提示音？



@.



是的

在搞清楚客户需求之后，便可以开始分析；

5.4.2 解决问题

指导：GFH

客户需求：将原本由三击实现的功能改为由双击实现，并且添加提示音；

1. 分别找到三击和双击的功能所在位置；

按键消息设置，将原本的三击功能关闭，即 NULL 空就好，在将双击的功能改为辅听；

```

#define __this (&status_config)

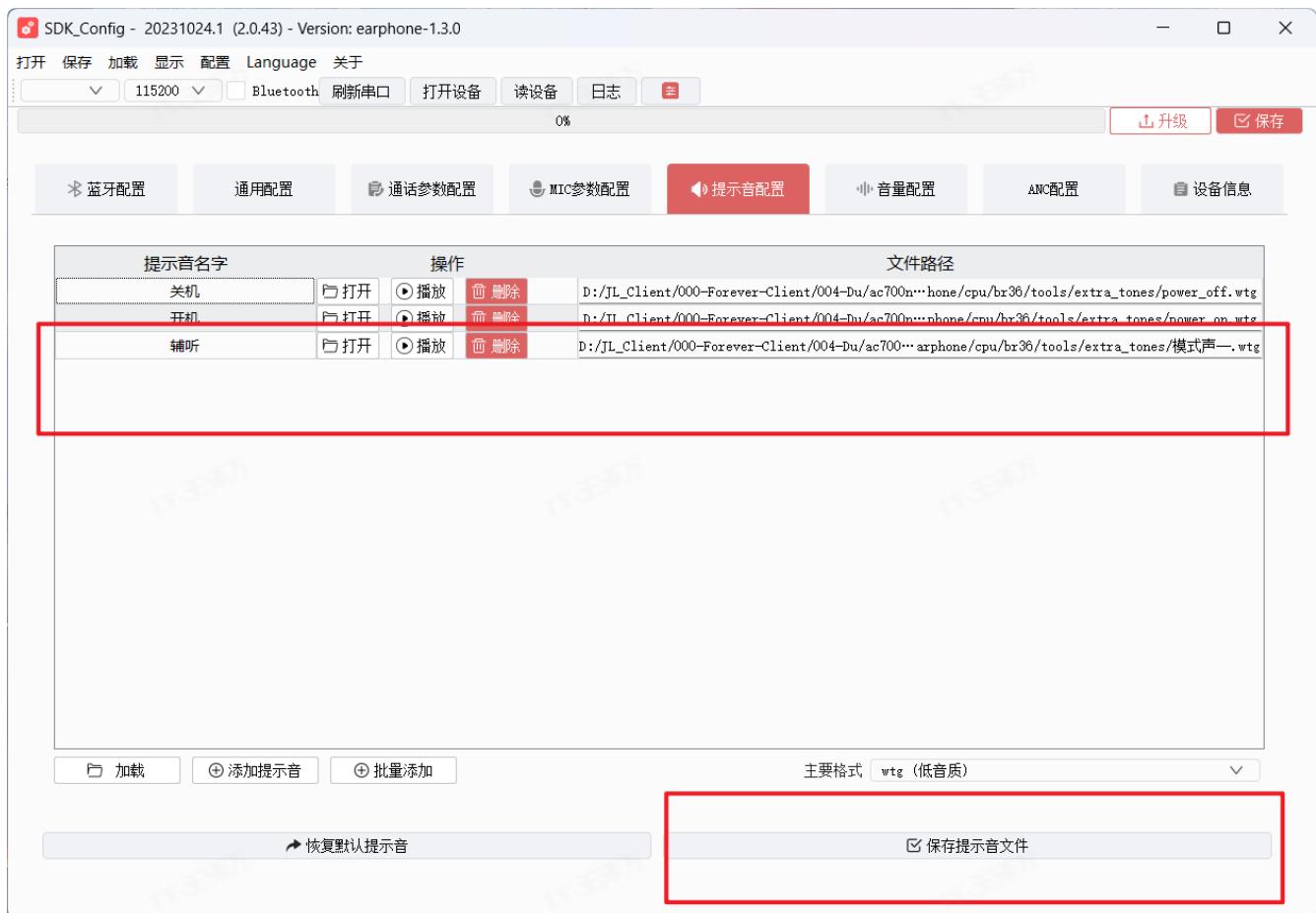
/********************* KEY MSG****************************/
/*各个按键的消息设置，如果USER_CFG中设置了USE_CONFIG_KEY_SETTING为1，则会从配置文件读取对应的配置来填充改结构体*/
u8 key_table[KEY_NUM_MAX][KEY_EVENT_MAX] = {
    // SHORT      LONG      HOLD      UP
#if TCFG_APP_LINEIN_EN
    {KEY_ANC_SWITCH,   KEY_POWEROFF,  KEY_POWEROFF_HOLD,  KEY_NULL,
#else
    {KEY_NULL,        KEY_LONG_CLICK, KEY_POWEROFF_HOLD,  KEY_LONG_UP,
     //20240904客户需求:将三击关闭改为NULL, 双击改为开启辅听
#endif
    {KEY_NULL,        KEY_MUSIC_NEXT, KEY_VOL_UP,       KEY_NULL,
     {KEY_MUSIC_PREV, KEY_VOL_DOWN,  KEY_VOL_DOWN,      KEY_NULL,
     // 上滑          下滑          左滑          右滑
     {KEY_MUSIC_NEXT, KEY_MUSIC_PREV, KEY_NULL,         KEY_NULL,
     {KEY_NULL,        KEY_NULL,       KEY_NULL,         KEY_NULL},           //触摸按键滑动时的消息
};

    DOUBLE          TRIPLE
    KEY_MODE_SWITCH,  KEY_LOW_LANTECY, //KEY_0
    KEY_HEARING_AID_TOGGLE, KEY_NULL}, //KEY_0
    KEY_OPEN_SIRI,    KEY_NULL},     //KEY_1
    KEY_HID_CONTROL, KEY_NULL},     //KEY_2
    KEY_NULL},         KEY_NULL},     //触摸按键滑动时的消息
};

```

2. 添加双击切换提示音

打开配置工具，将提示音音频放进去，点击保存；



再保存 bin 文件；

SDK_Config - 20231024.1 (2.0.43) - Version: earphone-1.3.0

打开 保存 加载 显示 配置 Language 关于
保存bin文件
保存默认值文件

Bluetooth 刷新串口 打开设备 读设备 日志 全屏

0%

上升

* 蓝牙配置 通用配置 通话参数配置 MIC参数配置 提示音配置 音量配置 ANC配置

提示音名字	操作			文件路径
关机	<input type="checkbox"/> 打开	<input checked="" type="radio"/> 播放	<input type="checkbox"/> 删除	D:/JL_Client/000-Forever-Client/004-Du/ac700n...hone/cpu/br36/tools/extra_tones/po...
开机	<input type="checkbox"/> 打开	<input checked="" type="radio"/> 播放	<input type="checkbox"/> 删除	D:/JL_Client/000-Forever-Client/004-Du/ac700n...hone/cpu/br36/tools/extra_tones/po...
辅听	<input type="checkbox"/> 打开	<input checked="" type="radio"/> 播放	<input type="checkbox"/> 删除	D:/JL_Client/000-Forever-Client/004-Du/ac700n...arphone/cpu/br36/tools/extra_tones/...

在回到程序里继续配置提示音；

模仿上面的提示音路径配置：（以后注意，不要有中文路径）

The screenshot shows a terminal window titled "ac700n_earphone [Administrator]" with the following file structure and content:

```
user_cfg.c 6 | C key_event_deal.h | C app_config.c | C app_config.h | C board_config.h | C bc
apps > earphone > include > C tone_player.h > TONE_MODE
```

```
6
7 #define TONE_NUM_0
8 #define TONE_NUM_1
9 #define TONE_NUM_2
10 #define TONE_NUM_3
11 #define TONE_NUM_4
12 #define TONE_NUM_5
13 #define TONE_NUM_6
14 #define TONE_NUM_7
15 #define TONE_NUM_8
16 #define TONE_NUM_9
17 #define TONE_BT_MODE
18 #define TONE_BT_CONN
19 #define TONE_BT_DISCONNECT
20 #define TONE_TWS_CONN
21 #define TONE_TWS_DISCONNECT
22 #define TONE_LOW_POWER
23 #define TONE_POWER_OFF
24 #define TONE_POWER_ON
25 #define TONE_RING
26 #define TONE_MAX_VOL
27 #define TONE_ANC_OFF
28 #define TONE_ANC_ON
29 #define TONE_TRANSPARENCY
30 #define TONE_EAR_CHECK
31 #define TONE_MUSIC_MODE
32 #define TONE_PC_MODE
33 #define TONE_HEARAID_ON
34 #define TONE_HEARAID_OFF
35 #define TONE_LINEIN
36 #define TONE_MODE
37 #define TONE_MIN_VOL
38 #define TONE_HEARING_AID
39
40 #define STNF_WTONE_NORAMI
```

The line "#define TONE_HEARING_AID" is highlighted with a red box.

最后到按键处理函数里的辅听功能处添加提示音（注意要看宏是否有正确的值）

```
break;

#if TCFG_AUDIO_HEARING_AID_ENABLE
/*辅听功能测试demo开关*/
case KEY_HEARING_AID_TOGGLE:
    extern void hearing_volume_set(u8 volume);
    printf("KEY_HEARING_AID_TOGGLE\n");
    tone_play(TONE_HEARING_AID,1);
    if(user_hearing_lev == 0){
        hearing_volume_set(16);
        user_hearing_lev = 1;
    }else if(user_hearing_lev == 1){
        hearing_volume_set(13);
        user_hearing_lev = 2;
    }else{
        hearing_volume_set(10);
        user_hearing_lev = 0;
    }
    // extern void audio_hearing_aid_demo(void);
    // audio_hearing_aid_demo();
    break;
#endif/*TCFG_AUDIO_HEARING_AID_ENABLE*/
```

5.4.3 客户反馈问题

5.4.3.1 每次双击都是重启耳机，不是切换助听等级



昨天 15:37
@图扬-王泽万 现在每次双击都是重启耳机，不是切换助听等级

图扬-王泽万： YTS[Branch_c015e055][
主左]-主控AC7003D4-{设备名G5-HEA...



