

H3 项目

sys_config 配置说明书 V1.1

文档履历

[illegible]

注意事项:

1. 文档中蓝色字体标识为模块芯片引脚配置，黑色字体标识为模块内部控制配置项；红色字体标识为客户端不允许修改部分；
2. 描述 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>
3. 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。

Confidential

目 录

1. 系统 (SYSTEM)	5
1.1. [product]	5
1.2. [platform]	5
1.3. [target]	5
1.4. [key_detect_en]	6
1.5. [fel_key]	6
1.6. [card_boot]	6
1.7. [box_start_os]	6
1.8. [boot_init_gpio]	7
1.9. [recovery_para]	7
1.10. [pm_para]	7
1.11. [card0_boot_para]	7
1.12. [card2_boot_para]	8
1.13. [twi_para]	8
1.14. [uart_para]	9
1.15. [force_uart_para]	9
1.16. [jtag_para]	9
1.17. [gpio_para]	9
1.18. [led_assign]	10
1.19. [clock]	10
2. dram 配置	11
2.1. [dram_para]	11
3. 以太网配置	13
3.1. [gmac0]	13
4. Statndby	15
4.1. [wakeup_src_para]	15
5. I2C 总线	16
5.1. [twi0]	16
5.2. [twi1]	16
5.3. [twi2]	16
6. 串口 (UART)	17
6.1. [uart0]	17
6.2. [uart1]	17
6.3. [uart2]	18
6.4. [uart3]	18
7. SPI 总线	19
7.1. [spi0]	19
7.2. [spi1]	19
7.3. [spi_devices]	19
7.4. [spi_board0]	20
8. 温控档位配置	21
8.1. [ths_para]	21
8.2. [cooler_table]	22
9. 闪存 (nand flash)	23

9.1. [nand0_para].....	23
10. 显示	25
10.1. [boot_disp]	25
10.2. [disp_init]	25
10.3. [hdmi_para]	26
10.4. [tv_para]	26
10.5. [di_para]	27
11. PWM.....	28
11.1. [pwm0_para].....	28
12. 摄像头	29
12.1. [csi0].....	29
13. tvout/tvin.....	33
13.1. [tvout_para]	33
13.2. [tvin_para]	33
14. SD/MMC	34
14.1. [mmc0_para]	34
14.2. [mmc1_para]	34
14.3. [mmc2_para]	35
15. SIM 卡	37
15.1. [smc_para].....	37
16. USB 控制器标志	38
16.1. [usbc0].....	38
16.2. [usbc1].....	39
16.3. [usbc2].....	39
17. USB DEVICE.....	41
17.1. [usb_feature].....	41
17.2. [msc_feature].....	41
17.3. [serial_feature].....	41
18. WIFI	42
18.1. [module_para]	42
18.2. [wifi_para].....	42
19. 蓝牙	44
19.1. [bt_para]	44
20. 音频配置 (audio)	45
20.1. [pcm0]	45
20.2. [audio0]	46
20.3. [spdif0]	47
20.4. [audiohub]	47
21. 红外(ir).....	48
21.1. [s_cir0]	48
21.2. [cir]	48
22. Vf 表设置	50
22.1. [dvfs_table].....	50
22.2. [gpu_dvfs_table].....	51
23. Board Vendor Id.....	52
24. Transport Stream Controller	53

24.1. [ts0].....	53
25. GPIO 模拟电源按键.....	54
25.1. [gpio_power_key]	54
26. CPUS	55
26.1. [s_uart0]	55
26.2. [s_rsb0].....	55
26.3. [s_jtag0].....	55
26.4. [s_powchk]	55
Declaration.....	57

confidential

1. 系统 (SYSTEM)

1.1. [product]

配置项	配置项含义
version = "100"	sdk 版本号 1.00
machine = "dolphin-p1"	sdk 代号

配置举例：

```
version      = "100"
machine      = "dolphin-p1"
```

1.2. [platform]

配置项	配置项含义
eraseflag	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）
debug_mode	0：boot 阶段启动无串口输出（不建议使用） 1：boot 阶段启动串口输出
next_work	定义 USB 量产完成后的动作，只对量产工具 PhoenixUSBpro 有效 (2：重启 3：关机 4：重新量产)
secure_without_OS	1：不支持安全系统； 0：支持安全系统 V1.1 SDK 默认是非安全系统，如果客户端方案不使用安全系统，请不要更改此处配置，否则会锁定 IC 造成无法使用的严重问题
secure_bit	0：不支持 EFUSE 烧写； 1：支持 EFUSE 烧写 V1.1 SDK 默认是非安全系统，如果客户端方案不使用安全系统，请不要更改此处配置，否则会锁定 IC 造成无法使用的严重问题

配置举例：

```
debug_mode      = 1
eraseflag       = 1
next_work       = 3
secure_without_OS = 1
secure_bit      = 0
```

1.3. [target]

配置项	配置项含义
boot_clock	启动频率，单位 MHZ
storage_type	启动介质选择 0：nand，1：card0，2：card2，-1：（default）自动扫描启动介质
burn_key	Dragonkey 烧号工具检测模块，1：开启， 0：关闭

配置举例：

```
boot_clock      = 1008
storage_type     = -1
burn_key        = 0
```

1.4. [key_detect_en]

配置项	配置项含义
keyen_flag	按键键值检测功能 1: 支持按键检测功能; 0: 不支持按键检测功能

配置举例:

```
keyen_flag      = 0
```

1.5. [fel_key]

配置项	配置项含义
fel_key_max	一键进烧写功能~的按键值配置, 按住位于 min~max 范围内的按键都可以强制进烧写
fel_key_min	一键进烧写功能~的按键值配置, 按住位于 min~max 范围内的按键都可以强制进烧写

配置举例:

```
fel_key_max      =07
fel_key_min      =02
```

1.6. [card_boot]

配置项	配置项含义
logical_start	启动卡逻辑起始扇区
sprite_work_delay	正常卡量产和一键 recovery 指示灯的闪烁间隔
sprite_err_delay	非正常卡量产和一键 recovery 指示灯的闪烁间隔
sprite_gpio0	卡量产/一键 recovery led 指示灯 GPIO 配置
next_work	卡量产完成后状态: 1: 不做任何动作 2: 重启 3: 关机 4: 量产

配置举例:

```
logical_start    = 40960
sprite_work_delay = 500
sprite_err_delay = 200
sprite_gpio0     = port:PA15<1><default><default><default>
next_work        = 3
```

1.7. [box_start_os]

配置项	配置项含义
used	是否启用该项功能: 1: 启用 0: 不启用
start_type	是否上电启动系统, 1: 直接启动系统; 0: 上电不允许直接启动系统
irkey_used	是否启用 ir 控制启动 1: 启用 ir 按键启动 0: 禁用 ir 按键启动
pmukey_used	是否启用 power 按键启动 1: 启动 power 按键启动 0: 禁用 power 按键启动
led_power	电源指示灯, 1: on, 0: off
led_state	状态指示灯, 1: on, 0: off

配置举例:


```

used          = 1
start_type    = 1
irkey_used    = 1
pmukey_used   = 0
led_power     = 0
led_state     = 0

```

1.8. [boot_init_gpio]

配置项	配置项含义
used	Boot 启动阶段初始化 GPIO，1：开启 0：禁用
gpio0	GPIO 配置

配置举例：

```

used      = 1
gpio0     = port:PA15<1><default><default><1>

```

1.9. [recovery_para]

配置项	配置项含义
used	是否开启一键 recovery 功能，1：开启 0：禁用
mode	模式配置 1：一键进入 OTA 2：一键进入固件恢复模式(通过 sysrecovery 分区来恢复)
recovery_key	按键 GPIO 配置

配置举例：

```

used      = 1
mode      = 2
recovery_key = port:PL04<0><default><default><default>

```

1.10.[pm_para]

配置项	配置项含义
standby_mode	1: super standby other: normal standby

配置举例：

```

standby_mode = 1

```

1.11.[card0_boot_para]

