全志科技
Allwinner Technology

Y10

System Configuration 说明书

文档履历

版本号	日期	制/修订人	制/修订记录
V1.0	2013-04-03		初始版本
V2.0	2014-05-30		update

目录

1. 系统

1.1. [product]

配置项	配置项含义
Version = "100"	配置的版本号
machine = "evb"	方案名字

配置举例：
[product]
version = "100"
machine = "evb"

1.2. [platform]

配置项	配置项含义
eraseflag=1	量产时是否擦除。0：不擦，1：擦除（仅仅对量产工具，升级工具无效）
next_work = 2	PhoenixUSBPro 量产：1-不做任何动作，2-重启，3-关机，4-量产，5-正常启动，6-量产结束进入关机关机充电

配置举例：
[platform]
eraseflag = 1
next_work =3

1.3. [target]

配置项	配置项含义
boot_clock=xx	启动频率(A7启动频率); xx 表示多少 MHZ
storage_type = -1	启动介质选择 0 ： nand, 1：card0,2：card2,-1（defualt）自动扫描启动介质:

配置举例：
[target]
boot_clock = 1008
storage_type = -1

1.4. [charging_type]

配置项	配置项含义
charging_type = 1	为 1，才能进入安卓关机充电模式；为 0 是进入 boot standby 进行充电。

注：如果想进入安卓关机充电功能，应该务必加上以上内容。

1.5. [key_detect_en]

配置项	配置项含义
keyen_flag = 1	当 keyen_flag = 1 时，支持按键检测; 当 keyen_flag = 0 时，不支持按键检测

1.6. [power_sply]

配置项	配置项含义
dcdc1_vol = 3000	dcdc1 的输出电压，mV
dcdc2_vol = 1100	dcdc2 的输出电压，mV
dcdc3_vol = 1200	dcdc3 的输出电压，mV,
dcdc4_vol = 0	dcdc4 的输出电压，mV
dcdc5_vol = 1500	dcdc5)的输出电压，mV
aldo2_vol = 2500	aldo2 的输出电压，mV
aldo3_vol = 3000	aldo3 的输出电压，mV

配置举例：

```
[power_sply]
dcdc1_vol           = 3000
dcdc2_vol           = 1100
dcdc3_vol           = 1200
dcdc4_vol           = 0
dcdc5_vol           = 1500
aldo2_vol           = 2500
aldo3_vol           = 3000
```

1.7. [card_boot]

配置项	配置项含义
logical_start = 40960	启动卡逻辑起始扇区
sprite_gpio0 =	卡量产 gpio led 灯配置
next_work = 2	卡量产完成后：1-不做任何动作，2-重启，3-关机，4-量产，5-正常启动

举例配置：

```
[card_boot]
logical_start      = 40960
sprite_gpio0       =
next_work          = 2
```

1.8. [card0_boot_para]

配置项	配置项含义
card_ctrl=0	卡量产相关的控制器选择 0
card_high_speed=xx	速度模式 0 为低速，1 为高速

card_line=4	代表 4 线卡
sdc_d1=xx	sdc 卡数据 1 线信号的 GPIO 配置
sdc_d0=xx	sdc 卡数据 0 线信号的 GPIO 配置
sdc_clk=xx	sdc 卡时钟信号的 GPIO 配置
sdc_cmd=xx	sdc 命令信号的 GPIO 配置
sdc_d3=xx	sdc 卡数据 3 线信号的 GPIO 配置
sdc_d2=xx	sdc 卡数据 2 线信号的 GPIO 配置

配置举例：

```
[card0_boot_para]
card_ctrl          = 0
card_high_speed = 1
card_line          = 4
sdc_d1              = port:PF0<2><1><2><default>
sdc_d0              = port:PF1<2><1><2><default>
sdc_clk             = port:PF2<2><1><2><default>
sdc_cmd             = port:PF3<2><1><2><default>
sdc_d3              = port:PF4<2><1><2><default>
sdc_d2              = port:PF5<2><1><2><default>
```

1.9. [card2_boot_para]

配置项	配置项含义
card_ctrl=2	卡启动控制器选择 2
card_high_speed=xx	速度模式 0 为低速，1 为高速
card_line=4	4 线卡
sdc_ cmd =xx	sdc 命令信号的 GPIO 配置
sdc_ clk =xx	sdc 卡时钟信号的 GPIO 配置
sdc_ d0 =xx	sdc 卡数据 0 线信号的 GPIO 配置
sdc_ d1 =xx	sdc 卡数据 1 线信号的 GPIO 配置
sdc_d3=xx	sdc 卡数据 3 线信号的 GPIO 配置
sdc_d2=xx	sdc 卡数据 2 线信号的 GPIO 配置

配置举例：

```
[card2_boot_para]
card_ctrl          = 2
card_high_speed = 1
card_line          = 4
sdc_clk             = port:PC05<3><1><2><default>
sdc_cmd             = port:PC06<3><1><2><default>
sdc_d0              = port:PC08<3><1><2><default>
sdc_d1              = port:PC09<3><1><2><default>
sdc_d2              = port:PC10<3><1><2><default>
sdc_d3              = port:PC11<3><1><2><default>
sdc_ex_dly_used     = 1
```

sdc_odly_400K =
sdc_odly_25M = 0
sdc_odly_50M = 3
sdc_odly_50MDDR =
sdc_odly_50MDDR_8BIT =
sdc_odly_100M =
sdc_odly_200M =
sdc_sdly_400K =
sdc_sdly_25M = 4
sdc_sdly_50M = 7
sdc_sdly_50MDDR =
sdc_sdly_50MDDR_8BIT =
sdc_sdly_100M =
sdc_sdly_200M =

1.10. [twi_para]

配置项	配置项含义
twi_port= xx	Boot 的 twi 控制器编号
twi_scl=xx	Boot 的 twi 的时钟的 GPIO 配置
twi_sda=xx	Boot 的 twi 的数据的 GPIO 配置

配置举例：
[twi_para]
twi_port = 0
twi_scl = port:PH02<2><default><default><default>
twi_sda = port:PH03<2><default><default><default>

1.11. [uart_para]

配置项	配置项含义
uart_debug_port=xx	Boot 串口控制器编号
uart_debug_tx=xx	Boot 串口发送的 GPIO 配置
uart_debug_rx=xx	Boot 串口接收的 GPIO 配置

配置举例：
[uart_para]
uart_debug_port = 0
uart_debug_tx = port:PF02<3><1><default><default>
uart_debug_rx = port:PF04<3><1><default><default>

1.12. [force_uart_para]

配置项	配置项含义
force_uart_port=xx	强制 debug 模式 Boot 串口控制器编号
force_uart_tx=xx	强制 debug 模式 Boot 串口发送的 GPIO 配置

force_uart_rx=xx	强制 debug 模式 Boot 串口接收的 GPIO 配置

配置举例：

```
force_uart_port    = 0
force_uart_tx      = port:PF02<3><1><default><default>
force_uart_rx      = port:PF04<3><1><default><default>
```

1.13. [jtag_para]

配置项	配置项含义
jtag_enable=xx	JTAG 使能
jtag_ms=xx	测试模式选择输入(TMS) 的 GPIO 配置
jtag_ck=xx	测试时钟输入(TMS) 的 GPIO 配置
jtag_do=xx	测试数据输出(TDO) 的 GPIO 配置
jtag_di=xx	测试数据输入（TDI）的 GPIO 配置

配置举例：

```
[jtag_para]
jtag_enable                = 1
jtag_ms                    = port:PF00<3><default><default><default>
jtag_ck                    = port:PF05<3><default><default><default>
jtag_do                    = port:PF03<3><default><default><default>
jtag_di                    = port:PF01<3><default><default><default>
```

1.14. [clock]

配置项	配置项含义
Pll3 =297	Video0 时钟频率
Pll4 =300	Ve 时钟频率
Pll6 =600	Peripherals 时钟频率
Pll8 =408	GPU（通信）时钟频率
Pll9 =480	GPU（运算）时钟频率
Pll10=468	De 时钟频率
pll_cpupat = 0	PLL_CPU pattern 参数，请勿随意修改
pll_gpupat = 0xc440e666	PLL_GPU pattern 参数，请勿随意修改
pll_videopat = 0	PLL_VIDEO pattern 参数，请勿随意修改
pll_vepat = 0	PLL_VE pattern 参数，请勿随意修改
pll_hsicpat = 0	PLL_HSI pattern 参数，请勿随意修改
pll_depat = 0	PLL_DE pattern 参数，请勿随意修改
pll_mipipat = 0	PLL_MIPI pattern 参数，请勿随意修改
pll_mipitun = 0x8a002008	PLL_MIPI pattern 参数，请勿随意修改
pll_mipibias= 0xf8100400	PLL_MIPI bias 参数，请勿随意修改

配置举例：

[clock]
pll3 = 297
pll4 = 300
pll6 = 600
pll8 = 408
pll9 = 480
pll10 = 297
pll_cpupat = 0
pll_gpupat = 0xc440e666
pll_videopat = 0
pll_vepat = 0
pll_hsicpat = 0
pll_depat = 0
pll_mipipat = 0
pll_mipitun = 0x8a002008
pll_mipibias= 0xf8100400

1.15. [pm_para]

配置项	配置项含义
standby_mode = x	if 1 == standby_mode, then support super standby; else, support normal standby.

配置举例：
[pm_para]
standby_mode = 1

2. SDRAM

2.1. [dram_para]

配置项	配置项含义
dram_clk =xx	DRAM 的时钟频率，单位为 MHz;它为 24 的整数倍，最低不得低于 120，
dram_type =xx	DRAM 类型： 2 为 DDR2 3 为 DDR3
dram_zq=xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_odt_en=xx	ODT 是否需要使能 0： 不使能 1： 使能

	一般情况下，为了省电，此项为 0
dram_para1=xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_para2 =xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_mr0 =xx	DRAM CAS 值，可为 6,7,8,9；具体需根据 DRAM 的规格书和速度来确定
dram_mr1 =xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_mr2 =xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_mr3 =xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr0=xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr1=xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr2=xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr3=xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr4=xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr5=xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr6=xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr7=xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr8=xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr9=xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr10=xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr11=xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr12=xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr13=xx	DRAM 控制器内部参数，由原厂来进行调节，请勿修改

配置举例：

[dram_para]

dram_clk = 552
dram_type = 3
dram_zq = 0x3bbb
dram_odt_en = 1
dram_para1 = 0x10F20200
dram_para2 = 0x00
dram_mr0 = 0x1840
dram_mr1 = 0x40
dram_mr2 = 0x8
dram_mr3 = 0
dram_tpr0 = 0x0048A192
dram_tpr1 = 0x01B1B18d
dram_tpr2 = 0x00076052
dram_tpr3 = 0x0
dram_tpr4 = 0x0
dram_tpr5 = 0x0
dram_tpr6 = 0x0
dram_tpr7 = 0x0
dram_tpr8 = 0x0
dram_tpr9 = 0x0
dram_tpr10 = 0x0
dram_tpr11 = 0x0
dram_tpr12 = 168
dram_tpr13 = 0x900

3. [wakeup_src_para]

配置项	配置项含义
cpu_en	power on or off. ; 1: mean power on ; 0: mean power off
cpu_freq	indicating lowest freq. unit is Mhz;
pll_ratio	Indicating cpu:apb:ahb frequency ratio.
dram selfresh_en	selfresh or not. ; 1: enable enter selfresh ; 0: disable enter selfresh
dram_pll	if not enter selfresh, indicating lowest freq. unit is Mhz;
wakeup_src	1. to make the scenario work, the wakeup src is needed. 2. The name is insensitive; 3. The module which need the wakeup src need to config the wakeup_src.