昨天 16:13

发现是在配置提示音时，使用了中文路径；

并且播放提示音的函数调用错了；

修改：

```
18 #define TONE_BT_CONN SDFILE_RES_ROOT_PATH"tone/bt_conn.*"
19 #define TONE_BT_DISCONNECT SDFILE_RES_ROOT_PATH"tone/bt_dconn.*"
20 #define TONE_TWS_CONN SDFILE_RES_ROOT_PATH"tone/tws_conn.*"
21 #define TONE_TWS_DISCONNECT SDFILE_RES_ROOT_PATH"tone/tws_dconn.*"
22 #define TONE_LOW_POWER SDFILE_RES_ROOT_PATH"tone/low_power.*"
23 #define TONE_POWER_OFF SDFILE_RES_ROOT_PATH"tone/power_off.*"
24 #define TONE_POWER_ON SDFILE_RES_ROOT_PATH"tone/power_on.*"
25 #define TONE_RING SDFILE_RES_ROOT_PATH"tone/ring.*"
26 #define TONE_MAX_VOL SDFILE_RES_ROOT_PATH"tone/vol_max.*"
27 #define TONE_ANC_OFF SDFILE_RES_ROOT_PATH"tone/anc_off.*"
28 #define TONE_ANC_ON SDFILE_RES_ROOT_PATH"tone/anc_on.*"
29 #define TONE_TRANSPARENCY SDFILE_RES_ROOT_PATH"tone/anc_trans.*"
30 #define TONE_EAR_CHECK SDFILE_RES_ROOT_PATH"tone/ear_check.*"
31 #define TONE_MUSIC_MODE SDFILE_RES_ROOT_PATH"tone/music.*"
32 #define TONE_PC_MODE SDFILE_RES_ROOT_PATH"tone/pc.*"
33 #define TONE_HEARAID_ON SDFILE_RES_ROOT_PATH"tone/hearing_on.*"
34 #define TONE_HEARAID_OFF SDFILE_RES_ROOT_PATH"tone/hearing_off.*"
35 #define TONE_LINEIN SDFILE_RES_ROOT_PATH"tone/linein.*"
36 #define TONE_MODE SDFILE_RES_ROOT_PATH"tone/CSR_mode.*"
37 #define TONE_MIN_VOL SDFILE_RES_ROOT_PATH"tone/CSR_minvol.*"
38 #define TONE_HEARING_AID SDFILE_RES_ROOT_PATH"tone/DuHigh.*"  
39  
40 #define SINE_WTONE_NORMAL 0
41 #define SINE_WTONE_TWS_CONNECT 1
42 #define SINE_WTONE_TWS_DISCONNECT 2
43 #define SINE_WTONE_LOW_POWER 4
```

改为英文

一定要使用有实际值的宏；

```
#endif /* TCFG_USER_TWS_ENABLE */
break;

#if TCFG_AUDIO_HEARING_AID_ENABLE
/*辅听功能测试demo开关*/
case KEY_HEARING_AID_TOGGLE:
    extern void hearing_volume_set(u8 volume);
    printf("KEY HEARING AID TOGGLE\n");
    tone_play(TONE_HEARING_AID,1); //改成了tone_play()
    if(user_hearing_lev == 0){
        hearing_volume_set(16);
        user_hearing_lev = 1;
    }else if(user_hearing_lev == 1){
        hearing_volume_set(13);
        user_hearing_lev = 2;      改成tone_play()
    }else{
        hearing_volume_set(10);
        user_hearing_lev = 0;
    }
    // extern void audio_hearing_aid_demo(void);
    // audio_hearing_aid_demo();
    break;
#endif/*TCFG_AUDIO_HEARING_AID_ENABLE*/
```

5.4.3.2 助听只有一档，且双击后需要等很久才有提示音

点击进入 tone_play() 函数，发写说明里写着辅听和提示音有着较强的相关性；

```

}
static void tone_stop(u8 force);
int tone_play(const char *name, u8 preemption)
{
#if (TCFG_AUDIO_HEARING_AID_ENABLE && TCFG_AUDIO_DHA_FITTING_ENABLE)
/*辅听验配过程不允许播放提示音*/
extern u8 get_hearing_aid_fitting_state(void);
if (get_hearing_aid_fitting_state()) {
    printf("hearing aid fitting : %d\n !!!", get_hearing_aid_fitting_state());
    return 0;
}
#endif /*TCFG_AUDIO_HEARING_AID_ENABLE && TCFG_AUDIO_DHA_FITTING_ENABLE*/

g_printf("tone_play:%s,preemption:%d", IS_DEFAULT_SINE(name) ? "sine" : name, preemption)

if (strcmp(os_current_task(), "app_core") != 0) {
    g_printf("Tone play not in right task.");

    int *ptr = malloc(2 * sizeof(int));
    ptr[0] = (int)name;
    ptr[1] = (int)preemption;
    OS_SEM *sem = malloc(sizeof(OS_SEM) + 4);

```

于是直接在整个程序里直接搜索“辅听”二字；发现在板级文件里“辅听与提示音互斥”

于是关掉该功能

```

2 /*
3  * MIC_TO_DAC低延时通道功能
4  */
5
6 #define TCFG_AD2DA_LOW_LATENCY_ENABLE          (0)
7 #define TCFG_AD2DA_LOW_LATENCY_SAMPLE_RATE      (48000)           // 目前混叠仅支持固定
8 #define TCFG_AD2DA_LOW_LATENCY_MIC_CHANNEL     (LADC_CH_MIC_L) // 目前仅支持单声道
9
0 //*****
1 //                         Digital Hearing Aid(DHA)耳机配置
2 //*****
3 /*辅听功能使能*/
4 #define TCFG_AUDIO_HEARING_AID_ENABLE          1///*DISABLE_THIS_MOUDLE
5 /*听力验配功能*/
6 #define TCFG_AUDIO_DHA_FITTING_ENABLE          DISABLE
7 /*辅听功能互斥配置*/
8 #define TCFG_AUDIO_DHA_AND_MUSIC_MUTEX        ENABLE //辅听功能和音乐播放互斥(默认互斥, 资源有限)
9 #define TCFG_AUDIO_DHA_AND_CALL_MUTEX         ENABLE //辅听功能和通话互斥(默认互斥, 资源有限)
0 #define TCFG_AUDIO_DHA_AND_TONE_MUTEX         DISABLE //辅听功能和提示音互斥, 20240904客户双击辅听修改enable->disable
1 /*辅听功能MIC的采样率*/
2 #define TCFG_AUDIO_DHA_MIC_SAMPLE_RATE        32000
3 #if TCFG_AUDIO_HEARING_AID_ENABLE
4 #define TCFG_AUDIO_MUSIC_SAMPLE_RATE          TCFG_AD2DA_LOW_LATENCY_SAMPLE_RATE
5 #endif/*TCFG_AUDIO_HEARING_AID_ENABLE*/
6

```

5.4.3.3 客户认为提示音声量过小

去网上将客户发的音量文件给增益，然后直接将文件导入配置工具；



音量增强

X | ⚡ 🎧 🔍



全部 视频 图片 新闻 图书 网页 财经

工具

增强器 windows

软件

音频



Google

https://chrome.google.com › volume-booster-increase-s ...

音量增强器- 增加音效

2024年2月20日 — 带有音量控制的易于使用的声音增强器。将音量放大高达600%。如果歌曲或视频太安静，音量放大器可以帮助您，即使是最大音量。使用我们的音量助推器，您 ...



Chrome Web Store

https://chromewebstore.google.com › detail › jghecgabf... ...

Volume Master - 音量控制器- Chrome 应用商店

音量提升高达600%. 最简单, 最可靠的音量增强器 特征 ★ 音量提升高达600% ★ 控制任何标签的音量 ★ 细粒度控制: 0%-600% ★ 一键切换到任何播放音频的标签 全屏 ★ 当您使用 ...



Google

https://chrome.google.com › sound-booster-by-audiomax ...

AudioMax声音增强器- Chrome 应用商店

我们独特的音频增强技术可以让您前所未有地控制、定制和放大声音。此音量增强扩展可以将音量提升至600%，使其成为提升YouTube Music、视频、电影或任何需要音量增强的网站 ...



Google Play

https://play.google.com › store › apps › details ...



完成客户要求；

昨天 21:55

YTS[Branch_c015e05
5][主左]-主控AC70...



16.6M

微信电脑版

可以了确认这个

谢谢🙏

昨天 22:17

OK



早点下班



5.4.5 提交并推送文件到 GitLab

注意：在提交时不要勾选修改上次提交，不然可能会产生冲突；

D:\VL_Client\vvv-Forever-Client\vvv_Du_Adv\action_earphone - 提交 -

提交至: Branch_c015e055 新建分支

日志信息(M):

修改上次提交(L)

设置作者日期(D)

设置作者(T)

变更列表(双击文件查看差异):

选中: 全部(A) 无(N) 未版本控制 已版本控制 已添加 已删除

路径	扩展名	状态	添加行数	删除行数
修改的文件	.layout	已修改	1	1

成功提交!

自由品牌 > 永泰晟 > Ac700n Earphone > Commits > 428f0647

Commit 428f0647 authored just now by 王泽万

时间:20240905152738

作者:WZW

项目名:G5-AC7003D4-ANCG5-V136-G5-HEAR

主控:AC7003D4

蓝牙名:"G5-HEAR"

SDK版本:"1.3.6"

声道配置方式:"主左"

GIT分支:"Branch_c015e055"

GIT远端地址:"http://192.168.1.245/freebrand/yts/ac700n_earphone.git"

本地地址:"D:\JL_Client\000-Forever-Client\007_Du_Add\ac700n_earphone"

更新内容:

1. 将原本由三击来切换辅听开关和等级改由双击实现，并且添加双击切换提示音；

2. 增加双击提示音音量

⌚ parent 7c10cf58 Branch_c015e055

修改内容:

apps/earphone/board/br36/board_ac700n_demo.c

View file @428f0647

```
@@ -83,7 +83,7 @@ u8 key_table[KEY_NUM_MAX][KEY_EVENT_MAX] = {  
83     #if TCFG_APP_LINEIN_EN  
84         {KEY_ANC_SWITCH, KEY_POWEROFF, KEY_POWEROFF_HOLD, KEY_NULL, KEY_MODE_SWITCH, KEY_LOW_LANTECY},  
85         //KEY_0  
85     #else  
86         - {KEY_NULL, KEY_LONG_CLICK, KEY_POWEROFF_HOLD, KEY_LONG_UP, KEY_NULL, KEY_HEARING_AID_TOGGLE},  
86         //KEY_0  
86     + {KEY_NULL, KEY_LONG_CLICK, KEY_POWEROFF_HOLD, KEY_LONG_UP, KEY_HEARING_AID_TOGGLE, KEY_NULL},  
86         //KEY_0 20240904客户双击辅听修改double:KEY_HEARING_AID_TOGGLE  
87     #endif  
88     {KEY_MUSIC_NEXT, KEY_VOL_UP, KEY_VOL_UP, KEY_NULL, KEY_OPEN_SIRI, KEY_NULL}, //KEY_1  
89     {KEY_MUSIC_PREV, KEY_VOL_DOWN, KEY_VOL_DOWN, KEY_NULL, KEY_HID_CONTROL, KEY_NULL}, //KEY_2  
...  
...  
按键修改
```

apps/earphone/board/br36/board_ac700n_demo_cfg.h

View file @428f0647

...	...	@@ -277,7 +277,7 @@	
277	277	/*辅听功能互斥配置*/	
278	278	#define TCFG_AUDIO_DHA_AND_MUSIC_MUTEX	ENABLE //辅听功能和音乐播放互斥(默认互斥, 资源有限)
279	279	#define TCFG_AUDIO_DHA_AND_CALL_MUTEX	ENABLE //辅听功能和通话互斥(默认互斥, 资源有限)
280	- #define TCFG_AUDIO_DHA_AND_TONE_MUTEX	ENABLE //辅听功能和提示音互斥	
	+ #define TCFG_AUDIO_DHA_AND_TONE_MUTEX	DISABLE //辅听功能和提示音互斥, 20240904客户双击辅听修改enable->disable	
281	281	/>辅听功能HDC的采样率/	
282	282	#define TCFG_AUDIO_DHA_MIC_SAMPLE_RATE	32000
283	283	#if TCFG_AUDIO_HEARING_AID_ENABLE	
...	...		

