全 志 科 技
Allwinner Technology

A80 sys_config.fex 配置手册

V1.0

2014-07-25

Revision History

Version	Date	Changes compared to previous issue
1.0	2014-07-25	

目录

1. 概述

1.1. 编写目的

介绍 A80 平台系统配置方法。

1.2. 使用范围

A80 开发板各版本 sdk。

1.3. 目标读者

工程开发人员。

备注：蓝色字体为模块芯片引脚配置，黑色字体为模块内部控制配置项。
描述 gpio 的 GPIO 配置的形式：
Port:端口+组内序号<功能分配><内部电阻状态><驱动能力><输出电平状态>
配置举例中的管脚不一定为真实可用的，实际使用时需向技术支持人员询问。

2. 系统(System)

2.1. [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）

配置举例：

[platform]

eraseflag = 1

2.2. [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
boot_clock_c1=xx	A15的启动频率; 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

boot_clock_c1 = 1008

storage_type = -1

2.3. [power_sply](axp809)

配置项	配置项含义
dcdc1_vol = 1003000	Dcdc1(IO，PA/PH/PD/NAND/USB0)的输出电压 xx mV
dcdc2_vol = 1000900	Dcdc2(GPU)的输出电压 xx mV
dcdc3_vol = 1000900	Dcdc3(CPUA)的输出电压 xx mV,
dcdc4_vol = 1000900	Dcdc4(IO)的输出电压 xx mV
dc5ldo_vol = 1000900	Dc5ldo(cpus)的输出电压 xx mV
dldo2_vol = 1001800	Dldo2(IO，PL)的输出电压 xx mV
eldo3_vol = 1001800	Eldo3(IO, PM)的输出电压 xx mV

aldo1_vol = 1003000	Aldo1(USBH)的输出电压 xx mV
aldo3_vol = 1003000	Aldo3(CARD0)电压

注：电压名称 = 100XXXX ：表示把该路电压设置为 XXXX 指定的电压值，同时打开输出开关

电压名称 = 000XXXX ：表示把该路电压设置为 XXXX 指定的电压值，同时关闭输出开关，当有需要时由内核驱动打开

电压名称 = 0 ：表示关闭该路电压输出开关，不修改原有的值

配置举例：

```
[power_sply]
dcdc1_vol                = 1003000
dcdc2_vol                = 1000900
dcdc3_vol                = 1000900
dcdc4_vol                = 1000900
dc5ldo_vol               = 1000900
dldo2_vol                = 1001800
eldo3_vol                = 1001800
aldo1_vol                = 1003000
aldo3_vol                = 1003000
```

2.4. [slave_power_sply](axp806)

配置项	配置项含义
dcdca_vol=1000900	Dcdca(CPUB)的输出电压 xx mv
dcdcb_vol = 1000900	Dcdcb(CPUB)的输出电压 xx mV
dcdcc_vol = 1000900	Dcdcc(CPUB)的输出电压 xx mV,
dcdcd_vol = 1000900	Dcdcd(VPU)的输出电压 xx mV
dcdce_vol = 1002100	Dcdce(BLDOIN/LDOIN)的输出电压 xx mV
Aldo1_vol = 1003000	Aldo1(AVCC) xx mV
Bldo1_vol = 1001800	Bldo1(EFUSE/HDMI/EDP/NAND/SSP/CSI/DSI)的输出电压 xx mV

Bldo2_vol = 1001800	Bldo2(DRAMPLL/LPDDR/PLL/CPVDD)的输出电压 xx mV

注：电压名称 = 100XXXX ：表示把该路电压设置为 XXXX 指定的电压值，同时打开输出开关

电压名称 = 000XXXX ：表示把该路电压设置为 XXXX 指定的电压值，同时关闭输出开关，当有需要时由内核驱动打开

电压名称 = 0 ：表示关闭该路电压输出开关，不修改原有的值

配置举例：

```
[slave_power_sply]
dcdca_vol                = 1000900
dcdcb_vol                = 1000900
dcdcc_vol                = 1000900
dcdcd_vol                = 1000900
dcdce_vol                = 1002100
aldo1_vol                = 1003000
bldo1_vol                = 1001800
bldo2_vol                = 1001800
```

2.5. [gpio_bias]

配置项	配置项含义
pa_bias = xx	GPIO_A 的供应电压设置
pb_bias = xx	GPIO_B 的供应电压设置
pc_bias = xx	GPIO_C 的供应电压设置
pd_bias =xx	GPIO_D 的供应电压设置
Pe_bias =xx	GPIO_E 的供应电压设置
Pf_bias = xx	GPIO_F 的供应电压设置
Pg_bias = xx	GPIO_G 的供应电压设置
Ph_bias = xx	GPIO_H 的供应电压设置
Pl_bias =xx	GPIO_L 的供应电压设置
Pm_bias = xx	GPIO_M 的供应电压设置

配置项举例：

```
[gpio_bias]
pa_bias          = "axp809:dcdc1:3000"
pb_bias          = "axp806:aldo2:2800"
```

pc_bias = "axp809:dcdc1:3000"
pd_bias = "axp809:dcdc1:3000"
pe_bias = "axp809:eldo2:2800"
pf_bias = "axp809:aldo3:3000"
pg_bias = "axp806:cldo1:3000"
ph_bias = "axp809:dcdc1:3000"
pl_bias = "axp809:dldo2:3000"
pm_bias = "axp809:eldo3:1800"

注：pa_bias = "axp809:dcdc1:3000" ，表示的意思为 GPIO_A 的供应电压由 axp809 供应，供应电路为 dcdc1，供应电压为 3000mv

2.6. [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：

```
;-------------------------------------------------------------------------------------------
; if 1 == standby_mode, then support super standby;else, support normal standby.
;-------------------------------------------------------------------------------------------
[pm_para]
standby_mode = 1
```

2.7. [card_boot]

配置项	配置项含义
Logical_start=xx	
Sprite_gpio0=	

配置举例：

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

2.8. [card_boot0_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：

```
card_ctrl           = 0
card_high_speed         = 1
card_line           = 4
sdc_d1                  = port:PF00<2><1><default><default>
sdc_d0                  = port:PF01<2><1><default><default>
sdc_clk                 = port:PF02<2><1><default><default>
sdc_cmd             = port:PF03<2><1><default><default>
sdc_d3                  = port:PF04<2><1><default><default>
sdc_d2                  = port:PF05<2><1><default><default>
```