配置举例：

[wakeup_src_para]
cpu_en = 0
cpu_freq = 48
; (cpu:apb:ahb)
pll_ratio= 0x111
dram_selfresh_en= 1
dram_freq = 36
wakeup_src_wl = port:PL07<4><default><default><0>
wakeup_src_bt = port:PL09<4><default><default><0>
;bb_wake_ap = port:PL02<4><default><default><0>

4. I2C 总线

主控有 4 个 I2C（twi）控制器

4.1. [twi0_para]

配置项	配置项含义
twi0_used =xx	TWI 使用控制：1 使用，0 不用
twi0_scl =xx	TWI SCK 的 GPIO 配置
twi0_sda=xx	TWI SDA 的 GPIO 配置

配置举例：
[twi0]
twi_used = 1
twi_scl = port:PH02<2><default><default><default>
twi_sda = port:PH03<2><default><default><default>

4.2. [twi1_para]

配置项	配置项含义
twi1_used =xx	TWI 使用控制：1 使用，0 不用
twi1_scl =xx	TWI SCK 的 GPIO 配置
twi1_sda=xx	TWI SDA 的 GPIO 配置

配置举例：
[tw1_para]
twi_used = 1
twi_scl = port:PH04<2><default><default><default>
twi_sda = port:PH05<2><default><default><default>

4.3. [twi2_para]

配置项	配置项含义
twi2_used =xx	TWI 使用控制：1 使用，0 不用

twi2_scl =xx	TWI SCK 的 GPIO 配置
twi2_sda=xx	TWI SDA 的 GPIO 配置

配置举例：

```
[twi2_para]
twi_used            = 1
twi_scl             = port:PE12<3><default><default><default>
twi_sda             = port:PE13<3><default><default><default>
```

5. 串口(UART)

主控有 5 路 uart 接口，5 路支持 4 线或者 2 线通讯（但十分不建议用 uart0 作为控制台以外的用途），实例中，有些路仅仅写出 2 路的配置形式，但实际使用时只要将其按照 4 路的格式补全，也能支持 4 线通讯

5.1. [uart0]

配置项	配置项含义
uart_used =xx	UART 使用控制：1 使用，0 不用
uart_port =xx	UART 端口号
uart_type = xx	UART 类型
uart_tx =xx	UART TX 的 GPIO 配置
uart_rx=xx	UART RX 的 GPIO 配置

配置举例：

```
[uart0]
uart_used           = 1
uart_port           = 0
uart_type           = 2
uart_tx             = port:PF02<3><1><default><default>
uart_rx             = port:PF04<3><1><default><default>
```

5.2. [uart1]

配置项	配置项含义
uart_used	UART 使用控制：1 使用，0 不用
uart_port	UART 端口号
uart_type	UART 类型
uart_tx	UART TX 的 GPIO 配置
uart_rx	UART RX 的 GPIO 配置
uart_rts	UART RTS 的 GPIO 配置
uart_cts	UART CTS 的 GPIO 配置

配置举例：

[uart1]
uart_used = 1
uart_type = 4
uart_tx = port:PG06<2><1><default><default>
uart_rx = port:PG07<2><1><default><default>
uart_rts = port:PG08<2><1><default><default>
uart_cts = port:PG09<2><1><default><default>

5.3. [uart2]

配置项	配置项含义
uart_used =xx	UART 使用控制：1 使用，0 不用
uart_port =xx	UART 端口号
uart_type =xx	UART 类型
uart_tx =xx	UART TX 的 GPIO 配置
uart_rx＝xx	UART RX 的 GPIO 配置
uart_rts=xx	UART RTS 的 GPIO 配置
uart_cts=xx	UART CTS 的 GPIO 配置

配置举例：
[uart_para2]
uart_used = 0
uart_type = 4
uart_tx = port:PB00<2><1><default><default>
uart_rx = port:PB01<2><1><default><default>
uart_rts = port:PB02<2><1><default><default>
uart_cts = port:PB03<2><1><default><default>

5.4. [uart3]

配置项	配置项含义
uart_used =xx	UART 使用控制：1 使用，0 不用
uart_port =xx	UART 端口号
uart_type =xx	UART 类型
uart_tx =xx	UART TX 的 GPIO 配置
uart_rx＝xx	UART RX 的 GPIO 配置
uart_rts=xx	UART RTS 的 GPIO 配置
uart_cts=xx	UART CTS 的 GPIO 配置

配置举例：
uart_used = 0
uart_type = 4
uart_tx = port:PH06<3><1><default><default>
uart_rx = port:PH07<3><1><default><default>
uart_rts = port:PH08<3><1><default><default>
uart_cts = port:PH09<3><1><default><default>

5.5. [uart4]

配置项	配置项含义
uart_used =xx	UART 使用控制：1 使用，0 不用
uart_port =xx	UART 端口号
uart_type =xx	UART 类型
uart4_tx =xx	UART TX 的 GPIO 配置
uart4_rx＝xx	UART RX 的 GPIO 配置

配置举例：

```
[uart4]
uart_used                = 0
uart_port                = 4
uart_type                = 4
uart_tx                  = port:PG12<2><1><default><default>
uart_rx                  = port:PG13<2><1><default><default>
uart_rts                 = port:PG14<2><1><default><default>
uart_cts                 = port:PG15<2><1><default><default>
```

6. SPI 总线

6.1. [spi0]

配置项	配置项含义
spi_used =xx	SPI 使用控制：1 使用，0 不用
spi_cs_bitmap =xx	由于 SPI 控制器支持多个 CS，这一个参数表示 CS 的掩码；
spi_cs0 =xx	SPI CS0 的 GPIO 配置
spi_sclk =xx	SPI CLK 的 GPIO 配置
spi_mosi＝xx	SPI MOSI 的 GPIO 配置
spi_miso=xx	SPI MISO 的 GPIO 配置

配置举例：

```
[spi0]
spi_used            = 0
spi_cs_bitmap       = 1
spi_mosi            = port:PC00<3><default><default><default>
spi_miso            = port:PC01<3><default><default><default>
spi_sclk            = port:PC02<3><default><default><default>
spi_cs0             = port:PC03<3><1><default><default>
```

6.2. [spi1]

配置项	配置项含义
spi_used =xx	SPI 使用控制：1 使用，0 不用
spi_cs_bitmap =xx	由于 SPI 控制器支持多个 CS，这一个参数表示 CS 的掩码；
spi_cs0 =xx	SPI CS0 的 GPIO 配置
spi_sclk =xx	SPI CLK 的 GPIO 配置
spi_mosi＝xx	SPI MOSI 的 GPIO 配置
spi_miso=xx	SPI MISO 的 GPIO 配置

配置举例：

```
[spi1]
spi_used            = 0
spi_cs_bitmap    = 1
spi_cs0             = port:PA00<2><1><default><default>
spi_sclk            = port:PA01<2><default><default><default>
spi_mosi            = port:PA02<2><default><default><default>
spi_miso            = port:PA03<2><default><default><default>
```

6.3. [spi_devices]

配置项	配置项含义
spi_dev_num=xx	该项目直接和下面的[spi_board0]相关，它指定主板连接 spi 设备的数目，假如有 N 个 SPI 设备那么[spi_devices]中就要有 N 个（[spi_board0]到[spi_board（N-1）]）配置

配置举例：

```
[spi_devices]
spi_dev_num = 1
```

6.4. [spi_board0]

配置项	配置项含义
modalias=xx	Spi 设备名字，
max_speed_hz =xx	最大传输速度（HZ）
bus_num =xx	Spi 设备控制器序号
chip_select=xx	理论上可以选 0，1，2，3，目前只支持 1，2（芯片没引出接口）
mode＝xx	SPI MOSI 的 GPIO 配置可选值 0-3

配置举例：

```
[spi_board0]
```

```
modalias          = "at25df641"
max_speed_hz    = 50000000
bus_num           = 0
chip_select     = 0
mode                = 0
```

7. 电容屏

7.1. [ctp_para]

配置项	配置项含义
ctp_used	该选项为是否开启电容触摸，支持的话置 1，反之置 0
ctp_name	tp 的 name，必须配，与驱动保持一致
ctp_twi_id	用于选择 i2c adapter, 可选 1，2
ctp_twi_addr	指明 i2c 设备地址，与具体硬件相关
ctp_screen_max_x	触摸板的 x 轴最大坐标
ctp_screen_max_y	触摸板的 y 轴最大坐标
ctp_revert_x_flag	是否需要翻转 x 坐标，需要则置 1，反之置 0
ctp_revert_y_flag	是否需要翻转 y 坐标，需要则置 1，反之置 0
ctp_exchange_x_y_flag	是否需要 x 轴 y 轴坐标对换
ctp_int_port	电容屏中断信号的 GPIO 配置
ctp_wakeup	电容屏唤醒信号的 GPIO 配置
ctp_power_ldo	电容屏供电 ldo
ctp_power_ldo_vol	电容屏供电 ldo 电压
ctp_power_io	电容屏供电 gpio

配置举例：

```
[ctp_para]
ctp_used                 = 1
ctp_name                 = "ft5x_ts"
ctp_twi_id               = 0
ctp_twi_addr             = 0x38
ctp_screen_max_x       = 768
ctp_screen_max_y       = 1024
ctp_revert_x_flag     = 0
ctp_revert_y_flag     = 0
ctp_exchange_x_y_flag = 0

ctp_int_port             = port:PB05<4><default><default><default>
```

ctp_wakeup = port:PH01<1><default><default><1>
ctp_power_ldo = "axp22_eldo1"
ctp_power_ldo_vol = 3000
ctp_power_io =

8. 触摸屏自动扫描配置

8.1. [ctp_list_para]

配置项	配置项含义
ctp_det_used	0 or 1，是否使用自动检测功能
ft5x_ts	0 or 1，检测时是否扫描此类触屏
gt82x	0 or 1，检测时是否扫描此类触屏
gslX680	0 or 1，检测时是否扫描此类触屏
gt9xx_ts	0 or 1，检测时是否扫描此类触屏
gt811	0 or 1，检测时是否扫描此类触屏
zet622x	0 or 1，检测时是否扫描此类触屏
aw5306_ts	0 or 1，检测时是否扫描此类触屏

配置举例：
[ctp_list_para]
ctp_det_used = 1
ft5x_ts = 1
gt82x = 1
gslX680 = 1
gslX680new = 0
gt9xx_ts = 1
gt9xxf_ts = 0
tu_ts = 0
gt818_ts = 1
zet622x = 1
aw5306_ts = 1
icn83xx_ts = 0

9. 马达

9.1. [motor_para]

配置项	配置项含义
motor_used =xx	是否启用马达，启用置 1，反之置 0
motor_shake=xx	马达使用的 GPIO 配置

motor_ldo = xx	指明马达由 axp 哪一路电压供电
motor_ldo_voltage = xx	Axp 供电的该路电压为 xx mv

配置举例：

```
[motor_para]
motor_used              = 0
motor_shake             = port:power3<1><default><default><1>
motor_ldo               = ""
motor_ldo_voltage       = 3300
```

注意事项：

motor_shake = port:power3<1><default><default><1>

默认 io 口的输出应该为 1，这样就不会初始化之后就开始震动了。

假设 motor_shake = 0 ，说明没有指定 gpio 引脚，那么就会设置 axp 的引脚为马达供电，优先考虑 gpio 配置。

10. thermal configuration

配置项	配置项含义
ths_used = 1	总开关，为 0 thermal 功能关闭
ths_trend = 0	Thermal 趋势判断功能，目前关闭
ths_trip_count = 3	Trip_point 数目（注意必须和下面的 ths_trip1_*一致）
ths_trip1_0 = 75 ths_trip1_1 = 90 ths_trip1_2 = 110	对应的 trip_point 温度 **注意，最后一个强制作为关机使用**
ths_trip1_0_min = 0 ths_trip1_0_max = 1 ths_trip1_1_min = 1 ths_trip1_1_max = 3 ths_trip1_2_min = 0 ths_trip1_2_max = 0	和上面的表对应，注意 1）0 和 3 对应的 cooler_count=4 (下节描述)的对应关系 2) ths_trip1_2 是强制作为关机用的，min & max =0 3) 注意必须 trip1_0_max == trip1_1_min， 后面以此类推

