



Linux NOR 开发指南

版本号: 1.1
发布日期: 2022.02.22

版本历史

版本号	日期	制/修订人	内容描述
1.0	2021.12.21	AWA1669	建立初始版本
1.1	2022.02.22	AWA1669	增加 uboot shell 使用



目 录

1	引言	1
1.1	编写目的	1
1.2	适用范围	1
1.3	相关人员	1
2	模块介绍	2
2.1	模块功能介绍	2
2.2	相关术语介绍	3
2.3	模块配置介绍	3
2.3.1	longan 的配置和打包	3
2.3.2	sys_config 配置	4
2.3.3	UBOOT 配置	5
2.3.3.1	编译和配置	5
2.3.3.2	Menuconfig 配置	5
2.3.4	KERNEL 配置	8
2.3.4.1	SPINOR-驱动配置	8
2.3.4.2	cmdline 方式选择	11
2.3.4.3	文件系统配置	12
2.4	源码目录介绍	14
2.4.1	UBOOT 源码目录	14
2.4.2	KERNEL 源码目录	14
3	接口描述	15
3.1	驱动物理层接口	15
3.1.1	spi_nor_erase	15
3.1.2	spi_nor_read	15
3.1.3	spi_nor_write	16
3.1.4	spi_nor_lock	16
3.1.5	spi_nor_unlock	16
3.1.6	spi_nor_is_locked	17
3.1.7	spi_nor_has_lock_erase	17
3.1.8	spi_nor_has_lock_write	17
3.2	Uboot 应用接口	18
3.2.1	sunxi_flash_spinor_probe	18
3.2.2	sunxi_flash_spinor_init	18
3.2.3	sunxi_flash_spinor_exit	18
3.2.4	sunxi_flash_spinor_write	18
3.2.5	sunxi_flash_spinor_write	19
3.2.6	sunxi_flash_spinor_erase	19
3.2.7	sunxi_flash_spinor_force_erase	19
3.2.8	sunxi_flash_spinor_flush	19
3.2.9	sunxi_flash_spinor_download_spl	20

3.2.10 sunxi_flash_spinor_download_toc	20
4 使用例子	21
4.1 uboot shell 使用	21
4.1.1 sunxi_flash	21



插 图

2-1 SPINOR 软件框架	2
2-2 uboot_menuconfig1	6
2-3 uboot_menuconfig2	7
2-4 uboot_menuconfig3	8
2-5 kernel_menuconfig1	9
2-6 kernel_menuconfig2	10
2-7 kernel_menuconfig3	11
2-8 kernel_menuconfig5	11
2-9 kernel_menuconfig6	12
2-10 kernel_menuconfig7	12
2-11 kernel_menuconfig8	13
2-12 kernel_menuconfig9	13
4-1 sunxi flash read	21
4-2 hexdump	22
4-3 mm - md	22
4-4 sunxi flash write	22
4-5 sunxi flash read2	22

