



Android 10 sys_config.fex 使用指南

版本号: 1.0
发布日期: 2020.08.06

版本历史

版本号	日期	制/修订人	内容描述
1.0	2020.08.06	AW0681	初始版本文档



目 录

1 概述	1
1.1 系统 (SYSTEM)	1
1.1.1 [product]	1
1.1.2 [platform]	1
1.1.3 [target]	2
1.1.4 [power_sply]	2
1.1.5 [card_boot]	3
1.1.6 [card0_boot_para]	3
1.1.7 [card2_boot_para]	3
1.1.8 [gpio_bias]	4
1.1.9 [uart_para]	5
1.1.10 [jtag_para]	5
1.2 clockM 配置	5
1.3 DRAM 配置	6
1.3.1 [dram_para]	6
1.4 UART	7
1.4.1 [uart0]	7
1.5 NAND FLASH	7
1.5.1 [nand0_para]	7
2 FAQ	10
2.1 1.sys_config.fex 跟 dts 配置同一个节点，会冲突吗？	10
2.2 2. 为什么创建跟 dts 同名的节点，但是驱动一直加载不成功？	10
2.3 3. 如果看到同一 pin 脚被两个节点复用，是否有问题？	10
2.4 为何 sys_config.fex 相比以往的 Android 版本配置少了这么多？	11

1 概述

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。使用于 Android 10 平台，本文以 T509 作为例子说明。

1.1 系统 (SYSTEM)

1.1.1 [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 = "evb"
```

1.1.2 [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）
debug_mode	uboot 打印等级。0=LOG_LEVEL_NONE; 1=LOG_LEVEL_ERROR; 2=LOG_LEVEL_WARNING; 3=LOG_LEVEL_NOTICE;4=LOG_LEVEL_INFO

配置举例：

```
eraseflag    = 1  
debug_mode   = 3
```

1.1.3 [target]

配置项	配置项含义
boot_clock	启动频率，单位：MHz
storage_type	启动介质选择 0：nand, 1: card0, 2: card2, -1 (default) :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	axp_type, 0:axp81X, 1:dummy, 2:axp806, 3:axp2202, 4:axp858

配置举例：

```
boot_clock      = 1008
storage_type    = -1
burn_key        = 1
dragonboard_test= 0
power_mode      = 4
```

1.1.4 [power_sply]

配置项	配置项含义
dc1sw_vol	在 uboot 阶段生效，前面三位为 100 表示开启电压设置，1003300 表示 3.3v
dc1sw_vol	dc1sw_vol 电压设置，在 uboot 阶段生效
dc1sw_vol	dc1sw_vol 电压设置，在 uboot 阶段生效
.....	xxx_vol 电压设置，在 uboot 阶段生效
dc1sw_vol	dc1sw_vol 电压设置，在 uboot 阶段生效

配置举例：

```
dc1sw_vol      = 1003300
dc1sw_vol      = 1000940
dc1sw_vol      = 1003300
aldo1_vol      = 1003300
aldo2_vol      = 1001800
aldo3_vol      = 1003300
aldo4_vol      = 1003300
aldo5_vol      = 1001800
bl1sw_vol      = 1001800
bl1sw_vol      = 1001000
bl1sw_vol      = 1001800
bl1sw_vol      = 1001800
bl1sw_vol      = 1001800
cl1sw_vol      = 1001800
cl1sw_vol      = 1002500
cl1sw_vol      = 1001200
cl1sw_vol      = 1001200
cl1sw_vol      = 1001200
dc1sw_vol      = 1003300
```

1.1.5 [card_boot]

配置项	配置项含义
logical_start	启动卡逻辑起始扇区
sprite_gpio0	卡量产，一键 recovery led 指示灯 GPIO 配置

配置举例：

```
logical_start      = 40960
sprite_gpio0      = port:PH6<1><default><default><1>
```

1.1.6 [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
sd_c_d1	sd_c 卡数据 1 线信号的 GPIO 配置
sd_c_d0	sd_c 卡数据 0 线信号的 GPIO 配置
sd_c_clk	sd_c 卡时钟信号的 GPIO 配置
sd_c_cmd	sd_c 命令信号的 GPIO 配置
sd_c_d3	sd_c 卡数据 3 线信号的 GPIO 配置
sd_c_d2	sd_c 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
sd_c_d1            = port:PF0<2><1><3><default>
sd_c_d0            = port:PF1<2><1><3><default>
sd_c_clk           = port:PF2<2><1><3><default>
sd_c_cmd           = port:PF3<2><1><3><default>
sd_c_d3            = port:PF4<2><1><3><default>
sd_c_d2            = port:PF5<2><1><3><default>
```

1.1.7 [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
sdcsds	ds 信号的 GPIO 配置
sdcd1	sdcs 卡数据 1 线信号的 GPIO 配置
sdcd0	sdcs 卡数据 0 线信号的 GPIO 配置
sdccclk	sdcs 卡时钟信号的 GPIO 配置
sdcccmd	sdcs 命令信号的 GPIO 配置
sdcd3	sdcs 卡数据 3 线信号的 GPIO 配置
sdcd2	sdcs 卡数据 2 线信号的 GPIO 配置
sdcd4	sdcs 卡数据 4 线信号的 GPIO 配置
sdcd5	sdcs 卡数据 5 线信号的 GPIO 配置
sdcd6	sdcs 卡数据 6 线信号的 GPIO 配置
sdcd7	sdcs 卡数据 7 线信号的 GPIO 配置
sdcemmc_rst	emmc_rst 信号的 GPIO 配置
sdcs_ex_dly_used	ex_dly_used 信号
sdcs_io_1v8	sdcs_io_1v8 高速 emmc 模式配置

