RK808 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 RK808 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
RK808	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

RK808 开发指南

1 基础

1.1 概述

1.2 功能

1.3 芯片引脚功能

1.4 重要概念

1.5 上电条件和时序

2 配置

2.1 驱动和 menuconfig

3.10 内核配置

4.4 内核配置

4.19 内核配置

2.2 DTS 配置

3.10 内核 DTS 配置

4.4 内核 DTS 配置

4.19 内核 DTS 配置

2.3 函数接口

3 Debug

3.10 内核
4.4 内核
4.19 内核

1 基础

1.1 概述

RK808 是一款高性能 PMIC，RK808 集成 4 个大电流 DCDC、8 个 LDO、2个开关SWITCH、1 个 RTC、可调上电时序等功能。

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）：

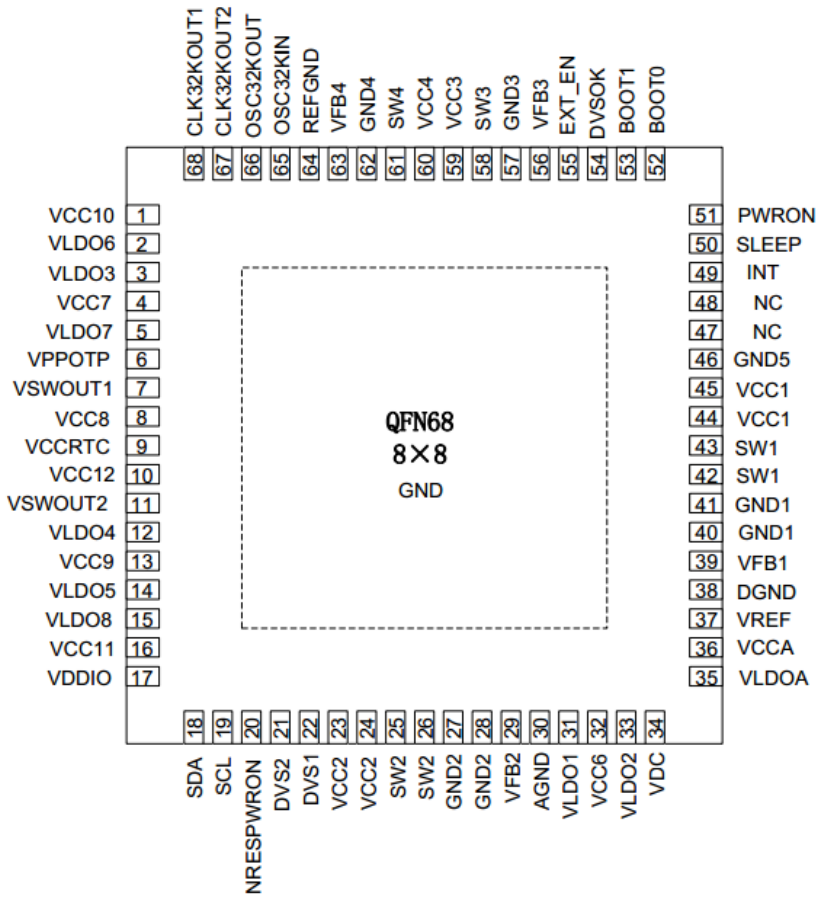
1. DCDC：输入输出压差大时，效率高，但是存在纹波比较大的问题，成本高，所以大压差，大电流负载时使用。一般有两种工作模式。PWM 模式：纹波瞬态响应好，效率低；PFM 模式：效率高，但是负载能力差。
2. LDO：输入输出压差大时，效率低，成本低，为了提高 LDO 的转换效率，系统上会进行相关优化如：LDO 输出电压为 1.1V，为了提高效率，其输入电压可以从 VCCIO_3.3V 的 DCDC 给出。所以电路上如果允许尽量将 LDO 接到 DCDC 输出回路，但是要注意上电时序。