1 引言

1.1 编写目的

此文档描述 Sunxi NOR 模块的使用方法，为相关人员调试提供指导

1.2 适用范围

boot0: 适用于 brandy-2.0

u-boot: 适用于 u-boot-2018

kernel: 适用于 linux-4.9/linux-5.4 内核

1.3 相关人员

BSP 的开发人员、测试人员

2 模块介绍

2.1 模块功能介绍

Linux 中 SPINOR 体系结构如下图所示：

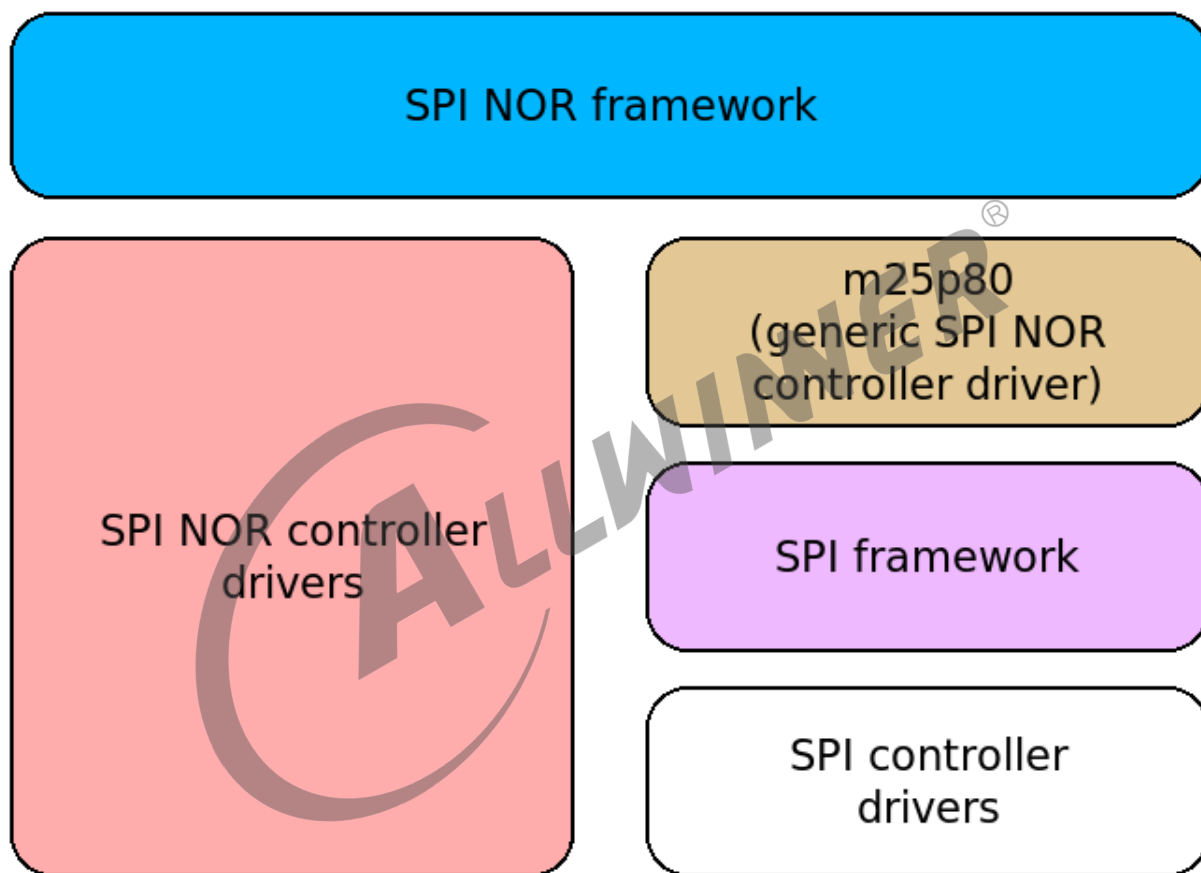


图 2-1: SPINOR 软件框架

SPI NOR Framework: 这层主要是处理不同厂家的 NOR 物理特色差异，初始化 SPINOR 的工作状态，如工作线宽（1 线、2 线、4 线、8 线）、有效地址位（16M 以上的 NOR 需要使用 4 地址模式），为上层 MTD 提供读写擦接口。

对应代码目录：drivers/mtd/spi-nor/spi-nor.c

M25P80 (generic SPI NOR controller driver) : 这层主要对 SPI NOR Framework 层传下来的数据封装成 msg，传递给 SPI framework 层。

对应代码目录：drivers/mtd/devices/m25p80.c

SPI Framework: 这层主要是将 msg 加入 ctl 的工作队列中，启动内核线程队列，处理队列中的 msg。

对应代码目录：drivers/spi/spi.c

SPI controller driver: 这层初始化 SPI 控制器频率、时钟模式、cs 有效电平、大小端等配置，同时处理上层传下来的 msg，通过 CPU/DMA 方式传输数据到 FIFO，再传输给外设 SPINOR。

对应代码目录：drivers/spi/spi-sunxi.c

2.2 相关术语介绍

术语	解释说明
Sunxi	指 Allwinner 的一系列 SOC 硬件平台
SPI	Serial Peripheral Interface，同步串行外设接口
NOR Flash	NOR Flash 是一种非易失闪存技术，是 Intel 在 1988 年创建
MTD	MTD(memory technology device 内存技术设备) 是用于访问 memory 设备 (ROM、flash) 的 Linux 的子系统

2.3 模块配置介绍

2.3.1 longan 的配置和打包

```
./build.sh config
All available platform:
  0. android
  1. linux
Choice [linux]: 1
...
All available flash: //配置根据需求选择
  0. default //flash类型，只区分nor和非nor方案，Android方案无此选项，默认非nor
  1. nor
Choice [default]: 1
```

1. 打包普通固件

```
#!/build.sh clean
#!/build.sh
#!/build.sh pack
```

2. 打包卡打印固件


```
#!/build.sh clean
#!/build.sh
#!/build.sh pack_debug
```

在配置的过程中会把平台目录下的 BoardConfig.mk 的信息拷贝到 buildconfig 中。

2.3.2 sys_config 配置

SPINOR 的 boot0 启动阶段，部分参数是从 boot0 头部获取的，而这些参数是我们在打包固件时，通过工具 update_boot0 将 sys_config.fex 中 [spinor_para]，更新到 boot0 头部的，sys_config.fex 的 [spinor_para] 配置参数如下：

```
[spinor_para]
;readcmd          =0x6b
;read_mode        =4
;write_mode        =4
;flash_size        =16
;delay_cycle       =1
;frequency         =100000000
;erase_size        =64
;lock_flag         =0
;sample_delay      =0
;sample_mode       =2

spi_sclk           = port:PC00<4><0><2><default>
spi_cs             = port:PC01<4><1><2><default>
spi0_mosi          = port:PC02<4><0><2><default>
spi0_miso          = port:PC03<4><0><2><default>
spi0_wp            = port:PC04<4><0><2><default>
spi0_hold          = port:PC05<4><0><2><default>
```

其中：

readcmd: boot0 用于读取数据的命令，不填默认用 uboot 传递过来的 readcmd

read_mode、**write_mode**: boot0 的工作线宽（1、2、4），不填默认更加 readcmd 决定线宽

flash_size: flash 的大小

delay_cycle: boot0 的采样延时配置，大于 60MHZ 配置为 1，小于 24MHZ 配置为 2，大于 24MHZ 小于 60HZ 配置为 3

frequency: boot0 的 SPI 工作频率，不填使用默认值 50M

erase_size: boot0 的擦除单位

lock_flag: 锁功能是否打开

sample_delay: boot0 的细调采样的采样延时，uboot、kernel 也会用到，默认不填等于 0xaaaaffff

sample_mode: boot0 的细调采样的采样模式，uboot、kernel 也会用到，默认不填等于 0xaaaaffff

spi_sclk、**spi_cs**、**spi0_mosi**、**spi0_miso**、**spi0_wp** 和 **spi0_hold** 用于配置相应的

GPIO。

2.3.3 UBOOT 配置

2.3.3.1 编译和配置

```
#make clean
#make sun8iw19p1_nor_config ----启动的uboot  (#make sun8iw19p1_config----烧写uboot)
#make -j32
```