2.9. [card_boot2_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：

```
card_ctrl           = 2
card_high_speed         = 1
card_line           = 4
sdc_cmd             = port:PC06<3><1><default><default>
sdc_clk                 = port:PC07<3><1><default><default>
sdc_d0                  = port:PC08<3><1><default><default>
sdc_d1                  = port:PC09<3><1><default><default>
sdc_d2                  = port:PC10<3><1><default><default>
sdc_d3                  = port:PC11<3><1><default><default>
```


2.10. [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_port = 0
twi_scl = port:PH14<2><default><default><default>
twi_sda = port:PH15<2><default><default><default>

2.11. [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_debug_port = 0
uart_debug_tx = port:PH12<2><1><default><default>
uart_debug_rx = port:PH13<2><1><default><default>

2.12. [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>

2.13. [clock]

配置项	配置项含义
Pll4 =300	Peripherals 时钟频率
Pll6 =600	DDR 时钟频率
Pll7 =1188	Video0 时钟频率
Pll9 =297	GPU（运算）时钟频率

配置举例：

[clock]

pll4 = 300

pll6 = 600

pll7 = 1188

pll9 = 297

3. SDRAM

3.1. [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

;**********

;sdram configuration

;

;dram_para2 = 0x00001200 ;代表启用 dram 双通道

;

;dram_para2 = 0x00001100 ;代表启用 dram 单通道

;

;**********

配置举例:

[dram_para]

dram_clk = 552
dram_type = 3
dram_zq = 0x007B7BbB
dram_odt_en = 0
dram_para1 = 0x10f41000
dram_para2 = 0x00001100
dram_mr0 = 0x1A50
dram_mr1 = 0x4
dram_mr2 = 0x10
dram_mr3 = 0
dram_tpr0 = 0x04E214EA
dram_tpr1 = 0x004214AD
dram_tpr2 = 0x10A75030
dram_tpr3 = 0
dram_tpr4 = 0
dram_tpr5 = 0
dram_tpr6 = 0
dram_tpr7 = 0
dram_tpr8 = 0
dram_tpr9 = 0
dram_tpr10 = 0
dram_tpr11 = 0
dram_tpr12 = 150
dram_tpr13 = 0x23

4. GMAC

4.1. [gmac_para]

配置项	配置项含义
gmac_used=0	Gmac 模块是否使能：1：enable0：disable
gmac_txd0=xx	Gmac tx0 的 GPIO 配置
gmac_txd1=xx	Gmac tx1 的 GPIO 配置
gmac_txd2=xx	Gmac tx2 的 GPIO 配置
gmac_txd3=xx	Gmac tx3 的 GPIO 配置
gmac_txd4=xx	Gmac tx4 的 GPIO 配置
gmac_txd5=xx	Gmac tx5 的 GPIO 配置
gmac_txd6=xx	Gmac tx6 的 GPIO 配置
gmac_txd7=xx	Gmac tx7 的 GPIO 配置
gmac_txclk=xx	Gmac MII 接口发送时钟

gmac_txen=xx	Gmac 发送使能 GPIO 配置
gmac_gtxclk=xx	Gmac GMII 接口发送时钟
gmac_rxd0=xx	Gmac rx0 的 GPIO 配置
gmac_rxd1=xx	Gmac rx1 的 GPIO 配置
gmac_rxd2=xx	Gmac rx2 的 GPIO 配置
gmac_rxd3=xx	Gmac rx3 的 GPIO 配置
gmac_rxd4=xx	Gmac rx4 的 GPIO 配置
gmac_rxd5=xx	Gmac rx5 的 GPIO 配置
gmac_rxd6=xx	Gmac rx6 的 GPIO 配置
gmac_rxd7=xx	Gmac rx7 的 GPIO 配置
gmac_rxdv=xx	Gmac 接收数有效使能
gmac_rxclk=xx	Gmac 接收时钟
gmac_txerr=xx	Gmac 发送错误使能
gmac_rxerr=xx	Gmac 接收错误使能
gmac_col=xx	Gmac 冲突检测(仅用于半双工)
gmac_crs=xx	Gmac 载波监测(仅用于半双工)
gmac_clkin=xx	Gmac GMII 外部时钟
gmac_mdc=xx	Gmac 配置接口时钟
gmac_mdio=xx	Gmac 配置接口数据 I/O

```
[gmac_para]
gmac_used       = 0
gmac_txd0       = port:PA00<2><default><3><default>
gmac_txd1       = port:PA01<2><default><3><default>
gmac_txd2       = port:PA02<2><default><3><default>
gmac_txd3       = port:PA03<2><default><3><default>
gmac_txd4       = port:PA04<2><default><3><default>
gmac_txd5       = port:PA05<2><default><3><default>
gmac_txd6       = port:PA06<2><default><3><default>
gmac_txd7       = port:PA07<2><default><3><default>
gmac_txclk      = port:PA08<2><default><3><default>
gmac_txen       = port:PA09<2><default><3><default>
gmac_gtxclk     = port:PA10<2><default><3><default>
gmac_rxd0       = port:PA11<2><default><3><default>
gmac_rxd1       = port:PA12<2><default><3><default>
gmac_rxd2       = port:PA13<2><default><3><default>
gmac_rxd3       = port:PA14<2><default><3><default>
gmac_rxd4       = port:PA15<2><default><3><default>
gmac_rxd5       = port:PA16<2><default><3><default>
gmac_rxd6       = port:PA17<2><default><3><default>
gmac_rxd7       = port:PA18<2><default><3><default>
gmac_rxdv       = port:PA19<2><default><3><default>
gmac_rxclk      = port:PA20<2><default><3><default>
```


gmac_txerr = port:PA21<2><default><3><default>
gmac_rxerr = port:PA22<2><default><3><default>
gmac_col = port:PA23<2><default><3><default>
gmac_crs = port:PA24<2><default><3><default>
gmac_clkin = port:PA25<2><default><3><default>
gmac_mdc = port:PA26<2><default><3><default>
gmac_mdio = port:PA27<2><default><3><default>

