

# ui 接口设计说明书

珠海市杰理科技股份有限公司

**Zhuhai Jieli Technologyco.,LTD**

版权所有，未经许可，禁止外传

## 修改记录

版本	更新日期	描述
V1.0		
V1.1		



## 目录

1. 文档介绍.....	4
1.1. 文档目的.....	4
1.2. 参考文献.....	4
[1].....	4
1.3. 术语与缩写词.....	4
2. 功能概述.....	4
3. 其他系统/模块的调用关系.....	4
4. 接口模块功能介绍.....	5
4.1. 显示主页（led7 与 lcd 屏通用）.....	5
4.1.1. 功能介绍.....	5
4.1.2. 接口介绍.....	5
4.2. 关闭主页（led7 与 lcd 屏通用）.....	6
4.2.1. 实现介绍.....	6
4.2.2. 接口介绍.....	6
4.3. 获取当前主页、子页面.....	7
4.3.1. 功能介绍.....	7
4.3.2. 接口.....	7
4.4. 显示子页面（led7 特有）.....	7
4.4.1. 功能介绍.....	7
4.4.2. 接口.....	8
4.5. 刷新主页（led7 特有）.....	8
4.5.1. 功能介绍.....	8
4.5.2. 接口介绍.....	8
4.6. 消息交互（lcd 屏特有）.....	9
4.6.1. 功能介绍.....	9
4.6.2. 接口介绍.....	9
4.7. 模块初始化（led7、lcd 通用）.....	11
4.7.1. 功能介绍.....	11
4.7.2. 接口介绍.....	11
4.8. Ui 框架消息处理.....	11
4.8.1. Lcd 屏架构消息处理.....	11
4.8.2. Led7 架构消息处理.....	13
4.9. 板卡配置.....	16
4.9.1. Ui 风格配置.....	16
4.9.2. Led7 Ui 驱动配置.....	16

## 1. 文档介绍

### 1.1. 文档目的

在每个模式中，我们都可以添加相应的 UI 显示来达到与用户交互的目的，本文介绍 UI 的使用与开发。

### 1.2. 参考文献

[1].

### 1.3. 术语与缩写词

缩写、术语	解 释
AP	Application, 应用程序

## 2. 功能概述

本模块讲述 ui 接口的说明。

## 3. 其他系统/模块的调用关系

Ui 主页打开接口在 app\_core 调用，然后 ui 内部的响应函数均在 ui 的任务内响应。

## 4. 接口模块功能介绍

### 4.1. 显示主页（led7 与 lcd 屏通用）

#### 4.1.1. 功能介绍

显示 ui 界面的主页

#### 4.1.2. 接口介绍

函数原型	UI_SHOW_WINDOW(a)
功能描述	显示 ui 界面的主页
参数说明	<p>参数：主页 id</p> <p>Led7 id 由用户自行定义</p> <p>Lcd 屏 id 由工具自行生成</p> <p>如：</p> <pre>#if(CONFIG_UI_STYLE == STYLE_JL_SOUNDBOX) #include "ui/style_jl02.h"//点阵// #define ID_WINDOW_MAIN      PAGE_0 #define ID_WINDOW_BT        PAGE_1 #define ID_WINDOW_FM        PAGE_2 #define ID_WINDOW_CLOCK     PAGE_3 #define ID_WINDOW_MUSIC     PAGE_4 #define ID_WINDOW_LINEIN    PAGE_0 #define ID_WINDOW_POWER_ON  PAGE_5 #define ID_WINDOW_POWER_OFF PAGE_6 #define ID_WINDOW_SYS       PAGE_7 #endif  #if(CONFIG_UI_STYLE == STYLE_JL_LED7)//led7 显示 #define ID_WINDOW_BT        UI_BT_MENU_MAIN #define ID_WINDOW_FM        UI_FM_MENU_MAIN #define ID_WINDOW_CLOCK     UI_RTC_MENU_MAIN #define ID_WINDOW_MUSIC     UI_MUSIC_MENU_MAIN #define ID_WINDOW_LINEIN    UI_AUX_MENU_MAIN #define ID_WINDOW_PC        UI_PC_MENU_MAIN #define ID_WINDOW_REC       UI_RECORD_MENU_MAIN</pre>

