



Hi309a

系统编译运行指南

文档版本 02

发布日期 2025-02-04

版权所有 © 海思技术有限公司2026。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HISILICON、海思和其他海思商标均为海思技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受海思公司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的购买或使用范围之内。除非合同另有约定，海思公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

海思技术有限公司

地址：上海市青浦区虹桥港路2号101室 邮编：201721

网址：<https://www.hisilicon.com/cn/>

客户服务邮箱：support@hisilicon.com



前言

概述

本文档为使用海思Hi309a芯片进行二次开发的工程师而写，提供在开发过程中查阅Hi309a编译及升级方法。

读者对象

本文档主要适用于以下工程师：

- 技术支持工程师
- 软件开发工程师

符号约定

在本文中可能出现下列标志，它们所代表的含义如下。

符号	说明
 危险	表示如不避免则将会导致死亡或严重伤害的具有高等级风险的危害。
 警告	表示如不避免则可能导致死亡或严重伤害的具有中等级风险的危害。
 注意	表示如不避免则可能导致轻微或中度伤害的具有低等级风险的危害。
 须知	用于传递设备或环境安全警示信息。如不避免则可能会导致设备损坏、数据丢失、设备性能降低或其它不可预知的结果。 “须知”不涉及人身伤害。
 说明	对正文中重点信息的补充说明。 “说明”不是安全警示信息，不涉及人身、设备及环境伤害信息。



修订记录

修订记录累积了每次文档更新的说明。最新版本的文档包含以前所有文档版本的更新内容。

修订日期	版本	修订说明
2025-10-30	01	第1次正式版本发布。
2025-02-04	02	第2次正式版本发布。 更新编译参数选项，刷新 统一镜像构建 、 编译zImage 、 设备树制作 章节内容。



目录

前言.....	i
1 编译参考.....	1
1.1 基于 oebuild 快速构建.....	1
1.2 统一镜像构建.....	2
1.3 BIOS 模块编译.....	4
1.3.1 源码准备.....	4
1.3.2 编译环境准备.....	4
1.3.3 BIOS 编译.....	4
1.4 分模块编译.....	5
1.4.1 源码及编译环境准备.....	5
1.4.1.1 源码准备.....	5
1.4.1.2 编译环境准备.....	6
1.4.2 编译 zImage.....	6
1.4.3 编译 Hi309a SDK.....	6
1.4.4 文件系统制作.....	7
1.4.5 设备树制作.....	7
2 升级指南.....	9
2.1 wftpd32 使用指南.....	9
2.2 网络配置.....	10
2.3 Hi309a 升级.....	11
3 eMMC 启动使用指南.....	15
3.1 Hi309a 首次使用.....	15
3.2 升级方式.....	20
4 Flash 启动使用指南.....	22
5 USB 启动使用指南.....	24
6 GRUB 引导启动使用指南.....	25



插图目录

图 1-1 启动类型.....	5
图 2-1 wftpd 工具界面.....	9
图 2-2 wftpd32 创建传输目录文档路径图.....	10
图 2-3 本地文件存储路径图.....	10
图 2-4 启动 wftpd 功能图.....	10
图 2-5 网络配置图.....	11
图 2-6 BIOS 运行图.....	12
图 2-7 网络配置图.....	12
图 2-8 zImage 升级图.....	12
图 2-9 设备树升级图.....	13
图 2-10 文件系统升级图.....	13
图 2-11 openEuler 系统界面图.....	13
图 3-1 BIOS 切换串口图.....	15
图 3-2 BIOS 切换 DDR 主频图.....	16
图 3-3 DDR 频率参考图.....	17
图 3-4 zImage 升级图.....	18
图 3-5 设备树升级图.....	18
图 3-6 文件系统升级图.....	18
图 3-7 eMMC 分区图.....	18
图 3-8 新分区图.....	19
图 3-9 主分区图.....	19
图 3-10 p2 分区图.....	19
图 3-11 p2 分区起始大小图.....	19
图 3-12 p2 分区结束大小图.....	19
图 3-13 保存分区图.....	19
图 6-1 参考 grub 启动配置文件.....	25
图 6-2 grub 启动菜单.....	26



