



# Linux MMC 开发指南

版本号: 3.3  
发布日期: 2023.11.24

## 版本历史

| 版本号 | 日期         | 制/修订人   | 内容描述  |
|-----|------------|---------|---|
| 1.0 | 2021.11.23 | AWA0332 | 初始版本  |
| 2.1 | 2021.11.25 | AWA0332 | 添加版本历史, 适用范围改用表格, 修复调试节点信息, 添加 dts 或者 sysconfig 一些需要客户自行确认修改的项目 |
| 2.2 | 2021.11.26 | AWA0332 | 添加封面, 修改配置项目格式混乱问题, 修改常见问题的格式问题, 删除多余的版本历史                      |
| 2.3 | 2021.12.21 | AWA0332 | 去掉名字版本号和空格问题  |
| 2.4 | 2022.2.18  | AWA1579 | 适配 linux4.9 配置, 并且进行节点修改  |
| 2.5 | 2022.3.11  | AWA1767 | 增加一些常用的 dts 配置项说明   |
| 2.6 | 2022.4.18  | AWA1579 | 增加性能测试节点说明  |
| 2.7 | 2022.7.11  | AWA1767 | 增加新的适用平台  |
| 2.8 | 2022.11.3  | AWA1767 | 根据评审意见修改格式  |
| 2.9 | 2022.11.28 | AWA1767 | 根据评审意见修改格式和内容   |
| 3.0 | 2023.03.09 | AWA1767 | 更新 A523 新内容   |
| 3.1 | 2023.04.26 | AWA1767 | 增加 A523 相关平台适配  |
| 3.2 | 2023.05.23 | AWA1579 | 增加 A523 相关平台选择适配  |
| 3.3 | 2023.11.24 | AWA1767 | 1、增加 A523 相关平台选择适配 2、增加耐压模式说明                                   |

# 目 录

|  |           |
|--|-----------|
| <b>1 前言</b>                                  | <b>1</b>  |
| 1.1 文档简介                                     | 1         |
| 1.2 目标读者                                     | 1         |
| 1.3 适用范围                                     | 1         |
| <b>2 CMDQ 适用平台</b>                           | <b>2</b>  |
| <b>3 模块介绍</b>                                | <b>3</b>  |
| 3.1 模块功能介绍                                   | 3         |
| 3.2 相关术语介绍                                   | 3         |
| 3.2.1 硬件术语                                   | 3         |
| 3.2.2 软件术语                                   | 3         |
| 3.3 模块配置介绍                                   | 4         |
| 3.3.1 sys_config.fex 配置说明                    | 4         |
| 3.3.2 Device Tree 配置说明                       | 5         |
| 3.3.2.1 uboot 阶段                             | 5         |
| 3.3.2.2 内核阶段                                 | 7         |
| 3.3.3 kernel menuconfig 配置说明（linux5.4 及其以前）  | 12        |
| 3.3.4 kernel menuconfig 配置说明（linux5.10 及其以后） | 14        |
| 3.4 源码结构介绍                                   | 19        |
| <b>4 FAQ</b>                                 | <b>20</b> |
| 4.1 调试方法                                     | 20        |
| 4.1.1 调试节点                                   | 20        |
| 4.1.1.1 寄存器信息                                | 20        |
| 4.1.1.2 emmc 信息                              | 21        |
| 4.1.1.3 性能验证节点                               | 22        |
| 4.2 常见问题                                     | 23        |

## 插图

|        |                                     |    |
|--------|-------------------------------------|----|
| 图 3-1  | menuconfig 主界面 . . . . .            | 13 |
| 图 3-2  | Device drivers 界面 . . . . .         | 13 |
| 图 3-3  | sdmmc 支持界面 . . . . .                | 14 |
| 图 3-4  | menuconfig 主界面 . . . . .            | 14 |
| 图 3-5  | Device Drivers . . . . .            | 15 |
| 图 3-6  | MMC/SD/SDIO card support . . . . .  | 16 |
| 图 3-7  | menuconfig 主界面 . . . . .            | 17 |
| 图 3-8  | menuconfig Allwinner BSP . . . . .  | 17 |
| 图 3-9  | menuconfig Device Drivers . . . . . | 18 |
| 图 3-10 | menuconfig SD/MMC Drivers . . . . . | 19 |
| 图 4-1  | retry . . . . .                     | 20 |
| 图 4-2  | sunxi_host_perf . . . . .           | 22 |

# 1 前言

## 1.1 文档简介

介绍 Linux 内核中 SD/MMC 子系统的接口及使用方法，为 SD/MMC 设备驱动的开发提供参考。

## 1.2 目标读者

SD/MMC 驱动的开发/维护人员。

## 1.3 适用范围

| 产品名称           | 内核版本       | 驱动文件       |
|----------------|------------|------------|
| A133           | Linux-5.4  | sunxi_mmc* |
| T113           | Linux-5.4  | sunxi_mmc* |
| h616           | Linux-5.4  | sunxi_mmc* |
| v853           | Linux-4.9  | sunxi_mmc* |
| R853           | Linux-4.9  | sunxi_mmc* |
| F133           | Linux-5.4  | sunxi_mmc* |
| T5             | Linux-5.10 | sunxi_mmc* |
| T3/T3-C/T3-Pro | Linux-5.10 | sunxi_mmc* |
| A40i-H/A40i-C  | Linux-5.10 | sunxi_mmc* |
| A133           | Linux-5.15 | sunxi_mmc* |
| A523           | Linux-5.15 | sunxi_mmc* |
| MR527          | Linux-5.15 | sunxi_mmc* |
| A527           | Linux-5.15 | sunxi_mmc* |
| T527           | Linux-5.15 | sunxi_mmc* |
| AI985          | Linux-5.15 | sunxi_mmc* |

## 2 CMDQ 适用平台

| 产品名称  | 内核版本       |
|-------|------------|
| A523  | Linux-5.15 |
| MR527 | Linux-5.15 |
| A527  | Linux-5.15 |
| T527  | Linux-5.15 |
| AI985 | Linux-5.15 |

## 3 模块介绍

### 3.1 模块功能介绍

Linux 提供了 MMC 子系统来实现对各种 SD/MMC/EMMC/SDIO 设备访问，MMC 子系统由上到下可以分为三层，MMC/SD card 层，MMC/SD core 层以及 MMC/SD host 层，它们之间的层次关系如下所示。

MMC/SD card 层负主要是按照 LINUX 块设备驱动程序的框架实现一个卡的块设备驱动。负责块设备请求的处理，以及请求队列的管理。

MMC/SD core 层负责通信协议的处理，包括 SD/MMC/eMMC/SDIO，为上一层提供具体读写接口，同时为下一层提供 host 端接口。

MMC/SD host 层是实现对 SD/MMC 控制器相关的操作，直接操作硬件，也是主要实现部分。

### 3.2 相关术语介绍

#### 3.2.1 硬件术语

| 术语    | 解释说明                       |
|-------|----------------------------|
| Sunxi | 指 Allwinner 的一系列 SOC 硬件平台。 |
| SD    | Secure Digital Memory Card |
| MMC   | Multimedia Card            |
| eMMC  | Embedded MultiMediaCard    |
| host  | 指具体的 SD/MMC 控制器            |

#### 3.2.2 软件术语

无

## 3.3 模块配置介绍

### 3.3.1 sys\_config.fex 配置说明

#### [card0\_boot\_para]

```
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>
```

各个配置项的意义如下：

| 配置项             | 配置项含义                |
|-----------------|----------------------|
| card_ctrl       | 0：选择卡量产相关的控制器        |
| card_high_speed | 速度模式 0 为低速，1 为高速     |
| card_line       | 代表卡总线宽度，分别有 1,4,8    |
| sd_c_d1         | sd_c data1 的 GPIO 配置 |
| sd_c_d0         | sd_c data0 的 GPIO 配置 |
| sd_c_clk        | sd_c clk 的 GPIO 配置   |
| sd_c_cmd        | sd_c cmd 的 GPIO 配置   |
| sd_c_d3         | sd_c data3 的 GPIO 配置 |
| sd_c_d2         | sd_c data2 的 GPIO 配置 |

