sys_config 配置说明文档

1.0

2017.09.07

文档履历

版本号	日期	制/修订人	内容描述
1.0	2017.09.07	Allwinner	

目录

1. 系统 (SYSTEM)

1.1 [product]

配置项	配置项含义
version	sdk 版本号
machine	sdk 代号

配置举例:

```
version    = "100"
machine    = "dvb"
```

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（仅对量产有效，OTA 无效）
next_work	USB 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:

```
eraseflag    = 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（defualt）:auto scan
burn_key	启动时是否需要烧 key 0：不烧 1：烧
dragonboard_test	是否编译支持卡启动的 dragonboard 固件。1：是 0：否

配置举例：

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

1.4 [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_gpio0 =

1.5 [pm_para]

配置项	配置项含义
standby_mode	standby 模式 1：super standby 0：normal standby

配置举例：

standby_mode = 1

1.6 [recovery_para]

配置项	配置项含义
recovery_para_used	是否开启一键 recovery 功能，1：开启 0：禁用
mode	1：一键进入 OTA 2：一键进入 recovery 模式
recovery_key	按键 GPIO 配置

配置举例：

recovery_para_used = 1
mode = 2
recovery_key = port:PL10<0><default><default><default>

1.7 [boot_init_gpio]

配置项	配置项含义
boot_init_gpio_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:PL10<1><default><default><1>

一般用于系统启动 LED GPIO 初始化，必须以 gpio0、gpio1、gpio2 递增去命名，才可以申请到对应的 gpio

1.8 [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:PF0<2><1><2><default>
sdc_d0 = port:PF1<2><1><2><default>
sdc_clk = port:PF2<2><1><2><default>
sdc_cmd = port:PF3<2><1><2><default>
sdc_d3 = port:PF4<2><1><2><default>
sdc_d2 = port:PF5<2><1><2><default>

1.9 [card2_boot_para]

配置项	配置项含义
card_ctrl	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_ds	ds 信号的 GPIO 配置
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
sdc_d4	sdc 卡数据 4 线信号的 GPIO 配置
sdc_d5	sdc 卡数据 5 线信号的 GPIO 配置
sdc_d6	sdc 卡数据 6 线信号的 GPIO 配置
sdc_d7	sdc 卡数据 7 线信号的 GPIO 配置
sdc_emmc_rst	emmc_rst 信号的 GPIO 配置
sdc_ex_dly_used	ex_dly_used 信号的 GPIO 配置

配置举例：

```
card_ctrl      = 2
card_high_speed = 1
card_line      = 8
sdc_ds         = port:PC1<3><1><3><default>
sdc_clk        = port:PC4<3><1><3><default>
sdc_cmd        = port:PC5<3><1><3><default>
sdc_d0         = port:PC6<3><1><3><default>
sdc_d1         = port:PC7<3><1><3><default>
sdc_d2         = port:PC8<3><1><2><default>
sdc_d3         = port:PC9<3><1><3><default>
sdc_d4         = port:PC10<3><1><3><default>
```

```
sdc_d5          = port:PC11<3><1><3><default>
sdc_d6          = port:PC12<3><1><3><default>
sdc_d7          = port:PC13<3><1><3><default>
sdc_emmc_rst    = port:PC14<3><1><3><default>
sdc_ex_dly_used = 2
```

1.10 [twi_para]

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

配置举例:

```
twi_port        = 0
twi_scl         = port:PH0<2><default><default><default>
twi_sda         = port:PH1<2><default><default><default>
```

1.11 [uart_para]

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

配置举例:

```
uart_debug_port = 3
```

uart_debug_tx = port:PH05<2><1><default><default>
uart_debug_rx = port:PH6<2><1><default><default>

1.12 [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 (CLK) 的 GPIO 配置
jtag_do	测试数据输出 (TDO) 的 GPIO 配置
jtag_di	测试数据输出 (TDI) 的 GPIO 配置

配置举例:

jtag_enable = 1
jtag_ms = port:PH9<4><default><default><default>
jtag_ck = port:PH10<4><default><default><default>
jtag_do = port:PH11<4><default><default><default>
jtag_di = port:PH12<4><default><default><default>

1.13 [clock]

配置项	配置项含义
pll4	pll4 时钟频率
pll6	pll6 时钟频率
pll8	pll8 时钟频率
pll9	pll9 时钟频率
pll10	pll10 时钟频率

配置举例:

pll4 = 300
pll6 = 600
pll8 = 360
pll9 = 297
pll10 = 264

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
dram_type	DRAM 类型：2 为 DDR2 3 为 DDR3
dram_zq	DRAM 控制器内部参数，由源厂调节，请勿修改
dram_odt_en	ODT 是否需要使能，为了省电，一般设置为 0
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_xxx	由源厂调节，请勿修改

配置举例：

```
dram_clk      = 792
dram_type     = 7
dram_zq       = 0x3B3BFB
dram_odt_en   = 0x1
dram_para1    = 0x000010f4
dram_para2    = 0x1000
dram_mr0      = 0x1c70
dram_mr1      = 0x40
dram_mr2      = 0x18
dram_mr3      = 0x2
dram_tpr0     = 0x0047194f
dram_tpr1     = 0x01b1a94b
dram_tpr2     = 0x00061043
dram_tpr3     = 0xB4787896
dram_tpr4     = 0x0
dram_tpr5     = 0x0
dram_tpr6     = 100
dram_tpr7     = 0x0F220F1C
```

dram_tpr8 = 0x0
dram_tpr9 = 0x0
dram_tpr10 = 0x0000
dram_tpr11 = 0x00000000
dram_tpr12 = 0x00000000
dram_tpr13 = 0x04000284

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：1：enable0：disable
phy-mode	PHY 接口类型，``MII" 或者 ``RGMII"
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 接收错误提示
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_txclk	Gmac MII 接口发送时钟
gmac_txen	Gmac 发送使能 GPIO 配置
gmac_clkin	Gmac 时钟输入配置
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
gmac_power2	Gmac 供电配置
gmac_power3	Gmac 供电配置
tx-delay	调整 tx clk 相位，无需设置，如需设置请联系原厂
rx-delay	调整 rx clk 相位，无需设置，如需设置请联系原厂

配置举例：

```
gmac0_used      = 1
```

```
phy-mode        = "rgmii"
gmac_rxd3       = port:PA00<2><default><default><default>
gmac_rxd2       = port:PA01<2><default><default><default>
gmac_rxd1       = port:PA02<2><default><default><default>
gmac_rxd0       = port:PA03<2><default><default><default>
gmac_rxclk      = port:PA04<2><default><default><default>
gmac_rxdv       = port:PA05<2><default><default><default>
gmac_rxerr      = port:PA06<2><default><default><default>
gmac_txd3       = port:PA07<2><default><default><default>
gmac_txd2       = port:PA08<2><default><default><default>
gmac_txd1       = port:PA09<2><default><default><default>
gmac_txd0       = port:PA10<2><default><default><default>
gmac_txclk      = port:PA11<2><default><default><default>
gmac_txen       = port:PA12<2><default><default><default>
gmac_clkin      = port:PA13<2><default><default><default>
gmac_mdc        = port:PA14<2><default><default><default>
gmac_mdio       = port:PA15<2><default><default><default>
gmac_power0     ="vcc-ephy"
gmac_power1     =""
gmac_power2     =""
tx-delay        = 7
rx-delay        = 31
use_ephy25m     = 1
```

4. I2C 总线

4.1 [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：

```
twi0_used       = 1
twi0_scl        = port:PH0<2><default><default><default>
twi0_sda        = port:PH1<2><default><default><default>
```

4.2 [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：

```
twi1_used       = 1
twi1_scl        = port:PH2<2><default><default><default>
twi1_sda        = port:PH3<2><default><default><default>
```

4.3 [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：

twi2_used = 0
twi2_scl = port:PI01<5><default><default><default>
twi2_sda = port:PI02<5><default><default><default>

4.4 [twi3]

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

配置举例：

twi3_used = 1
twi3_scl = port:PI8<5><default><default><default>
twi3_sda = port:PI9<5><default><default><default>

4.5 [twi0/twi_board0]

配置项	配置项含义
compatible	TWI 从设备名称
reg	TWI 从设备地址

5. UART

5.1 [uart0]

配置项	配置项含义
uart0_used	UART 使用控制：1 使用，0 不用
uart0_port	UART 端口号
uart0_type	2：2 线模式;4：4 线模式;8：8 线模式。
uart0_tx	UART TX 的 GPIO 配置
uart0_rx	UART RX 的 GPIO 配置

配置举例：

```
uart0_used      = 0
uart_port       = 0
uart_type       = 2
uart_tx         = port:PB09<2><1><default><default>
uart_rx         = port:PB10<2><1><default><default>
```

5.2 [uart1]

配置项	配置项含义
uart1_used	UART 使用控制：1 使用，0 不用
uart1_port	UART 端口号

配置项	配置项含义
uart1_type	2：2 线模式;4：4 线模式;8：8 线模式。
uart1_tx	UART TX 的 GPIO 配置
uart1_rx	UART RX 的 GPIO 配置
uart1_rts	GPIO 配置
uart1_cts	GPIO 配置

配置举例：

```
uart1_used      = 1
uart1_port      = 1
uart1_type      = 4
uart1_tx        = port:PG6<2><1><default><default>
uart1_rx        = port:PG7<2><1><default><default>
uart1_rts       = port:PG8<2><1><default><default>
uart1_cts       = port:PG9<2><1><default><default>
```

5.3 [uart1_suspend]

配置项	配置项含义
uart1_tx	UART TX 的 GPIO 配置
uart1_rx	UART RX 的 GPIO 配置
uart1_rts	GPIO 配置
uart1_cts	GPIO 配置

配置举例：

```
uart1_tx        = port:PG6<7><1><default><default>
uart1_rx        = port:PG7<7><1><default><default>
uart1_rts       = port:PG8<7><1><default><default>
uart1_cts       = port:PG9<7><1><default><default>
```

5.4 [uart3]

配置项	配置项含义
uart3_used	UART 使用控制：1 使用，0 不用
uart3_port	UART 端口号
uart3_type	2：2 线模式;4：4 线模式;8：8 线模式。
uart3_tx	UART TX 的 GPIO 配置
uart3_rx	UART RX 的 GPIO 配置
uart3_rts	GPIO 配置
uart3_cts	GPIO 配置

配置举例：

```
[uart3]
uart3_used      = 1
uart3_port      = 0
uart3_type      = 2
uart3_tx        = port:PH05<2><1><default><default>
uart3_rx        = port:PH06<2><1><default><default>
```

5.5 [uart3_suspend]

配置项	配置项含义
uart3_tx	UART TX 的 GPIO 配置
uart3_rx	UART RX 的 GPIO 配置

配置举例：