表格目录

表 1-1 统一构建编译参数说明.....	2
-----------------------	---



1 编译参考

1.1 基于 oebuild 快速构建

由于openEuler Embedded构建过程需要基于openEuler操作系统，且需要安装较多系统工具和构建工具，为方便开发人员快速搭建构建环境，将构建过程所依赖的操作系统和工具封装到一个容器中，使得开发人员可以快速搭建一个构建环境，进而投入到代码开发中去，避免在准备环境阶段消耗大量时间。

前提条件：编译环境的准备请参考文档[安装步骤 — openEuler Embedded在线文档 24.03 documentation](#)。

编译步骤如下：

步骤1 执行如下命令，初始化oebuild工作目录以及构建分支代码。

```
oebuild init sp4-buildwork -b openEuler-24.03-LTS
cd /home/user/sp4-buildwork
oebuild update
```

须知

- 假设执行路径位于“/home/user/”，执行后根据提示进入对应目录：执行路径可以根据自己需求修改执行路径。
- sp4-buildwork：存放目录。
- openEuler-24.03-LTS为当前支持本构建的源码分支。
- 执行oebuild update完成后，将在“/home/user/sp4-buildwork/src/”目录下载好主构建源码，并初始化构建虚拟环境。

步骤2 进入oebuild工作目录，执行如下命令进行创建对应的编译配置文件。

```
oebuild generate -p hi309a -d build_hi309a
```

说明

该命令默认使用sysV init的系统初始化方式，添加-f systemd选项则使用systemd的系统初始化方式。

步骤3 进入构建容器内，执行如下命令构建镜像。



```
cd /home/user/sp4-buildwork/build/build_hi309a
oebuild bitbake
bitbake openeuler-image
```

须知

oebuild bitbake执行后将进入构建虚拟环境。

步骤4 构建完成后，输出件见“/home/user/sp4-buildwork/build/build_hi309a/output/[时间戳]”，内容如下：

```
└─openeuler-xxx.rootfs.cpio.gz
```

----结束

1.2 统一镜像构建

编译环境：

- Python 3.4+、OpenSSL 1.1.1f、CMake 3.20.5+，GLIBC_2.35及以上。
- 如果出现yaml.h不存在的错误，需要依赖libyaml-dev，请安装libyaml-dev。

具体步骤如下：

步骤1 下载release代码包并解压。

步骤2 将[基于oebuild快速构建](#)中输出件openeuler-xxx.rootfs.cpio.gz拷贝到“mpu_hi309a_solution/custom/open_euler”目录下。

步骤3 执行编译脚本，具体参数说明如[表1-1](#)所示。

```
cd mpu_hi309a_solution/build
./build.sh -f tmpfs -n kernel_5.10.0 -u uart0 -i sysV -b demo
```

步骤4 编译生成的文件在“mpu_hi309a_solution/build/output”目录下。

- hi309a**.dtb：设备树。
- zImage：升级的系统镜像。
- initrd_boot.cpio.gz：文件系统。

----结束

表 1-1 统一构建编译参数说明

编译选项	取值范围	说明	备注
-f	tmpfs（内存文件系统）、ext4、mini	表示编译出的文件系统类型。	-
-u	uart0、uart1	表示编译出的文件系统和设备采用的串口。	-