配置项	配置项含义
card_ctrl	卡量产相关的控制器选择 0
card_high_speed	速度模式 0 为低速，1 为高速
card_line	4: 4 线卡，8: 8 线卡
sdc_d1	sdc 卡数据 1 线信号的 GPIO 配置
sdc_d0	sdc 卡数据 0 线信号的 GPIO 配置
sdc_clk	sdc 卡时钟信号的 GPIO 配置
sdc_cmd	sdc 命令信号的 GPIO 配置

sdc_d3	sdc 卡数据 3 线信号的 GPIO 配置
sdc_d2	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><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>

```

1.12.[card2_boot_para]

配置项	配置项含义
card_ctrl	卡启动控制器选择 2
card_high_speed	速度模式 0 为低速，1 为高速
card_line	4：4 线卡，8：8 线卡
sdc_cmd	sdc 命令信号的 GPIO 配置
sdc_clk	sdc 卡时钟信号的 GPIO 配置
sdc_d0	sdc 卡数据 0 线信号的 GPIO 配置
sdc_d1	sdc 卡数据 1 线信号的 GPIO 配置
sdc_d3	sdc 卡数据 3 线信号的 GPIO 配置
sdc_d2	sdc 卡数据 2 线信号的 GPIO 配置
sdc_2xmode	设置新采样模式，该项必须配置成 sdc_2xmode=1
sdc_ddrmode	MMC 通信模式 DDR 模式， 使用 ddr 模式，必须设置成 sdc_ddrmode = 1

配置举例：

```

card_ctrl      = 2
card_high_speed = 1
card_line      = 8
sdc_cmd        = port:PC06<3><1><2><default>
sdc_clk        = port:PC05<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>
sdc_2xmode     = 1
sdc_ddrmode    = 1

```

1.13.[twi_para]

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

配置举例：

```
twi_port      = 0
twi_scl       = port:PA11<2><default><default><default>
twi_sda       = port:PA12<2><default><default><default>
```

1.14.[uart_para]

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

配置举例：

```
uart_debug_port = 0
uart_debug_tx   = port:PA04<2><1><default><default>
uart_debug_rx   = port:PA05<2><1><default><default>
```

1.15.[force_uart_para]

配置项	配置项含义
force_uart_port	强制调试串口控制器编号
force_uart_tx	强制调试串口发送 GPIO 配置
force_uart_rx	强制调试串口接收 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>
```

1.16.[jtag_para]

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

配置举例：

```
jtag_enable    = 1
jtag_ms        = port:PA00<3><default><default><default>
jtag_ck        = port:PA01<3><default><default><default>
jtag_do        = port:PA02<3><default><default><default>
jtag_di        = port:PA03<3><default><default><default>
```

1.17.[gpio_para]

配置项	配置项含义
gpio_used	内核 GPIO 初始化使能功能，1：开启 0：禁用
gpio_num	GPIO 引脚数目
gpio_pin_1	GPIO 引脚配置
gpio_pin_2	GPIO 引脚配置

配置举例：

```

gpio_used      = 1
gpio_num       = 2
gpio_pin_1     = port:PL10<1><default><default><1>
gpio_pin_2     = port:PA15<1><default><default><0>

```

1.18.[led_assign]

配置项	配置项含义
normal_led	正常工作状态下，状态指示灯(LED)引脚名称
standby_led	standby 工作状态下，状态指示灯(LED)引脚名称

配置举例：

```

normal_led     = "gpio_pin_2"
standby_led    = "gpio_pin_1"

```

1.19.[clock]

配置项	配置项含义
pll_video	video 时钟频率
pll_ve	ve 时钟频率
pll_periph0	periph0 时钟频率
Pll_gpu	gpu 时钟频率
pll_periph1	periph1 时钟频率
pll_de	de 时钟频率

配置举例：

```

pll_video      = 297
pll_ve         = 402
pll_periph0    = 600
pll_gpu        = 576
pll_periph1    = 600
pll_de         = 864

```

2. dram 配置

2.1. [dram_para]

配置项	配置项含义
dram_clk	DRAM 的时钟频率，单位为 MHz；它为 24 的整数倍，最低不得低于 120，在不开启自动适应功能的情况有效
dram_type	DRAM 类型：2 为 DDR2、3 为 DDR3、6 为 lpddr2、7 为 lpddr3，在开启自动适应功能的基础上，不允许更改。
dram_zq	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_odt_en	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	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_para2	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_mr0	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	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_mr2	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_mr3	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr0	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr1	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr2	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr3	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr4	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr5	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr6	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr7	自动适应功能下 dram clk 设置： dram_tpr7[29:20] to set lpddr2 clk, dram_tpr7[19:10] to set lpddr3 clk, dram_tpr7[9:0] to set ddr3 or ddr3l clk
dram_tpr8	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr9	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr10	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr11	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr12	DRAM 控制器内部参数，由原厂来进行调节，请勿修改
dram_tpr13	bit[13]：是否开启自动适应功能， 0：不开启， 1：开启

配置举例：

```
[dram_para]
dram_clk      = 576
dram_type     = 3
dram_zq       = 0x3b3bfb
dram_odt_en   = 0x1
dram_para1    = 0x10E40000
```

dram_para2	= 0x0000
dram_mr0	= 0x1840
dram_mr1	= 0x40
dram_mr2	= 0x18
dram_mr3	= 0x2
dram_tpr0	= 0x0048A192
dram_tpr1	= 0x01C2418D
dram_tpr2	= 0x00076051
dram_tpr3	= 0
dram_tpr4	= 0
dram_tpr5	= 0
dram_tpr6	= 100
dram_tpr7	= 0x13890240
dram_tpr8	= 0
dram_tpr9	= 0
dram_tpr10	= 0
dram_tpr11	= 0x6aaa0000
dram_tpr12	= 0x7979
dram_tpr13	= 0x1802800

Confidential

3. 以太网配置

3.1. [gmac0]

配置项	配置项含义
gmac_used	Gmac 模块是否使能: 0: not used, 1: external phy, 2: internal phy
gmac_rxd3	Gmac rx3 的 GPIO 配置
gmac_rxd2	Gmac rx2 的 GPIO 配置
gmac_rxd1	Gmac rx1 的 GPIO 配置
gmac_rxd0	Gmac rx0 的 GPIO 配置
gmac_rxclk	Gmac 接收时钟
gmac_rxdv	Gmac 接收数有效使能
gmac_rxerr	Gmac 接收错误使能
gmac_txd3	Gmac tx3 的 GPIO 配置
gmac_txd2	Gmac tx2 的 GPIO 配置
gmac_txd1	Gmac tx1 的 GPIO 配置
gmac_txd0	Gmac tx0 的 GPIO 配置
gmac_crs	Gmac 载波监测 (仅用于半双工)
gmac_txclk	Gmac MII 接口发送时钟
gmac_txen	Gmac 发送使能 GPIO 配置
gmac_txer	Gmac 发送错误使能
gmac_col	Gmac 冲突监测 (仅用于半双工)
gmac_mdc	Gmac 配置接口时钟
gmac_mdio	Gmac 配置接口数据
gmac_power1	Gmac 供电配置

配置举例:

[gmac0]

```

gmac_used          = 2
;gmac_rxd3         = port:PD00<2><default><3><default>
;gmac_rxd2         = port:PD01<2><default><3><default>
;gmac_rxd1         = port:PD02<2><default><3><default>
;gmac_rxd0         = port:PD03<2><default><3><default>
;gmac_rxclk        = port:PD04<2><default><3><default>
;gmac_rxdv         = port:PD05<2><default><3><default>
;gmac_rxerr        = port:PD06<2><default><3><default>
;gmac_txd3         = port:PD07<2><default><3><default>
;gmac_txd2         = port:PD08<2><default><3><default>
;gmac_txd1         = port:PD09<2><default><3><default>
;gmac_txd0         = port:PD10<2><default><3><default>
;gmac_crs          = port:PD11<2><default><3><default>
;gmac_txclk        = port:PD12<2><default><3><default>
;gmac_txen         = port:PD13<2><default><3><default>
;gmac_txerr        = port:PD14<2><default><3><default>
;gmac_col          = port:PD15<2><default><3><default>