```
uart3_tx        = port:PH05<7><1><default><default>
uart3_rx        = port:PH06<7><1><default><default>
```

6. SPI 总线

6.1 [spi0]

配置项	配置项含义
spi0_used	SPI 使用控制：1 使用，0 不用
spi0_cs_number	片选
spi0_cs_bitmap	由于 SPI 控制器支持多个 CS，这一个参数表示 CS 的掩码；
spi0_cs0	SPI CS0 的 GPIO 配置
spi0_cs1	SPI CS1 的 GPIO 配置
spi0_sclk	SPI CLK 的 GPIO 配置
spi0_mosi	SPI MOSI 的 GPIO 配置
spi0_miso	SPI MISO 的 GPIO 配置
spi0_hold	SPI HOLD 的 GPIO 配置
spi0_wp	SPI WP 的 GPIO 配置

配置举例：

```
spi0_used      = 0
spi0_cs_number  = 2
spi0_cs_bitmap  = 3
spi0_cs0       = port:PC15<4><1><default><default>
spi0_cs1       = port:PC16<4><1><default><default>
spi0_sclk      = port:PC0<4><default><default><default>
spi0_mosi       = port:PC2<4><default><default><default>
spi0_miso       = port:PC3<4><default><default><default>
spi0_hold       = port:PC6<4><default><default><default>
spi0_wp        = port:PC7<4><default><default><default>
```

6.2 [spi1]

配置项	配置项含义
spi1_used	SPI 使用控制：1 使用，0 不用
spi1_cs_number	片选
spi1_cs_bitmap	由于 SPI 控制器支持多个 CS，这一个参数表示 CS 的掩码；
spi1_cs0	SPI CS0 的 GPIO 配置
spi1_cs1	SPI CS1 的 GPIO 配置
spi1_sclk	SPI CLK 的 GPIO 配置
spi1_mosi	SPI MOSI 的 GPIO 配置
spi1_miso	SPI MISO 的 GPIO 配置

配置举例：

```
spi1_used      = 0
spi1_cs_number  = 2
spi1_cs_bitmap  = 3
spi1_cs0       = port:PH7<3><1><default><default>
 spi1_cs1       = port:PH8<3><1><default><default>
spi1_sclk      = port:PH15<3><default><default><default>
spi1_mosi       = port:PH14<3><default><default><default>
spi1_miso       = port:PH6<3><default><default><default>
```

6.3 [spi2]

配置项	配置项含义
spi2_used	SPI 使用控制：1 使用，0 不用
spi2_cs_number	片选
spi2_cs_bitmap	由于 SPI 控制器支持多个 CS，这一个参数表示 CS 的掩码；
spi2_cs0	SPI CS0 的 GPIO 配置
spi2_sclk	SPI CLK 的 GPIO 配置
spi2_mosi	SPI MOSI 的 GPIO 配置
spi2_miso	SPI MISO 的 GPIO 配置

配置举例:

```
spi2_used      = 0
spi2_cs_number  = 1
spi2_cs_bitmap  = 1
spi2_cs0       = port:PI4<4><1><default><default>
spi2_sclk      = port:PI1<4><default><default><default>
spi2_mosi      = port:PI2<4><default><default><default>
spi2_miso      = port:PI3<4><default><default><default>
```

[spi3]

配置项	配置项含义
spi3_used	SPI 使用控制：1 使用，0 不用
spi3_cs_number	片选
spi3_cs_bitmap	由于 SPI 控制器支持多个 CS，这一个参数表示 CS 的掩码；
spi3_cs0	SPI CS0 的 GPIO 配置
spi3_sclk	SPI CLK 的 GPIO 配置
spi3_mosi	SPI MOSI 的 GPIO 配置
spi3_miso	SPI MISO 的 GPIO 配置

配置举例:

```
spi1_used      = 0
spi1_cs_number  = 1
spi1_cs_bitmap  = 1
spi1_cs0       = port:PI11<4><1><default><default>
spi1_sclk      = port:PI8<4><default><default><default>
spi1_mosi      = port:PI9<4><default><default><default>
spi1_miso      = port:PI10<4><default><default><default>
```

6.4 [spi0/spi_board0]

配置项	配置项含义
compatible	Spi 设备名字
spi-max-frequency	最大传输速度（HZ）
reg Spi	从设备地址
spi-cpha	可选，spi 时钟相位
spi-cpol	可选，spi 时钟极性
spi-cs-high	片选信号，高电平有效

配置举例：

```
compatible          ="m25p80"
spi-max-frequency     = 1000000
reg                 = 0
spi-rx-bus-width      = 1
spi-tx-bus-width      = 1
```

7. ADC 配置

7.1 [gpadc]

配置项	配置项含义
gpadc_used	是否使用gpadc，1：使用；0：不使用
channel_num	平台支持度的最大gpadc 通道数
channel_select	通道使能选择， channel0:0x01； channel1:0x02； channel2:0x04； channel3:0x08
channel_compare_select	数据低比较功能通道选择， channel0:0x01； channel1:0x02； channel2:0x04； channel3:0x08
channel_cld_select	小数据中断功能使能通道选择； channel0:0x01； channel1:0x02； channel2:0x04； channel3:0x08

配置项	配置项含义
channel_chd_select	比较功能的低数据通道使能选择； channel0:0x01； channel1:0x02;channel2:0x04； channel3:0x08
channel0_compare_low_data	通道 0 低于设定值时产生中断并采集数据，单位为 uv。
channel0_compare_high_data	通道 0 高于设定值时产生中断并采集数据，单位为 uv
channel1_compare_low_data	通道 1 低于设定值时产生中断并采集数据，单位为 uv
channel1_compare_high_data	通道 1 高于设定值时产生中断并采集数据，单位为 uv

配置项	配置项含义
channel2_compare_lowdata	通道 2 低于设定值时产生中断并采集数据，单位为 uv
channel2_compare_higdata	通道 2 高于设定值时产生中断并采集数据，单位为 uv
channel3_compare_lowdata	通道 3 低于设定值时产生中断并采集数据，单位为 uv
channel3_compare_higdata	通道 3 高于设定值时产生中断并采集数据，单位为 uv

配置举例：

gpadc_used = 1
channel_num = 4
channel_select = 0x05
channel_data_select = 0
channel_compare_select = 0x05

channel_cld_select = 0x01
channel_chd_select = 0x04
channel0_compare_lowdata = 1700000
channel0_compare_higdata = 1200000
channel1_compare_lowdata = 460000
channel1_compare_higdata = 1200000
channel2_compare_lowdata = 460000
channel2_compare_higdata = 200000
channel3_compare_lowdata = 460000
channel3_compare_higdata = 1200000

8. motor 配置

[motor_para]

配置项	配置项含义
motor_used	马达使用控制，1：使用；0：不使用
motor_shake	马达 power GPIO 配置

配置举例：

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

9. 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_ce1	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	nand1 数据总线信号的 GPIO 配置
nand0_d2	nand2 数据总线信号的 GPIO 配置
nand0_d3	nand3 数据总线信号的 GPIO 配置
nand0_d4	nand4 数据总线信号的 GPIO 配置
nand0_d5	nand5 数据总线信号的 GPIO 配置
nand0_d6	nand6 数据总线信号的 GPIO 配置
nand0_d7	nand7 数据总线信号的 GPIO 配置
nand0_ndqs	nand0 ddr 时钟信号的 GPIO 配置
nand0_ce2	nand0 片选 2 信号的 GPIO 配置
nand0_ce3	nand0 片选 3 信号的 GPIO 配置

配置举例:

```
nand0_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_ce0          = port:PC03<2><1><1><default>
nand0_nre          = port:PC04<2><1><1><default>
nand0_rb0          = port:PC05<2><0><1><default>
nand0_d0           = port:PC06<2><1><1><default>
nand0_d1           = port:PC07<2><1><1><default>
nand0_d2           = port:PC08<2><0><1><default>
nand0_d3           = port:PC09<2><0><1><default>
nand0_d4           = port:PC10<2><0><1><default>
nand0_d5           = port:PC11<2><0><1><default>
nand0_d6           = port:PC12<2><0><1><default>
nand0_d7           = port:PC13<2><0><1><default>
nand0_ndqs         = port:PC14<2><0><1><default>
nand0_ce1          = port:PC15<2><1><1><default>
nand0_rb1          = port:PC16<2><1><1><default>
nand0_regulator1        = "vcc-nand"
nand0_regulator2        = "none"
nand0_cache_level = 0x55aaaa55
nand0_flush_cache_num = 0x55aaaa55
nand0_capacity_level = 0x55aaaa55
nand0_id_number_ctl = 0x55aaaa55
nand0_print_level = 0x55aaaa55
nand0_p0 = 0x55aaaa55
nand0_p1 = 0x55aaaa55
nand0_p2 = 0x55aaaa55
nand0_p3 = 0x55aaaa55
```

10. 显示

10.1 [disp]

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 1：screen1
screen0_output_type	屏 0 输出类型 (0:none; 1:lcd; 2:tv; 3:hdmi; 4:vga)
screen0_output_mode	0:480i 1:576i 2:480p 3:576p 4:720p50 5:720p60 6:1080i50 7:1080i60 8:1080p249:1080p50 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	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:BGRA)
fb0_width	fb0 的宽度, 为 0 时将按照输出设备的分辨率
fb0_height	fb0 的高度，为 0 时将按照输出设备的分辨率
fb1_format	fb1 的格式 (0:ARGB 1:ABGR 2:RGBA 3:BGRA)
fb1_width	Fb1 的宽度, 为 0 时将按照输出设备的分辨率
fb1_height	Fb1 的高度，为 0 时将按照输出设备的分辨率
lcd0_backlight	lcd0 初始化背光，范围：0 ～ 256，默认：197
lcd1_backlight	lcd1 初始化背光，范围：0 ～ 256，默认：197
lcd0_bright	lcd0 的亮度值，0 ～ 100，默认 50
lcd0_contrast	lcd0 的对比度值，0 ～ 100，默认 50
lcd0_saturation	lcd0 的饱和度值，0 ～ 100，默认 57
lcd0_hue	lcd0 的色调值，0 ～ 100，默认 50
lcd1_bright	lcd1 的亮度值，0 ～ 100，默认 50
lcd1_contrast	lcd1 的对比度值，0 ～ 100，默认 50
lcd1_saturation	lcd1 的饱和度值，0 ～ 100，默认 57
lcd1_hue	lcd1 的色调值，0 ～ 100，默认 50

配置举例：

```
disp_init_enable        = 1
disp_mode               = 0
screen0_output_type     = 1
screen0_output_mode     = 4
screen1_output_type     = s1
screen1_output_mode     = 4
disp_rotation_used      = 0
degree0                 = 0
degree0_0               = 0
fb0_format              = 0
fb0_width               = 720
fb0_height              = 1280
fb1_format              = 0
fb1_width               = 0
fb1_height              = 0
lcd0_backlight          = 50
lcd1_backlight          = 50
lcd0_bright             = 50
lcd0_contrast           = 50
lcd0_saturation         = 57
lcd0_hue                = 50
lcd1_bright             = 50
lcd1_contrast           = 50
lcd1_saturation         = 57
lcd1_hue                = 50
```

11. HDMI

11.1 [hdmi]

配置项	配置项含义
hdmi_used	是否使用 hdmi。1：使用；0：不使用
hdmi_hdcp_enable	是否使能 hdcp
hdmi_cts_compatibility	cts 兼容性使能设置
hdmi_power	内核阶段 hdmi 电源配置
hdmi_io_regulator	hdmi 模组 io 使用哪一路 AXP 供电
hdmi_io_0	hdmi 模组 io0 GPIO 配置
hdmi_io_1	hdmi 模组 io1 GPIO 配置
hdmi_io_2	hdmi 模组 io2 GPIO 配置

配置举例：