5. I2C 总线

主控有 4 个 I2C（twi）控制器

5.1. [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_used = 1
twi0_scl = port:PH00<2><default><default><default>
twi0_sda = port:PH01<2><default><default><default>

5.2. [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]
twi1_used = 1
twi1_scl = port:PH02<2><default><default><default>
twi1_sda = port:PH03<2><default><default><default>

5.3. [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]
twi2_used                = 0
twi2_scl                 = port:PH04<2><default><default><default>
twi2_sda                 = port:PH05<2><default><default><default>
```

5.4. [twi3_para]

配置项	配置项含义
twi3_used =xx	TWI 使用控制：1 使用，0 不用
twi3_scl =xx	TWI SCK 的 GPIO 配置
twi3_sda=xx	TWI SDA 的 GPIO 配置

配置举例：

```
[twi2_para]
twi2_used                = 1
twi2_scl                 = port:PG10<2><default><default><default>
twi2_sda                 = port:PG11<2><default><default><default>
```

5.5. [twi4_para]

配置项	配置项含义
Twi4_used =xx	TWI 使用控制：1 使用，0 不用
Twi4_scl =xx	TWI SCK 的 GPIO 配置
Twi4_sda=xx	TWI SDA 的 GPIO 配置

配置举例：

```
[twi4]
twi_used          = 0
twi_scl           = port:PB15<4><default><default><default>
twi_sda           = port:PB16<4><default><default><default>
```

6. 串口(UART)

主控有 6 路 uart 接口，其中 uart1 支持完整的 8 线通讯，而其他 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

6.1. [uart_para0]

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
uart0_tx =xx	UART TX 的 GPIO 配置
uart0_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:PH12<2><1><default><default>
uart_rx = port:PH13<2><1><default><default>

6.2. [uart_para1]

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
uart1_tx =xx	UART TX 的 GPIO 配置
uart1_rx＝xx	UART RX 的 GPIO 配置
uart1_rts=xx	UART RTS 的 GPIO 配置
uart1_cts=xx	UART CTS 的 GPIO 配置
uart1_dtr=xx	UART DTR 的 GPIO 配置
uart1_dsr=xx	UART DSR 的 GPIO 配置
uart1_dcd=xx	UART DCD 的 GPIO 配置
uart1_ring=xx	UART RING 的 GPIO 配置

配置举例：

[uart1]
uart_used = 0
uart_port = 1
uart_type = 8
uart_tx = port:PA0<4><1><default><default>
uart_rx = port:PA1<4><1><default><default>
uart_rts = port:PA2<4><1><default><default>

uart_cts = port:PA3<4><1><default><default>
uart_dtr = port:PA4<4><1><default><default>
uart_dsr = port:PA5<4><1><default><default>
uart_dcd = port:PA6<4><1><default><default>
uart_ring = port:PA7<4><1><default><default>

6.3. [uart_para2]

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
uart2_tx =xx	UART TX 的 GPIO 配置
uart2_rx＝xx	UART RX 的 GPIO 配置
uart2_rts=xx	UART RTS 的 GPIO 配置
uart2_cts=xx	UART CTS 的 GPIO 配置

配置举例：

[uart_para2]
uart_used = 0
uart_port = 2
uart_type = 4
uart2_tx = port:PG06<2><1><default><default>
uart2_rx = port:PH07<2><1><default><default>
uart2_rts = port:PH08<2><1><default><default>
uart2_cts = port:PH09<2><1><default><default>

6.4. [uart_para3]

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
uart3_tx =xx	UART TX 的 GPIO 配置
uart3_rx＝xx	UART RX 的 GPIO 配置
uart3_rts=xx	UART RTS 的 GPIO 配置
uart3_cts=xx	UART CTS 的 GPIO 配置

配置举例：

[uart3]
uart_used = 0
uart_port = 3
uart_type = 4
uart_tx = port:PB05<3><1><default><default>

uart_rx = port:PB06<3><1><default><default>
uart_rts = port:PB04<3><1><default><default>
uart_cts = port:PB00<3><1><default><default>

6.5. [uart_para4]

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：
[uart_para4]
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>

6.6. [uart_para5]

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
uart5_tx =xx	UART TX 的 GPIO 配置
uart5_rx＝xx	UART RX 的 GPIO 配置

配置举例：
[uart5]
uart_used = 0
uart_port = 5
uart_type = 4
uart_tx = port:PE04<4><1><default><default>
uart_rx = port:PE05<4><1><default><default>
uart_rts = port:PE06<4><1><default><default>
uart_cts = port:PE07<4><1><default><default>

7. SPI 总线

7.1. [spi0_para]

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            = 1
spi_cs_bitmap    = 1
spi_cs0             = port:PC19<3><1><default><default>
spi_sclk            = port:PC02<3><default><default><default>
spi_mosi            = port:PC00<3><default><default><default>
spi_miso            = port:PC01<3><default><default><default>
```

7.2. [spi1_para]

