RK816 开发指南

文件标识：RK-KF-YF-67

发布版本：V1.0.1

日期：2022-01-06

文件密级：□绝密 □秘密 □内部资料 ■公开

免责声明

本文档按“现状”提供，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。

瑞芯微电子股份有限公司

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

本文档主要介绍 RK816 的各个子模块，介绍相关概念、功能、dts 配置和一些常见问题的分析定位。

产品版本

芯片名称	内核版本
RK816	3.10、 4.4、4.19

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	作者	日期	修改说明
V1.0.0	张晴	2019-11-25	初始版本
V1.0.1	黄莹	2022-05-30	修改格式

目录

1. 基础

1.1 概述

RK816 是一款高性能 PMIC，RK816 集成 4 个大电流 DCDC、1个升压BOOST、6 个 LDO、1个OTG输出、1 个 RTC、可调上电时序，而且还集成了开关充电，智能功率路径管理，库仑计等功能。。

系统中各路电源总体分为两种：DCDC 和 LDO。两种电源的总体特性如下（详细资料请自行搜索）：

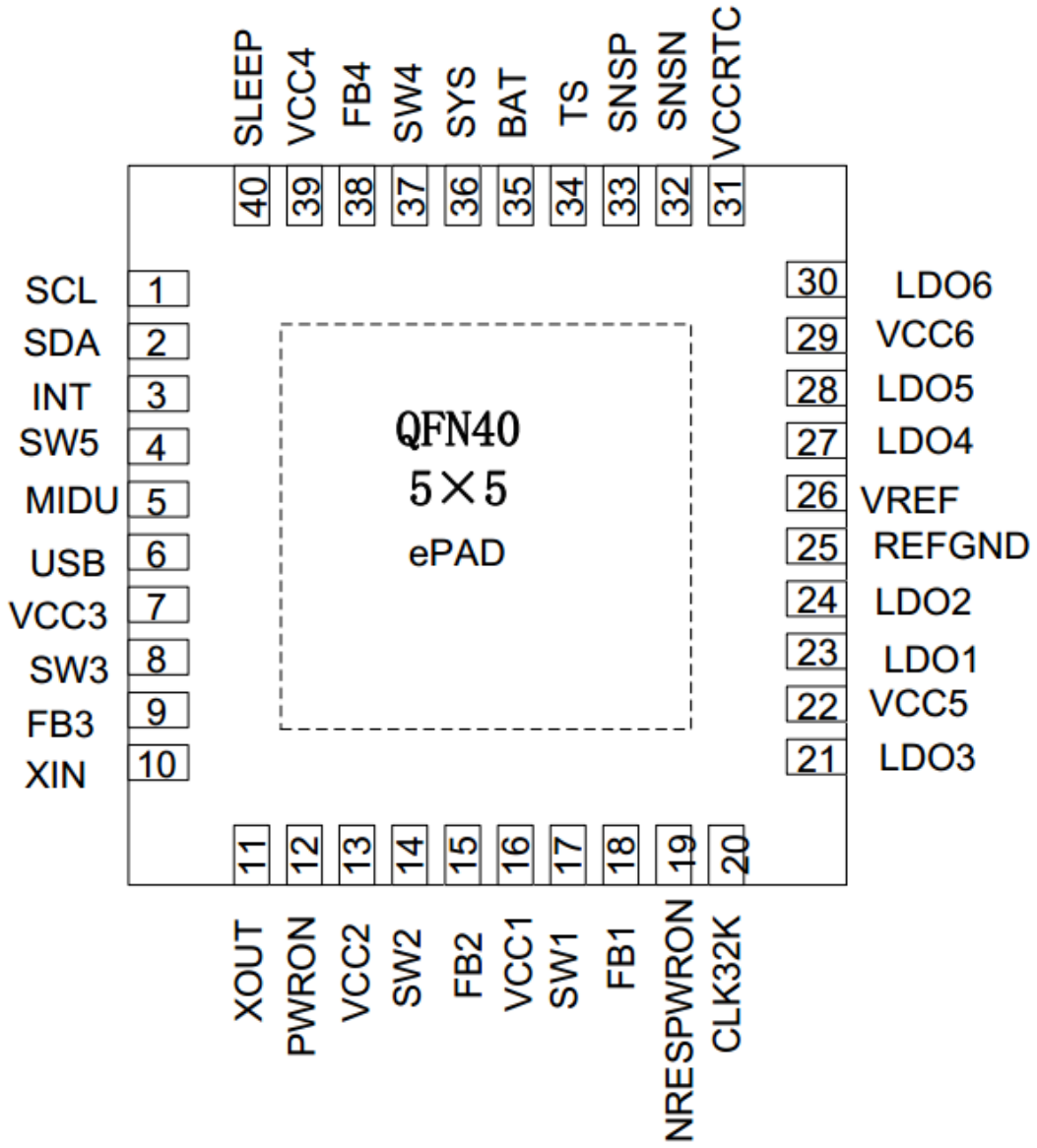
1. DCDC：输入输出压差大时，效率高，但是存在纹波比较大的问题，成本高，所以大压差，大电流负载时使用。一般有两种工作模式。PWM 模式：纹波瞬态响应好，效率低；PFM 模式：效率高，但是负载能力差。
2. LDO：输入输出压差大时，效率低，成本低，为了提高 LDO 的转换效率，系统上会进行相关优化如：LDO 输出电压为 1.1V，为了提高效率，其输入电压可以从 VCCIO_3.3V 的 DCDC 给出。所以电路上如果允许尽量将 LDO 接到 DCDC 输出回路，但是要注意上电时序。