2.3.3.2 Menuconfig 配置

```
#cd brandy/brandy-2.0/u-boot-2018
#make menuconfig
```

- 进入 Device Drivers

```
Device Drivers ---->
[*]SPI Support ---->
[*]Sunxi flash support ---->
```

```

Device Drivers
selects submenus ---> (or empty submenus ----). Highlighted letters are hotkeys. Press
* for Help, </> for Search. Legend: [*] built-in [ ] excluded <M> module < > module

^(-)
[ ] Bit-banged ethernet MII management channel support
[ ] Marvell 88E6352 switch support
[ ] Ethernet PHY (physical media interface) support ----
[ ] NXP PFE Ethernet driver ----
[ ] TI Common Platform Ethernet Switch
[ ] Network device support ----
[ ] PCI support ----
    PHY Subsystem ----
[ ] ComPhy SerDes driver
    Pin controllers ----
    Power --->
[ ] Enable support for the sandbox PWM
    PWM_SUNXI --->
    Remote Processor drivers ----
    Reset Controller Support ----
    Real Time Clock --->
[ ] Support SCSI controllers
    Serial drivers --->
    Sound support --->
    [*] SPI Support --->
    SPMI support
[ ] Sunxi power device support ----
    System reset device drivers --->
[ ] Driver support for thermal devices
    Timer Support ----
    TPM support ----
[ ] USB support ----
    Graphics support --->
    Watchdog Timer Support --->
    -* Sunxi flash support --->
    [*] Sunxi usb device support --->

```

图 2-2: uboot_menuconfig1

- 进入 SPI Support

```

Device Drivers ---->
  [*]SPI Support ---->
    [*]Sunxi SPI driver

```

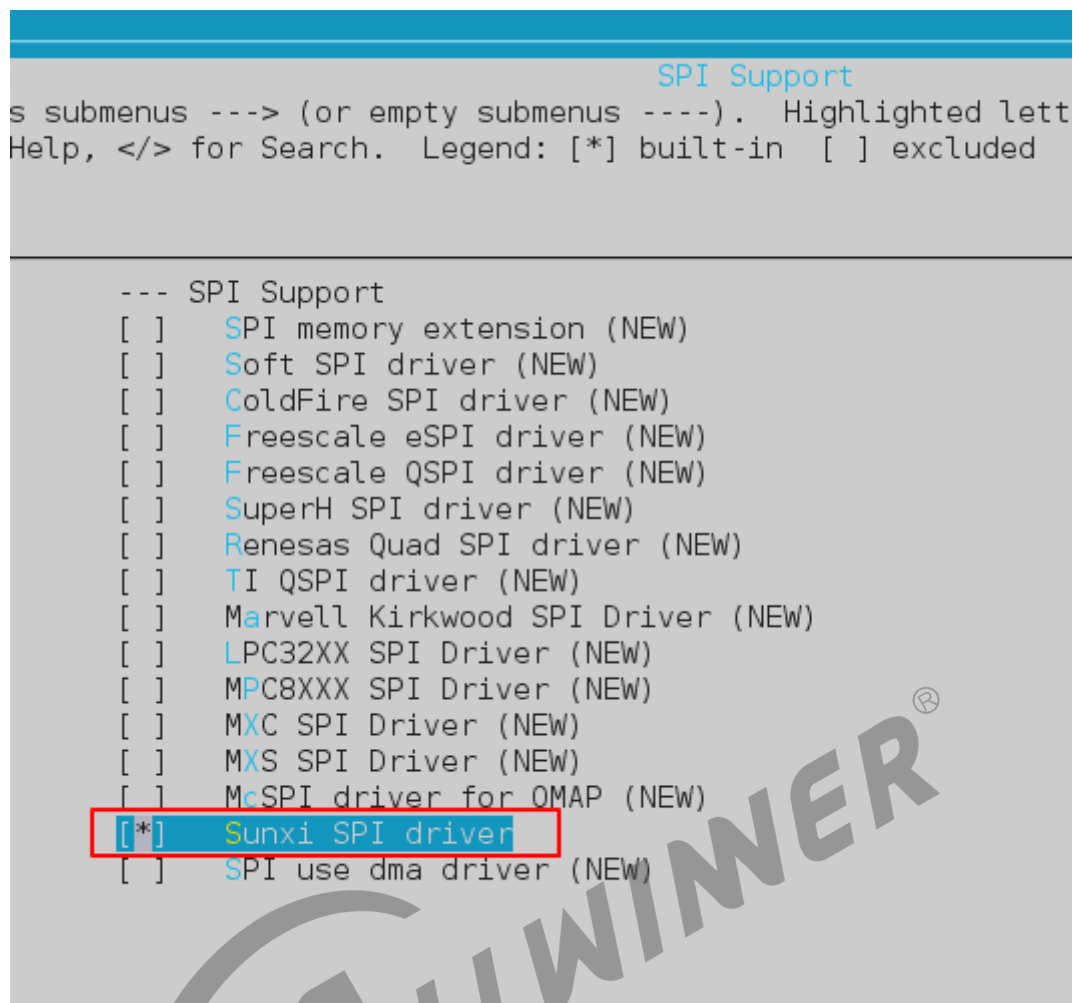


图 2-3: uboot_menuconfig2

- 进入 sunxi_flash_support

```
Device Drivers --->
[*]Sunxi flash support ---->
    [*]Support sunxi spinor devices
```

```

Sunxi flash support
submenus ---> (or empty submenus ----). Highlighted letters are hotkeys
lp, </> for Search. Legend: [*] built-in [ ] excluded <M> module < >

--- Sunxi flash support
[ ] Support sunxi nand devices
[ ] Support sunxi nand ubifs devices
[*] Support sunxi spinor devices
(2016) logic address for read/write (NEW)
(128) uboot offset for boot from spinor (NEW)
[*] support sunxi sdmmc devices
(40960) logic address for read/write
```

图 2-4: uboot_menuconfig3

2.3.4 KERNEL 配置

2.3.4.1 SPINOR-驱动配置

```
#cd kernel/linux-4.9
#make ARCH=arm menuconfig
```

- 进入 Device Drivers

```
Device Drivers ---->
<*>Memory Technology Device (MTD) support ---->
[*]SPI support ---->
```

