

RK3308 Led Interface Introduction

Document ID: RK-KF-YF-319

Release Version: V1.0.1

Date: 2019-03-29

Security Level: ☐Top-Secret ☐Secret ☐Internal ☒Public

DISCLAIMER

THIS DOCUMENT IS PROVIDED "AS IS". FUZHOU ROCKCHIP ELECTRONICS CO., LTD. ("ROCKCHIP") DOES NOT PROVIDE ANY WARRANTY OF ANY KIND, EXPRESSED, IMPLIED OR OTHERWISE, WITH RESPECT TO THE ACCURACY, RELIABILITY, COMPLETENESS, MERCHANTABILITY, FITNESS FOR ANY PARTICULAR PURPOSE OR NON-INFRINGEMENT OF ANY REPRESENTATION, INFORMATION AND CONTENT IN THIS DOCUMENT. THIS DOCUMENT IS FOR REFERENCE ONLY. THIS DOCUMENT MAY BE UPDATED OR CHANGED WITHOUT ANY NOTICE AT ANY TIME DUE TO THE UPGRADES OF THE PRODUCT OR ANY OTHER REASONS.

Trademark Statement

"Rockchip", "瑞芯微", "瑞芯" shall be Rockchip's registered trademarks and owned by Rockchip. All the other trademarks or registered trademarks mentioned in this document shall be owned by their respective owners.

All rights reserved. ©2019. Fuzhou Rockchip Electronics Co., Ltd.

Beyond the scope of fair use, neither any entity nor individual shall extract, copy, or distribute this document in any form in whole or in part without the written approval of Rockchip.

Fuzhou Rockchip Electronics Co., Ltd.

No.18 Building, A District, No.89, software Boulevard Fuzhou, Fujian, PRC

Website: www.rock-chips.com

Customer service Tel: +86-4007-700-590

Customer service Fax: +86-591-83951833

Customer service e-Mail: fae@rock-chips.com

Preface

Overview

This document describes the interfaces in RK3308 DeviceIo library.

Chipset

RK3308

Intended Audience

This document (this guide) is mainly intended for:

Technical support engineers Software development engineers

Revision History

Date	Revision No.	Author	Revision History
2019-3-29	V1.0.0	Jacky Ge	Initial version
2020-03-02	V1.0.1	Ruby Zhang	Update the format and the name of the document

Contents

RK3308 Led Interface Introduction

- Preface
 - Contents
 - 1. Overview
 - 2. Interface Introduction
 - 3. Application Example
-

1. Overview

This code module is integrated in the libDeviceIo.so dynamic library based on a single RGB Led driven by PWM and has packaged interfaces such as LED on and off, flashing and breathing light effects. The layered design meets requirement of different application cases, supports the priority setting of lighting effects, and builds complex lighting effect based on current interfaces.

The whole framework is divided into three layers: TEMP, REALTIME, and STABLE.

TEMP: contains only a single light effect, with the highest priority. It can be used to handle short-time light effects such as key indicator lights.

REALTIME: contains only a single light effect, and its priority is lower than TEMP. It can be used to handle LED status switching in the whole transaction processes, such as the status switching of recording, recognize and response of smart speakers.

STABLE: including a light effect stack that supports priority setting. The light effect at the top of the stack is always taken, and the priority is lower to REALTIME. It can be used to handle the status of the device, such as low battery, static MIC mode, and network setting mode.

In conclusion, if there is an element in TEMP layer, the TEMP layer element is always displayed; otherwise, it will check whether there is an element in REALTIME layer, and if there is an element in REALTIME layer, the top element of the STABLE layer is displayed. It is going to wait if STABLE layer stack is empty.

2. Interface Introduction

- `RK_Led_Effect_layer_e`

The enumeration type of the "effect layer", including TEMP, REALTIME, and STABLE layers. Need to be specified when setting light effect.

```
1 typedef enum RK_Led_Effect_layer {
2     Led_Effect_layer_TEMP = 0,
3     Led_Effect_layer_STABLE,
4     Led_Effect_layer_REALTIME
5 } RK_Led_Effect_layer_e;
```

- `RK_Led_Effect_type`

The structure type of the "effect type", including NONE, BLINK and BREATH light effect effects. Need to be specified when setting light effect.

```
1 typedef enum RK_Led_Effect_type {
2     Led_Effect_type_NONE = 0,
3     Led_Effect_type_BLINK,
4     Led_Effect_type_BREATH
5 } RK_Led_Effect_type_e;
```

- `RK_Led_Effect`

The structure type of light effect, need to be assigned structure parameters when setting the light effect.

```
1 typedef struct RK_Led_Effect {
2     int period; // Lighting effect period, for
3     // example, one breath is 3000ms. <= 0 means the period is infinite
4     int timeout; // Timeout length, <= 0 means
5     // infinite
6     int colors; // The RGB value that the lighting
7     // effect needs to display, such as 0xFFFFFF
8     int colors_blink; // linking light effect, no need to
9     // set other light effects
10    int priority; // Priority of light effect
11    char name[64]; // The name of light effect
12    RK_Led_Effect_type_e type; // Type of light effect
13    RK_Led_Effect_layer_e layer; // layer of light effect
14 } RK_Led_Effect_type_e;
```

- `int RK_led_init(void)`

Led module initialization, to initialize related parameters.