```
hdmi_used               = 1
hdmi_hdcp_enable        = 0
hdmi_cts_compatibility  = 0
hdmi_power              = "vcc18-hdmi"
hdmi_io_regulator       = "vdd09-hdmi"
hdmi_io_0               = port:PH11<2><0><default><default>
hdmi_io_1               = port:PH12<2><0><default><default>
hdmi_io_2               = port:PH13<2><0><default><default>
```

11.2 [hdmi_suspend]

配置项	配置项含义
hdmi_io_0	hdmi io0 GPIO 配置

配置项	配置项含义
hdmi_io_1	hdmi io1 GPIO 配置
hdmi_io_2	hdmi io2 GPIO 配置

配置举例:

```
hdmi_io_0          = port:PH11<7><0><default><default>
hdmi_io_1          = port:PH12<7><0><default><default>
hdmi_io_2          = port:PH13<7><0><default><default>
```

12. PWM

12.1 [pwm0]

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        = 1
pwm_positive     = port:PD22<2><0><default><default>
```

12.2 [pwm0_suspend]

配置项	配置项含义
pwm_positive	PWM 输出 GPIO 配置

配置举例:

```
pwm_positive     = port:PG06<7><0><default><default>
```

12.3 [pwm1]

配置项	配置项含义
pwm_used	是否使用 PWM1

配置项	配置项含义
pwm_positive	PWM 输出 GPIO 配置

配置举例：

```
pwm_used        = 0
pwm_positive     = port:PG07<2><0><default><default>
```

12.4 [pwm1_suspend]

配置项	配置项含义
pwm_positive	PWM 输出 GPIO 配置

配置举例：

```
pwm_positive     = port:PG07<7><0><default><default>
```

12.5 [pwm8]

配置项	配置项含义
pwm_used	是否使用 PWM8
pwm_positive	PWM 输出 GPIO 配置

配置举例：

```
pwm_used        = 1
pwm_positive     = port:PD22<2><0><default><default>
```

12.6 [pwm8_suspend]

配置项	配置项含义
pwm_positive	PWM 输出 GPIO 配置

配置举例:

pwm_positive = port:PD22<7><0><default><default>

12.7 [spwm0]

配置项	配置项含义
spwm0_used	是否使用 CPUS PWM0
pwm_positive	PWM 输出 GPIO 配置

配置举例:

s_pwm0_used = 0
pwm_positive = port:PL10<2><0><default><default>

12.8 [spwm0_suspend]

配置项	配置项含义
pwm_positive	PWM 输出 GPIO 配置

配置举例:

pwm_positive = port:PL10<7><0><default><default>

13. 摄像头

13.1 [vind0]

配置项	配置项含义
vind0_used	vin 驱动总开关, 对应 media 设备
vind0_clk	vin 模块时钟, 实际使用可根据 sensor 的帧率和分辨率设置

配置举例:

```
vind0_used      = 1
vind0_clk       = 432000000
```

13.2 [vind0/csi_cci0]

配置项	配置项含义
csi_cci0_used	vin 子模块 cci0 使能配置
csi_cci0_sck	cci0 sck 信号 GPIO 配置
csi_cci0_sda	cci0 sda 信号 GPIO 配置

配置举例:

```
csi_cci0_used      = 0
csi_cci0_sck       = port:PE16<2><default><default><default>
csi_cci0_sda       = port:PE17<2><default><default><default>
```

13.3 [vind0/csi_cci1]

配置项	配置项含义
csi_cci3_used	vin 子模块 cci1 使能配置
csi_cci3_sck	cci1 sck 信号 GPIO 配置
csi_cci3_sda	cci1 sda 信号 GPIO 配置

配置举例:

```
csi_cci3_used      = 0
csi_cci3_sck       = port:PJ16<2><default><default><default>
csi_cci3_sda       = port:PJ17<2><default><default><default>
```

13.4 [vind0/csi_cci2]

配置项	配置项含义
csi_cci1_used	vin 子模块 cci2 使能配置
csi_cci1_sck	cci2 sck 信号 GPIO 配置
csi_cci1_sda	cci2 sda 信号 GPIO 配置

配置举例:

```
csi_cci1_used      = 0
csi_cci1_sck       = port:PI01<3><default><default><default>
csi_cci1_sda       = port:PI02<3><default><default><default>
```

13.5 [vind0/csi_cci3]

配置项	配置项含义
csi_cci3_used	vin 子模块 cci3 使能配置
csi_cci3_sck	cci3 sck 信号 GPIO 配置
csi_cci3_sda	cci3 sda 信号 GPIO 配置

配置举例:

```
csi_cci3_used      = 0
csi_cci3_sck       = port:PI08<3><default><default><default>
csi_cci3_sda       = port:PI09<3><default><default><default>
```

13.6 [vind0/csi2]

配置项	配置项含义
csi2_used	parser2 的使能开关, 并口 sensor 可用;mipi 接口 sensor 连接 csi0 和 csi1, 无需配置
csi2_pck	pclk 信号的 GPIO 配置。
csi2_hsync	hsync 信号的 GPIO 配置
csi2_vsync	vsync 信号的 GPIO 配置
csi2_d0	csi2 d0 信号的 GPIO 配置
csi2_d1	csi2 d1 信号的 GPIO 配置
csi2_d2	csi2 d2 信号的 GPIO 配置
csi2_d3	csi2 d3 信号的 GPIO 配置
csi2_d4	csi2 d4 信号的 GPIO 配置
csi2_d5	csi2 d5 信号的 GPIO 配置
csi2_d6	csi2 d6 信号的 GPIO 配置
csi2_d7	csi2 d7 信号的 GPIO 配置
csi2_d8	csi2 d8 信号的 GPIO 配置
csi2_d9	csi2 d9 信号的 GPIO 配置
csi2_d10	csi2 d10 信号的 GPIO 配置
csi2_d11	csi2 d11 信号的 GPIO 配置

配置举例:

```
csi2_used          = 1
csi2_pck           = port:PE00<2><default><default><default>
csi2_hsync         = port:PE02<2><default><default><default>
csi2_vsync         = port:PE03<2><default><default><default>
csi2_d0            = port:PE04<2><default><default><default>
csi2_d1            = port:PE05<2><default><default><default>
csi2_d2            = port:PE06<2><default><default><default>
csi2_d3            = port:PE07<2><default><default><default>
csi2_d4            = port:PE08<2><default><default><default>
csi2_d5            = port:PE09<2><default><default><default>
csi2_d6            = port:PE10<2><default><default><default>
csi2_d7            = port:PE11<2><default><default><default>
csi2_d8            = port:PE12<2><default><default><default>
csi2_d9            = port:PE13<2><default><default><default>
csi2_d10           = port:PE14<2><default><default><default>
csi2_d11           = port:PE15<2><default><default><default>
```

13.7 [vind0/csi3]

配置项	配置项含义
csi3_used	parser3 的使能开关, 并口 sensor 可用;mipi 接口 sensor 连接 csi0 和 csi1, 无需配置
csi3_pck	pclk 信号的 GPIO 配置。
csi3_hsync	hsync 信号的 GPIO 配置
csi3_vsync	vsync 信号的 GPIO 配置
csi3_d0	csi3 d0 信号的 GPIO 配置
csi3_d1	csi3 d1 信号的 GPIO 配置
csi3_d2	csi3 d2 信号的 GPIO 配置
csi3_d3	csi3 d3 信号的 GPIO 配置
csi3_d4	csi3 d4 信号的 GPIO 配置
csi3_d5	csi3 d5 信号的 GPIO 配置
csi3_d6	csi3 d6 信号的 GPIO 配置

配置项	配置项含义
csi3_d7	csi3 d7 信号的 GPIO 配置
csi3_d8	csi3 d8 信号的 GPIO 配置
csi3_d9	csi3 d9 信号的 GPIO 配置
csi3_d10	csi3 d10 信号的 GPIO 配置
csi3_d11	csi3 d11 信号的 GPIO 配置

配置举例:

```
csi3_used          = 1
csi3_pck           = port:PJ00<2><default><default><default>
csi3_hsync          = port:PJ02<2><default><default><default>
csi3_vsync          = port:PJ03<2><default><default><default>
csi3_d0           = port:PJ04<2><default><default><default>
csi3_d1           = port:PJ05<2><default><default><default>
csi3_d2           = port:PJ06<2><default><default><default>
csi3_d3           = port:PJ07<2><default><default><default>
csi3_d4           = port:PJ08<2><default><default><default>
csi3_d5           = port:PJ09<2><default><default><default>
csi3_d6           = port:PJ10<2><default><default><default>
csi3_d7           = port:PJ11<2><default><default><default>
csi3_d8           = port:PJ12<2><default><default><default>
csi3_d9           = port:PJ13<2><default><default><default>
csi3_d10           = port:PJ14<2><default><default><default>
csi3_d11           = port:PJ15<2><default><default><default>
```

13.8 [vind0/flash0]

配置项	配置项含义
flash0_used	是否使用闪光灯
flash0_type	闪光灯类型
flash0_en	闪光灯 enable 引脚 GPIO 配置。

配置项	配置项含义
flash0_mode	闪光灯 flash mode 引脚 GPIO 配置。
flash0_flvdd	闪光灯 VDD 配置，如``axp22_eldo1"。
flash0_flvdd_vol	闪光灯 VDD 电压值参考 datasheet。

配置举例：

```
flash0_used   = 1
flash0_type   = 2
flash0_en     =
flash0_mode   =
flash0_flvdd        = ""
flash0_flvdd_vol    =
```

13.9 [vind0/actuator0]

配置项	配置项含义
actuator0_used	是否使用 vcm 驱动
actuator0_name	vcm 驱动名字
actuator0_slave	vcm 设备的 i2c 地址
actuator0_af_pwdn	vcm 电源开关 GPIO 配置
actuator0_flvdd	FLVDD 配置
actuator0_flvdd_vol	vcm 电压值

配置举例：

```
actuator0_used   = 0
actuator0_name   = "ad5820_act"
actuator0_slave    =0x18
actuator0_af_pwdn   =
actuator0_flvdd      = "avvcc-csi"
```

actuator0_flvdd_vol = 2800000

13.10 [csi0/sensor0]

