密级状态：绝密(　　)　　秘密(　　)　　内部资料(　　)　　公开(√)

Rk32xx 平台 LINUX3.10 上 pmu 的使用说明

(技术研发部，手机组)

文件状态： ［　］草稿 ［√］正式发布 ［　］正在修改	文件标识：	Rk32xx 平台 LINUX3.10 上 pmu 的使用说明
	当前版本：	1.1
	作　　者：	张晴
	完成日期：	2014-6-13

版 本 历 史

版本号	作者	修改日期	修改说明
1.0	张晴	2014-6-8	初稿
1.1	张晴	2014-6-13	增加 ricoh619 充电及电量计使用指导

目录

1 硬件原理介绍

1.1 pmic 电路原理介绍

电源类型概述：

系统中各路电源总体分为两种：DCDC、LDO。两种电源的总体特性如下，详细资料--百度。

DCDC：输入输出压差大时，效率高，但是有纹波问题，成本高，所以大压差，大电流时使用。

LDO ：输入输出压差大时，效率低，陈本低

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。

DCDC 一般有两种工作模式：PWM--纹波瞬态响应好,效率低；PFM：效率高，但是负载能力差。

Rk32 平台概述：

目前平台有几款可用的 PMIC：ricoh619(5 路 DCDC),rk808,act8846。这几款 PMIC 的 DCDC 路数不相同，现在产品上分配如下：

方案一(RK808+DCDC(PWM 控制))

DCDC1(1.0)： ARM

DCDC2(1.0)： GPU

DCDC3(1.2): DDR

DCDC4(3.3): VCCIO

DCDC5(1.0)(分立 PWM): LOGIC

方案二（PMIC+2*SYR82X)

DCDC1(1.0)： ARM (SYB827)

DCDC2(1.0)： GPU (SYB828)

DCDC3(1.0): LOGIC (PMIC)

DCDC4(1.2): DDR(PMIC)

DCDC5(2.0): VCC20(PMIC)

DCDC6(3.3): VCCIO(PMIC)

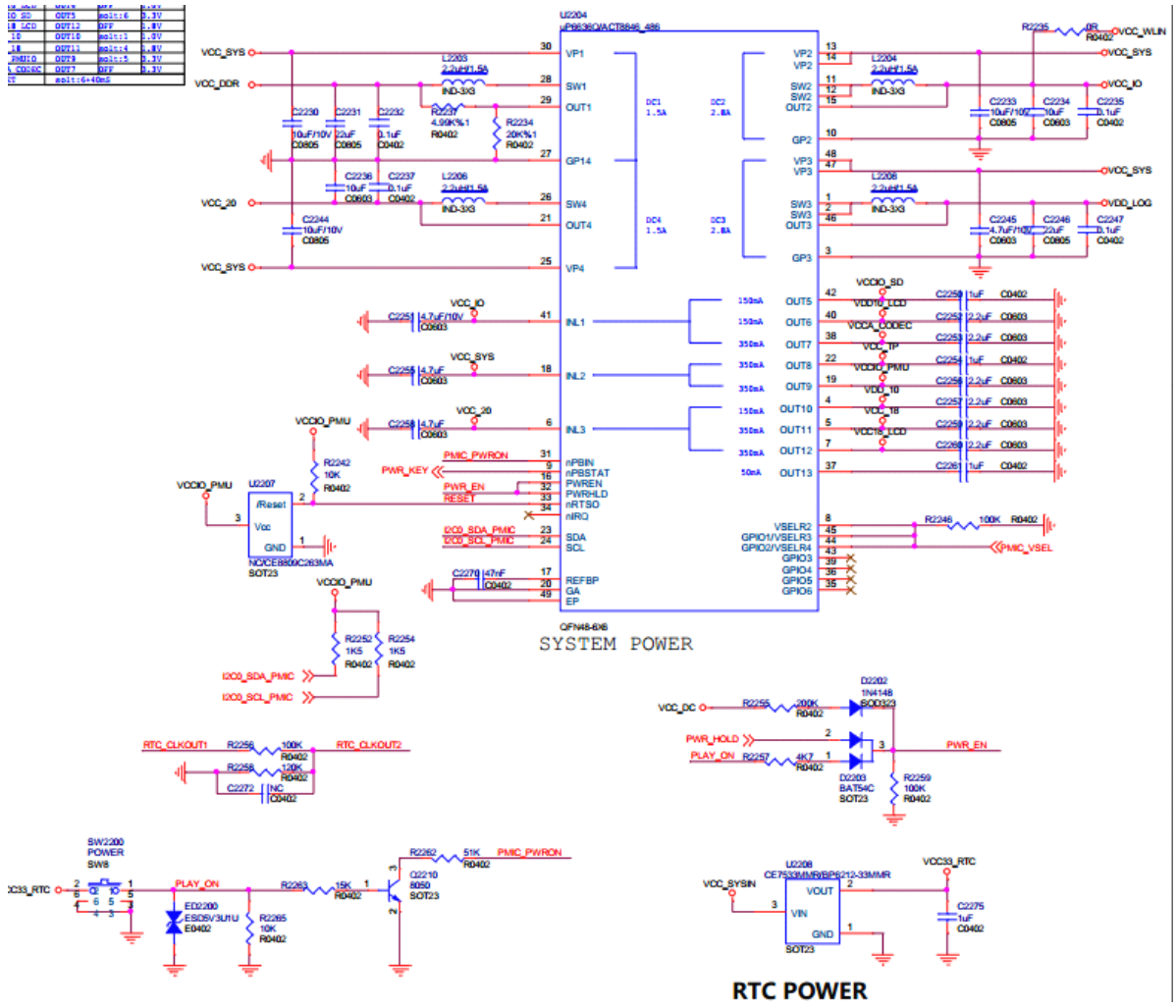
方案一，由于 logic 需要动态调压，如果用分立的 DCDC（PWM），调节精度、输出电压的一致性都不能够保证，不同机器差别很大（PWM 调压原理会讲到）。

对于上述不同的方案需要软件需要进行一些修改，详见软件配置。

目前双节串联电池方案才有 ACT8846+SYR82X

单节电池方案采用 RK808 或者 RICOH619+SYR82X

1.1.1 ACT8846 电路简介：



图一： act8846 PMIC 电路图

供电回路分布：

SW1~4 为 DCDC，OUT5~13 为 LDO。

Act8846 的 LDO 的输入端为 INL1~3（具体看电路）。

各路上电默认电压及时序：

Resource	SW1	SW2	SW3	SW4	OUT5	OUT6	OUT7
VOL	1.2V	3.3V	1.1V	2.0V	3.3V	1.0V	3.3V
Time Slot	2	6	3	0	6	OFF	OFF
Resource	**OUT8**	**OUT9**	**OUT10**	**OUT11**	**OUT12**	**OUT13**	
VOL	3.3V	3.3V	1.0V	1.8V	1.8V	1.8V	
Time Slot	OFF	5	1	4	OFF	ALWAYS ON	