1.2 功能

从使用者的角度看，RK808 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。

1.3 芯片引脚功能



下面描述中，SLEEP 和 INT 引脚需要重点关注：

NO	NAME	SUPPLIES	FUNCTIONAL BLOCK	TYPE	I/O	DESCRIPTION	PU/PD
9	VCCRTC	VCCRTC /AGND	RTC	Power	O	RTC power supply	NO
65	OSC32KIN	VCCRTC /DGND		Analog	I	32KHz crystal oscillator input	NO
66	OSC32KOUT	VCCRTC /DGND		Analog	I	32KHz crystal oscillator output	NO
68	CLK32KOUT1	VCCRTC /DGND		Digital	O	32KHz clock output 1,OD output (always on)	NO
67	CLK32KOUT2	VCCRTC /DGND		Digital	O	32KHz clock output 2,OD output	PD
37	VREF	VCCA /REFGND	REFERENCE	Analog	O	bandgap voltage	PD
64	VREFGND	REFGND		Analog	Gnd	reference ground	NO
36	VCCA	VCCA /GNDA	Analog Power	Power	I	power supply for	NO
6	VPPOTP	VPPOTP /GNDA	Analog Power	Power	I	OTP power supply	NO
45	VCC1	VCC1 /GND1	BUCK1	Power	I/O	buck1 dc-dc power supply	NO
44	VCC1	VCC1 /GND1		Power	I/O	buck1 dc-dc power supply	NO
43	SW1	VCC1 /GND1		Power	I/O	buck1 dc-dc switch output	PD
42	SW1	VCC1 /GND1					
41	GND1	VCC1 /GND1		Power	Gnd	buck1 dc-dc switch ground	NO
40	GND1	VCC1 /GND1					
39	VFB1	VCC1 /REFGND		Analog	I	buck1 dc-dc switch feedback voltage	PD
23	VCC2	VCC2 /GND2	BUCK2	Power	I	buck2 dc-dc power supply	NO

24	VCC2	VCC2 /GND2		Power	I	buck2 dc-dc power supply	NO
25	SW2	VCC2 /GND2		Power	I/O	buck2 dc-dc switch output	PD
26	SW2	VCC2 /GND2		Power	I/O	buck2 dc-dc switch output	PD
27	GND2	VCC2 /GND2		Power	Gnd	buck2 dc-dc switch ground	NO
28	GND2	VCC2 /GND2		Power	Gnd	buck2 dc-dc switch ground	NO
29	VFB2	VCC2 /REFGND		Analog	I	buck2 dc-dc switch feedback voltage	PD
59	VCC3	VCC3 /GND3	BUCK3	Power	I	buck3 dc-dc power supply	NO
58	SW3	VCC3 /GND3		Power	I/O	buck3 dc-dc switch output	PD
57	GND3	VCC3 /GND3		Power	Gnd	buck3 dc-dc switch ground	NO
56	VFB3	VCC3 /REFGND		Analog	I	buck3 dc-dc switch feedback voltage	PD
60	VCC4	VCC4 /GND4	BUCK4	Power	I	buck4 dc-dc power supply	NO
61	SW4	VCC4 /GND4		Power	I/O	buck4 dc-dc switch output	PD
62	GND4	VCC4 /GND4		Power	Gnd	buck4 dc-dc switch ground	NO
63	VFB4	VCC4 /REFGND		Analog	I	buck4 dc-dc switch feedback voltage	PD
47	NC						
46	GND5	VCCA /GND5		Power	Gnd	ground	NO
48	NC						
32	VCC6	VCC6 /AGND	LDO 1~8, SWITCH1,2	Power	I	LDO1,LDO2 power supply	NO
4	VCC7	VCC7 /AGND		Power	I	LDO3,LDO7 power supply	NO
8	VCC8	VCC8 /AGND		Power	I	SWITCH1 power supply	NO

