

Rockchip Developer Guide RT-Thread KEYCTRL

文件标识: RK-KF-YF-071

发布版本: V1.0.0

日期: 2019-12-23

文件密级: 公开资料

免责声明

本文档按“现状”提供，福州瑞芯微电子股份有限公司（“本公司”，下同）不对本文档的任何陈述、信息和内容的准确性、可靠性、完整性、适销性、特定目的性和非侵权性提供任何明示或暗示的声明或保证。本文档仅作为使用指导的参考。

由于产品版本升级或其他原因，本文档将可能在未经任何通知的情况下，不定期进行更新或修改。

商标声明

“Rockchip”、“瑞芯微”、“瑞芯”均为本公司的注册商标，归本公司所有。

本文档可能提及的其他所有注册商标或商标，由其各自拥有者所有。

版权所有© 2019福州瑞芯微电子股份有限公司

超越合理使用范畴，非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

福州瑞芯微电子股份有限公司

Fuzhou Rockchip Electronics Co., Ltd.

地址: 福建省福州市铜盘路软件园A区18号

网址: www.rock-chips.com

客户服务电话: +86-4007-700-590

客户服务传真: +86-591-83951833

客户服务邮箱: fae@rock-chips.com

前言

概述

产品版本

芯片名称	内核版本
RK2108	RT-Thread V10.0.1

读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师 软件开发工程师

修订记录

日期	版本	作者	修改说明
2019-12-23	V1.0.0	Simon	第一次版本发布

目录

Rockchip Developer Guide RT-Thread KEYCTRL

1 HAL KEYCTRL 配置

 1.1 HAL CONFIG

 1.2 HAL 差异部分

 1.3 HAL 常用 API

2 RT-Thread KEYCTRL配置

 2.1 RT-Thread CONFIG

 2.2 RT-Thread 常用API

 2.3 RT-Thread 使用示例

3 TEST

 3.1 CONFIG配置

 3.2 USAGE

1 HAL KEYCTRL 配置

1.1 HAL CONFIG

依赖driver开启：

```
1 #ifdef RT_USING_KEYCTRL
2 #define HAL_KEYCTRL_MODULE_ENABLED
3 #endif
```

1.2 HAL 差异部分

不同芯片差异主要在CLK ID，可以在rk2108.h或者soc.h中查找

CLK_KEY_GATE: SCLK GATE ID，用于CLK开关；

PCLK_KEY_GATE: PCLK GATE ID，用于CLK开关；

1.3 HAL 常用 API

```
1 HAL_Status HAL_KeyCtrl_Init(struct KEY_CTRL_REG *reg, uint32_t
keyDetectionTh, uint32_t keyCalculatePeriodTh, uint32_t keyFilterIrqTh);
2 uint32_t HAL_KeyCtrl_GetValue(struct KEY_CTRL_REG *reg);
3 void HAL_KeyCtrl_ClearInt(struct KEY_CTRL_REG *reg);
```

2 RT-Thread KEYCTRL配置

2.1 RT-Thread CONFIG

```
1 scons --menuconfig
2
3 → RT-Thread rockchip rk2108 drivers
4     -*- Enable KEYCTRL
```

2.2 RT-Thread 常用API

```
1 static rt_bool_t rt_keyctrl_check_range(rt_uint32_t key_value, rt_uint32_t
target);
2 static rt_uint8_t rt_keyctrl_map_long_time_keycode(rt_uint8_t keycode);
3 static rt_uint8_t rt_keyctrl_map_keycode(rt_uint32_t key_value);
4 static void rt_keyctrl_save_key_code(char key_code);
5 static void rt_keyctrl_ind_callback(void);
6 static void rt_keyctrl_scan_timer_func(void *parameter);
7 static void rt_keyctrl_irqhandler(void);
8 static rt_err_t rt_keyctrl_init(rt_device_t dev);
9 static rt_size_t rt_keyctrl_read(rt_device_t dev, rt_off_t pos, void
*buffer, rt_size_t size);
10 static rt_err_t rt_keyctrl_control(rt_device_t dev, int cmd, void *args);
11 static int rt_keyctrl_dev_init(void);
```

2.3 RT-Thread 使用示例

使用示例:

```
1 rt_keyctrl_dev_init(void); /* 初始化clock,注册设备 */
2 rt_keyctrl_init(rt_device_t dev); /* 初始化设备,使能设备 */
3 rt_keyctrl_scan_timer_func(void *parameter); /* 处理按键 */
4 rt_keyctrl_save_key_code(char key_code); /* 保存按键 */
5 rt_keyctrl_ind_callback(void); /* 主动上传按键 */
```

3 TEST

3.1 CONFIG配置

```
1 RT-Thread bsp test case --->
2     RT-Thread Common Test case --->
3         [*]     Enable BSP Common KEYCTRL TEST
```

3.2 USAGE

使用示例:

```
1 keyctrl_test probe keyctrl /* 打开KEYCTRL设备 */
2 keyctrl_test read /* 读会保存的按键值 */
3 keyctrl_test set_ind callback 1 /* 主动上传按键值 */
```