Rockchip FreeRTOS PMIC, Chager, Powerkey Developer Guide

文件标识：RK-GL-YF-059

发布版本：V1.0.0

日期：2019-12-02

文件密级：公开资料

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

本文主要描述了RK2206 PMIC，Charger，Power key等驱动基本介绍与使用方法。

产品版本

芯片名称	内核版本
RK2206	FreeRTOS 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

版本号	作者	修改日期	修改说明
V1.0.0	黄小东	2019-12-02	初始版本

目录

---

1 PMIC

1.1 概述

PMIC全称Power management integrated circuit，一般情况下是一颗独立于主控的芯片，集成了电源控制，电源键控制，充电控制等模块。

1.2 配置

menuconfig中的配置：

使能PMIC驱动：

```
    BSP Driver  --->
        [*] Enable PMIC
```

PMIC驱动是一个核心驱动，为其他相关驱动提供接口，所以单独使能PMIC驱动并没有实际作用，需要根据具体情况使能特定模块驱动，如下所示：

```
BSP Driver  --->
    [*] Enable PMIC
    [*]     Enable PMIC Key
    [*]     Enable PMIC Charger
    [*]         Enable RK812 Charger
```

对于特定的某款PMIC芯片需要填充struct rk_pmic_desc来配置特定的信息，如下：

```
struct rk_pmic_desc
{
    int8 i2c_id;              /* 读写PMIC寄存器所使用的i2c的id */
    uint8 i2c_addr;           /* i2c设备地址 */
    uint8 on_src_reg;         /* pmic_on_source的寄存器偏低地址 */
    uint8 int_st_reg;         /* pmic_int_st的寄存器偏低地址 */
    uint8 int_reg_num;        /* pmic_int寄存器的个数 */
    uint8 int_num;            /* PMIC内部中断的个数 */
    uint8 on_src_num;         /* on source的个数 */
    uint8 power_ctrl_reg_num[PMIC_PWR_MAX]; /* 各个power_ctrl_reg[i]的数组长度 */
    int pmic_int_pin;         /* pmic_int pin序号 */
    eGPIO_pinLevel pmic_int_level;            /* pmic_int pin 的触发电平 */
    ePINCTRL_configParam pmic_int_ioparam;    /* pmic_int pin 的初始化参数 */
    int pmic_sleep_pin;                       /* pmic_sleep pin 序号 */
    eGPIO_pinLevel pmic_sleep_level;          /* pmic_sleep pin 的生效电平 */
    ePINCTRL_configParam pmic_sleep_ioparam;  /* pmic_sleep pin 的初始化参数 */
    const struct pmic_reg_data *power_ctrl_reg[PMIC_PWR_MAX]; /* PMIC处于各个电源
状态下需要配置的寄存器 */
    pmic_int_id *int_map;                     /* PMIC内部中断映射表 */
    pmic_on_src_t *on_src_map;                /* on source 映射表 */
    struct I2C_DEVICE_CLASS *i2cbus;          /* I2C设备指针 */
};
```