各路电流限制及电压调整范围

SW1：1500MA，0.6~3.9V

SW2：2800MA，0.6~3.9V

SW3：2800MA，0.6~3.9V

SW4：1500MA，0.6~3.9V

OUT5：150MA，0.6~3.9V

OUT6：150MA，0.6~3.9V

OUT7：350MA，0.6~3.9V

OUT8：350MA，0.6~3.9V

OUT9：350MA，0.6~3.9V

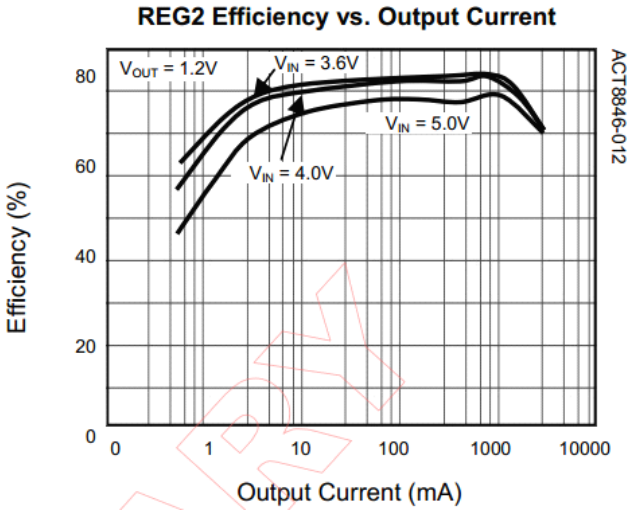
OUT10：150MA，0.6~3.9V

OUT11：350MA，0.6~3.9V

OUT12：350MA，0.6~3.9V

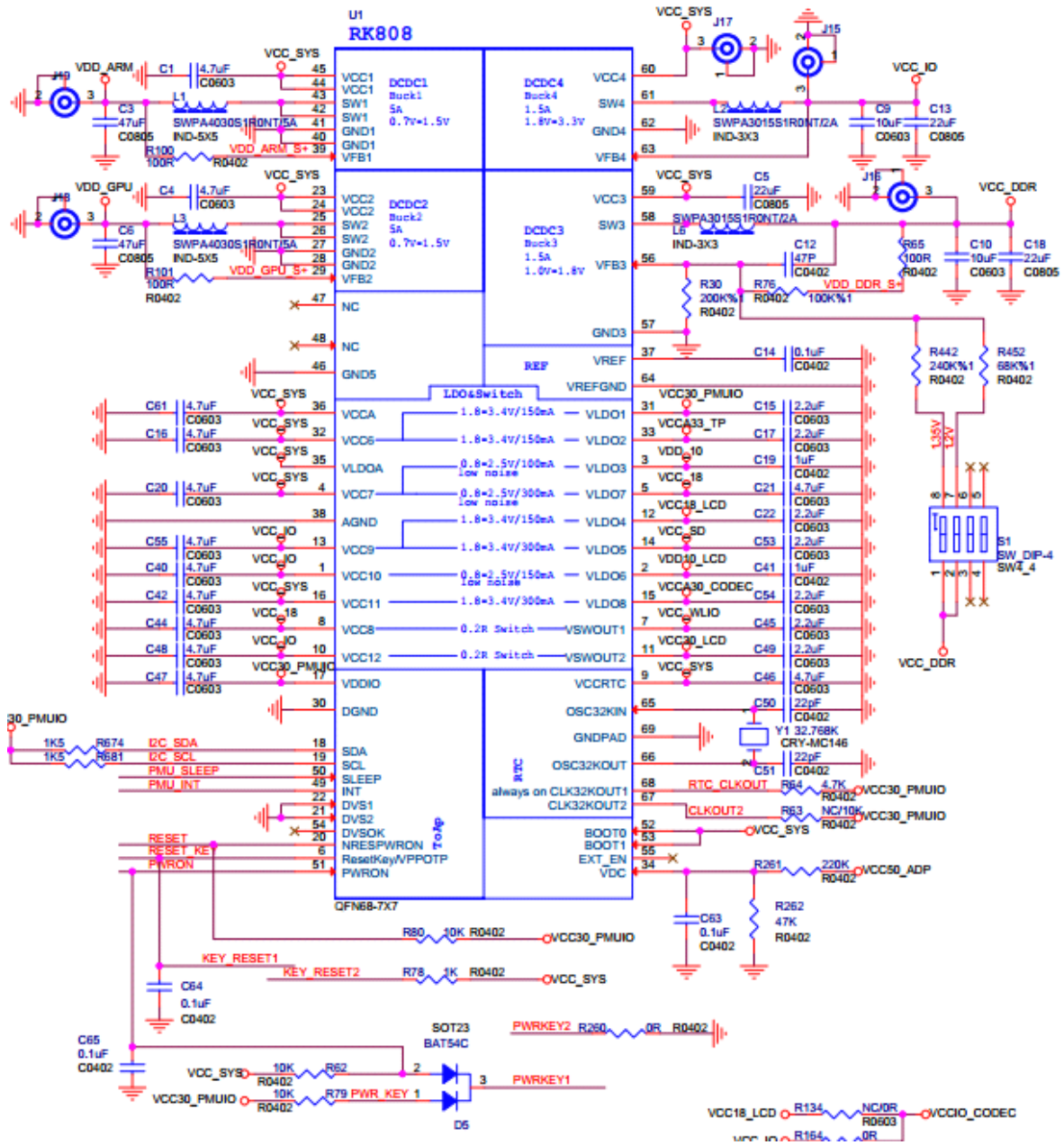
所有路的步进如下：0.6~1.2V（25MV），1.2~2.4V（50MV），2.4~3.9V(100MV)

效率、纹波数据



其他路详见 DATASHEET。

1.1.2 RK808 电路简介：



图二： RK808 PMIC 电路图

供电回路分布：

VCC1~4 为 DCDC，SW5 为升压 BOOST，VLDO1~8 为 LDO,VSWOUT1~2 为两路 SWITCH。RK808 的 LDO 的输入端为 VCC6~12 （具体看电路）。

各路上电默认电压及时序：

Resource	VCC1	VCC2	VCC3	VCC4	LDO1	LDO2	LDO3
VOL	1.0V	1.0V	1.2V	3.3V	3.3V	3.3V	1.0V
Time Slot	2	2	4	2	1	OFF	2
Resource	LDO4	LDO5	LDO6	LDO7	LDO8	VSWOUT1	VSWOUT2

