全志科技
Allwinner Technology

H6 sys_config

配置说明文档

0.8

2017.03.01

文档履历

版本号	日期	制/修订人	内容描述
0.8	2017.03.01		

目录

1. 前言

1.1 编写目的

本文档目的是介绍 sys_config.fex 各个节点配置的意义，让用户明确掌握 sys_config.fex 配置和使用方法。

1.2 适用范围

适用于 H5、H6 芯片相关平台。

1.3 相关人员

用户、板级配置维护相关人员。

2. 节点配置说明

2.1 系统 (SYSTEM)

2.1.1 [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    = "petrel-p1"
```

2.1.2 [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 = 1

2.1.3 [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
power_mode	0: 使用 axp 1: 使用 dummy axp

配置举例：

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

2.1.4 [ir_boot_recovery]

配置项	配置项含义
ir_boot_recovery_used	1: 启动时通过 ir 判断是否进入 Android recovery mode 0：close
ir_addr_code0	遥控器 0 IR 按键值
burn_key	遥控器 0 按键值对应的地址码
ir_recovery_key_code1	遥控器 0 IR 按键值
ir_addr_code1	遥控器 1 按键值对应的地址码

配置举例：

```
ir_boot_recovery_used       = 0
ir_recovery_key_code0       = 0x57
ir_addr_code0               = 0x9f00
```



```
ir_recovery_key_code1       = 0x15
ir_addr_code1               = 0xbc00
```

2.1.5 [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	1: 启用 PMU power key 启动 0：禁用 PMU power key

配置举例：

```
used      = 1
start_type  = 1
irkey_used  = 1
pmukey_used = 0
```

2.1.6 [box_standby_led]

配置项	配置项含义
gpio0	进入假关机，设置 led0 状态为开
gpio1	进入假关机，设置 led1 状态为关

配置举例：


```
gpio0     = port:PL10<1><default><default><1>
gpio1     = port:PA15<1><default><default><0>
```

注意：用于系统进入假关机设置 led 状态，必须以 gpio0、gpio1、gpio2 递增去命名，才可以申请到对应的 gpio

2.1.7 [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:PL10<1><default><default><default>
next_work           = 3
```

2.1.8 [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>
```

2.1.9 [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

2.1.10 [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
```


2.1.11 [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: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>
```

2.1.12 [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

配置项	配置项含义
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:PC5<3><1><3><default>
sdc_cmd        = port:PC6<3><1><3><default>
sdc_d0         = port:PC8<3><1><3><default>
sdc_d1         = port:PC9<3><1><3><default>
sdc_d2         = port:PC10<3><1><2><default>
sdc_d3         = port:PC11<3><1><3><default>
sdc_d4         = port:PC12<3><1><3><default>
sdc_d5         = port:PC13<3><1><3><default>
sdc_d6         = port:PC14<3><1><3><default>
sdc_d7         = port:PC15<3><1><3><default>
sdc_emmc_rst   = port:PC16<3><1><3><default>
sdc_ex_dly_used = 2
```


2.1.13 [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:PH0<2><default><default><default>
twi_sda       = port:PH1<2><default><default><default>
```

2.1.14 [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:PA4<2><1><default><default>
uart_debug_rx   = port:PA5<2><1><default><default>
```

2.1.15 [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:PB0<4><default><default><default>
jtag_ck          = port:PB1<4><default><default><default>
jtag_do          = port:PB2<4><default><default><default>
jtag_di          = port:PB3<4><default><default><default>
```

2.1.16 [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
```

2.2 DRAM 配置

2.2.1 [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      = 672
dram_type     = 3
dram_zq       = 0x3b3bf9
dram_odt_en   = 0x1
dram_para1    = 0x10f410f4
dram_para2    = 0x1000
dram_mr0      = 0x1840
dram_mr1      = 0x40
dram_mr2      = 0x18
dram_mr3      = 0x2
dram_tpr0     = 0x0048a192
dram_tpr1     = 0x01b1a94b
```