	<pre>#define ID_WINDOW_POWER_ON      UI_IDLE_MENU_MAIN #define ID_WINDOW_POWER_OFF     UI_IDLE_MENU_MAIN #define ID_WINDOW_SPDIF         UI_IDLE_MENU_MAIN #define ID_WINDOW_IDLE          UI_IDLE_MENU_MAIN #endif</pre>
输出	无
例子	UI_SHOW_WINDOW(ID_WINDOW_BT);

## 4.2. 关闭主页（led7 与 lcd 屏通用）

### 4.2.1. 实现介绍

根据主页 id 关闭 ui 主页

### 4.2.2. 接口介绍

#### 4.2.2.1. 根据 id 关闭主页

函数原型	UI_HIDE_WINDOW(a)
功能描述	根据 id 关闭主页
参数说明	参数：主页 id
输出	无
例子	UI_HIDE_WINDOW(ID_WINDOW_BT);

#### 4.2.2.2. 关闭当前主页

函数原型	UI_HIDE_CURR_WINDOW ()
功能描述	自动根据当前主页进行关闭，不需要用户选择 id
参数说明	无
输出	无
关联模块	ui_api.h, led7_ui_api.c, lcd_api.c
例子	UI_HIDE_CURR_WINDOW()
补充说明	推荐使用这个接口关闭 ui

### 4.3. 获取当前主页、子页面

#### 4.3.1. 功能介绍

该功能主要是让用户可以获取当前所打开的主页、子页面

#### 4.3.2. 接口

##### 4.3.2.1. 获取当前主页

函数原型	UI_GET_WINDOW_ID()
功能描述	获取当前主页
参数说明	无
输出	主页 id
关联模块	ui_api.h, led7_ui_api.c, lcd_api.c
例子	U16 id = UI_GET_WINDOW_ID();
补充说明	

##### 4.3.2.2. 获取当前子页面（led7 特有）

函数原型	UI_GET_CURR_MENU()
功能描述	获取当前子页面
参数说明	无
输出	子页面 id
关联模块	ui_api.h, led7_ui_api.c, lcd_api.c
例子	U16 id = UI_GET_CURR_MENU();
补充说明	

### 4.4. 显示子页面（led7 特有）

#### 4.4.1. 功能介绍

该接口可以在主页上临时显示子页面。

#### 4.4.2. 接口

函数原型	UI_SHOW_MENU (...) 或者 void ui_set_tmp_menu(u8 app_menu, u16 ret_time, s32 arg, void (*timeout_cb)(u8 menu))
功能描述	显示子页面
参数说明	app_menu: 对应子界面 ret_time: 持续时间 arg: 显示参数 menu: 返回的界面
输出	无
关联模块	ui_api.h, led7_ui_api.c
例子	UI_SHOW_MENU(MENU_MAIN_VOL,1000,app_audio_get_volume(APP_AUDIO_CURRENT_STATE), NULL);
补充说明	该接口应用子页面显示。case 在注册的 ui_user 函数进行处理，如没有对应的 case 则在 ui_common 函数进行处理。使用事例，例如在模式界面显示搜索到的文件数目，可以使用该接口。持续时间到，自动回到主页。如果持续时间 0，则不会回到主页，只有新的子界面显示完毕或者调用了刷新主页的接口。如果注册了 timeout_cb，则，持续时间到了或者被打断了有该回调。

#### 4.5. 刷新主页（led7 特有）

##### 4.5.1. 功能介绍

可以调用这个函数可以刷新主页，如果是子界面显示状态，参数存入打断显示，可以直接打断子界面显示，恢复主页显示。

##### 4.5.2. 接口介绍

###### 4.5.2.1. 刷新主页

函数原型	UI_REFLASH_WINDOW(a)
功能描述	打开资源文件
参数说明	参数 1:打断显示 0: 不打断显示 // 打断显示可以理解为当前正在显示临时页面并非主页，例如设置页面，打断显示可以打断当前显示，恢复主页刷新 // 实例化函数: ui_menu_reflash_action
输出	无



关联模块	ui_api.h, led7_ui_api.c
例子	UI_REFLASH_WINDOW(true);
补充说明	可以调用这个函数可以刷新主页，如果是子界面显示状态，参数存入打断显示，可以直接打断子界面显示，恢复主页显示。例如子界面是显示加载字符，加载成功后，可以调用该接口恢复主页。