辅听与提示音互斥关闭

apps/earphone/include/tone_player.h

...	...	@@ -35,6 +35,7 @@	
35	35	#define TONE_LINEIN	SDFILE_RES_ROOT_PATH"tone/linein.*"
36	36	#define TONE_MODE	SDFILE_RES_ROOT_PATH"tone/CSR_mode.*"
37	37	#define TONE_MTN_VOI	SDFILE_RES_ROOT_PATH"tone/CSR_minvol.*"
38	+ #define TONE_HEARING_AID	SDFILE_RES_ROOT_PATH"tone/DuHigh.*"	
39	39		
40	#define SINE_WTONE_NORAML	0	
41	#define SINE_WTONE_TWS_CONNECT	1	
...	...		

增加双击提示音

apps/earphone/key_eventDeal.c

...	...	@@ -709,6 +709,7 @@ int app_earphone_key_event	*event)
709	709	case KEY_HEARING_AID_TOGGLE:	
710	710	extern void hearing_volume_set(u8 volume);	
711	711	printf("KEY_HEARING_AID_TOGGLE\n");	
712	+ tone_play(TONE_HEARING_AID,1);		
713	713	if(user_hearing_lev == 0){	
714	714	hearing_volume_set(16);	
715		user_hearing_lev = 1;	
...	...		

切换时添加提示音

6. 阅读产品 SDK 文档与熟悉代码

6.1 board_config.h 板级配置文件

未被注释的即为该样机的板级配置，选中找到对应头文件位置；

可以得知该样机的芯片型号为 AC700N；

```
/*
#define CONFIG_BOARD_AC700N_DEMO
// #define CONFIG_BOARD_AC7006F_EARPHONE
// #define CONFIG_BOARD_AC700N_SD_PC_DEMO
// #define CONFIG_BOARD_AC7003F_DEMO
// #define CONFIG_BOARD_AC700N_ANC
// #define CONFIG_BOARD_AC700N_DMS
// #define CONFIG_BOARD_AC700N_TEST //ANC+ENC双mic+DNS+AEC+BLE+TWS+叠加提示音播放
// #define CONFIG_BOARD_AC700N_HEARING_AID //辅听模式板级
// #define CONFIG_BOARD_AC700N_IIS_LINEIN //linein anc板级

#include "media/audio_def.h"
#include "board_ac700n_demo_cfg.h"
#include "board_ac7006f_earphone_cfg.h"
#include "board_ac700n_sd_pc_demo_cfg.h"
#include "board_ac7003f_demo_cfg.h"
#include "board_ac700n_anc_cfg.h"
#include "board_ac700n_dms_cfg.h"
#include "board_ac700n_test_cfg.h"
#include "board_ac700n_hearing_aid_cfg.h"
#include "board_ac700n_iis_linein_cfg.h"

#define DUT_AUDIO_DAC_LDO_VOLT           DACVDD_LDO_1_35V

#ifndef CONFIG_NEW_CFG_TOOL_ENABLE
#define CONFIG_ENTRY_ADDRESS           0x1e00100
#endif

#endif
```

board_ac700n_demo_cfg.h 为各功能配置文件；

```

C board_config.h
C board_ac700n_demo_cfg.h M X

apps > earphone > board > br36 > C board_ac700n_demo_cfg.h > ...
1 #ifndef CONFIG_BOARD_AC700N_DEMO_CFG_H
2 #define CONFIG_BOARD_AC700N_DEMO_CFG_H
3
4 #include "board_ac700n_demo_global_build_cfg.h"
5
6 #ifdef CONFIG_BOARD_AC700N_DEMO
7
8 #define CONFIG_SDFILE_ENABLE
9
10 //*****配置开始***** //////////////////////////////////////////////////////////////////
11 //////////////////////////////////////////////////////////////////
12 //*****配置结束***** //////////////////////////////////////////////////////////////////
13 #define ENABLE_THIS_MODULE 1
14 #define DISABLE_THIS_MODULE 0
15
16 #define ENABLE 1
17 #define DISABLE 0
18
19 #define NO_CONFIG_PORT (-1)
20 //*****USER自定义功能配置***** //////////////////////////////////////////////////////////////////
21 //////////////////////////////////////////////////////////////////
22 //*****0//0: 纯辅听耳机无蓝牙功能***** //////////////////////////////////////////////////////////////////
23 #define USER_HEARING_USER_BT 0
24
25 //*****UART配置***** //////////////////////////////////////////////////////////////////
26 //////////////////////////////////////////////////////////////////
27 //////////////////////////////////////////////////////////////////
28 #define TCFG_UART0_ENABLE DISABLE_THIS_MODULE
29 #define TCFG_UART0_RX_PORT NO_CONFIG_PORT
30 #define TCFG_UART0_TX_PORT IO_PORT_D0
31 #define TCFG_UART0_BAUDRATE 1000000
32 #ifdef CONFIG_DEBUG_ENABLE
33 #undef TCFG_UART0_ENABLE
34 #define TCFG_UART0_ENABLE ENABLE_THIS_MODULE
35#endif

```

要使用在线配置工具，得先查看是否需要在程序代码上使能

```

//*****新配置工具 && 调音工具***** //
//*****是否支持在线配置工具***** //
#define TCFG_CFG_TOOL_ENABLE /*DISABLE*/ //是否支持在线配置工具
#define TCFG_EFFECT_TOOL_ENABLE /*DISABLE*/ //是否支持在线音效调试,使能该项还需使能EQ总使能TCFG_EQ_ENABLE,
#define TCFG_NULL_COMM 0 //不支持通信
#define TCFG_UART_COMM 1 //串口通信
#define TCFG_USB_COMM 2 //USB通信
#define TCFG_SPP_COMM 3 //蓝牙SPP通信
#if (TCFG_CFG_TOOL_ENABLE || TCFG_EFFECT_TOOL_ENABLE)
#define TCFG_COMM_TYPE TCFG_SPP_COMM //通信方式选择
#endif

```

6.2 app_config.h 应用模式配置文件

串口打印开关，一些APP模式在这个文件

```

terminal Help
apps > earphone > include > C app_config.h > ...
1  #ifndef APP_CONFIG_H
2  #define APP_CONFIG_H
3
4  /*
5   * 系统打印总开关
6   */
7
8  #define LIB_DEBUG      0
9  #define CONFIG_DEBUG_LIB(x)      (x & LIB_DEBUG)
10
11 // #define CONFIG_DEBUG_ENABLE
12
13 #ifndef CONFIG_DEBUG_ENABLE
14 // #define CONFIG_DEBUG_LITE_ENABLE //轻量级打印开关, 默认关闭
15 #endif
16
17
18 //*****AI配置*****
19 //
20 //*****AI配置*****

```



```

//*****电量显示方式*****
// 对耳电量显示方式
//*****电量显示方式*****

```

```

#if BT_SUPPORT_DISPLAY_BAT
#define CONFIG_DISPLAY_TWS_BAT_LOWER          1 //对耳手机端电量显示, 显示低电量耳机的电量
#define CONFIG_DISPLAY_TWS_BAT_HIGHER         2 //对耳手机端电量显示, 显示高电量耳机的电量
#define CONFIG_DISPLAY_TWS_BAT_LEFT           3 //对耳手机端电量显示, 显示左耳的电量
#define CONFIG_DISPLAY_TWS_BAT_RIGHT          4 //对耳手机端电量显示, 显示右耳的电量

#define CONFIG_DISPLAY_TWS_BAT_TYPE           CONFIG_DISPLAY_TWS_BAT_LOWER
#endif //BT_SUPPORT_DISPLAY_BAT

#define CONFIG_DISPLAY_DETAIL_BAT            0 //BLE广播显示具体的电量
#define CONFIG_NO_DISPLAY_BUTTON_ICON        1 //BLE广播不显示按键界面, 智能充电仓置1

#endif //TCFG_USER_TWS_ENABLE

#define CONFIG_BT_RX_BUFF_SIZE  (20 * 1024)

#ifndef CONFIG_APP_BT_ENABLE

```

6.3 user_cfg.h/c 用户配置文件

客户可以通过配置工具对这个文件里的参数进行修改；

客户一般改的最多的是样机的状态配置；

The screenshot shows a configuration interface with tabs at the top: 蓝牙配置, 通用配置 (selected), 通话参数配置, MIC参数配置, 提示音配置, and 音量配置. Below the tabs are sub-tabs: 状态配置 (selected), 按键消息配置, 充电配置, and 内置触摸按键. A green box highlights the '状态配置' tab. A red box highlights the 'ON' switch for the '状态同步使能开关'. A green box highlights the title '不同状态下 LED 灯显示效果和提示音设置'. The main content is a table:

状态	LED 显示效果	提示音
开始充电	红灯亮	无提示音
充电完成	蓝灯亮	无提示音
开机	蓝灯亮	开机
关机	红灯闪烁三下	关机
低电	红灯五秒内闪烁一下	低电量
最大音量	无LED灯显示效果	最大音量
来电	无LED灯显示效果	来电
去电	无LED灯显示效果	无提示音
通话中	无LED灯显示效果	无提示音
蓝牙初始化完成	红灯和蓝灯交替闪烁(慢闪)	蓝牙模式
蓝牙连接成功	蓝灯和红灯全灭	连接成功
蓝牙断开连接	红灯和蓝灯交替闪烁(快闪)	断开连接
对耳连接成功	红灯和蓝灯交替闪烁(快闪)	对耳连接成功
对耳断开连接	红灯和蓝灯交替闪烁(慢闪)	对耳断开连接

状态配置恢复默认值

1.7 是否使用配置文件的参数配置

默认 sdk 是从配置文件读取配置来设置灯状态和提示音，如果不想从配置文件读取配置，只需修改

```
#define USE_CONFIG_BIN_FILE          0
#define USE_CONFIG_STATUS_SETTING      1
#define USE_CONFIG_AUDIO_SETTING      USE_CONFIG_BIN_FILE
#define USE_CONFIG_CHARGE_SETTING     USE_CONFIG_BIN_FILE
#define USE_CONFIG_KEY_SETTING        USE_CONFIG_BIN_FILE
#define USE_CONFIG_PWMLED_SETTING    USE_CONFIG_BIN_FILE
#define USE_CONFIG_MIC_TYPE_SETTING   USE_CONFIG_BIN_FILE
```

成

```
#define USE_CONFIG_BIN_FILE          0
#define USE_CONFIG_STATUS_SETTING      0
#define USE_CONFIG_AUDIO_SETTING      USE_CONFIG_BIN_FILE
#define USE_CONFIG_CHARGE_SETTING     USE_CONFIG_BIN_FILE
#define USE_CONFIG_KEY_SETTING        USE_CONFIG_BIN_FILE
#define USE_CONFIG_PWMLED_SETTING    USE_CONFIG_BIN_FILE
#define USE_CONFIG_MIC_TYPE_SETTING   USE_CONFIG_BIN_FILE
```

即可，然后默认配置在每个 board.c 里面设置

```
STATUS_CONFIG status_config = {
    //灯状态设置
    led = {
        .charge_start = PWM_LED1_ON,
        .charge_full = PWM_LED0_ON,
        .power_on = PWM_LED0_ON,
        .power_off = PWM_LED1_FLASH_THREE,
        .lowpower = PWM_LED1_SLOW_FLASH,
        .max_vol = PWM_LED_NULL,
        .phone_in = PWM_LED_NULL,
        .phone_out = PWM_LED_NULL,
        .phone_activ = PWM_LED_NULL,
        .bt_init_ok = PWM_LED0_LED1_FAST_FLASH,
        .bt_connect_ok = PWM_LED0_SLOW_FLASH,
        .bt_dtsconnect = PWM_LED0_LED1_FAST_FLASH,
    },
    //提示音设置
    tone = {
        .charge_start = IDEX_TONE_NONE,
        .charge_full = IDEX_TONE_NONE,
        .power_on = IDEX_TONE_POWER_ON,
        .power_off = IDEX_TONE_POWER_OFF,
        .lowpower = IDEX_TONE_LOW_POWER,
        .max_vol = IDEX_TONE_MAX_VOL,
        .phone_in = IDEX_TONE_NONE,
        .phone_out = IDEX_TONE_NONE,
        .phone_activ = IDEX_TONE_NONE,
        .bt_init_ok = IDEX_TONE_BT_MODE,
        .bt_connect_ok = IDEX_TONE_CONN,
        .bt_dtsconnect = IDEX_TONE_DISCONNECT,
    }
};
```