VOL	1.8V	3.3V	1.2V	1.8V	3.3V	3.0V	3.0V
Time Slot	OFF	5	OFF	3	OFF	OFF	OFF

各路电流限制及电压调整范围

VCC1：3000MA，0.7~1.5V

VCC2：3000MA，0.7~1.5V

VCC3：1500MA，1.0~1.8V

VCC4：1000MA，1.8~3.3V

BOOST: 800MA, 5V

LDO1：150MA，1.8~3.4V

LDO2：150MA，1.8~3.4V

LDO3：100MA，0.8~2.5V

LDO4：150MA，1.8~3.4V

LDO5：300MA，1.8~3.4V

LDO6：150MA，0.8~2.5V

LDO7：300MA，0.8~2.5V

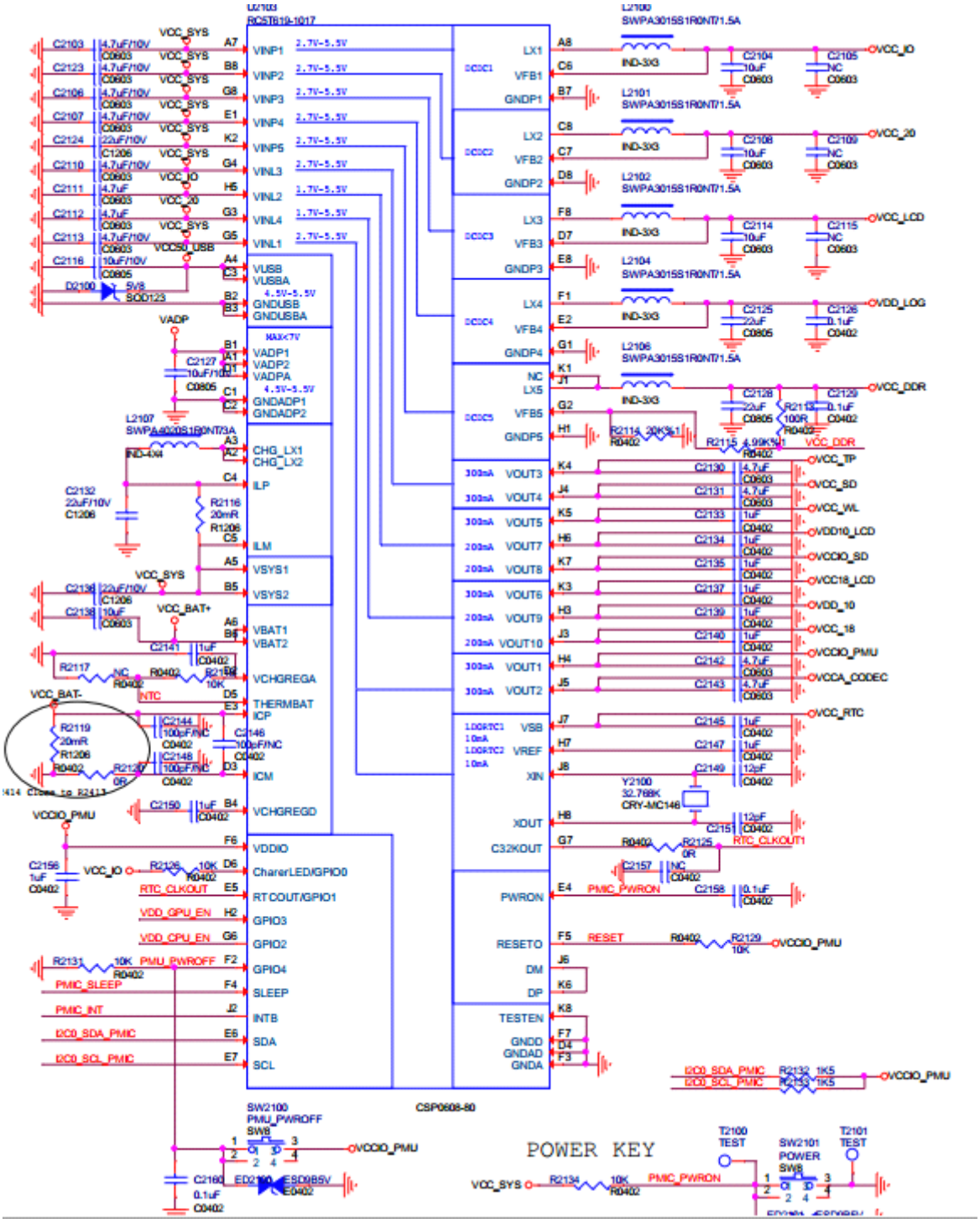
LDO8：300MA，1.8~3.4V

步进如下：DCDC 步进 12.5MV，LDO 步进 100MV

效率、纹波数据

详见 DATASHEET。

1.1.3 RICOH619 电路简介：



图三： RICOH619 PMIC 电路图

供电回路分布：

LX1~5 为 DCDC，VOUT1~10 为 LDO。

RICOH619 的 LDO 的输入端为 VINL1~4 （具体看电路）。

各路上电默认电压及时序：

Resource	LX1	LX2	LX3	LX4	LX5	LDO1	LDO2	LDO3
VOL	3.3V	2.0V	3.3V	1.0V	1.2V	3.3V	3.3V	3.3V

Time Slot	8	0	OFF	5	4	7	OFF	OFF
Resource	LDO4	LDO5	LDO6	LDO7	LDO8	LDO9	LDO10	
VOL	3.0V	1.8V	1.8V	1.0V	3.0V	1.0V	1.8V	
Time Slot	9	OFF	OFF	OFF	9	1	6	

各路电流限制及电压调整范围

VCC1：3000MA，0.6~3.5V

VCC2：3000MA，0.6~3.5V

VCC3：3000MA，0.6~3.5V

VCC4：2000MA，0.6~3.5V

VCC5：2000MA，0.6~3.5V

LDO1：300MA，0.9~3.5V, ECO

LDO2：300MA，0.9~3.5V, ECO

LDO3：300MA，0.9~3.5V, ECO

LDO4：300MA，0.9~3.5V, ECO

LDO5：300MA，0.6~3.5V, ECO

LDO6：300MA，0.6~3.5V, ECO

LDO7：200MA，0.9~3.5V

LDO8：200MA，0.9~3.5V

LDO9：200MA，0.9~3.5V

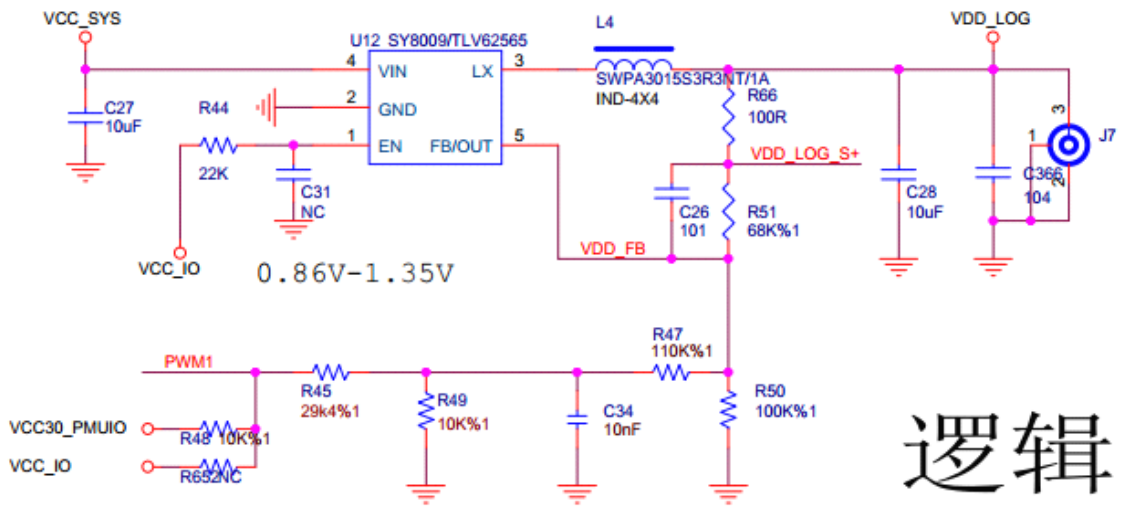
LDO10：200MA，0.9~3.5V

步进如下：DCDC 步进 12.5MV，LDO 步进 25MV

效率、纹波数据

详见 DATASHEET。

1.1.4 分立 DCDC 电路介绍：



图四： PWM 电路图

如上所述：目前 rk32 平台分立 DCDC 有给 LOGIC 供电。通过 PWM 占空比控制调压。

PWM 调整默认电压：

如果使用 PWM 调整 logic 电压，下面参数为硬件电阻控制：

电压调节范围：PWM 固定为高时对应最低电压，否则相反；目前平台上上限值 1.4v，下限 0.9v，受电阻决定

默认启动电压：目前 32 平台上 1.0v，调整方法：由于上电时 PWM1 这个 IO 是默认 GPIO 功能，内部下拉，外部用 VCCIO(或者 VCC30_PMUIO)上拉，主要受上拉电阻 r48 阻值影响。

备注：Reset 后 PWM_LOG 恢复成默认的 GPIO，ARM 电压恢复成默认值。

1.2 PMIC 相关 pin 定义及测试

参照图一~四中的 io。

通用 Io 介绍：

PMIC_sleep: 用于 PMIC 进入 sleep 模式，此 IO 口必须是内部下拉，不要有复用功能，此 IO 口正常情况下是低的，拉高进入 sleep 模式，进入 sleep 模式后会降压个关闭部分电压。ACT846 目前没有使用此功能。

PMIC_hold: 用于维持 PMU 电源，开机键按下后，PMIC 会自动维持一段时间，等芯片送 power_hold 为高后锁住 PMIC 的电源，即可松开按键了。

PMIC_int:：用于 PMIC 的中断，包括 RTC 闹铃、过流、过压、低电、充电等。正常为低，中断触发时为高。

VSEL:用于 DCDC 内部两组寄存器值切换。

Pin 脚 dts 解析在 rk3288-tb.dts 中（以 ricoh619 为例）：

gpios =<&gpio0 GPIO_A4 GPIO_ACTIVE_HIGH>,<&gpio0 GPIO_B3 GPIO_ACTIVE_LOW>,<&gpio0 GPIO_B0 GPIO_ACTIVE_HIGH>;