配置举例：

```
[ths_para]
ths_used                = 1
ths_trip1_count         = 3
ths_trip1_0             = 75
ths_trip1_1             = 90
```

ths_trip1_2 = 110
ths_trip1_0_min = 0
ths_trip1_0_max = 1
ths_trip1_1_min = 1
ths_trip1_1_max = 3
ths_trip1_2_min = 0
ths_trip1_2_max = 0

11. cooler_table

配置项	配置项含义
cooler_count = 4	Cooler 支持的 state 数目，每个代表一档
cooler0 = "1344000 4 4294967295 0" cooler1 = "1200000 4 4294967295 0" cooler2 = "1008000 4 4294967295 0" cooler3 = "648000 4 4294967295 0"	每个 cooler 代表一档，分别表示 允许 cpu 允许的最大频率，允许 cpu 的最大数目，保留做其它用途，保留做其它用途 4294967295 用来表示 0xffffffff，用来区分正常频率

配置举例：
[cooler_table]
cooler_count = 4
cooler0 = "1344000 4 4294967295 0"
cooler1 = "1200000 4 4294967295 0"
cooler2 = "1008000 4 4294967295 0"
cooler3 = "648000 4 4294967295 0"

12. 闪存

12.1. [nand0_para]

配置项	配置项含义
nand_support_2ch	nand0 是否使能双通道
nand0_used =xx	nand0 模块使能标志
nand0_we =xx	nand0 写时钟信号的 GPIO 配置
nand0_ale =xx	nand0 地址使能信号的 GPIO 配置
nand0_cle =xx	nand0 命令使能信号的 GPIO 配置
nand0_ce1 =xx	nand0 片选 1 信号的 GPIO 配置
nand0_ce0 =xx	nand0 片选 0 信号的 GPIO 配置
nand0_nre =xx	nand0 读时钟信号的 GPIO 配置
nand0_rb0=xx	nand0 Read/Busy 1 信号的 GPIO 配置

nand0_rb1 =xx	nand0 Read/Busy 0 信号的 GPIO 配置
nand0_d0=xx	nand0 数据总线信号的 GPIO 配置
nand0_d1=xx	/
nand0_d2=xx	/
nand0_d3=xx	/
nand0_d4=xx	/
nand0_d5=xx	/
nand0_d6=xx	/
nand0_d7=xx	/
nand0_ce2=xx	nand0 片选 2 信号的 GPIO 配置
nand0_ce3=xx	nand0 片选 3 信号的 GPIO 配置
nand0_ndqs=xx	nand0 ddr 时钟信号的 GPIO 配置

配置举例：

```
[nand0_para]
nand_support_2ch       = 0

nand0_used              = 1
nand0_we                = port:PC00<2><default><default><default>
nand0_ale               = port:PC01<2><default><default><default>
nand0_cle               = port:PC02<2><default><default><default>
nand0_ce1               = port:PC03<2><default><default><default>
nand0_ce0               = port:PC04<2><default><default><default>
nand0_nre               = port:PC05<2><default><default><default>
nand0_rb0               = port:PC06<2><default><default><default>
nand0_rb1               = port:PC07<2><default><default><default>
nand0_d0                = port:PC08<2><default><default><default>
nand0_d1                = port:PC09<2><default><default><default>
nand0_d2                = port:PC10<2><default><default><default>
nand0_d3                = port:PC11<2><default><default><default>
nand0_d4                = port:PC12<2><default><default><default>
nand0_d5                = port:PC13<2><default><default><default>
nand0_d6                = port:PC14<2><default><default><default>
nand0_d7                = port:PC15<2><default><default><default>
nand0_ndqs              = port:PC16<2><default><default><default>
nand0_ce2               = port:PC17<2><default><default><default>
nand0_ce3               = port:PC18<2><default><default><default>
```

13. 显示

13.1. [disp_init]

配置项	配置项含义
disp_init_enable=xx	是否进行显示的初始化设置
disp_mode =xx	显示模式： 0:screen0<screen0,fb0>
screen0_output_type=xx	屏 0 输出类型(0:none; 1:lcd; 2:tv; 3:hdmi; 4:vga)
screen0_output_mode =xx	屏 0 输出模式(used for tv/hdmi output, 0:480i 1:576i 2:480p 3:576p 4:720p50 5:720p60 6:1080i50 7:1080i60 8:1080p24 9:1080p50 10:1080p60 11:pal 14:ntsc)
screen1_output_type＝xx	屏 1 输出类型(0:none; 1:lcd; 2:tv; 3:hdmi; 4:vga)
screen1_output_mode=xx	屏 1 输出模式(used for tv/hdmi output, 0:480i 1:576i 2:480p 3:576p 4:720p50 5:720p60 6:1080i50 7:1080i60 8:1080p24 9:1080p50 10:1080p60 11:pal 14:ntsc)
fb0_format=xx	fb0 的格式(4:RGB655 5:RGB565 6:RGB556 7:ARGB1555 8:RGBA5551 9:RGB888 10:ARGB8888 12:ARGB4444)
fb0_pixel_sequence=xx	fb0 的 pixel sequence(0:ARGB 1:BGRA 2:ABGR 3:RGBA)
fb0_scaler_mode_enable=xx	fb0 是否使用 scaler mode，即使用 FE
fb0_width=xx	fb0 的宽度,为 0 时将按照输出设备的分辨率
fb0_height=xx	fb0 的高度，为 0 时将按照输出设备的分辨率
fb1_format=xx	fb1 的格式(4:RGB655 5:RGB565 6:RGB556 7:ARGB1555 8:RGBA5551 9:RGB888 10:ARGB8888 12:ARGB4444)
fb1_pixel_sequence=xx	fb1 的 pixel sequence(0:ARGB 1:BGRA 2:ABGR 3:RGBA)
fb1_scaler_mode_enable=xx	fb1 是否使用 scaler mode，即使用 FE
fb1_width=xx	Fb1 的宽度,为 0 时将按照输出设备的分辨率

fb1_height=xx	Fb1 的高度，为 0 时将按照输出设备的分辨率
lcd0_backlight	Lcd0 的背光初始值，0~255
lcd1_backlight	Lcd1 的背光初始值，0~255
lcd0_bright	Lcd0 的亮度值，0~100
lcd0_contrast	Lcd0 的对比度，0~100
lcd0_saturation	Lcd0 的饱和度，0~100
lcd0_hue	Lcd0 的色度，0~100
lcd1_bright	Lcd1 的亮度值，0~100
lcd1_contrast	Lcd1 的对比度，0~100
lcd1_saturation	Lcd1 的饱和度，0~100
lcd1_hue	Lcd1 的色度，0~100

配置举例：

```
[disp_init]
disp_init_enable            = 1
disp_mode                   = 0

screen0_output_type         = 1
screen0_output_mode         = 4

screen1_output_type         = 1
screen1_output_mode         = 4

fb0_format                  = 10
fb0_pixel_sequence          = 0
fb0_scaler_mode_enable      = 0
fb0_width                   = 0
fb0_height                  = 0

fb1_format                  = 10
fb1_pixel_sequence          = 0
fb1_scaler_mode_enable      = 0
fb1_width                   = 0
fb1_height                  = 0

lcd0_backlight              = 50
lcd1_backlight              = 50

lcd0_bright                 = 50
lcd0_contrast               = 50
lcd0_saturation             = 57
lcd0_hue                    = 50
```

lcd1_bright = 50
lcd1_contrast = 50
lcd1_saturation = 57
lcd1_hue = 50

14. LCD 屏 0

14.1. [lcd0_para]

配置项	配置项含义
lcd_used=xx	是否使用 lcd0
lcd_driver_name = "xx"	定义驱动名称
lcd_if =xx	lcd 接口 (0:hv(sync+de); 1:8080; 2:ttl; 3:lvds，4:dsi; 5:edp)
lcd_x=xx	Lcd 分辨率 x
lcd_y =xx	Lcd 分辨率 y
lcd_width = xx	Lcd 屏宽度
lcd_height = xx	Lcd 屏高度
lcd_dclk_freq = xx	Lcd 频率
lcd_pwm_used =	Pwm 是否使用
lcd_pwm_ch =	Pwm 通道
lcd_pwm_freq＝xx	Pwm 频率
lcd_pwm_pol =xx	pwm 属性, 0:positive; 1:negative
lcd_hbp=xx	Lcd 行后沿时间
lcd_ht=xx	Lcd 行时间
lcd_hspw = xx	Lcd 行同步脉宽
lcd_vbp=xx	Lcd 场后沿时间
lcd_vt=xx	Lcd 场时间
lcd_vspw=xx	Lcd 场同步脉宽
lcd_lvds_if=xx	Lcd lvds 接口，0:single link; 1:dual link
lcd_lvds_colordepth =xx	Lcd lvds 颜色深度 0:8bit; 1:6bit
lcd_lvds_mode=xx	Lcd lvds 模式， 0:NS mode; 1:JEIDA mode
lcd_frm=xx	Lcd 格式， 0:disable; 1:enable rgb666 dither; 2:enable rgb656 dither
lcd_hv_clk_phase	Lcd hv 时钟 相位 0:noraml; 1:intert phase(0~3bit: vsync phase;4~7bit:hsync phase; 8~11bit:dclk phase; 12~15bit:de phase)
lcd_hv_sync_polarity	Lcd io 属性，0:not invert; 1:invert
lcd_gamma_en=xx	Lcdgamma 校正使能
lcd_bright_curve_en=xx	Lcd 亮度曲线校正使能

lcd_cmap_en=xx	Lcd 调色板函数使能
deu_mode=xx	Deu 模式 0:smoll lcd screen; 1:large lcd screen(larger than 10inch)
lcdgamma4iep=xx	只能背光参数，lcd gamma vale*10;decrease it while lcd is not bright enough; increase while lcd is too bright
smart_color=xx	丽色系统，90:normal lcd screen 65:retina lcd screen(9.7inch)
lcd_bl_en=xx	背光使能的 GPIO 配置
lcd_power=xx	Lcd 电源
lcdd0=xx	Lcd 数据 0 线信号的 GPIO 配置
Lcdd1=xx	Lcd 数据 1 线信号的 GPIO 配置
Lcdd2=xx	Lcd 数据 2 线信号的 GPIO 配置
Lcdd3=xx	Lcd 数据 3 线信号的 GPIO 配置
Lcdd4=xx	Lcd 数据 4 线信号的 GPIO 配置
Lcdd5=xx	Lcd 数据 5 线信号的 GPIO 配置
Lcdd6=xx	Lcd 数据 6 线信号的 GPIO 配置
Lcdd7 = xx	Lcd 数据 7 线信号的 GPIO 配置

配置举例：

```
[lcd0_para]
lcd_used                = 1

lcd_driver_name         = "starry768x1024"
lcd_if                  = 3
lcd_x                   = 768
lcd_y                   = 1024
lcd_width               = 120
lcd_height              = 160
lcd_dclk_freq           = 60
lcd_pwm_used            = 1
lcd_pwm_ch              = 0
lcd_pwm_freq            = 50000
lcd_pwm_pol             = 1
lcd_hbp                 = 80
lcd_ht                  = 928
lcd_hspw                = 10
lcd_vbp                 = 23
lcd_vt                  = 1065
lcd_vspw                = 3
lcd_lvds_if             = 0
lcd_lvds_colordepth = 0
lcd_lvds_mode           = 0
lcd_frm                 = 1
```

```
lcd_gamma_en            = 0
lcd_bright_curve_en = 0
lcd_cmap_en             = 0

deu_mode                = 0
lcdgamma4iep            = 22
smart_color             = 90

lcd_bl_en               = port:PH06<1><0><default><1>
lcd_power               = port:power2<1><0><default><1>
lcd_gpio_0              = port:PH07<1><0><default><1>
lcd_gpio_1                      = port:PL04<1><0><default><0>
lcd_gpio_2                      = port:PL11<1><0><default><1>

lcdd0                   = port:PD18<3><0><default><default>
lcdd1                   = port:PD19<3><0><default><default>
lcdd2                   = port:PD20<3><0><default><default>
lcdd3                   = port:PD21<3><0><default><default>
lcdd4                   = port:PD22<3><0><default><default>
lcdd5                   = port:PD23<3><0><default><default>
lcdd6                   = port:PD24<3><0><default><default>
lcdd7                   = port:PD25<3><0><default><default>
lcdd8                   = port:PD26<3><0><default><default>
lcdd9                   = port:PD27<3><0><default><default>
```

