RK818 开发指南

发布版本：1.0

作者邮箱：zhangqing@rock-chips.com

日期：2019.11

文档密级：公开资料

前言

概述

本文档主要介绍 RK818 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
RK818	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

日期	版本	作者	修改说明
2019.11.25	V1.0	张晴	初始版本

---

1 基础

1.1 概述

RK818 是一款高性能 PMIC，RK818 集成 4 个大电流 DCDC、1个大电流升压BOOST、9 个 LDO、1个SWITCH、一个HDIM5V输出、一个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。

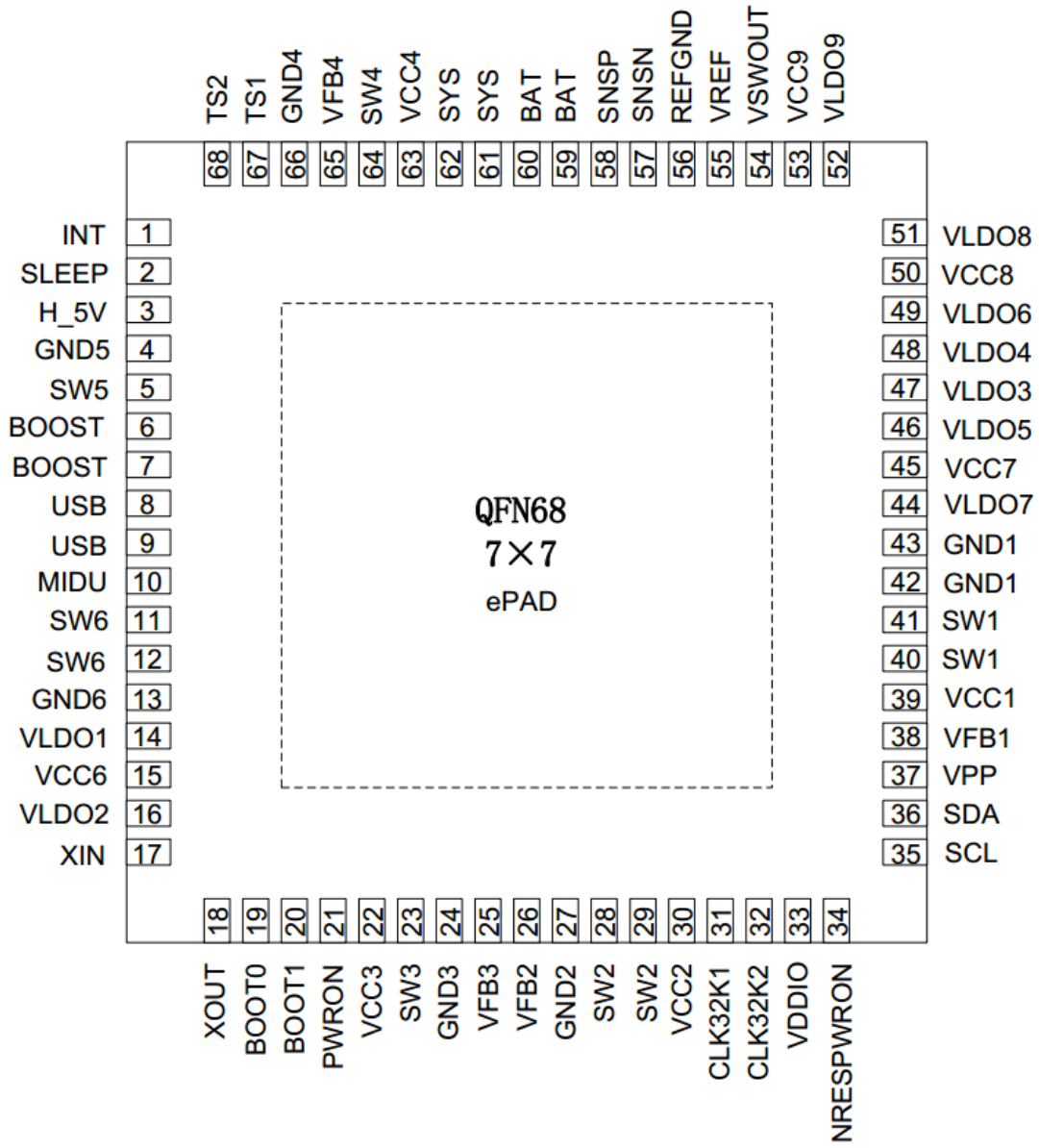
1.2 功能

从使用者的角度看，RK818 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. clk 功能：有两个32.768KHZ时钟输出，一个不可以控常开，一个是软件可控。
4. 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