注意：如果想生成的 fw 文件能单独配置某些选项，要把对应的宏打开，如果想全部打开则设置

```
#define USE_CONFIG_BIN_FILE 1  
即可
```

Vs Code 页面：

```

C board_config.h C board_ac700n_demo_cfg.h M ● C uartPcmSender.c C gSensor_manage.c 1 C user_cfg.h ×
include_lib > system > C user_cfg.h > _AUDIO_CONFIG > sw
1 #ifndef __USER_CFG_H__
2 #define __USER_CFG_H__
3
4 #include "typedef.h"
5 #include "app_config.h"
6
7 #define LOCAL_NAME_LEN 32 /*BD_NAME_LEN_MAX*/
8
9 //bt bin结构
10 typedef struct __BT_CONFIG {
11     u8 edr_name[LOCAL_NAME_LEN];           //经典蓝牙名
12     u8 mac_addr[6];                      //蓝牙MAC地址
13     u8 rf_power;                         //发射功率
14     u8 dac_analog_gain;                 //通话DAC模拟增益
15     u8 mic_analog_gain;                  //通话MIC增益
16     u16 tws_device_indicate;            /*设置对箱搜索标识, inquiry时候用, 搜索到相应的标识才允许连接*/
17     u8 tws_local_addr[6];
18 } __GNU_PACKED__ BT_CONFIG;
19
20 //audio bin结构
21 typedef struct __AUDIO_CONFIG {
22     u8 sw;                                //最大系统音量
23     u8 max_sys_vol;                      //开机默认音量
24     u8 default_vol;                     //提示音音量
25     u8 tone_vol;                         //提示音音量
26 } __GNU_PACKED__ AUDIO_CONFIG;

```

user_cfg.h

```

C board_config.h C board_ac700n_demo_cfg.h M ● C uartPcmSender.c C gSensor_manage.c 1 C user_cfg.c 6 ×
apps > earphone > C user_cfg.c > status_c
168 u8 get_tone_vol(void)
169 {
170     if (audio_cfg.tone_vol > get_max_sys_vol()) {
171         return (get_max_sys_vol());
172     }
173
174     return (audio_cfg.tone_vol);
175 }
176
177
178 }
179
180 #define USE_CONFIG_BIN_FILE          1
181
182 #define USE_CONFIG_STATUS_SETTING    0
183 #define USE_CONFIG_MIC_TYPE_SETTING  0 //USE_CONFIG_BIN_FILE
184 #define USE_CONFIG_BIN_FILE          //MIC类型设置          //状态设置, 包括灯状态和提示音
185 //充电设置
186 //按键消息设置
187 //MIC类型设置
188 //低电提示设置
189 //自动关机时间设置
190 //联合音量读配置
191
192 #define USE_CONFIG_DEBUG_INFO        0
193 //打印配置信息用于调试和测试, 打开该宏编译后会多出4K
194
195 #if USE_CONFIG_DEBUG_INFO
196 #if USE_CONFIG_STATUS_SETTING
197     static char *status_c[] = {

```

user_cfg.c

6.4 配置工具的使用

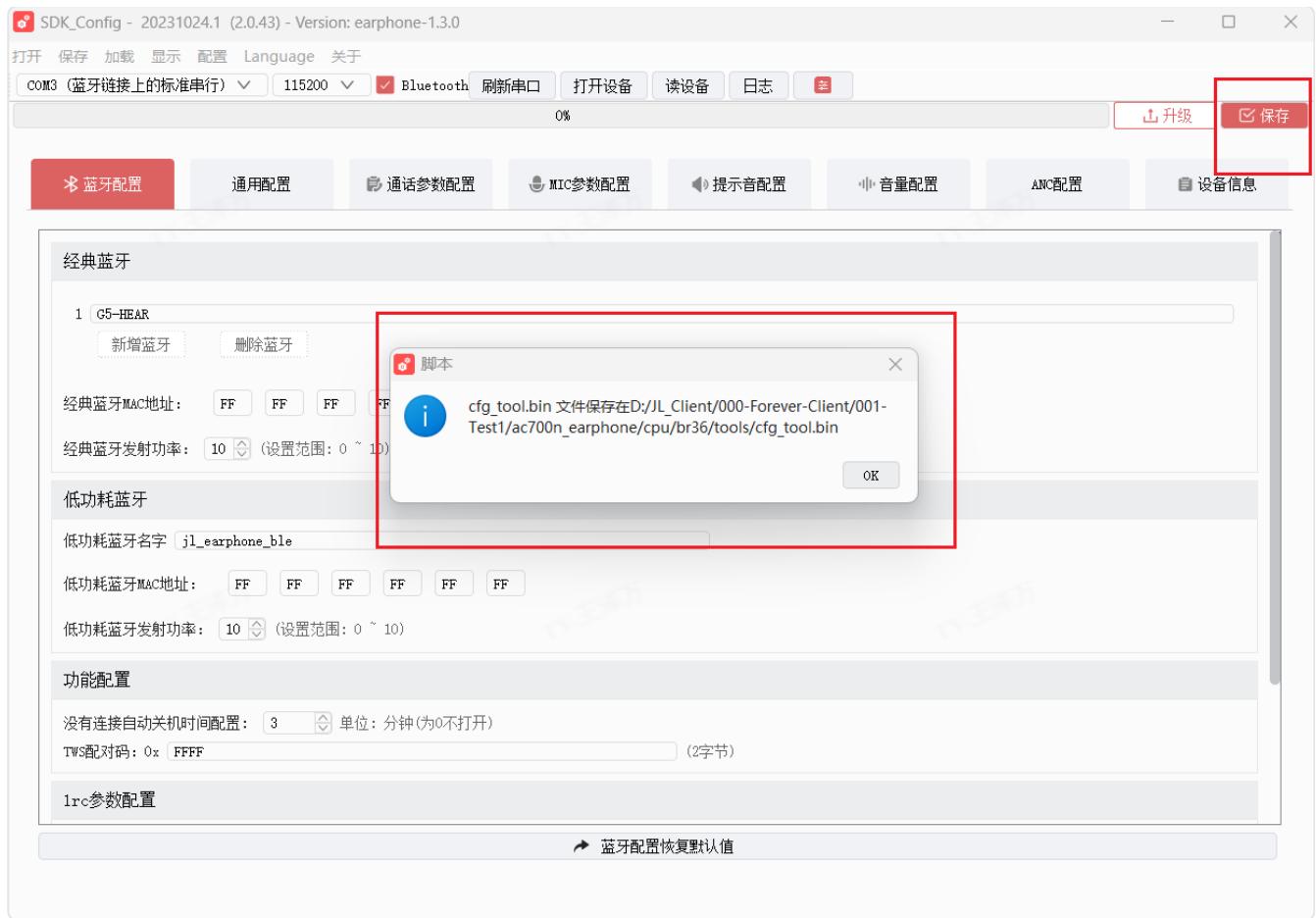
配置工具入口

001-Test1 > ac700n_earphone > cpu > br36 > tools					在 tools 中搜索
	名称	修改日期	类型	大小	
re安装包]	0627降噪女英文提示音	2024/8/29 9:53	文件夹		
	download	2024/8/29 9:53	文件夹		
	extra_tones	2024/8/29 9:53	文件夹		
	513永泰晟AC690X-5D0B.key	2024/8/28 15:22	KEY 文件	1 KB	
	aac.bin	2024/8/28 20:19	BIN 文件	12 KB	
	aaco.bin	2024/8/28 20:19	BIN 文件	35 KB	
	AC700N_配置工具入口.jlxproj	2024/8/28 15:22	杰理工程文件	5 KB	
	aec.bin	2024/8/28 20:19	BIN 文件	13 KB	
	anc_coeff.bin	2024/8/28 15:22	BIN 文件	4 KB	

