Mode Management Interface

Zhuhai Jieli Technology Co., Ltd.

Zhuhai Jieli Technologyco.,LTD

August 2020

Table of contents

1.Mode Management Description

Mode management interface is application layerappThe mode provides switching, querying and other operations to ensure orderly switching and response of various application scenarios.

2.Interface Introduction

//Switch to the previous valid mode

void app_task_switch_prev(); //

Switch to the next valid mode

void app_task_switch_next(); //

Return to previous mode

int app_task_switch_back(); //

Switch to the specified mode

int app_task_switch_to(u8 app_task); //Get

the current modeid

u8 app_get_curr_task();

//passidCheck if it is the current mode

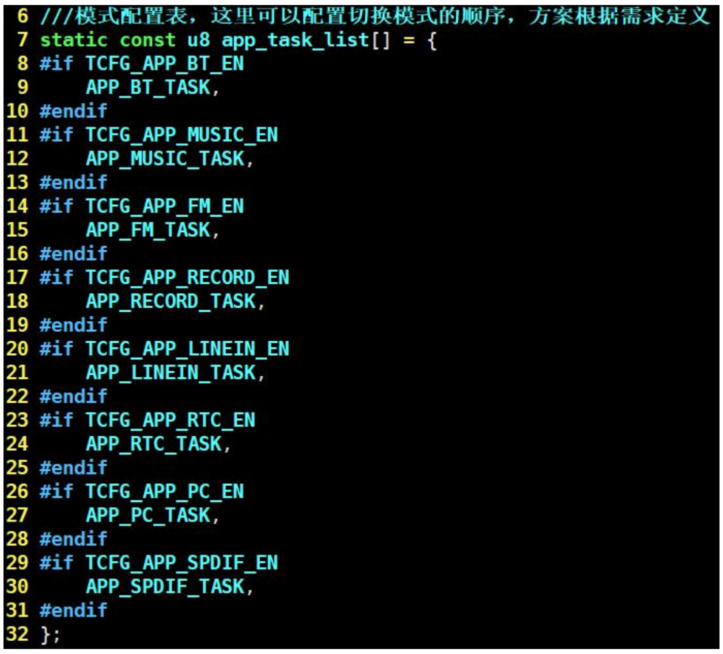
u8 app_check_curr_task(u8 app); //

Mode switch exit detection

u8 app_task_exitting();

3.Mode Switching Table

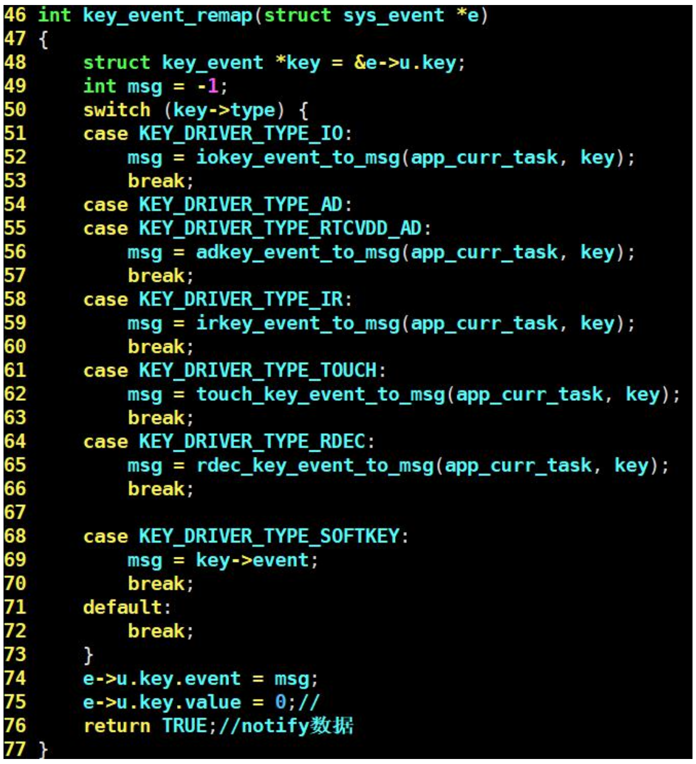
///Mode configuration table, where you can configure the order of switching modes, and the scheme is defined according to requirements



```
///模式配置表，这里可以配置切换模式的顺序，方案根据需求定义
static const u8 app_task_list[] = {
#if TCFG_APP_BT_EN
    APP_BT_TASK,
#endif
#if TCFG_APP_MUSIC_EN
    APP_MUSIC_TASK,
#endif
#if TCFG_APP_FM_EN
    APP_FM_TASK,
#endif
#if TCFG_APP_RECORD_EN
    APP_RECORD_TASK,
#endif
#if TCFG_APP_LINEIN_EN
    APP_LINEIN_TASK,
#endif
#if TCFG_APP_RTC_EN
    APP_RTC_TASK,
#endif
#if TCFG_APP_PC_EN
    APP_PC_TASK,
#endif
#if TCFG_APP_SPDIF_EN
    APP_SPDIF_TASK,
#endif
};
```