```
dram_tpr2     = 0x00061043
dram_tpr3     = 0x04787896
dram_tpr4     = 0x0
dram_tpr5     = 0x0
dram_tpr6     = 100
dram_tpr7     = 0x1e08a1e0
dram_tpr8     = 0x0
dram_tpr9     = 0x0
dram_tpr10    = 0x2535
dram_tpr11    = 0x23330000
dram_tpr12    = 0x00008897
dram_tpr13    = 0x4002990
```

2.3 以太网配置

2.3.1 [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

配置项	配置项含义
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        = "rmii"
;gmac_rxd3      = port:PD08<4><default><3><default>
;gmac_rxd2      = port:PD09<4><default><3><default>
gmac_rxd1       = port:PD10<4><default><3><default>
gmac_rxd0       = port:PD11<4><default><3><default>
;gmac_rxclk     = port:PD12<4><default><3><default>
gmac_rxdv       = port:PD13<4><default><3><default>
;gmac_rxerr     = port:PD14<4><default><3><default>
;gmac_txd3      = port:PD15<4><default><3><default>
;gmac_txd2      = port:PD16<4><default><3><default>
gmac_txd1       = port:PD17<4><default><3><default>
gmac_txd0       = port:PD18<4><default><3><default>
gmac_txclk      = port:PD19<4><default><3><default>
gmac_txen       = port:PD20<4><default><3><default>
;gmac_clkin     = port:PD21<4><default><3><default>
gmac_mdc        = port:PD22<4><default><3><default>
gmac_mdio       = port:PD23<4><default><3><default>
gmac_power1     =
gmac_power2     =
```



```
gmac_power3      =
tx-delay         = 0
rx-delay         = 0
```

2.4 I2C 总线

2.4.1 [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:PA11<2><default><default><default>
twi0_sda       = port:PA12<2><default><default><default>
```

2.4.2 [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:PA18<2><default><default><default>
twi1_sda        = port:PA19<2><default><default><default>
```

2.4.3 [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:PE12<3><default><default><default>
twi2_sda        = port:PE13<3><default><default><default>
```

2.4.4 [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

2.5 UART

2.5.1 [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      = 1
uart0_port      = 0
uart0_type      = 2
uart0_tx        = port:PA4<2><1><default><default>
uart0_rx        = port:PA5<2><1><default><default>
```

2.5.2 [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
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>
```

2.5.3 [uart2]

配置项	配置项含义
uart2_used	UART 使用控制：1 使用，0 不用
uart2_port	UART 端口号
uart2_type	2：2 线模式;4：4 线模式;8：8 线模式。
uart2_tx	UART TX 的 GPIO 配置
uart2_rx	UART RX 的 GPIO 配置
uart2_rts	GPIO 配置
uart2_cts	GPIO 配置

配置举例：

```
uart2_used      = 0
uart2_port      = 2
uart2_type      = 4
uart2_tx        = port:PA0<2><1><default><default>
uart2_rx        = port:PA1<2><1><default><default>
uart2_rts       = port:PA2<2><1><default><default>
uart2_cts       = port:PA3<2><1><default><default>
```


2.5.4 [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      = 0
uart3_port      = 3
uart3_type      = 4
uart3_tx        = port:PA13<2><1><default><default>
uart3_rx        = port:PA14<2><1><default><default>
uart3_rts       = port:PA15<2><1><default><default>
uart3_cts       = port:PA16<2><1><default><default>
```


2.6 SPI 总线

2.6.1 [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；

配置项	配置项含义
spi0_cs0	SPI CS0 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

配置举例：

```
spi0_used      = 0
spi0_cs_number  = 1
spi0_cs_bitmap  = 1
spi0_cs0       = port:PC3<3><1><default><default>
spi0_sclk      = port:PC2<3><default><default><default>
spi0_mosi      = port:PC0<3><default><default><default>
spi0_miso      = port:PC4<4><default><default><default>
```

2.6.2 [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_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  = 1
spi1_cs_bitmap  = 1
spi1_cs0       = port:PA13<2><1><default><default>
spi1_sclk      = port:PA14<2><default><default><default>
spi1_mosi      = port:PA15<2><default><default><default>
spi1_miso      = port:PA16<2><default><default><default>
```

2.6.3 [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

2.7 NAND FLASH

2.7.1 [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

配置项	配置项含义
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_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_ce1          = 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>
nand0_ce2         = port:PC17<2><1><1><default>
nand0_ce3         = port:PC18<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
```