15. 摄像头(CSI)

Y10 CSI 的 sysconfig 部分对应的字段为：[csi0]。

下面举例说明在实际使用中应该如何配置：需要配置[csi0]的公用部分和[csi0]的 vip_dev(x)部分。

15.1. [csi0_para]

配置项	配置项含义
vip_used	使用 mipi 接口的 sensor 请填 1。
vip_mode	一般填 0。
vip_dev_qty	mipi 接口暂时不支持复用，填 1。
vip_define_sensor_list	如果配置了 system/etc/hawkview/sensor_list_ cfg.ini 文件，填 1，默认填 0。
vip_csi_mck	Mipi mclk 信号的 GPIO 配置。
vip_csi_sck	CSI0 CCI 时钟信号的 GPIO 配置。

	如果使用 CSI0 内部 CCI 需要配置该项
vip_csi_sda	CSI0 CCI 数据信号的 GPIO 配置。 如果使用 CSI0 内部 CCI 需要配置该项
vip_dev0_mname	设置 sensor 0 名称，如“ov5648“。
vip_dev0_pos	摄像头位置前置填"front"，后置填"rear"。
vip_dev0_lane	请参考实际模组的 lane 数目填写。
vip_dev0_twi_id	CSI 使用的 IIC 通道序号，查看具体方案原理图，使用 twi0 填 0，如果使用 CSI 内部 CCI 接口则可以不填
vip_dev0_twi_addr	请参考实际模组的 8bit ID 填写，如 0x6c。
vip_dev0_isp_used	如果是 RAW sensor 必须填 1，YUV 填 0。
vip_dev0_fmt	如果是 RAW 格式填 1，YUV 填 0。
vip_dev0_stby_mode	填 0。
vip_dev0_vflip	Sensor 图像垂直翻转。
vip_dev0_hflip	Sensor 图像水平翻转。
vip_dev0_iovdd	IOVDD 配置，请参考实际原理图填写。
vip_dev0_iovdd_vol	IOVDD 电压值一般为 2.8V(2800000)。
vip_dev0_avdd	AVDD 配置，如”axp15_aldo2”。
vip_dev0_avdd_vol	AVDD 电压值，一般为 2.8V(2800000)。
vip_dev0_dvdd	DVDD 配置，如"axp22_eldo1"。
vip_dev0_dvdd_vol	DVDD 电压值参考 datasheet，1.2/1.5/1.8V。
vip_dev0_afvdd	VCM 电源配置一般不用配置。
vip_dev0_afvdd_vol	VCM 电压值为 2.8V。
vip_dev0_power_en	Sensor power enable 引脚 GPIO 配置。
vip_dev0_reset	Sensor reset 引脚 GPIO 配置。
vip_dev0_pwdn	Sensor power down 引脚 GPIO 配置。
vip_dev0_flash_en	闪光灯 enable 引脚 GPIO 配置。
vip_dev0_flash_mode	闪光灯 flash mode 引脚 GPIO 配置。
vip_dev0_af_pwdn	VCM driver power down 引脚 GPIO 配置。
vip_dev0_act_used	模组包含 VCM driver 时候填 1。
vip_dev0_act_name	VCM driver 名字，如“ad5820_act“。
vip_dev0_act_slave	VCM driver slave 地址。
vip_dev1_mname	设置 sensor 1 名称，如“ov5648“。
vip_dev1_pos	摄像头位置前置填"front"，后置填"rear"。
vip_dev1_lane	请参考实际模组的 lane 数目填写。
vip_dev1_twi_id	CSI 使用的 IIC 通道序号，查看具体方案原理图，使用 twi0 填 0，如果使用

	CSI 内部 CCI 接口则可以不填。
vip_dev1_twi_addr	请参考实际模组的 8bit ID 填写，如 0x6c。
vip_dev1_isp_used	如果是 RAW sensor 必须填 1，YUV 填 0。
vip_dev1_fmt	如果是 RAW 格式填 1，YUV 填 0。
vip_dev1_stby_mode	填 0。
vip_dev1_vflip	Sensor 图像垂直翻转。
vip_dev1_hflip	Sensor 图像水平翻转。
vip_dev1_iovdd	IOVDD 配置，请参考实际原理图填写。
vip_dev1_iovdd_vol	IOVDD 电压值一般为 2.8V(2800000)。
vip_dev1_avdd	AVDD 配置，如”axp15_aldo2”。
vip_dev1_avdd_vol	AVDD 电压值，一般为 2.8V(2800000)。
vip_dev1_dvdd	DVDD 配置，如"axp22_eldo1"。
vip_dev1_dvdd_vol	DVDD 电压值参考 datasheet，1.2/1.5/1.8V。
vip_dev1_afvdd	VCM 电源配置一般不用配置。
vip_dev1_afvdd_vol	VCM 电压值为 2.8V。
vip_dev1_power_en	Sensor power enable 引脚 GPIO 配置。
vip_dev1_reset	Sensor reset 引脚 GPIO 配置。
vip_dev1_pwdn	Sensor power down 引脚 GPIO 配置。
vip_dev1_flash_en	闪光灯 enable 引脚 GPIO 配置。
vip_dev1_flash_mode	闪光灯 flash mode 引脚 GPIO 配置。
vip_dev1_af_pwdn	VCM driver power down 引脚 GPIO 配置。
vip_dev1_act_used	模组包含 VCM driver 时候填 1。
vip_dev1_act_name	VCM driver 名字，如“ad5820_act“。
vip_dev1_act_slave	VCM driver slave 地址。

配置举例：

```
[csi0]
vip_used                = 1
vip_mode                = 0
vip_dev_qty             = 2
vip_define_sensor_list  = 0
vip_csi_pck             = port:PE00<2><default><default><default>
vip_csi_mck             = port:PE01<2><default><default><default>
vip_csi_hsync           = port:PE02<2><default><default><default>
vip_csi_vsync           = port:PE03<2><default><default><default>
vip_csi_d0              = port:PE04<2><default><default><default>
vip_csi_d1              = port:PE05<2><default><default><default>
vip_csi_d2              = port:PE06<2><default><default><default>
vip_csi_d3              = port:PE07<2><default><default><default>
vip_csi_d4              = port:PE08<2><default><default><default>
```

```
vip_csi_d5                = port:PE09<2><default><default><default>
vip_csi_d6                = port:PE10<2><default><default><default>
vip_csi_d7                = port:PE11<2><default><default><default>

vip_dev0_mname              = "gc2035"
vip_dev0_pos               = "rear"
vip_dev0_lane              = 1
vip_dev0_twi_id            = 2
vip_dev0_twi_addr          = 0x78
vip_dev0_isp_used          = 0
vip_dev0_fmt               = 0
vip_dev0_stby_mode         = 0
vip_dev0_vflip             = 1
vip_dev0_hflip             = 1
vip_dev0_iovdd             = "axp22_dldo3"
vip_dev0_iovdd_vol         = 2800000
vip_dev0_avdd              = "axp22_dldo3"
vip_dev0_avdd_vol          = 2800000
vip_dev0_dvdd              = "axp22_eldo2"
vip_dev0_dvdd_vol          = 1800000
vip_dev0_afvdd             = "axp22_dldo3"
vip_dev0_afvdd_vol         = 2800000
vip_dev0_power_en          =
vip_dev0_reset             = port:PE14<1><default><default><0>
vip_dev0_pwdn                = port:PE17<1><default><default><1>
vip_dev0_flash_en          =
vip_dev0_flash_mode        =
vip_dev0_af_pwdn           =


vip_dev1_mname              = "gc0328"
vip_dev1_pos               = "front"
vip_dev1_lane              = 1
vip_dev1_twi_id            = 2
vip_dev1_twi_addr          = 0x42
vip_dev1_isp_used          = 0
vip_dev1_fmt               = 0
vip_dev1_stby_mode         = 0
vip_dev1_vflip             = 1
vip_dev1_hflip             = 1
vip_dev1_iovdd             = "axp22_dldo3"
vip_dev1_iovdd_vol         = 2800000
vip_dev1_avdd              = "axp22_dldo3"
vip_dev1_avdd_vol          = 2800000
```

```
vip_dev1_dvdd                = "axp22_eldo2"
vip_dev1_dvdd_vol            = 1800000
vip_dev1_afvdd               = "axp22_dldo3"
vip_dev1_afvdd_vol           = 2800000
vip_dev1_power_en            =
vip_dev1_reset               = port:PE14<1><default><default><0>
vip_dev1_pwdn                = port:PE15<1><default><default><1>
vip_dev1_flash_en            =
vip_dev1_flash_mode          =
vip_dev1_af_pwdn             =
```

16. SD / MMC

16.1. [mmc0_para]

配置项	配置项含义
sdc_used=xx	SDC 使用控制：1 使用，0 不用
sdc_detmode=xx	检测模式：1-gpio 轮询检测，2-gpio 中断检测，3-卡常在(不可拔插)，4 - manual mode(from proc file system node)，5-DAT3 检测
Sdc_buswidth=xx	位宽：1-1bit，4-4bit
sdc_d1=xx	SDC DATA1 的 GPIO 配置
sdc_d0＝xx	SDC DATA0 的 GPIO 配置
sdc_clk=xx	SDC CLK 的 GPIO 配置
sdc_cmd=xx	SDC CMD 的 GPIO 配置
sdc_d3=xx	SDC DATA3 的 GPIO 配置
sdc_d2=xx	SDC DATA2 的 GPIO 配置
sdc_det=xx	SDC DET 的 GPIO 配置
sdc_use_wp=xx	SDC 写保护配置：1 使用，0 不用
sdc_wp=xx	SDC WP 的 GPIO 配置
sdc_power_supply=xx	SDC 模块的供电电压
sdc_isio=xx	是否是 sdio card,0:不是，1：是
sdc_regulator=xx	假如过卡支持 SD3.0 或者 emmc4.5 的 UHS-I/DDR、HS200，这里就要写成 sdc_regulator = "axp22_eldo2"（原理图上对应的 PMU 供电输出）

配置举例：

```
[mmc0_para]
sdc_used                = 1
sdc_detmode             = 2
```

```
sdc_buswidth          = 4
sdc_d1                 = port:PF00<2><1><2><default>
sdc_d0                 = port:PF01<2><1><2><default>
sdc_clk               = port:PF02<2><1><2><default>
sdc_cmd                = port:PF03<2><1><2><default>
sdc_d3                 = port:PF04<2><1><2><default>
sdc_d2                 = port:PF05<2><1><2><default>
sdc_det               = port:PB04<4><1><2><default>
sdc_use_wp            = 0
sdc_wp                 =
sdc_isio              = 0
sdc_regulator         = "none"
sdc_power_supply = “axp22_dcdc1”
```

16.2. [mmc1_para]

配置项	配置项含义
sdc_used=xx	SDC 使用控制：1 使用，0 不用
sdc_detmode=xx	检测模式：1-gpio 轮询检测，2-gpio 中断检测，3-卡常在(不可拔插)，4 - manual mode(from proc file system node)，5-DAT3 检测
bus_width=xx	位宽：1-1bit，4-4bit
sdc_d1=xx	SDC DATA1 GPIO 配置
sdc_d0＝xx	SDC DATA0 GPIO 配置
sdc_clk=xx	SDC CLK GPIO 配置
sdc_cmd=xx	SDC CMD GPIO 配置
sdc_d3=xx	SDC DATA3 GPIO 配置
sdc_d2=xx	SDC DATA2 GPIO 配置
sdc_det=xx	SDC DET GPIO 配置
sdc_use_wp=xx	SDC 写保护配置：1 使用，0 不用
sdc_wp=xx	SDC WP GPIO 配置
sdc_isio =xx	是否是 sdio card,0:不是，1：是
sdc_regulator =xx	假如过卡支持 SD3.0 或者 emmc4.5 的 UHS-I/DDR、HS200，这里就要写成 sdc_regulator = "axp22_eldo2"（原理图上对应的 PMU 供电输出）