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:PA14<4><1><default><default>
spi_sclk            = port:PA15<4><default><default><default>
spi_mosi            = port:PA16<4><default><default><default>
```

spi_miso = port:PA17<4><default><default><default>

7.3. [spi2_para]

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：

[spi2]
spi_used = 0
spi_cs_bitmap = 1
spi_cs0 = port:PE04<3><1><default><default>
spi_sclk = port:PE05<3><default><default><default>
spi_mosi = port:PE06<3><default><default><default>
spi_miso = port:PE07<3><default><default><default>

7.4. [spi3_para]

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	SPI CS0 的 GPIO 配置
spi_cs0 =xx	SPI CS0 的 GPIO 配置
spi_cs1 =xx	SPI CS1 的 GPIO 配置
spi_cs2＝xx	SPI CS2 的 GPIO 配置
spi_cs3=xx	SPI CS3 的 GPIO 配置
spi_sclk	SPI CLK 的 GPIO 配置
spi_mosi	SPI MOSI 的 GPIO 配置
spi_miso	SPI MISO 的 GPIO 配置

配置举例：

[spi3]
spi_used = 0
spi_cs_bitmap = 1
spi_cs0 = port:PH17<2><1><default><default>
spi_cs1 = port:PH18<2><1><default><default>
spi_cs2 = port:PH12<3><1><default><default>

spi_cs3 = port:PH13<3><default><default><default>
spi_sclk = port:PH14<2><default><default><default>
spi_mosi = port:PH15<2><default><default><default>
spi_miso = port:PH16<2><default><default><default>

7.5. [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

7.6. [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

8. 电阻屏(rtp)

8.1. [rtp_para]

配置项	配置项含义
rtp_used=xx	该模块在方案中是否启用，
rtp_screen_size =xx	屏幕尺寸设置，以斜对角方向长度为准，以寸为单位
rtp_regidity_level=xx	表屏幕的硬度，以指覆按压，抬起时开始计时，多少个 10ms 时间单位之后，硬件采集不到数据为准；通常，我们建议的屏，5 寸屏设为 5,7 寸屏设为 7，对于某些供应商提供的屏，硬度可能不合要求，需要适度调整
rtp_press_threshold_enable=xx	是否开启压力的们门限制，建议选 0 不开启
rtp_press_threshold＝xx	这配置项当 rtp_press_threshold_enable 为 1 时才有效，其数值可以是 0 到 0xFFFFFF 的任意数值，数值越小越敏感，推荐值为 0xF

rtp_sensitive_level=xx	敏感等级，数值可以是 0 到 0xF 之间的任意数值，数值越大越敏感，0xF 为推荐值
rtp_exchange_x_y_flag=xx	当屏的 x,y 轴需要转换的时候，这个项目该置 1，一般情况下则该置 0

9. 电容屏(capacitor tp)

9.1. [ctp_para]

配置项	配置项含义
ctp_used=xx	该选项为是否开启电容触摸，支持的话置 1，反之置 0
ctp_twi_id=xx	用于选择 i2c adapter，可选 1，2
ctp_twi_addr =xx	指明 i2c 设备地址，与具体硬件相关
ctp_screen_max_x＝xx	触摸板的 x 轴最大坐标
ctp_screen_max_y=xx	触摸板的 y 轴最大坐标
ctp_revert_x_flag=xx	是否需要翻转 x 坐标，需要则置 1，反之置 0
ctp_revert_y_flag=xx	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=xx	电容屏中断信号的 GPIO 配置
ctp_wakeup=xx	电容屏唤醒信号的 GPIO 配置

配置举例：

```
[ctp_para]
ctp_used          = 1
ctp_twi_id              = 0
ctp_twi_addr      = 0x5d
ctp_screen_max_x  = 1280
ctp_screen_max_y  = 800
ctp_revert_x_flag     = 1
ctp_revert_y_flag     = 1
ctp_exchange_x_y_flag = 1
ctp_int_port              = port:PA12<6><default><default><default>
ctp_wakeup                = port:PA11<1><default><default><1>
```


10. 触摸按键(touch key)

10.1. [tkey_para]

配置项	配置项含义
tkey_used =xx	支持触摸按键的置 1，反之置 0
tkey_twi_id=xx	用于选择 i2c adapter, 可选 1，2
tkey_twi_addr=xx	指明 i2c 设备地址，与具体硬件相关
tkey_int＝xx	触摸按键中断信号的 GPIO 配置

配置举例：

tkey_used = 0

tkey_twi_id =

tkey_twi_addr =

tkey_int =

注意事项：

若支持，则将 tkey_used 置 1 并配置相应子键值；否则，tkey_used 置 0；

11. 马达(motor)

11.1. [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_used = 1

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

motor_ldo = axp22_ldoio0

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。

12. 闪存（nand0 flash）

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        = 1
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_ce2                   = port:PC17<2><default><default><default>
nand0_ce3                   = port:PC18<2><default><default><default>
nand0_ndqs                  = port:PC16<2><default><default><default>
```

12.2. [nand1_para]

配置项	配置项含义
nand1_used =xx	nand1 模块使能标志
nand1_we =xx	nand1 写时钟信号的 GPIO 配置
nand1_ale =xx	nand1 地址使能信号的 GPIO 配置
nand1_cle =xx	nand1 命令使能信号的 GPIO 配置
nand1_ce1 =xx	nand1 片选 1 信号的 GPIO 配置
nand1_ce0 =xx	nand1 片选 0 信号的 GPIO 配置
nand1_nre =xx	nand1 读时钟信号的 GPIO 配置
nand1_rb0=xx	nand1 Read/Busy 1 信号的 GPIO 配置
nand1_rb1 =xx	nand1 Read/Busy 0 信号的 GPIO 配置
nand1_d0=xx	nand1 数据总线信号的 GPIO 配置
nand1_d1=xx	/
nand1_d2=xx	/
nand1_d3=xx	/
nand1_d4=xx	/
nand1_d5=xx	/
nand1_d6=xx	/
nand1_d7=xx	/
nand1_ce2=xx	nand1 片选 2 信号的 GPIO 配置
nand1_ce3=xx	nand1 片选 3 信号的 GPIO 配置
nand1_ndqs=xx	nand0 ddr 时钟信号的 GPIO 配置

配置举例：

```
[nand1_para]
nand1_used              = 1
nand1_we                 = port:PC00<2><default><default><default>
nand1_ale               = port:PC01<2><default><default><default>
nand1_cle               = port:PC03<2><default><default><default>
nand1_ce1                = port:PC03<2><default><default><default>
nand1_ce0                = port:PC04<2><default><default><default>
nand1_nre               = port:PC05<2><default><default><default>
```



```
nand1_rb0           = port:PC06<2><default><default><default>
nand1_rb1           = port:PC07<2><default><default><default>
nand1_d0            = port:PC08<2><default><default><default>
nand1_d1            = port:PC09<2><default><default><default>
nand1_d2            = port:PC10<2><default><default><default>
nand1_d3            = port:PC11<2><default><default><default>
nand1_d4            = port:PC12<2><default><default><default>
nand1_d5            = port:PC13<2><default><default><default>
nand1_d6            = port:PC14<2><default><default><default>
nand1_d7            = port:PC15<2><default><default><default>
nand1_ce2           = port:PC17<2><default><default><default>
nand1_ce3           = port:PC18<2><default><default><default>
nand1_ndqs          = port:PC16<2><default><default><default>
```

13. 显示初始化(disp init)

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 = 197
lcd1_backlight = 197

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     = "default_lcd"
lcd_if              = 3
lcd_x               = 1280
lcd_y               = 800
```