```

```
;gmac_mdc          = port:PD16<2><default><3><default>  
;gmac_mdio         = port:PD17<2><default><3><default>  
gmac_power1       =
```

Confidential

4. Statndby

4.1. [wakeup_src_para]

配置项	配置项含义
cpu_en	Power on or off 1: power on 0: power off
cpu_freq	Indicating lowest freq, uint 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_freq	if not enter selfresh, indicating lowest freq, uint is Mhz
wakeup_src0	休眠唤醒源
wakeup_src_wl	wifi 唤醒 pin 配置
wakeup_src_bt	Bluetooth 唤醒 pin 配置

配置举例：

```

cpu_en           = 0
cpu_freq         = 48
pll_ratio        = 0x111
dram_selfresh_en = 1
dram_freq        = 36
wakeup_src0      =
wakeup_src_bt    = port:PL03<6><default><default><0>

```

5. I2C 总线

主控有 3 个 I2C（twi）控制器，通过 twi0/twi1/twi2 区分配置。

5.1. [twi0]

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

配置举例：

```
twi_used    = 1
twi_scl     = port:PA11<2><default><default><default>
twi_sda     = port:PA12<2><default><default><default>
```

5.2. [twi1]

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

配置举例：

```
twi_used    = 1
twi_scl     = port:PA18<3><default><default><default>
twi_sda     = port:PA19<3><default><default><default>
```

5.3. [twi2]

配置项	配置项含义
twi2_used	TWI 使用控制：1 使用，0 不用
twi2_scl	TWI SCK 的 GPIO 配置
twi2_sda	TWI SDA 的 GPIO 配置

配置举例：

```
twi_used    = 0
twi_scl     = port:PE12<3><default><default><default>
twi_sda     = port:PE13<3><default><default><default>
```

6. 串口(UART)

主控有 4 路 uart 接口, 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. [uart0]

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, 有效值为: 2/4/8; 2: 2 线模式; 4: 4 线模式; 8: 8 线模式。
uart_tx	UART TX 的 GPIO 配置
uart_rx	UART RX 的 GPIO 配置

配置举例:

```
uart_used      = 1
uart_port      = 0
uart_type      = 2
uart_tx        = port:PA03<2><1><default><default>
uart_rx        = port:PA05<2><1><default><default>
```

6.2. [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, 有效值为: 2/4/8; 2: 2 线模式; 4: 4 线模式; 8: 8 线模式。
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:

```
uart_used      = 1
uart_port      = 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>
```

6.3. [uart2]

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，有效值为：2/4/8； 2：2 线模式； 4：4 线模式； 8：8 线模式
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：

```

uart_used      = 0
uart_port      = 2
uart_type      = 4
uart_tx        = port:PA00<2><1><default><default>
uart_rx        = port:PA01<2><1><default><default>
uart_rts       = port:PA02<2><1><default><default>
uart_cts       = port:PA03<2><1><default><default>

```

6.4. [uart3]

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，有效值为：2/4/8； 2：2 线模式； 4：4 线模式； 8：8 线模式
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：

```

uart_used      = 0
uart_port      = 3
uart_type      = 4
uart_tx        = port:PA13<3><1><default><default>
uart_rx        = port:PA14<3><1><default><default>
uart_rts       = port:PA15<3><1><default><default>
uart_cts       = port:PA16<3><1><default><default>

```

7. SPI 总线

7.1. [spi0]

配置项	配置项含义
spi_used	SPI 使用控制：1 使用，0 不用
spi_cs_bitmap	由于 SPI 控制器支持多个 CS，这一个参数表示 CS 的掩码；
spi_cs0	SPI CS0 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：

```

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>

```

7.2. [spi1]

配置项	配置项含义
spi_used	SPI 使用控制：1 使用，0 不用
spi_cs_bitmap	由于 SPI 控制器支持多个 CS，这一个参数表示 CS 的掩码；
spi_cs0	SPI CS0 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：

```

spi_used          = 0
spi_cs_bitmap     = 1
spi_cs0           = port:PA13<2><1><default><default>
spi_sclk          = port:PA14<2><default><default><default>
spi_mosi          = port:PA15<2><default><default><default>
spi_miso          = port:PA16<2><default><default><default>

```

7.3. [spi_devices]

配置项	配置项含义
spi_dev_num	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_dev_num       = 1

```

7.4. [spi_board0]

配置项	配置项含义
modalias	Spi 设备名字
max_speed_hz	最大传输速度 (HZ)
bus_num	Spi 设备控制器序号
chip_select	理论上可以选 0, 1
mode	SPI MOSI 的 GPIO 配置可选值 0-3

配置举例:

```
modalias      = "m25p32"
max_speed_hz  = 33000000
bus_num       = 0
chip_select   = 0
mode          = 0
```

confidential

8. 温控档位配置

IC 温度控制配置表，此配置有原厂提供，请勿修改！擅自修改将严重影响整机稳定性

8.1. [ths_para]

配置项	配置项含义
ths_used	启用置 1，反之置 0，此项必须置 1，CPU 温控保护功能
ths_trip1_count	CPU 温度控制档位数
ths_trip1_0	第一档温控档位
ths_trip1_1	第二档温控档位
ths_trip1_2	第三档温控档位
ths_trip1_3	第四档温控档位
ths_trip1_4	同上规律类推
ths_trip1_5	同上规律类推
ths_trip1_6	同上规律类推
ths_trip1_7	同上规律类推
ths_trip1_0_min	第一档温控范下限对应的 cool 档位，0 对应于 cool0 档位
ths_trip1_0_max	第一档温控范上限对应的 cool 档位，1 对应于 cool1 档位
ths_trip1_1_min	同上规律类推
ths_trip1_1_max	同上规律类推
ths_trip1_2_min	同上规律类推
ths_trip1_2_max	同上规律类推
ths_trip1_3_min	同上规律类推
ths_trip1_3_max	同上规律类推
ths_trip1_4_min	同上规律类推
ths_trip1_4_max	同上规律类推
ths_trip1_5_min	同上规律类推
ths_trip1_5_max	同上规律类推
ths_trip1_6_min	同上规律类推
ths_trip1_6_max	同上规律类推
ths_trip2_count	CPU 过热温控保护档位数目
ths_trip2_0	第一档过热档位

配置举例：

```
ths_used           = 1
ths_trip1_count    = 5
ths_trip1_0        = 70
ths_trip1_1        = 80
ths_trip1_2        = 90
ths_trip1_3        = 100
ths_trip1_4        = 105
ths_trip1_5        = 0
ths_trip1_6        = 0
```

```

ths_trip1_7           = 0
ths_trip1_0_min       = 0
ths_trip1_0_max       = 1
ths_trip1_1_min       = 1
ths_trip1_1_max       = 2
ths_trip1_2_min       = 2
ths_trip1_2_max       = 3
ths_trip1_3_min       = 3
ths_trip1_3_max       = 4
ths_trip1_4_min       = 4
ths_trip1_4_max       = 4
ths_trip1_5_min       = 0
ths_trip1_5_max       = 0
ths_trip1_6_min       = 0
ths_trip1_6_max       = 0
ths_trip2_count       = 1
ths_trip2_0           = 105

```

8.2. [cooler_table]

配置项	配置项含义
cooler_count	散热配置档位数目
cooler0	散热配置档位 0 配置，此配置和 ths_para 紧密相连，禁止修改
cooler1	散热配置档位 1 配置
cooler2	散热配置档位 2 配置
cooler3	同上规律类推
cooler4	同上规律类推

配置举例：