配置项	配置项含义
sensor0_used	是否使用 sensor0。
sensor0_mname	设置 sensor0 名称，如 ``ov5648``。
sensor0_twi_cci_spi	使用 twi 选 0, 使用 cci 选 1, 使用 spi 选 2
sensor0_twi_cci_id	根据 sensor0_twi_cci_spi 选择配置 id
sensor0_twi_addr	CSI 使用的 IIC 通道号地址
sensor0_mclk_id	CSI 使用的 mclk ID
sensor0_pos	摄像头位置前置填 ``front", 后置填 ``rear"。
sensor0_isp_used	如果是 RAW sensor 必须填 1，YUV 填 0。
sensor0_fmt	如果是 RAW 格式填 1，YUV 填 0。
sensor0_vflip	Sensor 图像垂直翻转。
sensor0_hflip	Sensor 图像水平翻转。
sensor0_iovdd	IOVDD 配置，请参考实际原理图填写。
sensor0_iovdd_vol	IOVDD 电压值一般为 2.8V(2800000)。
sensor0_avdd	AVDD 配置，如"axp15_aldo2"。
sensor0_avdd_vol	AVDD 电压值，一般为 2.8V(2800000)。
sensor0_dvdd	DVDD 配置，如 ``axp22_eldo1"。
sensor0_dvdd_vol	DVDD 电压值参考 datasheet，1.2/1.5/1.8V。
sensor0_power_en	Sensor power enable 引脚 GPIO 配置。
sensor0_reset	Sensor reset 引脚 GPIO 配置。
sensor0_pwdn	Sensor power down 引脚 GPIO 配置。
sensor0_sm_hs	
sensor0_sm_vs	

配置举例:

[vind0/sensor0]
sensor0_used = 1

```
sensor0_mname          = "imx317_mipi"
sensor0_twi_cci_spi    = 0
sensor0_twi_cci_id     = 2
sensor0_twi_addr       = 0x34
sensor0_mclk_id        = 2
sensor0_pos        = "front"
sensor0_isp_used       = 1
sensor0_fmt            = 1
sensor0_vflip          = 0
sensor0_hflip          = 0
sensor0_iovdd          = "iovdd-csi"
sensor0_iovdd_vol      = 1800000
sensor0_avdd           = ""
sensor0_avdd_vol       = 2800000
sensor0_dvdd           = ""
sensor0_dvdd_vol       = 1800000
sensor0_power_en       =
sensor0_reset          = port:PI12<1><0><1><0>
sensor0_pwdn           = port:PI14<1><0><1><0>
sensor0_sm_hs          = port:PI5<1><0><1><0>
sensor0_sm_vs          = port:PI6<1><0><1><0>
```

13.11 [csi0/sensor1]

配置项	配置项含义
sensor1_used	是否使用 sensor1。
sensor1_mname	设置 sensor1 名称，如 ``ov5648``。
sensor1_twi_cci_spi	使用 twi 选 0, 使用 cci 选 1, 使用 spi 选 2
sensor1_twi_cci_id	根据 sensor1_twi_cci_spi 选择配置 id
sensor1_twi_addr	CSI 使用的 IIC 通道号地址
sensor1_mclk_id	CSI 使用的 mclk ID
sensor1_pos	摄像头位置前置填 ``front", 后置填 ``rear"。
sensor1_isp_used	如果是 RAW sensor 必须填 1，YUV 填 0。

配置项	配置项含义
sensor1_fmt	如果是 RAW 格式填 1，YUV 填 0。
sensor1_vflip	Sensor 图像垂直翻转。
sensor1_hflip	Sensor 图像水平翻转。
sensor1_iovdd	IOVDD 配置，请参考实际原理图填写。
sensor1_iovdd_vol	IOVDD 电压值一般为 2.8V(2800000)。
sensor1_avdd	AVDD 配置，如"axp15_aldo2"。
sensor1_avdd_vol	AVDD 电压值，一般为 2.8V(2800000)。
sensor1_dvdd	DVDD 配置，如 ``axp22_eldo1"。
sensor1_dvdd_vol	DVDD 电压值参考 datasheet，1.2/1.5/1.8V。
sensor1_power_en	Sensor power enable 引脚 GPIO 配置。
sensor1_reset	Sensor reset 引脚 GPIO 配置。
sensor1_pwdn	Sensor power down 引脚 GPIO 配置。
sensor1_sm_hs	
sensor1_sm_vs	

配置举例:

```
[vind0/sensor1]
sensor1_used        = 1
sensor1_mname       = "imx317_mipi_2"
sensor1_twi_cci_spi  = 0
sensor1_twi_cci_id   = 3
sensor1_twi_addr     = 0x34
sensor1_mclk_id      = 3
sensor1_pos         = "rear"
sensor1_isp_used     = 1
sensor1_fmt          = 1
sensor1_vflip        = 0
sensor1_hflip        = 0
sensor1_iovdd        = "iovdd-csi"
sensor1_iovdd_vol    = 1800000
sensor1_avdd         = ""
sensor1_avdd_vol     = 2800000
```

```
sensor1_dvdd        = ""
sensor1_dvdd_vol    = 1800000
sensor1_power_en    =
sensor1_reset       = port:PI13<1><0><1><0>
sensor1_pwdn        = port:PI15<1><0><1><0>
sensor1_sm_hs       = port:PI5<1><0><1><0>
sensor1_sm_vs       = port:PI6<1><0><1><0>
```

13.12 [vind0/vinc0]

配置项	配置项含义
vinc0_used	vipp 使能开关
vinc0_csi_sel	配置 pipeline 上 parser 的 id, 必须配置
vinc0_mipi_sel	配置 pipeline 上 mipi 的 id, 不使用时配置为 oxff
vinc0_isp_sel	配置 pipeline 上 isp 的 id, 必须配置;isp 为空时, 只表示路由不做 isp 的效果处理
vinc0_isp_tx_ch	配置 pipeline 上 isp 的 ch, 必须配置, 默认为 0
vinc0_rear_sensor_sel	配置 pipeline 上使用的后置 sensor 的 id
vinc0_front_sensor_sel	配置 pipeline 上使用的前置 sensor 的 id
vinc0_sensor_list	是否使用 sensor_list 来适配不同的模组

配置举例:

```
vinc0_used          = 1
vinc0_csi_sel       = 0
vinc0_mipi_sel      = 0
vinc0_isp_sel       = 1
vinc0_isp_tx_ch     = 0
vinc0_rear_sensor_sel  = 0
vinc0_front_sensor_sel = 0
vinc0_sensor_list   = 0
```

13.13 [vind0/vinc1]

配置项	配置项含义
vinc1_used	vipp 使能开关
vinc1_csi_sel	配置 pipeline 上 parser 的 id, 必须配置
vinc1_mipi_sel	配置 pipeline 上 mipi 的 id, 不使用时配置为 oxff
vinc1_isp_sel	配置 pipeline 上 isp 的 id, 必须配置;isp 为空时, 只表示路由不做 isp 的效果处理
vinc1_isp_tx_ch	配置 pipeline 上 isp 的 ch, 必须配置, 默认为 0
vinc1_rear_sensor_sel	配置 pipeline 上使用的后置 sensor 的 id
vinc1_front_sensor_sel	配置 pipeline 上使用的前置 sensor 的 id
vinc1_sensor_list	是否使用 sensor_list 来适配不同的模组

配置举例:

```
vinc1_used          = 1
vinc1_csi_sel       = 0
vinc1_mipi_sel      = 0
vinc1_isp_sel       = 1
vinc1_isp_tx_ch     = 0
vinc1_rear_sensor_sel  = 0
vinc1_front_sensor_sel = 0
vinc1_sensor_list   = 0
```

13.14 [vind0/vinc2]

配置项	配置项含义
vinc2_used	vipp 使能开关
vinc2_csi_sel	配置 pipeline 上 parser 的 id, 必须配置
vinc2_mipi_sel	配置 pipeline 上 mipi 的 id, 不使用时配置为 oxff
vinc2_isp_sel	配置 pipeline 上 isp 的 id, 必须配置;isp 为空时, 只表示路由不做 isp 的效果处理
vinc2_isp_tx_ch	配置 pipeline 上 isp 的 ch, 必须配置, 默认为 0

配置项	配置项含义
vinc2_rear_sensor_sel	配置 pipeline 上使用的后置 sensor 的 id
vinc2_front_sensor_sel	配置 pipeline 上使用的前置 sensor 的 id
vinc2_sensor_list	是否使用 sensor_list 来适配不同的模组

配置举例:

```
vinc2_used          = 1
vinc2_csi_sel       = 0
vinc2_mipi_sel      = 0
vinc2_isp_sel       = 1
vinc2_isp_tx_ch     = 0
vinc2_rear_sensor_sel  = 0
vinc2_front_sensor_sel = 0
vinc2_sensor_list   = 0
```

13.15 [vind0/vinc3]

配置项	配置项含义
vinc3_used	vipp 使能开关
vinc3_csi_sel	配置 pipeline 上 parser 的 id, 必须配置
vinc3_mipi_sel	配置 pipeline 上 mipi 的 id, 不使用时配置为 oxff
vinc3_isp_sel	配置 pipeline 上 isp 的 id, 必须配置;isp 为空时, 只表示路由不做 isp 的效果处理
vinc3_isp_tx_ch	配置 pipeline 上 isp 的 ch, 必须配置, 默认为 0
vinc3_rear_sensor_sel	配置 pipeline 上使用的后置 sensor 的 id
vinc3_front_sensor_sel	配置 pipeline 上使用的前置 sensor 的 id
vinc3_sensor_list	是否使用 sensor_list 来适配不同的模组

配置举例:

```
vinc3_used          = 1
```

vinc3_csi_sel = 0
vinc3_mipi_sel = 0
vinc3_isp_sel = 1
vinc3_isp_tx_ch = 0
vinc3_rear_sensor_sel = 0
vinc3_front_sensor_sel = 0
vinc3_sensor_list = 0

13.16 [vind0/vinc4]

配置项	配置项含义
vinc4_used	vipp 使能开关
vinc4_csi_sel	配置 pipeline 上 parser 的 id, 必须配置
vinc4_mipi_sel	配置 pipeline 上 mipi 的 id, 不使用时配置为 oxff
vinc4_isp_sel	配置 pipeline 上 isp 的 id, 必须配置;isp 为空时, 只表示路由不做 isp 的效果处理
vinc4_isp_tx_ch	配置 pipeline 上 isp 的 ch, 必须配置, 默认为 0
vinc4_rear_sensor_sel	配置 pipeline 上使用的后置 sensor 的 id
vinc4_front_sensor_sel	配置 pipeline 上使用的前置 sensor 的 id
vinc4_sensor_list	是否使用 sensor_list 来适配不同的模组

配置举例:

vinc4_used = 1
vinc4_csi_sel = 0
vinc4_mipi_sel = 0
vinc4_isp_sel = 1
vinc4_isp_tx_ch = 0
vinc4_rear_sensor_sel = 0
vinc4_front_sensor_sel = 0
vinc4_sensor_list = 0

13.17 [vind0/vinc5]