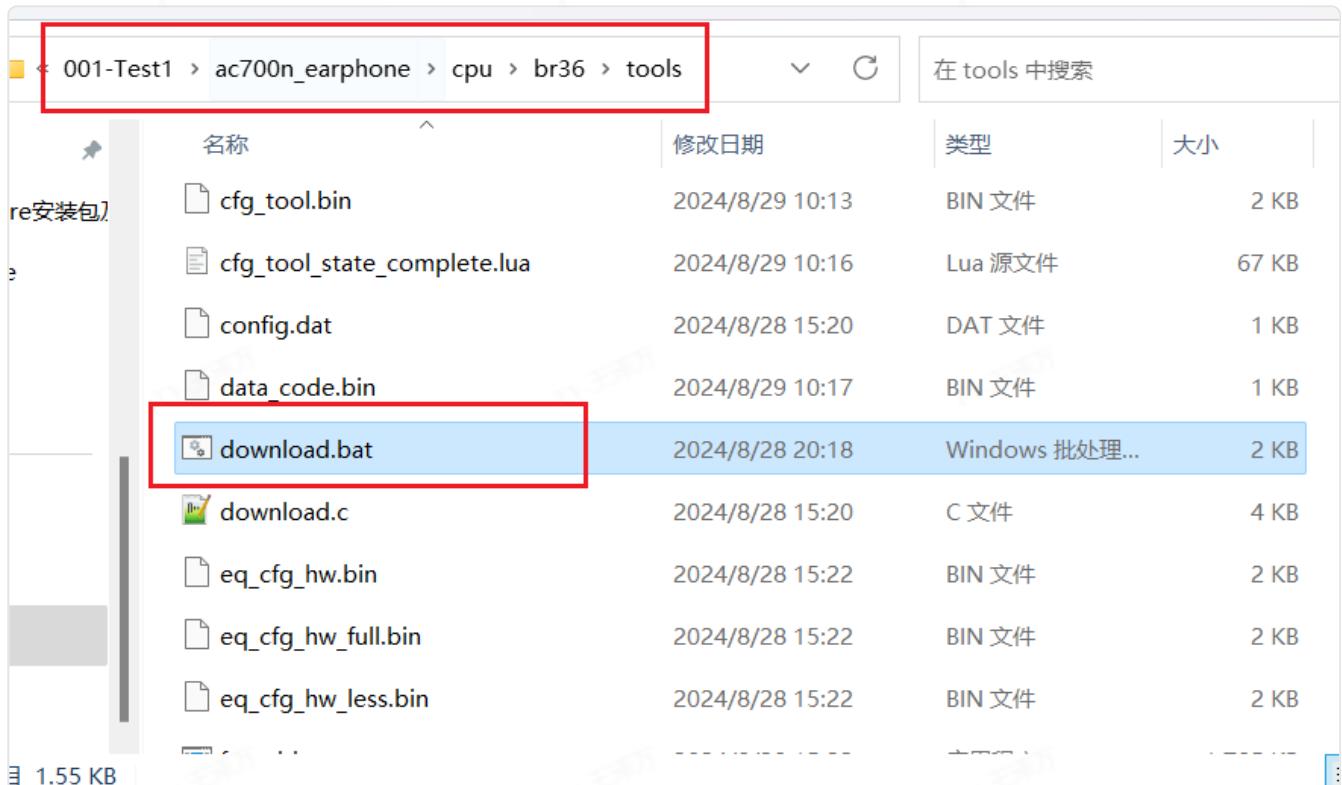
打开各个工具，每个工具都有相应的功能；



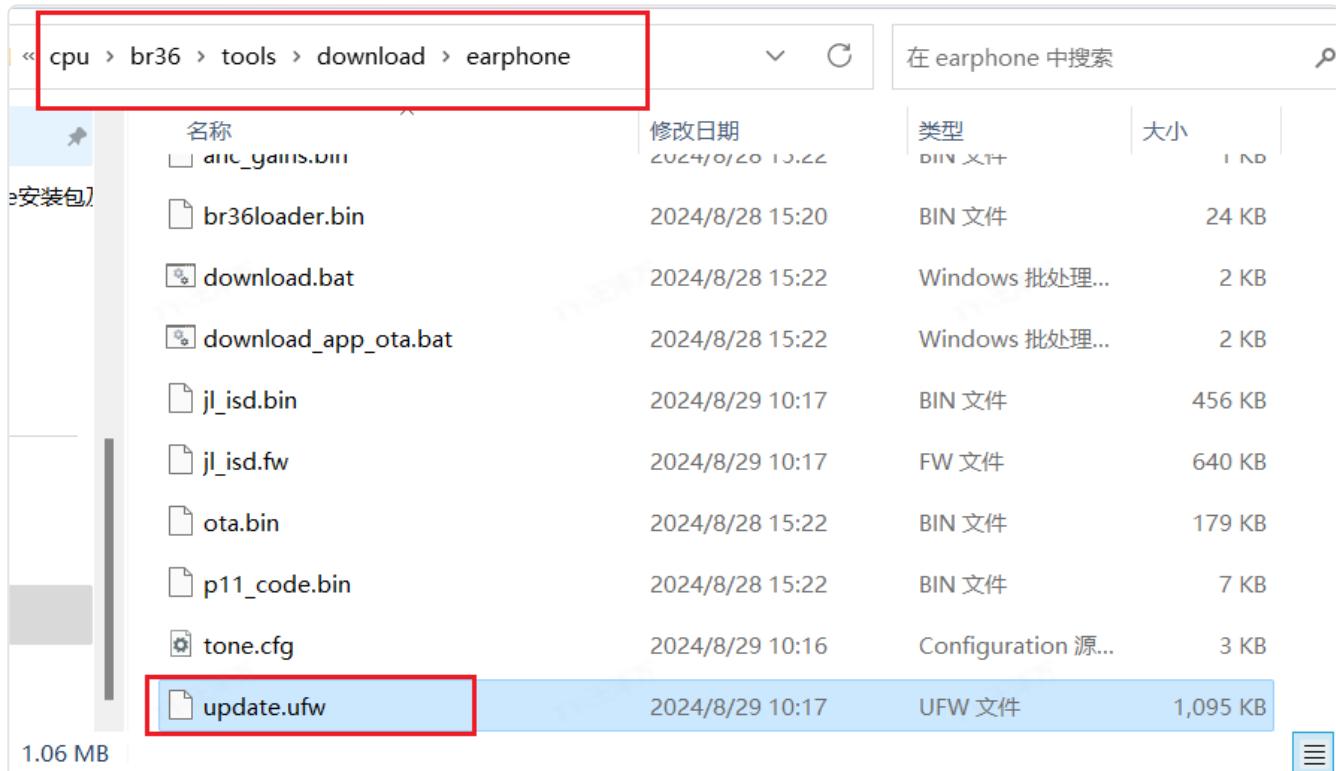
配置工具，配置好想要的功能，点击保存会生成 cf_tool.bin 文件



找到 download.bat 或者到 Code Blocks 里编译



找到 update.ufw 升级文件，打开编辑 FW 文件，

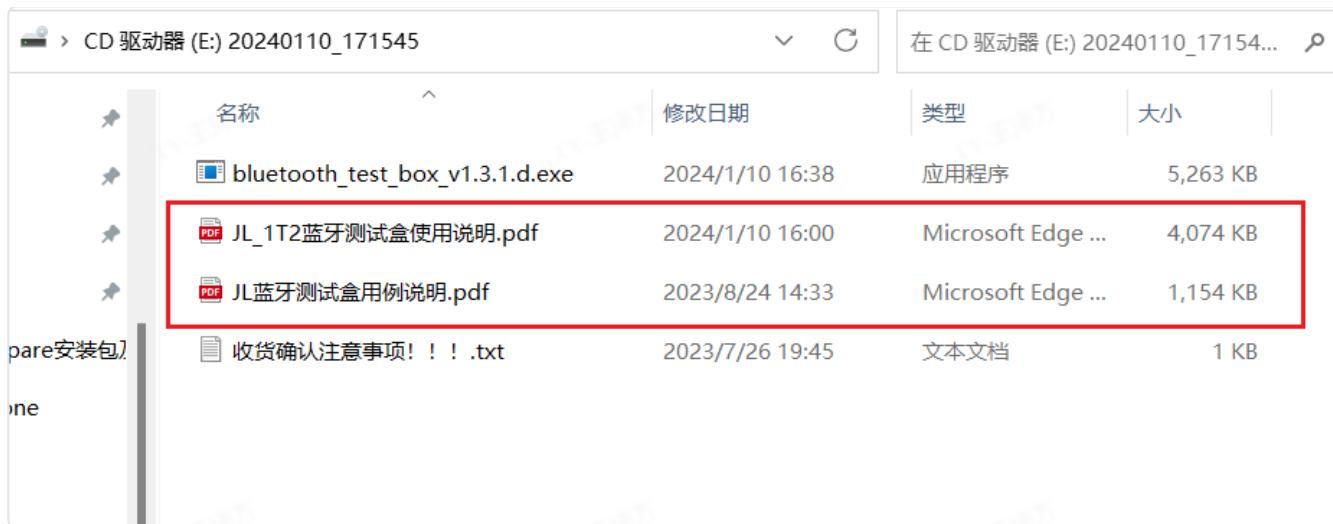


6.5 一拖二蓝牙测试盒的使用

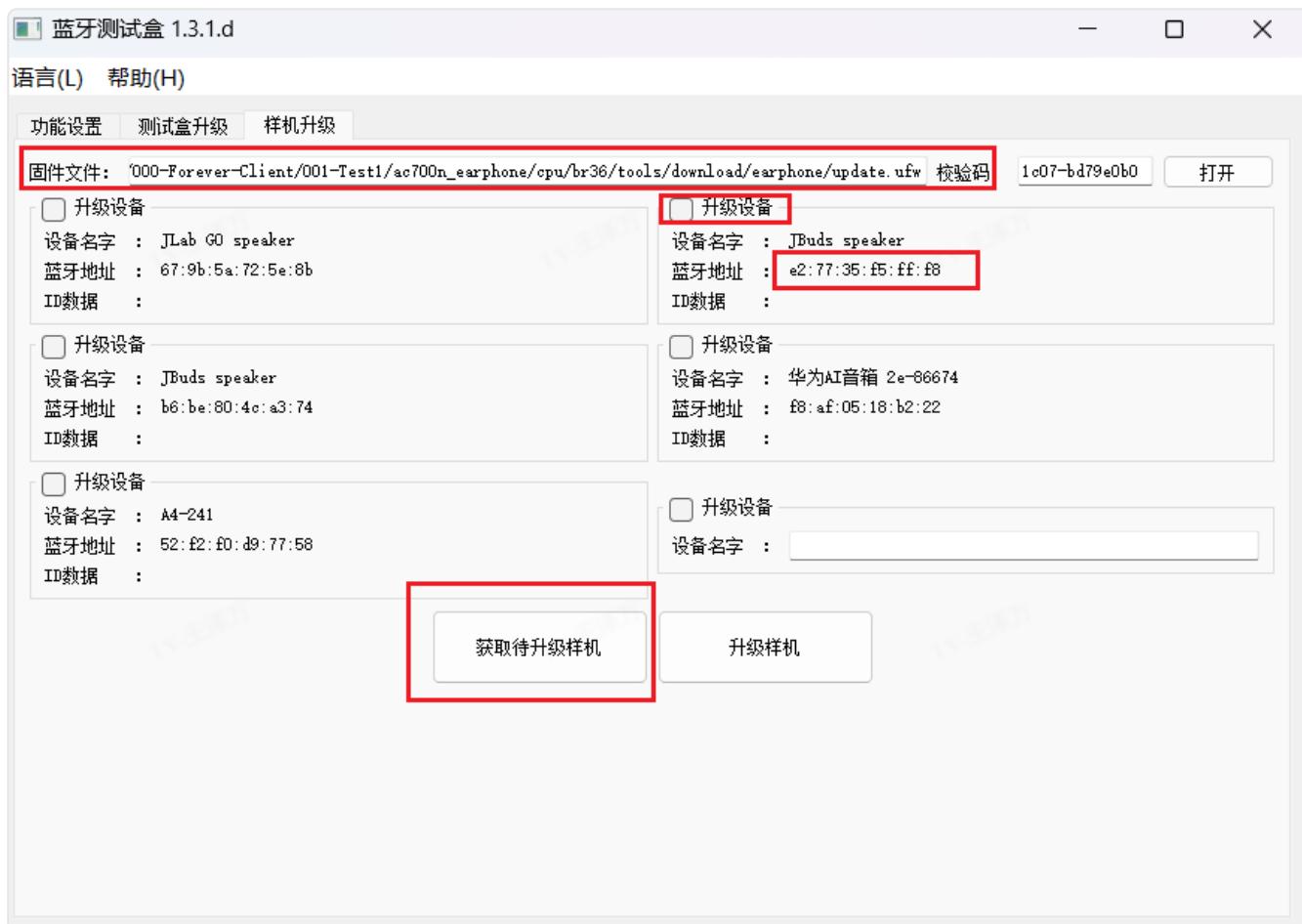
[杰理科技一拖二蓝牙测试盒用户手册](#)

长按 PC 模式按键，电脑会出现弹窗；

会出现 CD 驱动器，里面文件有蓝牙测试盒升级工具以及使用文档；



将 update.ufw 升级文件拖至固件文件，蓝牙测试盒连上样机，点击获取待升级样机，在蓝牙地址有效的设备上勾选升级设备，点击升级样机；



6.6 TWS 耳机开机流程

工程应用技术应用文档

A screenshot of a search interface. On the right, there is a search bar with the text "tws" and a red box highlighting it. Below the search bar are buttons for "查找" (Search) and "替换" (Replace). To the right of the search bar are buttons for "上一个" (Previous) and "下一个" (Next). At the bottom right of the search interface is the number "1/8". The background shows a table with three rows, each containing a module name and its description. The first row is for "HDMI (ARC)", the second for "SPDIF", and the third for "PDM LINK".

