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

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

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.

```
typedef enum RK_Led_Effect_layer {
    Led_Effect_layer_TEMP = 0,
    Led_Effect_layer_STABLE,
    Led_Effect_layer_REALTIME
} 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.

```
typedef enum RK_Led_Effect_type {
    Led_Effect_type_NONE = 0,
    Led_Effect_type_BLINK,
    Led_Effect_type_BREATH
} 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.

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

```
#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <DeviceIo/Rk_led.h>

static void rk_led_effect_default(RK_Led_Effect_t *effect)
{
    effect->period = -1;
    effect->timeout = -1;
    memset(effect->name, 0, sizeof(effect->name));
```

```
    effect->layer = Led_Effect_layer_TEMP;
    effect->colors = 0;
    effect->colors_blink = 0;
    effect->priority = 0;
}

static int remove_layer(const RK_Led_Effect_layer_e layer, const char *name)
{
    if (!name || strlen(name) == 0) {
        if (Led_Effect_layer_STABLE == layer) {
            return -1;
        } else {
            RK_set_led_effect_off(layer, "");
            return 0;
        }
    }

    RK_set_led_effect_off(layer, name);
    return 0;
}

//Red Led breathing light on STABLE level with a period of 1000ms
int stable_breath_red(const char *name)
{
    if (name == NULL)
        return -1;

    RK_Led_Effect_t effect;
    rk_led_effect_default(&effect);

    effect.colors = 0xFF0000;
    effect.period = 1000;
    effect.type = Led_Effect_type_BREATH;
    effect.layer = Led_Effect_layer_STABLE;
    strncpy(effect.name, name, sizeof(effect.name));

    RK_set_led_effect(&effect);
    return 0;
}

// Red Led flashing light on STABLE layer, with a period of 1000ms
int stable_blink_red(const char *name)
{
    if (name == NULL)
        return -1;

    RK_Led_Effect_t effect;
    rk_led_effect_default(&effect);

    effect.colors = 0xFF0000;
    effect.period = 1000;
    effect.type = Led_Effect_type_BLINK;
    effect.layer = Led_Effect_layer_STABLE;
    strncpy(effect.name, name, sizeof(effect.name));

    RK_set_led_effect(&effect);
    return 0;
}
```

```
//Green Led flashing light on REALTIME layer, with a period of 1000ms
int realtime_blink_green(void)
{
    RK_Led_Effect_t effect;
    rk_led_effect_default(&effect);

    effect.colors = 0x00FF00;
    effect.period = 1000;
    effect.type = Led_Effect_type_BLINK;
    effect.layer = Led_Effect_layer_REALTIME;

    RK_set_led_effect(&effect);
    return 0;
}

// White Led lights on TEMP layer
int temp_none_white(void)
{
    RK_Led_Effect_t effect;
    rk_led_effect_default(&effect);

    effect.colors = 0xFFFFFF;
    effect.type = Led_Effect_type_NONE;
    effect.layer = Led_Effect_layer_TEMP;

    RK_set_led_effect(&effect);
    return 0;
}

int main(int argc, char **argv)
{
    RK_led_init();
    // Reset Led state
    RK_set_all_led_effect_off();

    //Display red LED breathing light effect
    stable_breath_red("stable_breath_red");
    sleep(10);

    //Display red flashing light effect
    stable_blink_red("stable_blink_red");
    sleep(10);

    // Remove the red flashing light effect and automatically display the
previous light effect, that is, the red breathing light effect
    remove_layer(Led_Effect_layer_STABLE, "stable_blink_red");
    sleep(10);

    // Show green flashing light effect on REALTIME layer
    realtime_blink_green();
    sleep(10);

    // Always display wihte on the TEMP layer
    temp_none_white();
    sleep(10);
```

```
    // For there are elements on the TEMP layer, it still display white on the
TEMP layer.
    realtime_blink_green();
    sleep(10);

    // Remove white light effect of the TEMP layer and automatically display
green flashing light on the REALTIME layer
    remove_layer(Led_Effect_layer_TEMP, "");
    sleep(10);

    // Remove light effect of the REALTIME layer , automatically display red
breathing light effect of the STABLE
    remove_layer(Led_Effect_layer_REALTIME, "");
    sleep(10);

    // Clear all lighting effects and turn off LED light
    RK_set_all_led_effect_off();

    for (;;);
    RK_led_exit();

    return 0;
}
```