#### [card2\_boot\_para]

```
card_ctrl    = 2
card_high_speed = 1
card_line    = 8
sd_c_clk     = port:PC5<3><1><3><default>
sd_c_cmd     = port:PC6<3><1><3><default>
sd_c_d0      = port:PC10<3><1><3><default>
sd_c_d1      = port:PC13<3><1><3><default>
sd_c_d2      = port:PC15<3><1><3><default>
sd_c_d3      = port:PC8<3><1><3><default>
sd_c_d4      = port:PC9<3><1><3><default>
sd_c_d5      = port:PC11<3><1><3><default>
sd_c_d6      = port:PC14<3><1><3><default>
sd_c_d7      = port:PC16<3><1><3><default>
sd_c_emmc_rst = port:PC1<3><1><3><default>
sd_c_ds      = port:PC0<3><2><3><default>
sd_c_ex_dly_used = 2
sd_c_io_1v8   = 1
```

各个配置项的意义如下：



| 配置项                    | 配置项含义   |
|------------------------|---|
| card_ctrl              | 0: 选择卡量产相关的控制器  |
| card_high_speed        | 速度模式 0 为低速, 1 为高速   |
| card_line              | 代表卡总线宽度, 分别有 1,4,8  |
| sdclk                  | sdclk 的 GPIO 配置   |
| sdcmd                  | sdcmd 的 GPIO 配置   |
| sdcd0                  | sdcd data0 的 GPIO 配置  |
| sdcd1                  | sdcd data1 的 GPIO 配置  |
| sdcd2                  | sdcd data2 的 GPIO 配置  |
| sdcd3                  | sdcd data3 的 GPIO 配置  |
| sdcd4                  | sdcd data4 的 GPIO 配置  |
| sdcd5                  | sdcd data5 的 GPIO 配置  |
| sdcd6                  | sdcd data6 的 GPIO 配置  |
| sdcd7                  | sdcd data7 的 GPIO 配置  |
| sdcmmc_rst             | emmc 复位信号的 GPIO 配置  |
| sdcds                  | sdcds 线的 GPIO 配置  |
| sdcd_ex_dly_used       | 采样模式控制, 2: tune 采样点; 1: 固定采样点方式, 烧写阶段和启动阶段, 通过 sys_config 配置采样点; 其它值: 烧写阶段和启动阶段使用预设的采样点, 通常用 2, 不建议修改                                 |
| sdcd_io_1v8            | 1: 表示 eMMC IO 电平是 1.8V, 需要根据实际 emmc io 供电配置   |
| sdcd_force_boot_tuning | 1: 强制启动 tuning  |
| sdcd_tm4_win_th        | Tune 采样最小可选窗口   |
| sdcd_dis_host_caps     | 禁止某一种速度模式, bit1: Host 不支持 HS-SDR; Bit3: Host 不支持 8 线模式; bit6: Host 不支持 HS-DDR; bit7: Host 不支持 HS200; bit8: Host 不支持 HS400; 其他值, 不建议使用 |

## 3.3.2 Device Tree 配置说明

### 3.3.2.1 uboot 阶段

放到板级目录下: uboot-board.dts

#### 3.3.2.1.1 sdc0

```
&sdc0_pins_a {
    allwinner,pins = "PF0", "PF1", "PF2",
        "PF3", "PF4", "PF5";
    allwinner,function = "sdc0";
    allwinner,muxsel = <2>;
    allwinner,drive = <3>;
```

```
allwinner,pull = <1>;  
};  
  
&card0_boot_para {  
    /* reg = <0x0 0x2 0x0 0x0>; */  
    device_type = "card0_boot_para";  
    card_ctrl = <0x0>;  
    card_high_speed = <0x1>;  
    card_line = <0x4>;  
    pinctrl-0 = <&sdc0_pins_a>;  
};
```

| 配置项             | 配置项含义   |
|-----------------|---|
| card_ctrl       | 0: 选择卡量产相关的控制器  |
| card_high_speed | 速度模式 0 为低速, 1 为高速   |
| card_line       | 代表卡总线宽度, 分别有 1,4,8  |
| pinctrl-0       | 代表卡的 pin 设置   |
| sdc0_pins_a     | 具体卡的 pin 设置, allwinner,pins 代表具体的 pin 名字,<br>allwinner,function 表示 pin 选择的功能, 这里选择 sdc0,<br>allwinner,muxsel 代表 sdc0 对应 spec 里面的功能值, allwinner,drive 代<br>表驱动能力, allwinner,pull 代表上下拉 |

### 3.3.2.1.2 sdc2

```
&sdc2_pins_a {  
    allwinner,pins = "PC1", "PC5", "PC6",  
        "PC8", "PC9", "PC10", "PC11",  
        "PC13", "PC14", "PC15", "PC16";  
    allwinner,function = "sdc2";  
    allwinner,muxsel = <3>;  
    allwinner,drive = <3>;  
    allwinner,pull = <1>;  
};  
  
&sdc2_pins_b {  
    allwinner,pins = "PC0", "PC1", "PC5", "PC6",  
        "PC8", "PC9", "PC10", "PC11",  
        "PC13", "PC14", "PC15", "PC16";  
    allwinner,function = "io_disabled";  
    allwinner,muxsel = <7>;  
    allwinner,drive = <1>;  
    allwinner,pull = <1>;  
};  
  
&sdc2_pins_c {  
    allwinner,pins = "PC0";  
    allwinner,function = "sdc2";  
    allwinner,muxsel = <3>;  
    allwinner,drive = <3>;  
    allwinner,pull = <2>;  
};
```

```

&card2_boot_para {
    /* reg = <0x0 0x3 0x0 0x0>; */
    device_type = "card2_boot_para";
    card_ctrl = <0x2>;
    card_high_speed = <0x1>;
    card_line = <0x8>;
    pinctrl-0 = <&sdc2_pins_a &sdc2_pins_c>;
    sdc_ex_dly_used = <0x2>;
    sdc_io_1v8 = <0x1>;
    sdc_tm4_win_th = <0x08>;
    sdc_tm4_hs200_max_freq = <150>;
    sdc_tm4_hs400_max_freq = <100>;
    sdc_type = "tm4";
};

```

| 配置项   | 配置项含义   |
|---|---|
| card_ctrl   | 0: 选择卡量产相关的控制器  |
| card_high_speed                                   | 速度模式 0 为低速, 1 为高速   |
| card_line   | 代表卡总线宽度, 分别有 1,4,8  |
| sdc_ex_dly_used                                   | 采样模式控制, 2: tune 采样点; 1: 固定采样点方式, 烧写阶段和启动阶段, 通过 sys_config 配置采样点; 其它值: 烧写阶段和启动阶段使用预设的采样点, 通常用 2, 不建议修改   |
| sdc_io_1v8  | 1: 表示 eMMC IO 电平是 1.8V, 需要根据实际 emmc io 供电配置   |
| sdc_tm4_win_th                                    | Tune 采样最小可选窗口   |
| sdc_tm4_hs200_max_freq/<br>sdc_tm4_hs400_max_freq | 代表 emmc 的 hs200/hs400 最大频率设置, 不建议修改   |
| sdc2_pins_a,sdc2_pins_c                           | 具体卡的 pin 设置, allwinner,pins 代表具体的 pin 名字, allwinner,function 表示 pin 选择的功能, 这里选择 sdc0, allwinner,muxsel 代表 sdc0 对应 spec 里面的功能值, allwinner,drive 代表去掉能力, allwinner,pull 代表上下拉 |

### 3.3.2.2 内核阶段

linux5.4 以前内核版本存放在 board.dts 或者内核目录下面 arch/armXX/boot/dts/sunxi/sunxiIiwXpX 中

linux5.10 以后内核版本存放在 board.dts 或者 bsp 仓库下面 configs/linux-X/sunxiIiwXpX 中

在不同的 Sunxi 硬件平台中, SD/MMC 控制器的数目也不一定相同, 但对于每一个 SD/MMC 控制器来说, 在 board.dts 中配置参数相似。

各个控制器的配置以及属性的意义如下:

### 3.3.2.2.1 [sdc0 (in dts)]

通常用作 SD 卡

```
sdc0: sdmmc@04020000 {
    device_type = "sdc0";
    cd-used-24M;
    disable-wp;
    pinctrl-0 = <&sdc0_pins_a>;
    bus-width = <4>;
    cd-gpios = <&pio PF 6 1 3 0xffffffff>;
    /*non-removable;*/
    /*broken-cd;*/
    /*cd-inverted*/
    /*data3-detect;*/
    cap-sd-highspeed;
    sd-uhs-sdr50;
    sd-uhs-ddr50;
    sd-uhs-sdr104;
    no-sdio;
    no-mmc;
    sunxi-power-save-mode;
    /*sunxi-dis-signal-vol-sw;*/
    max-frequency = <150000000>;
    ctl-spec-caps = <0x8>;
    vmmc-supply = <&reg_dldo1>;
    vqmmc33sw-supply = <&reg_dldo1>;
    vdmmc33sw-supply = <&reg_dldo1>;
    vqmmc18sw-supply = <&reg_aldo1>;
    vdmmc18sw-supply = <&reg_aldo1>;
    status = "okay";
};
```

各个配置项的意义如下：

| 配置项              | 配置项含义              |
|------------------|--------------------|
| device_type      | phy 索引值的选择         |
| cd-used-24M      | 使用 24M 时钟检测插拔卡中断   |
| disable-wp       | 卡设置写保护，ro          |
| pinctrl-0        | 第一组 pin 脚的 GPIO 配置 |
| bus-width        | 线宽                 |
| cd-gpios         | 卡检测的 GPIO 配置       |
| non-removable    | 不可移除               |
| broken-cd        | sd 卡检测方式：轮训        |
| cd-inverted      | 卡检测的高电平有效还是低电平有效   |
| data3-detect     | data3 线检测卡         |
| cap-sd-highspeed | SD 卡的 High speed   |
| sd-uhs-sdr50     | SD 卡的 uhs-sdr50    |
| sd-uhs-ddr50     | SD 卡的 uhs-ddr50    |
| sd-uhs-sdr104    | SD 卡的 uhs-sdr104   |
| no-sdio          | 无 sdio             |
| no-mmc           | 无 mmc              |

| 配置项                     | 配置项含义                          |
|-------------------------|--------------------------------|
| sunxi-power-save-mode   | 发送数据或者命令才有时钟输出                 |
| sunxi-dis-signal-vol-sw | 关闭电压切换                         |
| max-frequency           | 最大频率                           |
| ctl-spec-caps           | 控制 spec 能力                     |
| vmmc-supply             | 供电电压、工作电压，需要根据实际 pmu 供电方案修改    |
| vqmmc33sw-supply        | 3.3V 的 IO 电压，需要根据实际 pmu 供电方案修改 |
| vdmmc33sw-supply        | 3.3V 的卡检测电压，需要根据实际 pmu 供电方案修改  |
| vqmmc18sw-supply        | 1.8V 的 IO 电压，需要根据实际 pmu 供电方案修改 |
| vdmmc18sw-supply        | 1.8V 的卡检测电压，需要根据实际 pmu 供电方案修改  |
| status                  | 设备树的状态                         |

### 3.3.2.2.2 [sdci1 (in dts)]

通常用作 SDIO WIFI

```
sdci1: sdmmc@04021000 {
    pinctrl-0 = <&sdci0_pins_a>;
    bus-width = <4>;
    cap-sd-highspeed;
    sd-uhs-sdr50;
    sd-uhs-ddr50;
    sd-uhs-sdr104;
    no-sd;
    no-mmc;
    /*sunxi-power-save-mode;*/
    /*sunxi-dis-signal-vol-sw;*/
    cap-sdio-irq;
    keep-power-in-suspend;
    ignore-pm-notify;
    max-frequency = <150000000>;
    ctl-spec-caps = <0x8>;
    sunxi-dly-208M = <1 1 0 0 0 1>;
    vmmc-supply = <&reg_dldo1>;
    vqmmc33sw-supply = <&reg_dldo1>;
    vdmmc33sw-supply = <&reg_dldo1>;
    vqmmc18sw-supply = <&reg_aldo1>;
    vdmmc18sw-supply = <&reg_aldo1>;
    status = "okay";
};
```

各个配置项的意义如下：

| 配置项              | 配置项含义              |
|------------------|--------------------|
| pinctrl-0        | 第一组 pin 脚的 GPIO 配置 |
| bus-width        | 线宽                 |
| cap-sd-highspeed | SDIO 卡的 High speed |
| sd-uhs-sdr50     | SDIO 卡的 uhs-sdr50  |
| sd-uhs-ddr50     | SDIO 卡的 uhs-ddr50  |

| 配置项                     | 配置项含义  |
|-------------------------|--|
| sd-uhs-sdr104           | SDIO 卡的 uhs-sdr104   |
| no-sd                   | 无 sd   |
| no-mmc                  | 无 mmc  |
| sunxi-power-save-mode   | 发送数据或者命令才有时钟输出   |
| sunxi-dis-signal-vol-sw | 关闭电压切换   |
| cap-sdio-irq            | 开启 SDIO 中断   |
| keep-power-in-suspend   | 休眠时保持电源  |
| ignore-pm-notify        | 忽略电源管理的通知  |
| max-frequency           | 最大频率   |
| ctl-spec-caps           | 控制 spec 能力   |
| sunxi-dly-208M          | <1(cmd driver phase) 1(data driver phase) 0<br>0 0(data sample phase) 0(cmd sample phase)> |
| vmmc-supply             | 供电电压、工作电压，需要根据实际 pmu 供电方案修改  |
| vqmmc33sw-supply        | 3.3V 的 IO 电压，需要根据实际 pmu 供电方案修改   |
| vdmmc33sw-supply        | 3.3V 的卡检测电压，需要根据实际 pmu 供电方案修改  |
| vqmmc18sw-supply        | 1.8V 的 IO 电压，需要根据实际 pmu 供电方案修改   |
| vdmmc18sw-supply        | 1.8V 的卡检测电压，需要根据实际 pmu 供电方案修改  |
| status                  | 设备树的状态   |

### 3.3.2.2.3 [sdcard (in dts)]

通常用作 eMMC

```
sdc2: sdmmc@04022000 {
    pinctrl-0 = <&sdcard_pins_a &sdcard_pins_c>;
    bus-width = <8>;
    non-removable;
    cap-mmc-hs200;
    mmc-ddr-1_8v;
    mmc-hs200-1_8v;
    mmc-hs400-1_8v;
    no-sdio;
    no-sd;
    cqe-on;
    ctl-cmdq-md = <0x2>;
    sunxi-power-save-mode;
    sunxi-dis-signal-vol-sw;
    max-frequency = <100000000>;
    ctl-spec-caps = <0x308>;
    vmmc-supply = <&reg_dldo1>;
    vqmmc-supply = <&reg_aldo1>;
    fixed-emmc-driver-type = <0x1>;
    sdc_tm4_sm0_freq0 = <0>;
    status = "disabled";
};
```

各个配置项的意义如下：



| 配置项                     | 配置项含义   |
|-------------------------|---|
| pinctrl-0               | 第一组 pin 脚的 GPIO 配置                              |
| bus-width               | 线宽  |
| non-removable           | 不可移除  |
| cap-mmc-highspeed       | MMC 卡的 High speed                               |
| mmc-ddr-1_8v            | MMC 卡的 ddr50                                    |
| mmc-hs200-1_8v          | MMC 卡的 hs200                                    |
| mmc-hs400-1_8v          | MMC 卡的 hs400                                    |
| no-sdio                 | 无 sdio  |
| no-sd                   | 无 sd  |
| ctl-cmdq-md             | cmdq 模式设置寄存器，详见下 (1) 注释                         |
| cap-hsq                 | 和 cqe-on 二选一，指使用软件队列来派发 mmc 请求，emmc 和 sd 卡均可使用  |
| cqe-on                  | emmc cmdq 使能开关 (2) (3)，定义即为使能，emmc 才能使用         |
| sunxi-power-save-mode   | 发送数据或者命令才有时钟输出                                  |
| sunxi-dis-signal-vol-sw | 关闭电压切换  |
| max-frequency           | 最大频率  |
| vmmc-supply             | 供电电压、工作电压，需要根据实际 pmu 供电方案修改                     |
| vqmmc-supply            | IO 电压，需要根据实际 pmu 供电方案修改                         |
| fixed-emmc-driver-type  | 驱动能力等级调整，对应设置 Extended CSD register 中 HS_TIMING |
| sdhci_tm4_sm0_freqn     | host timing setting                             |
| status                  | 设备树的状态  |