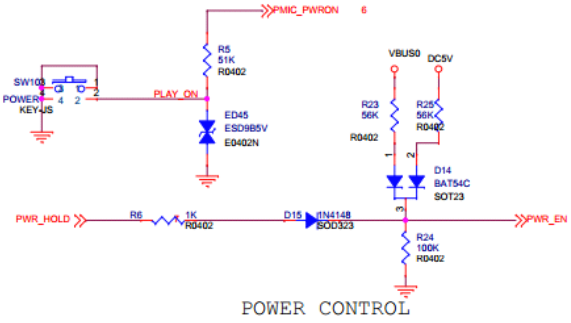
gpios 0 :irq pin gpio

gpios 1 :sleep pin gpio

gpios 2 :dc det pin

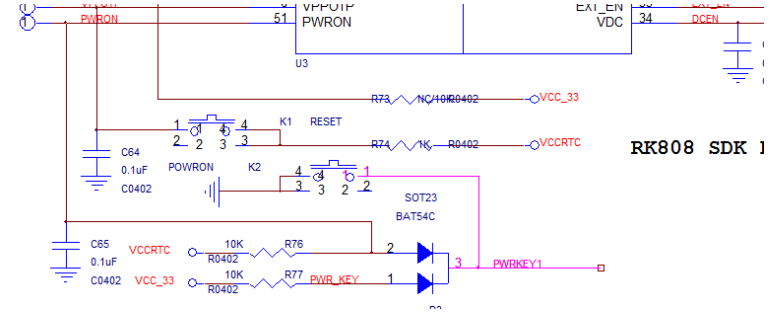
1.3 Pmic 开机流程

Act8846 开机流程：



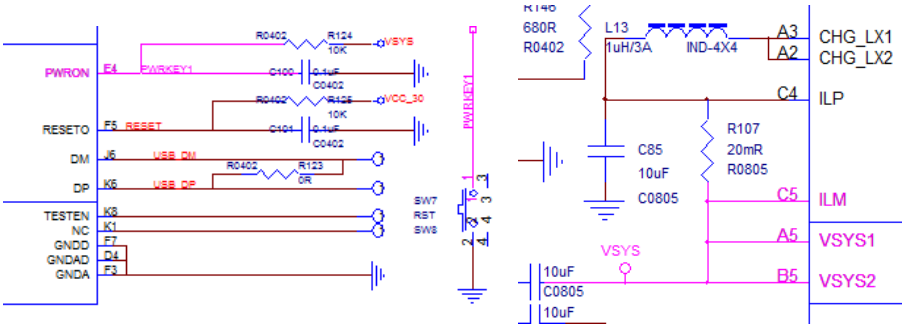
当接有适配器时，PMIC_ON 便被拉高，VSYS 上电，power_hold 为高，PMIC 电源自动锁住输出。当只接电池时，此时按开机键，PMIC 默认保持电源一段时间，等待 PWR_HOLD 拉高，从而保证 PMIC 一直供电。PMIC 启动，实现从 EPROM 中读取默认设置，各路上电完成后，发送 reset 信号，芯片上电，系统启动，PMIC 设备挂载，通过 I2C 重新配置 PMIC。

RK808 开机流程：



当接有适配器时，VDC 电压便被拉高，VSYS 上电，PMIC 上电并输出。当只接电池时，此时按开机键，PWRON 拉高，PMIC 上电并输出。PMIC 启动，实现从 EPROM 中读取默认设置，各路上电完成后，发送 reset 信号，芯片上电，系统启动，PMIC 设备挂载，通过 I2C 重新配置 PMIC。

RICOH619 开机流程：



当接有适配器时，由于 619 是具有充电 IC，并带有路径管理功能，所以充电器可以直接给 VSYS 供电，在 VSYS 电压大于 3.4V 时自动开机，PMIC 上电并输出。当只接电池时，此时按开机键，PWRON 拉高，PMIC 上电并输出。PMIC 启动，实现从 EPROM 中读取默认设置，各路上电完成后，发送 reset 信号，芯片上电，系统启动，PMIC 设备挂载，通过 I2C 重新配置 PMIC。

2 软件驱动搭建介绍

2.1 Regulator 注册

（1）Regulator 的 name 注册(act8846.dtsi rk808.dtsi ricoh619.dtsi)。

以 ricoh619 为例：

Ricoh619.dtsi:

```
ricoh619_dcdc1_reg: regulator@0 {
        reg = <0>;
        regulator-compatible = "ricoh619_dc1";
        regulator-always-on;
        regulator-boot-on;
};
```

Rk3288-tb.dts:

```
&ricoh619 {
    gpios =<&gpio0 GPIO_A4 GPIO_ACTIVE_HIGH>,<&gpio0 GPIO_B3 GPIO_ACTIVE_LOW>,<&gpio0 GPIO_B0 GPIO_ACTIVE_HIGH>
    ricoh619,system-power-controller;

    regulators {
        ricoh619_dcdc1_reg: regulator@0 {
            regulator-name = "vdd_logic";
            regulator-min-microvolt = < 700000>;
            regulator-max-microvolt = <1500000>;
            regulator-initial-mode = <0x2>;
            regulator-initial-state = <3>;
            regulator-state-mem {
                regulator-state-mode = <0x2>;
                regulator-state-enabled;
                regulator-state-uv = <900000>;
            };
        };
```

❖ regulator-min-microvolt 和 regulator-max-microvolt 一致时，开机初始化时会设置电压，如果不

一致则是电压可设置范围，在初始化时不设置此路电压，按照默认值。

- regulator-initial-mode ：是 DCDC 的工作模式

  REGULATOR_MODE_FAST 0x1

  REGULATOR_MODE_NORMAL 0x2

  REGULATOR_MODE_IDLE 0x4

  REGULATOR_MODE_STANDBY 0x8

- regulator-initial-state 是 suspend 模式下设置。
- regulator-state-mode：休眠下 DCDC 的工作模式。
- regulator-state-enabled：休眠下使能状态。
- regulator-state-uv ：休眠下电压。

通过 name 去设置 PMU 的电压、工作模式及使能状态。所以注册的 name 不要重名，否则会在设置的时候出错。

（2）在设置 DCDC、LDO 的工作模式时，要在定义的结构体成员中加入.valid_ops_mask 和.valid_modes_mask。（以 DCDC 为例）

regulator-valid-modes-mask = REGULATOR_MODE_FAST | REGULATOR_MODE_NORMAL;

注意：如果 regulator-always-on; 即不能使能关闭，是常开的，如果需要关闭，将此屏蔽。

（3）regulator_ops 注册，完成 PMU 驱动与 regulator 之间的连接：
以 ricoh619 为例，ricoh619-regulator.c 中：

```
static struct regulator_ops ricoh619_ops = {
      .list_voltage           = ricoh619_list_voltage,
      .set_voltage            = ricoh619_set_voltage,
      .get_voltage            = ricoh619_get_voltage,
      .set_suspend_voltage = ricoh619_set_suspend_voltage,
      .set_voltage_time_sel = ricoh619_dcdc_set_voltage_time_sel,
      .get_mode = ricoh619_dcdc_get_mode,
      .set_mode = ricoh619_dcdc_set_mode,
      .enable                 = ricoh619_reg_enable,
      .disable                = ricoh619_reg_disable,
      .set_suspend_mode = ricoh619_dcdc_set_suspend_mode,
      .set_suspend_enable             = ricoh619_reg_suspend_enable,
      .set_suspend_disable            = ricoh619_reg_suspend_disable,
      .is_enabled             = ricoh619_reg_is_enabled,
};
```

完成上述注册后就可以通过 regulator_set_voltage 设置电压，也可使用其他的 regulator 的接口了。

2.2 PMIC 驱动注册

在驱动中使用I2C通信，要在I2C设备上挂载驱动，在dts文件中注册如下：

```
&i2c0 {
	status = "okay";
	rk808: rk808@1b {
		reg = <0x1b>;
		status = "okay";
	};
	rk818: rk818@1c {
		reg = <0x1c>;
		status = "okay";
	};
		syr827: syr827@40 {
		compatible = "silergy,syr82x";
	reg = <0x40>;
		status = "okay";
		regulators {
			#address-cells = <1>;
			#size-cells = <0>;
			syr827_dc1: regulator@0 {
			reg = <0>;
			regulator-compatible = "syr82x_dcdc1";
			regulator-name = "vdd_arm";
			regulator-min-microvolt = <712500>;
			regulator-max-microvolt = <1500000>;
			regulator-always-on;
			regulator-boot-on;
			regulator-initial-mode = <0x2>;
			regulator-initial-state = <3>;
			regulator-state-mem {
				regulator-state-mode = <0x2>;
				regulator-state-disabled;
				regulator-state-uv = <900000>;
			};
		};
	  };
	};
	---    ------ {

syr828: syr828@41 {
	compatible = "silergy,syr82x";
	reg = <0x41>;
	status = "okay";
	regulators {
		#address-cells = <1>;
		#size-cells = <0>;
		syr828_dc1: regulator@0 {
		reg = <0>;
		regulator-compatible = "syr82x_dcdc1";
		regulator-name = "vdd_gpu";
		regulator-min-microvolt = <712500>;
		regulator-max-microvolt = <1500000>;
//		regulator-always-on;
		regulator-boot-on;
		regulator-initial-mode = <0x2>;
		regulator-initial-state = <3>;
		regulator-state-mem {
			regulator-state-mode = <0x2>;
			regulator-state-disabled;
			regulator-state-uv = <900000>;
		};
	};
  };
};
act8846: act8846@5a {
	reg = <0x5a>;
	status = "okay";
};

ricoh619: ricoh619@32 {
	reg = <0x32>;
	status = "okay";
};
```