1.2 功能

从使用者的角度看，RK816 的功能概况起来可以分为 4 个部分：

1. regulator 功能：控制各路 DCDC、LDO 电源状态；
2. rtc 功能：提供时钟计时、定时等功能；
3. gpio 功能：out1 和 out2 两个推挽输出引脚（只能 output），可当普通 gpio 使用；
4. pwrkey 功能：检测 power 按键的按下/释放，可以为 AP 节省一个 gpio。
5. 充电功能和电量计功能，在本文中不做详细介绍，详细可以参考文档《Rockchip_RK818_RK816_Developer_Guide_Fuel_Gauge_CN》

1.3 芯片引脚功能



下面描述中，SLEEP 和 INT 引脚需要重点关注：

PIN NO	PIN NAME	PIN DESCRIPTION
1	SCL	I2C clock input
2	SDA	I2C data input and output
3	INT	Interrupt request pin.

PIN NO	PIN NAME	PIN DESCRIPTION
4	SW5	Charger switching node / boost switching node
5	MIDU	Middle point of USB power supply / boost output
6	USB	USB power supply
7	VCC3	Power supply of buck3
8	SW3	Switching node of buck3
9	VFB3	Output feedback voltage of buck3
10	XIN	32.768KHz crystal oscillator input
11	XOUT	32.768KHz crystal oscillator output
12	PWRON	Power on or power off enable pin, active low, internal 20k pull high to power supply
13	VCC2	Power supply of buck2
14	SW2	Switching node of buck2
15	VFB2	Output feedback voltage of buck2
16	VCC1	Power supply of buck1
17	SW1	Switching node of buck1
18	VFB1	Output feedback voltage of buck1
19	NRESPWON	Reset pin after power on, active low.
20	CLK32K	32.768KHz clock output, open drain
21	LDO3	LDO3 output
22	VCC5	Power supply of LDO1/2/3
23	LDO1	LDO1 output
24	LDO2	LDO2 output
25	REFGND	Reference ground
26	VREF	Internal reference voltage
27	LDO4	LDO4 output
28	LDO5	LDO5 output
29	VCC6	Power supply of LDO4/5/6
30	LDO6	LDO6 output
31	VCCRTC	Power supply of RTC, decouple it to GND with a 1uF cap
32	SNSN	Battery charging or discharging current sense negative terminal
33	SNSP	Battery charging or discharging current sense postive terminal
34	TS(GPIO1)	Thermistor input. Connect a thermistor from this pin to ground. The thermistor is usually inside the battery pack. (multi-function for GPIO1)
35	BAT	Battery positive terminal
36	SYS	System terminal
37	SW4	Switching node of buck4
38	VFB4	Output feedback voltage of buck4
39	VCC4	Power supply of buck4
40	SLEEP	Sleep mode control input
Exposed pad	Exposed ground	Ground