```

cooler_count          = 5
cooler0               = "1200000 4 4294967295 0"
cooler1               = "1200000 3 4294967295 0"
cooler2               = "1200000 2 4294967295 0"
cooler3               = "1200000 1 4294967295 0"
cooler4               = "504000 1 4294967295 0"

```


9. 闪存(nand flash)

9.1. [nand0_para]

配置项	配置项含义
nand_support_2ch	nand0 是否使能双通道
nand0_used	nand0 模块使能标志
nand0_we	nand0 写时钟信号的 GPIO 配置
nand0_ale	nand0 地址使能信号的 GPIO 配置
nand0_cle	nand0 命令使能信号的 GPIO 配置
nand0_cel	nand0 片选 1 信号的 GPIO 配置
nand0_ce0	nand0 片选 0 信号的 GPIO 配置
nand0_nre	nand0 读时钟信号的 GPIO 配置
nand0_rb0	nand0 Read/Busy 1 信号的 GPIO 配置
nand0_rb1	nand0 Read/Busy 0 信号的 GPIO 配置
nand0_d0	nand0 数据总线信号的 GPIO 配置
nand0_d1	/
nand0_d2	/
nand0_d3	/
nand0_d4	/
nand0_d5	/
nand0_d6	/
nand0_d7	/
nand0_ce2	nand0 片选 2 信号的 GPIO 配置
nand0_ce3	nand0 片选 3 信号的 GPIO 配置
nand0_ndqs	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><0><1><default>
nand0_ale           = port:PC01<2><0><1><default>
nand0_cle           = port:PC02<2><0><1><default>
nand0_cel           = port:PC03<2><1><1><default>
nand0_ce0           = port:PC04<2><1><1><default>
nand0_nre           = port:PC05<2><0><1><default>
nand0_rb0           = port:PC06<2><1><1><default>
nand0_rb1           = port:PC07<2><1><1><default>
nand0_d0            = port:PC08<2><0><1><default>
nand0_d1            = port:PC09<2><0><1><default>
nand0_d2            = port:PC10<2><0><1><default>
nand0_d3            = port:PC11<2><0><1><default>
nand0_d4            = port:PC12<2><0><1><default>
nand0_d5            = port:PC13<2><0><1><default>
```

nand0_d6	= port:PC14<2><0><1><default>
nand0_d7	= port:PC15<2><0><1><default>
nand0_ndqs	= port:PC16<2><0><1><default>

Confidential

10.显示

10.1.[boot_disp]

配置项	配置项含义
advert_disp	0: do not hotplud for lcd; 1: need hotplud for hdmi/tv
auto_hpd	支持热插拔检测配置 1: 支持 0: 不支持
output_type	显示输出默认配置(0:none, 1:lcd, 2:tv, 4:hdmi, 8:vga)
hdmi_channel	Hdmi channel 默认配置
hdmi_mode	Hdmi 输出默认配置
cvbs_channel	Cvbs channel 默认配置
cvbs_mode	Cvbs 输出默认配置(11:PAL, 14:NTSC)
output_full	boot logo 是否全屏显示 (H8 必须是 1)
hdmi_mode_check	在 boot 阶段是否打开“检查电视是否支持 HDMI 模式”的功能 (1: 打开, 0: 关闭)

配置具体:

```
advert_disp      = 0
auto_hpd         = 1
output_type      = 4
hdmi_channel     = 0
hdmi_mode        = 4
cvbs_channel     = 1
cvbs_mode        = 11
output_full      = 1
hdmi_mode_check  = 1
```

在不能识别电视的唯一性的情况下，目前的 hdmi boot 识别电视支持的分辨率都是有缺陷的，boot 识别 hdmi 分辨率支持两种策略：

1. 优先解析电视 edid 是支持的，如果电视的 edid 显示不支持则恢复到默认 720p 50hz，这样会造成用户家里只有一台电视且电视的 edid 有问题的时候，用户设置了后每次盒子重启都恢复默认值 720p 50hz，这样用户体验差。即配置为 `hdmi_mode_check = 1`

2. 只认最后一次设置 hdmi 分辨率的保留值，不认电视的 edid，但这样会出现插不同电视没有显示，现在这个问题可以通过“遥控快捷按键---hdmi 分辨率切换按键”规避，**要求厂家的遥控器有切换分辨率的支持**，这种策略推荐使用。参见《H3 Android 定制化文档 V1.0》9.1.3 小节的“显示策略设置”。即配置为 `hdmi_mode_check = 0`

10.2.[disp_init]

配置项	配置项含义
disp_init_enable	是否进行显示的初始化设置
disp_mode	显示模式： 0:screen0<screen0,fb0> 1: screen1<screen1,fb0>
screen0_output_type	屏 0 输出类型 (0:none; 1:lcd; 2:tv; 3:hdmi; 4:vga)

screen0_output_mode	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	屏 1 输出类型 (0:none; 1:lcd; 2:tv; 3:hdmi; 4:vga)
screen1_output_mode	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	fb0 的格式 (0:ARGB 1:ABGR 2:RGBA 3:BGR)
fb0_width	fb0 的宽度, 为 0 时将按照输出设备的分辨率; 为 1280 时输出分辨率宽 1280
fb0_height	fb0 的高度, 为 0 时将按照输出设备的分辨率; 为 720 时输出分辨率高 1280
fb1_format	fb1 的格式 (0:ARGB 1:ABGR 2:RGBA 3:BGR)

配置举例:

```

disp_init_enable      = 1
disp_mode              = 0
screen0_output_type    = 3
screen0_output_mode    = 4
screen1_output_type    = 3
screen1_output_mode    = 4
fb0_format             = 0
fb0_width              = 1280
fb0_height             = 720
fb1_format             = 0
fb1_width              = 0
fb1_height             = 0

```

10.3.[hdmi_para]

配置项	配置项含义
para_used	是否使用 hdmi。1: 使用; 0: 不使用
hdmi_power	内核阶段 hdmi 电源配置

配置举例:

```

hdmi_used              = 1
hdmi_power              = "vcc-hdmi-18"

```

10.4.[tv_para]

配置项	配置项含义
tv_used	是否使用 CVBS
tv_dac_used	enable the tv driver module
tv_dac_src0	choose the tv function, cvbs is 0

配置举例:

```

tv_used                = 1
tv_dac_used            = 1
tv_dac_src0            = 0

```

10.5.[di_para]

配置项	配置项含义
di_used	是否使用 de-interlace; 0: 不使用, 1: 使用

配置举例:

di_used = 1

Confidential

11.PWM

11.1.[pwm0_para]

配置项	配置项含义
pwm_used	是否使用 PWM0
pwm_positive	PWM 输出 GPIO 配置

配置举例：

pwm_used = 0
pwm_positive = port:PA05<3><0><default><default>

confidential

12.摄像头

12.1.[csi0]

配置项	配置项含义
vip_used	使用 mipi 接口的 sensor 请填写 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_pck	pclk 信号的 GPIO 配置。
vip_csi_mck	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_csi_hsync	hsync 信号的 GPIO 配置
vip_csi_vsync	vsync 信号的 GPIO 配置
vip_csi_d0	csi d0 信号的 GPIO 配置
vip_csi_d1	csi d1 信号的 GPIO 配置
vip_csi_d2	csi d2 信号的 GPIO 配置
vip_csi_d3	csi d3 信号的 GPIO 配置
vip_csi_d4	csi d4 信号的 GPIO 配置
vip_csi_d5	csi d5 信号的 GPIO 配置
vip_csi_d6	csi d6 信号的 GPIO 配置
vip_csi_d7	csi d7 信号的 GPIO 配置
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。