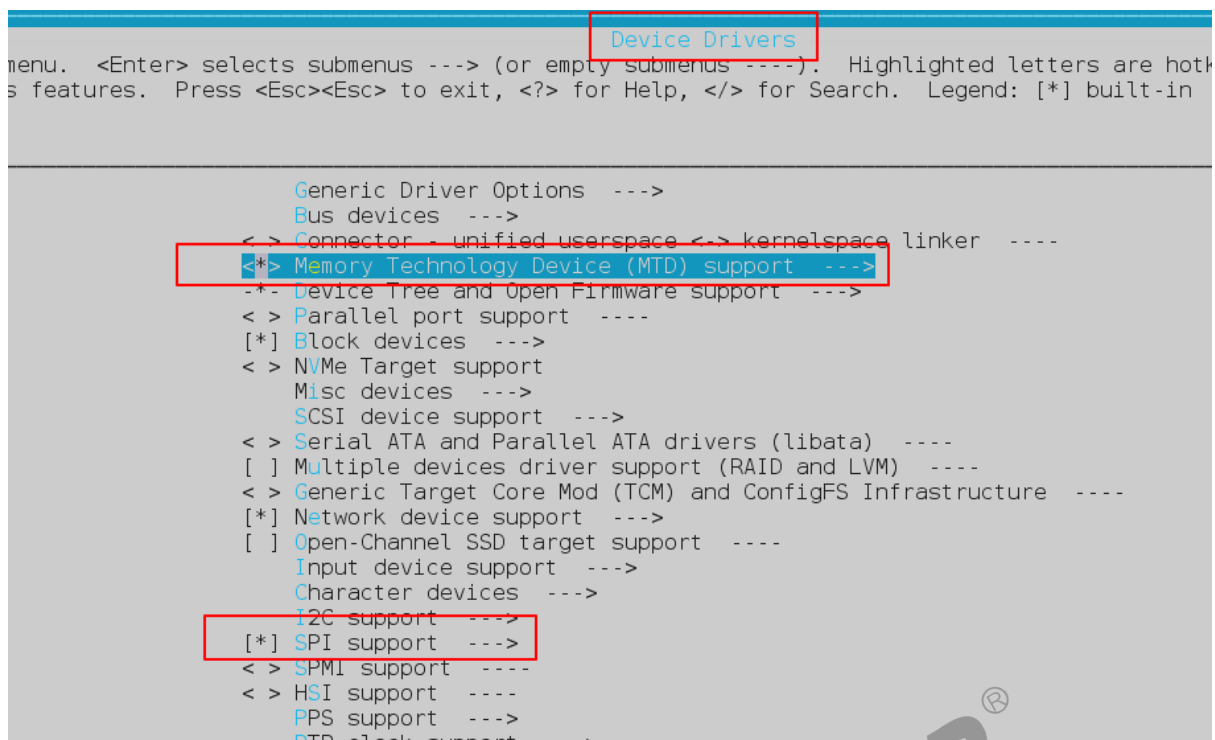


图 2-5: kernel_menuconfig1

- 进入 Memory Technology Device(MTD) support

```

Device Drivers ----->
<*>Memory Technology Device (MTD) support ----->
  <*>SUNXI partitioning support
  <*>Direct char device access to MTD devices
  <*>Caching block device access to MTD devices
  Self-contained MTD device drivers ----->
  SPI-NOR device support ----->

```

```

Memory Technology Device (MTD) support
menus ---> (or empty submenus ----). Highlighted letters are hotkeys. Pressing
, </> for Search. Legend: [*] built-in [ ] excluded <M> module <> module capa

-- Memory Technology Device (MTD) support
<> MTD tests support (DANGEROUS)
<> RedBoot partition table parsing
<> Command line partition table parsing
<> ARM Firmware Suite partition parsing
<*> OpenFirmware partitioning information support
<> TI AR7 partitioning support
<*> SUNXI partitioning support
[ ] SUNXI Uboot Disp Enable
Partition parsers --->
*** User Modules And Translation Layers ***
<*> Direct char device access to MTD devices
<*> Caching block device access to MTD devices
<> FTL (Flash Translation Layer) support
<> NFTL (NAND Flash Translation Layer) support
<> INFTL (Inverse NAND Flash Translation Layer) support
<> Resident Flash Disk (Flash Translation Layer) support
<> NAND SSFDC (SmartMedia) read only translation layer
<> SmartMedia/xD new translation layer
<> Log panic/oops to an MTD buffer
<> Swap on MTD device support
[ ] Retain master device when partitioned
RAM/ROM/Flash chip drivers --->
Mapping drivers for chip access --->
Self-contained MTD device drivers --->
<> OneNAND Device Support ----
<> Raw/Parallel NAND Device Support ----
<> SPI NAND device Support ----
sunxi-nand --->
LPDDR & LPDDR2 PCM memory drivers --->
<*> SPI-NOR device support --->
<> Enable UBI - Unsorted block images ----
<> HyperBus support ----

```

5.4内核不需要选择此项

图 2-6: kernel_menuconfig2

- 进入 Self-contained MTD device drivers (5.4 内核不需要选择此项)

```

Device Drivers ---->
  <*>Memory Technology Device (MTD) support ---->
    Self-contained MTD device drivers ---->
      <*>Support most SPI Flash chips (AT16DF, M25P.....)