1.4 重要概念

- I2C 地址

  7 位从机地址：0x1a

- PMIC 有 3 种工作模式

  1. PMIC normal 模式

  系统正常运行时 PMIC 处于 normal 模式，此时 pmic_sleep 为低电平。

  2. PMIC sleep 模式

  系统休眠时需要待机功耗尽量低，PMIC 会切到 sleep 模式减低自身功耗，这时候一般会降低某些路的输出电压，或者直接关闭输出，这可以根据实际产品需求进行配置。系统待机时 AP 通过 I2C 指令把 pmic_sleep 配置成 sleep 模式，然后拉高 pmic_sleep 即可让 PMIC 进入 sleep 状态；当 SoC 唤醒时 pmic_sleep 恢复为低电平，PMIC 退出休眠模式。

  3. PMIC shutdown 模式

  当系统进入关机流程的时候，PMIC 需要完成整个系统的电源下电操作。AP 通过 I2C 指令把 pmic_sleep 配置成 shutdown 模式，然后拉高 pmic_sleep 即可让 PMIC 进入 shutdown 状态。

- pmic_sleep 引脚

  常态为低电平，PMIC 处于 normal 模式。当引脚拉高的时候会切换到 sleep 或者 shutdown 的模式。

- pmic_int 引脚

  常态为高电平，当有中断产生的时候变为低电平。如果中断没有被处理，则会一直维持低电平。

- out1/out2 引脚

  这两个引脚可以当普通的 gpio 使用（推挽输出），但是只有 gpio 输出模式。

- pmic_pwron 引脚

  pwrkey 的功能需要硬件上将 power 按键接到这个引脚，驱动通过这个引脚来判断按下/释放。

- 各路 DCDC 的工作模式

  DCDC 有 PWM（也叫 force PWM）、PFM 模式，但是 PMIC 有一种模式会动态切换 PWM、PFM，这就是我们通常所说的 AUTO 模式。PMIC 支持 PWM、AUTO PWM/PFM 两种模式，AUTO 模式效率高但是纹波瞬态响应会差。出于系统稳定性考虑，运行时都是设置为 PWM 模式，系统进入休眠时会选择切换到 AUTO PWM/PFM。

- DCDC3 电压调节

  DCDC3 这路电源比较特殊，不能通过寄存器修改电压，只能通过外部电路的分压电阻进行调节，所以如果需要修改电压请修改外围硬件，在 Rockchip 的方案上一般作为 VCC_DDR 使用。

- DCDC 和 LDO 的运行时电压调节范围

1. DCDC 电压范围不连续：

电压范围(V)	**步进值(mV)**	**具体档位值(V)**
0.7125 ~ 1.45	12.5	0.7125、0.725、0.737.5、 ……、1.45
1.8 ~ 2.2	200	1.8、 2.0、2.2
2.3	无	2.3

2. LDO 电压连续：

电压范围(V)	步进值(mV)	具体档位值(V)
0.8 ~ 3.4	100	0.8、 0.9、1.0、1.1、 1.2、 …… 3.4

1.5 上电条件和时序

1. 上电条件

   只要满足下面任意一个条件即可以实现 PMIC 上电:

- EN 信号从低电平变高电平触发
- EN 信号保持高电平，且 RTC 闹钟中断触发
- EN 信号保持高电平，按 PWRON 键触发

2. 上电时序

   每款 SOC 平台对各路电源上电时序要求可能不一样，目前上电时序有如下情况，具体请参考最新的 datasheet:

BOOT(OTP)			1（RK816B-1）		0（RK816B-2）		0（RK816B-3）	
	Output voltage range	Rate Current	Default voltage	Power up sequence	Default voltage	Power up sequence	Default voltage	Power up sequence
BUCK1	0.7125V-2.3V（0.7125~1.45V，step 12.5mV)	2A	1.1V	2	1.0V	3	1.1V	2
BUCK2	0.7125V-2.3V（0.7125~1.45V，step 12.5mV)	2A	1.1V	1	2.2V	1	1.1V	1
BUCK3	setting by external resistors	1A	x	3	x	4	x	3
BUCK4	0.8V-3.5V(step 0.1V)	1A	3.3V	1	3.3V	6	3.3V	1
BOOST	4.7-5.4V(step 0.1V)	2A	5V	OFF	5V	OFF	5V	OFF
LDO1	0.8V-3.4V(step 0.1V)	300mA	1.0V	OFF	1.0V	2	1.8V	OFF
LDO2	0.8V-3.4V(step 0.1V)	300mA	1.8V	1	1.8V	5	1.8V	1
LDO3	0.8V-3.4V(step 0.1V)	100mA	1.1V	1	1.0V	2	1.1V	1
LDO4	0.8V-3.4V(step 0.1V)	300mA	1.0V	OFF	3.3V	6	3.3V	OFF
LDO5	0.8V-3.4V(step 0.1V)	300mA	3.0V	4	1.8V	OFF	3.3V	4
LDO6	0.8V-3.4V(step 0.1V)	300mA	3.0V	4	2.8V	OFF	3.3V	4