#### 4.5.2.2. 定时刷新主页

函数原型	void ui_set_auto_reflash(u32 msec)
功能描述	定时刷新主页
参数说明	msec: 刷新闻隔（毫秒）
输出	* \return 无
关联模块	ui_api.h, led7_ui_api.c
例子	ui_set_auto_reflash(500);
补充说明	该接口可以在 open 时候调用，设置主页的自动刷新时间。例如音乐模式设置了 500ms 自动刷新主页，可以在主页进行时间刷新

### 4.6. 消息交互（lcd 屏特有）

#### 4.6.1. 功能介绍

本接口用于 lcd 屏 ui 架构的触摸消息和 key 消息交互。

#### 4.6.2. 接口介绍

##### 4.6.2.1. Key 消息

函数原型	int ui_key_msg_post(int key)
功能描述	往 ui 发送 key 消息
参数说明	Key 按键消息
输出	0: 成功 非 0: 失败
关联模块	lcd_api.c
例子	ui_key_msg_post (KEY_OK) ;
补充说明	

#### 4.6.2.2. Touch 消息

函数原型	int ui_touch_msg_post(struct touch_event *event)
功能描述	往 ui 发送触摸消息
参数说明	触摸消息结构体 <pre>struct touch_event {     int event;//消息类型     int x;//坐标     int y;//坐标 };</pre>
输出	0: 成功 非 0: 失败
关联模块	lcd_api.c
例子	<pre>t.event = ELM_EVENT_TOUCH_UP; t.x = x; t.y = y; ui_touch_msg_post(&amp;t);</pre>
补充说明	

#### 4.6.2.3. 应用层往 ui 发送消息

函数原型	int ui_server_msg_post(const char *msg, ...), UI_MSG_POST(...)
功能描述	往 ui 发送消息
参数说明	应用往 ui 发送消息，ui 主页需要注册回调函数关闭当前主页消息发送方 demo <pre>UI_MSG_POST("test1:a=%4,test2:bcd=%4,test3:efgh=%4,test4:hijkl=%4", 1,2,3,4);</pre> 往 test1、test2、test3、test4 发送了字符为 a、bcd、efgh、hijkl，附带变量为 1、2、3、4 每次可以只发一个消息，也可以不带数据例如:UI_MSG_POST("test1")
输出	0: 成功 非 0: 失败
关联模块	lcd_api.c
例子	<pre>UI_MSG_POST("test1:a=%4,test2:bcd=%4,test3:efgh=%4,test4:hijkl=%4", 1,2,3,4); UI_MSG_POST("test1")</pre>
补充说明	

## 4.7. 模块初始化（led7、lcd 通用）

### 4.7.1. 功能介绍

该部分是 ui 模块初始化

### 4.7.2. 接口介绍

函数原型	UI_INIT(a)
功能描述	初始化
参数说明	显示的 io 句柄
输出	* \return 无
关联模块	ui_api.h, led7_ui_api.c, lcd_api.c
例子	UI_INIT(&ui_cfg_data);
补充说明	根据选择的类型 来初始化 led7_ui_init 或者 lcd_ui_init(a)

## 4.8. Ui 框架消息处理

### 4.8.1. Lcd 屏架构消息处理

#### 4.8.1.1. lcd 屏架构消息处理接口

函数原型	#define REGISTER_UI_EVENT_HANDLER(id) \ __REGISTER_UI_EVENT_HANDLER(STYLE_NAME, id)
功能描述	注册消息响应函数
参数说明	* \param[in] ID //显示的控制件
输出	* \return * \retval TRUE 成功 * \retval FALSE 失败
关联模块	
补充说明	

#### 4.8.1.2. lcd 屏架构应用层消息处理接口

函数原型	int ui_register_msg_handler(int id, const struct uimsg_handl *handler)
功能描述	lcd 屏架构应用层消息处理接口
参数说明	* \param[in] ID //主页 id Handler //注册的消息响应列表
输出	* \return * \retval TRUE 成功 * \retval FALSE 失败
关联模块	
例子	<pre> static int bt_list_handler(const char *type, u32 arg) {     return 0; }  static const struct uimsg_handl ui_msg_handler[] = {     {"bt_list",      bt_list_handler      }, /* 设置音量 */     { NULL, NULL},      /* 必须以此结尾! */ };  static int bt_menu_onchange(void *ctr, enum element_change_event e, void *arg) {     struct window *window = (struct window *)ctr;     static int id = 0;     switch (e) {         case ON_CHANGE_INIT:             puts("\n***bt menu onchange init***\n");             key_ui_takeover(1);             ui_register_msg_handler(ID_WINDOW_BT_MENU,             ui_msg_handler);             /*      ui_register_msg_handler(ID_WINDOW_VIDEO_REC,             rec_msg_handler); */             /*              * 注册 APP 消息响应              */             bt_menu_status = 1; </pre>

```
        break;
    case ON_CHANGE_RELEASE:
        bt_menu_status = 0;
        /* ui_hide(ID_WINDOW_VIDEO_SYS); */
        /*
         * 要隐藏一下系统菜单页面，防止在系统菜单插入 USB 进入
         * USB 页面
         */
        break;
    default:
        return false;
    }
    return false;
}
```