```

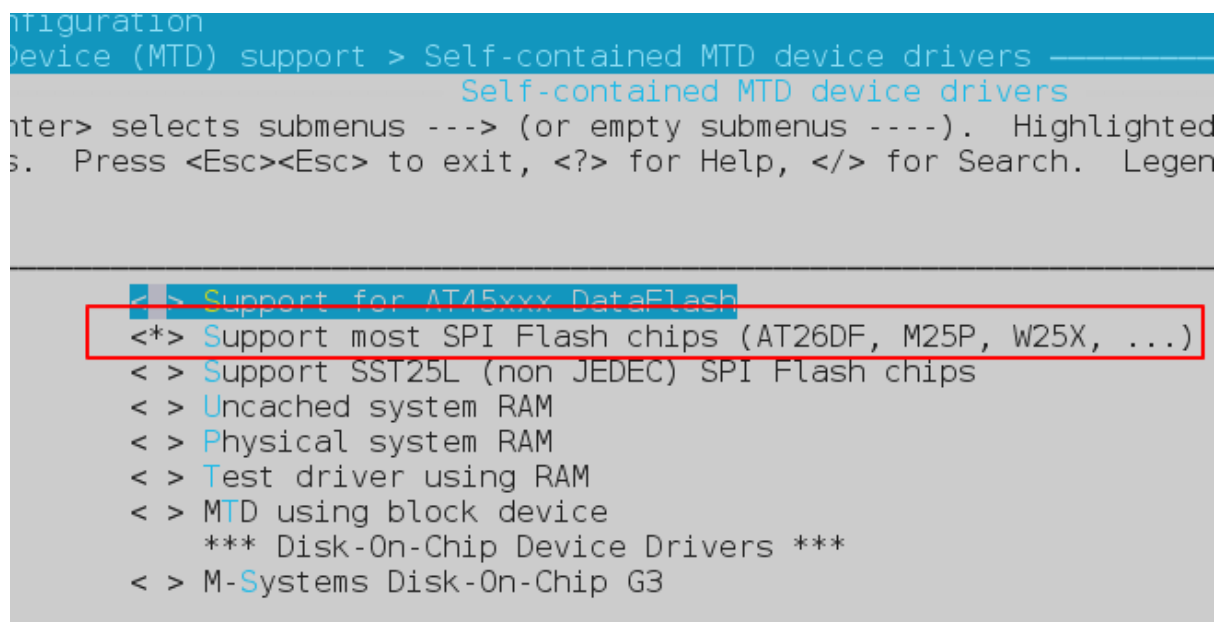


图 2-7: kernel_menuconfig3

2.3.4.2 cmdline 方式选择

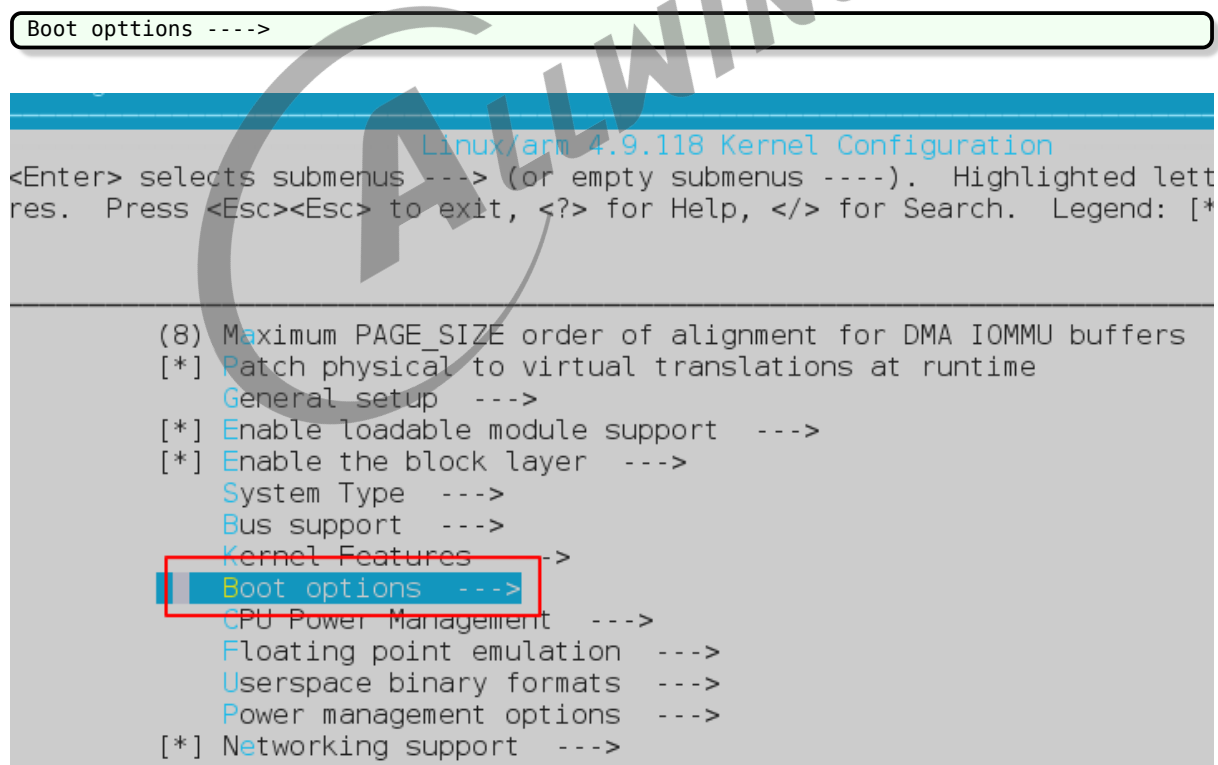


图 2-8: kernel_menuconfig5

- 进入 Boot options


```
Boot options ---->
  Kernel command line type ---->
```

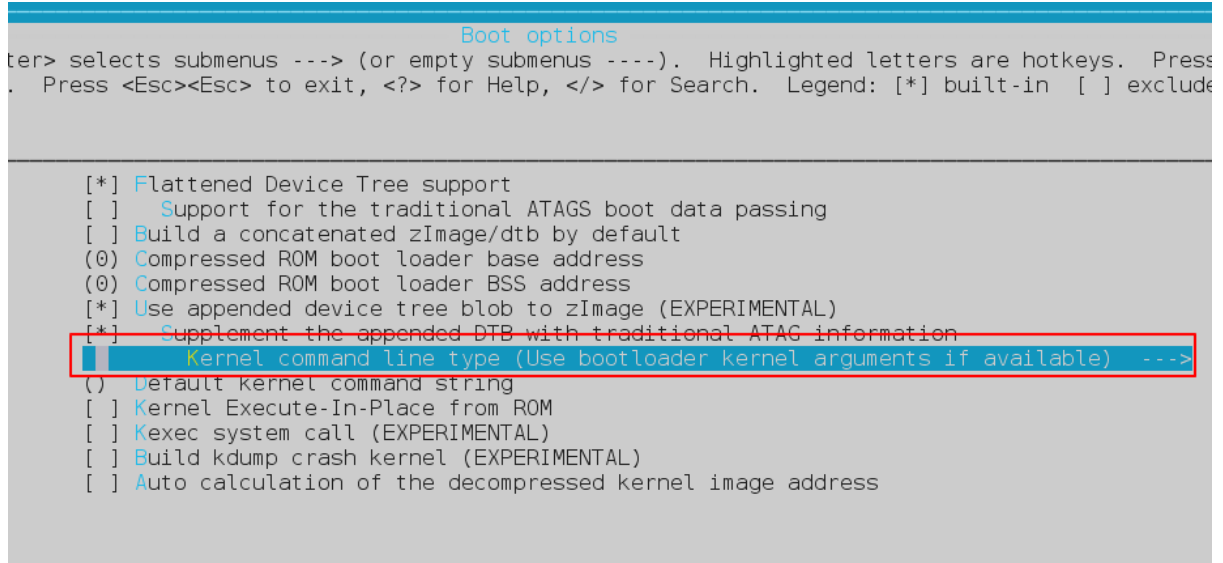


图 2-9: kernel_menuconfig6

- 进入 kernel command line type

```
Boot options ---->
  Kernel command line type ---->
    (X)Use bootloade kernel arguments if available
```

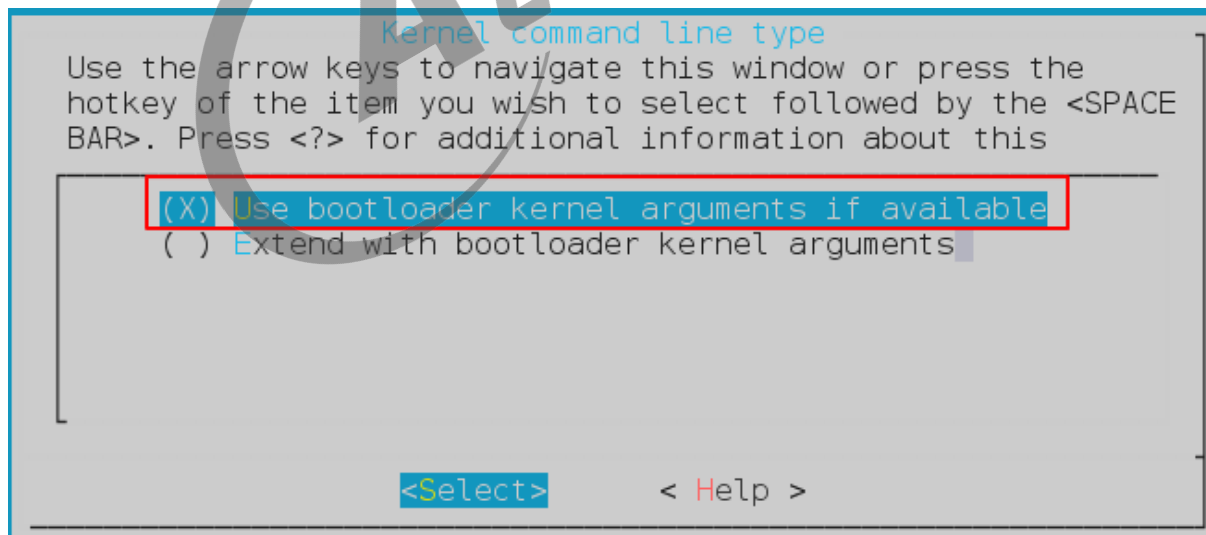


图 2-10: kernel_menuconfig7

2.3.4.3 文件系统配置

- 进入 File systems

```
File system ---->
[*]Miscellaneous filesystems ---->
```

```
[ ] Filesystem wide access notification
[ ] Quota support
< > Kernel automounter version 4 support (also supports v3)
< > FUSE (Filesystem in Userspace) support
<*> Overlay filesystem support
Caches --->
CD-ROM/DVD Filesystems --->
DOS/FAT/NT Filesystems --->
Pseudo filesystems --->
[*] Miscellaneous filesystems --->
[*] Network File Systems --->
-*- Native language support --->
< > Distributed Lock Manager (DLM) ----
```

图 2-11: kernel_menuconfig8

- 进入 Miscellaneous filesystems
- Include support for ZLIB compressed file systems (NEW)
- Include support for LZ4 compressed file systems (NEW)
- Include support for LZO compressed file systems (NEW)
- Include support for XZ compressed file systems (NEW)

```
File system ---->
[*]Miscellaneous filesystems ---->
[*]Include support for XZ compressed file systems (NEW) (压缩方式选择如下)
```

```
JFFS2 default compression mode (priority) --->
< > LogFS file system
< > Compressed ROM file system support (cramfs) (OBSOLETE)
<*> SquashFS 4.0 - Squashed file system support
File decompression options (Decompress file data into an intermediate buffer) --->
Decompressor parallelisation options (Single threaded compression) --->
[ ] Squashfs XATTR support
[ ] Include support for ZLIB compressed file systems
[ ] Include support for LZ4 compressed file systems
[ ] Include support for LZO compressed file systems
[*] Include support for XZ compressed file systems
[ ] Use 4K device block size?
[ ] Additional option for memory-constrained systems
< > FreeVxFS file system support (VERITAS VxFS(TM) compatible)
```