2. 配置

2.1 驱动和 menuconfig

3.10 内核配置

RK816 驱动文件（复用 RK816 驱动）：

```
drivers/mfd/rk816.c
drivers/input/misc/rk816-pwrkey.c
drivers/rtc/rtc-rk816.c
drivers/gpio/gpio-rk816.c
drivers/regulator/rk816-regulator.c
drivers/power/rk816_battery.c
```

RK816 dts文件可参考:

```
arch/arm/boot/dts/rk816.dtsi
arch/arm/boot/dts/rk3126-86v-rk816.dts
```

menuconfig 里对应的宏配置:

```
CONFIG_MFD_RK816
CONFIG_GPIO_RK816
CONFIG_RTC_RK816
CONFIG_REGULATOR_RK816
CONFIG_INPUT_RK816_PWRKEY
CONFIG_BATTERY_RK816
```

4.4 内核配置

RK816 驱动文件:

```
drivers/mfd/rk808.c
drivers/input/misc/rk8xx-pwrkey.c
drivers/rtc/rtc-rk808.c
drivers/gpio/gpio-rk8xx.c
drivers/regulator/rk808-regulator.c
drivers/clk/clk-rk808.c
drivers/power/rk816_battery.c
```

menuconfig 里对应的宏配置:

```
CONFIG_MFD_RK808
CONFIG_RTC_RK808
CONFIG_GPIO_RK8XX
CONFIG_REGULATOR_RK808
CONFIG_INPUT_RK8XX_PWRKEY
CONFIG_COMMON_CLK_RK808
CONFIG_BATTERY_RK816
```

4.19 内核配置

RK816 驱动文件:

```
drivers/mfd/rk808.c
drivers/input/misc/rk816-pwrkey.c       // 跟4.4内核不同
drivers/rtc/rtc-rk808.c
drivers/pinctrl/pinctrl-rk816.c         // 跟4.4内核不同
drivers/regulator/rk808-regulator.c     // 跟4.4内核不同
drivers/clk/clk-rk808.c
drivers/power/rk816_battery.c
```

menuconfig 里对应的宏配置：

```
CONFIG_MFD_RK808
CONFIG_RTC_RK808
CONFIG_PINCTRL_RK816
CONFIG_REGULATOR_RK808
CONFIG_INPUT_RK816_PWRKEY
CONFIG_COMMON_CLK_RK808
CONFIG_BATTERY_RK816
```

2.2 DTS 配置

3.10 内核配置

DTS 的配置包括：I2C 挂载、主体、regulator、rtc、poweroff 等部分。

```
&i2c1 {
    rk816: rk816@1a {
        reg = <0x1a>;
        status = "okay";
    };
};

/include/ "rk816.dtsi"
&rk816 {
    gpios = <&gpio1 GPIO_A5 GPIO_ACTIVE_HIGH>, <&gpio1 GPIO_A1 GPIO_ACTIVE_LOW>;
    rk816,system-power-controller;
    rk816,support_dc_chg = <1>;/*1: dc chg; 0:usb chg*/
    io-channels = <&adc 0>;
    gpio-controller;
    #gpio-cells = <2>;
    rtc {
        status = "okay";
    };

    regulators {
        rk816_dcdc1_reg: regulator@0{
            regulator-name= "vdd_arm";
            regulator-min-microvolt = <700000>;
            regulator-max-microvolt = <1500000>;
            regulator-initial-mode = <0x1>;
            regulator-initial-state = <3>;
            regulator-always-on;
            regulator-state-mem {
```

```
                regulator-state-mode = <0x2>;
                regulator-state-disabled;
                regulator-state-uv = <900000>;
            };
        };

        rk816_dcdc2_reg: regulator@1 {
            regulator-name= "vdd_logic";
            regulator-min-microvolt = <700000>;
            regulator-max-microvolt = <1500000>;
            regulator-initial-mode = <0x1>;
            regulator-initial-state = <3>;
            regulator-always-on;
            regulator-state-mem {
                regulator-state-mode = <0x2>;
                regulator-state-enabled;
                regulator-state-uv = <1000000>;
            };
        };
        rk816_dcdc3_reg: regulator@2 {
                    .............
        };
        ..................................
    };
};
```