配置举例：

```
[mmc1_para]
sdc_used               = 1
sdc_detmode            = 4
sdc_buswidth           = 4
sdc_clk                = port:PG00<2><1><1><default>
sdc_cmd                = port:PG01<2><1><1><default>
```

```
sdc_d0                  = port:PG02<2><1><1><default>
sdc_d1                  = port:PG03<2><1><1><default>
sdc_d2                  = port:PG04<2><1><1><default>
sdc_d3                  = port:PG05<2><1><1><default>
sdc_det                 =
sdc_use_wp              = 0
sdc_wp                  =
sdc_isio                = 1
sdc_regulator           = "none"
```

16.3. [mmc2_para]

配置项	配置项含义
sdc_used=xx	SDC 使用控制：1 使用，0 不用
sdc_detmode=xx	检测模式：1-gpio 轮询检测，2-gpio 中断检测，3-卡常在(不可拔插)，4 - manual mode(from proc file system node)，5-DAT3 检测
Sdc_bus_width=xx	位宽：1-1bit，4-4bit，8-8bit
Sdc_2xmode=xx	2x 采样模式，1：开启，0：关闭，默认开启
sdc_d1=xx	SDC DATA1 GPIO 配置
sdc_d0＝xx	SDC DATA0 GPIO 配置
sdc_clk=xx	SDC CLK GPIO 配置
sdc_cmd=xx	SDC CMD GPIO 配置
sdc_d3=xx	SDC DATA3 GPIO 配置
sdc_d2=xx	SDC DATA2 GPIO 配置
sdc_d4 =xx	SDC DATA4GPIO 配置
sdc_d5 =xx	SDC DATA5 GPIO 配置
sdc_d6 =xx	SDC DATA6 GPIO 配置
sdc_d7 =xx	SDC DATA7 GPIO 配置
sdc_det=xx	SDC DET GPIO 配置
sdc_use_wp=xx	SDC 写保护配置：1 使用，0 不用
sdc_wp=xx	SDC WP GPIO 配置
sdc_power_supply	SDC 模块的供电电压
emmc_rst = xx	Emmc 的 reset 管脚
sdc_isio=xx	是否是 sdio card,0:不是，1：是
sdc_regulator=xx	假如过卡支持 SD3.0 或者 emmc4.5 的 UHS-I/DDR、HS200，这里就要写成 sdc_regulator = "axp22_eldo2"

配置举例：

```
[mmc2_para]
sdc_used                = 0
sdc_detmode             = 3
```

```
sdc_buswidth          = 8
sdc_2xmode       = 1
sdc_clk                = port:PC05<3><1><2><default>
sdc_cmd                 = port:PC06<3><1><2><default>
sdc_d0                  = port:PC08<3><1><2><default>
sdc_d1                  = port:PC09<3><1><2><default>
sdc_d2                  = port:PC10<3><1><2><default>
sdc_d3                  = port:PC11<3><1><2><default>
sdc_d4                  = port:PC12<3><1><2><default>
sdc_d5                  = port:PC13<3><1><2><default>
sdc_d6                  = port:PC14<3><1><2><default>
sdc_d7                  = port:PC15<3><1><2><default>
emmc_rst                = port:PC16<3><1><2><default>
sdc_power_supply    = "axp22_dcdc1"
sdc_det                 =
sdc_use_wp              = 0
sdc_wp                  =
sdc_isio                = 0
sdc_regulator        = "none"
```

17. SIM 卡

17.1. [smc_para]

配置项	配置项含义
smc_used =xx	
smc_rst=xx	
smc_vppen=xx	
smc_vppp=xx	
smc_det＝xx	
smc_vccen=xx	
smc_sck=xx	
smc_sda=xx	

配置举例：

18. USB 控制标志

18.1. [usbc0]

配置项	配置项含义
usb_used =xx	USB 使能标志(xx=1 or 0)。置 1，表示系统中 USB 模块可用，置 0，则表示系统 USB 禁用。此标志只对具体的 USB 控制器模块有效。
usb_port_type =xx	USB 端口的使用情况。(xx=0/1/2) 0：device only 1：host only 2：OTG
usb_detect_type=xx	USB 端口的检查方式。 0：无检查方式 1：vbus/id 检查
usb_id_gpio=xx	USB ID pin 脚配置。具体请参考 gpio 配置说明。《配置与 GPIO 管理.doc》
usb_det_vbus_gpio＝xx	USB DET_VBUS pin 脚配置。如果 GPIO 提供 pin，请参考 gpio 配置说明《配置与 GPIO 管理.doc》。如果的 AXP 提供 pin,则配置为："axp_ctrl"。
usb_drv_vbus_gpio=xx	USB DRY_VBUS pin 脚配置。具体请参考 gpio 配置说明。《配置与 GPIO 管理.doc》
usb_restrict_gpio=xx	USB 限流控制 pin 脚 USB RESTRICT_GPIO pin 脚配置。具体请参考 gpio 配置说明。《配置与 GPIO 管理.doc》
usb_host_init_state=xx	host only 模式下，Host 端口初始化状态。 0：初始化后 USB 不工作 1：初始化后 USB 工作
usb_restric_flag=xx	Usb 限流标志位 0：不使能限流功能 1：使能限流功能
usb_restric_voltage=xx	限流开启的条件 电压值小于设置值，则开启限流
usb_restric_capacity=xx	限流开启的条件 电量值小于设置值，则开启限流

配置举例：

```
[usbc0]
usb_used                = 1
usb_port_type           = 2
usb_detect_type         = 1
usb_id_gpio             = port:PH08<0><1><default><default>
```

```
usb_det_vbus_gpio     = "axp_ctrl"
usb_drv_vbus_gpio     = port:power4<1><0><default><0>
usb_restrict_gpio     =
usb_host_init_state = 0
usb_restric_flag      = 0
usb_restric_voltage = 3550000
usb_restric_capacity= 5

[usbc1]
usb_used                = 1
usb_port_type           = 1
usb_detect_type         = 0
usb_id_gpio             =
usb_det_vbus_gpio       =
usb_drv_vbus_gpio       =
usb_restrict_gpio       =
usb_host_init_state = 1
usb_restric_flag        = 0
```

18.2. [usbc1]

配置项	配置项含义
usb_used =xx	USB 使能标志(xx=1 or 0)。置 1，表示系统中 USB 模块可用，置 0，则表示系统 USB 禁用。此标志只对具体的 USB 控制器模块有效。
usb_port_type =xx	USB 端口的使用情况。(xx=0/1/2) 0：device only　1：host only　2：OTG
usb_detect_type=xx	USB 端口的检查方式。 0：无检查方式　1：vbus/id 检查
usb_det_vbus_gpio＝xx	USB DET_VBUS pin 脚配置。具体请参考 gpio 配置说明。《配置与 GPIO 管理.doc》
usb_restrict_gpio=xx	USB 限流控制 pin 脚 USB RESTRICT_GPIO pin 脚配置。具体请参考 gpio 配置说明。《配置与 GPIO 管理.doc》
usb_host_init_state=xx	host only 模式下，Host 端口初始化状态。 0：初始化后 USB 不工作　1：初始化后 USB 工作
usb_restric_flag=xx	Usb 限流标志位 0：表不设限流，1 开启限流

配置举例：

```
[usbc1]
usb_used                = 1
usb_port_type           = 1
usb_detect_type         = 0
usb_id_gpio             =
usb_det_vbus_gpio       =
usb_drv_vbus_gpio       =
usb_restrict_gpio       =
usb_host_init_state = 1
usb_restric_flag        = 0
```

19. USB Device

19.1. [usb_feature]

配置项	配置项含义
vendor_id=xx	USB 厂商 ID
mass_storage_id =xx	U 盘 ID
adb_id =xx	USB 调试桥 ID
manufacturer_name=xx	USB 厂商名
product_name＝xx	USB 产品名
serial_number=xx	USB 序列号

配置举例：

```
[usb_feature]
vendor_id               = 0x1F3A
mass_storage_id         = 0x1000
adb_id                  = 0x1001

manufacturer_name       = "USB Developer"
product_name            = "Android"
serial_number           = "20080411"
```

19.2. [msc_feature]

配置项	配置项含义
vendor_name=xx	U 盘 厂商名
product_name=xx	U 盘产品名
release=xx	发布版本
luns=xx	U 盘逻辑单元的个数（PC 可以看到的 U 盘盘符的个数）

配置举例：

```
[msc_feature]
vendor_name             = "USB 2.0"
product_name            = "USB Flash Driver"
release                 = 100
luns                    = 3
```

20. serial_feature

配置项	配置项含义
serial_unique	usb 序列号开关，0：不用，1：用

配置举例：

```
[serial_feature]
serial_unique           = 0
```

21. 重力感应(G Sensor)

21.1. [gsensor_para]

配置项	配置项含义
gsensor_used=xx	是否支持 gsensor
gsensor_twi_id =xx	I2C 的 BUS 控制选择，0：TWI0;1:TWI1;2:TWI2
gsensor_twi_addr=xx	芯片的 I2C 地址
gsensor_int1＝xx	中断 1 的 GPIO 配置
gsensor_int2=xx	中断 2 的 GPIO 配置

配置举例：

```
[gsensor_para]
gsensor_used            = 1
gsensor_twi_id          = 1
gsensor_twi_addr        = 0x18
gsensor_int1            = port:PB06<4><1><default><default>
gsensor_int2            =
```

22. 重力感应(G Sensor)自动扫描配置

注：目前方案中支持 gsensor 的类型有以下列表，作为自动检测加载的时候使用，为‘1’时检测加载，为‘0’时不检测。

22.1. [gsensor_list_para]

配置项	配置项含义
gsensor_det_used	0 or 1，是否使用自动检测功能
bma250	0 or 1，检测时是否扫描此类触屏
mma8452	0 or 1，检测时是否扫描此类触屏
mma7660	0 or 1，检测时是否扫描此类触屏
mma865x	0 or 1，检测时是否扫描此类触屏
afa750	0 or 1，检测时是否扫描此类触屏
lis3de_acc	0 or 1，检测时是否扫描此类触屏
lis3dh_acc	0 or 1，检测时是否扫描此类触屏
kxtik	0 or 1，检测时是否扫描此类触屏
dmard10	0 or 1，检测时是否扫描此类触屏
dmard06	0 or 1，检测时是否扫描此类触屏
mxc622x	0 or 1，检测时是否扫描此类触屏
fxos8700	0 or 1，检测时是否扫描此类触屏
lsm303d	0 or 1，检测时是否扫描此类触屏

配置举例：

```
[gsensor_list_para]
gsensor_det_used          = 1
bma250                    = 1
mma8452                   = 1
mma7660                   = 1
mma865x                   = 1
afa750                    = 1
lis3de_acc                = 1
lis3dh_acc                = 1
kxtik                     = 1
dmard10                   = 0
dmard06                   = 1
mxc622x                   = 1
fxos8700                  = 1
lsm303d                   = 1
```

23. WiFi

23.1. [wifi_para]

配置项	配置项含义
wifi_used =xx	是否要使用 wifi
wifi_sdc_id =xx	sdio wifi 选用的是哪个 sdc 作为接口
wifi_usbc_id =xx	usb wifi 选用的是哪个 usb 作为接口
wifi_usbc_type=xx	usb 接口类型，1 为 ehci，0 为 ohci
wifi_mod_sel =xx	具体选择哪一款模组 1-ap6181， 2 - ap6210; 3- rtl8188eu, 4 -rtl8723au 5 - rtl8723bs, 6 - esp8089
wifi_power="xx" wifi_power_ext1 = "xx" wifi_power_ext2 = "xx"	给模组供电的 axp 引脚名
wifi_power_switch=xx	WiFi 模组电源开关