3. 蓝牙音频各模块整理文档

模块	各模块具体细节说明
外设 (GPIO、SPI、UART等)	外设相关应用
BT(A2DP/LE) (蓝牙发射 / 接收)	蓝牙相关应用 BLE应用 TWS耳机配对方式使用及生产把控指南 杰理蓝牙距离调试文档 关于提升BLE传输速率说明 GFP谷歌快连及其认证流程指引
MUSIC	MUSIC 模式相关应用 MIDI音乐实现
PC	PC相关应用
FM	FM相关应用 AC695N_AC696N FM天线ESD静电防护措施说明V1.1_20201218 fm应用详细设计说明书 杰理FM硬件指南-20230315
AUX/ADC	LINEIN_AUX相关应用 ADC模块
Audio(EQ, 混响, IIS,HDMI)	软件： AUDIO相关应用 新Tools的音效调试使用 混响应用使用说明 音效调式 音箱耳机低音小音量增强实现方法 硬件：

C/C++

```
1 void app_main(){
2
3     int update = 0;
4     u32 addr = 0, size = 0;
5     struct intent it;
6
7
8     log_info("app_main\n");
9     app_var.start_time = timer_get_ms();
10
11 ...
12
13     if (!UPDATE_SUPPORT_DEV_IS_NULL()) {
14         update = update_result_deal();
15     }
16
17     app_var_init(); //app模式初始化，该函数会播放开机提示音;
18
19     init_intent(&it);
20     it.name = "earphone";
21     it.action = ACTION_EARPHONE_MAIN;
22     start_app(&it); //启动耳机功能，包括TWS模式;
23
24 ...
25
26 }
```

搜索 earphone，找到 earphone.c 文件，并且找到 star_app() 函数；

```
ard_config.h | C board_ac700n_demo_cfg.h M ● | C bt_tws.c | C app_main.c | C earphone.c 9+ X | C app_core.h | C uart

> earphone > C earphone.c > state_machine(application *, app_state state, intent *)
int bt_app_exit(void *priv)
    sys_auto_shut_down_disable();
    return 1;
}

/*
 * earphone 模式状态机, 通过start_app()控制状态切换
 */
/* extern int audio_mic_init(); */

static int state_machine(struct application *app, enum app_state state, struct intent *it)
{
    int error = 0;

#endif
}
#endif
break;
case APP_STA_START:
    //进入蓝牙模式,UI退出充电状态
    ui_update_status(STATUS_EXIT_LOWPOWER);
    if (!it) {
        break;
    }
    switch (it->action) {
case ACTION_EARPHONE_MAIN:
    /*
     * earphone 模式初始化
     */

        /*set_mode_normal 0:user normal parm 1:user bqb parm*/
        /* set_mode_dut 0:user normal parm 1:user bqb parm */
        /* 库里模式设置 set set_mode_normal=0,set_mode_dut=0 */
    }
}
```

```
/*
 * 注册earphone模式
 */
REGISTER_APPLICATION(app_earphone) = {
    .name    = "earphone",
    .action  = ACTION_EARPHONE_MAIN,
    .ops     = &app_earphone_ops,
    .state   = APP_STA_DESTROY,
};
```

7. Git 环境配置

7.1 SSH Key 配置

公司项目需要使用公司所给账号和公司网络才能登上；



找到 PuTTY 生成密钥，生成密钥时不断在软件窗口内滑动，加快生成密钥；
并保存密钥和私钥，密钥需要自己起名字保存；

电脑 > Data (D:) > TortoiseGit > Key

名称	修改日期
prikey.ppk	2024/9/2 11:0
pubkey	2024/9/2 11:0

PuTTY Key Generator

File Key Conversions Help

Key

Public key for pasting into OpenSSH authorized_keys file:

```
ssh-rsa AAAQAB3Nj...C1...2EAAAQABAAQAAQCCUf7j41ITVDXwUsjQtWNjY4MoKV/i+0fyBwcBF  
+YpsMJP0...6F1PhTQd7Z4oF0eXMZv6fhcTdldX0vozV4m4UQQXrb8uEECyfd  
MC6nnllkp...nLTgAgRSg2DdNx  
+1UJHG6bd04000IMBGt2IwVREngxIRyfTw4nqZov1tsWbQUKciUWz  
+yADzbgrOU3cl15ZPBhQuv7/2VIHe1vhzS5YqlbeMGLTFRkRxvV59mXBCTasIO7D4ZNDT9QFYqaj2nbjBHJaaZs5AV
```

Key fingerprint: ssh-rsa 2048 SHA256:yMeDjonfdFpilaKO73mvv5/7prNSlxNz6uCEIB7Visg

Key comment: rsa-key-20240902

Key passphrase:

Confirm passphrase:

Actions

Generate a public/private key pair **Generate**

Load an existing private key file **Load**

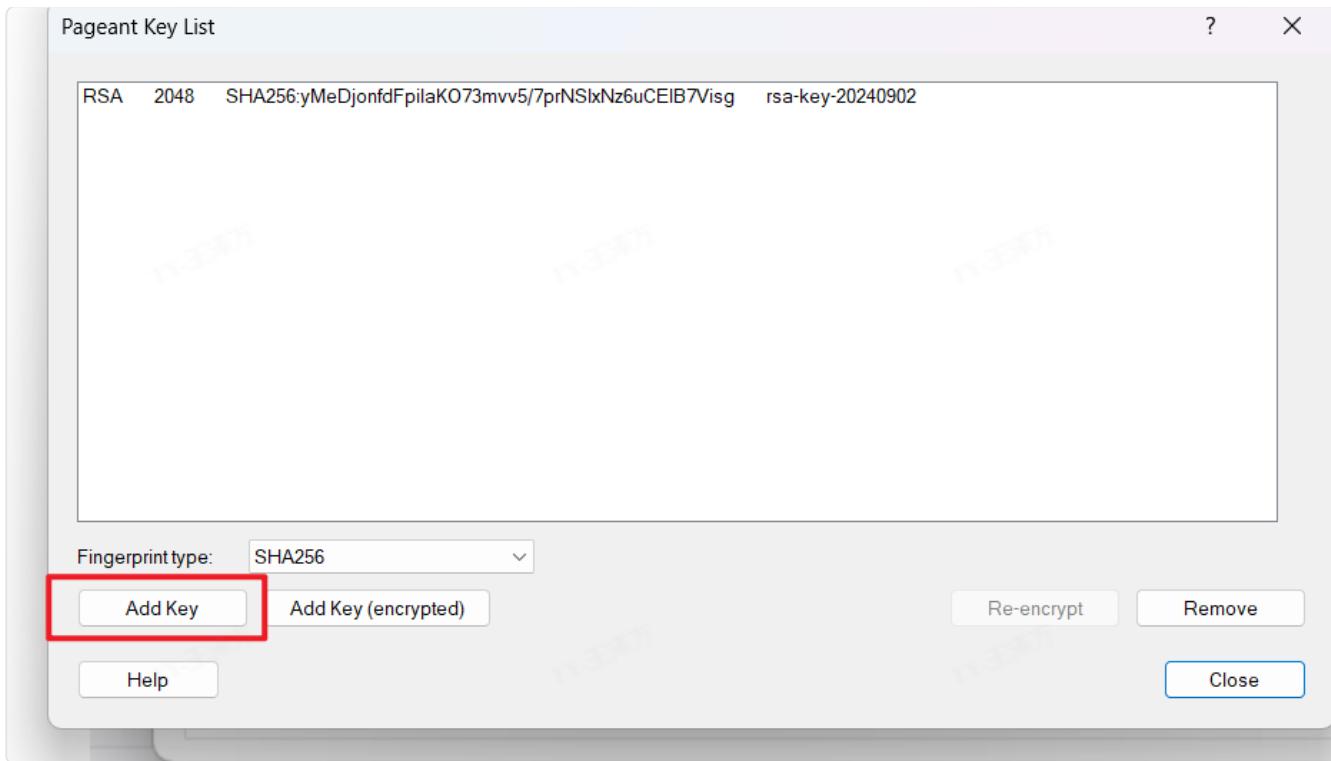
Save the generated key **Save public key** **Save private key**

Parameters

Type of key to generate: RSA DSA ECDSA EdDSA SSH-1 (RSA)

Number of bits in a generated key: 2048

在打开 Pageant，添加私钥；



将公钥复制到 Gitlab 上的 SSH Keys 配置

The screenshot shows the 'User Settings / SSH Keys' page on GitLab. On the left, there's a sidebar with 'User settings' and a list of options: Profile, Account, Billing, Applications, Chat, Access tokens, Emails, Password, Notifications, SSH Keys (which is selected and highlighted with a grey background), GPG Keys, and Preferences. The main content area is titled 'SSH Keys' and contains the following text: 'SSH keys allow you to establish a secure connection between your computer and GitLab. SSH fingerprints verify that the client is connecting to the correct host. Check the current instance configuration.' Below this, there's a table titled 'Your SSH keys' with one entry: user1@DESKTOP-JK899VO d0:17:14:b7:f7:1d:26:9d:ef:ec:69:fe:eb:4f:5b:a9. The table has columns: Title, Key, Usage type, Created, Last used, Expires, and Actions. The 'Actions' column for the key has two buttons: 'Revoke' and a trash can icon. A red box highlights the 'Add new key' button at the top right of the table.

7.2 提交与推送

点击克隆，选择 HTTP 的地址复制；

Create merge request

Branch_c015e055 ac700n_earphone / +

时间:20240902123316 王泽万 authored 13 minutes ago

Name	Last commit	
apps	时间:20240902123316	
cpu/br36	时间:20240902123316	
doc	继承自:	
include_lib	继承自:	3 months ago
obj/Release	时间:20240902123316	12 minutes ago
tools	继承自:	3 months ago

History Find file Web IDE Clone

Clone with SSH
git@192.168.1.245:freebrand/yts/ac

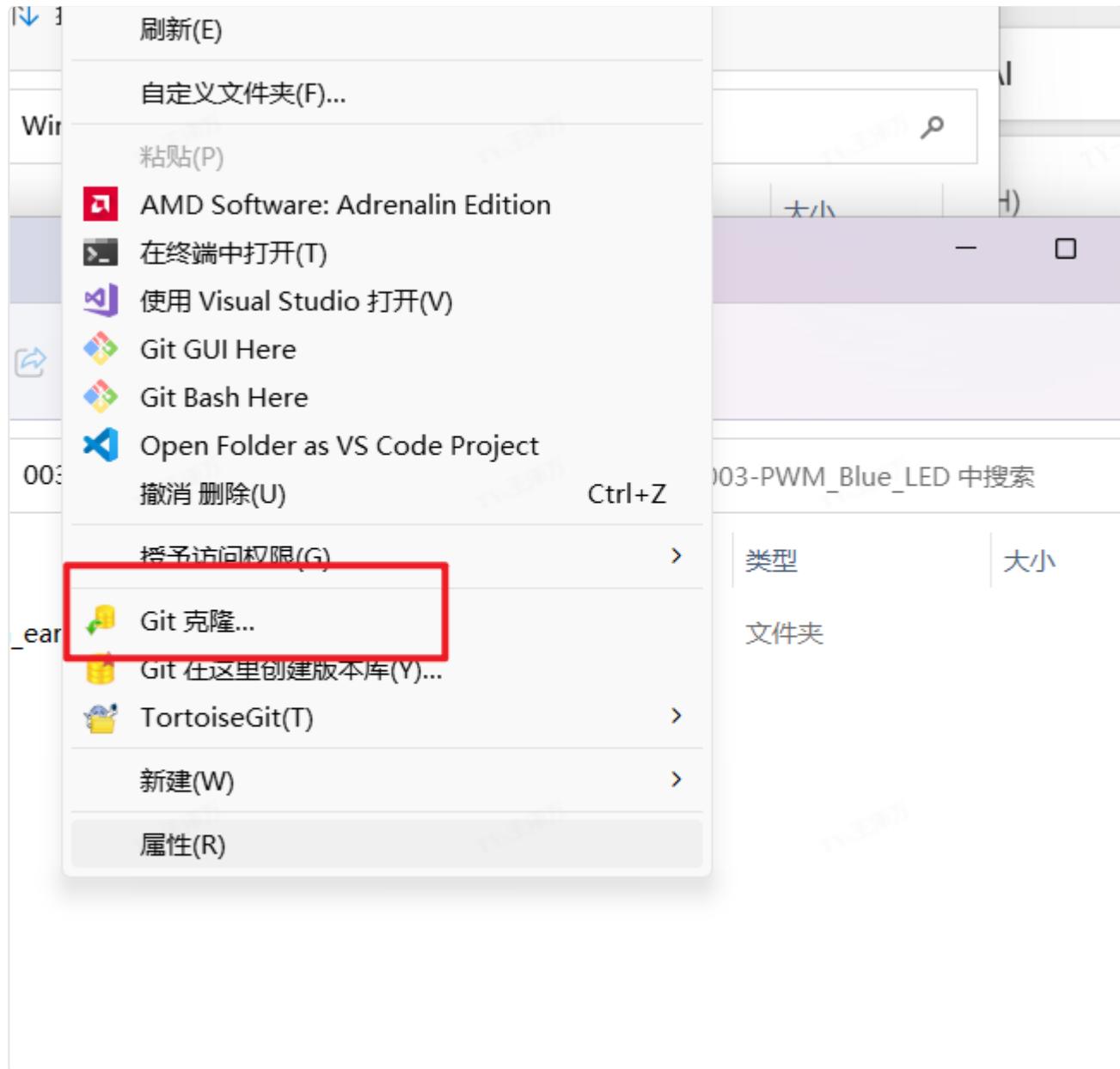
Clone with HTTP
http://192.168.1.245/freebrand/yts