4.Key Mapping

After the key driver detects the key, it willnotifyBefore the key event, the key is mapped. The mapping process is as follows (mapping is performed according to different key types):



```
int key_event_remap(struct sys_event *e)
{
    struct key_event *key = &e->u.key;
    int msg = -1;
    switch (key->type) {
    case KEY_DRIVER_TYPE_IO:
        msg = iokey_event_to_msg(app_curr_task, key);
        break;
    case KEY_DRIVER_TYPE_AD:
    case KEY_DRIVER_TYPE_RTCVDD_AD:
        msg = adkey_event_to_msg(app_curr_task, key);
        break;
    case KEY_DRIVER_TYPE_IR:
        msg = irkey_event_to_msg(app_curr_task, key);
        break;
    case KEY_DRIVER_TYPE_TOUCH:
        msg = touch_key_event_to_msg(app_curr_task, key);
        break;
    case KEY_DRIVER_TYPE_RDEC:
        msg = rdec_key_event_to_msg(app_curr_task, key);
        break;

    case KEY_DRIVER_TYPE_SOFTKEY:
        msg = key->event;
        break;
    default:
        break;
    }
    e->u.key.event = msg;
    e->u.key.value = 0;//
    return TRUE;//notify数据
}
```

5.Mode message sending and receiving interface

//appCustom message sending interface

int app_task_put_usr_msg(int msg, int arg_num, ...);

//appMessage acquisition interface (blockThe parameters are0Indicates internalpend,1Direct return) void app_task_get_msg(int *msg, int msg_size, int block); //appKey message sending interface

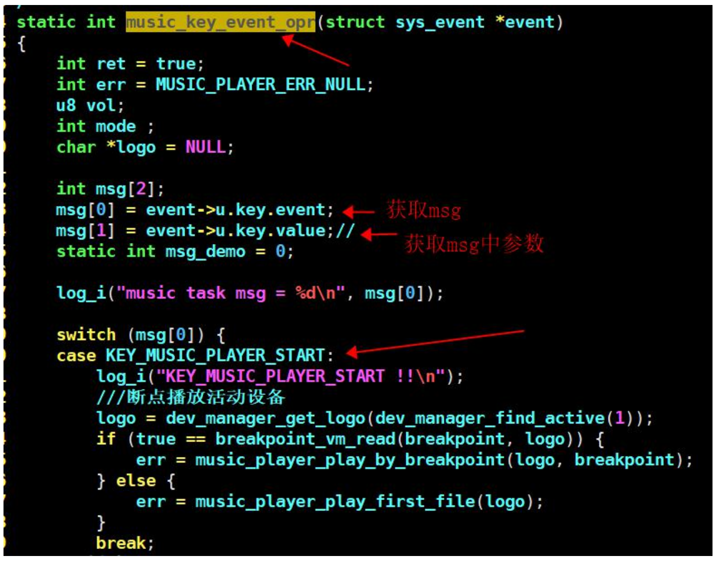
int app_task_put_key_msg(int msg, int value);

Application process message sending interface **app_task_put_key_msg**, the message is enumerated inkey_event_deal.hDefined in the respective modes Respond to key events (**SYS_KEY_EVENT**),like:

keyMessage sending

app_task_put_key_msg(KEY_MUSIC_PLAYER_START, 0);