说明：[wifi_para]下的配置项是 usb 和 sdio 接口 wifi 共用的。

23.2. sdio 接口 wifi rtl8723bs demo

```
[wifi_para]
wifi_used               = 1
wifi_sdc_id             = 1
wifi_usbc_id            = 1
wifi_usbc_type          = 1
wifi_mod_sel            = 5
wifi_power              = ""
wifi_power_ext1         = ""
wifi_power_ext2         = ""
wifi_power_switch= port:power0<1><0><default><0>

;5 - rtl8723bs sdio wifi + bt
;rtl8723bs_chip_en            = port:PL11<1><default><default><0>
rtl8723bs_wl_regon           = port:PL06<1><default><default><0>
rtl8723bs_wl_host_wake    = port:PL07<4><default><default><0>
rtl8723bs_bt_regon           = port:PL08<1><default><default><0>
rtl8723bs_bt_wake            = port:PL10<1><default><default><0>
rtl8723bs_bt_host_wake    = port:PL09<4><default><default><0>
rtl8723bs_lpo_use_apclk = 0
```

以上配置意思是要使用序号为 5 的 SDIO 接口 rtl8723bs 模组，选用 SDC1 接口。
Rtl8723bs 模组采用电池供电，3 组 wifi_power 都为空串，且有 axp power0 作为电源开

关。
SDC1 对应是 mmc1，需要确定[mmc1_para]配置项如下：

```
[mmc1_para]
sdc_used            = 1
sdc_detmode         = 4
sdc_buswidth        = 4
sdc_clk             = port:PG00<2><1><2><default>
sdc_cmd             = port:PG01<2><1><2><default>
sdc_d0              = port:PG02<2><1><2><default>
sdc_d1              = port:PG03<2><1><2><default>
sdc_d2              = port:PG04<2><1><2><default>
sdc_d3              = port:PG05<2><1><2><default>
sdc_det             =
sdc_use_wp          = 0
sdc_wp              =
sdc_isio            = 1
sdc_regulator       = "none"
```

23.3. usb 接口 wifi rtl8188eu demo

```
[wifi_para]
wifi_used           = 1
wifi_sdc_id         = 1
wifi_usbc_id        = 1
wifi_usbc_type      = 1
wifi_mod_sel        = 3
wifi_power          = "axp22_aldo1"
wifi_power          = "axp22_aldo2"
wifi_power_ext2     = ""
wifi_power_ext2     =
```

以上配置意思是要使用序号为 6 的 ehci USB 接口 rtl8188eu 模组，选用 usb1 接口。需要确定[usbc1]配置项如下：

```
[usbc1]
usb_used            = 1
usb_port_type       = 1
usb_detect_type     = 0
usb_id_gpio         =
usb_det_vbus_gpio   =
usb_drv_vbus_gpio   =
usb_restrict_gpio   =
usb_host_init_state = 0
usb_restric_flag    = 0
```

24. 3G

24.1. [3g_para]

配置项	配置项含义
3g_used	3G 使能标志位。 0：禁用；1：使能
3g_usbc_num	3G 使用到的 USB 控制器编号。 0：USB0；1：USB1；2：USB2；3：USB3 等
3g_uart_num	3G 使用到的 UART 控制器编号。 0：UART0；1：UART1；2：UART2；3：UART3 等
bb_name	3G 模组名称。如“mu509”
bb_vbat	gpio 配置，电池引脚。
bb_on	保留
bb_pwr_on	gpio 配置，供电引脚。
bb_wake	gpio 配置，A31 睡眠唤醒 3G 模组。
bb_rf_dis	gpio 配置，用来控制无线发射模块。
bb_rst	gpio 配置，用来复位 3G 模组。
bb_dldo	电源控制配置
bb_dldo_min_uV	电压下限
bb_dldl_max_uV	电压上限

配置举例：

```
[3g_para]
3g_used       = 0
3g_usbc_num     = 1
3g_uart_num     = 2
bb_name          = "em66"
bb_vbat       =
bb_on         =
bb_pwr_on   = port:PL03<1><default><default><0>
bb_wake          = port:PL04<1><default><default><0>
bb_rf_dis     = port:PL11<1><default><default><0>
bb_rst          = port:PL05<1><default><default><0>
bb_dldo              = "axp22_aldo1"
bb_dldo_min_uV      = 2800000
bb_dldo_max_uV      = 2800000
```

25. 光感(light sensor)

25.1. [ls_para]

配置项	配置项含义
ls_used =xx	是否支持 ls
ls_twi_id=xx	I2C 的 BUS 控制选择，0：TWI0;1:TWI1;2:TWI2
ls_twi_addr =xx	芯片的 I2C 地址
ls_int=xx	中断的 GPIO 配置

配置举例：

```
[ls_para]
ls_used             = 0
ls_twi_id           = 1
ls_twi_addr         = 0x23
ls_int              = port:PB07<4><1><default><default>
```

26. 罗盘 Compass

26.1. [compass_para]

配置项	配置项含义
compass_used=xx	是否支持 compass
compass_twi_id=xx	I2C 的 BUS 控制选择，0：TWI0;1:TWI1;2:TWI2
compass_twi_addr =xx	芯片的 I2C 地址
compass_int =xx	中断的 GPIO 配置

配置举例：

```
[compass_para]
compass_used        = 0
compass_twi_id      = 1
compass_twi_addr    = 0x0d
compass_int         =
```

27. 蓝牙(blueteeth)

27.1. [bt_para]

配置项	配置项含义
bt_used=xx	BLUETOOTH 使用控制：1 使用，0 不用
bt_uart_id=xx	BLUETOOTH 使用的 UART 控制器号

配置举例：

```
[bt_para]
bt_used                 = 1
bt_uart_id              = 1
```

28. 数字音频总线（I2S）

28.1. [i2s0]

配置项	配置项含义
i2s_used	使能配置
i2s_channel	通道数配置
i2s_master	主从模式配置，1：主模式，0：从模式
i2s_select	I2S、PCM 选择，1：I2S 模式，0：PCM 模式
audio_format	1:SND_SOC_DAIFMT_I2S(standard i2s format). 2:SND_SOC_DAIFMT_RIGHT_J(right justfied format). 3:SND_SOC_DAIFMT_LEFT_J(left justfied format) 4:SND_SOC_DAIFMT_DSP_A(pcm. MSB is available on 2nd BCLK rising edge after LRC rising edge). 5:SND_SOC_DAIFMT_DSP_B(pcm. MSB is available on 1nd BCLK rising edge after LRC
signal_inversion	1:SND_SOC_DAIFMT_NB_NF(normal bit clock + frame) 2:SND_SOC_DAIFMT_NB_IF(normal BCLK + inv FRM) 3:SND_SOC_DAIFMT_IB_NF(invert BCLK + nor FRM) 4:SND_SOC_DAIFMT_IB_IF(invert BCLK + FRM)
over_sample_rate	过采样率，128fs/192fs/256fs/384fs/512fs/768fs
sample_resolution	采样精度，16bits/20bits/24bit
word_select_size	
pcm_sync_period	同步周期，16/32/64/128/256
msb_lsb_first	0: msb first; 1: lsb first
sign_extend	信号扩展，0: zero pending; 1: sign extend

slot_index	slot 标签，0: the 1st slot - 3: the 4th slot
slot_width	slot 位宽，8 bit width / 16 bit width
frame_width	帧模式，0: long frame = 2 clock width;　1: short frame
tx_data_mode	发送模式：0: 16bit linear PCM; 1: 8bit linear PCM; 2: 8bit u-law; 3: 8bit a-law
rx_data_mode	接收模式：0: 16bit linear PCM; 1: 8bit linear PCM; 2: 8bit u-law; 3: 8bit a-law
i2s_mclk	IO 信号定义
i2s_bclk	IO 信号定义
i2s_lrclk	IO 信号定义
i2s_dout0	IO 信号定义
i2s_dout1	IO 信号定义
i2s_dout2	IO 信号定义
i2s_dout3	IO 信号定义
i2s_din	IO 信号定义

配置举例：

```
[i2s0]
i2s0_used               = 0
i2s0_channel            = 2
i2s0_master             = 4
i2s0_select             = 1
audio_format            = 1
signal_inversion = 1
over_sample_rate        = 512
sample_resolution       = 16
word_select_size        = 32
pcm_sync_period         = 256
msb_lsb_first           = 0
sign_extend             = 0
slot_index              = 0
slot_width              = 16
frame_width             = 1
tx_data_mode            = 1
rx_data_mode            = 1
i2s0_mclk               =
i2s0_bclk               = port:PB04<2><1><default><default>
i2s0_lrclk              = port:PB05<2><1><default><default>
i2s0_dout0              = port:PB06<2><1><default><default>
i2s0_dout1              =
i2s0_dout2              =
i2s0_dout3              =
i2s0_din                = port:PB07<2><1><default><default>
```

29. 数字音频总线(pcm)

29.1. [pcm_para]

配置项	配置项含义
pcm_used	使能配置
pcm_channel	通道数配置
pcm_master	主从模式配置，1：主模式，0：从模式
pcm_select	I2S、PCM 选择，1：I2S 模式，0：PCM 模式
audio_format	1:SND_SOC_DAIFMT_I2S(standard i2s format). 2:SND_SOC_DAIFMT_RIGHT_J(right justfied format). 3:SND_SOC_DAIFMT_LEFT_J(left justfied format) 4:SND_SOC_DAIFMT_DSP_A(pcm. MSB is available on 2nd BCLK rising edge after LRC rising edge). 5:SND_SOC_DAIFMT_DSP_B(pcm. MSB is available on 1nd BCLK rising edge after LRC
signal_inversion	1:SND_SOC_DAIFMT_NB_NF(normal bit clock + frame) 2:SND_SOC_DAIFMT_NB_IF(normal BCLK + inv FRM) 3:SND_SOC_DAIFMT_IB_NF(invert BCLK + nor FRM) 4:SND_SOC_DAIFMT_IB_IF(invert BCLK + FRM)
over_sample_rate	过采样率，128fs/192fs/256fs/384fs/512fs/768fs
sample_resolution	采样精度，16bits/20bits/24bit
word_select_size	
pcm_sync_period	同步周期，16/32/64/128/256
msb_lsb_first	0: msb first; 1: lsb first
sign_extend	信号扩展，0: zero pending; 1: sign extend
slot_index	slot 标签，0: the 1st slot - 3: the 4th slot
slot_width	slot 位宽，8 bit width / 16 bit width
frame_width	帧模式，0: long frame = 2 clock width; 1: short frame
tx_data_mode	发送模式：0: 16bit linear PCM; 1: 8bit linear PCM; 2: 8bit u-law; 3: 8bit a-law
rx_data_mode	接收模式：0: 16bit linear PCM; 1: 8bit linear PCM; 2: 8bit u-law; 3: 8bit a-law
i2s_mclk	IO 信号定义
i2s_bclk	IO 信号定义
i2s_lrclk	IO 信号定义
i2s_dout0	IO 信号定义
i2s_din	IO 信号定义

配置举例：
[i2s1]
i2s1_used = 0

```
i2s1_channel          = 2
i2s1_master           = 4
i2s1_select           = 1
audio_format          = 1
signal_inversion = 1
over_sample_rate      = 512
sample_resolution     = 16
word_select_size      = 32
pcm_sync_period       = 64
msb_lsb_first         = 0
sign_extend           = 0
slot_index            = 0
slot_width            = 16
frame_width           = 1
tx_data_mode          = 0
rx_data_mode          = 0
i2s1_mclk             =
i2s1_bclk             = port:PG11<2><1><default><default>
i2s1_lrclk            = port:PG10<2><1><default><default>
i2s1_dout             = port:PG12<2><1><default><default>
i2s1_din              = port:PG13<2><1><default><default>
```