配置项	配置项含义
vinc5_used	vipp 使能开关
vinc5_csi_sel	配置 pipeline 上 parser 的 id, 必须配置
vinc5_mipi_sel	配置 pipeline 上 mipi 的 id, 不使用时配置为 oxff
vinc5_isp_sel	配置 pipeline 上 isp 的 id, 必须配置;isp 为空时, 只表示路由不做 isp 的效果处理
vinc5_isp_tx_ch	配置 pipeline 上 isp 的 ch, 必须配置, 默认为 0
vinc5_rear_sensor_sel	配置 pipeline 上使用的后置 sensor 的 id
vinc5_front_sensor_sel	配置 pipeline 上使用的前置 sensor 的 id
vinc5_sensor_list	是否使用 sensor_list 来适配不同的模组

配置举例:

```
vinc5_used          = 1
vinc5_csi_sel       = 0
vinc5_mipi_sel      = 0
vinc5_isp_sel       = 1
vinc5_isp_tx_ch     = 0
vinc5_rear_sensor_sel  = 0
vinc5_front_sensor_sel = 0
vinc5_sensor_list      = 0
```

13.18 [vind0/vinc6]

配置项	配置项含义
vinc6_used	vipp 使能开关
vinc6_csi_sel	配置 pipeline 上 parser 的 id, 必须配置
vinc6_mipi_sel	配置 pipeline 上 mipi 的 id, 不使用时配置为 oxff
vinc6_isp_sel	配置 pipeline 上 isp 的 id, 必须配置;isp 为空时, 只表示路由不做 isp 的效果处理
vinc6_isp_tx_ch	配置 pipeline 上 isp 的 ch, 必须配置, 默认为 0

配置项	配置项含义
vinc6_rear_sensor_sel	配置 pipeline 上使用的后置 sensor 的 id
vinc6_front_sensor_sel	配置 pipeline 上使用的前置 sensor 的 id
vinc6_sensor_list	是否使用 sensor_list 来适配不同的模组

配置举例:

```
vinc6_used          = 1
vinc6_csi_sel       = 0
vinc6_mipi_sel      = 0
vinc6_isp_sel       = 1
vinc6_isp_tx_ch     = 0
vinc6_rear_sensor_sel   = 0
vinc6_front_sensor_sel  = 0
vinc6_sensor_list       = 0
```

13.19 [vind0/vinc7]

配置项	配置项含义
vinc7_used	vipp 使能开关
vinc7_csi_sel	配置 pipeline 上 parser 的 id, 必须配置
vinc7_mipi_sel	配置 pipeline 上 mipi 的 id, 不使用时配置为 oxff
vinc7_isp_sel	配置 pipeline 上 isp 的 id, 必须配置;isp 为空时, 只表示路由不做 isp 的效果处理
vinc7_isp_tx_ch	配置 pipeline 上 isp 的 ch, 必须配置, 默认为 0
vinc7_rear_sensor_sel	配置 pipeline 上使用的后置 sensor 的 id
vinc7_front_sensor_sel	配置 pipeline 上使用的前置 sensor 的 id
vinc7_sensor_list	是否使用 sensor_list 来适配不同的模组

配置举例:

```
vinc7_used          = 1
```

```
vinc7_csi_sel          = 0
vinc7_mipi_sel         = 0
vinc7_isp_sel          = 1
vinc7_isp_tx_ch        = 0
vinc7_rear_sensor_sel  = 0
vinc7_front_sensor_sel = 0
vinc7_sensor_list      = 0
```

14. TV

14.1 [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 =

14.2 [tvin_para]

配置项	配置项含义
tvin_used	是否使用 tvin 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 =

14.3 [tv0]

配置项	配置项含义
used	是否使用 tv0 1：使用 0：不使用
dac_src0	dac 号，支持 dac_src0~dac_src3
dac_type0	0:composite, 1:luma, 2:chroma, 3:reserved, 4:y/green, 5:u/pb/blue, 6:v/pr/red
interface	接口型号；1：cvbs，2：YPBPR, 4:y/green

配置举例：

```
used            = 1
dac_src0        = 0
dac_type0       = 0
interface       = 1
```

14.4 [di]

配置项	配置项含义
di_used	是否使用反交错 1：使用 0：不使用

配置举例：

```
di_used         = 1
```

15. SD/MMC

15.1 [sdc0]

配置项	配置项含义
sdc0_used	SDC 使用控制：1 使用，0 不用
bus-width	位宽：1-1bit，4-4bit
sdc0_d1	SDC DATA1 的 GPIO 配置
sdc0_d0	SDC DATA0 的 GPIO 配置
sdc0_clk	SDC DATA1 的 GPIO 配置
sdc0_d1	SDC CLK 的 GPIO 配置
sdc0_cmd	SDC CMD 的 GPIO 配置
sdc0_d3	SDC DATA3 的 GPIO 配置
sdc0_d2	SDC DATA2 的 GPIO 配置
cd-gpios	SDC 卡检测信号的 GPIO 配置
wp-gpios	
wp-inverted	
broken-cd	
sunxi-power-save-mode	SDC CLK 信号无数据传输时暂停
sunxi-dis-signal-vol-sw	
sd_uhs-sdr50	
sd_uhs-ddr50	
sd_uhs-sdr104	
vmmc	SDC 供电电源配置
vqmmc	SDC IO 供电电源配置
vdmmc	SDC 卡检测信号上拉电阻的电源配置

举例说明：

```
sdc0_used       = 1
bus-width       = 4
sdc0_d1         = port:PF00<2><1><2><default>
```

```
sdc0_d0          = port:PF01<2><1><2><default>
sdc0_clk         = port:PF02<2><1><2><default>
sdc0_cmd          = port:PF03<2><1><2><default>
sdc0_d3          = port:PF04<2><1><2><default>
sdc0_d2          = port:PF05<2><1><2><default>
cd-gpios         = port:PF06<0><1><2><default>
wp-gpios        = port:PG01<0><1><3><default>
wp-inverted      =
broken-cd        =
sunxi-power-save-mode =
sunxi-dis-signal-vol-sw =
sd_uhs-sdr50            =
sd_uhs-ddr50            =
sd_uhs-sdr104            =
max-frequency            = 150000000
vmmc="vcc-card"
vqmmc="vcc-pf"
vdmmc="vcc-pf"
```

15.2 [sdc1]

配置项	配置项含义
sdc1_used	SDC 使用控制：1 使用，0 不用
bus-width	位宽：1-1bit，4-4bit
sdc1_d1	SDC DATA1 的 GPIO 配置
sdc1_d0	SDC DATA0 的 GPIO 配置
sdc1_clk	SDC CLK 的 GPIO 配置
sdc1_cmd	SDC CMD 的 GPIO 配置
sdc1_d3	SDC DATA3 的 GPIO 配置
sdc1_d2	SDC DATA2 的 GPIO 配置
sunxi-power-save-mode	SDC CLK 信号无数据传输时暂停
sd-uhs-sdr50	支持 SDR50 速度模式
sd-uhs-ddr50	支持 DDR50 速度模式置

配置项	配置项含义
sd-uhs-sdr104	支持 SDR104 速度模式置
cap-sdio-irq	目前只用于 SDIO WIFI 驱动，表示控制器支持 SDIO 中断。
keep-power-in-suspend	目前只用于 SDIO WIFI 驱动，表示休眠时器件供电保持不变
ignore-pm-notify	目前只用于 SDIO WIFI 驱动，表示休眠唤醒时忽略内核的 pm notify
max-frequency	最高接口配置频率配置

举例说明:

```
sdc1_used        = 1
bus-width        = 4
sdc1_clk         = port:PG00<2><1><3><default>
sdc1_cmd         = port:PG01<2><1><3><default>
sdc1_d0          = port:PG02<2><1><3><default>
sdc1_d1          = port:PG03<2><1><3><default>
sdc1_d2          = port:PG04<2><1><3><default>
sdc1_d3          = port:PG05<2><1><3><default>
sunxi-power-save-mode =
sd-uhs-sdr50          =
sd-uhs-ddr50          =
sd-uhs-sdr104         =
cap-sdio-irq          =
keep-power-in-suspend   =
ignore-pm-notify    =
max-frequency     = 150000000
```

15.3 [sdc2]

配置项	配置项含义
sdc2_used	SDC 使用控制：1 使用，0 不用
non-removable	SDC 连接 Device 具备不可移除属性
bus-width	SDC DATA1 的 GPIO 配置

配置项	配置项含义
sdc1_d0	位宽：1-1bit，4-4bit 置
sdc2_ds	SDC eMMC Data Strobe 的 GPIO 配置置
sdc2_d1	SDC DATA1 的 GPIO 配置置
sdc2_d0	SDC DATA0 的 GPIO 配置
sdc2_clk	SDC CLK 的 GPIO 配置置
sdc2_cmd	SDC CMD 的 GPIO 配置
sdc2_d3	SDC DATA3 的 GPIO 配置
sdc2_d2	SDC DATA2 的 GPIO 配置
sdc2_d4	SDC DATA4GPIO 配置
sdc2_d5	SDC DATA5GPIO 配置
sdc2_d6	SDC DATA6GPIO 配置
sdc2_d7	SDC DATA7GPIO 配置
sdc2_emmc_rst	SDC eMMC Hardware Reset 的 GPIO 配置
cd-gpios	SDC 卡检测信号的 GPIO 配置
sunxi-power-save-mode	SDC CLK 信号无数据传输时暂停
sunxi-dis-signal-vol-sw	MMC 驱动支持开关 IO 电压但不修改 IO 电压值
sdc_tm4_sm0_freq0	SDC 采样参数控制
sdc_tm4_sm0_freq1	SDC 采样参数控制
sdc_tm4_sm1_freq0	SDC 采样参数控制
sdc_tm4_sm1_freq1	SDC 采样参数控制
sdc_tm4_sm2_freq0	SDC 采样参数控制
sdc_tm4_sm2_freq1	SDC 采样参数控制
sdc_tm4_sm3_freq0	SDC 采样参数控制
sdc_tm4_sm3_freq1	SDC 采样参数控制
sdc_tm4_sm4_freq0	SDC 采样参数控制
sdc_tm4_sm4_freq1	SDC 采样参数控制
vmmc	SDC 供电电源配置
vqmmc	SDC IO 供电电源配置
vdmmc	SDC 卡检测信号上拉电阻的电源配置