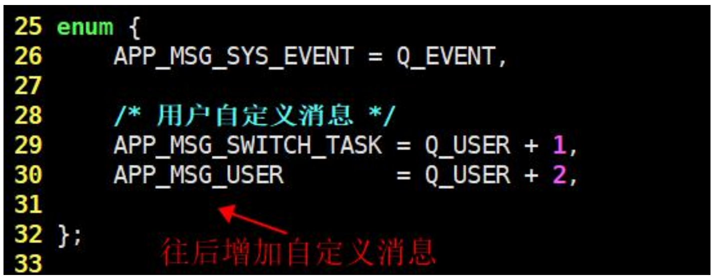
keyMessage Response



```c
static int music_key_event_opr(struct sys_event *event)
{
    int ret = true;
    int err = MUSIC_PLAYER_ERR_NULL;
    u8 vol;
    int mode ;
    char *logo = NULL;

    int msg[2];
    msg[0] = event->u.key.event;
    msg[1] = event->u.key.value;//
    static int msg_demo = 0;

    log_i("music task msg = %d\n", msg[0]);

    switch (msg[0]) {
    case KEY_MUSIC_PLAYER_START:
        log_i("KEY_MUSIC_PLAYER_START !!\n");
        ///断点播放活动设备
        logo = dev_manager_get_logo(dev_manager_find_active(1));
        if (true == breakpoint_vm_read(breakpoint, logo)) {
            err = music_player_play_by_breakpoint(logo, breakpoint);
        } else {
            err = music_player_play_first_file(logo);
        }
        break;
```

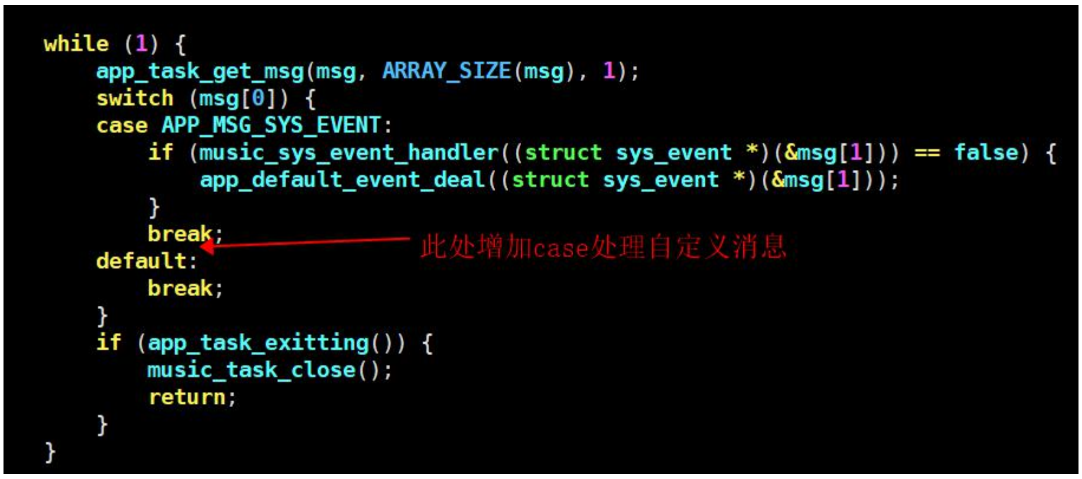
Message Extensions

It is not recommended to use it without special requirements.app_task_put_usr_msg, only used when multiple parameters need to be transmitted, the message is enumerated in app_task.hThe definition is as follows:



```c
enum {
    APP_MSG_SYS_EVENT = Q_EVENT,

    /* 用户自定义消息 */
    APP_MSG_SWITCH_TASK = Q_USER + 1,
    APP_MSG_USER        = Q_USER + 2,

};
```

Customize message acquisition processing and add it to the message acquisition in the current modecaseJust respond:



```c
while (1) {
    app_task_get_msg(msg, ARRAY_SIZE(msg), 1);
    switch (msg[0]) {
    case APP_MSG_SYS_EVENT:
        if (music_sys_event_handler((struct sys_event *)(&msg[1])) == false) {
            app_default_event_deal((struct sys_event *)(&msg[1]));
        }
        break;
    default:
        break;
    }
    if (app_task_exitting()) {
        music_task_close();
        return;
    }
}
```