2.8 显示

2.8.1 [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)

配置项	配置项含义
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
dev0_output_type	boot 阶段显示配置：screen0 对应的输出类型（4：hdmi，2：cvbs）
dev0_output_mode	boot 阶段显示配置：screen0 对应的输出模式
dev0_screen_id	boot 阶段显示配置：screen0 对应的 DE 设备号
dev0_do_hpd	boot 阶段显示配置：是否支持热插拔检测
dev1_output_type	boot 阶段显示配置：screen1 对应的输出类型（4：hdmi，2：cvbs）
dev1_output_mode	boot 阶段显示配置：screen1 对应的输出模式
dev1_screen_id	boot 阶段显示配置：screen1 对应的 DE 设备号
dev1_do_hpd	boot 阶段显示配置：是否支持热插拔检测
dev2_output_type	保留
def_output_dev	保留
hdmi_mode_check	boot 阶段显示配置：是否支持 hdmi 输出模式检查

配置举例：

```
disp_init_enable    = 1
disp_mode           = 0
screen0_output_type = 3
screen0_output_mode = 4
screen1_output_type = 2
screen1_output_mode = 11
dev0_output_type    = 4
dev0_output_mode    = 4
dev0_screen_id      = 0
dev0_do_hpd         = 1
dev1_output_type    = 2
dev1_output_mode    = 11
dev1_screen_id      = 1
```



```
dev1_do_hpd       = 1
dev2_output_type   = 0
def_output_dev     = 0
hdmi_mode_check    = 1
fb0_format         = 0
fb0_width          = 1280
fb0_height         = 720
fb1_format         = 0
fb1_width          = 0
fb1_height         = 0
```

2.8.2 [lcd0]

盒子产品不适用

2.9 HDMI

2.9.1 [hdmi]

配置项	配置项含义
hdmi_used	是否使用 hdmi。1：使用；0：不使用
hdmi_power	内核阶段 hdmi 电源配置
hdmi_hdcp_enable	是否使能 hdcp
hdmi_cts_compatibility	cts 兼容性使能设置
hdmi_hpd_mask	hpd 掩码设置
hdmi_cec_support	是否支持 CEC 功能

配置举例：


```
hdmi_used            = 1
hdmi_power           = "vcc-hdmi-33"
hdmi_hdcp_enable     = 0
hdmi_cts_compatibility = 0
hdmi_hpd_mask        = 0
hdmi_cec_support     = 1
```

2.10 CVBS

2.10.1 [tv0]

配置项	配置项含义
used	是否使用 TV 输出
dac_src0	TV Encode 使用的 DAC 设备号
dac_type0	TV Encode 输出格式：0:composite,1:luma,2:chroma, 3:reserved,4:y/green,5:u/pb/blue,6:v/pr/red
dinterface	接口类型：1<->cvbs,2<->YPBPR,4<->SVIDEO

配置举例：

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

2.11 PWM

2.11.1 [pwm]

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:PD22<2><0><default><default>
```

2.11.2 [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:PD22<7><0><default><default>
```

2.11.3 [s_pwm0]

配置项	配置项含义
s_pwm0_used	是否使用 CPUS PWM0
pwm_positive	PWM 输出 GPIO 配置

配置举例：


```
s_pwm0_used      = 1
pwm_positive     = port:PL10<2><0><default><default>
```

2.11.4 [s_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:PL10<7><0><default><default>
```

2.12 摄像头

2.12.1 [csi0]

配置项	配置项含义
csi0_used	使用 mipi 接口的 sensor 请填 1。
csi0_sensor_list	如果配置了 system/etc/hawkview/sensor_list_ cfg.ini 文件，填 1，默认填 0。
csi0_pck	pclk 信号的 GPIO 配置。
csi0_mck	mclk 信号的 GPIO 配置。
csi0_hsync	hsync 信号的 GPIO 配置
csi0_vsync	vsync 信号的 GPIO 配置
csi0_d0	csi d0 信号的 GPIO 配置
csi0_d1	csi d1 信号的 GPIO 配置
csi0_d2	csi d2 信号的 GPIO 配置
csi0_d3	csi d3 信号的 GPIO 配置
csi0_d4	csi d4 信号的 GPIO 配置

配置项	配置项含义
csi0_d5	csi d5 信号的 GPIO 配置
csi0_d6	csi d6 信号的 GPIO 配置
csi0_d7	csi d7 信号的 GPIO 配置
csi0_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。
csi0_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。

配置举例：