编译选项	取值范围	说明	备注
-n	kernel_5.10.0、kernel_5.4.18	内核转发模式，编译名为 hi309a_xxx_xxx_kernel.dts的设备树。	-
	dpdk_21.11.0、dpdk_21.11.1、 kernel_5.10.0_with_dpdk_21.11.0、 kernel_5.10.0_with_dpdk_21.11.1、 kernel_5.4.18_with_dpdk_21.11.0、 kernel_5.4.18_with_dpdk_21.11.1	DPDK转发模式，编译名为 hi309a_xxx_xxx_dpdk.dts的设备树。	kernel_5.xx.x_with_dpdk_21.xx.x格式支持指定内核版本，仅dpdk_21.xx.x格式默认内核版本kernel_5.10.0。
	kernel_5.10.0_with_lsw, kernel_5.4.18_with_lsw	lsw二层硬转模式，编译名为 hi309a_xxx_xxx_lsw.dts的设备树。	-
	kernel_5.10.0_and_dpdk_21.11.0、 kernel_5.10.0_and_dpdk_21.11.1、 kernel_5.4.18_and_dpdk_21.11.0、 kernel_5.4.18_and_dpdk_21.11.1	内核和DPDK共存转发模式，编译名为 hi309a_xxx_xxx_kernel_dpdk.dts的设备树。	-
-b	demo、m900、m901、m902	表示单板类型。	-
-i	sysV (sysV init) 、systemd	表示系统的初始化方式。	-
-d	-	表示编译用户指定的设备树，设备树文件支持任意名称，参数不包含后缀.dts字符。例：-d user_target。	将忽略-n参数中的指定的设备树名称，-n中的其余内核版本和DPDK版本信息依然生效。
-c	linaro_5.3.1-2016.05、 openeuler_24.03	表示使用的编译工具链。	当选择linaro_5.3.1-2016.05工具链时内核版本只能选择kernel_5.4.18。



须知

- 本解决方案不包括openEuler系统，为提升用户构建效率，一键构建脚本中会通过网络下载openEuler-24.03-LTS，用户可通过build.sh和修改版本，这部分的网络安全问题由用户负责。
- 编译脚本需要添加管理员权限。
- -n参数缺省时，使用内核转发模式，内核版本kernel_5.10.0。
- 构建脚本支持多工具链编译，涉及编译工具链切换时必须先执行“./build.sh clean -c xxxx”，参数c为工具链版本（linaro_5.3.1-2016.05、openeuler_24.03）。

1.3 BIOS 模块编译

编译环境：

- Python 3.4+、OpenSSL 1.1.1f、CMake 3.20.5+、GLIBC_2.35及以上。
- 如果出现yaml.h不存在的错误，需要依赖libyaml-dev，请安装libyaml-dev。

1.3.1 源码准备

步骤1 下载release代码包并解压。

----结束

1.3.2 编译环境准备

步骤1 首先确认“/home/\${USER}”目录下是否存在“hi309A_tool_bios”文件夹，如果“hi309A_tool_bios”文件夹不存在，请在“/home/\${USER}”目录下创建“hi309A_tool_bios”文件夹。

步骤2 从以下路径下载“openEuler-22.03-LTS-SP3”交叉编译链到“~/hi309A_tool_bios”目录。

https://mirror.truenetwork.ru/openeuler/openEuler-22.03-LTS-SP3/embedded_img/aarch64/qemu-aarch64/openeuler-glibc-x86_64-openeuler-image-aarch64-qemu-aarch64-toolchain-22.03-LTS-SP3.sh

步骤3 在“~/hi309A_tool_bios”目录下，执行如下命令。

```
sh openeuler-glibc-x86_64-openeuler-image-aarch64-qemu-aarch64-toolchain-22.03-LTS-SP3.sh -d /home/${USER}/hi309A_tool_bios/toolchain/ -y
```

步骤4 配置openEuler交叉编译环境，执行如下命令。

```
export PATH=/home/${USER}/hi309A_tool_bios/toolchain/sysroots/x86_64-openeulersdk-linux/usr/bin:$PATH
```

----结束

1.3.3 BIOS 编译

目前BIOS下支持从eMMC启动和USB启动，启动方式是通过“/mpu_hi309a_solution/src/BIOS/BIOS/HwPkg/UEFI/Products/Network/script/hi309a/fw_hi309a.pcfx”来实现的。



图 1-1 启动类型

```
# 本地启动的类型: emmc=0, usb=1  
boottype=0
```

eMMC启动，默认boottype = 0。从USB启动，将boottype改为1，然后编译。

步骤1 编译BIOS。

```
cd mpu_hi309a_solution/build/build_bios/  
./build.sh
```

须知

当前的BIOS编译选项为交互式，有debug、release和clean三种编译选项，release关闭BIOS启动时的shell入口，debug打开BIOS启动过程中的shell入口，clean选项用于清理编译中间产物。

可根据具体需求选择对应的编译选项，如没有调试需要，默认应使用release编译选项。

```
please select your build option:  
options: release | debug | clean  
release : close UEFI Shell  
debug   : open  UEFI Shell  
clean   : clean build
```