注：(1) cmdq 模式设置支持配置范围是 1 和 2。1 表示 cmdq\_half\_mode，cmd 和 data 无法同时传输，但是当前任务可以在上一个任务传输完成前送到 device；2 表示 cmdq\_almost\_mode，cmd 可以在 block 开始时被传输，当前任务可以在上一个任务传输完成前送到 device。

(2) cqe-on 能为并行场景带来随机读写 io 的总性能的提高，但是由于 cmdq 的协议变的复杂，某些性能可能会下降；所以客户在选择配置时，可以根据方案是否存在并行 io 场景来决定实际的配置。cap-hsq 能保证单笔随机 io 性能前提下，降低软件消耗，提高并行 io 性能；所以在实际使用中可以根据不同的物料表现和使用场景和性能数据进行二选一。一般原则 android 默认选用 cmdq，linux 平台默认选用 hsq。

(3) 仅能将默认配置为 cqe-on 的配置改成 cap-hsq，不能将默认配置为 cap-hsq 的平台改成 cqe-on。

## 耐压配置

linux5.4 以上的版本，需要在内核进行耐压模式的配置，在 board.dts 的 pio 节点增加你对应使用引脚的耐压模式配置 vcc-px-supply = <&reg\_piox\_x>;，如下所示：

```
&pio {
    vcc-pg-supply = <&reg_pio1_8>;
    vcc-pc-supply = <&reg_pio1_8>;
    vcc-pf-supply = <&reg_pio1_8>;
}
```

```
vcc-pfo-supply = <&reg_pio3_3>;
```

```
.....
```

vcc-px-supply 则对应你要配置的 gpio 口 px 的耐压值，后面的 reg\_piox\_x 则需要根据实际的硬件方案确定该口的耐压值，如 PC 口，如果你的 VCC-PC 是 1.8V 供电，则配置为 vcc-pc-supply = <&reg\_pio1\_8>，如果是 3.3V 供电则配置为 vcc-pc-supply = <&reg\_pio3\_3>

注意，上述的 PF 口比较特殊，因为存在切换电压的过程，所以会多一个 vcc-pfo-supply 的关键字。对于 PF 口配置耐压模式的配置不用修改，始终如下配置：

```
vcc-pf-supply = <&reg_pio1_8>;  
vcc-pfo-supply = <&reg_pio3_3>;
```

PF 口的电压切换会根据使用引脚的 dts 配置 pin 的 power-source(如下) 来决定，如果是 3.3V 就配置 power-source 等于 3300,1.8V 则为 1800。

pin 的 power-source 软件会根据使用的 sd 速度模式自动切换使用，正常情况下这个不需要额外配置。

```
sdc0_pins_a: sdc0@0 {  
    pins = "PF0", "PF1", "PF2",  
           "PF3", "PF4", "PF5";  
    function = "sdc0";  
    drive-strength = <40>;  
    bias-pull-up;  
    -----> power-source = <3300>;  
};  
  
sdc0_pins_b: sdc0@1 {  
    pins = "PF0", "PF1", "PF2",  
           "PF3", "PF4", "PF5";  
    function = "sdc0";  
    drive-strength = <40>;  
    bias-pull-up;  
    -----> power-source = <1800>;  
};
```