2.3 PWM 驱动注册

1、驱动注册：需要 PWM 调整外挂 DCDC 的电压时，需要注册 PWM 驱动，在 RK3288-TB.DTS 文件注册如下：

```
pwm_regulator {
	compatible = "rockchip_pwm_regulator";
	pwms = <&pwm1 0 2000>;
	rockchip,pwm_id= <1>;
	rockchip,pwm_voltage_map= <925000 950000 975000 1000000 1025000 1050000 1075000 1100000 1125000 1150000 1175000 1200000 122500
	rockchip,pwm_voltage= <1000000>;
	rockchip,pwm_min_voltage= <925000>;
	rockchip,pwm_max_voltage= <1400000>;
	rockchip,pwm_suspend_voltage= <950000>;
	rockchip,pwm_coefficient= <475>;
	regulators {
		#address-cells = <1>;
		#size-cells = <0>;
		pwm_reg0: regulator@0 {
			regulator-compatible = "pwm_dcdc1";
			regulator-name= "vdd_logic";
			regulator-min-microvolt = <925000>;
			regulator-max-microvolt = <1400000>;
			regulator-always-on;
			regulator-boot-on;
		};
	};
};
```

并打开相应的PWM口：

```
&pwm1 {
	status = "okay";
};
```

2、PWM驱动介绍

对于PWM调整原理，其实就是通过占空比设置电压。

占空比计算：

在pwm_regulator_set_voltage中将设置的电压转换成占空比，计算公式：

pwm_value = (max-vol)/coefficient/10;　//占空比计算 ，max及coefficient由板级传参

占空比的设置：

在rockchip-pwm-regulator.c中pwm_set_rate（）

RATE为0时设置PWM为GPIO口输出低，控制LOGIC电压最高（1.4V）。

RATE为100时设置PWM为GPIO口输出高，控制LOGIC电压最低（0.9V）

RATE在0~100之间：

根据当前PWM的CLK计算高电平和低电平的值，然后写到PWM控制寄存器中即可。

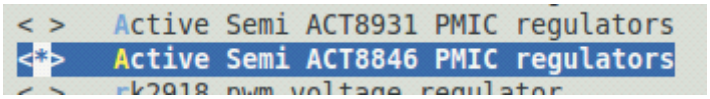
注意：

PWM控制器设置必须按照特定流程：先将PWM控制器disable并RESET，然后设置占空比，最后enable PWM控制器，而且配置过程尽量不受打扰。

3 软件配置及接口说明

3.1 配置说明

Act8846:

[*] Voltage and Current Regulator Support ---> 

由于 act8846 内部不带 RTC，所以 RTC 相关部分需要单独配置，依据项目上实际使用。

RK808:

[*] Multifunction device drivers --->

[*] Real Time Clock --->

RICOH619:

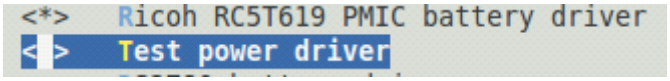
[*] Multifunction device drivers --->

[*] Voltage and Current Regulator Support --->

[*] Real Time Clock --->

Input device support --->[*] Miscellaneous devices --->

<*> Power supply class support --->



[*] Voltage and Current Regulator Support --->

分立 DCDC：

PWM 控制 DCDC

[*] Voltage and Current Regulator Support ---->

SYR82X 控制 DCDC

[*] Voltage and Current Regulator Support ---->

目前软件上已经完成了上述 PMIC 的兼容，所以可以同时打开上面所有宏，软件会根据当前硬件是哪款 PMIC 而自动识别。

3.2 修改各路 DCDC 及 LDO 的接口

1、设置初始化电压接口：

在 rk3288-tb.dts 中：

PMIC:

```
ricoh619_dcdc4_reg: regulator@0 {
                regulator-name = "vccio";
                regulator-min-microvolt = < 3300000>;
                regulator-max-microvolt = <3300000>;
};
ricoh619_ldo1_reg: regulator@5    {
                regulator-name = "ricoh619_ldo1";
                regulator-min-microvolt = <3300000>;
                regulator-max-microvolt = <3300000>;
      };
```

PWM:

```
      pwm_regulator {
            rockchip,pwm_voltage= <1000000>;
};
```

SYR82X:

```
syr827: syr827@40 {
                regulator-min-microvolt = <712500>;
                regulator-max-microvolt = <1500000>;
};
```

2、系统运行中修改电压

```
Struct regulator *dcdc;
dcdc =regulator_get(NULL, "name");
regulator_set_voltage(dcdc, min_uv, max_uv);
regulator_enable(dcdc);
regulator_put(dcdc);
```

3、设置工作模式接口：

上面所述，DCDC 有 PWM、PFM 模式，但是 PMIC 有一种模式会动态调整 PWM、PFM，这就是我们通常所说的 ATUO 模式，对于 PMIC 来说，支持 PWM、AUTO 两种模式，AUTO 模式效率高但是纹波瞬态响应会差。

现在处于系统稳定性考虑，都是设置为 PWM 模式。我们系统支持根据 ARM 频率调整模式即 ARM 运行低频率下时切换到 ATUO 模式提高效率。

初始化工作模式设置：

regulator-initial-mode = <0x2>; 详细见本文档 2.1 章

运行下切换工作模式：

```
dcdc = regulator_get(NULL, "name");
regulator_set_mode(dcdc, REGULATOR_MODE_STANDBY);
//pwm：REGULATOR_MODE_NORMAL   pfm: REGULATOR_MODE_STANDBY
```

初始化使能状态设置：

详细见本文档 2.1 章

运行后设置各路使能状态：

用户使用时可能会由于不同的电路，需要使能部分输出回路，可以使用如下接口（所有的 PMU 都支持此接口）：

```
ldo =regulator_get(NULL, "act_ldo1");
regulator_enable(ldo);   //开启 ldo1
//regulator_disable(ldo);   //关闭 ldo1
```

注意：

在 regulator 架构上要求 enable 和 disable 必须要成对出现，否则设置即会失效。详细见 core.c

3.3 PMU 休眠模式及系统待机时的电压控制方式

休眠模式：系统休眠时，电流很小，所以，有的 PMIC 需要配置为 SLEEP 模式以减低 PMIC 自身的功耗，不同的 PMIC 操作不同。

待机电压设置：待机时，arm、logic 需要低电压就可以运行，所以，待机时会更加 PMIC 的支持进行设置。

3.3.1 pmu 进入 sleep 模式

由于 act8846 没有 sleep 功能，所以在此不作介绍。
RK808 和 RICOH619 使用 PMU_SLEEP 脚切换 SLEEP 和 NORMAL 两种工作模式，在切换模式时，各路的工作电压、使能状态、工作模式可以直接切换。
目前 32 平台上发布的参考设计图，PMU_SLEEP 接到主控的 GLOBAL_PWROFF/PMUGPIO0_A0，此脚在 CPU 进去 idel 模式下可以关闭 ARM 内核时输出高电平，恢复时低电平，用此 IO 控制 PMIC 的 SLEEP 切换。

注意：
实现上述功能，必须要在 rk3288-tb.dts 文件中传参进来详细见本文档 1.2 章。
具体项目上使用哪个 io 根据硬件配置。
ACT8846 没有此功能，在次不作介绍。

3.3.2 系统待机时电压设置

关闭相关回路输出：支持 SLEEP 功能的 PMIC，一般都支持在 PMIC 进入 SLEEP 状态时，同时关闭预先定义的需要关闭的输出回路。

（2）（1）PMIC 在进入 SLEEP 模式关闭部分输出回路或者降低各路休眠电压

```
ricoh619_dcdc1_reg: regulator@0 {
        regulator-name = "vdd_logic";
        regulator-initial-state = <3>;
        regulator-state-mem {
            regulator-state-mode = <0x2>;
            regulator-state-enabled;   //sleep 模式下使能
            regulator-state-uv = <900000>;//sleep 模式下电压
        };
    };
```