13	VCC9	VCC9 /AGND		Power	I	LDO4,LDO5 power supply	NO
1	VCC10	VCC11 /AGND		Power	I	LDO6 power supply	NO
16	VCC11	VCC11 /AGND		Power	I	LDO8 power supply	NO
10	VCC12	VCC12 /AGND		Power	I	SWITCH2 power supply	NO
31	VLDO1	VCC7 /AGND		Power	O	LDO1 regulator output	PD
33	VLDO2	VCC7 /AGND		Power	O	LDO2 regulator output	PD
3	VLDO3	VCC8 /AGND		Power	O	LDO3 regulator output	PD
12	VLDO4	VCC9 /AGND		Power	O	LDO4 regulator output	PD
14	VLDO5	VCC10 /AGND		Power	O	LDO5 regulator output	PD
2	VLDO6	VCC9 /AGND		Power	O	LDO6 regulator output	PD
5	VLDO7	VCC1 1/AGND		Power	O	LDO7 regulator output	PD
15	VLDO8	VCC11 /AGND		Power	O	LDO8 regulator output	PD
7	VSWOUT1	VCC8 /AGND		Power	O	Switch 1 output	PD
11	VSWOUT2	VCC12 /AGND		Power	O	Switch 2 output	PD
30	AGND	POWER PAD	Analog ground	Power	Gnd	Analog ground	NO
35	VLDOA	POWER PAD	LDOA	Power	I/O	supply for internal analog circuit	NO
38	DGND	POWER PAD	Digital ground	Power	Gnd	Digital ground	NO
17	VDDIO	VDDIO /DGND		Power	I/O	Digital I/O power supply	NO
50	SLEEP	VDDIO /DGND	IO	Digital	I	Active-Sleep state transition control signal	NO
20	NRESPWRON	VDDIO /DGND		Digital	I/O	Power off reset for AP/ External reset digital core(excludes RTC)	PD in power-off state

49	INT	VDDIO /DGND	IO	Digital	O	Interrupt flag (polarity is I2C programmable, default active high)	Programmable PU/PD
51	PWRON	VCCRTC /DGND		Digital	I	External switch-on control signal(ON button)	NO
18	SDA	VDDIO /DGND		Digital	I/O	I2C data signal	NO
19	SCL	VDDIO /DGND		Digital	I/O	I2C clock signal	NO
52	BOOT0	VCCRTC /DGND	IO	Digital	I	Power-up sequence selection	NO
53	BOOT1	VCCRTC /DGND		Digital	I	Power-up sequence selection	NO
55	EXT_EN	VCCRTC /DGND		Digital	O	Output enable for external BUCK in two-battery-cells application	PD
22	DVS1	VDDIO /DGND		Digital	I	BUCK1 DVS voltage /normal voltage transition control signal(polarity is I2C programmable, default active high)	NO
21	DVS2	VDDIO /DGND		Digital	I	BUCK2 DVS voltage /normal voltage transition control signal(polarity is I2C programmable, default active high)	NO
54	DVSOK	VDDIO /DGND		Digital	O	BUCK1 and BUCK2 power good flag after dynamic voltage setting	PD
34	VDC	VDC /AGND		Digital	I	Adapter voltage detect input	NO

1.4 重要概念

- I2C 地址

  7 位从机地址：0x1b

- 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：

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：

9 POWER SEQUENCE

	Power On Sequence	Preset Voltage	Power On Sequence	Preset Voltage	Power On Sequence	Preset Voltage	Power On Sequence	Preset Voltage
Boot1, Boot0	00		01		10		11	
BUCK1	4	1.1V/ON	4	1.2V/ON	4	1.0V/ON	OTP	OTP
BUCK2	5	1.1V/ON	5	1.2V/ON	4	1.0V/ON	OTP	OTP
BUCK3	2	1.2V/ON	2	1.2V/ON	3	1.2VON	OTP	OTP
BUCK4	1	3.0V/ON	1	3.0V/ON	1	3.0V/ON	OTP	OTP
LDO1		3.3V/OFF		3.3V/OFF	1	3.3v/ON	OTP	OTP
LDO2		3.3V/OFF	2	3.3V/ON		3.3V/OFF	OTP	OTP
LDO3	3	1.1V/ON	3	1.2V/ON	2	1.0V/ON	OTP	OTP
LDO4	3	2.5V/ON		2.5V/OFF	2	1.8V/ON	OTP	OTP
LDO5		2.8V/OFF		2.8V/OFF		2.8V/OFF	OTP	OTP
LDO6		1.2V/OFF		1.2V/OFF		1.2V/OFF	OTP	OTP
LDO7		1.8/OFF		1.8V/OFF		1.8V/OFF	OTP	OTP
LDO8		3.3V/OFF		1.8V/OFF		3.3V/OFF	OTP	OTP