图 2-12: kernel_menuconfig9

以上的压缩方式（ZLIB/LZ4/LZO/XZ）具体选择哪一种需要根据 longan/build/mkcmd.sh 中如下代码使用的压缩方式而定，如下代码使用的是 gzip 压缩方式，则内核 File systems 中配置需选择 LZO 压缩方式，若使用的是 xz，则需选择 XZ 压缩方式。

```
${R00TFS} ${LICHEE_PLAT_OUT}/rootfs.squashfs -root-owned -no-progress -comp gzip -noappend
```

2.4 源码目录介绍

2.4.1 UBOOT 源码目录

```
\u-boot-2018\drivers
├─sunxi_flash      ---sunxi_flash的初始化/退出/读/写/擦除等flash接口
├─mmc              ---mmc接口代码
├─nand             ---nand接口代码
├─spinor          ---spi nor接口代码
├─sunxi_flash.c    ---sunxi_flash操作接口
├─其他
├─ spi            --sunxi_spi的接口代码
├─sunxi_spi.c      ---具体代码的实现
├─mtd
├─spi
├─sf_probe.c       ---nand接口代码
├─spinor          ---spi nor接口代码
├─sunxi_flash.c    ---sunxi_flash操作接口
├─ makefile        ---编译文件
```

2.4.2 KERNEL 源码目录

```
\longan\kernel\linux-4.9\drivers\
├─ mtd
├─spi-nor
├─spi-nor.c        ---spi nor驱动代码
├─其他
├─ spi            --spi的接口代码
├─ makefile        ---编译文件
```

3 接口描述

3.1 驱动物理层接口

3.1.1 spi_nor_erase

```
static int spi_nor_erase(struct mtd_info *mtd, struct erase_info *instr)
```

description: mtd erase interface

@mtd: MTD device structure

@instr: erase operation description structure

return: success return 0, fail return fail code

3.1.2 spi_nor_read

```
static int spi_nor_read(struct mtd_info *mtd, loff_t from, size_t len,  
                        size_t *retlen, u_char *buf)
```

description: mtd read interface

@mtd: MTD device structure

@from: offset to read from MTD device

@len: data len

@retlen: had read data len

@buf: data buffer

return: success return max_bitflips, fail return fail code

3.1.3 spi_nor_write

```
static int spi_nor_write(struct mtd_info *mtd, loff_t to, size_t len,  
                        size_t *retlen, const u_char *buf)
```

description: mtd write data interface

@to: offset to MTD device

@len: want write data len

@retlen: return the writen len

@buf: data buffer

return: success return 0, fail return code fail

3.1.4 spi_nor_lock

```
static int spi_nor_lock(struct mtd_info *mtd, loff_t ofs, uint64_t len)
```

description: check block is badblock or not

@mtd: MTD device structure

@ofs: offset the mtd device start (align to simu block size)

@len: The length of the operating

return: success return 0, fail return code fail

3.1.5 spi_nor_unlock

```
static int spi_nor_unlock(struct mtd_info *mtd, loff_t ofs, uint64_t len)
```

description: check block is badblock or not

@mtd: MTD device structure

@ofs: offset the mtd device start (align to simu block size)

@len: The length of the operating

return: success return 0, fail return code fail

3.1.6 spi_nor_is_locked

```
static int spi_nor_is_locked(struct mtd_info *mtd, loff_t ofs, uint64_t len)
```

description: check block is badblock or not

@mtd: MTD device structure

@ofs: offset the mtd device start (align to simu block size)

@len: The length of the operating

return: Is lock return 1, else return 0

3.1.7 spi_nor_has_lock_erase

```
static int spi_nor_has_lock_erase(struct mtd_info *mtd, struct erase_info *instr)
```

description: mtd has lock erase interface, First unlock to operate space, after the completion of the flash lock up

@mtd: MTD device structure

@instr: erase operation description structure

return: success return 0, fail return fail code

3.1.8 spi_nor_has_lock_write

```
static int spi_nor_has_lock_write(struct mtd_info *mtd, loff_t to, size_t len,  
                                size_t *retlen, const u_char *buf)
```

description: mtd has lock write data interface, First unlock to operate space, after the completion of the flash lock up

@to: offset to MTD device

@len: want write data len

@retlen: return the writen len

@buf: data buffer

return: success return 0, fail return code fail

3.2 Uboot 应用接口

3.2.1 sunxi_flash_spinor_probe

```
static int sunxi_flash_spinor_probe(void)
```

description: SPINOR initialization, Set the storage type.

return: zero on success, else a negative error code.

3.2.2 sunxi_flash_spinor_init

```
static int sunxi_flash_spinor_init(int boot_mode, int res)
```

description: SPINOR initialization.

@boot_mode: Working mode

@res: The default is 0

return: zero on success, else a negative error code.

3.2.3 sunxi_flash_spinor_exit

```
int sunxi_flash_spinor_exit(void)
```

description: Release registration is a resource for applications.

return: zero on success, else a negative error code.

3.2.4 sunxi_flash_spinor_write

```
static int sunxi_flash_spinor_write(uint start_block, uint nblock, void *buffer)
```

description: mtd write data interface.

@start_block: want write start sector

@nblock: want write sectorcount

@buffer: data buffer

return: zero on success, else a negative error code.

3.2.5 sunxi_flash_spinor_write

```
static int sunxi_flash_spinor_write(uint start_block, uint nblock, void *buffer)
```

description: mtd readdata interface.

@start_block: want read start sector

@nblock: want read sector count

@buffer: data buffer

return: zero on success, else a negative error code.

3.2.6 sunxi_flash_spinor_erase

```
static int sunxi_flash_spinor_erase(int erase, void *mbr_buffer)
```

description: erase boot || partition data.

@erase: erase flag

@buffer: The default is NULL

return: zero on success, else a negative error code.

3.2.7 sunxi_flash_spinor_force_erase

```
int sunxi_flash_spinor_force_erase(void)
```

description: erase boot & partition data.

return: zero on success, else a negative error code.