Default: release

步骤2 编译完成，生成hi309a.fd文件位于“mpu_hi309a_solution/build/build_bios”目录下。

----结束

1.4 分模块编译

1.4.1 源码及编译环境准备

编译环境：

- Python 3.4+、OpenSSL 1.1.1f、CMake 3.20.5+，GLIBC_2.35及以上。
- 如果出现yaml.h不存在的错误，需要依赖libyaml-dev，请安装libyaml-dev。

1.4.1.1 源码准备

步骤1 下载release代码包并解压。

步骤2 将“[基于oebuild快速构建](#)”中输出件“openeuler-xxx.rootfs.cpio.gz”拷贝到“mpu_hi309a_solution/custom/open_euler”目录下。

----结束



1.4.1.2 编译环境准备

步骤1 从以下路径下载“openEuler-24.03-LTS”交叉编译链到“~/hi309A_tool”目录。

https://mirror.truenetwork.ru/openeuler/openEuler-24.03-LTS/embedded_img/aarch64/qemu-aarch64/openeuler-glibc-x86_64-openeuler-image-aarch64-qemu-aarch64-toolchain-24.03-LTS.sh

步骤2 在“~/hi309A_tool”目录下，执行如下命令。

```
sh openeuler-glibc-x86_64-openeuler-image-aarch64-qemu-aarch64-toolchain-24.03-LTS.sh -d /home/${USER}/hi309A_tool/toolchain/ -y \
```

步骤3 配置openEuler交叉编译环境，执行如下命令。

```
./home/${USER}/hi309A_tool/toolchain/environment-setup-aarch64-openeuler-linux
```

----结束

须知

- 路径前是“点(.) 空格()”，点和空格都不能少。
- 在分模块编译过程中，编译模块所在服务器的CMake需要在3.20.5版本及以上。

1.4.2 编译 zImage

步骤1 修改编译脚本权限。

```
cd mpu_hi309a_solution/  
chmod u+x -R build
```

步骤2 配置编译时所需要的环境。

```
cd mpu_hi309a_solution/build  
source build_prepare.sh  
source build_exports
```

步骤3 编译内核。

```
cd mpu_hi309a_solution/build/build_hi309a  
./build_kernel.sh kernel_5.10.0  
可选参数说明：  
内核版本支持：kernel_5.10.0、kernel_5.4.18。当配置缺省时使用kernel_5.10.0版本。
```

----结束

内核编译完成后会打包生成一个文件：zImage存放在“mpu_hi309a_solution/build/output”目录下。

1.4.3 编译 Hi309a SDK

步骤1 修改编译脚本权限。

```
cd mpu_hi309a_solution/  
chmod u+x -R build
```

步骤2 配置编译时所需要的环境。

```
cd mpu_hi309a_solution/build  
source build_prepare.sh  
source build_exports  
source /home/${USER}/hi309A_tool/toolchain/environment-setup-aarch64-openeuler-linux
```



步骤3 编译内核。

```
cd mpu_hi309a_solution/build/build_hi309a  
./build_kernel.sh
```

步骤4 执行如下命令编译SDK。

```
cd mpu_hi309a_solution/build/build_sdk/  
./build_sdk.sh
```

----结束

SDK编译完成后会打包生成一个文件：“hi309a_sdk.tar.gz”存放在
mpu_hi309a_solution/build/output”目录下。

1.4.4 文件系统制作

前提条件：

- 已经编译执行完“[编译Hi309a SDK](#)”，并成功编译出“hi309a_sdk.tar.gz”。
- 将“[基于oebuild快速构建](#)”中输出件“openeuler-xxx.rootfs.cpio.gz”拷贝到“mpu_hi309a_solution/custom/open_euler”目录下。
- 单板的网口如果需要使用相关的phy驱动，需要把相关的phy驱动源码编译成ko形式，同时把phy编译完成的ko放入文件系统里面，再重新制作文件系统。

文件系统制作步骤，执行如下命令编译：

```
cd mpu_hi309a_solution/build/build_fs  
./build_hi309a_cpio.sh -f tmpfs -u uart0 -i sysV
```