QFN68 7mm x 7mm, pitch0.35mm



下面描述中，SLEEP 和 INT 引脚需要重点关注：

管脚序号	名称	描述
1	INT	Interrupt request pin. Active low.
2	SLEEP	Input pin for switching state between sleep and non-sleep state.
3	H_5V	5v supply output for HDMI
4	GND5	Power ground

5	SW5	Switch output
6,7	BOOST	BOOST output
8,9	USB	Power input from USB
10	MIDU	Middle point of USB power supply
11,12	SW6	Switch output
13	GND6	Power ground
14	VLDO1	LDO1 output
15	VCC6	Power supply for LDO
16	VLDO2	LDO2 output
17	XIN	32.768KHz crystal oscillator input
18	XOUT	32.768KHz crystal oscillator output
19	BOOT0	Boot sequence selection, low bit
20	BOOT1	Boot sequence selection, high bit
21	PWRON	Power on or power off enable pin, active low, internal 100K pull high to power supply
22	VCC3	Power supply for DCDC3
23	SW3	Switch output of DCDC3
24	GND3	Power ground for DCDC3
25	VFB3	feedback voltage for DCDC3
26	VFB2	DCDC2 output voltage feedback input
27	GND2	Power ground for DCDC2
28,29	SW2	Switch output of DCDC2
30	VCC2	Power supply for DCDC2
31	CLK32K1	32.768K clock1 output, open drain,
32	CLK32K2	32.768K clock2 output, open drain,
33	VDDIO	Power supply for IO
34	NRESPWON	Reset pin after power on, active low
35	SCL	Clock input of I2C
36	SDA	Data input/output of I2C
37	VPP	Power supply for testing, floating in the application
38	VFB1	DCDC1 output voltage feedback input
39	VCC1	Power supply for DCDC1
40,41	SW1	Switch output of DCDC1
42,43	GND1	Power ground for DCDC1
44	VLDO7	LDO7 output
45	VCC7	Power supply for LDO
46	VLDO5	LDO5 output

47	VLDO3	LDO3 output
48	VLDO4	LDO4 output
49	VLDO6	LDO6 output
50	VCC8	Power supply for switch
51	VLDO8	LDO8 output
52	VLDO9	LDO9 output
53	VCC9	Power supply for LDO
54	VSWOUT	Switch output
55	VREF	Internal reference voltage
56	REFGND	Reference ground
57	SNSN	Bat charging and discharging sense current negative pin
58	SNSP	Bat charging and discharging sense current positive pin
59,60	BAT	Positive battery terminal
61,62	SYS	DC-DC regulator output to power the system load and charge the battery
63	VCC4	Power supply for DCDC4
64	SW4	Switch output of DCDC4
65	VFB4	DCDC4 output voltage feedback input
66	GND4	Power ground for DCDC4
67	TS1	Thermistor1 input. Connect a thermistor from this pin to ground. The thermistor is usually inside the battery pack.
68	TS2	Thermistor2 input. Connect a thermistor from this pin to ground. Or it can be used as analog input pin of internal ADC if the control bit is set to ADC function.
Exposed pad	Exposed ground	It must be connected to ground for thermal and electrical enhancement.

1.4 重要概念

- I2C 地址

  7 位从机地址：0x1c

- 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。

- 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 电压范围连续：

    1. DCDC 电压范围连续：

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 ~ 3.3	100	1.8、1.9、2.0、2.2….、3.3

    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：

13 上电启动时序（POWER SEQUENCE）

AP	RK3188/RK3168/ RK3188M/RK3168M/ RK3028A/RK3028 /RK2928		部分 otp/BUCK1~4,LDO3/LDO4 / LDO5/LDO7		RK3066		RK3288/RK3368		S-Product	
BOOT	11		10		01		00			
							RK818-1		RK818-2	
	电压默认值	上电时序	电压默认值	上电时序	电压默认值	上电时序	电压默认值	上电时序	电压默认值	上电时序
BUCK1	1.1V	3	OTP	OTP	1.2V	3	1.1V	3	1.0V	12
BUCK2	1.1V	1	OTP	OTP	1.2V	1	1.1V	1	1.0V	12
BUCK3	x	4	x	OTP	x	4	X	3	X	13
BUCK4	3.0V	1	OTP	OTP	3.0V	1	3.3V	4	3.3V	14
LDO1	3.3V	x	3.3V	x	3.3V	x	3.3V	x	1.8V	11
LDO2	3.0V	x	3V	x	3.0V	x	3.0V	x	X	X