6.Detailed interface comments

//*--- --------------------------*//
**@brief Switch to the previous mode
@param
@return
@note
* /
/*-- ----------------------------*/

void app_task_switch_prev()

//*--- --------------------------*//
**@brief Switch to next mode
@param
@return
@note
* /
/*-- ----------------------------*/

void app_task_switch_next()

//*--- --------------------------*//
**@brief Switch to the specified mode
@param app_task:Specifying a Mode
@return
@note
* /
/*-- ----------------------------*/

int app_task_switch_to(u8 app_task)

//*--- --------------------------*//
**@brief Jump back to the original mode
@param
@return
@note
* /
/*-- ----------------------------*/

int app_task_switch_back()

```
//*-------------------------------------------------- --------------------------*//
**@brief        Mode switch exit detection
    @param
    @return      1:Respond to exit mode,0:Not Responding
    @note
* /
/*------------------------------------------------ ----------------------------*/
```

u8 app_task_exitting()//

```
//*-------------------------------------------------- --------------------------*//
**@brief        Get the current mode
    @param
    @return      Current Modeid
    @note
* /
/*------------------------------------------------ ----------------------------*/
```

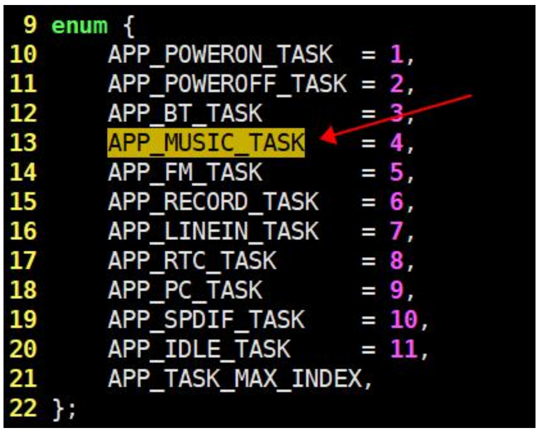
u8 app_get_curr_task()

```
//*-------------------------------------------------- --------------------------*//
**@brief        By specifyingidCheck if it is the current mode
    @param
    @return      true:is the current mode,false:Not current mode
    @note
* /
/*------------------------------------------------ ----------------------------*/
```

u8 app_check_curr_task(u8 app)