```
lcd_width               = 150
lcd_height              = 94
lcd_dclk_freq           = 70
lcd_pwm_used            = 1
lcd_pwm_ch              = 0
lcd_pwm_freq            = 50000
lcd_pwm_pol             = 1
lcd_hbp                 = 20
lcd_ht                  = 1418
lcd_hspw                = 10
lcd_vbp                 = 10
lcd_vt                  = 814
lcd_vspw                = 5
lcd_lvds_if             = 0
lcd_lvds_colordepth = 1
lcd_lvds_mode           = 0
lcd_frm                 = 1
lcd_io_phase            = 0x0000
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:PA6<1><0><default><1>
lcd_power               = port:power2<1><0><default><1>

lcdd0                   = port:PD00<3><0><default><default>
lcdd1                   = port:PD01<3><0><default><default>
lcdd2                   = port:PD02<3><0><default><default>
lcdd3                   = port:PD03<3><0><default><default>
lcdd4                   = port:PD04<3><0><default><default>
lcdd5                   = port:PD05<3><0><default><default>
lcdd6                   = port:PD06<3><0><default><default>
lcdd7                   = port:PD07<3><0><default><default>
```


15. LCD 屏 1

15.1. [lcd1_para]

所有配置跟 lcd0 一样

16. HDMI

16.1. [hdmi_para]

配置项	配置项含义
para_used =xx	是否使用 hdmi

17. 摄像头(CSI)

A80 CSI 的 sysconfig 部分对应的字段为：[csi0]和[csi1]。

下面举例说明在实际使用中应该如何配置：

1. **使用一个并口 camera 模组：**需要配置[csi1]的公用部分和[csi1]的 vip_dev0_(x)部分，需要注意的是[csi1]公用部分中的 vip_dev_qty 需要设置为 1，另外[csi0]中 vip_used 设置为 0，[csi1]中 vip_used 设置为 1。

2. **使用两个并口的 camera 模组：**同样需要配置[csi1]公用部分，另外需要配置[csi1]的 vip_dev0_(x)部分和[csi1]的 vip_dev1_(x)部分，[csi1]公用部分中的 vip_dev_qty 需要设置为 2，[csi0]中 vip_used 设置为 0，[csi1]中 vip_used 设置为 1。

3. **使用一个 MIPI camera 模组：**需要配置[csi0]的公用部分和 vip_dev0_(x)部分，[csi0]公用部分中 vip_dev_qty 设置为 1。[csi1]中 vip_used 设置为 0，[csi0]中 vip_used 设置为 1。

4. **使用一个 MIPI camera 模组和一个并口 camera 模组：**需要配置[csi0]的公用部分和[csi1]的公用部分，以及两者的 vip_dev0_(x)部分，[csi0]和[csi1]中 vip_dev_qty 都设置为 1。[csi0]和[csi1]中 vip_used 都设置为 1。

17.1. [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                = 0
vip_mode                = 0
vip_dev_qty             = 1
vip_define_sensor_list  = 1

vip_csi_mck             = port:PB14<3><default><default><default>

vip_dev0_mname          = "ov8825"
vip_dev0_pos            = "rear"
```

```
vip_dev0_lane           = 1
vip_dev0_twi_id         = 0
vip_dev0_twi_addr       = 0x78
vip_dev0_isp_used       = 1
vip_dev0_fmt            = 1
vip_dev0_stby_mode      = 0
vip_dev0_vflip          = 0
vip_dev0_hflip          = 0
vip_dev0_iovdd          = "axp22_aldo2"
vip_dev0_iovdd_vol      = 2800000
vip_dev0_avdd           = "axp15_aldo2"
vip_dev0_avdd_vol       = 2800000
vip_dev0_dvdd           = "axp22_eldo1"
vip_dev0_dvdd_vol       = 1500000
vip_dev0_afvdd          = ""
vip_dev0_afvdd_vol      = 2800000
vip_dev0_power_en       =
vip_dev0_reset          = port:PB5<1><default><default><default>
vip_dev0_pwdn           = port:PB6<1><default><default><default>
vip_dev0_flash_en       =
vip_dev0_flash_mode     =
vip_dev0_af_pwdn        =

vip_dev0_act_used       = 0
vip_dev0_act_name       = "ov8825_act"
vip_dev0_act_slave      =


vip_dev1_mname          = ""
vip_dev1_pos            = "front"
vip_dev1_lane           = 1
vip_dev1_twi_id         = 0
vip_dev1_twi_addr       =
vip_dev1_isp_used       = 0
vip_dev1_fmt            = 1
vip_dev1_stby_mode      = 0
vip_dev1_vflip          = 0
vip_dev1_hflip          = 0
vip_dev1_iovdd          = "axp22_eldo3"
vip_dev1_iovdd_vol      = 2800000
vip_dev1_avdd           = "axp22_dldo4"
vip_dev1_avdd_vol       = 2800000
```



```
vip_dev1_dvdd                = "axp22_eldo2"
vip_dev1_dvdd_vol            = 1500000
vip_dev1_afvdd               = ""
vip_dev1_afvdd_vol           = 2800000
vip_dev1_power_en            =
vip_dev1_reset               =
vip_dev1_pwdn                =
vip_dev1_flash_en            =
vip_dev1_flash_mode          =
vip_dev1_af_pwdn             =
```

17.2. [csi1_para]

配置项	配置项含义
vip_used	是否使用 csi1。
vip_mode	一般填 0。
vip_dev_qty	支持复用，如果有两个并口 camera 模组，填 2，如果只有一个并口模组填 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_pck	并口 pclk 信号 GPIO 配置。
vip_csi_mck	并口 mclk 信号 GPIO 配置。
vip_csi_hsync	Hsync 信号 GPIO 配置。
vip_csi_vsync	Vsync 信号 GPIO 配置。
vip_csi_d0=xx … vip_csi_d11=xx	并口 csi 数据信号 GPIO 配置。支持 12 位 sensor。vip_csi_d11 为高位，vip_csi_d0 为低位。如果只有 8 位则只配置 vip_csi_d4~vip_csi_d11。
vip_csi_sck	CSI1 CCI 时钟信号的 GPIO 配置。 如果使用 CSI1 内部 CCI 需要配置该项。
vip_csi_sda	CSI1 CCI 时钟信号的 GPIO 配置。 如果使用 CSI1 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。