30. 内置音频 codec

30.1. [audio_para]

配置项	配置项含义
audio_used	Audiocodec 是否使用， 1：打开（默认）0：关闭
audio_hp_ldo	独立电源控制
headphone_vol	耳机音量大小
cap_vol	录音音量大小
pa_single_vol	喇叭音量大小（硬件只有一个喇叭）
pa_double_used	硬件是否支持双喇叭配置
pa_double_vol	双喇叭音量大小（如果只有一个喇叭， 这个可以不配置）
headphone_direct_used	耳机的直驱交驱选择（建议 3g-phone 方案选择交驱，pad 方案选择直驱）
audio_pa_ctrl	喇叭的 gpio 口控制。
headphone_mute_used	耳机静音设置
headset_mic_vol	耳机 mic 增益
main_mic_vol	主 mic 增益

配置举例：

```
[audio0]
audio_used                  = 1
headphone_vol               = 0x3b
earpiece_vol = 0x3b
cap_vol                     = 0x5
pa_single_vol               = 0x3e
pa_double_used              = 0
pa_double_vol               = 0x1f
headphone_direct_used    = 1
headset_mic_vol = 0x6
main_mic_vol              = 0x6
audio_hp_ldo            =none
audio_pa_ctrl                = port:PH09<1><default><default><0>
;audio_pa_ctrl               = port:PA18<1><default><default><0>
aif2_used                    = 0
aif3_used                    = 0
headphone_mute_used = 0
DAC_VOL_CTRL_SPK              = 0x9e9e
DAC_VOL_CTRL_HEADPHONE = 0xa0a0
;main_mic_vol            = 6
```

31. PMU 电源

31.1. [pmu1_para]

配置项	相关说明
pmu_used	是否使用 AXPxx：0:不使用,1:使用
pmu_twi_addr	AXPxx 通信 I2C 地址
pmu_twi_id	AXPxx 挂接在主控的哪个 I2C 控制口（0，1，2 ...）
pmu_irq_id	irq 号 (0 irq0,1 irq1,……)
pmu_battery_rdc	电池通路内阻，单位 mΩ
pmu_battery_cap	电池容量,单位 mAh，如果配置改值，计量方式为库仑计方式，否则为电压方式
pmu_batdeten	电池检查使能控制：0:使能 1:使能
pmu_runtime_chgcur	设置开机时充电电流大小,单位 mA，仅支持:300/450/600/750 /900/1050/1200/1350/1500/1650/1800/1950/2100
pmu_earlysuspend_chgcur	设置关屏时充电电流大小，单位 mA，仅支持：300/4500/600/750/900/1050/1200/1350/1500/1650/1800/

	1950/ 2100
pmu_suspend_chgcur	设置待机时充电电流大小，单位 mA，仅支持： 300/4500/600/750 /900/1050/1200/1350/1500/1650/1800/1950/2100
pmu_shutdown_chgcur	设置关机时充电电流大小，单位 mA，仅支持： 300/4500/600/750 /900/1050/1200/1350/1500/1650/1800/1950/2100
pmu_init_chgvol	设置充电完成时电池目标电压,仅支持： 4100/4200/4220/4240mV
pmu_init_chgend_rate	设置充电结束时电流占恒流值的百分比：10/15
pmu_init_chg_enabled	开机后充电使能初始值：0：不开充电，1：开充电
pmu_init_adc_freq	ADC 采样频率设定值：100/200/400/800 Hz
pmu_init_adcts_freq	TS ADC 采样频率设定值：100/200/400/800 Hz
pmu_init_chg_pretime	涓流充电超时时间：40/50/60/70 分钟
pmu_init_chg_csttime	恒流超时时间：360/480/600/720 分钟
pmu_batt_cap_correct	满足电池容量校正条件后是否校正电池容量控制 0： 不校正 1：校正
pmu_bat_regu_en	充电结束时，充电开关是否关闭：0：关闭 1：不关闭
pmu_bat_para1	电池空载电压为 3.13V 对应的电量值
pmu_bat_para2	电池空载电压为 3.27V 对应的电量值
pmu_bat_para3	电池空载电压为 3.34V 对应的电量值
pmu_bat_para4	电池空载电压为 3.41V 对应的电量值
pmu_bat_para5	电池空载电压为 3.58V 对应的电量值
pmu_bat_para6	电池空载电压为 3.52V 对应的电量值
pmu_bat_para7	电池空载电压为 3.55V 对应的电量值
pmu_bat_para8	电池空载电压为 3.57V 对应的电量值
pmu_bat_para9	电池空载电压为 3.59V 对应的电量值
pmu_bat_para10	电池空载电压为 3.61V 对应的电量值
pmu_bat_para11	电池空载电压为 3.63V 对应的电量值
pmu_bat_para12	电池空载电压为 3.64V 对应的电量值
pmu_bat_para13	电池空载电压为 3.66V 对应的电量值
pmu_bat_para14	电池空载电压为 3.7V 对应的电量值
pmu_bat_para15	电池空载电压为 3.73V 对应的电量值
pmu_bat_para16	电池空载电压为 3.77V 对应的电量值
pmu_bat_para17	电池空载电压为 3.78V 对应的电量值
pmu_bat_para18	电池空载电压为 3.8V 对应的电量值
pmu_bat_para19	电池空载电压为 3.82V 对应的电量值
pmu_bat_para20	电池空载电压为 3.84V 对应的电量值
pmu_bat_para21	电池空载电压为 3.85V 对应的电量值
pmu_bat_para22	电池空载电压为 3.87V 对应的电量值
pmu_bat_para23	电池空载电压为 3.91V 对应的电量值
pmu_bat_para24	电池空载电压为 3.94V 对应的电量值

pmu_bat_para25	电池空载电压为 3.98V 对应的电量值
pmu_bat_para26	电池空载电压为 4.01V 对应的电量值
pmu_bat_para27	电池空载电压为 4.05V 对应的电量值
pmu_bat_para28	电池空载电压为 4.08V 对应的电量值
pmu_bat_para29	电池空载电压为 4.1V 对应的电量值
pmu_bat_para30	电池空载电压为 4.12V 对应的电量值
pmu_bat_para31	电池空载电压为 4.14V 对应的电量值
pmu_bat_para32	电池空载电压为 4.15V 对应的电量值
pmu_usbvol_limit	USB 适配器限压功能控制 0：不使能 1：使能
pmu_usbcur_limit	USB 适配器限流功能控制 0：不使能 1：使能
pmu_usbvol	设置 USB 适配器限压值： 4000/4100/4200/4300/4400/4500/4600 4700 mV，0-不限压
pmu_usbcur	设置 USB 适配器限流值：500/900mA，0-不限流
pmu_usbvol_pc	设置 USB 连接 PC 时限压值： 4000/4100/4200/4300/4400/4500 4600/4700 mV，0-不限压
pmu_usbcur_pc	设置 USB 连接 PC 时限流值：500/900mA，0-不限流
pmu_pwroff_vol	PMU 关机时，硬件低电保护电压设置值： 2600/2700/2800/2900 /3000/3100/3200/3300 mV
pmu_pwron_vol	PMU 开机后，硬件低电保护电压设置值： 2600/2700/2800/2900 /3000/3100/3200/3300 mV
pmu_pekoff_time	长按键关机时间设置值：4000/6000/8000/10000 ms
pmu_pekoff_func	长按键功能配置项：0：长按键后关机 1：长按键后重启
pmu_pekoff_en	长按键后是否关闭 PMU：0：不关闭 1：关闭
pmu_pekoff_delay_time	长按键关机激活时间设置， 0/10/20/30/40/50/60/70 秒
pmu_peklong_time	报长按键消息时间设定值：1000/1500/2000/2500 ms
pmu_pekon_time	关机情况下按键多长时间后启动设置： 128/1000/2000/3000 ms
pmu_pwrok_time	PWROK 启动延时时间设置值：8/16/32/64 ms
pmu_pwrok_shutdown_en	长按 PWROK 键 6s 是否关机，使能位
pmu_battery_warning_level1	低电报警门限 level 1 设置值百分比：5~20，每步设置 1%
pmu_battery_warning_level2	低电报警门限 level 2 设置值百分比：0~15，每步设置 1%
pmu_restvol_time	电池电量更新时间设置值：30/60/120 s
pmu_ocv_cou_adjust_time	根据 OCV 校正电池电量更新时间值：30/60/120 s
pmu_chgled_func	CHGLED 功能控制：0：马达驱动 1：充电状态指示

pmu_chgled_type	CHGLED 作为充电状态指示时指示功能控制：0：方式 A 1：方式 B
pmu_vbusen_func	N_VBUSEN 工作方式控制：0：作为输入脚 1：作为输出脚
pmu_reset	长按键 16s 后 PMU 是否重启控制：0：不重启 1：重启
pmu_IRQ_wakeup	在关机和休眠状态下 IRQ 为低电平时是否触发开机和唤醒控制 0：不开机或不唤醒 1：开机或唤醒
pmu_hot_shutdowm	PMU 过温后是否关机 0：不关机 1：关机
pmu_inshort	是否手动设置 ACIN/VBUS 短路控制 0: PMU 自动检测 1：手动设置 ACIN 和 VBUS 为短路
power_start	火牛开机选择 0: 不允许插火牛直接开机，必须通过判断：满足以下条件可以直接开机：长按 power 按键，前次是系统状态，如果电池电量过低，则不允许开机 1: 任意状态下，允许插火牛直接开机，同时要求电池电量足够高 2: 不允许插火牛直接开机，必须通过判断：满足以下条件可以直接开机：长按 power 按键，前次是系统状态，不要求电池电量 3: 任意状态下，允许插火牛直接开机，不要求电池电量
pmu_temp_enable	电池温度检测使能控制：0：disable 1：enable
pmu_charge_ltf	充电下限电池温度对应的电压
pmu_charge_htf	充电上限电池温度对应的电压
pmu_discharge_ltf	关机下限电池温度对应的电压
pmu_discharge_htf	关机上限电池温度对应的电压
pmu_temp_para1	电池温度-25 度对应的电压
pmu_temp_para2	电池温度-15 度对应的电压
pmu_temp_para3	电池温度-10 度对应的电压
pmu_temp_para4	电池温度-5 度对应的电压
pmu_temp_para5	电池温度 0 度对应的电压
pmu_temp_para6	电池温度 5 度对应的电压
pmu_temp_para7	电池温度 10 度对应的电压
pmu_temp_para8	电池温度 20 度对应的电压
pmu_temp_para9	电池温度 30 度对应的电压
pmu_temp_para10	电池温度 40 度对应的电压
pmu_temp_para11	电池温度 45 度对应的电压
pmu_temp_para12	电池温度 50 度对应的电压
pmu_temp_para13	电池温度 55 度对应的电压
pmu_temp_para14	电池温度 60 度对应的电压
pmu_temp_para15	电池温度 70 度对应的电压
pmu_temp_para16	电池温度 80 度对应的电压