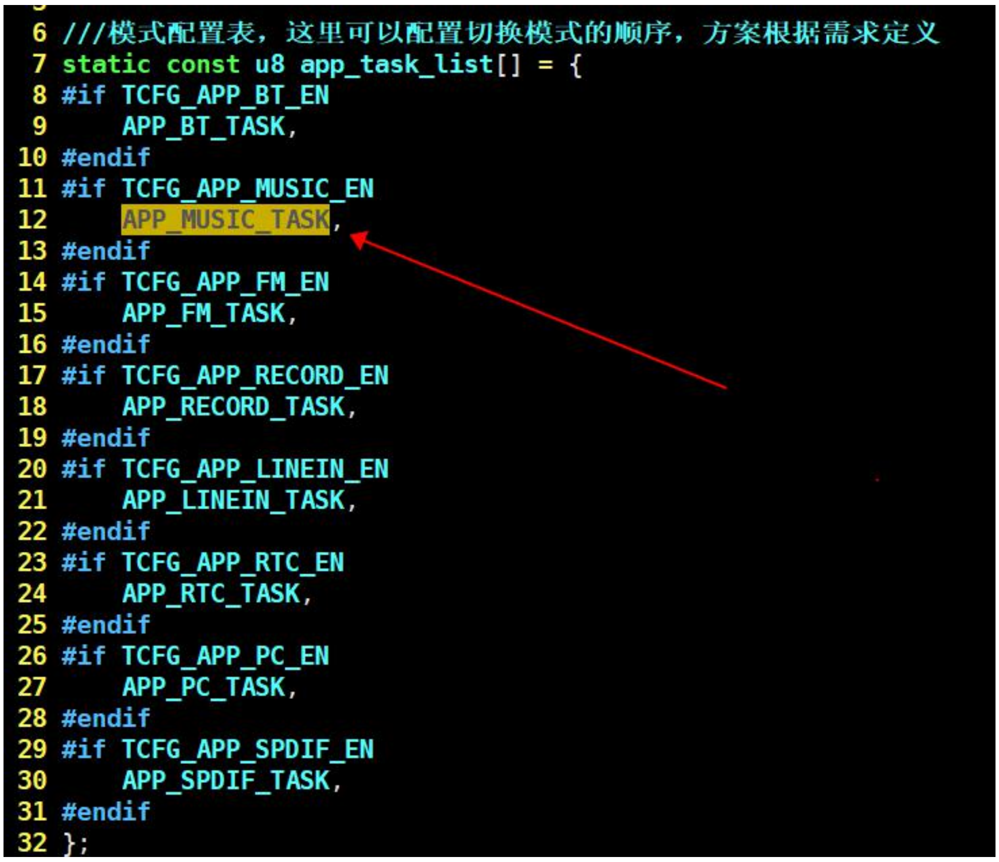
7.Added mode description

(1)existapp_task.hAdd modeid(bymusicFor example)



```
enum {
    APP_POWERON_TASK  = 1,
    APP_POWEROFF_TASK = 2,
    APP_BT_TASK       = 3,
    APP_MUSIC_TASK    = 4,
    APP_FM_TASK       = 5,
    APP_RECORD_TASK   = 6,
    APP_LINEIN_TASK   = 7,
    APP_RTC_TASK      = 8,
    APP_PC_TASK       = 9,
    APP_SPDIF_TASK    = 10,
    APP_IDLE_TASK     = 11,
    APP_TASK_MAX_INDEX,
};
```

(2)Add the new modeidJoinapp_task_switch.cMedium Mode Configuration Tableapp_task_list



```
///模式配置表，这里可以配置切换模式的顺序，方案根据需求定义
static const u8 app_task_list[] = {
#if TCFG_APP_BT_EN
    APP_BT_TASK,
#endif
#if TCFG_APP_MUSIC_EN
    APP_MUSIC_TASK,
#endif
#if TCFG_APP_FM_EN
    APP_FM_TASK,
#endif
#if TCFG_APP_RECORD_EN
    APP_RECORD_TASK,
#endif
#if TCFG_APP_LINEIN_EN
    APP_LINEIN_TASK,
#endif
#if TCFG_APP_RTC_EN
    APP_RTC_TASK,
#endif
#if TCFG_APP_PC_EN
    APP_PC_TASK,
#endif
#if TCFG_APP_SPDIF_EN
    APP_SPDIF_TASK,
#endif
};
```

(3)refer totask_key.cRefer to Add Mode Key Conversion Table (ad,io,irwait)

(4)existtask_managerAdd the corresponding mode directory (and the corresponding header file directory)

(5)Implement mode-related interfaces (refer to existing modes, the followingmusicTake this as an example)

① Implement the following basic necessary interfaces:

void app_music_task()

int music_app_check(void)

static int music_sys_event_handler(struct sys_event *event) static int

music_key_event_opr(struct sys_event *event) The following basic

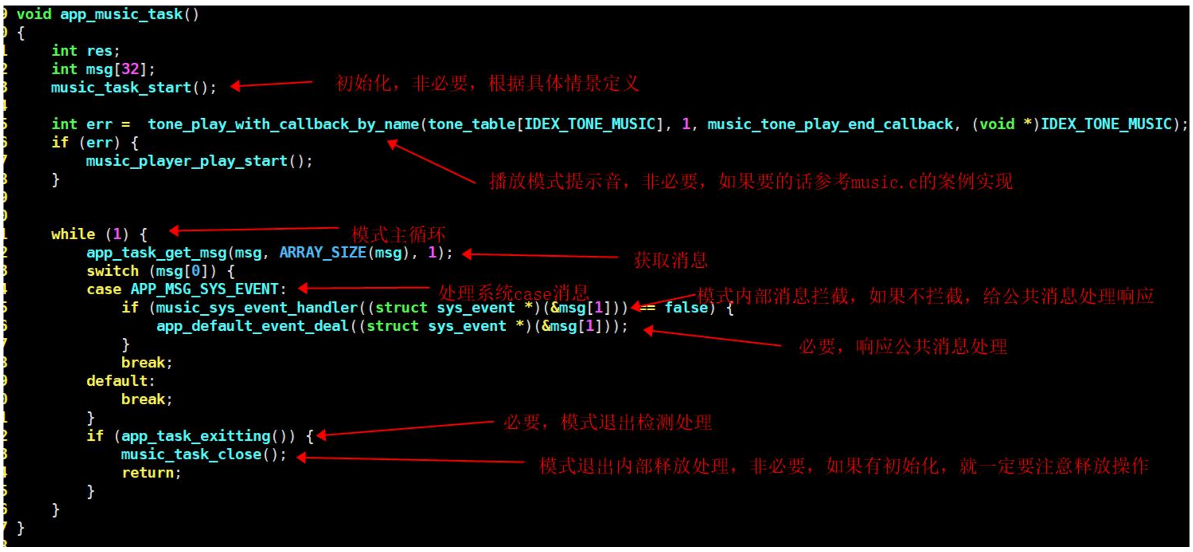
② operations are completed in the main loop of the mode:app_music_task) Get