### 3.3.3 kernel menuconfig 配置说明 (linux5.4 及其以前)

在命令行中进入内核 longan 根目录，执行 ./build.sh menuconfig 进入配置界面，并按以下步骤操作：

1.menuconfig 主界面。



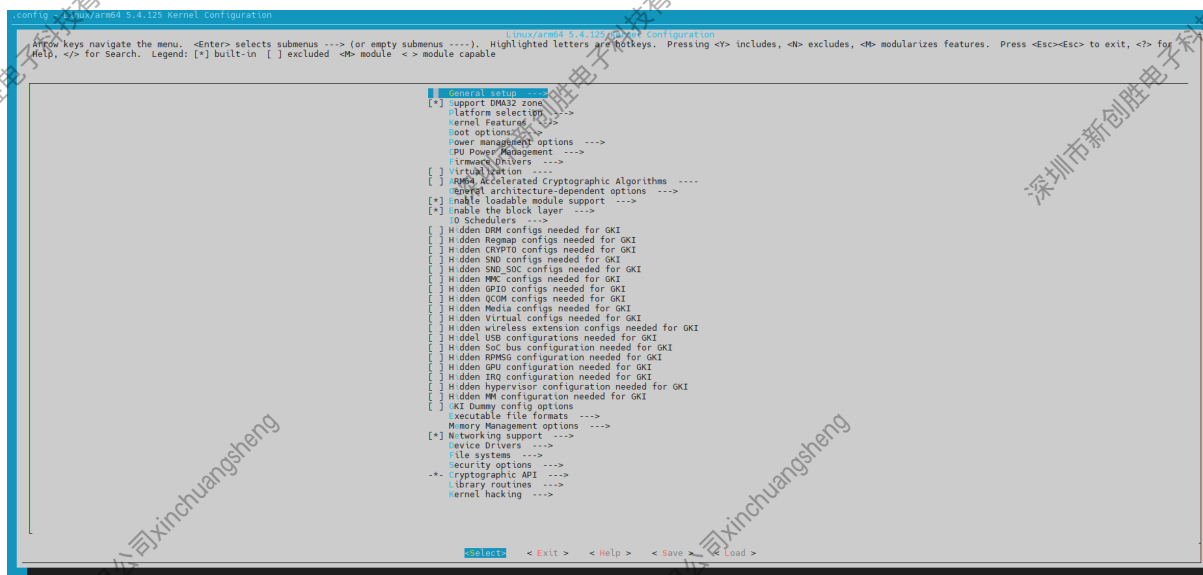


图 3-1: menuconfig 主界面

2. 进入 Device Drivers，并选中 MMC/SD/SDIO card support 为 “\*”。

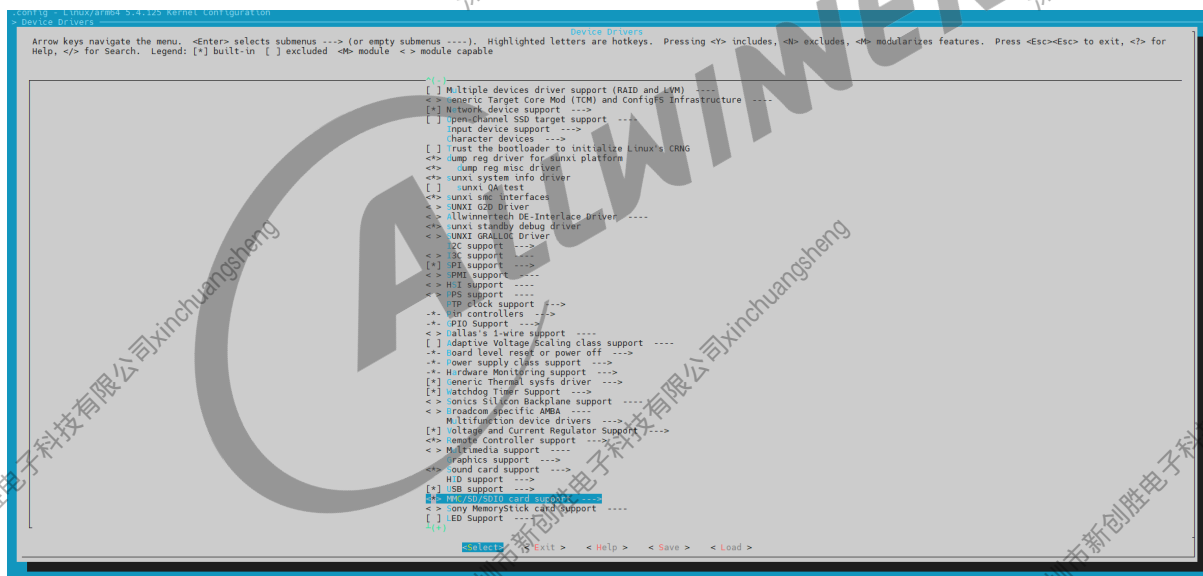


图 3-2: Device drivers 界面

3. 选择 Allwinner sunxi SD/MMC Host Controller support 为 “\*”，编译进内核（M 为编译进模块）。

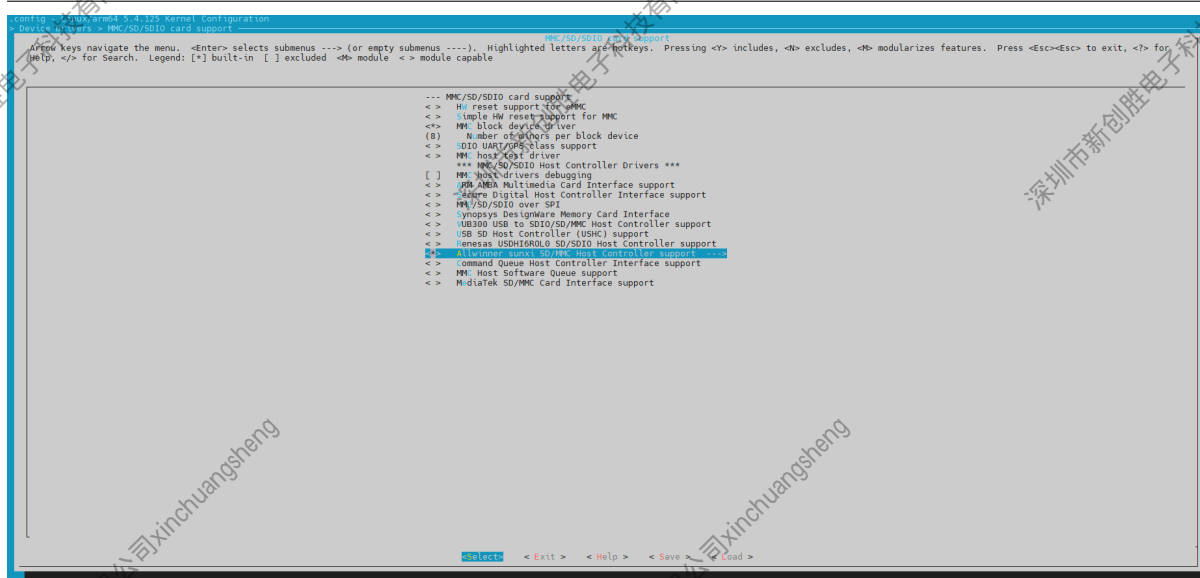


图 3-3: sdmmc 支持界面

### 3.3.4 kernel menuconfig 配置说明（linux5.10 及其以后）

#### 1. menuconfig 主界面。

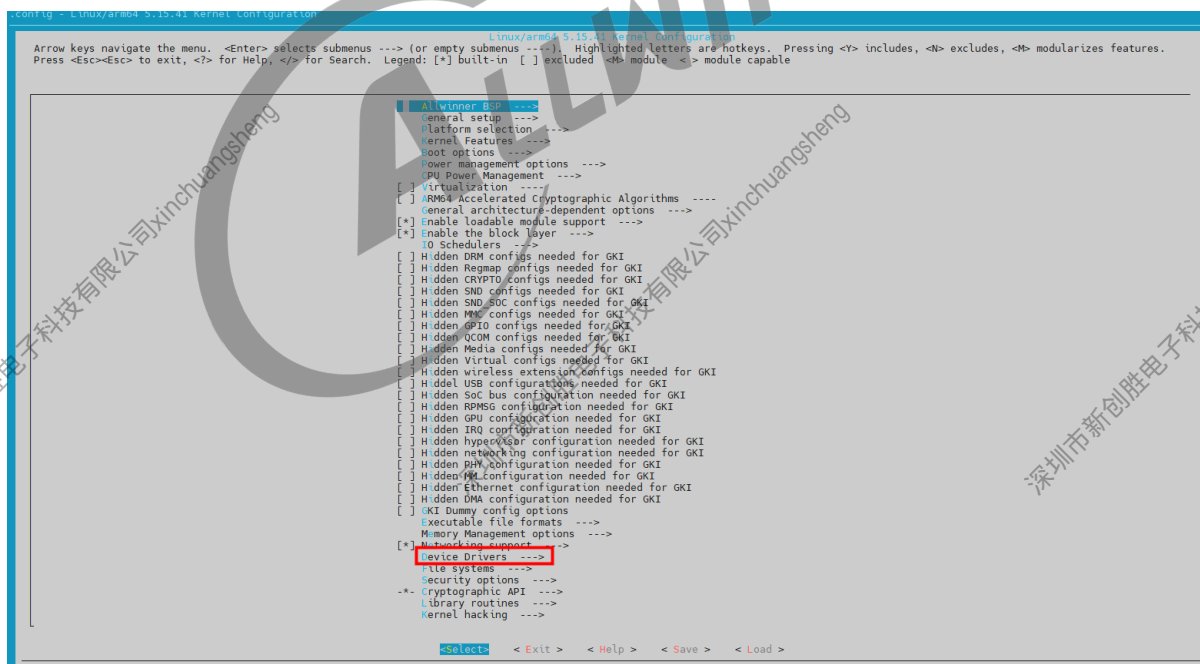


图 3-4: menuconfig 主界面

#### 2. 选择进入 Device Drivers，然后在该页面内选中 MMC/SD/SDIO card support 为 \*。

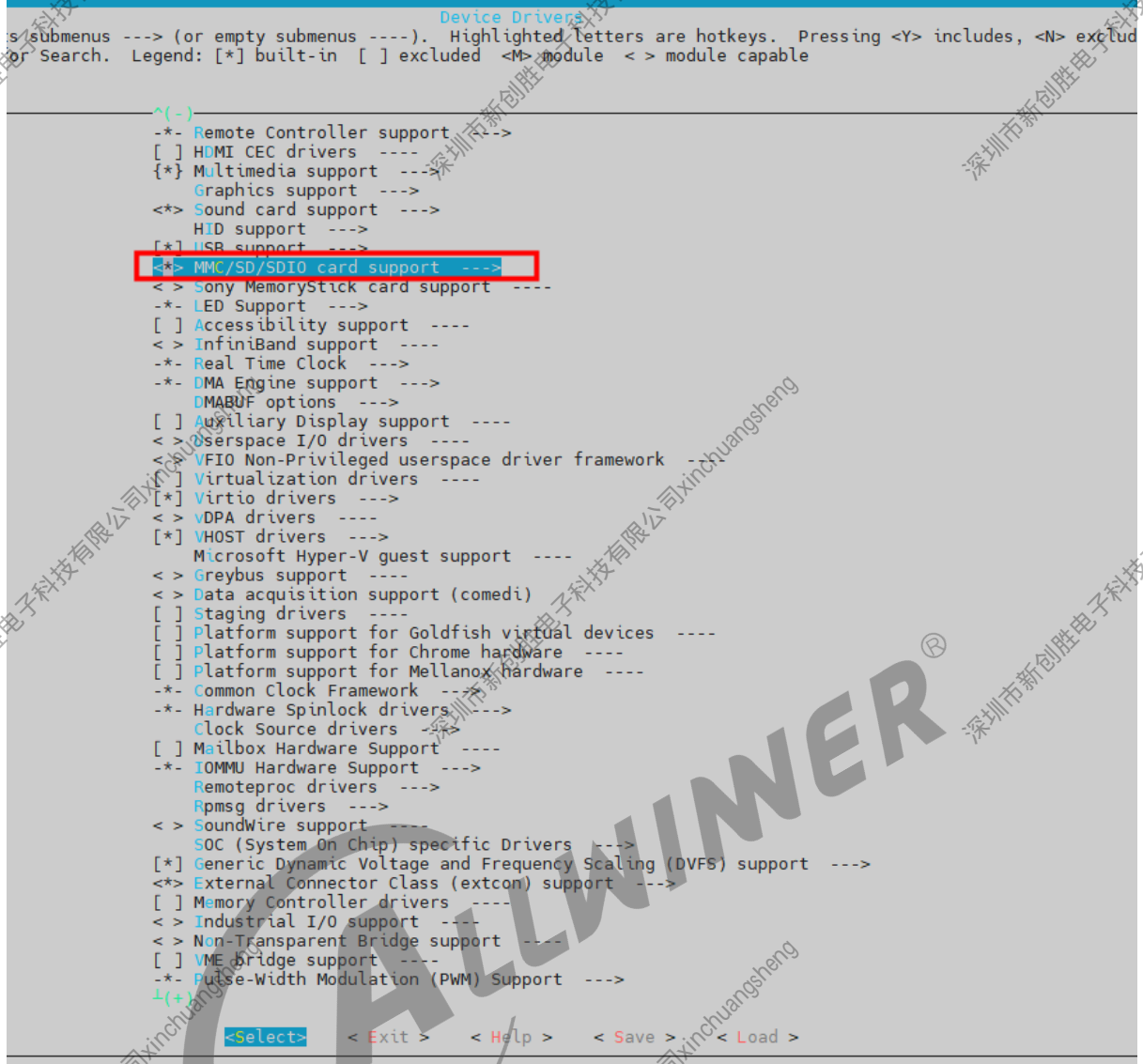


图 3-5: Device Drivers

MMC/SD/SDIO card support 中断配置如下图所示:

```
MMC/SD/SDIO card support
menus ---> (or empty submenus ----). Highlighted letters are hotkeys. Pressir
arch. Legend: [*] built-in [ ] excluded <M> module <> module capable

-- MMC/SD/SDIO card support
<*> HW reset support for eMMC
<*> Simple HW reset support for MMC
<*> MMC block device driver
(8) Number of minors per block device
< > SDIO UART/GPS class support
< > MMC host test driver
*** MMC/SD/SDIO Host Controller Drivers ***
[ ] MMC host drivers debugging
< > ARM AMBA Multimedia Card Interface support
< > Secure Digital Host Controller Interface support
< > TI Flash Media MMC/SD Interface support
< > MMC/SD/SDIO over SPI
< > ENE CB710 MMC/SD Interface support
< > VIA SD/MMC Card Reader Driver
< > Synopsys DesignWare Memory Card Interface
< > VUB300 USB to SDIO/SD/MMC Host Controller support
< > USB SD Host Controller (USHC) support
< > Renesas USDHI6R0L0 SD/SDIO Host Controller support
< > Allwinner sunxi SD/MMC Host Controller support
< > Command Queue Host Controller Interface support
< > MMC Host Software Queue support
< > Toshiba Type A SD/MMC Card Interface Driver
< > MediaTek SD/MMC Card Interface support
```