LDO3	1.1V	1	OTP	OTP	1.1V	1	1.1V	x	1.8V	15
LDO4	2.5V	2	OTP	OTP	2.5V	2	2.5V	x	1.8V	1
LDO5	3V	1	OTP	OTP	3.0V	2	1.8V	4	1.8V	11
LDO6	1.2V	x	1.2V	x	1.1V	x	1.1V	x	X	X
LDO7	1.8V	2	OTP	OTP	1.8V	2	1.8V	3	1.1V	15
LDO8	1.8V	x	1.8V	x	1.8V	x	1.8V	x	3.0V	14
LDO9	3.0V	4	3.0V	5	3.0V	4	3.3V	10	1.8V	15
SWITCH	x	x	x	x	x	x	x	10	x	x
OTG	5V	x	5V	x	5V	x	5V	x	5V	x
HDMI_5V	5V	x	5V	x	5V	x	5V	x	5V	x

2 配置

2.1 驱动和 menuconfig

3.10 内核配置

RK818 驱动文件：

```
drivers/mfd/rk818.c
drivers/mfd/rk818-irq.c
drivers/rtc/rtc-rk818.c
drivers/power/rk818-battery.c
```

RK818 dts文件可参考：

```
arch/arm/boot/dts/rk818.dtsi
arch/arm64/boot/dts/rk3368-p9_818.dts
```

menuconfig 里对应的宏配置：

```
CONFIG_MFD_RK818
CONFIG_RTC_RK818
CONFIG_BATTERY_RK818
```

4.4 内核配置

RK818 驱动文件：

```
drivers/mfd/rk808.c
drivers/rtc/rtc-rk808.c
drivers/regulator/rk808-regulator.c
drivers/clk/clk-rk808.c
drivers/power/rk818_charger.c
drivers/power/rk818_battery.c
```

menuconfig 里对应的宏配置：

```
CONFIG_MFD_RK808
CONFIG_RTC_RK808
CONFIG_REGULATOR_RK808
CONFIG_BATTERY_RK818
CONFIG_CHARGER_RK818
CONFIG_COMMON_CLK_RK808
```

4.19 内核配置

RK818 驱动文件：

```
drivers/mfd/rk808.c
drivers/rtc/rtc-rk808.c
drivers/regulator/rk808-regulator.c      // 跟4.4内核不同
drivers/clk/clk-rk808.c
drivers/power/supply/rk818_battery.c
drivers/power/supply/rk818_charger.c
```

menuconfig 里对应的宏配置：

```
CONFIG_MFD_RK808
CONFIG_RTC_RK808
CONFIG_REGULATOR_RK808
CONFIG_BATTERY_RK818
CONFIG_CHARGER_RK818
CONFIG_COMMON_CLK_RK808
```

2.2 DTS 配置

3.10 内核 DTS 配置

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

```
&i2c1 {
    rk818: rk818@1c {
        reg = <0x1c>;
        status = "okay";
    };
};

/include/ "../../../arm/boot/dts/rk818.dtsi"
&rk818 {
    gpios =<&gpio0 GPIO_A1 GPIO_ACTIVE_HIGH>,<&gpio0 GPIO_A0 GPIO_ACTIVE_LOW>;
    rk818,system-power-controller;
    pinctrl-names = "default";
    pinctrl-0 = <&gpio0_c1>;

    regulators {
        rk818_dcdc1_reg: regulator@0{
            regulator-name= "vdd_arm";/*vcc arm*/
            regulator-min-microvolt = <700000>;/*<725000>;*/
            regulator-max-microvolt = <1500000>;
            regulator-initial-mode = <0x2>;
            regulator-initial-state = <3>;
            regulator-state-mem {
                regulator-state-mode = <0x2>;
```

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

        rk818_dcdc2_reg: regulator@1 {
            regulator-name= "vdd_logic";/*vcc gpu*/
            regulator-min-microvolt = <700000>;
            regulator-max-microvolt = <1200000>;
            regulator-initial-mode = <0x2>;
            regulator-initial-state = <3>;
            regulator-state-mem {
                regulator-state-mode = <0x2>;
                regulator-state-enabled;
                regulator-state-uv = <1200000>;
            };
        };
        rk818_dcdc3_reg: regulator@2 {
                    ............
        };
        ................................
    };
};
```

1. I2C 挂载

   整个完整的 rk818 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

```
rk818,system-power-controller：声明RK818具备管理系统下电的功能；
```

- 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

因为 RK808 驱动自动拦截关机命令，执行写I2C关闭PMIC输出。

rk818_shutdown是注册syscore shutdown,用于一些准备工作，如关闭RTC中断等特殊操作。