配置举例：

```
[pmu1_para]
pmu_used                    = 1
pmu_twi_addr                = 0x34
pmu_twi_id                  = 1
pmu_irq_id                  = 0
pmu_battery_rdc             = 100
pmu_battery_cap             = 0
pmu_batdeten                = 1
pmu_runtime_chgcur          = 900
pmu_earlysuspend_chgcur     = 1200
pmu_suspend_chgcur          = 1500
pmu_shutdown_chgcur         = 1500
pmu_init_chgvol             = 4200
pmu_init_chgend_rate        = 15
pmu_init_chg_enabled        = 1
pmu_init_adc_freq           = 800
pmu_init_adcts_freq         = 800
pmu_init_chg_pretime        = 70
pmu_init_chg_csttime        = 720
pmu_batt_cap_correct        = 1
pmu_bat_regu_en             = 0

pmu_bat_para1               = 0
pmu_bat_para2               = 0
pmu_bat_para3               = 0
pmu_bat_para4               = 0
pmu_bat_para5               = 0
pmu_bat_para6               = 0
pmu_bat_para7               = 0
pmu_bat_para8               = 0
pmu_bat_para9               = 5
pmu_bat_para10              = 8
pmu_bat_para11              = 9
pmu_bat_para12              = 10
pmu_bat_para13              = 13
pmu_bat_para14              = 16
pmu_bat_para15              = 20
pmu_bat_para16              = 33
pmu_bat_para17              = 41
pmu_bat_para18              = 46
pmu_bat_para19              = 50
pmu_bat_para20              = 53
pmu_bat_para21              = 57
```

pmu_bat_para22 = 61
pmu_bat_para23 = 67
pmu_bat_para24 = 73
pmu_bat_para25 = 78
pmu_bat_para26 = 84
pmu_bat_para27 = 88
pmu_bat_para28 = 92
pmu_bat_para29 = 93
pmu_bat_para30 = 94
pmu_bat_para31 = 95
pmu_bat_para32 = 100

pmu_usbvol_limit = 0
pmu_usbcur_limit = 0
pmu_usbvol = 4000
pmu_usbcur = 0
pmu_usbvol_pc = 4400
pmu_usbcur_pc = 500
pmu_pwroff_vol = 3300
pmu_pwron_vol = 2600
pmu_pekoff_time = 6000
pmu_pekoff_func = 1
pmu_pekoff_en = 1
pmu_peklong_time = 1500
pmu_pekon_time = 1000
pmu_pwrok_time = 64
pmu_battery_warning_level1 = 15
pmu_battery_warning_level2 = 0
pmu_restvol_adjust_time = 60
pmu_ocv_cou_adjust_time = 60
pmu_chgled_func = 0
pmu_chgled_type = 0
pmu_vbusen_func = 1
pmu_reset = 0
pmu_IRQ_wakeup = 0
pmu_hot_shutdowm = 1
pmu_inshort = 0
power_start = 0

pmu_temp_enable = 1
pmu_charge_ltf = 2261
pmu_charge_htf = 388
pmu_discharge_ltf = 3200
pmu_discharge_htf = 237

```
pmu_temp_para1          = 7466
pmu_temp_para2          = 4480
pmu_temp_para3          = 3518
pmu_temp_para4          = 2786
pmu_temp_para5          = 2223
pmu_temp_para6          = 1788
pmu_temp_para7          = 1448
pmu_temp_para8          = 969
pmu_temp_para9          = 664
pmu_temp_para10         = 466
pmu_temp_para11         = 393
pmu_temp_para12         = 333
pmu_temp_para13         = 283
pmu_temp_para14         = 242
pmu_temp_para15         = 179
pmu_temp_para16         = 134

[pmu2_para]
pmu_used                 = 0
pmu_twi_addr            = 0x34
pmu_twi_id              = 1
pmu_irq_id              = 0
```

31.2. [pmu2_para]

pmu_used=xx	Pmu 使能标志(xx=1 or 0)， 0：不使用，1：使用
pmu_twi_addr=xx	Pmu 设备地址
pmu_twi_id=xx	Pmu 挂载的 i2c 控制器号， 0：twi0，1：twi1，2：twi2
pmu_irq_id=xx	Pmu 中断号，0：NMI， 1：1 号中断 2：2 号中断……

配置举例：

```
[pmu2_para]
pmu_used                 = 1
pmu_twi_addr            = 0x34
pmu_twi_id              = 1
pmu_irq_id              = 0
```

32. Recovery 键配置

key_min = 3	作为 recovery 功能的按键的键值范围下限

key_max = 5	作为 recovery 功能的按键的键值范围上限

配置举例：
[recovery_key]
key_min = 3
key_max = 5

说明：
通常情况下，一块方案板上的按键个数不同，或者排列不同，这都导致了方案商在选择作为开机阶段 recovery 功能的按键有所不同。该键值配置用于作为 recovery 功能的按键的键值范围落在 key_min 到 key_max 之间。

33. DVFS

33.1. CPU DVFS

配置项	配置项含义
extremity_freq	极限超频模式频率上限
max_freq	最大运行频率
min_freq	最小运行频率
LV_count	VF 表项数
LV1_freq	LV1 对应频率段分界上限，0 表示结束符
LV1_volt	LV1 频率段的电压值，表示(LV2_freq,LV1_freq]范围类电压设置
LV2_freq	LV2 对应频率段分界上限，0 表示结束符
LV2_volt	LV2 频率段的电压值，表示(LV3freq,LV2_freq]范围类电压设置
LV3_freq	LV3 对应频率段分界上限，0 表示结束符
LV3_volt	LV3 频率段的电压值，表示(LV2_freq,LV1_freq]范围类电压设置
LV4_freq	LV4 对应频率段分界上限，0 表示结束符
LV4_volt	LV4 频率段的电压值，表示(LV2_freq,LV1_freq]范围类电压设置
LV5_freq	LV5 对应频率段分界上限，0 表示结束符
LV5_volt	LV5 频率段的电压值，表示(LV2_freq,LV1_freq]范围类电压设置

LV6_freq	LV6 对应频率段分界上限，0 表示结束符
LV6_volt	LV6 频率段的电压值，表示(LV2_freq,LV1_freq]范围类电压设置
LV7_freq	LV7 对应频率段分界上限，0 表示结束符
LV7_volt	LV7 频率段的电压值，表示(LV2_freq,LV1_freq]范围类电压设置
LV8_freq	LV8 对应频率段分界上限，0 表示结束符
LV8_volt	LV8 频率段的电压值，表示(LV2_freq,LV1_freq]范围类电压设置

配置举例：

```
[dvfs_table]
extremity_freq = 1536000000
max_freq = 1200000000
min_freq = 60000000

LV_count = 8

LV1_freq = 1536000000
LV1_volt = 1500

LV2_freq = 1344000000
LV2_volt = 1460

LV3_freq = 1200000000
LV3_volt = 1320

LV4_freq = 1008000000
LV4_volt = 1200

LV5_freq = 816000000
LV5_volt = 1100

LV6_freq = 648000000
LV6_volt = 1040

LV7_freq = 0
LV7_volt = 1040

LV8_freq = 0
```

LV8_volt = 1040

34. Pinctrl 测试

配置项	配置项含义
Vdevice_used	作为 pinctrl test 的虚拟设备，为 1 使能
Vdevice_0	虚拟设备的 gpio0 脚设置
Vdevice_1	虚拟设备的 gpio1 脚设置

配置举例：

```
[Vdevice]
Vdevice_used            = 1
Vdevice_0               = port:PA01<5><1><2><default>
Vdevice_1               = port:PA02<5><1><2><default>
```

35. [s_uart0]

配置项	配置项含义
s_uart_used	使能 cpus 的 uart，为 1 使能，为 0 关闭
s_uart_tx	Uart 口发送引脚配置
s_uart_rx	Uart 接收引脚配置

配置举例：

```
[s_uart0]
s_uart_used         = 1
s_uart_tx           = port:PL00<3><default><default><default>
s_uart_rx           = port:PL01<3><default><default><default>
```

36. [s_rsb0]

配置项	配置项含义
s_rsb_used	使能 cpus 使用 rsb 总线，为 1 使能，为 0 关闭
s_rsb_sck	Rsb 时钟引脚设置
s_rsb_sda	Rsb 数据引脚设置

配置举例：

```
[s_rsb0]
s_uart_used         = 0
```

```
s_uart_tx           = port:PL02<2><default><default><default>
s_uart_rx           = port:PL03<2><default><default><default>
```

37. [s_jtag0]

配置项	配置项含义
s_jtag_used=xx	JTAG 使能
s_jtag_tms=xx	测试模式选择输入(TMS) 的 GPIO 配置
s_jtag_tck=xx	测试时钟输入(TMS) 的 GPIO 配置
s_jtag_tdo=xx	测试数据输出(TDO) 的 GPIO 配置
s_jtag_tdi=xx	测试数据输入（TDI）的 GPIO 配置

配置举例：

```
[s_jtag0]
s_jtag_used         = 0
s_jtag_tms          = port:PL04<2><1><2><default>
s_jtag_tck          = port:PL05<2><1><2><default>
s_jtag_tdo          = port:PL06<2><1><2><default>
s_jtag_tdi          = port:PL07<2><1><2><default>
```

38. [s_powchk]

配置项	配置项含义
s_powchk_used= 0x80000000	是否打开这个功能，如果 bit31 为 1 则使用这个功能，bit0 和 bit1 为 1 分别代表当电源状态和功耗异常时唤醒系统
s_power_reg=0x02309621	电源状态的描述，一般为 1 代表，这里电在休眠时是打开的状态，bit 的具体定义要看 aw_pm.h 里的定义
s_system_power=50	是休眠是允许的最大功耗是 50mW，如果大于此数，代表异常

配置举例：

```
;-----------------------------------------------------------------------------
;s_powchk cpus power check
;s_powchk_used    --power check whether used for arisc in super standby
;    bit31:enable power updat, bit1:wakeup when power state exception
;    bit0:wakeup when power consumption exception
;s_power_reg the expected regs stand for power on/off state
;s_system_power the limit maxmum power consumption when super standby (unit: mw)
;;----------------------------------------------------------------------------
```

```
[s_powchk]
s_powchk_used          = 0x80000000
s_power_reg            = 0x00008061
s_system_power         = 50
```

39. [dram_dvfs_table]

配置项	配置项含义
LV_count = 3	LV 数目
LV1_freq = 552000000	LV1 对应的频率
LV1_volt = 1100	LV1 对应的 core 电压
LV2_freq = 360000000	LV2 对应的频率
LV2_volt = 1000	LV2 对应的 core 电压
LV3_freq = 0	LV3 对应的频率
LV3_volt = 1000	LV3 对应的 core 电压

配置举例：

```
;----------------------------------------------------------------------------------
; dram dvfs voltage-frequency table configuration
;
; LV_count: count of LV_freq/LV_volt
;
; LV1: core vdd is 0.9v if dram frequency is (360Mhz, 672Mhz]
; LV2: core vdd is 0.8v if dram frequency is (    0Mhz, 168Mhz]
; LV3: core vdd is 0.8v if dram frequency is (    0Mhz, 168Mhz]
;----------------------------------------------------------------------------------
[dram_dvfs_table]
LV_count = 3

LV1_freq = 552000000
LV1_volt = 1100

LV2_freq = 360000000
LV2_volt = 1000

LV3_freq = 0
LV3_volt = 1000
```

40. [dram_scene_table]

配置项	配置项含义
LV_count = 3	不同应用场景下 dram 频率等级数目
LV1_scene = 1	场景 1（home）对应的 index
LV1_freq = 360000000	场景 1 对应的 dram 频率。支持 360MHz/dram_clk
LV2_scene = 2	场景 2（video play）对应的 index
LV2_freq = 240000000	场景 2 对应的 dram 频率。支持 240MHz/360MHz/dram_clk
LV3_scene = 3	场景 3（黑屏音乐）对应的 index
LV3_freq = 168000000	场景 3 对应的 dram 频率。支持 168MHz/240MHz/360MHz/dram_clk

配置举例：

```
;-------------------------------------------------------------------------------
; dram scene frequency table configuration
;
; LV_count: count of LV_scene/LV_freq
;
; LV1: dram frequency default is 360MHz in home, supported for 360MHz/552MHz
; LV2: dram frequency default is 240MHz in video play, supported for 240MHz/360MHz/552MHz
; LV3: dram frequency default is 168MHz in bgmusic play
;
;-------------------------------------------------------------------------------
[dram_scene_table]
LV_count = 3

LV1_scene = 1
LV1_freq    = 360000000

LV2_scene = 2
LV2_freq    = 240000000

LV3_scene = 3
LV3_freq    = 168000000
```

41. [charging_type]

配置项	配置项含义
charging_type = 1	为 1，才能进入安卓关机充电模式；为 0 是进入 boot standby 进行充电。

配置举例：
[charging_type]
charging_type = 1

注：如果想进入安卓关机充电功能，应该务必加上以上内容。

42. Declaration