以RK812为例，需进行如下配置：

- src/bsp/RK2206/board/rk2206_evb/board.c

```
#include "driver/drv_pmic.h"

#ifdef CONFIG_DRIVER_PMIC
/* PMIC内部中断映射表 */
static pmic_int_id rk812_int_map[] =
{
    PKEY_FALL_INT,
    PKEY_RISE_INT,
    PKEY_INT,
    PKEY_LP_INT,
    HOTDIE_INT,
    PSW_FALL_INT,
    CLASSD_OCP_INT,
    VB_LO_INT,
    PLUG_IN_INT,
    PLUG_OUT_INT,
    CHRG_TERM_INT,
    CHRG_TIME_INT,
    CHRG_CT_INT,
    USB_OV_INT,
    VB_OV_INT,
```

```
    CHRG_BAT_HI_INT,
};

/* PMIC ON SOURCE映射表 */
static pmic_on_src_t rk812_on_src_map[] =
{
    PMIC_ON_PSW_RISE,
    PMIC_RESTART_DEV_RST,
    PMIC_RESTART_RESETB,
    PMIC_ON_PLUG_IN,
    PMIC_ON_SRC_UNKNOWN,
    PMIC_ON_SRC_UNKNOWN,
    PMIC_ON_SRC_UNKNOWN,
    PMIC_ON_KEY,
};

static const struct pmic_reg_data rk812_off_reg[] =
{
    PMIC_REG_DATA(RK812_SYS_CFG3, RK812_DEV_OFF_SHFT, RK812_DEV_OFF_MSK, 1),
};

static const struct pmic_reg_data rk812_on_reg[] =
{
    PMIC_REG_DATA(RK812_BUCK1_ON_VSEL_REG, PWM_MODE_SHFT, PWM_MODE_MSK,
AUTO_PWM_MODE),
    PMIC_REG_DATA(RK812_GPIO_INT_CFG, RK812_INT_POL_SHFT, RK812_INT_POL_MSK,
RK812_INT_POL_L),
};

static const struct pmic_reg_data rk812_slp_reg[] =
{
    PMIC_REG_DATA(RK812_BUCK1_ON_VSEL_REG, PWM_MODE_SHFT, PWM_MODE_MSK,
AUTO_PWM_MODE),
    PMIC_REG_DATA(RK812_POWER_CFG, RK812_BUCK_LDO_BYPASS_SHFT,
RK812_BUCK_LDO_BYPASS_MSK, 1),
};

static struct rk_pmic_desc pmic_desc =
{
    .pmic_int_pin = 28,
    .pmic_int_level = GPIO_LOW,
    .pmic_int_ioparam = 0,
    .pmic_sleep_pin = -1,
    .i2c_id = I2C_DEV2,
    .i2c_addr = RK812_I2C_ADDR,
    .int_st_reg = RK812_INT_STS_REG0,
    .int_reg_num = 2,
    .on_src_reg = RK812_ON_SOURCE_REG,
    .int_map = rk812_int_map,
    .int_num = HAL_ARRAY_SIZE(rk812_int_map),
    .on_src_map = rk812_on_src_map,
    .on_src_num = HAL_ARRAY_SIZE(rk812_on_src_map),
    PMIC_PWR_CTRL_REG_INIT(PMIC_PWR_OFF, rk812_off_reg),
    PMIC_PWR_CTRL_REG_INIT(PMIC_PWR_ON, rk812_on_reg),
    PMIC_PWR_CTRL_REG_INIT(PMIC_PWR_SLP, rk812_slp_reg),
};
#endif
```

```
COMMON API void System_Power_Init(void)
{
    ...

#ifdef CONFIG_DRIVER_PMIC
    pmic_desc_init(&pmic_desc);/* 将上面填充好的struct rk_pmic_desc注册到PMIC驱动里
*/
#endif

    ...
}
```

1.3 代码和API

- src/driver/pmic/drv_pmic.c
- include/driver/drv_pmic.h

```
 /* 读PMIC寄存器 */
uint32 pmic_read(struct rk_pmic_desc *pmic_desc, uint16 reg);
 /* 写PMIC寄存器 */
rk_err_t pmic_write(struct rk_pmic_desc *pmic_desc, uint16 reg, uint8 data);
 /* 读PMIC寄存器特定的位域 */
int pmic_reg_field_read(const struct pmic_reg_field *field, unsigned int *val);
 /* 写PMIC寄存器特定的位域 */
int pmic_reg_field_write(const struct pmic_reg_field *field, unsigned int val);
/* 进行PMIC reset操作 */
rk_err_t pmic_power_reset(void);
/* 进行PMIC off操作 */
rk_err_t pmic_power_off(void);
/* 进行PMIC suspend操作 */
rk_err_t pmic_power_suspend(void);
/* 进行PMIC resume操作 */
rk_err_t pmic_power_resume(void);
/* 获取PMIC ON SOURCE */
pmic_on_src_t pmic_get_on_source(void);
/* 绑定PMIC中断处理函数 */
int pmic_attach_irq(int irq, void (*hdr)(void *args), void *args);
/* 解绑PMIC中断处理函数 */
int pmic_detach_irq(int irq);
/* 使能或关闭指定PMIC中断 */
int pmic_irq_enable(int irq, int enable);
/* 初始化PMIC desc */
void pmic_desc_init(struct rk_pmic_desc *desc);
/* 初始化PMIC */
int pmic_setup(void);
```

1.4 使用范例

PMIC的API由特定模块进行调用，以power key为例：

- src/driver/key/drv_pmic_key.c

```
#include "driver/drv_pmic.h"

void pmic_key_setup(void)
{
    /* 绑定PMIC中断处理函数 */
    pmic_attach_irq(PKEY_FALL_INT, pmic_key_fall_hdl, NULL);
    pmic_attach_irq(PKEY_RISE_INT, pmic_key_rise_hdl, NULL);

    /* 使能PMIC中断 */
    pmic_irq_enable(PKEY_FALL_INT, 1);
    pmic_irq_enable(PKEY_RISE_INT, 1);
}
```