```
csi0_used            = 0
csi0_sensor_list      = 0
csi0_pck             = port:PE00<2><default><default><default>
csi0_mck              = port:PE01<1><0><1><0>
csi0_hsync            = port:PE02<2><default><default><default>
csi0_vsync            = port:PE03<2><default><default><default>
csi0_d0              = port:PE04<2><default><default><default>
csi0_d1              = port:PE05<2><default><default><default>
csi0_d2              = port:PE06<2><default><default><default>
csi0_d3              = port:PE07<2><default><default><default>
csi0_d4              = port:PE08<2><default><default><default>
csi0_d5              = port:PE09<2><default><default><default>
csi0_d6              = port:PE10<2><default><default><default>
csi0_d7              = port:PE11<2><default><default><default>
csi0_sck             = port:PE12<2><default><default><default>
csi0_sda             = port:PE13<2><default><default><default>
```


2.12.2 [csi0/csi0_dev0]

配置项	配置项含义
csi0_dev0_used	是否使用 csi0_dev0，盒子上默认关闭此功能。
csi0_dev0_mname	设置 sensor 0 名称，如 ``ov5648``。
csi0_dev0_twi_addr	CSI 使用的 IIC 通道号, 用 twi0 填 0, 用 CSI 内部 CCI 不填

配置项	配置项含义
csi0_dev0_pos	摄像头位置前置填 ``front"，后置填 ``rear"。
csi0_dev0_isp_used	如果是 RAW sensor 必须填 1，YUV 填 0。
csi0_dev0_fmt	如果是 RAW 格式填 1，YUV 填 0。
csi0_dev0_stby_mode	填 0。
csi0_dev0_vflip	Sensor 图像垂直翻转。
csi0_dev0_hflip	Sensor 图像水平翻转。
csi0_dev0_iovdd	IOVDD 配置，请参考实际原理图填写。
csi0_dev0_iovdd_vol	IOVDD 电压值一般为 2.8V(2800000)。
csi0_dev0_avdd	AVDD 配置，如"axp15_aldo2"。
csi0_dev0_avdd_vol	AVDD 电压值，一般为 2.8V(2800000)。
csi0_dev0_dvdd	DVDD 配置，如 ``axp22_eldo1"。
csi0_dev0_dvdd_vol	DVDD 电压值参考 datasheet，1.2/1.5/1.8V。
csi0_dev0_afvdd	VCM 电源配置一般不用配置。
csi0_dev0_afvdd_vol	VCM 电压值为 2.8V。
csi0_dev0_power_en	Sensor power enable 引脚 GPIO 配置。
csi0_dev0_reset	Sensor reset 引脚 GPIO 配置。
csi0_dev0_pwdn	Sensor power down 引脚 GPIO 配置。
csi0_dev0_flash_used	是否使用闪光灯
csi0_dev0_flash_type	闪光灯类型
csi0_dev0_flash_en	闪光灯 enable 引脚 GPIO 配置。
csi0_dev0_flash_mode	闪光灯 flash mode 引脚 GPIO 配置。
csi0_dev0_flvdd	闪光灯 VDD 配置，如 ``axp22_eldo1"。
csi0_dev0_flvdd_vol	闪光灯 VDD 电压值参考 datasheet。
csi0_dev0_af_pwdn	VCM driver power down 引脚 GPIO 配置。
csi0_dev0_act_used	模组包含 VCM driver 时候填 1。
csi0_dev0_act_name	VCM driver 名字，如 ``ad5820_act``。
csi0_dev0_act_slave	VCM driver slave 地址。

配置举例：