<code>vip_dev0_afvdd</code>	VCM 电源配置一般不用配置。
<code>vip_dev0_afvdd_vol</code>	VCM 电压值为 2.8V。
<code>vip_dev0_power_en</code>	Sensor power enable 引脚 GPIO 配置。
<code>vip_dev0_reset</code>	Sensor reset 引脚 GPIO 配置。
<code>vip_dev0_pwn</code>	Sensor power down 引脚 GPIO 配置。
<code>vip_dev0_flash_en</code>	闪光灯 enable 引脚 GPIO 配置。
<code>vip_dev0_flash_mode</code>	闪光灯 flash mode 引脚 GPIO 配置。
<code>vip_dev0_af_pwn</code>	VCM driver power down 引脚 GPIO 配置。
<code>vip_dev0_act_used</code>	模组包含 VCM driver 时候填 1。
<code>vip_dev0_act_name</code>	VCM driver 名字，如 “ad5820_act”。
<code>vip_dev0_act_slave</code>	VCM driver slave 地址。
<code>vip_dev1_mname</code>	设置 sensor 1 名称，如 “ov5648”。
<code>vip_dev1_pos</code>	摄像头位置前置填“front”，后置填“rear”。
<code>vip_dev1_lane</code>	请参考实际模组的 lane 数目填写。
<code>vip_dev1_twi_id</code>	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。
<code>vip_dev1_twi_addr</code>	请参考实际模组的 8bit ID 填写，如 0x6c。
<code>vip_dev1_isp_used</code>	如果是 RAW sensor 必须填 1，YUV 填 0。
<code>vip_dev1_fmt</code>	如果是 RAW 格式填 1，YUV 填 0。
<code>vip_dev1_stby_mode</code>	填 0。
<code>vip_dev1_vflip</code>	Sensor 图像垂直翻转。
<code>vip_dev1_hflip</code>	Sensor 图像水平翻转。
<code>vip_dev1_iovdd</code>	IOVDD 配置，请参考实际原理图填写。
<code>vip_dev1_iovdd_vol</code>	IOVDD 电压值一般为 2.8V(2800000)。
<code>vip_dev1_avdd</code>	AVDD 配置，如 “axp15_aldo2”。
<code>vip_dev1_avdd_vol</code>	AVDD 电压值，一般为 2.8V(2800000)。
<code>vip_dev1_dvdd</code>	DVDD 配置，如 “axp22_eldo1”。
<code>vip_dev1_dvdd_vol</code>	DVDD 电压值参考 datasheet，1.2/1.5/1.8V。
<code>vip_dev1_afvdd</code>	VCM 电源配置一般不用配置。
<code>vip_dev1_afvdd_vol</code>	VCM 电压值为 2.8V。
<code>vip_dev1_power_en</code>	Sensor power enable 引脚 GPIO 配置。
<code>vip_dev1_reset</code>	Sensor reset 引脚 GPIO 配置。
<code>vip_dev1_pwn</code>	Sensor power down 引脚 GPIO 配置。
<code>vip_dev1_flash_en</code>	闪光灯 enable 引脚 GPIO 配置。
<code>vip_dev1_flash_mode</code>	闪光灯 flash mode 引脚 GPIO 配置。
<code>vip_dev1_af_pwn</code>	VCM driver power down 引脚 GPIO 配置。
<code>vip_dev1_act_used</code>	模组包含 VCM driver 时候填 1。
<code>vip_dev1_act_name</code>	VCM driver 名字，如 “ad5820_act”。
<code>vip_dev1_act_slave</code>	VCM driver slave 地址。

配置举例：

```

vip_used          = 0
vip_mode          = 0
vip_dev_qty       = 1
vip_define_sensor_list = 0

```


<code>vip_csi_pck</code>	<code>= port:PE00<2><default><default><default></code>
<code>vip_csi_mck</code>	<code>= port:PE01<2><default><default><default></code>
<code>vip_csi_hsync</code>	<code>= port:PE02<2><default><default><default></code>
<code>vip_csi_vsync</code>	<code>= port:PE03<2><default><default><default></code>
<code>vip_csi_d0</code>	<code>= port:PE04<2><default><default><default></code>
<code>vip_csi_d1</code>	<code>= port:PE05<2><default><default><default></code>
<code>vip_csi_d2</code>	<code>= port:PE06<2><default><default><default></code>
<code>vip_csi_d3</code>	<code>= port:PE07<2><default><default><default></code>
<code>vip_csi_d4</code>	<code>= port:PE08<2><default><default><default></code>
<code>vip_csi_d5</code>	<code>= port:PE09<2><default><default><default></code>
<code>vip_csi_d6</code>	<code>= port:PE10<2><default><default><default></code>
<code>vip_csi_d7</code>	<code>= port:PE11<2><default><default><default></code>
<code>vip_csi_sck</code>	<code>= port:PE12<2><default><default><default></code>
<code>vip_csi_sda</code>	<code>= port:PE13<2><default><default><default></code>

<code>vip_dev0_mname</code>	<code>= "ov5640"</code>
<code>vip_dev0_pos</code>	<code>= "rear"</code>
<code>vip_dev0_lane</code>	<code>= 1</code>
<code>vip_dev0_twi_id</code>	<code>= 0</code>
<code>vip_dev0_twi_addr</code>	<code>= 0x78</code>
<code>vip_dev0_isp_used</code>	<code>= 0</code>
<code>vip_dev0_fmt</code>	<code>= 0</code>
<code>vip_dev0_stby_mode</code>	<code>= 0</code>
<code>vip_dev0_vflip</code>	<code>= 0</code>
<code>vip_dev0_hflip</code>	<code>= 0</code>
<code>vip_dev0_iovdd</code>	<code>= ""</code>
<code>vip_dev0_iovdd_vol</code>	<code>= 2800000</code>
<code>vip_dev0_avdd</code>	<code>= ""</code>
<code>vip_dev0_avdd_vol</code>	<code>= 2800000</code>
<code>vip_dev0_dvdd</code>	<code>= ""</code>
<code>vip_dev0_dvdd_vol</code>	<code>= 1500000</code>
<code>vip_dev0_afvdd</code>	<code>= ""</code>
<code>vip_dev0_afvdd_vol</code>	<code>= 2800000</code>
<code>vip_dev0_power_en</code>	<code>=</code>
<code>vip_dev0_reset</code>	<code>=</code>
<code>vip_dev0_pwdn</code>	<code>=</code>
<code>vip_dev0_flash_en</code>	<code>=</code>
<code>vip_dev0_flash_mode</code>	<code>=</code>
<code>vip_dev0_af_pwdn</code>	<code>=</code>

<code>vip_dev0_act_used</code>	<code>= 0</code>
<code>vip_dev0_act_name</code>	<code>= "ad5820_act"</code>
<code>vip_dev0_act_slave</code>	<code>= 0x18</code>

<code>vip_dev1_mname</code>	<code>= ""</code>
-----------------------------	-------------------

```
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    = ""
vip_dev1_iovdd_vol = 2800000
vip_dev1_avdd     = ""
vip_dev1_avdd_vol = 2800000
vip_dev1_dvdd     = ""
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  =
```

13.tvout/tvin

13.1.[tvout_para]

配置项	配置项含义
tvout_used	是否使用 tvout。 1：使用 0：不使用
tvout_channel_num	使用的 tvout 通道号
tv_en	tvout 通道使能

配置举例：

```
tvout_used          = 0
tvout_channel_num   =
tv_en               =
```

13.2.[tvin_para]

配置项	配置项含义
tvin_used	是否使用 tvint。 1：使用 0：不使用
tvin_channel_num	使用的 tvin 通道号

配置举例：

```
tvin_used          = 0
tvin_channel_num   =
```

14.SD/MMC

14.1.[mmc0_para]

配置项	配置项含义
sdc_used	SDC 使用控制：1 使用，0 不用
sdc_detmode	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	位宽：1-1bit，4-4bit
sdc_d1	SDC DATA1 的 GPIO 配置
sdc_d0	SDC DATA0 的 GPIO 配置
sdc_clk	SDC CLK 的 GPIO 配置
sdc_cmd	SDC CMD 的 GPIO 配置
sdc_d3	SDC DATA3 的 GPIO 配置
sdc_d2	SDC DATA2 的 GPIO 配置
sdc_det	SDC DET 的 GPIO 配置
sdc_use_wp	SDC 写保护配置：1 使用，0 不用
sdc_wp	SDC WP 的 GPIO 配置
sdc_isio	是否是 sdio card, 0:不是，1: 是
sdc_regulator	SDC0 默认给 TF 卡使用，因此 mmc0 配置中此处不用配置
sdc_power_supply	供电 regulator 名字