配置举例：

```
[csi1_para]
csi_used            = 1
csi_mode            = 0
csi_dev_qty         = 2
csi_stby_mode       = 0

csi_mname           = "ov5640"
csi_twi_id          = 0
csi_twi_addr        = 0x78
csi_if              = 0
csi_vflip           = 0
csi_hflip           = 1
csi_iovdd           = "axp22_eldo3"
csi_avdd            = "axp22_dldo4"
csi_dvdd            = "axp22_eldo2"
csi_vol_iovdd       = 2800
csi_vol_avdd        = 2800
csi_vol_dvdd        = 1800
csi_flash_pol       = 1

csi_mname_b         = "gc0307"
csi_twi_id_b        = 0
csi_twi_addr_b      = 0x42
csi_if_b            = 0
csi_vflip_b         = 1
csi_hflip_b         = 1
csi_iovdd_b         = "axp22_eldo3"
csi_avdd_b          = "axp22_dldo4"
csi_dvdd_b          = "axp22_eldo2"
csi_vol_iovdd_b     = 2800
csi_vol_avdd_b      = 2800
csi_vol_dvdd_b      = 1800
csi_flash_pol_b     = 1
```


csi_pck = port:PE00<2><default><default><default>
csi_mck = port:PE01<2><default><default><default>
csi_hsync = port:PE02<2><default><default><default>
csi_vsync = port:PE03<2><default><default><default>
csi_d0 =
csi_d1 =
csi_d2 =
csi_d3 =
csi_d4 = port:PE08<2><default><default><default>
csi_d5 = port:PE09<2><default><default><default>
csi_d6 = port:PE10<2><default><default><default>
csi_d7 = port:PE11<2><default><default><default>
csi_d8 = port:PE12<2><default><default><default>
csi_d9 = port:PE13<2><default><default><default>
csi_d10 = port:PE14<2><default><default><default>
csi_d11 = port:PE15<2><default><default><default>
csi_reset = port:PE04<1><default><default><0>
csi_power_en =
csi_stby = port:PE05<1><default><default><1>
csi_flash =
csi_af_en =
csi_reset_b = port:PE06<1><default><default><0>
csi_power_en_b =
csi_stby_b = port:PE07<1><default><default><1>
csi_flash_b =
csi_af_en_b =

18. SD / MMC

18.1. [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 检测，2-data3 检测，3-无检测，卡常在(不卡拔插)，4 - manual mode(from proc file system node)
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_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：

```
[mmc0_para]
sdc_used            = 1
sdc_detmode         = 2
sdc_buswidth        = 4
sdc_clk             = port:PF02<2><1><2><default>
sdc_cmd             = port:PF03<2><1><2><default>
sdc_d0              = port:PF01<2><1><2><default>
sdc_d1              = port:PF00<2><1><2><default>
sdc_d2              = port:PF05<2><1><2><default>
sdc_d3              = port:PF04<2><1><2><default>
sdc_det             = port:PA10<6><1><2><default>
sdc_use_wp          = 0
sdc_wp              =
sdc_isio            = 0
sdc_regulator       = "none"
```

18.2. [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 检测，2-data3 检测，3-无检测，卡常在(不卡拔插)，4 - manual mode(from proc file system node)
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"

配置举例：

```
[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"
```

18.3. [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 检测，2-data3 检测，3-无检测，卡常在(不卡拔插)，4 - manual mode(from proc file system node)
Sdc_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_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
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            = 1
sdc_detmode         = 3
sdc_buswidth        = 8
sdc_clk             = port:PC07<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_det             =
sdc_use_wp          = 0
sdc_wp              =
sdc_isio            = 0
sdc_regulator       = "none"
```

18.4. [mmc3_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 检测，2-data3 检测，3-无检测，卡常在(不卡拔插)，4 - manual mode(from proc file system node)
Sdc_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_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
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：

```
[mmc3_para]
sdc_used            = 1
sdc_detmode         = 4
sdc_buswidth        = 8
sdc_clk             = port:PC07<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_det             =
sdc_use_wp          = 0
sdc_wp              =
sdc_isio            = 0
sdc_regulator       = "none"
```

19. SIM 卡

19.1. [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：

20. USB 控制标志

20.1. [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:PA15<0><1><default><default>
usb_det_vbus_gpio　　= "axp_ctrl"
usb_drv_vbus_gpio　　= port:power4<1><0><default><1>
usb_restrict_gpio　　=
usb_host_init_state = 0
usb_restric_flag　　= 0
usb_restric_voltage = 3550000
usb_restric_capacity= 5

20.2. [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_drv_vbus_gpio = port:PA14<1><0><default><0>

usb_restrict_gpio = port:PA13<1><0><default><0>

usb_host_init_state = 1

usb_restric_flag = 0

20.3. [usbc2]

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_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 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：

[usbc2]

usb_used = 1

usb_port_type = 1

usb_detect_type = 0

usb_drv_vbus_gpio =

usb_restrict_gpio =

usb_host_init_state = 1

usb_restric_flag = 0

20.4. [usbc3]

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_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 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：

```
[usbc3]
usb_used                = 1
usb_port_type           = 1
usb_detect_type         = 0
usb_drv_vbus_gpio       =
usb_restrict_gpio       =
usb_host_init_state = 1
usb_restric_flag        = 0
```

21. USB Device

21.1. [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                = 0x18D1
mass_storage_id          = 0x0001
adb_id                   = 0x0002
manufacturer_name        = "USB Developer"
product_name             = "Android"
serial_number            = "20080411"
```

21.2. [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
```

22. 重力感应(G Sensor)

22.1. [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          = 2
gsensor_twi_addr        = 0x18
gsensor_int1            = port:PH17<6><1><default><default>
gsensor_int2            =
```

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。

```
[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-bcm40181;　2-bcm40183;

	3-rtl8723as; 4-rt l8189es; 5 - rtl8192cu; 6 - rtl8188eu; 7 – rtl8723au;
wifi_power=xx	给模组供电的 axp 引脚名

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 rtl8723as 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"

; 3 - rtl8723as sdio wifi + bt gpio config
rtk_rtl8723as_wl_dis          = port:PG10<1><default><default><0>
rtk_rtl8723as_bt_dis          = port:PG11<1><default><default><0>
rtk_rtl8723as_wl_host_wake = port:PG12<0><default><default><0>
rtk_rtl8723as_bt_host_wake  = port:PG17<0><default><default><0>
```