参数说明：

- f：表示编译出的文件系统类型，可选择tmpfs（内存文件系统）、ext4、mini
- u：表示编译出的文件系统和设备采用的串口，可选择uart0、uart1
- i：表示系统的初始化方式，可选择sysV（sysV init）、systemd

编译完成后会打包生成一个文件：“initrd_boot*.cpio.gz”存放在
“mpu_hi309a_solution/build/output”目录下。

1.4.5 设备树制作

前提条件：

- 服务器已经存在dtc生成设备树命令。
- 如果需要使用QSGMII和XGE模式，可以参考“mpu_hi309a_solution/build/build_hi309a”目录下的DTS进行配置。

DTB生成步骤：

```
cd mpu_hi309a_solution/build/build_hi309a  
./build_dtb.sh -n kernel_5.10.0 -u uart0 -b demo  
./build_dtb.sh -d user_target
```

参数说明：

- u：表示编译出的文件系统和设备采用的串口，可选择uart0、uart1
- n：表示编译指定的内核版本、DPDK版本以及对应的设备树。
- n：表示编译指定的内核版本、DPDK版本以及对应的设备树，支持缺省配置。
参数可选：内核转发（kernel_5.10.0、kernel_5.4.18），编译名为hi309a_xxx_xxx_kernel.dts的设备树；
DPDK转发（dpdk_21.11.0、dpdk_21.11.1、kernel_5.10.0_with_dpdk_21.11.0、
kernel_5.10.0_with_dpdk_21.11.1、kernel_5.4.18_with_dpdk_21.11.0、kernel_5.4.18_with_dpdk_21.11.1），编译名为hi309a_xxx_xxx_dpdk.dts的设备树，kernel_5.xx.x_with_dpdk_21.xx.x格式支持指定内核版本，仅dpdk_21.xx.x格式默认内核版本kernel_5.10.0；
lsw二层硬转（kernel_5.10.0_with_lsw，kernel_5.4.18_with_lsw），编译名为hi309a_xxx_xxx_lsw.dts的设备树；
内核和DPDK共存转发（kernel_5.10.0_and_dpdk_21.11.0、kernel_5.10.0_and_dpdk_21.11.1、kernel_5.4.18_and_dpdk_21.11.0、kernel_5.4.18_and_dpdk_21.11.1），编译名为hi309a_xxx_xxx_kernel_dpdk.dts的设备树。



参数支持缺省配置时，①参数全缺省时，转发模式为内核转发（内核版本kernel_5.10.0）；②DPDK转发模式，-n dpdk_21.11.0（默认内核版本kernel_5.10.0）。

-d：表示编译用户指定的设备树，单独编译设备树时建议使用-n参数和-d参数二选一，当-n参数和-d参数共存时将忽略-n参数中指定的设备树。设备树文件支持任意名称，参数不包含后缀.dts字符。示例：-d user_target。