the message

Responding to messages and events

Response mode internal messages and events

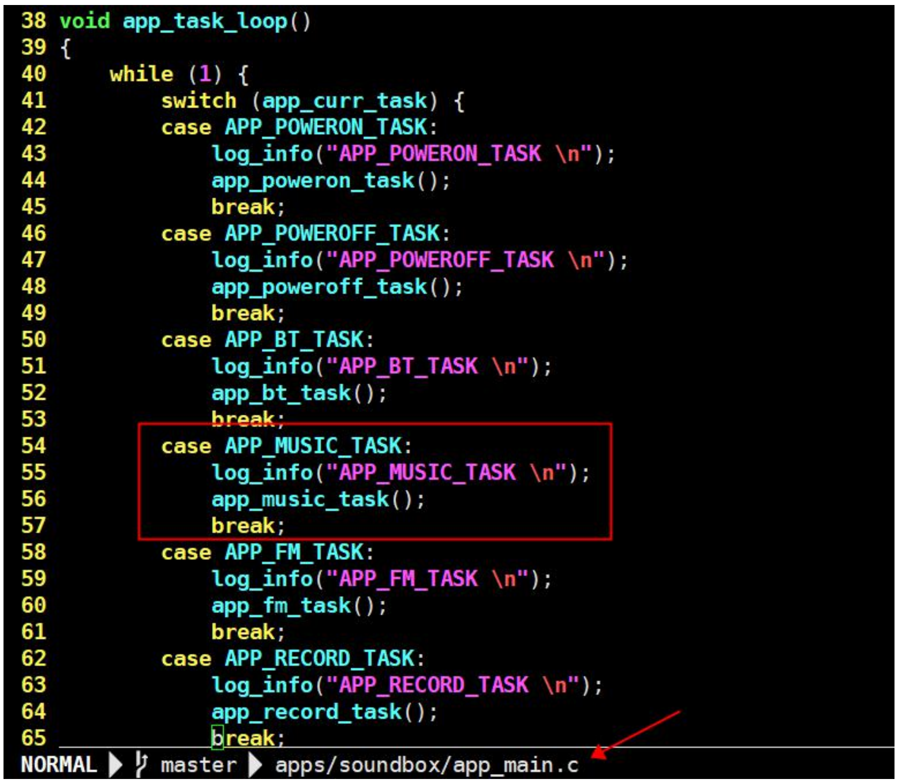
Respond to public messages and events



```
void app_music_task()
{
    int res;
    int msg[32];
    music_task_start();

    int err =  tone_play_with_callback_by_name(tone_table[IDEX_TONE_MUSIC], 1, music_tone_play_end_callback, (void *)IDEX_TONE_MUSIC);
    if (err) {
        music_player_play_start();
    }


    while (1) {
        app_task_get_msg(msg, ARRAY_SIZE(msg), 1);
        switch (msg[0]) {
        case APP_MSG_SYS_EVENT:
            if (music_sys_event_handler((struct sys_event *)(&msg[1])) == false) {
                app_default_event_deal((struct sys_event *)(&msg[1]));
            }
            break;
        default:
            break;
        }
        if (app_task_exitting()) {
            music_task_close();
            return;
        }
    }
}
```