配置举例：

```
sdc2_used        = 1
```

```
non-removable =
bus-width  = 8
sdc2_clk          = port:PC04<3><1><3><default>
sdc2_cmd          = port:PC05<3><1><3><default>
sdc2_d0           = port:PC06<3><1><3><default>
sdc2_d1           = port:PC07<3><1><3><default>
sdc2_d2           = port:PC8<3><1><3><default>
sdc2_d3           = port:PC9<3><1><3><default>
sdc2_d4           = port:PC10<3><1><3><default>
sdc2_d5           = port:PC11<3><1><3><default>
sdc2_d6           = port:PC12<3><1><3><default>
sdc2_d7           = port:PC13<3><1><3><default>
sdc2_emmc_rst     = port:PC14<3><1><3><default>
sdc2_ds           = port:PC01<3><1><3><default>
cd-gpios     =
sunxi-power-save-mode =
sunxi-dis-signal-vol-sw =
mmc-ddr-1_8v     =
mmc-hs200-1_8v      =
mmc-hs400-1_8v      =
max-frequency    = 150000000
sdc_tm4_sm0_freq0 = 0
sdc_tm4_sm0_freq1 = 0
sdc_tm4_sm1_freq0 = 0x00000000
sdc_tm4_sm1_freq1 = 0
sdc_tm4_sm2_freq0 = 0x00000000
sdc_tm4_sm2_freq1 = 0
sdc_tm4_sm3_freq0 = 0x05000000
sdc_tm4_sm3_freq1 = 0x00000405
sdc_tm4_sm4_freq0 = 0x00050000
sdc_tm4_sm4_freq1 = 0x00000408
vmmc       = "vcc-nand"
vqmmc      = "vcc-emmcvq33"
vdmmc          = "none"
```

15.4 [gpio_para]

配置项	配置项含义
compatible	该配置的名字
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
normal_led	正常状态灯使用的 GPIO
standby_led	休眠状态灯使用的 GPIO

配置举例：

```
compatible      = "allwinner,sunxi-init-gpio"
gpio_used       = 1
gpio_num        = 2
gpio_pin_1      =  port:PL10<1><default><default><1>
gpio_pin_2      =  port:PA15<1><default><default><0>
normal_led      = "gpio_pin_1"
standby_led     = "gpio_pin_2"
```

16. USB 控制器标志

16.1 [usbc0]

配置项	配置项含义
usb0_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_detect_mode	usb otg 开关有两个配置. 0-thread scan, 1-id gpio interrupt
usb_id_gpio	USB ID pin 脚配置
usb_det_vbus_gpio	USB DET_VBUS pin 脚配置
usb_drv_vbus_gpio	USB DRY_VBUS pin 脚配置
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_regulator_io	usb 供电的 regulator GPIO
usb_wakeup_suspend	支持 usb 唤醒功能 0：关闭 usb 唤醒功能 1：当进入 normal standby 时候，支持 usb 唤醒（例如鼠标等外设）
usb_luns	使用 mass storage 功能时的盘符数量
usb_serial_unique	usb device 的序列号是否唯一。1：唯一，使用 chip id; 0：相同：由 usb_serial_number 指定
usb_serial_number	usb device 的序列号量

配置举例：

```
usbc0_used        = 1
usb_port_type     = 2
usb_detect_type   = 1
usb_detect_mode   = 0
usb_id_gpio       =port:PB4<0><1><default><default>
```

usb_det_vbus_gpio = "axp_ctrl"
usb_drv_vbus_gpio = "axp_ctrl"
usb_host_init_state = 0
usb_wakeup_suspend = 0
--- USB Device
usb_luns = 3
usb_serial_unique = 0
usb_serial_number = "20080411"

16.2 [usbc1]

配置项	配置项含义
usb1_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_drv_vbus_gpio	USB DRY_VBUS 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_wakeup_suspend	支持 usb 唤醒功能 0：关闭 usb 唤醒功能 1：当进入 normal standby 时候，支持 usb 唤醒（例如鼠标等外设）

配置举例:

usbc1_used = 1
usb_drv_vbus_gpio = port:PB5<0><1><default><default>
usb_host_init_state = 1
usb_wakeup_suspend = 0

17. gsensor

17.1 [gsensor_para]

配置项	配置项含义
gsensor_used	是否要使用 gsensor
gsensor_twi_id	gsensor 的 i2c 控制 id 号，0：TWI0，1：TWI1, 2:TWI2
gsensor_twi_addr	gsensor 使用的 i2c 地址
gsensor_int1	gsensor 使用的中断 1GPIO 配置
gsensor_int2	gsensor 使用的中断 2GPIO 配置

配置举例：

```
gsensor_used       = 1
gsensor_twi_id     = 2
gsensor_twi_addr   = 0x18
gsensor_int1       = port:PA09<6><1><default><default>
gsensor_int2       =
```

18. WIFI

18.1 [wlan]

配置项	配置项含义
wlan_used	是否要使用 wifi
wlan_busnum	所使用的 USB 号，如使用的是 USB3，则此值为 3
clocks	低功耗时钟，此值固定为 &clk_losc_out
wlan_power	wifi 模组使用哪一路 AXP 供电
wlan_io_regulator	wifi 模组 io 使用哪一路 AXP 供电
wlan_hostwake	wifi 唤醒主控脚
wlan_regon	Wifi 使能脚
wlan_en	wifi 电源控制所使用的 IO

配置举例:

```
wlan_used          = 1
wlan_busnum        = 1
wlan_usbnum        = 3
wlan_power         = "vcc33-wifi"
wlan_io_regulator  = "vcc18-wifi"
wlan_en            = port:PL08<1><default><default><0>
wlan_regon         = port:PL04<1><default><default><0>
wlan_hostwake      = port:PL05<6><default><default><0>
```

19. 蓝牙

19.1 [bt_para]

配置项	配置项含义
bt_used	蓝牙使用控制：1 使用，0 不用
clocks	低功耗为 &clk_losc_out
bt_power	bt 模组使用哪一路 AXP 供电（通常情况下和 wifi 相同）
bt_io_regulator	bt 模组 io 使用哪一路 AXP 供电（通常情况下和 wifi 相同）
bt_rst_n	bt 使能脚

配置举例：

```
bt_used          = 1
clocks           = "&clk_losc_out"
bt_power         = "vcc18-wifi"
bt_io_regulator  = "vcc-pg"
bt_rst_n         = port:PL12<1><default><default><0>
```

19.2 [btlpm]

配置项	配置项含义
btlpm_used	蓝牙低功耗使用控制：1 使用，0 不用
uart_index	使用的串口序号，如使用 ttyS1，则此值为 1
bt_wake	主控唤醒 bt 引脚
bt_hostwake	bt 唤醒主控引脚

配置举例：

```
btlpm_used      = 1
uart_index      = 1
bt_wake         = port:PL11<1><default><default><1>
bt_hostwake     = port:PL06<6><default><default><0>
```

20. 数字音频总线（TDM）

20.1 [daudio2]

配置项	配置项含义
daudio2_used	是否开启 daudio2，1：开启，0：不开启

配置举例：

```
daudio2_used = 1
```

20.2 [sndhdmi]

配置项	配置项含义
sndhdmi_used	是否开启 sndhdmi，1：开启，0：不开启

配置举例：

```
sndhdmi_used  = 1
```

注意：要生成并注册 HDMI 声卡，就必须要把 sndhdmi_used 和 daudio2_used 都设置为 1。

20.3 [snddaudio0]

配置项	配置项含义
snddaudio0_used	是否使用该接口，默认配置为 0 1：使用 0：不使用

配置举例：

```
snddaudio0_used         = 0
```

20.4 [daudio0]

配置项	配置项含义
daudio0_used	是否使用 daudio0 接口，默认要配置为 1 1：使用 0：不使用
pcm_lrck_period	每声道 bclk 个数/lrck 个数，设置如下：PCM mode: Number of BCLKs within(Left + Right)channel width 注意在 pcm 模式下，pcm_lrck_period 代表左和右声道相加，2 个声道的大小；I2S/ Left-Justified/Right-Justified mode: Number of BCLKs within each individual channel width(Left or Right)，pcm_lrck_period 代表左或者右声道，一个声道的大小；在 i2s 模式下，一个 lrck 的宽度：232。假如 fs=48k，那么需要的 bclk 是 3.072M = 23248k; bclk_div = 24.576M/3.072M=8; 在 pcm 模式下，一个 lrck 的宽度就是 32。假如 fs=8k，那么需要的 bclk 是：328k=256k
pcm_lrckr_period	未使用
slot_width_select	数据 word 的宽度，对 i2s 模式，pcm 模式都有效。16bits/20bits/ 24bits/32bits
pcm_lsb_first	数据 endian，0: msb first; 1: lsb first
tx_data_mode	数据格式，0: 16bit linear PCM; 1: 8bit linear PCM; 2: 8bit u-law; 3: 8bit a-law
rx_data_mode	数据格式，0: 16bit linear PCM; 1: 8bit linear PCM; 2: 8bit u-law; 3: 8bit a-law
daudio_master	Master/slave 模式：1:daudio0 slave; 4:daudio0 master

配置项	配置项含义
audio_format	1 SND_SOC_DAIFMT_I2S(standard i2s format). use 表示标准 i2s 格式；2 SND_SOC_DAIFMT_RIGHT_J(right justfied format). 表示右对齐格式；3 SND_SOC_DAIFMT_LEFT_J(left justfied format) 表示左对齐格式；4 SND_SOC_DAIFMT_DSP_A 短帧模式并设置 frame_width 为 0. 短帧；5 SND_SOC_DAIFMT_DSP_B 长帧模式并设置 frame_width 为 1. 长帧；
signal_inversion	信号的翻转，比如标准的 I2S 模式，如果 lrck 翻转是模式，那么用示波器测量，左右声道是跟标准 i2s 模式相反的。如果 bclk 是翻转模式，那么用示波器测量，BCLK 信号是翻转的。1 SND_SOC_DAIFMT_NB_NF(normal bit clock + frame) use 表示 bclk 采用正常模式，lrck 也正常模式 2 SND_SOC_DAIFMT_NB_IF(normal BCLK + inv FRM) 表示 bclk 采用正常模式，lrck 采用翻转模式 3 SND_SOC_DAIFMT_IB_NF(invert BCLK + nor FRM) use 表示 bclk 采用翻转模式，lrck 采用正常模式 4 SND_SOC_DAIFMT_IB_IF(invert BCLK + FRM) 表示 bclk 采用翻转模式，lrck 采用翻转模式
frametype	长帧或短帧 0: long frame = 2 clock width; 1: short frame
tdm_config	I2S 或 PCM 选择 0:pcm 1:i2s

配置举例:

```
pcm_lrck_period =   0x20
pcm_lrckr_period =   0x01
slot_width_select =   0x20
pcm_lsb_first =   0x0
tx_data_mode =   0x0
rx_data_mode =   0x0
daudio_master =   0x04
audio_format =   0x01
signal_inversion =   0x01
frametype =   0x0
tdm_config =   0x01
```

mclk_div = 0x0
daudio0_used = 0

注意：要生成并注册 Daudio0 声卡，就必须要把 snddaudio0_used 和 daudio0_used 都设置为 1。

20.5 [snddaudio1]

配置项	配置项含义
snddaudio1_used	是否使用该接口，默认配置为 0 1：使用 0：不使用

配置举例：

snddaudio1_used = 0

20.6 [daudio1]