配置举例：

```
card_ctrl      = 2
card_high_speed = 1
card_line      = 8
sdcsclk        = port:PC5<3><1><3><default>
sdccmd         = port:PC6<3><1><3><default>
sdcd0          = port:PC10<3><1><3><default>
sdcd1          = port:PC13<3><1><3><default>
sdcd2          = port:PC15<3><1><3><default>
sdcd3          = port:PC8<3><1><3><default>
sdcd4          = port:PC9<3><1><3><default>
sdcd5          = port:PC11<3><1><3><default>
sdcd6          = port:PC14<3><1><3><default>
sdcd7          = port:PC16<3><1><3><default>
sdcemmc_rst    = port:PC1<3><1><3><default>
sdcsds         = port:PC0<3><2><3><default>
sdcs_ex_dly_used = 2
sdcs_io_1v8    = 1
```

1.1.8 [gpio_bias]

配置项	配置项含义
pc_bias	emmc 电压配置，高速 emmc 才使用

配置举例：

```
pc_bias = 1800
```

1.1.9 [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:PB09<2><1><default><default>
uart_debug_rx   = port:PB10<2><1><default><default>
```

1.1.10 [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<3><default><default><default>
jtag_ck     = port:PH10<3><default><default><default>
jtag_do     = port:PH11<3><default><default><default>
jtag_di     = port:PH12<3><default><default><default>
```

1.2 clockM 配置

配置项	配置项含义
pll4	pll4 时钟设置
pll6	pll6 时钟设置

配置项	配置项含义
pll8	pll8 时钟设置
pll9	pll9 时钟设置
pll10	pll10 时钟设置

配置举例：

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

1.3 DRAM 配置

1.3.1 [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：8 为 LPDDR4，由源厂调节，请勿修改
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，由源厂调节，请勿修改
dram_mr0	DRAM CAS 值，可为 6,7,8,9；由源厂调节，请勿修改
dram_xxx	由源厂调节，请勿修改

配置举例：

```
dram_clk      = 720
dram_type     = 3
dram_dx_odt   = 0x07070707
dram_dx_dri   = 0x0c0c0c0c
dram_ca_dri   = 0x0e0e
dram_para0    = 0x11121313
dram_para1    = 0x30FA
dram_para2    = 0x0000
dram_mr0      = 0x840
dram_mr1      = 0x4
dram_mr2      = 0x8
dram_mr3      = 0x0
dram_mr4      = 0x0
dram_mr5      = 0x0
dram_mr6      = 0x0
dram_mr11     = 0x0
dram_mr12     = 0x0
dram_mr13     = 0x0
```

```
dram_mr14    = 0x0
dram_mr16    = 0x0
dram_mr17    = 0x0
dram_mr22    = 0x0
dram_tpr0    = 0x0
dram_tpr1    = 0x0
dram_tpr2    = 0x0
dram_tpr3    = 0x0
dram_tpr6    = 0x33808080
dram_tpr10   = 0x002f7777
dram_tpr11   = 0x0d0e120f
dram_tpr12   = 0x14131414
dram_tpr13   = 0x40
dram_tpr14   = 0x1f1e1b1f
```

1.4 UART

1.4.1 [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:PB09<2><1><default><default>
uart0_rx      = port:PB10<2><1><default><default>
```

1.5 NAND FLASH

1.5.1 [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

配置项	配置项含义
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	/
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           = 0
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><0><1><default>
nand0_rb0            = port:PC05<2><1><1><default>
nand0_d0             = port:PC06<2><0><1><default>
nand0_d1             = port:PC07<2><0><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
```



2 FAQ

2.1 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。

2.2 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"
```

2.3 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。

2.4 为何 sys_config.fex 相比以往的 Android 版本配置少了这么多？

答：自 Android 10 开始，使用 longan 设计以后，大部分 BSP 的配置将不再体现在 sys_config.fex，而是写到 board.dts 文件，但是 sys_config.fex 依然保留给客户配置，方法不变。






著作权声明

珠海全志科技股份有限公司。保留一切权利。

本文档及内容受著作权法保护，其著作权由珠海全志科技股份有限公司（“全志”）拥有并保留一切权利。

本文档是全志的原创作品和版权财产，未经全志书面许可，任何单位和个人不得擅自摘抄、复制、修改、发表或传播本文档内容的部分或全部，且不得以任何形式传播。

商标声明

、 全志科技 （不完全列举）均为珠海全志科技股份有限公司的商标或者注册商标。在本文档描述的产品中出现的其它商标，产品名称，和服务名称，均由其各自所有人拥有。

免责声明

您购买的产品、服务或特性应受您与珠海全志科技股份有限公司（“全志”）之间签署的商业合同和条款的约束。本文档中描述的全部或部分产品、服务或特性可能不在您所购买或使用的范围内。使用前请认真阅读合同条款和相关说明，并严格遵循本文档的使用说明。您将自行承担任何不当使用行为（包括但不限于如超压，超频，超温使用）造成的不利后果，全志概不负责。

本文档作为使用指导仅供参考。由于产品版本升级或其他原因，本文档内容有可能修改，如有变更，恕不另行通知。全志尽全力在本文档中提供准确的信息，但并不确保内容完全没有错误，因使用本文档而发生损害（包括但不限于间接的、偶然的、特殊的损失）或发生侵犯第三方权利事件，全志概不负责。本文档中的所有陈述、信息和建议并不构成任何明示或暗示的保证或承诺。

本文档未以明示或暗示或其他方式授予全志的任何专利或知识产权。在您实施方案或使用产品的过程中，可能需要获得第三方的权利许可。请您自行向第三方权利人获取相关的许可。全志不承担也不代为支付任何关于获取第三方许可的许可费或版税（专利税）。全志不对您所使用的第三方许可技术做出任何保证、赔偿或承担其他义务。