-b：表示单板类型，可选择demo、m900、m901、m902。

hi309a**.dtb生成件存放在“mpu_hi309a_solution/build/output”目录下。



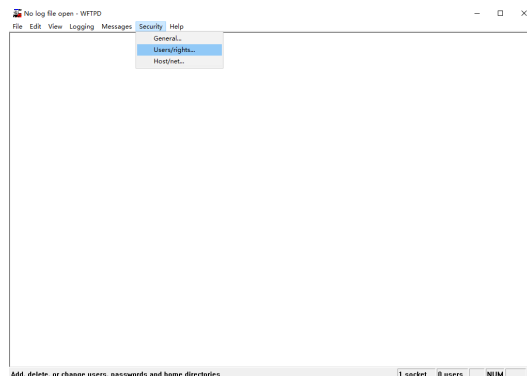
2 升级指南

2.1 wftpd32 使用指南

步骤1 下载wftpd32工具，并打开运行。

步骤2 打开wftpd32工具，界面如图2-1所示，点击Security——Users/rights…。

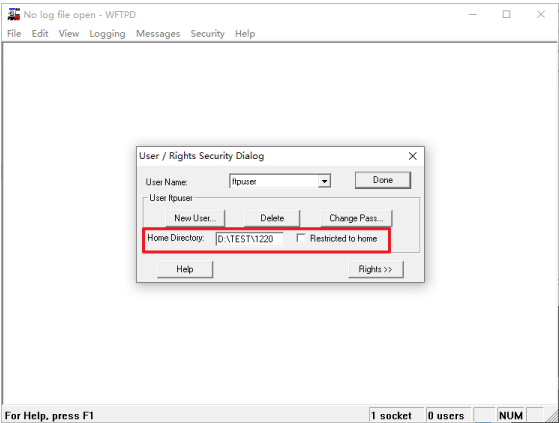
图 2-1 wftpd 工具界面



步骤3 点击“New User”创建一个用户并设置密码，设置完成后在Home Directory处设置ftp存放文件路径，如图2-2所示，并将需要上传的文件“zImage、initrd_boot.cpio.gz、hi309a**.dtb、hi309a.fd”放入该目录，如图2-3所示，如果勾选Restricted to home限定ftp用户只能看到该文件夹。



图 2-2 wftp32 创建传输目录文档路径图



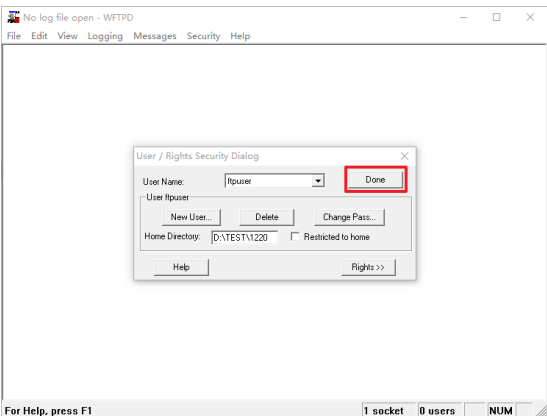
ftp路径：存放升级镜像的路径。

图 2-3 本地文件存储路径图

名称	修改日期	类型	大小
hi309a.dtb	2024/10/23 15:40	DTB 文件	53 KB
hi309a.fd	2024/10/23 15:51	FD 文件	4,032 KB
initrd_boot.cpio.gz	2024/10/23 15:40	WinRAR archive	60,420 KB
zImage	2024/10/23 15:39	文件	7,078 KB

步骤4 点击Done开启ftp服务，如图2-4所示。

图 2-4 启动 wftpd 功能图



----结束

2.2 网络配置

前提条件：

1. A-WLAN连接PC网口。
2. 单板已经上电完成。

步骤如下：

- 步骤1** 修改PC网口，控制面板-》网络和Internet-》网络和共享中心-》更改适配器-》选择（正在识别身份）的网口。
- 步骤2** 更改对应以太网的TCP/IPv4属性IP地址：192.168.0.10，子网掩码：255.255.255.0。网络配置根据需求配置，如图2-5所示。

图 2-5 网络配置图



----结束

2.3 Hi309a 升级

前提条件：

1. PC端打开串口连接工具，A-UART连接PC。
2. 通过网线将A-WLAN与PC连接。

升级步骤如下：

- 步骤1** 进入BIOS的A-UART，如果没有更新BIOS时，会进入如图2-6所示界面。如果单板已经更新BIOS及升级过“hi309a.fd”文件，当屏幕出现以下提示时，键盘输入Ctrl+T进入shell，如图2-6所示。

图 2-6 BIOS 运行图

```
begin to save log to flash
flash log full, free space = 0x212E, LogSize = 0x22FB

Press 'Ctrl+T' to skip boot exloader in 3 seconds..

map: No mapping found.
Press ESC in 1 seconds to skip startup.nsh or any other key to continue.
Shell>
Shell>
Shell> watchdog -close
Close watchdog
Shell>
Shell> mac -s 1
```

进入shell后输入以下命令：

```
关闭看门狗
watchdog -close
使能mac功能
mac -s 1
```

步骤2 在Hi309a的串口配置网络。

ifconfig eth0 IP地址 掩码 网关

目前镜像是通过A-WLAN网口进行升级，因此需要配置网络，A-WLAN的网络IP地址与电脑直连PC的以太网的IP地址需要在同一个网段，可参考如下命令进行网络配置，配置如图2-7所示。

```
ifconfig eth0 192.168.0.11 255.255.255.0 192.168.0.1
```