（3） 分立 DCDC 使用 PWM 控制待机电压操作

对于使用 PWM 控制的 DCDC 在系统进入待机后，PWM 的 CLK 关闭，无法输出占空比调整 DCDC（LOGIC）的电压，所以在待机时我们直接将 PWM 的 IO 口拉高（DCDC 输出最低 0.9V）：

4 充电及电量计配置说明

目前三种电源方案中，只有 RICOH619 是自身带有充电和电量计功能的。

4.1 RICOH619

4.1.1 RICOH619 充电

1、充电相关原理，其原理图见下：

我们可设置 ADP 输入最大电流 Ilim，实际往电池充电最大电流 Ichg。

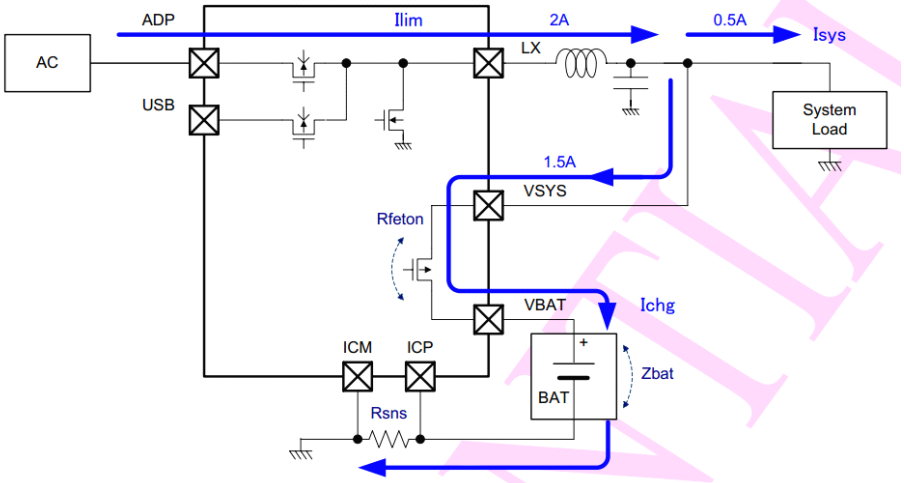
（1）限制电流的侦测方式是由 ILM， ILP 间的 Sense 电阻压降而得到。 ILM , ILP 端口需要直接采集电阻（ 20mohm) 上的压降，需要注意版图走线。

（2）ADP 进来的电流要先满足系统使用，剩余的用于充电，如果 ADP 输入的电流不满足系统使用，此时电池就要处于放电。(这就是路径管理)

（3）设置的 ADP 输入电流实际能提供系统的电流之间的关系如下，详见文档“关于 ILIM_USBILIM_ADP 的设定值”：

V_ADP * I_ADP * 效率 = VSYS * I_IN

I_IN = I_SYS + I_CHG



2、RICOH619 充电电流设置

充电支持：

方案一：单口 USB 充电

方案二：单口 ADP 充电

方案三：双口充电（ADP 和 USB）

方案四：单口 USB 充电（VBUS 直接接到 ADP 输入，较方案一充电电流会更大一些）

目前平台较多的是后面两种方案。

如果使用方案四，ricoh619-battery.c 中需要开启：

#define SUPPORT_USB_CONNECT_TO_ADP

其他三种方案直接关闭：//#define SUPPORT_USB_CONNECT_TO_ADP

（1）使用 ADP 充电时，充电电流设置在 ricoh619-battery.c 的 charger_irq_work 中，

```
if (reg_val & 0x40) { /* USE ADP */
        /* set adp limit current 2A */
ricoh619_write(info->dev->parent, REGISET1_REG, 0x13);
        /* set charge current 2A */
ricoh619_write(info->dev->parent, CHGISET_REG, 0xD3);
```

寄存器值：

ILIM_ADP[4:0]	Current Setting Values[mA]
00000 (00h)	100
00001 (01h)	200
00010 (02h)	300
00011 (03h)	400
00100 (04h)	500
00101 (05h)	600
00110 (06h)	700
00111 (07h)	800
01000 (08h)	900
01001 (09h)	1000
01010 (0Ah)	1100
01011 (0Bh)	1200
01100 (0Ch)	1300
01101 (0Dh)	1400
01110 (0Eh)	1500
01111 (0Fh)	1600
10000 (10h)	1700
10001 (11h)	1800
10010 (12h)	1900
10011 (13h)	2000
10100 (14h)	2100
10101 (15h)	2200
10110 (16h)	2300
10111 (17h)	2400
11000 (18h)	2500
11001 (19h)	2600 (*1)
11010 (1Ah)	2700 (*1)
11011 (1Bh)	2800 (*1)
11100 (1Ch)	2900 (*1)
Other Setting	3000 (*1)

ILIM_USB[4:0]	Current Setting Values [mA]
00000 (00h)	100
00001 (01h)	200
00010 (02h)	300
00011 (03h)	400
00100 (04h)	500
00101 (05h)	600
00110 (06h)	700
00111 (07h)	800
01000 (08h)	900
01001 (09h)	1000
01010 (0Ah)	1100
01011 (0Bh)	1200
01100 (0Ch)	1300
01101 (0Dh)	1400
01110 (0Eh)	1500
01111 (0Fh)	1600 (*1)
10000 (10h)	1700 (*1)
10001 (11h)	1800 (*1)
10010 (12h)	1900 (*1)
~	~
11011 (1Bh)	2800 (*1)
11100 (1Ch)	2900 (*1)
Other Setting	3000 (*1)

ICHG[4:0]	Current Setting Values [mA]
00000 (00h)	100
00001 (01h)	200
00010 (02h)	300
00011 (03h)	400
00100 (04h)	500
00101 (05h)	600
00110 (06h)	700
00111 (07h)	800
01000 (08h)	900
01001 (09h)	1000
01010 (0Ah)	1100
01011 (0Bh)	1200
01100 (0Ch)	1300
01101 (0Dh)	1400
01110 (0Eh)	1500
01111 (0Fh)	1600
10000 (10h)	1700
10001 (11h)	1800
10010 (12h)	1900 (*1)
10011 (13h)	2000 (*1)
10100 (14h)	2100 (*1)
10101 (15h)	2200 (*1)
10110 (16h)	2300 (*1)
10111 (17h)	2400 (*1)
11000 (18h)	2500 (*1)
11001 (19h)	2600 (*1)
11010 (1Ah)	2700 (*1)
11011 (1Bh)	2800 (*2)
11100 (1Ch)	2900 (*2)
Other Setting	3000 (*2)

（2）使用 USB 充电时，默认充电电流 500MA，主控会自动去识别当前是数据 USB 还是火牛充电器。然后重新设置充电电流。在 ricoh619-battery.c 中的 ricoh619_usb_charge_det 中：

```
static void ricoh619_usb_charge_det(void)
{
    struct ricoh619 *ricoh619 = g_ricoh619;
    ricoh619_set_bits(ricoh619->dev,REGISET2_REG,(1 << 7));    //set usb limit current    when SDP or other mode
    RICOH_FG_DBG("PMU:%s usb det dwc_otg_check_dpdm =%d\n", __func__,dwc_otg_check_dpdm(0));
    if(2 == dwc_otg_check_dpdm(0)){
    ricoh619_write(ricoh619->dev,REGISET2_REG,0x16);    //set usb limit current    2A
    ricoh619_write(ricoh619->dev,CHGISET_REG,0xD3);    //set charge current    2A
    }
    else {
    ricoh619_write(ricoh619->dev,REGISET2_REG,0x04);    //set usb limit current    500ma
    ricoh619_write(ricoh619->dev,CHGISET_REG,0xC4);    //set charge current    500ma
    }
    power_supply_changed(&powerac);
    power_supply_changed(&powerusb);
}
```