举例说明：

```

sdc_used          = 1
sdc_detmode       = 1
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:PF06<0><1><2><default>
sdc_use_wp        = 0
sdc_wp            =
sdc_isio          = 0
sdc_regulator     = "none"
sdc_power_supply  = "none"

```

14.2.[mmc1_para]

配置项	配置项含义
sdc_used	SDC 使用控制：1 使用，0 不用
sdc_detmode	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	位宽：1-1bit，4-4bit
sdc_d1	SDC DATA1 的 GPIO 配置
sdc_d0	SDC DATA0 的 GPIO 配置
sdc_clk	SDC CLK 的 GPIO 配置
sdc_cmd	SDC CMD 的 GPIO 配置
sdc_d3	SDC DATA3 的 GPIO 配置
sdc_d2	SDC DATA2 的 GPIO 配置
sdc_det	SDC DET 的 GPIO 配置
sdc_use_wp	SDC 写保护配置：1 使用，0 不用
sdc_wp	SDC WP 的 GPIO 配置
sdc_isio	是否是 sdio card, 0:不是，1: 是
sdc_regulator	SDC0 默认给 TF 卡使用，因此 mmc0 配置中此处不用配置
sdc_power_supply	供电 regulator 名字

举例说明：

```

sdc_used           = 1
sdc_detmode        = 4
sdc_buswidth       = 4
sdc_clk            = port:PG00<2><1><3><default>
sdc_cmd            = port:PG01<2><1><3><default>
sdc_d0             = port:PG02<2><1><3><default>
sdc_d1             = port:PG03<2><1><3><default>
sdc_d2             = port:PG04<2><1><3><default>
sdc_d3             = port:PG05<2><1><3><default>
sdc_det            =
sdc_use_wp         = 0
sdc_wp             =
sdc_isio           = 1
sdc_regulator       = "none"
sdc_power_supply    = "none"
sdc_2xmode         = 1
sdc_ddrmode        = 1

```

14.3.[mmc2_para]

配置项	配置项含义
sdc_used	SDC 使用控制：1 使用，0 不用
sdc_detmode	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	位宽：1-1bit，4-4bit
sdc_d1	SDC DATA1 的 GPIO 配置
sdc_d0	SDC DATA0 的 GPIO 配置
sdc_clk	SDC CLK 的 GPIO 配置
sdc_cmd	SDC CMD 的 GPIO 配置
sdc_d3	SDC DATA3 的 GPIO 配置
sdc_d2	SDC DATA2 的 GPIO 配置

sdc_d4	SDC DATA4GPIO 配置
sdc_d5	SDC DATA5 GPIO 配置
sdc_d6	SDC DATA6 GPIO 配置
sdc_d7	SDC DATA7 GPIO 配置
sdc_det	SDC DET 的 GPIO 配置
sdc_use_wp	SDC 写保护配置: 1 使用, 0 不用
sdc_wp	SDC WP 的 GPIO 配置
sdc_isio	是否是 sdio card, 0:不是, 1: 是
sdc_regulator	SDC0 默认给 TF 卡使用, 因此 mmc0 配置中此处不用配置
sdc_power_supply	供电 regulator 名字

配置举例:

```

sdc_used          = 0
sdc_detmode       = 3
sdc_buswidth      = 8
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_det           =
sdc_use_wp        = 0
sdc_wp            =
sdc_isio          = 0
sdc_regulator     = "none"
sdc_power_supply  = "none"
sdc_2xmode        = 1
sdc_ddrmode       = 1

```

15.SIM 卡

15.1.[smc_para]

配置项	配置项含义，盒子方案不使用，仅保留
smc_used	是否使用 SIM 卡，1：使用 0：不使用
smc_rst	SIM 卡复位 GPIO 设置
smc_vppen	SIM 卡电源使能设置
smc_vppp	SIM 卡电源设置
smc_det	SIM 卡检测 GPIO 设置
smc_vccen	SIM 卡电源使能 GPIO 设置
smc_sck	SIM 卡是在 GPIO 设置
smc_sda	SIM 卡数据 GPIO 设置

配置举例：

smc_used

smc_rst

smc_vppen

smc_vppp

smc_det

smc_vccen

smc_sck

smc_sda

= 0

= port:PA09<2><default><default><default>

= port:PA20<3><default><default><default>

= port:PA21<3><default><default><default>

= port:PA10<2><default><default><default>

= port:PA06<2><default><default><default>

= port:PA07<2><default><default><default>

= port:PA08<2><default><default><default>

16.USB 控制器标志

16.1.[usbc0]

配置项	配置项含义
usb_used	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	USB 端口的使用情况。(xx=0/1/2) 0: device only 1: host only 2: OTG
usb_detect_type	USB 端口的检查方式。 0: 无检查方式 1: vbus/id 检查
usb_id_gpio	USB ID pin 脚配置。 具体请参考 gpio 配置说明。
usb_det_vbus_gpio	USB DET_VBUS pin 脚配置。 如果 GPIO 提供 pin，请参考 gpio 配置说明; 如果的 AXP 提供 pin，则配置为: "axp_ctrl"。
usb_drv_vbus_gpio	USB DRY_VBUS pin 脚配置; 具体请参考 gpio 配置说明。
usb_restrict_gpio	USB 限流控制 pin 脚配置; 具体请参考 gpio 配置说明。
usb_host_init_state	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_restrict_gpio	usb 限流控制 pin 配置
usb_restric_flag	Usb 限流标志位 0: 不使能限流功能 1: 使能限流功能
usb_restric_voltage	限流开启的条件 电压值小于设置值，则开启限流
usb_restric_capacity	限流开启的条件 电量值小于设置值，则开启限流
usb_regulator_io	usb 电源控制 pin 配置
usb_regulator_vol	usb 控制器电压配置
usb_not_suspend	是否启用休眠唤醒功能，1: 启用; 0: 不启用

配置举例:

```

usb_used           = 1
usb_port_type      = 1
usb_detect_type    = 0
usb_id_gpio        =
usb_det_vbus_gpio  =
usb_drv_vbus_gpio  = port:PL02<1><0><default><0>
usb_host_init_state = 1
usb_restrict_gpio   =
usb_restric_flag    = 0
usb_restric_voltage = 3550000

```



```

usb_restric_capacity  = 5
usb_regulator_io      = "nocare"
usb_regulator_vol     = 0
usb_not_suspend       = 0

```

16.2.[usbc1]

配置项	配置项含义
usb_used	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_det_vbus_gpio	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	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	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	Usb 限流标志位 0：表不设限流，1 开启限流
usb_regulator_io	usb 电源控制 pin 配置
usb_regulator_vol	usb 控制器电压配置
usb_not_suspend	是否启用休眠唤醒功能，1：启用；0：不启用

配置举例：

```

usb_used          = 1
usb_drv_vbus_gpio = port:PL03<1><0><default><0>
usb_restrict_gpio =
usb_host_init_state = 1
usb_restric_flag   = 0
usb_regulator_io   = "nocare"
usb_regulator_vol  = 0
usb_not_suspend    = 0

```

16.3.[usbc2]

配置项	配置项含义
usb_used	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_det_vbus_gpio	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	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	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	Usb 限流标志位 0: 表不设限流, 1 开启限流
usb_regulator_io	usb 电源控制 pin 配置
usb_regulator_vol	usb 控制器电压配置
usb_not_suspend	是否启用休眠唤醒功能, 1: 启用; 0: 不启用

配置举例:

```
usb_used          = 0
usb_drv_vbus_gpio =
usb_restrict_gpio =
usb_host_init_state = 1
usb_restric_flag  = 0
usb_regulator_io  = "nocare"
usb_regulator_vol  = 0
usb_not_suspend   = 0
```