以上配置意思是要使用序号为 3 的 SDIO 接口 rtl8723as 模组，选用 SDC1 接口。
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        = 6
wifi_power              = "axp22_aldo1"
```

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。

配置举例：

```
[3g_para]
3g_used             = 1
3g_usbc_num         = 2
3g_uart_num         = 0
bb_name             = "mu509"
bb_vbat             = port:PL03<1><default><default><0>
bb_on               = port:PM01<1><default><default><0>
bb_pwr_on           = port:PM03<1><default><default><0>
bb_wake             = port:PM04<1><default><default><0>
bb_rf_dis           = port:PM05<1><default><default><0>
bb_rst              = port:PM06<1><default><default><0>
```

25. gyroscope

25.1. [gy_para]

配置项	配置项含义
gy_used=xx	是否支持 gyr
gy_twi_id=xx	I2C 的 BUS 控制选择，0：TWI0;1:TWI1;2:TWI2
gy_twi_addr=xx	芯片的 I2C 地址
gy_int1=xx	中断 1 的 GPIO 配置
gy_int2＝xx	中断 2 的 GPIO 配置

配置举例：

```
[gy_para]
gy_used             = 1
gy_twi_id           = 2
gy_twi_addr         = 0x6a
gy_int1             = port:PA10<6><1><default><default>
gy_int2             =
```

注：目前方案中支持 gyroscope 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。

```
[gy_list_para]
```


gy_det_used = 1
l3gd20_gyr = 1

26. 光感(light sensor)

26.1. [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 = 1
ls_twi_id = 1
ls_twi_addr = 0x23
ls_int = port:PH17<6><1><default><default>

注：目前方案中支持 gyroscope 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。

[ls_list_para]
ls_det_used = 1
ltr_501als = 1

27. 罗盘 Compass

27.1. [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           = port:PA11<6><1><default><default>
```

28. 蓝牙(blueteeth)

28.1. [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
bt_wakeup =xx	BT WAKEUP GPIO 配置
bt_gpio=xx	BT 可选 GPIO 配置
bt_rst=xx	BT RESET GPIO 配置

配置举例：

```
[bt_para]
bt_used          =
bt_uart_id       =
bt_wakeup        =
bt_gpio          =
bt_rst           =
```

29. 数字音频总线（TDM）

29.1. [s_i2s1]

配置项	配置项含义
daudio_used = xx	是否使用该接口，默认要配置为 1 1：使用 0：不使用
daudio_master =xx	该参数为 tdm 的主从设置，通常设为 4。

	1: SND_SOC_DAIFMT_CBM_CFM(codec clk & FRM master) use 2: SND_SOC_DAIFMT_CBS_CFM(codec clk slave & FRM master) not use 3: SND_SOC_DAIFMT_CBM_CFS(codec clk master & frame slave) not use 4: SND_SOC_DAIFMT_CBS_CFS(codec clk & FRM slave) use
daudio_select = 1	该参数为设置 tdm 支持 i2s 格式还是 pcm 格式，使用默认值 i2s 格式 0: pcm. 1: i2s
audio_format = xx	该参数配置传输的数据格式，默认配置配置为 i2s 标准格式 1:SND_SOC_DAIFMT_I2S(standard i2s format). use 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). use 5:SND_SOC_DAIFMT_DSP_B(pcm. MSB is available on 1nd BCLK rising edge after LRC rising edge)
signal_inversion = xx	1:SND_SOC_DAIFMT_NB_NF(normal bit clock + frame) use 2:SND_SOC_DAIFMT_NB_IF(normal BCLK + inv FRM) 3:SND_SOC_DAIFMT_IB_NF(invert BCLK + nor FRM) use 4:SND_SOC_DAIFMT_IB_IF(invert BCLK + FRM)
mclk_fs = xx	support 128fs/192fs/256fs/384fs/512fs/768fs（使用默认值）
sample_resolution=xx	该参数指采样精度，通常默认 16 16bits/20bits/24bits
slot_width_select=xx	Slot 宽度（使用默认值 16）
pcm_lrck_period=xx	单声道 blck 个数/lrck 个数（使用默认值 32）
pcm_lrckr_period=xx	Lrckr 参数：使用默认值
msb_lsb_first=xx	0: msb first; 1: lsb first（使用默认值）
sign_extend=xx	Pcm 格式参数：选用默认
slot_index=xx	Pcm 格式参数：选用默认
frame_width=xx	Pcm 格式参数：选用默认
tx_data_mode=xx	Pcm 格式参数：选用默认
rx_data_mode=xx	Pcm 格式参数：选用默认
s_i2s1_lrckr	s_i2s1_lrckr 信号的 GPIO 配置
s_i2s1_dout1	s_i2s1_dout1 信号的 GPIO 配置
s_i2s1_dout2	s_i2s1_dout2 信号的 GPIO 配置
s_i2s1_dout3	s_i2s1_dout3 信号的 GPIO 配置
s_i2s1_mclk	s_i2s1_mclk 信号的 GPIO 配置
s_i2s1_bclk	s_i2s1_bclk 信号的 GPIO 配置
s_i2s1_lrclk	s_i2s1_lrclk 信号的 GPIO 配置

s_i2s1_din	s_i2s1_din 信号的 GPIO 配置
s_i2s1_dout0	s_i2s1_dout0 信号的 GPIO 配置

配置举例：

```
daudio_used          = 1
daudio_master        = 4
daudio_select        = 1
audio_format         = 1
signal_inversion     = 1
mclk_fs              = 512
sample_resolution    = 16
slot_width_select    = 16
;pcm_sync_period     = 256
pcm_lrck_period      = 32
pcm_lrckr_period     = 1
msb_lsb_first        = 0
sign_extend          = 0
slot_index           = 0
frame_width          = 0
tx_data_mode         = 0
rx_data_mode         = 0
s_i2s1_lrckr         = port:PM04<3><1><default><default>
s_i2s1_dout1         = port:PM05<3><1><default><default>
s_i2s1_dout2         = port:PM06<3><1><default><default>
s_i2s1_dout3         = port:PM07<3><1><default><default>
s_i2s1_mclk          = port:PM10<3><1><default><default>
s_i2s1_bclk          = port:PM11<3><1><default><default>
s_i2s1_lrclk         = port:PM12<3><1><default><default>
s_i2s1_din           = port:PM13<3><1><default><default>
s_i2s1_dout0         = port:PM14<3><1><default><default>
```