```
csi0_dev0_used      = 1
csi0_dev0_mname     = "ov5640"
csi0_dev0_twi_addr  = 0x78
csi0_dev0_pos       = "rear"
csi0_dev0_isp_used  = 1
csi0_dev0_fmt       = 0
csi0_dev0_stby_mode = 0
csi0_dev0_vflip     = 0
csi0_dev0_hflip     = 0
csi0_dev0_iovdd     = "iovdd-csi"
csi0_dev0_iovdd_vol = 2800000
csi0_dev0_avdd      = "avdd-csi"
csi0_dev0_avdd_vol  = 2800000
csi0_dev0_dvdd      = "dvdd-csi-18"
csi0_dev0_dvdd_vol  = 1500000
csi0_dev0_afvdd     = "afvcc-csi"
csi0_dev0_afvdd_vol = 2800000
;csi0_dev0_power_en  = port:PB04<1><0><1><0>
csi0_dev0_power_en  =
csi0_dev0_reset     = port:PE14<1><0><1><0>
csi0_dev0_pwdn      = port:PE15<1><0><1><0>
csi0_dev0_flash_used = 0
csi0_dev0_flash_type = 2
csi0_dev0_flash_en  =
csi0_dev0_flash_mode =
csi0_dev0_flvdd     = ""
csi0_dev0_flvdd_vol =
csi0_dev0_af_pwdn   =
csi0_dev0_act_used  = 0
csi0_dev0_act_name  = "ad5820_act"
csi0_dev0_act_slave = 0x18
```


2.12.3 [csi0/csi0_dev1]

配置项	配置项含义
csi0_dev1_used	是否使用 csi0_dev1
csi0_dev1_mname	设置 sensor 1 名称，如 ``ov5648``。
csi0_dev1_twi_addr	请参考实际模组的 8bit ID 填写，如 0x6c。
csi0_dev1_pos	摄像头位置前置填 ``front"，后置填 ``rear"。
csi0_dev1_isp_used	如果是 RAW sensor 必须填 1，YUV 填 0。
csi0_dev1_fmt	如果是 RAW 格式填 1，YUV 填 0。
csi0_dev1_stby_mode	填 0。
csi0_dev1_vflip	Sensor 图像垂直翻转。
csi0_dev1_hflip	Sensor 图像水平翻转。
csi0_dev1_iovdd	IOVDD 配置，请参考实际原理图填写。
csi0_dev1_iovdd_vol	IOVDD 电压值一般为 2.8V(2800000)。
csi0_dev1_avdd	AVDD 配置，如"axp15_aldo2"。
csi0_dev1_avdd_vol	AVDD 电压值，一般为 2.8V(2800000)。
csi0_dev1_dvdd	DVDD 配置，如 ``axp22_eldo1"。
csi0_dev1_dvdd_vol	DVDD 电压值参考 datasheet，1.2/1.5/1.8V。
csi0_dev1_afvdd	VCM 电源配置一般不用配置。
csi0_dev1_afvdd_vol	VCM 电压值为 2.8V。
csi0_dev1_power_en	Sensor power enable 引脚 GPIO 配置。
csi0_dev1_reset	Sensor reset 引脚 GPIO 配置。
csi0_dev1_pwdn	Sensor power down 引脚 GPIO 配置。
csi0_dev1_flash_used	是否使用闪光灯
csi0_dev1_flash_type	闪光灯类型
csi0_dev1_flash_en	闪光灯 enable 引脚 GPIO 配置。
csi0_dev1_flash_mode	闪光灯 flash mode 引脚 GPIO 配置。
csi0_dev1_flvdd	闪光灯 VDD 配置，如 ``axp22_eldo1"。
csi0_dev1_flvdd_vol	闪光灯 VDD 电压值参考 datasheet。
csi0_dev1_af_pwdn	VCM driver power down 引脚 GPIO 配置。
csi0_dev1_act_used	模组包含 VCM driver 时候填 1。
csi0_dev1_act_name	VCM driver 名字，如 ``ad5820_act``。
csi0_dev1_act_slave	VCM driver slave 地址。

配置举例：