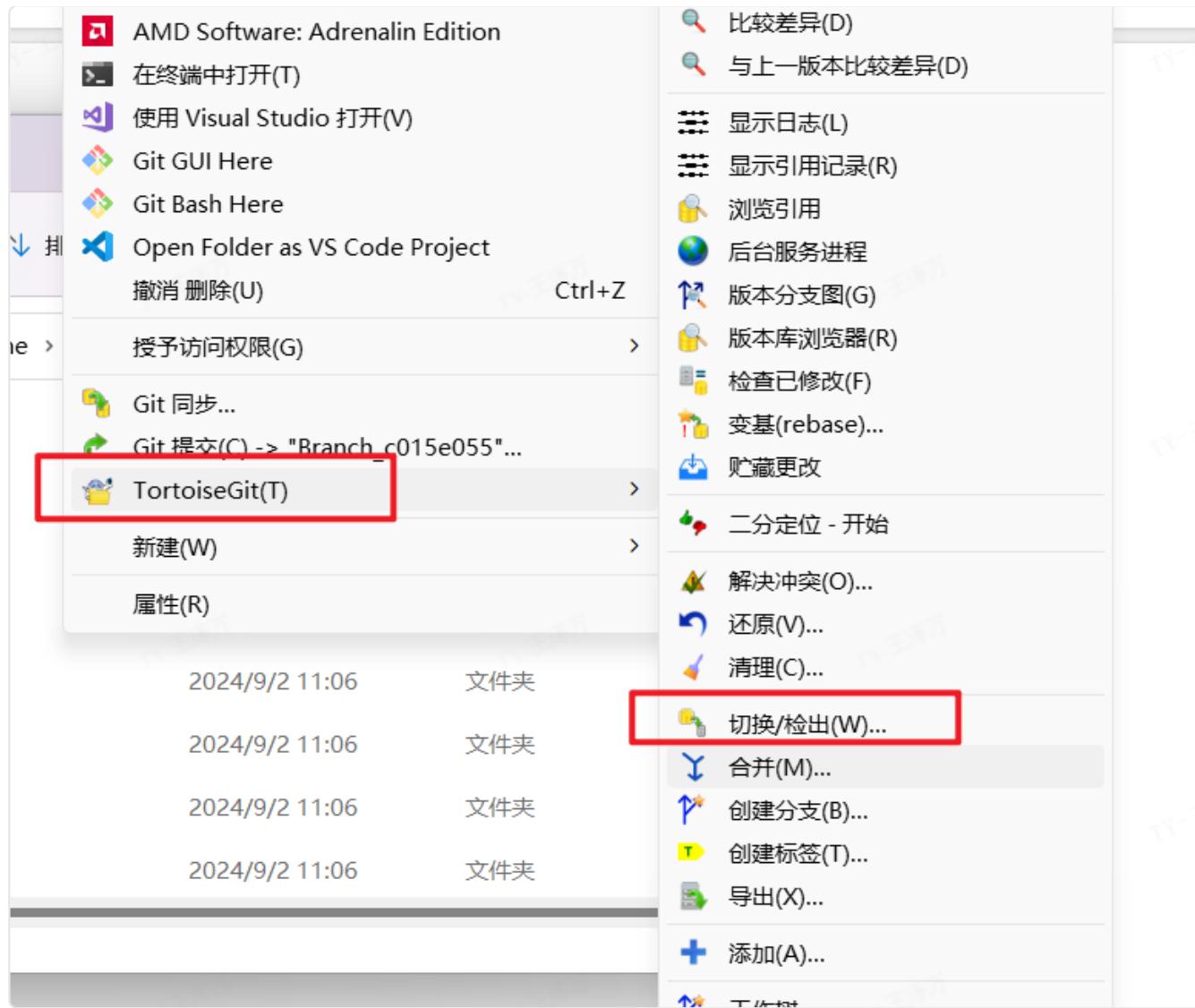
Open in your IDE
Visual Studio Code (SSH)
Visual Studio Code (HTTPS)

The screenshot shows a Git commit history interface. At the top left, there's a dropdown menu labeled 'Branch_c015e055' with a red box around it. To its right is the repository name 'ac700n_earphone /' and a '+' button. Below this is a commit from '王泽万' made 13 minutes ago. The main area lists several files or folders with their last commit times. On the right side, there's a 'Clone' button with a red box around it, followed by options for cloning via SSH or HTTP, and links to open the repository in Visual Studio Code via SSH or HTTPS.

注意：是在分支那克隆的，但是实际上当你在文件夹里克隆之后，显示的还是主文件；

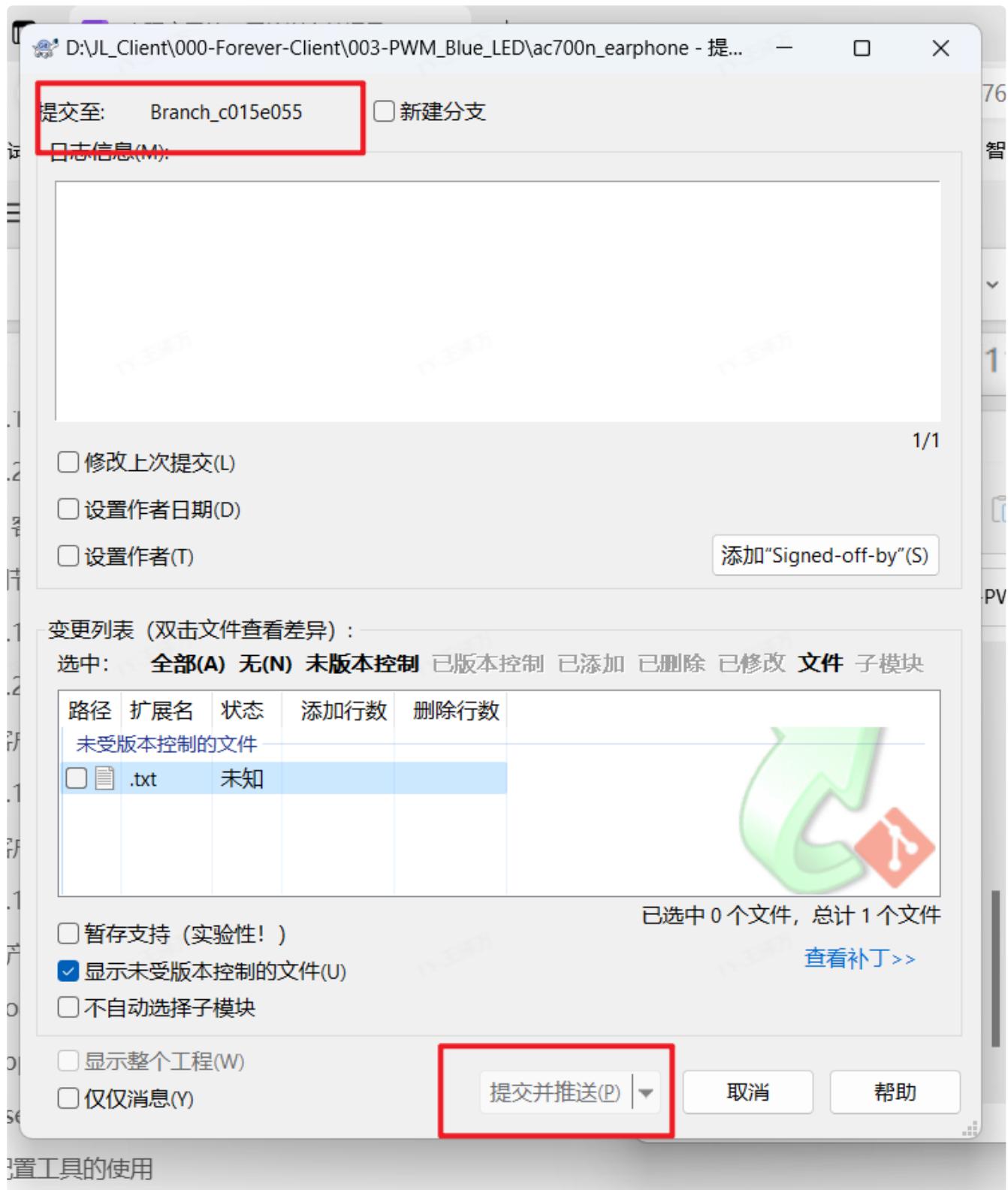
因此克隆完需要切换分支；





提交的话，要注意提交到哪里去；

提交只是提交到本地的库，而推送是推送的远程；



提交推送成功! 回到 GitLab 会有推送信息, 包括修改的信息;

Commit 7c10cf58 🏷 authored 47 minutes ago by 王泽万

时间:20240902123316

作者:WZW

项目名:G5-AC7003D4-ANCG5-V136-G5-HEAR

主控:AC7003D4

蓝牙名:"G5-HEAR"

SDK版本:"1.3.6"

声道配置方式:"主左"

GIT分支:"Branch_c015e055"

GIT远端地址:"http://192.168.1.245/freebrand/yts/ac700n_earphone.git"

本地地址:"D:\JL_Client\000-Forever-Client\003-PWM_Blue_LED\ac700n_earphone"

更新内容:

1. 开机后，蓝色呼吸灯开启

绿色为修改之后的部分，红色为修改前的部分；

Release_file.bat 📁



View file @7c10cf58

```
...   ... @@ -30,7 +30,7 @@ set "GET_CHECK_EN=Y"  
30      30 @REM set "EXCLUDE_TYPE=*.key"  
31      31  
32      32 @REM 运行打包处理  
33      33 - call C:\usertool\run.bat  
34      34 + call C:\pack-tools\run.bat  
35      35  
36      36  
...   ...
```