图 3-6: MMC/SD/SDIO card support

3. 回到 menuconfig 主界面，然后选择 Allwinner BSP 进入。

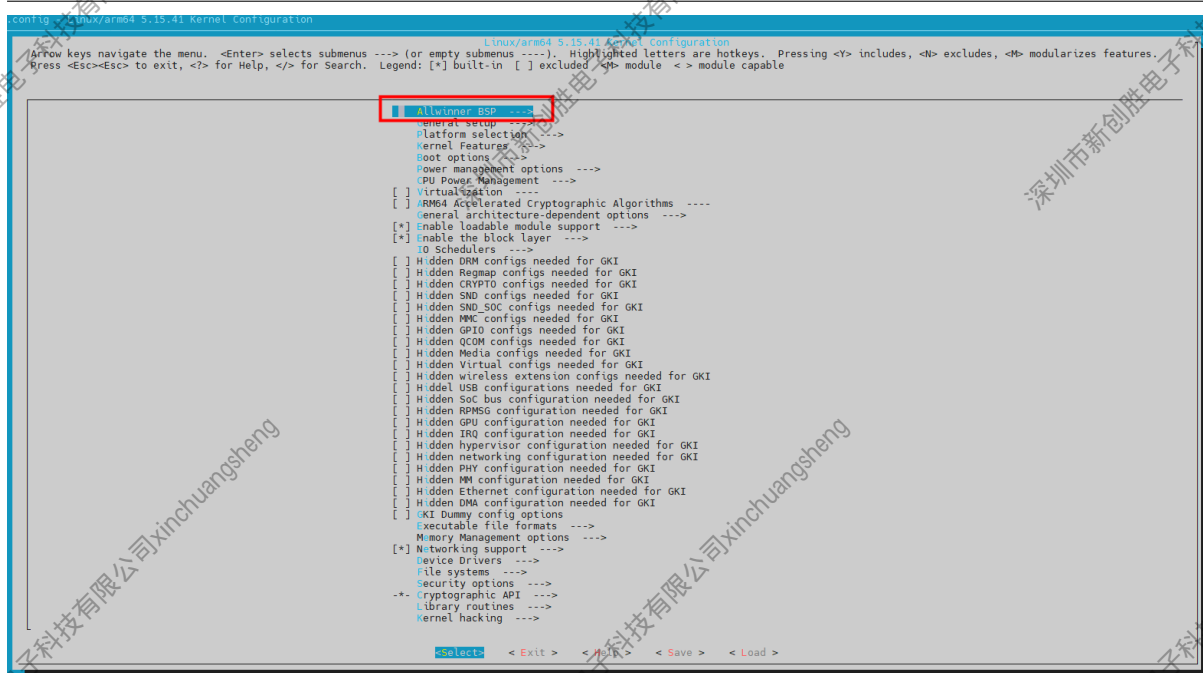


图 3-7: menuconfig 主界面

4. 进入后选择 Device Drivers 再次进入：

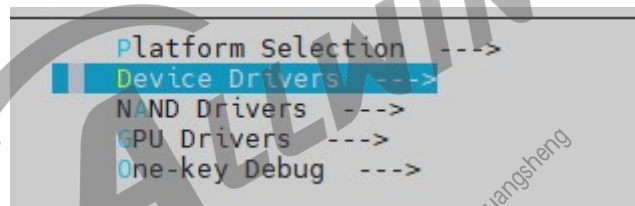


图 3-8: menuconfig Allwinner BSP

5. 进入后选择 SD/MMC Drivers 再次进入：

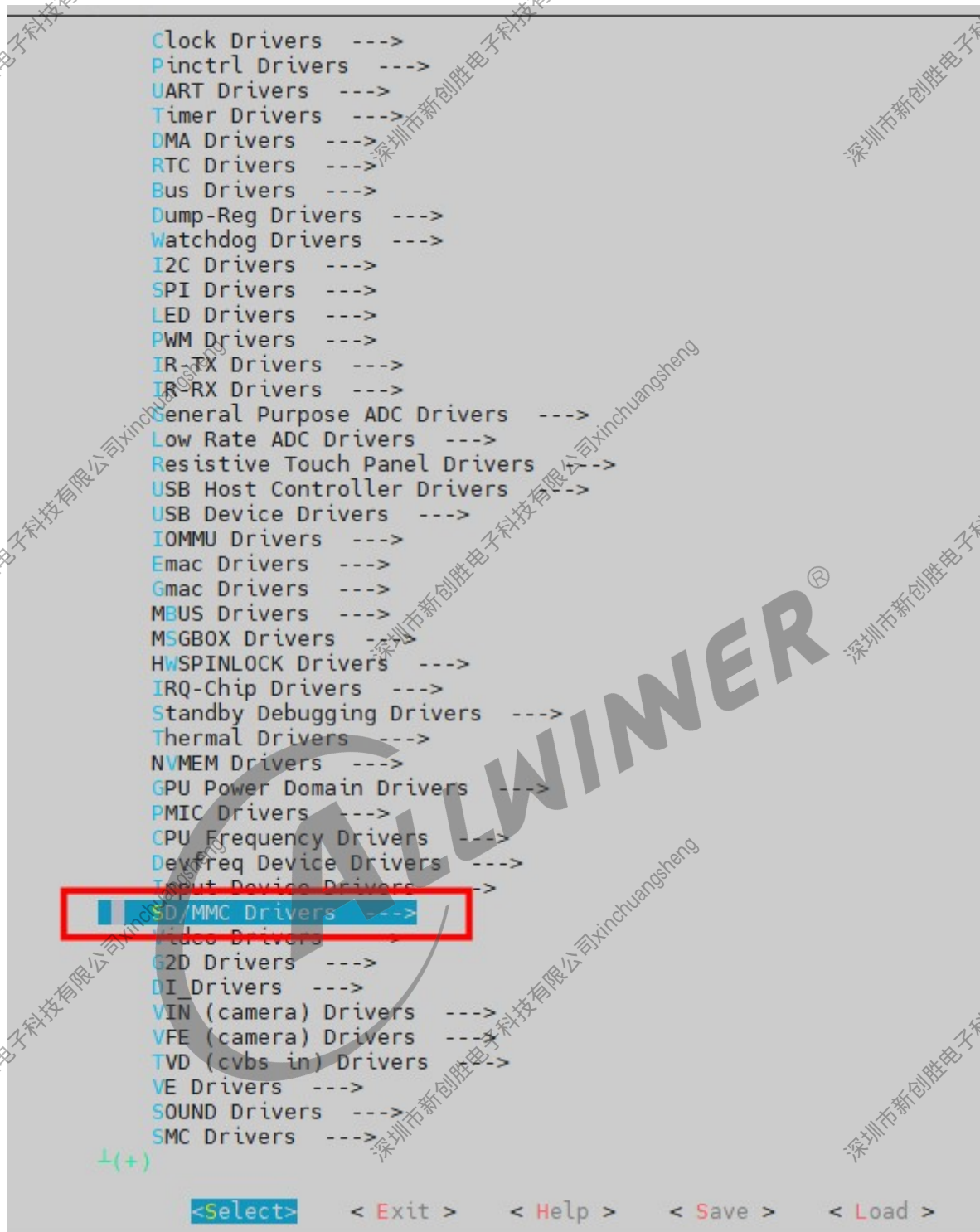


图 3-9: menuconfig Device Drivers

6. 进入后选中 Allwinner SD/MMC Host Controller Support “\*”，即可使能全志平台 MMC 驱动。



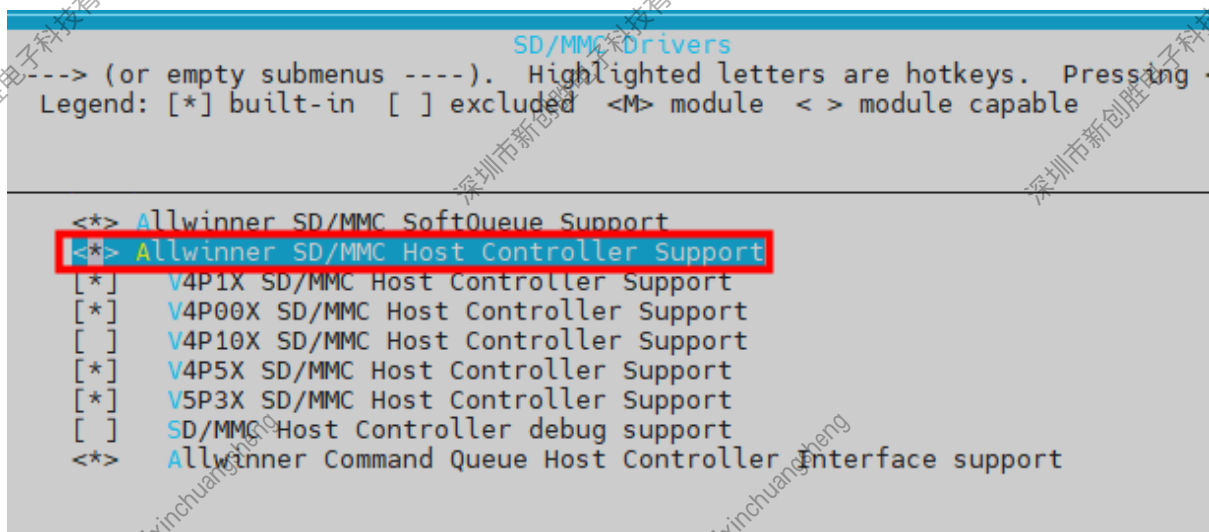


图 3-10: menuconfig SD/MMC Drivers

其中 Allwinner SD/MMC SoftQueue Support 为软队列对应 cap-hsq 的配置，Allwinner Command Queue Host Controller Interface support 为 cmdq 功能对应为 cqe-on 的配置，默认方案配置均有使能，不需要额外进行配置，可以根据具体使用情况进行选择。

### 3.4 源码结构介绍

SD/MMC 总线驱动的源代码 linux5.4 以前位于内核在 drivers/mmc/host 目录下：

SD/MMC 总线驱动的源代码 linux5.10 以后位于 bsp 仓库在 drivers/mmc/目录下：

```

|--sunxi-mmc.c // 集中了所有的控制器的公共部分以及主要的控制逻辑代码，以及大部分的资源申请和使用，包括中断，pin, ccmu等
|--sunxi-mmc.h // 为Sunxi平台的SD/MMC控制器驱动定义了一些公共的宏、数据结构
|--sunxi-mmc-v4p1x.c // 部分平台的sdc0、sdc1的控制器差异部分驱动代码
|--sunxi-mmc-v4p1x.h // 部分平台的sdc0、sdc1的控制器驱动定义了一些的宏、数据结构
|--sunxi-mmc-v4p5x.c // sdc2的控制器差异部分驱动代码
|--sunxi-mmc-v4p5x.h // sdc2的控制器驱动定义了一些的宏、数据结构
|--sunxi-mmc-v5p3x.c //部分平台的sdc0、sdc1的控制器差异部分驱动代码
|--sunxi-mmc-v5p3x.h//部分平台的sdc0、sdc1的控制器驱动定义了一些的宏、数据结构
|--sunxi-mmc-debug.c 用于debug的代码
|--sunxi-mmc-export.c 提供给其他模块的独立接口
|--cqhci*.c cmdq功能代码

```

## 4 FAQ

### 1、sd 卡和 wifi 经常出现 retry：start 等打印的原因和原理；

```
55529 20220303:10:27:16[83134.953729] sunxi-mmc sd0: smc 0 p0 err, cmd 25, WR EBE !!
55530 20220303:10:27:16[83134.964738] sunxi-mmc sd0: retry:start
55531 20220303:10:27:16[83134.969109] sunxi-mmc sd0: retry:stop
55532 20220303:10:27:16[83134.973355] sunxi-mmc sd0: retry:stop recover
55533 20220303:10:27:16[83134.982714] sunxi-mmc sd0: REG DRV_DL: 0x00030000
55534 20220303:10:27:16[83134.988084] sunxi-mmc sd0: REG SD_NTSR: 0x81710000
55535 20220303:10:27:16[83134.993551] sunxi-mmc sd0: REG NTDL_HS400: 0x20000010
55536 20220303:10:27:16[83134.999559] sunxi-mmc sd0: *****retry:re-send cmd*****
```

图 4-1: retry

答：出现这种情况是因为 sdio 通信失败了，一般是数据 crc 校验错误；对于这种错误主要怀疑，硬件信号受到干扰。而这个 retry 机制是通过重发改变相位等方法来规避单次通信失败。在通信无法回复的情况下，会出现大量的 retry log，此时请检查硬件信号，供电等关键信息。