1. I2C 挂载

   整个完整的 rk816 节点挂在对应的 i2c 节点下面，并且配置 status = "okay";

2. 主体部分

- 不可修改部分

```
    rk816,system-power-controller：声明RK816具备管理系统下电的功能;
    gpio-controller：声明RK816具有GPIO的功能;
    #gpio-cells：使用者引用RK816的GPIO时需要指定的参数个数;
```

说明：如果某个节个需要引用 RK816 的 GPIO 进行使用，引用格式如下：

`gpios = <&rk816 0 GPIO_ACTIVE_LOW>;`
第一个参数： &rk816 固定，不可改动；
第二个参数： 引用 rk816 的哪个 gpio，只能是 0 或者 1，其中 0：out1， 1：out2；
第三个参数： gpio 的极性。

- 可修改部分

  gpios：指定 pmic_int（第一个）和 pmic_sleep（第二个）引脚；

3. regulator 部分

- `regulator-name`: 电源名字，建议和硬件图上保持一致，使用 regulator_get 接口时需要匹配这个名字；
- `regulator-min-microvolt`：运行时可调节的最小电压；
- `regulator-max-microvolt`：运行时可调节的最大电压；
- `regulator-initial-mode`：运行时 DCDC 工作模式，一般配置为 1。 1：force pwm，2：auto pwm/pfm；

- `regulator-state-mode`： 休眠时 DCDC 工作模式，一般配置为 2。1：force pwm， 2：auto pwm/pfm；
- `regulator-initial-state`：suspend 时的模式，必须配置成 3；
- `regulator-boot-on`：存在这个属性时，在注册 regulator 的时候就会使能这路电源；
- `regulator-always-on`：存在这个属性时，运行时不允许关闭这路电源且会在注册的时候使能这路电源；
- `regulator-state-enabled`：休眠时保持上电状态，想要关闭该路电源，则改成”regulator-state-disabled”；
- `regulator-state-uv`：休眠不断电情况下的待机电压。

说明：

> 如果 `regulator-min-microvolt` 和 `regulator-max-microvolt` 的电压相等，则在注册这个 `regulator` 的时候系统框架默认会把这个电压设置下去并使能这路电源，不需要使用者干预。
>
> 如果 `regulator-boot-on` 或者 `regulator-always-on` 存在，则系统框架在注册这路 `regulator` 的时候默认会进行 `enable`，此时的这路 `regulator` 的电压有 `2` 种情况：如果 `regulator-min-microvolt` 和 `regulator-max-microvolt` 的电压相等，则系统框架会把这路电压设置为当前这个电压值；如果 `regulator-min-microvolt` 和 `regulator-max-microvolt` 的电压不相等，则此时的电压是 `PMIC` 的本身的硬件默认上电电压。

4. rtc 部分

如果不想使能 RTC 的功能（如 box 产品上），则需要像上面那样增加节点，显式指明为 status = "disabled"。如果需要使能的的话则可以把整个 RTC 节点去掉或者设置状态为 status = "okay"即可。

5. poweroff 部分

```
gpio_poweroff {
    compatible = "gpio-poweroff";
    gpios = <&gpio2 GPIO_D2 GPIO_ACTIVE_HIGH>;
    status = "okay";
};
```

因为 RK816 支持拉高 pmic_sleep 引脚进行整个 PMIC 的下电，所以需要在根节点下增加这个节点。其中 gpios 是可改部分，用于指明 pmic_sleep 引脚。