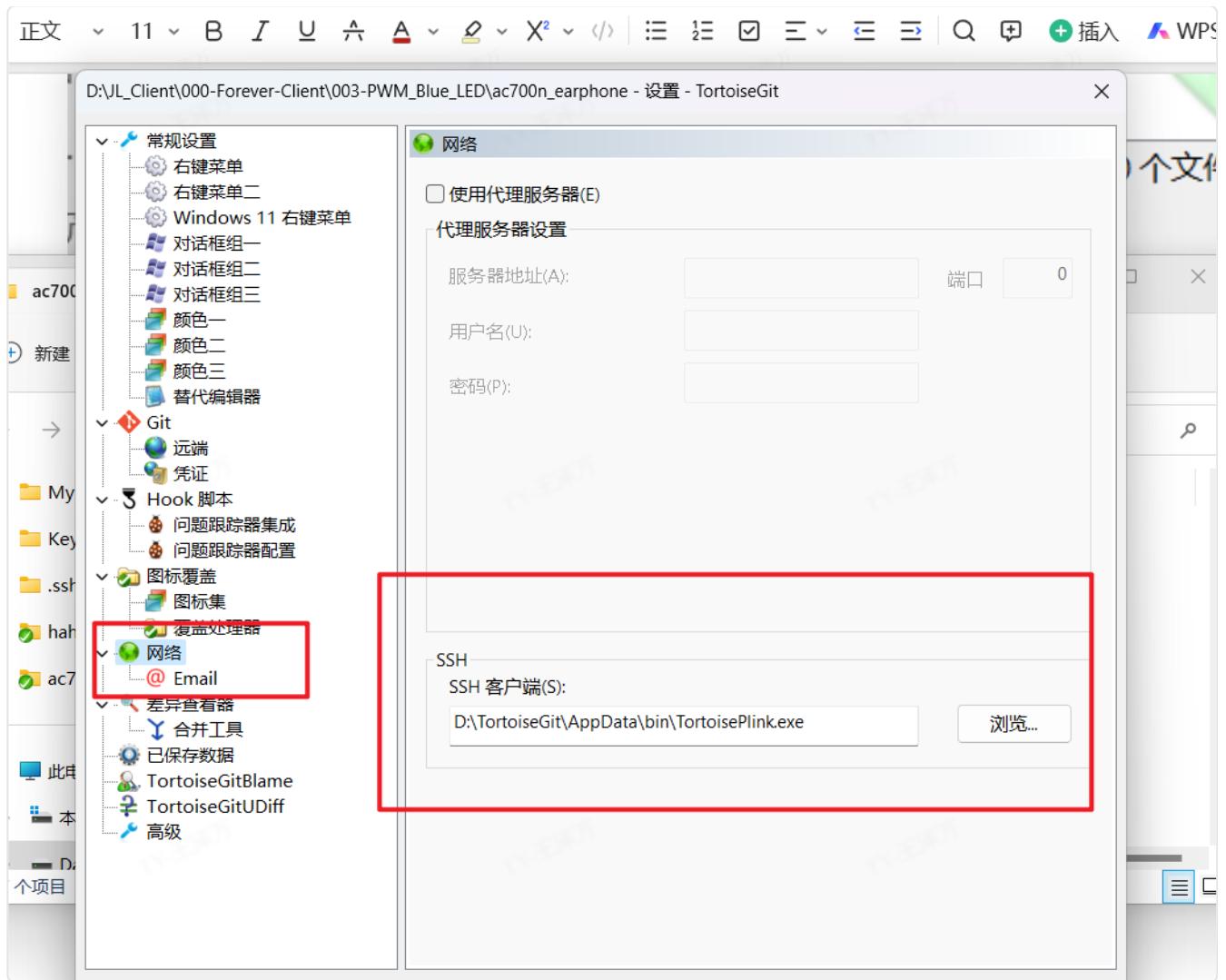
apps/earphone/board/br36/board_ac700n_demo.c

```
43 - .power_on = PWM_LED_ALL_OFF,
43 + .power_on = PWM_LED0_BREATHE, //客户需求:开机蓝色呼吸灯,在关机之前都改为蓝色呼吸灯模式
44 44 .power_off = PWM_LED_ALL_OFF,
45 - .lowpower = PWM_LED_ALL_OFF,
46 - .max_vol = PWM_LED_ALL_OFF,
47 - .phone_in = PWM_LED_ALL_OFF,
48 - .phone_out = PWM_LED_ALL_OFF,
49 - .phone_activ = PWM_LED_ALL_OFF,
50 - .bt_init_ok = PWM_LED_ALL_OFF,
51 - .bt_connect_ok = PWM_LED_ALL_OFF,
52 - .bt_disconnect = PWM_LED_ALL_OFF,
53 - .tws_connect_ok = PWM_LED_ALL_OFF,
54 - .tws_disconnect = PWM_LED_ALL_OFF,
45 + .lowpower = PWM_LED0_BREATHE,
46 + .max_vol = PWM_LED0_BREATHE,
47 + .phone_in = PWM_LED0_BREATHE,
48 + .phone_out = PWM_LED0_BREATHE,
49 + .phone_activ = PWM_LED0_BREATHE,
50 + .bt_init_ok = PWM_LED0_BREATHE,
51 + .bt_connect_ok = PWM_LED0_BREATHE,
52 + .bt_disconnect = PWM_LED0_BREATHE,
53 + .tws_connect_ok = PWM_LED0_BREATHE,
54 + .tws_disconnect = PWM_LED0_BREATHE.
```

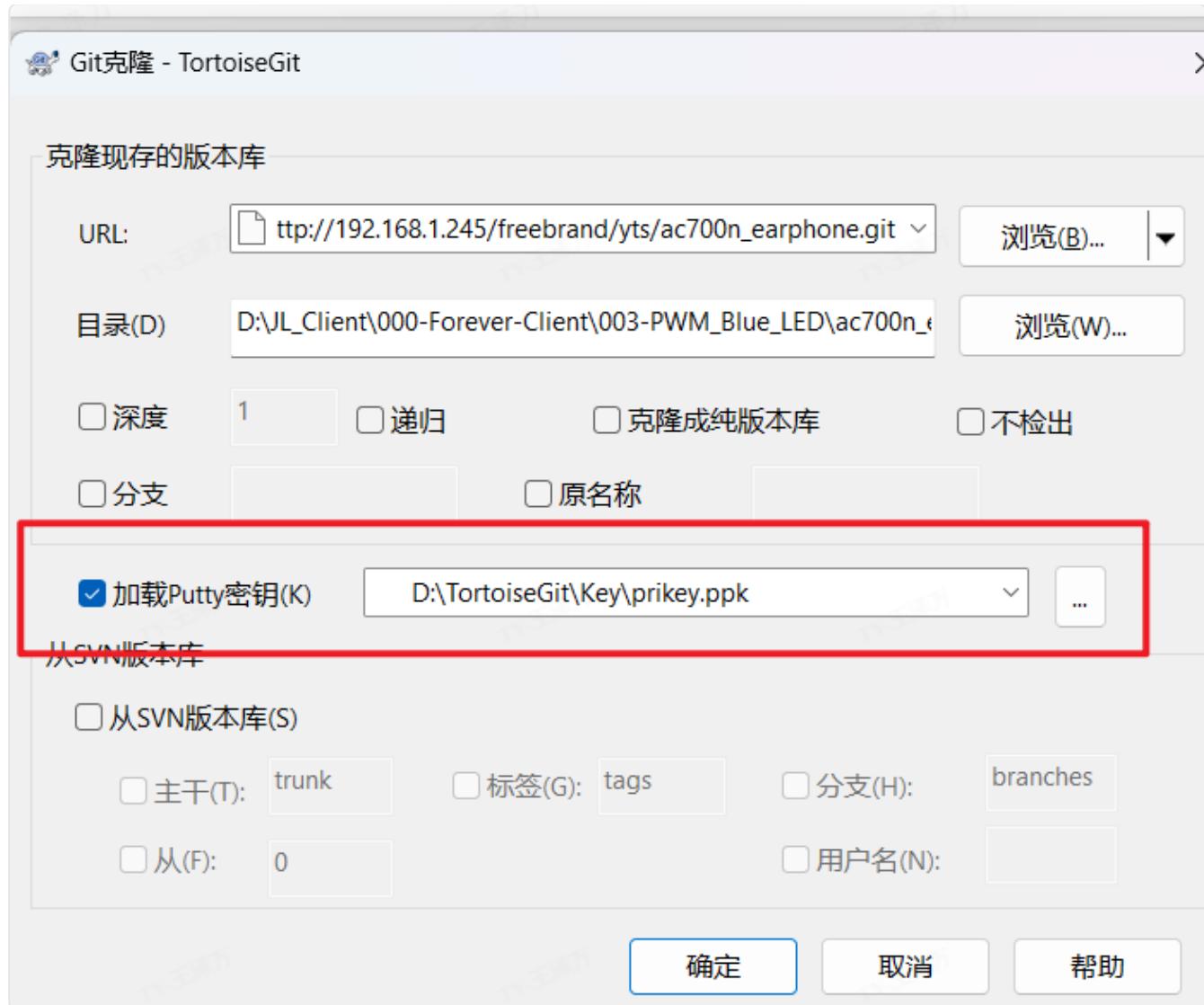
7.3 注意事项：克隆相关一直显示密码输入

解决方法：

需要在 TortoiseGit 的设置里，网络这里的默认启动应用改为 TortoisePlink.exe



找到刚刚保存好的密钥，在今后的操作里一旦出现加载密钥的选项就把它勾选上；



8. 软件调试方法

工程应用技术应用文档

进去文档详细查看即可；

1. 资料链接下载

环境安装	杰理WINDOWS环境安装及其他事项 LINUX安装和编译
最新SDK&补丁情况	音频类补丁汇总表 对外特别发布开发注意点信息汇总
认证资料获取汇总	认证资料获取汇总(最新资料后台会更新) 《Hi-Res申请指南v1.0》
杰理常用工具及软件调试方法文档	杰理工具文档 (开发环境、量产相关工具等) 软件调试方法 Window下使用Makefile编译代码教程 欢迎使用杰理OTA外接库开发文档(Android) — 杰理OTA外接库开发文档(Android) 1.6.0 documentation
常用工具使用	常规设备使用指导

二、杰理实习第三周日志

前言：

上周的任务是先熟悉好资料较为完善的 SDK，我所选择的是 AC695N，在熟悉完蓝牙的流程之后；蓝牙部门那边将某款客户的耳机测试部分交给了我帮忙，并且还帮忙为客户工程做维护工作；当然测试工作与客户工程的维护工作还未结束，在这空闲的档期，继续将第一周时委派的任务——验证公版程序是否存在如客户所说的 bug。

1. 问题分析：

检查这个 JL701N 1.0.0 版本的音响 SDK 是否有如下问题：

- 开机进入其他模式连接不了无线麦，并且开机进入其他模式需要初始化蓝牙；
- 报完提示音后有线话筒没有声音，需要重现初始化麦克风音量才有声音；

保存路径 G:\可视化内测项目 选择

* 名称 新建项目

* 类型 音箱

* SDK le_audio

* 芯片 JL701N

* 版本 1.0.0 插入特殊补丁密钥

* 封装 JL701N-demo

检测更新 创建

1.1 查询 JL701N 1.0.0 版本 SDK 的板级配置