Confidential

17.USB DEVICE

17.1.[usb_feature]

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

配置举例：

```

vendor_id           = 0x18D1
mass_storage_id     = 0x0001
adb_id              = 0x0002
manufacturer_name   = "USB Developer"
product_name         = "Android"
serial_number        = "20080411"

```

17.2.[msc_feature]

配置项	配置项含义
vendor_name	U 盘 厂商名
product_name	U 盘产品名
release	发布版本
luns	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：

```

vendor_name         = "USB 2.0"
product_name         = "USB Flash Driver"
release              = 100
luns                  = 3

```

17.3.[serial_feature]

配置项	配置项含义
serial_unique	USB 序列号是否配置唯一的，1：是 0：否

配置举例：

```

serial_unique        = 0

```

18.WIFI

18.1.[module_para]

配置项	配置项含义
module_power0	模组使用电源配置
module_power0_vol	模组使用电源电压配置
module_power1	模组使用电源配置
module_power1_vol	模组使用电源电压配置
module_power2	模组使用电源配置
module_power2_vol	模组使用电源电压配置
module_power3	模组使用电源配置
module_power3_vol	模组使用电源电压配置
chip_en	模组使能脚

配置举例：

```

module_power0      = port:PL07<1><default><default><1>
module_power0_vol  = 0
module_power1      = port:PG13<1><default><default><0>
module_power1_vol  = 1
module_power2      =
module_power2_vol  =
module_power3      =
module_power3_vol  =
chip_en            =
cpo_use_apclk      = "losc_out"

```

18.2.[wifi_para]

配置项	配置项含义
wifi_used	是否要使用 wifi
wifi_sdc_id	sdio wifi 选用的是哪个 sdc 作为接口
wifi_usbc_id	usb wifi 选用的是哪个 usb 作为接口
wifi_usbc_type	usb 接口类型，1 为 ehci，0 为 ohci
wl_reg_on	wifi 模块使能脚
wl_host_wake	wifi 唤醒主控脚
wl_host_wake_invert	wifi 中断脚和主控之间是否有反向器，0：无，1：有

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

配置举例：

```

wifi_used          = 1
wifi_sdc_id        = 1
wifi_usbc_id       = 3
wifi_usbc_type     = 1
wl_reg_on          = port:PL07<1><default><default><1>

```

```
wl_host_wake      =  
wl_host_wake_invert = 0
```

Confidential

19.蓝牙

19.1.[bt_para]

配置项	配置项含义
bt_used	BLUETOOTH 使用控制：1 使用，0 不用
bt_uart_id	BLUETOOTH 使用的 UART 控制器号
bt_uart_baud	蓝牙串口波特率
bt_rst_n	蓝牙模块使能脚
bt_wake	主控唤醒蓝牙脚
bt_host_wake	蓝牙唤醒主控脚
bt_host_wake_invert	蓝牙中断脚和主控之间是否有反向器，0：无，1：有

配置举例：

bt_used = 1
bt_uart_id = 1
bt_uart_baud = 1500000
bt_rst_n = port:PA13<1><default><default><0>
bt_wake = port:PG12<1><default><default><0>
bt_host_wake = port:PL06<0><default><default><0>
bt_host_wake_invert = 0

20. 音频配置 (audio)

20.1.[pcm0]

配置项	配置项含义
daudio_used	是否使用该接口，默认要配置为 1 1: 使用 0: 不使用
daudio_master	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 (CODEC as master, i2s as slave) 3: SND_SOC_DAIFMT_CBM_CFS(codec clk master & frame slave) not use 4: SND_SOC_DAIFMT_CBS_CFS(codec clk & FRM slave) use (CODEC as slave, i2s as master)
daudio_select	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	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 justified format). 3:SND_SOC_DAIFMT_LEFT_J(left justified 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	bclk 时钟信号和帧信号要不要翻转，一般都为 normal 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	support 128fs/192fs/256fs/384fs/512fs/768fs (使用默认值)
sample_resolution	该参数指采样精度，通常默认 16 16bits/20bits/24bits
slot_width_select	Slot 宽度 (使用默认值 16)
pcm_lrck_period	单声道 bclk 个数/lrck 个数 (使用默认值 32)
pcm_lrckr_period	Lrckr 参数: 使用默认值
msb_lsb_first	0: msb first; 1: lsb first (使用默认值)
sign_extend	Pcm 格式参数: 选用默认
slot_index	Pcm 格式参数: 选用默认
frame_width	Pcm 格式参数: 选用默认
tx_data_mode	Pcm 格式参数: 选用默认
rx_data_mode	Pcm 格式参数: 选用默认
i2s_mclk	i2s mclk GPIO 配置

i2s_bclk	i2s bclk GPIO 配置
i2s_lrclk	i2s lrclk GPIO 配置
i2s_dout0	i2s dout0 GPIO 配置
i2s_dout1	i2s dout1 GPIO 配置
i2s_dout2	i2s dout2 GPIO 配置
i2s_dout3	i2s dout3 GPIO 配置
i2s_din	i2s din GPIO 配置

配置举例：

```

audio_used          = 0
audio_master        = 4
audio_select        = 1
audio_format        = 1
signal_inversion    = 1
mclk_fs             = 128
sample_resolution   = 16
slot_width_select   = 32
;pcm_sync_period    = 256
pcm_lrck_period     = 32
pcm_lrckr_period    = 1
msb_lsb_first       = 0
sign_extend         = 0
slot_index          = 0
slot_width          = 32
frame_width         = 0
tx_data_mode        = 0
rx_data_mode        = 0
i2s_mclk            = port:PA18<2><1><default><default>
i2s_bclk            = port:PA19<2><1><default><default>
i2s_dout0           = port:PA20<2><1><default><default>
i2s_din             = port:PA21<2><1><default><default>

```

20.2.[audio0]

配置项	配置项含义
audio_used	是否使用 H3 模拟音频输入输出 1：使用，0：不使用
lineout_vol	Lineout 默认音量设置，最大是 0x1f
cap_vol	Mic 的增益，最大是 0x7
audio_hp_ldo	耳机口电压设置, H3 设置为 none，不需改动
adcagc_used	录音音效增强，1：开启，0：不开启
adcdrc_used	录音动态音效调节，1：开启，0：不开启
dacdrc_used	播放动态音效调节，1：开启，0：不开启
adchpf_used	录音通路高通滤波开启，1：开启，0：不开启
dachpf_used	播放通路高通滤波开启，1：开启，0：不开启
audio_pa_ctrl	外部功放使能脚

配置举例：

```

audio_used          = 1

```



```

lineout_vol      = 0x1f
cap_vol          = 0x5
audio_hp_ldo     = none
adcagc_used      = 0
adcdrc_used      = 0
dacdrc_used      = 0
adchpf_used      = 0
dachpf_used      = 0
audio_pa_ctrl    = port:PA16<1><default><default><0>

```

20.3.[spdif0]

配置项	配置项含义
spdif_used	是否开启 spdif, 1: 开启, 0: 不开启
spdif_dout	Spdif 输出管脚

配置举例:

```

spdif_used      = 1
spdif_dout      = port:PA17<2><1><default><default>

```

20.4.[audiohub]

配置项	配置项含义
hub_used	是否启用该模块, 0: 不启用; 1: 启用
codec_used	是否使用 codec 输出
spdif_used	是否使能 spdif 输出
hdmi_used	是否启用 hdmi audio 输出

配置举例:

```

hub_used        = 0
codec_used       = 1
spdif_used       = 1
hdmi_used        = 1

```

21. 红外(ir)

21.1.[s_cir0]

配置项	配置项含义
ir_used	是否使用该模块，1：使用，0：不使用
ir_rx	红外接收的 GPIO 配置
ir_power_key_codeN	第 N 个红外遥控器 powerkey 对应的按键值（N 从 0~15）
ir_addr_codeN	第 N 个红外遥控器地址码（N 从 0~15）