如果没有注册这个功能，也可以使用RK816的软件关机，流程跟RK808和RK818相同。

4.4 内核配置

DTS 的配置包括：i2c 挂载、主体、rtc、pwrkey、gpio、regulator 等部分。

```
&pinctrl {
    pmic {
        pmic_int_l: pmic-int-l {
        rockchip,pins =
            <2 6 RK_FUNC_GPIO &pcfg_pull_up>;   /* gpio2_a6 */
        };
    };
};

&i2c1 {
    status = "okay";
    rk816: pmic@1a {
```

```
        compatible = "rockchip,rk816";
        reg = <0x1a>;
        interrupt-parent = <&gpio0>;
        interrupts = <2 IRQ_TYPE_LEVEL_LOW>;
        pinctrl-names = "default";
        pinctrl-0 = <&pmic_int_l>;
        rockchip,system-power-controller;
        wakeup-source;
        gpio-controller;
        #gpio-cells = <2>;
        #clock-cells = <1>;
        clock-output-names = "rk816-clkout1", "rk816-clkout2";
        extcon = <&u2phy>;

        vcc1-supply = <&vcc_sys>;
        vcc2-supply = <&vcc_sys>;
        vcc3-supply = <&vcc_sys>;
        vcc4-supply = <&vcc_sys>;
        vcc5-supply = <&vcc_io>;
        vcc6-supply = <&vcc_sys>;

        gpio {
            status = "okay";
        };

        pwrkey {
            status = "okay";
        };

        rtc {
            status = "okay";
        };

        battery {
            compatible = "rk816-battery";
            ocv_table = < 3500 3625 3685 3697 3718 3735 3748
                    3760 3774 3788 3802 3816 3834 3853
                    3877 3908 3946 3975 4018 4071 4106>;
            design_capacity = <2500>;
            design_qmax = <2750>;
            bat_res = <100>;
            max_input_current = <1500>;
            max_chrg_current = <1300>;
            max_chrg_voltage = <4200>;
            sleep_enter_current = <300>;
            sleep_exit_current = <300>;
            sleep_filter_current = <100>;
            power_off_thresd = <3500>;
            zero_algorithm_vol = <3850>;
            max_soc_offset = <60>;
            monitor_sec = <5>;
            virtual_power = <0>;
            power_dc2otg = <0>;
            dc_det_adc = <0>;
        };
```

```
        regulators {

            vdd_arm: DCDC_REG1{
                regulator-name= "vdd_arm";
                regulator-min-microvolt = <750000>;
                regulator-max-microvolt = <1500000>;
                regulator-ramp-delay = <6001>;
                regulator-initial-mode = <1>;
                regulator-always-on;
                regulator-boot-on;
                regulator-state-mem {
                    regulator-off-in-suspend;
                    regulator-suspend-microvolt = <900000>;
                };
            };

            vdd_log: DCDC_REG2 {
                regulator-name= "vdd_logic";
                regulator-min-microvolt = <750000>;
                regulator-max-microvolt = <1500000>;
                regulator-ramp-delay = <6001>;
                regulator-initial-mode = <1>;
                regulator-always-on;
                regulator-boot-on;
                regulator-state-mem {
                    regulator-on-in-suspend;
                    regulator-suspend-microvolt = <1000000>;
                };
            };
            vcc_ddr: RK816_DCDC3@2 {
                ................
            };
            ..............................
        };
    };
};
```

1. i2c 挂载

整个完整的 rk816 节点挂在对应的 i2c 节点下面，并且配置 status = "okay";

2. 主体部分

- 不可修改:

```
compatible = "rockchip,rk816";
reg = <0x18>;
rockchip,system-power-controller;
wakeup-source;
gpio-controller;
#gpio-cells = <2>;
```

- 可修改（按照 pinctrl 规则）

interrupt-parent：pmic_int 隶属于哪个 gpio；
interrupts：pmic_int 在 interrupt-parent 的 gpio 上的引脚索引编号和极性；
pinctrl-names：不修改，固定为 "default"；
pinctrl-0：引用 pinctrl 里定义好的 pmic_int 引脚；

3. rtc、pwrkey、gpio

如果 menuconfig 选中了这几个模块，但是实际又不需要使能这几个驱动，那么可以在 dts 里增加 rtc、pwrkey、gpio 节点，并且显式指明状态为 status = "disabled"，这样就不会使能驱动，但是开机信息会有错误 log 报出，可以忽略；如果要使能驱动，则可以去掉相应的节点，或者设置状态为 status = "okay"。