## 4.1 调试方法

### 4.1.1 调试节点

#### 4.1.1.1 寄存器信息

linux5.4, linux5.10以及linux5.15内核

a.sd2

(1)sd2 gpio寄存器信息

cat /sys/devices/platform/soc@2900000/4022000.sdmmc/sunxi\_dump\_gpio\_register

(2)sd2 ccmu寄存器信息

cat /sys/devices/platform/soc@2900000/4022000.sdmmc/sunxi\_dump\_ccmu\_register

(3)sd2 host寄存器信息

cat /sys/devices/platform/soc@2900000/4022000.sdmmc/sunxi\_dump\_host\_register

b.sd0

(1)sd0 gpio寄存器信息

cat /sys/devices/platform/soc@2900000/4020000.sdmmc/sunxi\_dump\_gpio\_register

(2)sd0 ccmu寄存器信息

cat /sys/devices/platform/soc@2900000/4020000.sdmmc/sunxi\_dump\_ccmu\_register

(3)sd0 host寄存器信息

cat /sys/devices/platform/soc@2900000/4020000.sdmmc/sunxi\_dump\_host\_register

(4)手动扫描卡接口

echo 1 > /sys/devices/platform/soc@2900000/4020000.sdmmc/sunxi\_insert

c.sd1



```

(1)sdc1 gpio寄存器信息
cat /sys/devices/platform/soc@2900000/4021000.sdmmc/sunxi_dump_gpio_register
(2)sdc1 ccmu寄存器信息
cat /sys/devices/platform/soc@2900000/4021000.sdmmc/sunxi_dump_ccmu_register
(3)sdc1 host寄存器信息
cat /sys/devices/platform/soc@2900000/4021000.sdmmc/sunxi_dump_host_register

linux4.9内核
a.sdc2
(1)sdc2 gpio寄存器信息
cat /sys/devices/platform/soc/sdc2/sunxi_dump_gpio_register
(2)sdc2 ccmu寄存器信息
cat /sys/devices/platform/soc/sdc2/sunxi_dump_ccmu_register
(3)sdc2 host寄存器信息
cat /sys/devices/platform/soc/sdc2/sunxi_dump_host_register

b.sdc0
(1)sdc0 gpio寄存器信息
cat /sys/devices/platform/soc/sdc0/sunxi_dump_gpio_register
(2)sdc0 ccmu寄存器信息
cat /sys/devices/platform/soc/sdc0/sunxi_dump_ccmu_register
(3)sdc0 host寄存器信息
cat /sys/devices/platform/soc/sdc0/sunxi_dump_host_register
(4)手动扫描卡接口
echo 1 > /sys/devices/platform/soc/sdc0/sunxi_insert

c.sdc1
(1)sdc1 gpio寄存器信息
cat /sys/devices/platform/soc/sdc1/sunxi_dump_gpio_register
(2)sdc1 ccmu寄存器信息
cat /sys/devices/platform/soc/sdc1/sunxi_dump_ccmu_register
(3)sdc1 host寄存器信息
cat /sys/devices/platform/soc/sdc1/sunxi_dump_host_register

```

#### 4.1.1.2 emmc 信息

```

(1) 获取路径：cd /sys/block/mmcblk0/device.这里包含了大部分的emmc信息
block      ffw_capable preferred_erase_size
cid        fwrev   prv
cmdq_en     hwrev   raw_rpmb_size_mult
csd         life_time rca
date        manfid  rel_sectors
driver      mmcblk0rpmb rev
dsr         name    serial
enhanced_area_offset ocr    subsystem
enhanced_area_size  oemid  type
enhanced_rpmb_supported power  uevent
erase_size  pre_eol_info

```

```

(2) cat 需要的信息，
例如
获取寿命信息
cat life_time
cat pre_eol_info
获取emmc名字
cat namd
获取生产日期

```

```
cat data
获取唯一识别码
cat cid
获取制造商id
cat manfid
```

### 4.1.1.3 性能验证节点

该节点主要是用来记录底层存储的读写性能，该节点记录的数据不参含调度以及文件系统的影响。

为了描述方便，这里设定 base 目录这一概念，其中 X 代表控制器号；

```
内核linux4.9 base=/sys/devices/platform/soc/sdcX
内核linux5.4 base=/sys/devices/platform/soc@2900000/402X000.sdmmc
内核linux5.10 base=/proc/device-tree/soc@XX/402X000.sdmmc
内核linux5.15 base=/proc/device-tree/soc@XX/402X000.sdmmc
(1)开始测量：echo 1 > /$base/sunxi_host_perf
(2)进行读写操作
(3)获取测试结果：cat /$base/sunxi_host_perf
```

Write performance at host driver Level:1073741824 bytes in 105978400 microseconds

Read performance at host driver Level:0 bytes in 0 microseconds

图 4-2: sunxi\_host\_perf

```
(4) 速度计算：1073741824byte/105978400us = 9.66MB/s
(5)清除测量数据：echo 0 > /$base/sunxi_host_perf
```

### 动态设置

以下动态设置的节点均于 base 目录下：

```
sunxi_host_perf 总开关，打开后下面设置才有效。
sunxi_host_filter_w_sector：单笔数据传输的数据大于等于这个数据量，sunxi_host_filter_w_speed 才生效，单位是扇区。
sunxi_host_filter_w_speed：速度低于这个值就打印出来，单位是 B/S。
```

### 参考

```
echo 20971520 > /$base/sunxi_host_filter_w_speed
echo 8 > /$base/sunxi_host_filter_w_sector
echo 1 > /$base/sunxi_host_perf
```

### 效果

```
20190322_17:24:37.207 [ 64.922940] c=25,a=0x 3fc00,bs= 2560,t= 105463us,sp= 12136KB/s
20190322_17:24:37.586 [ 65.301113] c=25,a=0x 43800,bs= 2560,t= 92740us,sp= 13802KB/s
20190322_17:24:37.829 [ 65.544155] c=25,a=0x 46000,bs= 2560,t= 94162us,sp= 13593KB/s
20190322_17:24:37.967 [ 65.682744] c=25,a=0x 47400,bs= 2560,t= 77371us,sp= 16543KB/s
20190322_17:24:38.041 [ 65.755126] c=25,a=0x 47e00,bs= 2560,t= 64860us,sp= 19734KB/s
```

### 开机默认启动

在 dts 或者 sysconfig.fex 里面加入下面配置：

per\_enable：总开关，打开后下面设置才有效。  
fiter\_sector：传输的数据大于等于这个数据量，fiter\_speed 才生效，单位是扇区。  
fiter\_speed：速度低于这个值就打印出来，单位是 B/S。

参考

```
[card0_boot_para]
card_ctrl    = 0
card_high_speed = 1
card_line    = 4
sd_c_d1      = port:PF0<2><1><2><default>
sd_c_d0      = port:PF1<2><1><2><default>
sd_c_clk     = port:PF2<2><1><2><default>
sd_c_cmd     = port:PF3<2><1><2><default>
sd_c_d3      = port:PF4<2><1><2><default>
sd_c_d2      = port:PF5<2><1><2><default>
...
fiter_speed  = 20971520
fiter_speed  = 8
per_enable   = 1
```

效果

```
20190322_17:24:37.207 [ 64.922940] c=25,a=0x 3fc00,bs= 2560,t= 105463us,sp= 12136KB/s
20190322_17:24:37.586 [ 65.301113] c=25,a=0x 43800,bs= 2560,t= 92740us,sp= 13802KB/s
20190322_17:24:37.829 [ 65.544155] c=25,a=0x 46000,bs= 2560,t= 94162us,sp= 13593KB/s
20190322_17:24:37.967 [ 65.682744] c=25,a=0x 47400,bs= 2560,t= 77371us,sp= 16543KB/s
20190322_17:24:38.041 [ 65.755126] c=25,a=0x 47e00,bs= 2560,t= 64860us,sp= 19734KB/s
```

## 4.2 常见问题

参考《MMC 量产问题快速排查指南》、《eMMC 硬件排查指南》。




## 著作权声明

版权所有 © 2023 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。