3.2.8 sunxi_flash_spinor_flush

```
int sunxi_flash_spinor_flush(void)
```

description: Flush physical cache data to flash.

return: zero on success, else a negative error code.

3.2.9 sunxi_flash_spinor_download_spl

```
static int sunxi_flash_spinor_download_spl(unsigned char *buf, int len, unsigned int ext)
```

description: write boot0.

@buf: boot0 data buffer

@len: boot0 data len

@ext: storage type

return: zero on success, else a negative error code.

3.2.10 sunxi_flash_spinor_download_toc

```
static int sunxi_flash_spinor_download_toc(unsigned char *buf, int len, unsigned int ext)
```

description: write uboot.

@buf: uboot data buffer

@len: uboot data len

@ext: storage type

return: zero on success, else a negative error code.

4 使用例子

4.1 uboot shell 使用

4.1.1 sunxi_flash

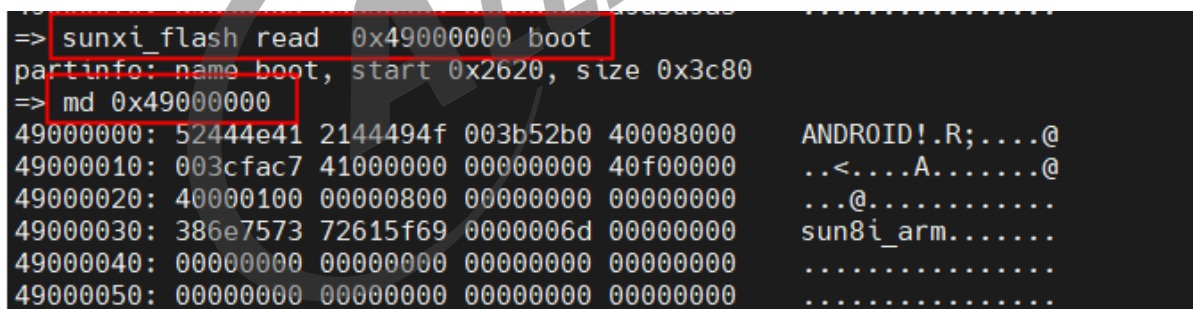
mem_addr: 内存地址, 0x40000000 之后可以随便选取如: 0x45000000, 0x46000000

part_name: 分区文件名, boot-resource、env、boot、rootfs

size: 可以省略, 默认读取整个分区文件

1. sunxi_flash read [size] 读取 flash 中的分区文件到内存中

例: 使用 sunxi_flash read 命令将 boot 分区读入到 0x49000000 中, 然后使用 md 命令读取 0x49000000 中的内容。



```
=> sunxi_flash read 0x49000000 boot
partinfo: name boot, start 0x2620, size 0x3c80
=> md 0x49000000
49000000: 52444e41 2144494f 003b52b0 40008000  ANDROID!.R;....@
49000010: 003cfac7 41000000 00000000 40f00000  ..<....A.....@
49000020: 40000100 00000800 00000000 00000000  ...@.....
49000030: 386e7573 72615f69 0000006d 00000000  sun8i_arm.....
49000040: 00000000 00000000 00000000 00000000  .....
49000050: 00000000 00000000 00000000 00000000  .....
```

图 4-1: sunxi flash read

验证方法:

1. 0x49000000 读入前与读入后数据有没有发生变化
2. 在 **out/pack_out** 目录下找到对应的分区文件, 使用 **hexdump -Cv boot.fex -n 500** 命令输出分区文件的数据, 对比一致即读入成功。

```

guanyanfei@AwExdroid100:~/workspace/longanV853/out/pack_out$ hexdump -Cv boot.tex -n 500
00000000 41 4e 44 52 4f 49 44 21 b0 52 3b 00 00 80 00 40 |ANDROID!.R;....@|
00000010 c7 fa 3c 00 00 00 00 41 00 00 00 00 00 00 f0 40 |..<...A.....@|
00000020 00 01 00 40 00 08 00 00 00 00 00 00 00 00 00 00 |...@.....|
00000030 73 75 6e 38 69 5f 61 72 6d 00 00 00 00 00 00 00 |sun8i_arm.....|
00000040 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|
00000050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|
00000060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|
00000070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|

```

图 4-2: hexdump

2. sunxi_flash write [size] 将内存中的数据，写入到分区中

例：

1) 使用 mm 命令修改内存内容

```

40000000: 00000000 00000000 00000000 00000000 .....
=> mm 0x44000000          修改内存中数据
44000000: fedcba98 ? 123
44000004: fedcba99 ? 456
44000008: fedcba9a ? 789
4400000c: fedcba9b ? ? ? 退出编辑
=> md 0x44000000          查看内存
44000000: 00000123 00000456 00000789 fedcba9b #...V.....
44000010: fedcba9c fedcba9d fedcba9e fedcba9f .....
44000020: fedcbaa0 fedcbaa1 fedcbaa2 fedcbaa3 .....
44000030: fedcbaa4 fedcbaa5 fedcbaa6 fedcbaa7 .....

```

图 4-3: mm - md

2) 使用 sunxi_flash write 0x44000000 env 将内存中的数据写入 env 分区

```

=> sunxi_flash write 0x44000000 env
guanyanfei::start: 0x2d00, len: 0x100

```

图 4-4: sunxi flash write

3) 重新将 env 分区读入内存中，对比一致表示写入成功

```

=> sunxi_flash read 0x45000000 env          读env分区
partinfo: name env, start 0x2520, size 0x100
=> md 45000000          显示内存数据
45000000: 00000123 00000456 00000789 fedcba9b #...V.....
45000010: fedcba9c fedcba9d fedcba9e fedcba9f .....
45000020: fedcbaa0 fedcbaa1 fedcbaa2 fedcbaa3 .....
45000030: fedcbaa4 fedcbaa5 fedcbaa6 fedcbaa7 .....

```

图 4-5: sunxi flash read2

著作权声明

版权所有 © 2022 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。