```
csi0_dev1_used      = 0
csi0_dev1_mname     = ""
csi0_dev1_twi_addr  = 0x78
csi0_dev1_pos       = "rear"
csi0_dev1_isp_used  = 1
csi0_dev1_fmt       = 0
csi0_dev1_stby_mode = 0
csi0_dev1_vflip     = 0
csi0_dev1_hflip     = 0
csi0_dev1_iovdd     = "iovdd-csi"
csi0_dev1_iovdd_vol = 2800000
csi0_dev1_avdd      = "avdd-csi"
csi0_dev1_avdd_vol  = 2800000
csi0_dev1_dvdd      = "dvdd-csi-18"
csi0_dev1_dvdd_vol  = 1500000
csi0_dev1_afvdd     = "afvcc-csi"
csi0_dev1_afvdd_vol = 2800000
csi0_dev1_power_en  =
csi0_dev1_reset     =
csi0_dev1_pwdn      =
csi0_dev1_flash_used = 0
csi0_dev1_flash_type = 2
csi0_dev1_flash_en  =
csi0_dev1_flash_mode =
csi0_dev1_flvdd     = ""
csi0_dev1_flvdd_vol =
csi0_dev1_af_pwdn   =
csi0_dev1_act_used  = 0
csi0_dev1_act_name  = "ad5820_act"
csi0_dev1_act_slave = 0x18
```


2.13 TV

2.13.1 [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               =
```

2.13.2 [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   =
```

2.13.3 [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

2.14 SD/MMC

2.14.1 [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
sunxi-power-save-mode	SDC CLK 信号无数据传输时暂停
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>
sunxi-power-save-mode   =
sunxi-dis-signal-vol-sw =
vmmc                ="vcc-sdcv"
vqmmc               ="vcc-sdcvq33"
vdmmc               ="vcc-sdcvd"
```

2.14.2 [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
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

配置项	配置项含义
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
```


2.14.3 [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
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

配置项	配置项含义
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_ds          = port:PC01<3><1><1><default>
```

```
sdc2_clk          = port:PC05<3><1><1><default>
sdc2_cmd          = port:PC06<3><1><1><default>
sdc2_d0           = port:PC08<3><1><1><default>
sdc2_d1           = port:PC09<3><1><1><default>
sdc2_d2           = port:PC10<3><1><1><default>
sdc2_d3           = port:PC11<3><1><1><default>
sdc2_d4           = port:PC12<3><1><1><default>
sdc2_d5           = port:PC13<3><1><1><default>
sdc2_d6           = port:PC14<3><1><1><default>
sdc2_d7           = port:PC15<3><1><1><default>
sdc2_emmc_rst     = port:PC16<3><1><1><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-emmc"
vqmmc                  = "none"
vdmmc                  = "none"
```


2.14.4 [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
gpio_pin_3	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
network_led	休眠状态灯使用的 GPIO
easy_light_used	是否为顶层接口开启硬件屏蔽，1：开启 0：禁用
normal_led_light	normal_led 灯亮 pin 口状态，1：高电平 0：低电平
standby_led_light	standby_led 灯亮 pin 口状态，1：高电平 0：低电平
network_led_light	network_led 灯亮 pin 口状态，1：高电平 0：低电平

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"
easy_light_used    = 1
normal_led_light   = 1
standby_led_light  = 1
```


2.15 USB 控制器标志

2.15.1 [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_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 device	
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
rndis_wceis	Wireless RNDIS 使能标志。1：使能;0：禁止

配置举例：

```
usbc0_used        = 1
usb_port_type     = 2
usb_detect_type   = 1
```



```
usb_id_gpio       =
usb_det_vbus_gpio  =
usb_drv_vbus_gpio  =
usb_host_init_state = 0
usb_regulator_io   = "nocare"
usb_wakeup_suspend  = 0
;---      USB Device
usb_luns          = 3
usb_serial_unique   = 0
usb_serial_number   = "20080411"
rndis_wceis       = 1
```

2.15.2 [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_regulator_io	给 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）

配置举例：

```
usbc1_used       = 1
usb_drv_vbus_gpio  = port:PL02<1><0><default><0>
usb_host_init_state = 1
```



```
usb_regulator_io    = "nocare"
usb_wakeup_suspend  = 0
```

2.16 WIFI

2.16.1 [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
;clocks           = "&clk_losc_out"
wlan_power         = "vcc-wifi"
wlan_io_regulator  = "vcc-wifi-io"
wlan_en            = port:PL07<1><default><default><0>
wlan_regon         = port:PL03<1><default><default><0>
wlan_hostwake      = port:PA11<6><default><default><0>
```