初始化，非必要，根据具体情景定义
播放模式提示音，非必要，如果要的话参考music.c的案例实现
模式主循环
获取消息
处理系统case消息
模式内部消息拦截，如果不拦截，给公共消息处理响应
必要，响应公共消息处理
必要，模式退出检测处理
模式退出内部释放处理，非必要，如果有初始化，就一定要注意释放操作

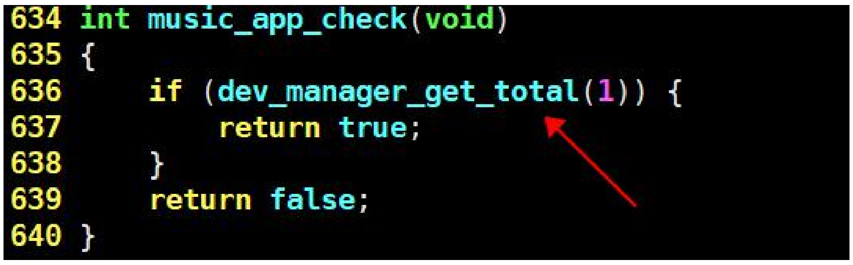
③ existapp_main.cCall the corresponding mode main loop interface (app_music_task)



```
void app_task_loop()
{
    while (1) {
        switch (app_curr_task) {
        case APP_POWERON_TASK:
            log_info("APP_POWERON_TASK \n");
            app_poweron_task();
            break;
        case APP_POWEROFF_TASK:
            log_info("APP_POWEROFF_TASK \n");
            app_poweroff_task();
            break;
        case APP_BT_TASK:
            log_info("APP_BT_TASK \n");
            app_bt_task();
            break;
        case APP_MUSIC_TASK:
            log_info("APP_MUSIC_TASK \n");
            app_music_task();
            break;
        case APP_FM_TASK:
            log_info("APP_FM_TASK \n");
            app_fm_task();
            break;
        case APP_RECORD_TASK:
            log_info("APP_RECORD_TASK \n");
            app_record_task();
            break;
NORMAL  master  apps/soundbox/app_main.c
```

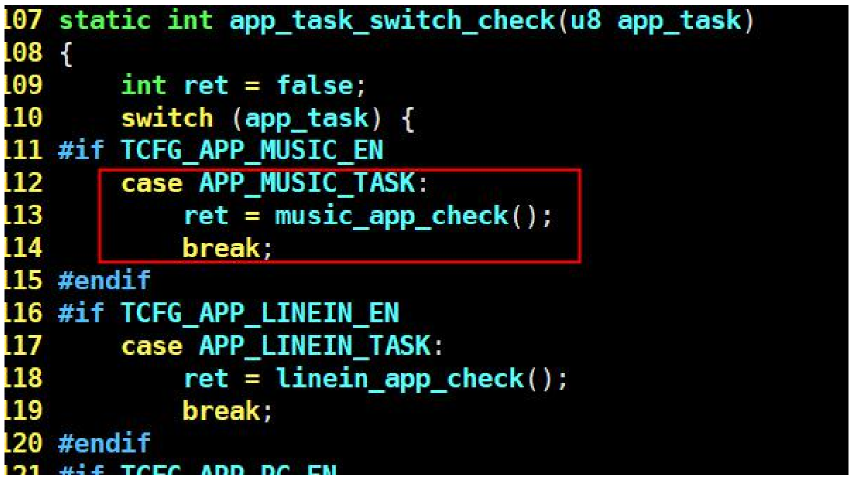
④ app_checkImplementation of the interface (musicFor example)

app_checkIn fact, when switching modes, whether the conditions are met to enter the mode,musicThe mode entry condition is to determine whether there is a device that can play online, so the interface is implemented as follows:



```
int music_app_check(void)
{
    if (dev_manager_get_total(1)) {
        return true;
    }
    return false;
}
```

⑤ existapp_task_switch_checkCallapp_check（musicFor example)



```
static int app_task_switch_check(u8 app_task)
{
    int ret = false;
    switch (app_task) {
#if TCFG_APP_MUSIC_EN
    case APP_MUSIC_TASK:
        ret = music_app_check();
        break;
#endif
#if TCFG_APP_LINEIN_EN
    case APP_LINEIN_TASK:
        ret = linein_app_check();
        break;
#endif
#if TCFG_APP_PC_EN
```