SWITCH1	1	3.0V/ON	1	3.0V/ON	5	3.0V/ON	OTP	OTP
SWITCH2		3.0V/OFF		3.0V/OFF		3.0V/OFF	OTP	OTP

2 配置

2.1 驱动和 menuconfig

3.10 内核配置

RK808 驱动文件：

```
drivers/mfd/rk808.c
drivers/mfd/rk808-irq.c
drivers/rtc/rtc-rk808.c
```

RK808 dts文件可参考：

```
arch/arm/boot/dts/rk808.dtsi
arch/arm/boot/dts/rk3288-evb-android-rk808-edp.dts
```

menuconfig 里对应的宏配置：

```
CONFIG_MFD_RK808
CONFIG_RTC_RK808
```

4.4 内核配置

RK808 驱动文件：

```
drivers/mfd/rk808.c
drivers/rtc/rtc-rk808.c
drivers/regulator/rk808-regulator.c
drivers/clk/clk-rk808.c
```

RK808 dts文件可参考：

```
arch/arm64/boot/dts/rockchip/rk3399-evb-rev3.dtsi
```

menuconfig 里对应的宏配置：

```
CONFIG_MFD_RK808
CONFIG_RTC_RK808
CONFIG_REGULATOR_RK808
CONFIG_COMMON_CLK_RK808
```

4.19 内核配置

RK808 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
```

menuconfig 里对应的宏配置：

```
CONFIG_MFD_RK808
CONFIG_RTC_RK808
CONFIG_REGULATOR_RK808
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 {
    rk808: rk808@1b {
        reg = <0x1b>;
        status = "okay";
    };
};

/include/ "rk808.dtsi"
&rk808 {
    gpios = <&gpio0 GPIO_A4 GPIO_ACTIVE_HIGH>,
            <&gpio0 GPIO_B3 GPIO_ACTIVE_LOW>;
    rk808,system-power-controller;
```

```
    rtc {
        status = "disabled";
    };

    regulators {
        rk808_dcdc1_reg: regulator@0 {
            regulator-always-on;
            regulator-boot-on;
            regulator-min-microvolt = <750000>;
            regulator-max-microvolt = <1400000>;
            regulator-init-microvolt = <1300000>;
            regulator-name = "vdd_arm";
            regulator-state-mem {
                regulator-off-in-suspend;
            };
        };
        rk808_dcdc2_reg: regulator@1 {
                    ............
        };
        rk808_dcdc3_reg: regulator@2 {
                    ............
        };
        ................................
    };
};
```

1. I2C 挂载

   整个完整的 rk808 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

```
rk808,system-power-controller：声明RK808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。

rk808_shutdown是注册syscore shutdown,用于一些准备工作，如打印关机电压，关闭RTC中断等。

```
static void rk808_shutdown(void)
{
    int ret,i,val;
    u16 reg = 0;
    struct rk808 *rk808 = g_rk808;

    printk("%s\n",__func__);
    /***************get dc1\dc2 voltage **********************/
    for(i=0;i<2;i++){
    reg = rk808_reg_read(rk808,rk808_BUCK_SET_VOL_REG(i));
    reg &= BUCK_VOL_MASK;
    val = 712500 + reg * 12500;
    printk("%s,line=%d dc[%d]= %d\n", __func__,__LINE__,(i+1),val);
    }
    /*****************************************************/
    ret = rk808_set_bits(rk808, RK808_INT_STS_MSK_REG1,(0x3<<5),(0x3<<5));
//close rtc int when power off
    ret = rk808_clear_bits(rk808, RK808_RTC_INT_REG,(0x3<<2)); //close rtc int
when power off
    mutex_lock(&rk808->io_lock);
    mdelay(100);
}

static struct syscore_ops rk808_syscore_ops = {
    .shutdown = rk808_shutdown,
};
```