2.17 蓝牙

2.17.1 [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
bt_power         = "vcc-wifi"
bt_io_regulator    = "vcc-wifi-io"
bt_rst_n         = port:PL04<1><default><default><0>
```

2.17.2 [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:PA12<1><default><default><1>
bt_hostwake     = port:PL06<6><default><default><0>
```

2.18 数字音频总线（S/PDIF）

2.18.1 [spdif]

配置项	配置项含义
spdif_used	是否开启 spdif，1：开启，0：不开启

配置举例

```
spdif_used = 1
```

2.18.2 [sndspdif]

配置项	配置项含义
sndspdif_used	是否开启 spdif platform，1：开启，0：不开启

配置举例：

```
sndspdif_used = 1
```

注意：要生成并注册 spdif 声卡，就必须要把 spdif_used 和 sndspdif_used 都设置为 1。

2.19 数字音频总线（TDM）

2.19.1 [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
```

2.19.2 [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。

2.19.3 [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
```

2.19.4 [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
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；

配置项	配置项含义
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。

2.19.5 [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
```

2.19.6 [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
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
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。

2.20 CODEC

2.20.1 [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
```

2.20.2 [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

配置项	配置项含义
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。

2.21 红外

2.21.1 [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_protocol_used	红外协议选择，1：RC5 0：NEC
ir_addr_cnt	红外遥控器数量，最多 64 个
ir_power_key_code0	红外遥控器 powerkey 对应的按键值 0
ir_addr_code0	红外遥控器地址码 0

配置项	配置项含义
ir_power_key_code1	红外遥控器 powerkey 对应的按键值 1
ir_addr_code1	红外遥控器地址码 1
ir_power_key_code2	红外遥控器 powerkey 对应的按键值 2
ir_addr_code2	红外遥控器地址码 2
ir_power_key_code3	红外遥控器 powerkey 对应的按键值 3
ir_addr_code3	红外遥控器地址码 3
ir_power_key_code4	红外遥控器 powerkey 对应的按键值 4
ir_addr_code4	红外遥控器地址码 4
ir_power_key_code5	红外遥控器 powerkey 对应的按键值 5
ir_addr_code5	红外遥控器地址码 5
ir_power_key_code6	红外遥控器 powerkey 对应的按键值 6
ir_addr_code6	红外遥控器地址码 6
ir_power_key_code7	红外遥控器 powerkey 对应的按键值 7
ir_addr_code7	红外遥控器地址码 7
ir_power_key_code8	红外遥控器 powerkey 对应的按键值 8
ir_addr_code8	红外遥控器地址码 8
ir_power_key_code9	红外遥控器 powerkey 对应的按键值 9
ir_addr_code9	红外遥控器地址码 9
ir_power_key_code10	红外遥控器 powerkey 对应的按键值 10
ir_addr_code10	红外遥控器地址码 10
ir_power_key_code11	红外遥控器 powerkey 对应的按键值 11
ir_addr_code11	红外遥控器地址码 11
ir_power_key_code12	红外遥控器 powerkey 对应的按键值 12
ir_addr_code12	红外遥控器地址码 12
ir_power_key_code13	红外遥控器 powerkey 对应的按键值 13
ir_addr_code13	红外遥控器地址码 13

配置举例：

```
s_cir0_used        = 1
ir_power_key_code0  = 0x57
ir_protocol_used    = 0
```



```
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  = 0xdc
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
ir_power_key_code14 = 0x0d
ir_addr_code14      = 0xbc00
ir_power_key_code15 = 0x0d
ir_addr_code15      = 0xfc00
```


2.22 PMU 电源

2.22.1 [pmu0]

配置项	相关说明
compatible	设备名
used	是否使用 AXPxx：0: 不使用,1: 使用
pmu_id	Pmu 的 id 号

配置举例：

```
compatible      = "axpdummy"
used            = 1
pmu_id          = 0x83
```

2.22.2 [regulator0]

配置项	相关说明
regulator_count	regulator 数量
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