配置项	配置项含义
daudio1_used	是否使用 daudio1 接口 1：使用 0：不使用
pcm_lrck_period	每声道 bclk 个数/lrck 个数，设置如下：PCM mode: Number of BCLKs within(Left + Right)channel width 注意在 pcm 模式下，pcm_lrck_period 代表左和右声道相加，2 个声道的大小；I2S/ Left-Justified/Right-Justified mode: Number of BCLKs within each individual channel width(Left or Right)，pcm_lrck_period 代表左或者右声道，一个声道的大小；在 i2s 模式下，一个 lrck 的宽度：232。假如 fs=48k，那么需要的 bclk 是 3.072M = 23248k; bclk_div = 24.576M/3.072M=8; 在 pcm 模式下，一个 lrck 的宽度就是 32。假如 fs=8k，那么需要的 bclk 是：328k=256k
pcm_lrckr_period	未使用

配置项	配置项含义
slot_width_select	数据 word 的宽度，对 i2s 模式，pcm 模式都有效。16bits/20bits/24bits/32bits
pcm_lsb_first	数据 endian，0: msb first; 1: lsb first
tx_data_mode	数据格式，0: 16bit linear PCM; 1: 8bit linear PCM; 2: 8bit u-law; 3: 8bit a-law
rx_data_mode	数据格式，0: 16bit linear PCM; 1: 8bit linear PCM; 2: 8bit u-law; 3: 8bit a-law
daudio_master	Master/slave 模式： 1:daudio0 slave; 4:daudio0 master
audio_format	1 SND_SOC_DAIFMT_I2S(standard i2s format). use 表示标准 i2s 格式；2 SND_SOC_DAIFMT_RIGHT_J(right justfied format). 表示右对齐格式；3 SND_SOC_DAIFMT_LEFT_J(left justfied format) 表示左对齐格式；4 SND_SOC_DAIFMT_DSP_A 短帧模式并设置 frame_width 为 0. 短帧；5 SND_SOC_DAIFMT_DSP_B 长帧模式并设置 frame_width 为 1. 长帧；
signal_inversion	信号的翻转，比如标准的 I2S 模式，如果 lrck 翻转是模式，那么用示波器测量，左右声道是跟标准 i2s 模式相反的。如果 bclk 是翻转模式，那么用示波器测量，BCLK 信号是翻转的。1 SND_SOC_DAIFMT_NB_NF(normal bit clock + frame) use 表示 bclk 采用正常模式，lrck 也正常模式 2 SND_SOC_DAIFMT_NB_IF(normal BCLK + inv FRM) 表示 bclk 采用正常模式，lrck 采用翻转模式 3 SND_SOC_DAIFMT_IB_NF(invert BCLK + nor FRM) use 表示 bclk 采用翻转模式，lrck 采用正常模式 4 SND_SOC_DAIFMT_IB_IF(invert BCLK + FRM) 表示 bclk 采用翻转模式，lrck 采用翻转模式
frametype	长帧或短帧 0: long frame = 2 clock width; 1: short frame
tdm_config	I2S 或 PCM 选择 0:pcm 1:i2s

配置举例：

```
pcm_lrck_period =   0x20
pcm_lrckr_period =   0x01
slot_width_select =   0x20
```

pcm_lsb_first = 0x0
tx_data_mode = 0x0
rx_data_mode = 0x0
daudio_master = 0x04
audio_format = 0x01
signal_inversion = 0x01
frametype = 0x0
tdm_config = 0x01
daudio1_used = 0

注意：要生成并注册 Daudio1 声卡，就必须要把 snddaudio1_used 和 daudio1_used 都设置为 1。

21. CODEC

21.1 [sndcodec]

配置项	配置项含义
sndcodec_used	是否使用该接口，默认要配置为 1 1：使用 0：不使用

配置举例：

```
sndcodec_used    = 1
```

21.2 [codec]

配置项	配置项含义
codec_used	是否使用 H5 模拟音频输入输出，0x1：使用，0x0：不使用
headphonevol	HP 默认音量设置，最大值是 0x3f
spkervol	SPK 默认音量设置，最大值是 0x1f
maingain	Mic1 前端增益，最大值是 0x7
adcagc_cfg	Adc 自动增益控制，0x1：开启，0x0：不开启
adcdrc_cfg	Drc 动态范围控制，0x1：开启，0x0：不开启
adchpf_cfg	Adc 端高通滤波，0x1：开启，0x0：不开启
dacdrc_cfg	播放动态音效调节，0x1：开启，0x0：不开启
dachpf_cfg	播放通路高通滤波开启，0x1：开启，0x0：不开启
pa_sleep_time	喇叭 pa 进入稳定状态需要的时间
gpio-spk	外部功放使能脚

配置举例：

```
codec_used = 0x1
```

headphonevol = 0x3b
spkervol = 0x1b
maingain = 0x4
adcagc_cfg = 0x0
adcdrc_cfg = 0x0
adchpf_cfg = 0x0
dacdrc_cfg = 0x0
dachpf_cfg = 0x0
pa_sleep_time = 0x32
gpio-spk = port:PA16<1><1><default><default>

注意：要生成并注册 audiocodec 声卡，就必须要把 sndcodec_used 和 codec_used 都设置为 1。

22. 红外

22.1 [s_cir0]

配置项	配置项含义
s_cir0_used	是否使用该模块，1：使用，0：不使用
ir_power_key_code	红外遥控器 powerkey 对应的按键值 0 使用
ir_protocol_used	使用红外协议；0x0：NEC protocols，0x01：RC5 protocols，0x02：NEC and RC5 protocols

配置举例：

```
s_cir0_used    = 1
ir_power_key_code   = 0x0
ir_protocol_used    = 0x00
```

23. 补光配置

23.1 [ir_cut]

配置项	配置项含义
compatible	补光模块名称
ir_cut_used	是否用补光模块
ir_cut_num	两路通道
ir_cut_gpio0	补光控制 GPIO0 配置
ir_cut_gpio1	补光控制 GPIO1 配置

配置举例:

```
compatible      = "allwinner,ir_cut"
ir_cut_used         = 1
ir_cut_num          = 2
ir_cut_gpio0        = port:PH04<1><default><default><0>
ir_cut_gpio1        = port:PH08<1><default><default><0>
```

24. PMU 电源

24.1 [pmu0]

配置项	相关说明
compatible	pmu0 名称，支持 axp233
used	是否使用 AXPxx： 0: 不使用， 1: 使用
pmu_id	0:axp19x, 1:axp209,2:axp22x, 3:axp806,4:axp808,5:axp809,6:axp803,7:axp813
pmu_vbusen_func	VBUSEN 功能选择， 0:as an output,1:as an input
pmu_reset	当 power key 按下超过 16s，PMU 是否复位， 0:not reset 1:reset

配置项	相关说明
pmu_irq_wakeup	当睡眠或者掉电时是否唤醒中断， 0:not wakeup 1:wakeup
pmu_hot_shutdown	pmu是否过温保护， 0:disable 1:enable
pmu_inshort	ACIN and VBUS inshort or not by software; 0:auto detect 1:inshort

配置举例：

```
compatible          = "axp233"
used                = 1
pmu_id              = 5
pmu_vbusen_func     = 0
pmu_reset           = 0
pmu_irq_wakeup      = 1
pmu_hot_shutdown    = 1
pmu_inshort         = 0
```

24.2 [charger0]

配置项	相关说明
compatible	axp233 charger 模块名称
pmu_bat_unused	智能充电，PMIC 温度配置选项。此功能目前没有开，配置为 0
pmu_chg_ic_temp	电池通路内阻，单位 mΩ
pmu_battery_rdc	电池容量，单位 mAh，如果配置，计量方式为库仑计方式，否则为电压方式
pmu_battery_cap	设置开机时充电电流大小，单位 mA，仅支持: 200-2800,200mA/steps
pmu_runtime_chgcur	

配置项	相关说明
pmu_suspend_chgcur	设置待机时充电电流大小，单位 mA，仅支持：200-2800,200mA/steps
pmu_shutdown_chgcur	设置关机时充电电流大小，单位 mA，仅支持：200-2800,200mA/steps
pmu_init_chgvol	设置充电完成时电池目标电压，仅支持：4100/4150/4200/4350mV
pmu_ac_vol	设置 USB 适配器限压值：4000/4100/4200/4300/4400/4500/4600/4700 mV,0-不限压
pmu_ac_cur	设置 USB 适配器限流值：500/900/1500/2000/2500/3000/3500/4000 mA，0-不限流

配置项	相关说明
pmu_usbpc_vol	设置 USB 连接 PC 时限压值：4000/4100/4200/4300/4400/4500/4600/4700 mV，0-不限压
pmu_usbpc_cur	设置 USB 连接 PC 时限流值：500/900/1500/2000/2500/3000/3500/4000 mA，0-不限流
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_chgled_func	CHGLED 功能控制：0：马达驱动 1：充电状态指示

配置项	相关说明
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
power_start	当系统在充电时，shutdown 是关机还是重启； 0: 重启， 1：关机
pmu_ocv_en	通过 ocv 充电， 1:enable， 必须使能
pmu_cou_en	通过 cou 充电， 0:disable， 1:enable
pmu_update_min_time	电池容量百分比更新最小时间， second， 0/5/10/20/30/60/120/164
pmu_bat_para1	电池空载电压为 3.13V 对应的电量值

配置项	相关说明
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

配置项	相关说明
pmu_bat_para0	电池空载电压为3.61V对应的电量值
pmu_bat_para1	电池空载电压为3.63V对应的电量值
pmu_bat_para2	电池空载电压为3.64V对应的电量值
pmu_bat_para3	电池空载电压为3.66V对应的电量值
pmu_bat_para4	电池空载电压为3.7V对应的电量值
pmu_bat_para5	电池空载电压为3.73V对应的电量值
pmu_bat_para6	电池空载电压为3.77V对应的电量值
pmu_bat_para7	电池空载电压为3.78V对应的电量值

配置项	相关说明
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

配置项	相关说明
pmu_bat_para26	26 电池空载电压为 4.01V 对应的电量值
pmu_bat_para27	27 电池空载电压为 4.05V 对应的电量值
pmu_bat_para28	28 电池空载电压为 4.08V 对应的电量值
pmu_bat_para29	29 电池空载电压为 4.1V 对应的电量值
pmu_bat_para30	30 电池空载电压为 4.12V 对应的电量值
pmu_bat_para31	31 电池空载电压为 4.14V 对应的电量值
pmu_bat_para32	32 电池空载电压为 4.15V 对应的电量值
pmu_bat_temp_enable	电池温度检测使能控制：0：disable 1：enable

配置项	相关说明
pmu_bat_charge_ltf	充电下限电池温度对应的电压
pmu_bat_charge_htf	充电上限电池温度对应的电压
pmu_bat_shutdown_ltf	关机下限电池温度对应的电压
pmu_bat_shutdown_htf	关机上限电池温度对应的电压
pmu_bat_temp_para1	电池温度-25 度对应的电压
pmu_bat_temp_para2	电池温度-15 度对应的电压
pmu_bat_temp_para3	电池温度-10 度对应的电压
pmu_bat_temp_para4	电池温度-5 度对应的电压
pmu_bat_temp_para5	电池温度0 度对应的电压
pmu_bat_temp_para6	电池温度5 度对应的电压