## 4.8.2. Led7 架构消息处理

### 4.8.2.1. struct ui\_dis\_api

```
120     return ret;
121
122 }
123
124 const struct ui_dis_api bt_main = {
125     .ui      = UI_BT_MENU_MAIN,
126     .open    = ui_open_bt,
127     .ui_main = ui_bt_main,
128     .ui_user = ui_bt_user,
129     .close   = ui_close_bt,
130 };
131
132
```

说明：

- 1、UI 界面枚举
- 2、初始化/开启函数
- 3、主界面显示
- 4、子界面显示
- 5、退出/关闭函数

每个模式需要增加一个类似的结构体进行注册，注册在 ui\_dis\_main 数组里

```
120     return ret;
121
122 }
123
124 const struct ui_dis_api bt_main = {
125     .ui      = UI_BT_MENU_MAIN,
126     .open    = ui_open_bt,
127     .ui_main = ui_bt_main,
128     .ui_user = ui_bt_user,
129     .close   = ui_close_bt,
130 };
131
132
```

#### 4.8.2.2. Led 7 UI 固定结构（以 MUSIC 模式为例）

##### 1、UI 界面枚举

```
.ui      = UI_MUSIC_MENU_MAIN
```

说明：表示该界面的名字，在 ui\_api.h 中加入即可

```
enum ui_menu_main {
    UI_MENU_MAIN_NULL = 0,
    UI_RTC_MENU_MAIN ,
    UI_MUSIC_MENU_MAIN,
    UI_AUX_MENU_MAIN,
    UI_BT_MENU_MAIN,
    UI_RECORD_MENU_MAIN,
    UI_FM_MENU_MAIN,
    UI_PC_MENU_MAIN,
};
```

在 music.c 中进入 app 时调用

```
ui_set_main_menu(UI_MUSIC_MENU_MAIN); //ui_set_main_menu 函数被宏封装定义
```

##### 2.UI 初始化/开启函数

```
.init    = ui_open_music
```

```
static void *ui_open_music(void )
```

说明：返回 music 模式的私有数据结构体（设备、文件数、当前文件序号等），如没有可以返回 NULL。

### 3.UI 主界面显示

```
.ui_main = ui_music_main
```

```
static void ui_music_main(void *hd,void *private)
```

说明：传入 UI 数据和模式私有数据结构体，调用显示字符串函数，达到显示目的。

如 ui\_led7\_show\_music\_time(hd,sencond) 显示当前音乐播放时间。hd 为显示显示的驱动句柄。

### 4.UI 子界面显示

```
.ui_user = ui_music_user
```

```
static int ui_music_user(void *hd ,void *private, u8 menu ,u32 arg)
```

说明：传入 UI 数据、模式私有数据、界面按键信息、显示参数，处理 menu 选择对应功能，返回 **true** 不继续传递，返回 **false** 由 common 统一处理。

### 5.UI 子界面显示

```
.uninit = ui_close_fm
```

```
static void ui_close_fm(void *hd , void *private)
```

说明：传入 UI 数据和私有数据，ui 数据清除，释放模式私有数据内存。

### 6.显示字符串

举例：

```
static void led7_show_pause(void *hd)
{
    UI_DIS_VAR *ui_dis_var = (UI_DIS_VAR*)hd;
    ui_dis_var->dis->clear();
    ui_dis_var->dis->setXY(0,0);
    ui_dis_var->dis->show_string((u8 *)" PAU");
}
```