配置举例：

```

ir_used                = 1
ir_rx                  = port:PL11<2><1><default><default>
ir_power_key_code0     = 0x57
ir_addr_code0          = 0x9f00
ir_power_key_code1     = 0x1a
ir_addr_code1          = 0xfb04
ir_power_key_code2     = 0x14
ir_addr_code2          = 0x7F80
ir_power_key_code3     = 0x15
ir_addr_code3          = 0x7F80
ir_power_key_code4     = 0x0b
ir_addr_code4          = 0xF708
ir_power_key_code5     = 0x03
ir_addr_code5          = 0x00EF
ir_power_key_code6     = 0x9f
ir_addr_code6          = 0x4CB3
ir_power_key_code7     = 0x0a
ir_addr_code7          = 0x7748
ir_power_key_code8     = 0x45
ir_addr_code8          = 0xbd02
ir_power_key_code9     = 0x4d
ir_addr_code9          = 0xde21
ir_power_key_code10    = 0x18
ir_addr_code10         = 0xfe01
ir_power_key_code11    = 0x57
ir_addr_code11         = 0xff00
ir_power_key_code12    = 0x4d
ir_addr_code12         = 0xff40
ir_power_key_code13    = 0x88
ir_addr_code13         = 0xdd22

```

多遥控器支持配置详细说明参见文档《H3 多遥控器使用说明书 V1.0》。

21.2.[cir]

配置项	配置项含义
ir_used	是否使用该模块，1：使用，0：不使用
ir_tx	红外发射的 GPIO 配置

配置举例：

ir_used = 0

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

Confidential

22.Vf 表设置

注意：vf 表（电压频率对应表）关乎系统稳定性，请勿私自修改！

22.1.[dvfs_table]

配置项	配置项含义
pmuic_type	cpu 电源 ic 控制接口类型： 0：无电源 ic ；1：gpio 控制；2：i2c 控制-带电源 IC
pmu_gpio0	控制电源的 pin 配置
pmu_level0	0~9999: voltage(mV), 10000~90000:gpio0 state
pmu_level1	0~9999: voltage(mV), 10000~90000:gpio0 state
max_freq	最大频率，最高可支持到 1500MHz，需要调压芯片的支持，无调压芯片默认支持到 1200MHz。
min_freq	最小频率
boot_freq	系统启动阶段 cpu 运行频率, 不要改动
LV_count	vf 表的级数
LVn_freq	LVx_volt 对应的最大频率(n 表示级数)
LVn_volt	第 n 级的电压

配置举例：

```

pmuic_type          = 0 //H3 原型机硬件默认带调压芯片，如果采用分立电源则配置为 0
pmu_gpio0           = port:PL06<1><1><2><1>
pmu_level0          = 11300
pmu_level1          = 01100
max_freq            = 1200000000
min_freq            = 480000000
boot_freq           = 1008000000
LV_count            = 8
LV1_freq            = 1200000000
LV1_volt            = 1300
LV2_freq            = 1008000000
LV2_volt            = 1200
LV3_freq            = 0
LV3_volt            = 1100
LV4_freq            = 0
LV4_volt            = 1100
LV5_freq            = 0
LV5_volt            = 1100
LV6_freq            = 0
LV6_volt            = 1100
LV7_freq            = 0
LV7_volt            = 1100
LV8_freq            = 0
LV8_volt            = 1100

```

22.2.[gpu_dvfs_table]

配置项	配置项含义
G_dvfs_enable	是否启用 gpu 调频
G_LV_count	频率档位总数
G_LVn_freq	第 n 档的频率
G_LVn_volt	第 n 档的电压

配置举例：

G_LV0_freq = 312000000
G_LV0_volt = 1200000

G_LV1_freq = 384000000
G_LV1_volt = 1200000

G_LV2_freq = 456000000
G_LV2_volt = 1200000

根据温控策略和场景自动调整频率。

confidential

23.Board Vendor Id

配置项	配置项含义
vid_used	是否启用该模块： 0：不启用，1：启用
vid_count	主板硬件 ID 编码号 的 GPIO 构成个数
vid_bit_0	主板硬件 ID 编码号最低位 bit0 对应的 GPIO 配置
vid_bit_1	主板硬件 ID 编码号 bit1 位对应的 GPIO 配置
vid_bit_2	主板硬件 ID 编码号 bit3 位对应的 GPIO 配置
vid_bit_3	主板硬件 ID 编码号 bit3 位对应的 GPIO 配置

配置举例：

```

vid_used          = 0
vid_count         = 0
vid_bit_0         = port:PD16<0><1><default><default>
vid_bit_1         = port:PD15<0><1><default><default>
vid_bit_2         = port:PD14<0><1><default><default>
vid_bit_3         = port:PD13<0><1><default><default>
vid_bit_4         = port:PD12<0><1><default><default>
vid_bit_5         = port:PD17<0><1><default><default>

```

V1.1 SDK 中不要打开此处的配置.没有任何实际作用.如果随意配置，会影响系统稳定性.

24.Transport Stream Controller

24.1.[ts0]

配置项	配置项含义
tsc_used	是否启用该模块： 0：不启用，1：启用
tsc_clk	tsc 模块 clk 信号的 pin 配置
tsc_err	tsc 模块 err 信号的 pin 配置
tsc_sync	tsc 模块 sync 信号的 pin 配置
tsc_dvld	tsc 模块 dvld 信号的 pin 配置
tsc_d0	tsc 模块数据 d0 信号的 pin 配置
tsc_d1	tsc 模块数据 d1 信号的 pin 配置
tsc_d2	tsc 模块数据 d2 信号的 pin 配置
tsc_d3	tsc 模块数据 d3 信号的 pin 配置
tsc_d4	tsc 模块数据 d4 信号的 pin 配置
tsc_d5	tsc 模块数据 d5 信号的 pin 配置
tsc_d6	tsc 模块数据 d6 信号的 pin 配置
tsc_d7	tsc 模块数据 d7 信号的 pin 配置

配置举例：

```

tsc_used          = 1
tsc_clk           = port:PE00<3><default><default><default>
tsc_err           = port:PE01<3><default><default><default>
tsc_sync          = port:PE02<3><default><default><default>
tsc_dvld          = port:PE03<3><default><default><default>
tsc_d0            = port:PE04<3><default><default><default>
tsc_d1            = port:PE05<3><default><default><default>
tsc_d2            = port:PE06<3><default><default><default>
tsc_d3            = port:PE07<3><default><default><default>
tsc_d4            = port:PE08<3><default><default><default>
tsc_d5            = port:PE09<3><default><default><default>
tsc_d6            = port:PE10<3><default><default><default>
tsc_d7            = port:PE11<3><default><default><default>

```

25.GPIO 模拟电源按键

25.1.[gpio_power_key]

配置项	配置项含义
key_used	是否使用该模块，1：使用，0：不使用
key_io	模拟电源按键的 GPIO 配置

配置举例：

```
key_used      = 0
key_io        = port:PL04<6><default><default><0>
```

Confidential

26.CPUS

26.1.[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_uart_used      = 0
s_uart_tx       = port:PL02<2><default><default><default>
s_uart_rx       = port:PL03<2><default><default><default>
```

26.2.[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_rsb_used      = 1
s_rsb_sck       = port:PL00<2><1><2><default>
s_rsb_sda       = port:PL01<2><1><2><default>
```

26.3.[s_jtag0]

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

配置举例：

```
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>
```

26.4.[s_powchk]

注意：此配置影响待机唤醒，请勿随意修改！

配置项	配置项含义
s_powchk_used	是否使能 standby 下的电源监测： bit31：使能电源状态更新

	bit1: 使能电源异常唤醒系统功能 bit0: 使能唤醒, 当功耗异常
s_power_reg	各路电源的开关状态
s_system_power	standby 状态下的最大功耗, 单位 mW

配置举例:

s_powchk_used = 0x80000000
s_power_reg = 0x00000000
s_system_power = 50

Confidential

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.

Confidential