- `int RK_set_all_led_status(const int Rval, const int Gval, const int Bval)`

Set the basic interface of Led light. The assigned parameter is the corresponding RGB values (0x00-0xFF).

- `int RK_set_all_led_off(void)`

Close the LED basic interface.

- `int RK_set_led_effect(RK_Led_Effect *effect)`

Set LED light effect, the parameter is effect structure.

- `int RK_set_led_effect_off(const RK_Led_Effect_layer_e layer, const char *name)`

Turn off the light effect with the specified name at the specified level. (If you turn off the current light effect, the previous light effect will be displayed automatically).

- `int RK_set_all_led_effect_off(void)`

Clear all set effects and turn off LED light.

- `int RK_led_exit(void)`

Led module de-initialization, release resources.

3. Application Example

```
1  #include <stdio.h>
2  #include <string.h>
3  #include <unistd.h>
4  #include <DeviceIo/Rk_led.h>
5
6  static void rk_led_effect_default(RK_Led_Effect_t *effect)
7  {
8      effect->period = -1;
9      effect->timeout = -1;
10     memset(effect->name, 0, sizeof(effect->name));
11     effect->layer = Led_Effect_layer_TEMP;
12     effect->colors = 0;
13     effect->colors_blink = 0;
14     effect->priority = 0;
15 }
16
17 static int remove_layer(const RK_Led_Effect_layer_e layer, const char
18 *name)
19 {
20     if (!name || strlen(name) == 0) {
21         if (Led_Effect_layer_STABLE == layer) {
22             return -1;
23         } else {
24             RK_set_led_effect_off(layer, "");
25             return 0;
26         }
27     }
28     RK_set_led_effect_off(layer, name);
29     return 0;
30 }
31
32 //Red Led breathing light on STABLE level with a period of 1000ms
33 int stable_breath_red(const char *name)
34 {
35     if (name == NULL)
36         return -1;
37
38     RK_Led_Effect_t effect;
39     rk_led_effect_default(&effect);
40
41     effect.colors = 0xFF0000;
42     effect.period = 1000;
43     effect.type = Led_Effect_type_BREATH;
44     effect.layer = Led_Effect_layer_STABLE;
45     strncpy(effect.name, name, sizeof(effect.name));
46
47     RK_set_led_effect(&effect);
48     return 0;
49 }
50
51 // Red Led flashing light on STABLE layer, with a period of 1000ms
52 int stable_blink_red(const char *name)
53 {
```

```

54     if (name == NULL)
55         return -1;
56
57     RK_Led_Effect_t effect;
58     rk_led_effect_default(&effect);
59
60     effect.colors = 0xFF0000;
61     effect.period = 1000;
62     effect.type = Led_Effect_type_BLINK;
63     effect.layer = Led_Effect_layer_STABLE;
64     strncpy(effect.name, name, sizeof(effect.name));
65
66     RK_set_led_effect(&effect);
67     return 0;
68 }
69
70 //Green Led flashing light on REALTIME layer, with a period of 1000ms
71 int realtime_blink_green(void)
72 {
73     RK_Led_Effect_t effect;
74     rk_led_effect_default(&effect);
75
76     effect.colors = 0x00FF00;
77     effect.period = 1000;
78     effect.type = Led_Effect_type_BLINK;
79     effect.layer = Led_Effect_layer_REALTIME;
80
81     RK_set_led_effect(&effect);
82     return 0;
83 }
84
85 // White Led lights on TEMP layer
86 int temp_none_white(void)
87 {
88     RK_Led_Effect_t effect;
89     rk_led_effect_default(&effect);
90
91     effect.colors = 0xFFFFFF;
92     effect.type = Led_Effect_type_NONE;
93     effect.layer = Led_Effect_layer_TEMP;
94
95     RK_set_led_effect(&effect);
96     return 0;
97 }
98
99 int main(int argc, char **argv)
100 {
101     RK_led_init();
102     // Reset Led state
103     RK_set_all_led_effect_off();
104
105     //Display red LED breathing light effect
106     stable_breath_red("stable_breath_red");
107     sleep(10);
108
109     //Display red flashing light effect
110     stable_blink_red("stable_blink_red");
111     sleep(10);

```



```
112
113     // Remove the red flashing light effect and automatically display the
previous light effect, that is, the red breathing light effect
114     remove_layer(Led_Effect_layer_STABLE, "stable_blink_red");
115     sleep(10);
116
117     // Show green flashing light effect on REALTIME layer
118     realtime_blink_green();
119     sleep(10);
120
121     // Always display white on the TEMP layer
122     temp_none_white();
123     sleep(10);
124
125     // For there are elements on the TEMP layer, it still display white on
the TEMP layer.
126     realtime_blink_green();
127     sleep(10);
128
129     // Remove white light effect of the TEMP layer and automatically
display green flashing light on the REALTIME layer
130     remove_layer(Led_Effect_layer_TEMP, "");
131     sleep(10);
132
133     // Remove light effect of the REALTIME layer , automatically display
red breathing light effect of the STABLE
134     remove_layer(Led_Effect_layer_REALTIME, "");
135     sleep(10);
136
137     // Clear all lighting effects and turn off LED light
138     RK_set_all_led_effect_off();
139
140     for (;;)
141     RK_led_exit();
142
143     return 0;
144 }
145
```