```
static void rk818_shutdown(void)
{
    int ret;
    struct rk818 *rk818 = g_rk818;

    pr_info("%s\n", __func__);
    ret = rk818_set_bits(rk818, RK818_INT_STS_MSK_REG1,(0x3<<5),(0x3<<5));
//close rtc int when power off
    ret = rk818_clear_bits(rk818, RK818_RTC_INT_REG,(0x3<<2)); //close rtc int
when power off
    /*disable otg_en*/
    ret = rk818_clear_bits(rk818, RK818_DCDC_EN_REG, (0x1<<7));

    mutex_lock(&rk818->io_lock);
    mdelay(100);
}

static struct syscore_ops rk818_syscore_ops = {
    .shutdown = rk818_shutdown,
};
```

rk818_device_shutdown是真正写I2C关闭PMIC输出。

```
void rk818_device_shutdown(void)
{
    int ret, i;
    u8 reg = 0;
    struct rk818 *rk818 = g_rk818;

    for (i = 0; i < 10; i++) {
        pr_info("%s\n", __func__);
        ret = rk818_i2c_read(rk818, RK818_DEVCTRL_REG, 1, &reg);
        if (ret < 0)
            continue;
        ret = rk818_i2c_write(rk818, RK818_DEVCTRL_REG, 1,
                    (reg | (0x1 << 0)));
        if (ret < 0) {
```

```
            pr_err("rk818 power off error!\n");
            continue;
        }
    }
    while(1) wfi();
}
EXPORT_SYMBOL_GPL(rk818_device_shutdown);
```

4.4 内核 DTS 配置

DTS 的配置包括：i2c 挂载、主体、rtc、clk、regulator、 charger、 battery 等部分。

```
&i2c1 {
    status = "okay";
    rk818: pmic@1c {
        compatible = "rockchip,rk818";
        reg = <0x1c>;
        status = "okay";

        clock-output-names = "rk818-clkout1", "wifibt_32kin";
        interrupt-parent = <&gpio0>;
        interrupts = <1 IRQ_TYPE_LEVEL_LOW>;
        pinctrl-names = "default";
        pinctrl-0 = <&pmic_int_l>;
        rockchip,system-power-controller;
        wakeup-source;
        #clock-cells = <1>;

        vcc1-supply = <&vcc_sys>;
        vcc2-supply = <&vcc_sys>;
        vcc3-supply = <&vcc_sys>;
        vcc4-supply = <&vcc_sys>;
        vcc6-supply = <&vcc_sys>;
        vcc7-supply = <&vcc_sys>;
        vcc8-supply = <&vcc_sys>;
        vcc9-supply = <&vcc_io>;

        regulators {
            vdd_logic: DCDC_REG1 {
                regulator-name = "vdd_logic";
                regulator-always-on;
                regulator-boot-on;
                regulator-min-microvolt = <750000>;
                regulator-max-microvolt = <1450000>;
                regulator-ramp-delay = <6001>;
                regulator-state-mem {
                    regulator-on-in-suspend;
                    regulator-suspend-microvolt = <1000000>;
                };
            };

            vdd_gpu: DCDC_REG2 {
                regulator-name = "vdd_gpu";
                regulator-always-on;
                regulator-boot-on;
                regulator-min-microvolt = <800000>;
```

```
                regulator-max-microvolt = <1250000>;
                regulator-ramp-delay = <6001>;
                regulator-state-mem {
                    regulator-on-in-suspend;
                    regulator-suspend-microvolt = <1000000>;
                };
            };
            vcc_ddr: RK818_DCDC3@2 {
                ..................
            };
            ..............................
        };
    };
};
```

1. i2c 挂载

整个完整的 rk818 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,rk818";
reg = <0x1c>;
rockchip,system-power-controller;
wakeup-source;
#clock-cells = <1>;
```

- 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

如果 menuconfig 选中了这个模块，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。

5. poweroff 部分

4.4上使用pm_power_off_prepare，实现PMIC关机前的准备工作，如关闭RTC中断，配置一些特殊寄存器等。
注册syscore shutdown, 真正用于PMIC关机。

6. clk 部分

如果某个节个需要引用 RK808 的 clk 进行使用，引用格式如下：

`clocks = <&rk818 1>;`
第一个参数： &rk818 固定，不可改动；
第二个参数： 引用 rk818 的哪个 clk，只能是 0 或者 1，其中 0：rk818-clkout1， 1：rk818-clkout2；

4.19 内核 DTS 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 = <&rk818 0 GPIO_ACTIVE_LOW>;`的方式引用和使用rk818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

3.10 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 rk818 的寄存器，通过如下节点：

`/sys/rk818/rk818_test`

读寄存器：

`echo r [addr]  > /sys/rk818/rk818_test`

写寄存器：

`echo w [addr] [value] > /sys/rk818/rk818_test`

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`

4.19 内核

请参考4.4内核命令。