说明：传入 UI 数据，清除显示，设置显示坐标，调用接口把要显示的 PAU 传进去。

## 4.9. 板卡配置

### 4.9.1. UI 风格配置

#### 1、UI 配置

```
1. #define TCFG_UI_ENABLE                ENABLE_THIS_MOUDLE //UI 总开关
2. #define CONFIG_UI_STYLE                STYLE_JL_LED7    //选择 ui 风格
3. #define TCFG_UI_LED7_ENABLE            ENABLE_THIS_MOUDLE //UI 使用 LED7 显示
4. // #define TCFG_LCD_ST7789VW_ENABLE    ENABLE_THIS_MOUDLE
5. #define TCFG_SPI_LCD_ENABLE            DISABLE_THIS_MOUDLE //spi lcd 开关
6. #define TCFG_TFT_LCD_DEV_SPI_HW_NUM    1// 1: SPI1    2: SPI2 配置 lcd 选择的 spi 口
```

说明：

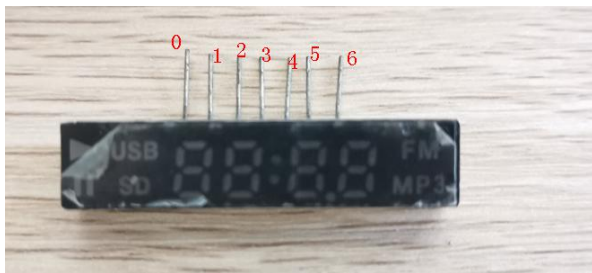
使能 UI 总开关后，可以根据需要打开或添加对应屏幕的宏开关，此处选用 LED7。因此打开了 TCFG\_UI\_LED7\_ENABLE，为了和彩屏风格进行统一 api，led7 选择了 STYLE\_JL\_LED7 风格。

### 4.9.2. Led7 UI 驱动配置

```
1. #if TCFG_UI_LED7_ENABLE
2. LED7_PLATFORM_DATA_BEGIN(led7_data)
3.     .pin_type = LED7_PIN7, //配置 led7pin 类型，可以拓展为 pin12、pin13 等
4.     .pin_cfg.pin7.pin[0] = IO_PORTC_01,
5.     .pin_cfg.pin7.pin[1] = IO_PORTC_02,
6.     .pin_cfg.pin7.pin[2] = IO_PORTC_03,
7.     .pin_cfg.pin7.pin[3] = IO_PORTC_04,
8.     .pin_cfg.pin7.pin[4] = IO_PORTC_05,
9.     .pin_cfg.pin7.pin[5] = IO_PORTB_06,
10.    .pin_cfg.pin7.pin[6] = IO_PORTB_07,
11. LED7_PLATFORM_DATA_END()
12.
13. const struct ui_devices_cfg ui_cfg_data = {
14.     .type = LED_7,
15.     .private_data = (void *)&led7_data,
16. };
17. #endif /* #if TCFG_UI_LED7_ENABLE */
```

注意：sdk io 的 0 下标从 led7 的左边 io 往右排如下图：





led7 的 pin 可以进行拓展为 pin12、pin13 等，如图：

```
45
46 struct led7_pin7 {
47     u8 pin[7];
48 };
49
50 struct led7_pin12 {
51     u8 pin_comh[5];
52     u8 pin_seg1[7];
53 };
54
55 struct led7_pin13 {
56     u8 pin_com[6];
57     u8 pin_seg[7];
58 };
59
60
61 union led7_pin_cfg {
62     struct led7_pin7 pin7;
63     struct led7_pin12 pin12;
64     struct led7_pin13 pin13;
65 };
```

io 配置的结构体也需要同样的增加，如图：

```
29
30 typedef enum _led7_pin_type {
31     LED7_PIN7, //7个引脚
32     LED7_PIN12, //12个引脚
33     LED7_PIN13, //13个引脚
34 } LED7_PIN_TYPE;
35
```

```
517
518 static LCD_API LED7_HW = {
519     .clear          = led7_show_null,
520     .setXY          = led7_setXY,
521     .FlashChar      = led7_Flashchar,
522     .Clear_FlashChar = led7_Clear_Flashchar,
523     .show_icon      = led7_show_icon,
524     .flash_icon     = led7_flash_icon,
525     .clear_icon     = led7_clear_icon,
526     .show_string    = led7_show_string,
527     .show_char      = led7_show_char,
528     .show_number    = led7_show_one_number,
529     .show_pic       = led7_show_pic,
530     .hide_pic       = led7_hide_pic,
531     .lock           = led7_show_lock,
532 };
533
```

sdk 对驱动进行了抽象接口，统一抽象出以下接口供 ui 统一使用，分别为 clean、设置起点、闪烁、显示图标、显示字符、显示字符串等通用接口，用户可以进行了拓展和组合使用。