配置项	相关说明
pmu_bat_temp_para17	电池温度 10 度对应的电压
pmu_bat_temp_para18	电池温度 20 度对应的电压
pmu_bat_temp_para19	电池温度 30 度对应的电压
pmu_bat_temp_para10	电池温度 40 度对应的电压
pmu_bat_temp_para11	电池温度 45 度对应的电压
pmu_bat_temp_para12	电池温度 50 度对应的电压
pmu_bat_temp_para13	电池温度 55 度对应的电压
pmu_bat_temp_para14	电池温度 60 度对应的电压
pmu_bat_temp_para15	电池温度 70 度对应的电压
pmu_bat_temp_para16	电池温度 80 度对应的电压

配置举例：

```
compatible              = "axp233-charger"
pmu_bat_unused          = 0
pmu_chg_ic_temp         = 0
pmu_battery_rdc         = 100
pmu_battery_cap         = 0
pmu_runtime_chgcur      = 450
pmu_suspend_chgcur      = 1500
pmu_shutdown_chgcur     = 1500
pmu_init_chgvol         = 4200
pmu_ac_vol              = 4000
pmu_ac_cur              = 0
pmu_usbpc_vol           = 4400
pmu_usbpc_cur = 900
pmu_battery_warning_level1 = 15
pmu_battery_warning_level2 = 0
pmu_chgled_func         = 1
pmu_chgled_type         = 0
power_start             = 0
pmu_ocv_en              = 1
pmu_cou_en              = 1
pmu_update_min_time     = 30

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_bat_temp_enable = 0
pmu_bat_charge_ltf = 2261
pmu_bat_charge_htf = 388
pmu_bat_shutdown_ltf = 3200
pmu_bat_shutdown_htf = 237
pmu_bat_temp_para1 = 7466
pmu_bat_temp_para2 = 4480
pmu_bat_temp_para3 = 3518
pmu_bat_temp_para4 = 2786
pmu_bat_temp_para5 = 2223
pmu_bat_temp_para6 = 1788
pmu_bat_temp_para7 = 1448
pmu_bat_temp_para8 = 969
pmu_bat_temp_para9 = 664
pmu_bat_temp_para10 = 466

```
pmu_bat_temp_para11     = 393
pmu_bat_temp_para12     = 333
pmu_bat_temp_para13     = 283
pmu_bat_temp_para14     = 242
pmu_bat_temp_para15     = 179
pmu_bat_temp_para16     = 134
```

24.3 [powerkey0]

配置项	配置项含义
compatible	powerkey0 名称
pmu_powkey_off_time	设置 powkey off 时间，ms，4000/6000/8000/10000
pmu_powkey_off_func	设置 powkey off 功能，0:shutdown,1:restart
pmu_powkey_off_en	使能 powkey off，0：not powerdown，1：powerdown
pmu_powkey_long_time	设置 powkey off 长中断时间，ms,1000/1500/2000/2500
pmu_powkey_on_time	设置 powkey on 时间，ms，128/1000/2000/3000

配置举例:

```
compatible              = "axp233-powerkey"
pmu_powkey_off_time     = 6000
pmu_powkey_off_func     = 0
pmu_powkey_off_en       = 1
pmu_powkey_long_time    = 1500
pmu_powkey_on_time      = 1000
```

24.4 [regulator0]

配置项	相关说明
regulator_count	regulator 数量

配置项	相关说明
regulator1	regulator1 对应的别名，请勿修改
regulator2	regulator2 对应的别名，请勿修改
regulator3	regulator3 对应的别名，请勿修改
regulator4	regulator4 对应的别名，请勿修改
regulator5	regulator5 对应的别名，请勿修改
regulator6	regulator6 对应的别名，请勿修改
regulator7	regulator7 对应的别名，请勿修改
regulator8	regulator8 对应的别名，请勿修改
regulator9	regulator9 对应的别名，请勿修改
regulator10	regulator10 对应的别名，请勿修改
regulator11	regulator11 对应的别名，请勿修改
regulator12	regulator12 对应的别名，请勿修改
regulator13	regulator13 对应的别名，请勿修改
regulator14	regulator14 对应的别名，请勿修改
regulator15	regulator15 对应的别名，请勿修改
regulator16	regulator16 对应的别名，请勿修改
regulator17	regulator17 对应的别名，请勿修改
regulator18	regulator18 对应的别名，请勿修改
regulator19	regulator19 对应的别名，请勿修改
regulator20	regulator20 对应的别名，请勿修改
regulator21	regulator21 应的别名，请勿修改
regulator22	regulator22 对应的别名，请勿修改
regulator23	regulator23 对应的别名，请勿修改

配置举例：

```
compatible      = "axp233-regulator"
regulator_count = 19
regulator1      = "axp233_dcdc1 none vcc-3v3 vcc-io vcc33-usb vcc33-pc vcc-pd vcc-pa vcc-card vcc-nand vcc-ephy audio-33"
regulator2      = "axp233_dcdc2 none vdd-cpua vdd-test"
regulator3      = "axp233_dcdc3 none vdd-sys vdd09-usb"
regulator4      = "axp233_dcdc4 none vcc18-pc vcc-lvds vcc18-lcd vcc18-dsi vcc18-hdmi vcc-
```

```
tv vcc18-cmbcsi0 vcc18-cmbcsi1 vcc-efuse"
  regulator5     = "axp233_dcdc5 none vcc-dram"
  regulator6     = "axp233_rtc none vcc-rtc"
  regulator7     = "axp233_aldo1 none vcc33-wifi"
  regulator8     = "axp233_aldo2 none"
  regulator9     = "axp233_aldo3 none vdd18-lpddr vdd18-dram vcc-pll avcc"
  regulator10    = "axp233_dldo1 none vcc-pe vcc-pi vcc-pj iovdd-csi iovcc-csi"
  regulator11    = "axp233_dldo2 none vdd09-hdmi"
  regulator12    = "axp233_eldo1 none vcc-pg vcc18-wifi vcc-dcxo-io"
  regulator13    = "axp233_eldo2 none vcc-pf"
  regulator14    = "axp233_eldo3 none vcc-pl"
  regulator15    = "axp233_ldoio0 none dvdd-csi"
  regulator16    = "axp233_ldoio1 none"
  regulator17    = "axp233_swout none"
  regulator18    = "axp233_dc1sw none vcc-lcd vcc-ctp"
  regulator19    = "axp233_dc5ldo none vdd-cpus"
```

24.5 [axp_gpio0]

配置项	配置项含义
compatible	axp gpio0 名称

配置举例：

```
compatible = "axp233-gpio"
```

25. CPUS

25.1 [s_uart0]

配置项	配置项含义
s_uart0_used	使能 cpus 的 uart，为 1 使能，为 0 关闭
s_uart0_tx	Uart 口发送引脚配置
s_uart0_rx	Uart 接收引脚配置

配置举例：

```
s_uart0_used    = 1
s_uart0_tx      = port:PL02<2><default><default><default>
s_uart0_rx      = port:PL03<2><default><default><default>
```

25.2 [s_rsb0]

配置项	配置项含义
s_rsb0_used	使能 cpus 使用 rsb 总线，为 1 使能，为 0 关闭
s_rsb0_sck	Rsb 时钟引脚设置
s_rsb0_sda	Rsb 数据引脚设置

配置举例：

```
s_rsb0_used     = 1
s_rsb0_sck      = port:PL00<2><1><1><default>
s_rsb0_sda      = port:PL01<2><1><1><default>
```

25.3 [s_jtag0]

配置项	配置项含义
s_jtag0_used=xx	JTAG 使能
s_jtag0_tms=xx	测试模式选择输入 (TMS) 的 GPIO 配置
s_jtag0_tck=xx	测试时钟输入 (TMS) 的 GPIO 配置
s_jtag0_tdo=xx	测试数据输出 (TDO) 的 GPIO 配置
s_jtag0_tdi=xx	测试数据输入（TDI）的 GPIO 配置

配置举例:

```
s_jtag0_used      = 0
s_jtag0_tms       = port:PL04<2><1><2><default>
s_jtag0_tck       = port:PL05<2><1><2><default>
s_jtag0_tdo       = port:PL06<2><1><2><default>
s_jtag0_tdi       = port:PL07<2><1><2><default>
```

26. VIRTUAL DEVICE

26.1 [Vdevice]

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_used     = 1
Vdevice_0        = port:PB01<4><1><2><default>
Vdevice_1        = port:PB02<4><1><2><default>
```

27. 管脚 Bias 值

27.1 [gpio_bias]

配置项	配置项含义
pa_bias	修改 pa 组管脚的 bias 值
pd_bias	修改 pd 组管脚的 bias 值
pe_bias	修改 pe 组管脚的 bias 值
pf_bias	修改 pf 组管脚的 bias 值
pg_bias	修改 pg 组管脚的 bias 值
pi_bias	修改 pi 组管脚的 bias 值
pj_bias	修改 pj 组管脚的 bias 值
vcc_bias	修改 vcc-io 管脚的 bias 值

配置举例:

```
;-----------------------------------------
;
; normal config: eg. px_bias = "pmu_name:supply_name:voltage"
;
;               pmu_name = axp809, axp806
;               supply_name = dcdc1, dcdc2, aldo1, gpio1ldo, etc...
;
; special config: eg. px_bias = "floating"
;               when the gpio is float, use it
;               and the register will ignore it(use default value)
;
; special config: eg. px_bias = "constant:voltage"
;               when the gpio connect a constant voltage, use it
;               and the register will set a matching value
;
;useage: pa_bias = "axp809:dcdc1:3300" meant seting pa group bias is 3.3V
```

```
;               （在最后冒号数字如果这样写 3300：表明只设置 gpio bias（耐压值）为 3.3V）

;    pa_bias = "axp809:dcdc1:1003300" meant seting pa group bias is 3.3V, and set
;                                    axp809 dcdc1 output voltage is 3.3v
;               （在最后冒号数字如果这样写 103300：表明先设置 gpio bias 值，继而设置供
给
;                 该路 gpio 的电压输出（PMU 输出）也设置成 3.3V）
;-----------------------------------------
[gpio_bias]
pa_bias       = "axp809:dcdc1:3300"
pd_bias       = "axp809:bldo1:3300"
pe_bias       = "axp809:dldo1:2800"
pf_bias       = "axp809:eldo2:3300"
pg_bias       = "axp809:eldo1:3300"
pi_bias       = "axp809:dldo1:2800"
pj_bias       = "axp809:dcdce:2800"
vcc_bias      = "axp809:dcdc1:3300"
```

（注意：gpio_bias 为 28 纳米工艺引入的电气特性参数，如果该管脚用于通讯，而 bias 值设置不恰当可能引起通信异常）

28. 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.tates nor implies warranty of any kind, including fitness for any particular application.