rk808_device_shutdown是真正写I2C关闭PMIC输出。

```
static void rk808_device_shutdown(void)
{
    int ret,i;
    u8 reg = 0;
```

```
    struct rk808 *rk808 = g_rk808;
    for(i=0;i < 10;i++){
        printk("%s\n",__func__);
        ret = rk808_i2c_read(rk808,RK808_DEVCTRL_REG,1,&reg);
        if(ret < 0)
            continue;
        ret = rk808_i2c_write(rk808, RK808_DEVCTRL_REG, 1,(reg |(0x1 <<3)));
        if (ret < 0) {
            printk("rk808 power off error!\n");
            continue;
        }
    }
    while(1)wfi();
}
EXPORT_SYMBOL_GPL(rk808_device_shutdown);
```

4.4 内核 DTS 配置

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

```
&i2c1 {
    status = "okay";
    rk808: pmic@1b {
        compatible = "rockchip,rk808";
        reg = <0x1b>;
        interrupt-parent = <&gpio1>;
        interrupts = <21 IRQ_TYPE_LEVEL_LOW>;
        pinctrl-names = "default";
        pinctrl-0 = <&pmic_int_l &pmic_dvs2>;
        rockchip,system-power-controller;
        wakeup-source;
        #clock-cells = <1>;
        clock-output-names = "rk808-clkout1", "rk808-clkout2";

        vcc1-supply = <&vcc3v3_sys>;
        vcc2-supply = <&vcc3v3_sys>;
        vcc3-supply = <&vcc3v3_sys>;
        vcc4-supply = <&vcc3v3_sys>;
        vcc6-supply = <&vcc3v3_sys>;
        vcc7-supply = <&vcc3v3_sys>;
        vcc8-supply = <&vcc3v3_sys>;
        vcc9-supply = <&vcc3v3_sys>;
        vcc10-supply = <&vcc3v3_sys>;
        vcc11-supply = <&vcc3v3_sys>;
        vcc12-supply = <&vcc3v3_sys>;
        vddio-supply = <&vcc1v8_pmu>;

        regulators {
            vdd_log: DCDC_REG1 {
                regulator-always-on;
                regulator-boot-on;
                regulator-min-microvolt = <750000>;
                regulator-max-microvolt = <1350000>;
                regulator-ramp-delay = <6001>;
                regulator-name = "vdd_log";
                regulator-state-mem {
```

```
                    regulator-on-in-suspend;
                    regulator-suspend-microvolt = <900000>;
                };
            };
            vdd_cpu_l: DCDC_REG2 {
                regulator-always-on;
                regulator-boot-on;
                regulator-min-microvolt = <750000>;
                regulator-max-microvolt = <1350000>;
                regulator-ramp-delay = <6001>;
                regulator-name = "vdd_cpu_l";
                regulator-state-mem {
                    regulator-off-in-suspend;
                };
            };
            vcc_ddr: DCDC_REG3 {
                ..................
            };
            ..............................
        };
    };
};
```

1. i2c 挂载

整个完整的 rk808 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,rk808";
reg = <0x1b>;
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 = <&rk808 1>;`
第一个参数： &rk808 固定，不可改动；
第二个参数： 引用 rk808 的哪个 clk，只能是 0 或者 1，其中 0：rk808-clkout1， 1：rk808-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 = <&rk808 0 GPIO_ACTIVE_LOW>;` 的方式引用和使用rk808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 rk808 的寄存器，通过如下节点：

`/sys/rk808/rk808_test`

读寄存器：

`echo r [addr]  > /sys/rk808/rk808_test`

写寄存器：

`echo w [addr] [value] > /sys/rk808/rk808_test`

一般写操作执行完之后最好再读一遍确认是否写成功。

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。