30. 数字音频总线（S/PDIF）

30.1. [spdif0]

配置项	配置项含义
spdif_used=xx	xx 为 0 时加载该模块，为 0 是不加载
spdif_dout =xx	Spdif out 的 gpio 控制
spdif_din=xx	Spdif din 的 gpio 控制

配置举例：

```
[spdif0]
spdif_used          = 1
spdif_dout          = port:PH18<3><1><default><default>
spdif_din           = port:PH17<3><1><default><default>
```

31. Codec 配置

31.1. [audio0]

配置项	配置项含义
audio_int_ctrl	Codec 耳机中断信号的 GPIO 配置
audio_pa_ctrl	Codec 喇叭 pa 信号的 GPIO 配置

配置举例：

```
[audio0]
audio_int_ctrl          = port:PL7<6><default><default><0>
audio_pa_ctrl           = port:PA16<1><default><default><0>
```

32. 红外(ir)

32.1. [s_cir0]

配置项	配置项含义
ir_used=xx	是否支持 ir
ir0_rx =xx	ir 的接收管脚 GPIO 配置
ir_power_key_code	ir 遥控器电源键的编码值
ir_addr_code	ir 遥控器设备地址

配置举例：

```
[s_cir0
ir_used             = 1
ir_rx               = port:PL06<3><1><default><default>
ir_power_key_code   = 0x0
ir_addr_code        = 0x0
```

32.2. [cir]

配置项	配置项含义
ir_used=xx	是否支持 ir
ir0_tx =xx	ir 的发送管脚 GPIO 配置

ir_used = 1
ir_tx = port:PH07<2><default><default><default>

33. PMU 电源

33.1. [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
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               = 0
pmu_irq_id               = 0
pmu_battery_rdc          = 100
pmu_battery_cap          = 0
pmu_batdeten             = 1
pmu_runtime_chgcur       = 900
pmu_earlysuspend_chgcur  = 900
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 = 0
pmu_pekoff_en = 1
pmu_pekoff_delay_time = 0
pmu_peklong_time = 1500
pmu_pekon_time = 1000
pmu_pwrok_time = 64
pmu_pwrok_shutdown_en = 0

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 = 0
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

33.2. [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……

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

34. Vf 表设置

34.1. cluster0 vf 表配置说明

c0_max_freq	cluster0 能达到的最高频率
c0_min_freq	cluster0 下降到最低频率
C0_LV_count	表示 cluster0 有 xx 个等级调频
C0_LV1_freq	在第一等级中能达到的频率
C0_LV1_volt	第一等级中的电压设置
C0_LV2_freq	第 2 等级中能达到的频率
C0_LV2_volt	第 2 等级中的电压设置
C0_LV3_freq	第 3 等级中能达到的频率
C0_LV3_volt	第 3 等级中的电压设置
C0_LV4_freq	第 4 等级中能达到的频率
C0_LV4_volt	第 4 等级中的电压设置
C0_LV5_freq	第 5 等级中能达到的频率
C0_LV5_volt	第 5 等级中的电压设置
C0_LV6_freq	第 6 等级中能达到的频率
C0_LV6_volt	第 6 等级中的电压设置
C0_LV7_freq	第 7 等级中能达到的频率
C0_LV7_volt	第 7 等级中的电压设置
C0_LV8_freq	第 8 等级中能达到的频率
C0_LV8_volt	第 8 等级中的电压设置

配置举例：

[dvfs_table]
c0_max_freq = 1008000000
c0_min_freq = 120000000

C0_LV_count = 8

C0_LV1_freq = 1104000000

C0_LV1_volt =

C0_LV2_freq = 1056000000
C0_LV2_volt =

C0_LV3_freq = 864000000
C0_LV3_volt =

C0_LV4_freq = 720000000
C0_LV4_volt =

C0_LV5_freq = 480000000
C0_LV5_volt =

C0_LV6_freq = 0
C0_LV6_volt =

C0_LV7_freq = 0
C0_LV7_volt =

C0_LV8_freq = 0
C0_LV8_volt =

34.2. Cluster1 vf 表配置说明

配置说明跟 cluster0 相同。
配置举例：
c1_max_freq = 1008000000
c1_min_freq = 120000000

C1_LV_count = 8

C1_LV1_freq = 1104000000
C1_LV1_volt = 900

C1_LV2_freq = 1056000000
C1_LV2_volt = 900

C1_LV3_freq = 864000000
C1_LV3_volt = 900

C1_LV4_freq = 720000000

C1_LV4_volt = 900

C1_LV5_freq = 480000000
C1_LV5_volt = 900

C1_LV6_freq = 0
C1_LV6_volt = 900

C1_LV7_freq = 0
C1_LV7_volt = 900

C1_LV8_freq = 0
C1_LV8_volt = 900

35. 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>
```

36. [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>

37. [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_rsb_used = 1
s_rsb_sck = port:PN00<3><1><2><default>
s_rsb_sda = port:PN01<3><1><2><default>

38. [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 = 1
s_jtag_tms = port:PL02<3><1><2><default>
s_jtag_tck = port:PL03<3><1><2><default>
s_jtag_tdo = port:PL04<3><1><2><default>
s_jtag_tdi = port:PL05<3><1><2><default>

39. Declaration

This document is the original work and copyrighted property of Allwinner Technology ("Allwinner"). Reproduction in whole or in part must obtain the written approval of Allwinner and give clear acknowledgement to the copyright owner.

The information furnished by Allwinner is believed to be accurate and reliable. Allwinner reserves the right to make changes in circuit design and/or specifications at any time without notice. Allwinner does not assume any responsibility and liability for its use. Nor for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of Allwinner. This datasheet neither states nor implies warranty of any kind, including fitness for any particular application.