备注：

如果整机充电电流偏低，请核对几个问题：

1、两个 20 毫欧采样电阻附近的走线是否正确，走线差会导致电流检测不准。

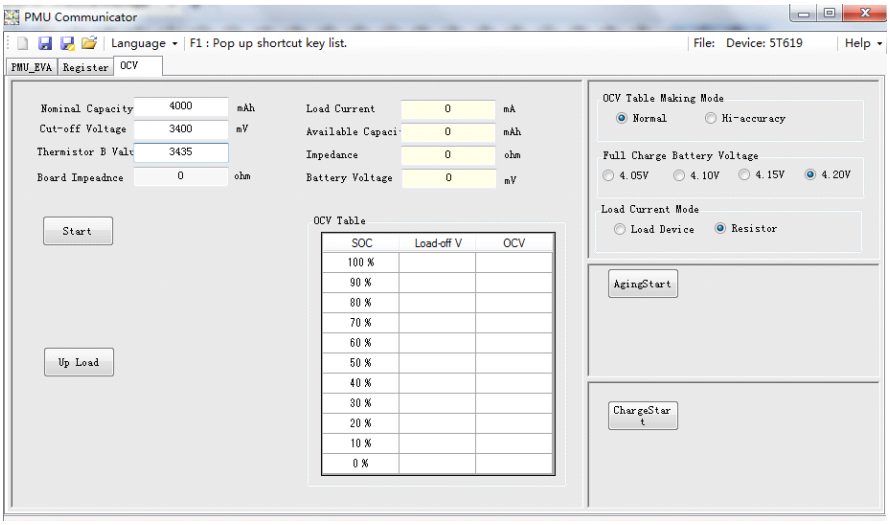
2、充电器进来的 5V 是否由于线损或者走线差，导致 PMIC 充电输入端的电压较低。

4.1.2 RICOH619 电量计

1、电池曲线测试

电池 OCV-SOC 对应曲线用于电池第一次插上时电池电量的计算，以及使用电压法计算电量时的根据。一般测试工具由原厂给出然后测出相应的曲线后替换驱动中相应的文件。

（1）测试工具如下图：



（2）测试方法：电池加上，连上 USB，接上 5V 的 ADP，设置好相应参数如上图。然后开始即可。客户只需要修改电池充满电压、截止电压、电池容量。

（3）测试后的.ocv 文件转换成.h 文件的方法详见文档“Battery_ocv_file_make_tool.xls”，生产的.h 文件替换驱动中 include/linux/power/ricoh61x_battery_init.h

2、电量计原理及参数调整

电量计有电流法和电压法两种，电流法是使用内部库仑计积分充放电电流，电压法是在负载电流较低时采样电池的 OCV 电压用于判断 SOC 值。使用两种方法结合来计算 SOC 值。

（1）电量计满充电压调整：

满充电压的设置需要修改 ricoh619,dtsi 文件中：

ricoh,ch-vfchg = <0xff>;/* VFCHG = 0 - 4 (4.05v, 4.10v, 4.15v, 4.20v, 4.35v) */

ricoh,ch-vrchg = <0xff>;/* VRCHG = 0 - 4 (3.85v, 3.90v, 3.95v, 4.00v, 4.10v) */

ricoh,ch-vbatovset = <0xff>;/* VBATOVSET = 0 or 1 (0 : 4.38v(up)/3.95v(down) 1: 4.53v(up)/4.10v(down)) */

上述值是修改满充电压、回充电压、过压，使用 0xff 就是 OTP 中默认 4.2V，4V，4.38v(up)/3.95v(down)，支持 4.2V 的电池。

例如：如果修改为 4.35V 的电池：

ricoh,ch-vfchg = <0x04>;/* VFCHG = 0 - 4 (4.05v, 4.10v, 4.15v, 4.20v, 4.35v) */

ricoh,ch-vrchg = <0x04>;/* VRCHG = 0 - 4 (3.85v, 3.90v, 3.95v, 4.00v, 4.10v) */

ricoh,ch-vbatovset = <0x01>;/* VBATOVSET = 0 or 1 (0 : 4.38v(up)/3.95v(down) 1: 4.53v(up)/4.10v(down)) */

（2）电量计截止电压调整：

如果电池曲线中设置的截至电压过高，通过软件是无法降低截至电压的。只有曲线中设置的截至

电压低，我们可以通过软件提高截至电压。所以一般建议客户电池曲线测试的截至电压尽量放低一些。

修改截至电压在 ricoh619.dtsi 中：

ricoh,fg-target-vsys = <3200>;

一般我们推荐 VSYS 设置 3.2V，因为此时 VBAT 要比 VSYS 高，OCV 值一般在 3.4-3.5V 左右。

备注：

如果拔下充电器充电图标还在，请确认充电输入端的电压，是否比 VBAT 低 0.1V 左右。如果是 VBAT 倒灌回来，导致 VADP 端始终有电压会使系统后面无法检测充电。

修改的方法：

如果是单口充电，可以将 VADP 跟 VUSB 两个充电输入端短接。

如果是双口充电，可以打开 ricoh619-battery.c 中#define RICOH619_VADP_DROP_WORK 1

5 Pmic 关机及复位

5.1 Act8846 关机及复位

Reset：

（1）、reset 后 PMIC 掉电，按开机键重新开机。由于 ACT8846无法清除 PMU 内部寄存器的，所以会出现在电压较低时按 reset 键无法重启。这种情况下使用 VSEL 脚切换，运行时 VSEL 脚拉高，使用内部寄存器 REG1设置。在按下 reset 时 GPIO 恢复默认值（拉低），可以实现切换到内部寄存器 REG0设置。REG0中在开机时写入保存电压不低于1.0V。目前已经实现此功能。

关机：

Act8846 关机函数中，由于 8846 自身 BUG，不能写 I2C 关机，我们关机采用直接拉低 Hold 脚，如果有充电器插着直接 restart 进入充电界面。

需要拦截关机命令：

```
if (pdev->pm_off && !pm_power_off) {
        pm_power_off = act8846_device_shutdown;
    }
```

8846 关机函数 act8846_device_shutdown();

```
    if (act8846->pmic_hold_gpio) {
            gpio_direction_output(act8846->pmic_hold_gpio,0);
            arm_pm_restart('h', "charge");
    }
```

5.2 RK808 关机及复位

Reset：

对于 RK808有复位按键检测功能的，复位键按下后，PMIC 可实现掉电重启。

关机：

通过 i2c 写寄存器，实现808的关机，可以恢复 PMU 内部寄存器为默认值。

需要拦截关机命令：

```
if (pdev->pm_off && !pm_power_off) {
        pm_power_off = rk808_device_shutdown;
    }
```

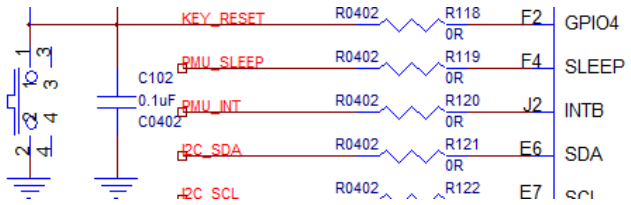
rk808_device_shutdown（）；

也可长按关机（清内部寄存器）。

5.3 RICOH619 关机及复位

Reset：

对于619复位功能，使用其 GPIO4的 N_OE 功能，按键按下后可以实现 PMIC 掉电或者是 PMIC 掉电并重启。可以在开机后通过软件设置。



软件修改方法：(board-pmu-ricoh619.c 的 pre_init 中)

```
ricoh619_set_bits(ricoh619->dev,RICOH619_PWR_REP_CNT,(1 << 0));    //set restart when power off
ricoh619_clr_bits(ricoh619->dev,RICOH619_PWR_REP_CNT,(1 << 0));    //set power off
```

关机：

通过 i2c 写寄存器，实现619的关机，可以恢复 PMU 内部寄存器为默认值。

ricoh619_power_off（）；

也可长按关机（清内部寄存器）

6 PMIC 低电检测

Act8846:

目前低电默认电压3.0v，一旦 VSYS 电压低于3.0v 产生中断。无法自动关机。低电的部分还没有实现。

SYS	0x00	[6]	nBATSTAT	R	Battery Voltage Status. Value is 1 when BATLEV interrupt is generated, value is 0 otherwise.
SYS	0x00	[5]	VBATDAT	R	Battery Voltage Monitor real time status. Value is 1 when VBAT < BATLEV, value is 0 otherwise.
SYS	0x00	[4]	-	R/W	Reserved.
SYS	0x00	[3:0]	BATLEV	R/W	Battery Voltage Detect Threshold. Defines the BATLEV voltage threshold. See the *Programmable Battary Voltage Monitor* section for more information.