配置项	相关说明
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     = "axpdummy-regulator"
regulator_count = 6
regulator1     = "axpdummy_ldo1 none avcc vcc-pll vcc-tv"
regulator2     = "axpdummy_ldo2 none vcc-dram"
regulator3     = "axpdummy_ldo3 none vdd-sys vdd-ehpy"
regulator4     = "axpdummy_ldo4 none vdd-cpua"
regulator5     = "axpdummy_ldo5 none vcc-wifi"
regulator6      = "axpdummy_ldo6 none vcc-rtc vdd-cpus vcc-io vcc-efuse vcc-sdc vcc-
nand vcc-usb vcc-uartx vcc-feled vcc-key vcc-audio vcc-ir"
```


2.23 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！

2.23.1 [dvfs_table]

配置项	配置项含义
extremity_freq	极限频率
max_freq	最大频率
min_freq	最小频率
lv_count	vf 表的级数
lvn_freq	对应的最大频率 (n 表示级数)
lvn_volt	第 n 级的电压

配置举例：

```
;extremity_freq = 1344000000
max_freq = 1008000000
min_freq = 480000000

lv_count = 8
lv1_freq = 1152000000
lv1_volt = 1100

lv2_freq = 1104000000
lv2_volt = 1100

lv3_freq = 1008000000
lv3_volt = 1100

lv4_freq = 816000000
lv4_volt = 1100

lv5_freq = 624000000
lv5_volt = 1100
```



```
lv6_freq = 0
lv6_volt = 1100

lv7_freq = 0
lv7_volt = 1100

lv8_freq = 0
lv8_volt = 1100
```

2.24 CPUS

2.24.1 [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>
```

2.24.2 [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

配置项	配置项含义
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>
```

2.24.3 [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>
```


2.25 VIRTUAL DEVICE

2.25.1 [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>
```

2.26 GPU

2.26.1 [gpu_mali450_0]

配置项	配置项含义
regulator_id	作供电通道 regulator 能
dvfs_status	
scene_ctrl_status	场景控制，1 为打开
temp_ctrl_status	温控开关，1 为打开
max_level	代表有 n 级频率
begin_level	对应最大频率
lvn_freq	第 n 级的频率 (n 表示级数)
lvn_volt	第 n 级的电压

配置举例：

```
regulator_id     = "vdd-gpu"
dvfs_status      = 0
temp_ctrl_status  = 1
scene_ctrl_status = 1

max_level        = 3
begin_level      = 3

lv0_freq         = 144
lv0_volt         = 1200

lv1_freq         = 264
lv1_volt         = 1200

lv2_freq         = 384
lv2_volt         = 1200

lv3_freq         = 456
lv3_volt         = 1200
```


3. FAQ

3.1 sys_config.fex 跟 dts 配置同一个节点，会冲突吗？

答：sys_config.fex 的配置会覆盖 dts 的配置。

3.2 为什么创建跟 dts 同名的节点，但是驱动一直加载不成功？

```
example:
[test_first]
test_used  = 1
test_para  = "first_test"
```

答：这是因为该节点的 used 节点命名问题导致该节点可能没打开，used 节点的命名必须为" 节点主键"+"_used"，这样才能有效的覆盖 dts 里面 status 状态，避免 dts 里面 test_first 节点的 status 的状态为 disabled，导致该节点不可用，正确配置如下：

```
example:
[test_first]
test_first_used  = 1
test_para  = "first_test"
```


3.3 如果看到同一 pin 脚被两个节点复用，是否有问题？

答：这个问题有以下两种可能。（1）首先判断两个节点的有没有同时开启，如果没有同时开启，就不会有问题。（2）如果两个节点同时开启，需要判断以下该节点被调用的阶段是不是相同。如果两个节点被调用的阶段不同，则没问题。例如以下例子 twi 在 boot 阶段调用，uart0 在 Linux kernel 调用，则此复用不会出现问题。

```
example:
[twi]
twi_port       = 0
twi_scl        = port:PH0<2><default><default><default>
twi_sda        = port:PH1<2><default><default><default>

[uart0]
uart0_used     = 1
uart0_port     = 0
uart0_type     = 2
uart0_tx       = port:PH0<3><1><default><default>
uart0_rx       = port:PH1<3><1><default><default>
```

注意：如果两个节点被调用的阶段相同，则不允许复用。

4. 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.