4. regulator

- `regulator-compatible`：驱动注册时需要匹配的名字，不能改动，否则会加载失败；
- `regulator-name`：电源的名字，建议和硬件图上保持一致，使用 regulator_get 接口时需要匹配这个名字；
- `regulator-init-microvolt`：u-boot阶段的初始化电压，kernel阶段无效；
- `regulator-min-microvolt`：运行时可以调节的最小电压；
- `regulator-max-microvolt`：运行时可以调节的最大电压；
- `regulator-initial-mode`：运行时 DCDC 的工作模式，一般配置为 1。 1：force pwm，2：auto pwm/pfm；
- `regulator-mode`：休眠时 DCDC 的工作模式，一般配置为 2。1：force pwm， 2：auto pwm/pfm；
- `regulator-initial-state`：suspend 时的模式，必须配置成 3；
- `regulator-boot-on`：存在这个属性时，在注册 regulator 的时候就会使能这路电源；
- `regulator-always-on`：存在这个属性时，表示运行时不允许关闭这路电源且会在注册的时候使能这路电源；
- `regulator-ramp-delay`：DCDC 的电压上升时间，固定配置为 12500；
- `regulator-on-in-suspend`：休眠时保持上电状态，想要关闭该路电源，则改成”regulator-off-in-suspend”；
- `regulator-suspend-microvolt`：休眠不断电情况下的待机电压。

4.19 内核配置

请参考4.4内核DTS配置。差异点：4.19内核的DTS配置不再需要gpio子节点，但其他模块依然使用`gpios = <&rk816 0 GPIO_ACTIVE_LOW>;`的方式引用和使用rk816的pin脚。

2.3 函数接口

如下几个接口基本可以满足日常使用，包括 regulator 开、关、电压设置、电压获取等：

1. 获取 regulator：

   `struct regulator *regulator_get(struct device *dev, const char *id)`

   dev 默认填写 NULL 即可，id 对应 dts 里的 regulator-name 属性。

2. 释放 regulator
   `void regulator_put(struct regulator *regulator)`

3. 打开 regulator
   `int regulator_enable(struct regulator *regulator)`

4. 关闭 regulator

   `int regulator_disable(struct regulator *regulator)`

5. 获取 regulator 电压

```
int regulator_get_voltage(struct regulator *regulator)
```

6. 设置 regulator 电压

```
int regulator_set_voltage(struct regulator *regulator, int min_uV, int max_uV)
```

传入的参数时保证 min_uV = max_uV，由调用者保证。

7. 范例

```
struct regulator *rdev_logic;

rdev_logic = regulator_get(NULL, "vdd_logic");          // 获取vdd_logic
regulator_enable(rdev_logic);                           // 使能vdd_logic
regulator_set_voltage(rdev_logic, 1100000, 1100000);    // 设置电压1.1v
regulator_disable(rdev_logic);                          // 关闭vdd_logic
regulator_put(rdev_logic);                              // 释放vdd_logic
```

说明：4.4或者4.19内核还提供了`devm_`开头的regulator接口帮开发者管理要申请的资源。

3. Debug

3.1 3.10内核

因为 PMIC 涉及的驱动在使用逻辑上都不复杂，重点都体现在最后的寄存器设置上。所以目前常用的 debug 方式就是直接查看 rk816 的寄存器，通过如下节点：

`/sys/rk816/rk816_test`

读寄存器：

`echo r [addr]  > /sys/rk816/rk816_test`

写寄存器：

`echo w [addr] [value] > /sys/rk816/rk816_test`

范例：

`echo r 0x2f > /sys/rk816/rk816_test          // 读取0x2f寄存器的值，为0x9b`

```
1|shell@rk3228h:/ # echo r 0x2f >  /sys/rk816/rk816_test
[  283.091704]  [2:              sh:  618] ------zhangqing: get cmd = r
[  283.091766]  [2:              sh:  618] CMD : r 2f
[  283.091914]  [2:              sh:  618]
[  283.091914]  [2:              sh:  618] 2f 9b
```

`echo w 0x2f 0x9c > /sys/rk816/rk816_test    // 设置0x2f寄存器的值为0x9c`

一般写操作执行完之后最好再读一遍确认是否写成功。

```
1|shell@rk3228h:/ # echo r 0x2f >  /sys/rk816/rk816_test
[  283.091704]  [2:              sh:  618] ------zhangqing: get cmd = r
[  283.091766]  [2:              sh:  618] CMD : r 2f
[  283.091914]  [2:              sh:  618]
[  283.091914]  [2:              sh:  618] 2f 9b
```

3.2 4.4内核

命令格式同 3.10 内核一样，只是节点路径不同，4.4 内核上的 debug 节点路径是：

`/sys/rk8xx/rk8xx_dbg`

3.3 4.19内核

请参考4.4内核命令。