RK808:

在 rk808.c 的 pre_init

```
/***************set vbat low **********/
      val = rk808_reg_read(rk808,RK808_VB_MON_REG);
          val &=(~(VBAT_LOW_VOL_MASK | VBAT_LOW_ACT_MASK));
          val |= (RK808_VBAT_LOW_3V5 | EN_VBAT_LOW_IRQ);
          ret = rk808_reg_write(rk808,RK808_VB_MON_REG,val);
             if (ret <0) {
                        printk(KERN_ERR "Unable to write RK818_VB_MON_REG reg\n");
                        return ret;
              }
/************************************/
```

目前默认电池电压低于3.4V 不能开机，开机后通过设置 vabt low 在电量低时触发中断。

RICOH619:

在 ricoh619-battery.c 中打开

#define ENABLE_LOW_BATTERY_DETECTION//低电电压设置为3.5v

目前默认 VSYS 低于3.4V 不能开机。

7 硬件测试

1、各路输出值确认

拿到机器后可以先测试下，各路的 DCDC 及 LDO 输出是否正确，是否跟软件设置相同。是否可以修改配置。

测试方法：

在 PMU 的 probe 完成后加一个 while(1);然后测试各路的电压时候跟串口打印出来的一致。也可以通过修改 set_init 中初始化的电压值然后测试实际电压是否跟设置值相同。

注意：

如果使用 PMU 给 DDR 供电，PMU 的实际输出电压 1.2V，通过外部升压电压升到 1.5V。（此用途主要是为了兼容 LPDDR2 和 LPDDR3）

2、纹波、瞬态响应测试

拿到机器后可以先测试下，各路 DCDC 的纹波，使用示波器测试峰峰值，PMU 理论偏差在±3%，如果纹波过大需要硬件查找电路是否有问题，是否走线太细等。或者换其他机器测一下对比下是否 PMU 异常。

瞬态响应主要是 DCDC 电路在负载突变时，电压会有一个瞬间的塌陷，这个塌陷会造成系统死机。使用示波器测试最小值，如果塌陷过大（一般不超过 100mv），可能需要硬件分析外部电容是否合适。

8 PMIC 调试相关问题分析

1、开机打印 PMU 注册失败，提示 i2c 通信失败，或者直接跑飞

1）测试 i2c 的 CLK 和 data 数据线是否被拉低；2）核对 i2c 有没有注册错，使用的 i2c 是否跟平台上一致；3）i2c 上是否还有其他设备；4）测试 PMIC 的各路默认的上电电压是否正确；5）测试 Power_hold 是否为高；6）测试 PMIC 默认上电完成后 reset 信号是否发送；7）PMIC 外围部件是否焊接错误，例如晶振有没有焊反，电感、电容等有没有焊错。

2、系统运行中死机

测试死机时各路的电压，是否有电压异常；2）测试 DVFS 调整是否正常；3）测试 arm、logic、DDR、VCCIO 的电压的纹波等，看下在系统异常时是否有明显的电压塌陷。

3、待机唤醒死机

1）测试 PMU_sleep 脚已经恢复成低；2）测试各路电压已经从待机电压恢复回来；3）用示波器测试唤醒时各路电压的的变化波形，有无异常；4）sleep 状态不去关闭电源。不降低电压；

4、Reset 后无法开机或者某些设备异常

如果出现此问题，请参考本文第三章 pmic 的关机及复位。

Reset 按下时，芯片复位，各个 io 口恢复成默认值，如果 PMU 没有复位功能或者没有完成硬复位，则 PMU 不会被复位，则无法完成 PMU 的断电并重新上电，这样可能会导致部分设备重启后工作不正常。（部分项目上发现有些设备在复位时必须要掉电，否则会工作异常）

5、RTC 无法正常写入

1）如果 PMU 是 act8846（不带 RTC）此时要看下 RTC 设备是否挂载正常；2）PMU 设备的 i2c 通信是否正常

6、运行时部分 LDO 没有输出

1)确认此 LDO 在其他地方没有被关闭，通过搜索此 LDO 的 name（使用此接口 regulator_disable()）；

2）测试 PMIC_SLEEP 脚是不是为高，如果为高，PMU 已经进入休眠模式，部分 LDO 会被关闭。PMIC_SLEEP 脚的配置应该是不对的，需要配置此 io 口:见本文档 1.2 章(注意：硬件上此 io 口，最好没有复用功能，而且是应该是默认内部下拉口)

7、关机后随机性的自动开机或者关机失败

关机失败：device_shutdown（）中会通过 i2c 写关机命令，i2c 通信失败，无法写关机命令。如果

遇到此问题。加一些保护锁，如果一直写失败，直接重启。

关机后自动开机：因为 RTC 闹铃具有开机的功能，在关机后如果 RTC 闹铃产生，即会触发开机条件而开机。目前已经解决过了，在关机函数中先关闭 RTC 闹铃的中断，在开机时再打开中断。

8、待机时随机性被唤醒

如果唤醒中断是 PMIC_INT,那么应该是 PMU 的 RTC 闹铃在唤醒系统。除了用户设置的闹铃外，还有一些谷歌的应用会产生 RTC 闹铃，需要移除相应的应用。查看方法如下：

请客户在命令行中敲入 dumpsys alarm 来查看当前系统的 alarm 申请情况。

android 总共有4种类型 alarm，其中只有2种 alarm 会在休眠唤醒系统，请在相关信息中查找对应的第三方软件。

这两种分别是 ELAPSED_WAKEUP 和 RTC_WAKEUP。

我们观察到，凡是带了谷歌相关的 apk 就会申请相关 alarm，如下：

RTC_WAKEUP #5: Alarm{413418b8 type 0 com.google.android.gsf}

type=0 when=+5d17h1m54s609ms repeatInterval=566316000 count=0

operation=PendingIntent{41294818: PendingIntentRecord{413ce158

com.google.android.gsf broadcastIntent}}

注意：此唤醒动作只在二级待机，不会到一级待机唤醒，点亮屏，如果点亮了屏，应该是屏那边没有进休眠。

9、RICOH619无法开机或者开机后电量显示异常

确认电池包上 TS 端是否有10K 电阻，如果没有，请确认板子上 TS 端对地有10K 电阻。如果没有10K 电阻，PMIC 默认电池是不存在的。

10、RICOH619充电异常，充电电流多小，或者充电充不满

充电电流过小：

我们上电有默认的充电电流，USB：500MA ， ADP ：1A，开机后 USB 根据枚举结果通过 I2C 重新设置充电电流，最大可设置 3A（一般推荐到 2A），ADP 电流开机后就会修改，最大 3A。

实际往电池充的电流跟设置电流会有一些不一样，有充电效率的关系。详细见下：

ILIM_USB/ILIM_ADP 的设置值 限制 USB/ADP 输入时 VSYS 系统输出的电流极限最大限流。 （ 实际限制的输出值 大约在 设置值 的 90% 左右 ）

因为 工作方式是 DCDC 的关系， 实际的输入电流会小于输出电流。

（ 功率转换： 理想情况下 输入 5V x 1A = 4V * 1.25A ） Ex. 5V = VADP, 1A = I_ADP, 4V = VSYS, ILIM_ADP = 1.25A.

另外，限制电流的侦测方式是 由 ILM， ILP 间的 Sense 电阻 压降而得到。

（ ILM , ILP 端口需要直接采集 电阻（ 20mohm ） 上的 压降，需要注意版图画法 ）

关于流入电池的电流 可以 查看一下 ICM , ICP 之间的 Sense 电阻的压降来判断。

当然，也需要小心制版时， PMU 的 ICM, ICP 是否取到的直接时 电阻两端的电压。

1. 1.5A 设置 输入限流， 实际输出限制 大约 1.3A 以上。 （ 可通过 检测 ILM / ILP 间 电阻压降 ）

2. 由于是 DCDC 工作的方式， V_USB * I_USB * 效率 = VSYS * I_SYS

IUSB = VSYS * I_SYS / (V_USB * 效率)

= 4.0 * 1.3 / (5 * 0.9 ）

= 1.1A （ 约 ）

充电充不满：

充电有充电时间的限制，一般是充电 5 小时就会关闭充电，所以存在一边使用一边充电时电池充不满，这个问题我们后期会使用软件解决。