2 Charger

2.1 概述

PMIC中可能集成有充电控制模块，用于控制电池的充电电压，电流，时间等。

2.2 配置

menuconfig中的配置：

使能PMIC Charger驱动（需要先使能PMIC驱动）：

```
    BSP Driver  --->
        [*] Enable PMIC
        [*]     Enable PMIC Charger
```

对于特定的某款PMIC芯片可以填充struct pmic_charger_desc配置特定的充电信息，如下：

```
struct pmic_charger_desc
{
    struct pmic_reg_field *reg_fields;
    uint32 max_chrg_current; /* 最大充电电流 */
    uint32 max_chrg_voltage; /* 最大充电电压 */

    uint32 max_input_current; /* 最大输入电流 */
    uint32 min_input_voltage; /* 最大输入电压 */
}
```

以RK812为例，可以进行如下配置：

- src/bsp/RK2206/board/rk2206_evb/board.c

```
#ifdef CONFIG_DRIVER_PMIC_CHARGER
static struct pmic_charger_desc pmic_charger =
{
    .max_chrg_current = 500, /* 设置最大充电电流500mA */
    .max_chrg_voltage = 4200, /* 设置最大充电电压4.2V */
};
#endif

COMMON API void System_Power_Init(void)
{
```

```
    ...

/* 注册配置好的struct pmic_charger_desc到pmic_charge驱动里 */
#ifdef CONFIG_DRIVER_PMIC_CHARGER
    pmic_charge_desc_init(&pmic_charger);
#endif

    ...
}
```

2.3 代码和API

- src/driver/charger/drv_pmic_charge.c
- include/driver/drv_pmic_charge.h

```
/* 读PMIC CHARGER寄存器特定的位域 */
int pmic_charge_field_read(struct pmic_charger_desc *desc, int id);
/* 写PMIC CHARGER寄存器特定的位域 */
int pmic_charge_field_write(struct pmic_charger_desc *desc, int id, uint8 val);
/* 获取当前充电进度 */
charge_progress_t pmic_charge_get_charge_progress(void);
/* 初始化pmic_charger_desc */
void pmic_charge_desc_init(struct pmic_charger_desc *desc);
/* 初始化pmic charge */
void pmic_charge_setup(void);
```

另外，下列函数是weak属性，需根据特定某款PMIC来具体实现：

```
charge_progress_t pmic_charge_get_charge_progress(void);
void pmic_charge_desc_init(struct pmic_charger_desc *desc);
void pmic_charge_setup(void);
```

以RK812为例：

```
BSP Driver  --->
    [*] Enable PMIC
    [*]     Enable PMIC Charger
    [*]         Enable RK812 Charger
```

- src/driver/charger/drv_rk812_charge.c

```
#include "driver/drv_pmic.h"
#include "driver/drv_pmic_charger.h"

charge_progress_t pmic_charge_get_charge_progress(void)
{
    return charger.charge_prg;
}

void pmic_charge_desc_init(struct pmic_charger_desc *desc)
{
    RK_ASSERT(desc);

    charger_desc = desc;
    charger_desc->reg_fields = rk812_charge_reg_fields;
```

```
}

void pmic_charge_setup(void)
{
    RK_ASSERT(charger_desc);
    charger.charge_prg = CHARGE_STATUS_NOT_CHARGING;

    pmic_charge_pre_init();
    pmic_charge_irqs_init();

    pmic_charge_get_charge_status();
    rk_printf("max_chrg_current: %d\n"
              "max_chrg_voltage: %d\n",
              charger_desc->max_chrg_current,
              charger_desc->max_chrg_voltage);
}
```

3 Powerkey

3.1 概述

PMIC中一般都集成有电源键控制模块，用于检测和控制电源的各种状态，如按下，抬起，短按，长按等。如果硬件上将电源键连接在PMIC_INT pin上，就可以用PMIC来检测和控制电源键。

3.2 配置

menuconfig中的配置：

使能PMIC_Key驱动（需要先使能PMIC驱动）：

```
    BSP Driver  --->
        [*] Enable PMIC
        [*]     Enable PMIC Key
```

上述配置后系统就能响应电源键。