图 2-7 网络配置图

```
Shell> ifconfig eth0 192.168.0.11 255.255.255.0 192.168.0.1
Set eth0 IpAddr: 192.168.0.11 Mask: 255.255.255.0 Gateway: 192.168.0.1
```

步骤3 如果单板首次升级openeuler系统时，需要升级BIOS，请按照[BIOS模块编译](#)进行BIOS的编译，其中生成件位于“hi309a.fd”存放在“mpu_hi309a_solution/build/build_bios”目录下。

网口上传镜像，在串口输入如下命令。

```
ftp -get 0xa000000 eth0 192.168.0.10 hi309a.fd wftpd的账号 wftpd的密码
flash -w 0 0x0 0xa000000 0x3f0000
```

步骤4 如果重新烧录BIOS镜像后，需要重启单板，新烧录的BIOS才生效。

步骤5 网口上传镜像，在串口输入如下命令。

```
ftp -get 0xa80000 eth0 192.168.0.10 zImage wftpd的账号 wftpd的密码
```

图 2-8 zImage 升级图

```
Shell> ftp -get 0xa80000 eth0 192.168.0.10 zImage admin admin
The local IP address is: 192.168.0.12
Connected to the ftp server.
Download file from server.
File Download Successfully, Size:0x732B58
Ftp get zImage from 192.168.0.10 via eth0 success, size is 0x732B58
```

```
ftp -get 0x60000000 eth0 192.168.0.10 hi309a**.dtb wftpd的账号 wftpd的密码
```

图 2-9 设备树升级图

```
shell> ftp -get 0x60000000 eth0 192.168.0.10 hi309a.dtb admin admin
The local IP address is: 192.168.0.11
Connected to the ftp server.
Download file from server.
File Download Successfully, Size:0xB3A8
Ftp get hi309a.dtb from 192.168.0.10 via eth0 to 0x60000000 success, size is 0xB3A8
```

```
ftp -get 0x20000000 eth0 192.168.0.10 initrd_boot.cpio.gz wftpd的账号 wftpd的密码
```

图 2-10 文件系统升级图

```
shell> ftp -get 0x20000000 eth0 192.168.0.10 initrd_boot.cpio.gz admin admin
The local IP address is: 192.168.0.12
Connected to the ftp server.
Download file from server.
File Download Successfully, Size:0x4664526
Ftp get initrd_boot.cpio.gz from 192.168.0.10 via eth0 success, size is 0x4664526
```

须知

zImage、initrd_boot.cpio.gz、hi309a**.dtb、hi309a.fd名称与wftp上传的名称一致。

步骤6 直接从内存启动。

boot -b zImage的加载地址 zImage的大小 文件系统的加载地址 0x60000000 dtb设备树加载地址 设备树的大小
例如：
boot -b 0xA80000 0x1EBDA00 0x20000000 0x60000000 0x60000000 0xFFFF

步骤7 升级结果如图2-11所示。

图 2-11 openEuler 系统界面图

```
Starting Record Runlevel Change in UTMP...
[ OK ] Finished Record Runlevel Change in UTMP.
openEuler Embedded(openEuler Embedded Reference Distro) 24.03-LTS qemu-aarch64 ttyS0

openEuler
Embedded

Authorized uses only. All activity may be monitored and reported.

qemu-aarch64 login: root
You are required to change your password immediately (administrator enforced).
New password:
Retype new password:
```

登录名: root



设置密码: Huawei12#\$

----结束



3 eMMC 启动使用指南

3.1 Hi309a 首次使用

步骤1 参考如下说明修改BIOS相关代码。

目前BIOS支持Demo板，DDR主频为2133M，调试串口为UART0，若有单板进行适配，做如下修改。

1. 单板调试串口切换为UART1，如[图3-1](#)所示。

mpu_hi309a_solution/src/BIOS/BIOS/HwPkg/UEFI/Products/Network/
Platforms/Hi309a/Hi309APkg.dsc

图 3-1 BIOS 切换串口图

```
209  ## Serial Terminal
210  gEfiMdeModulePkgTokenSpaceGuid.PcdSerialRegisterBase|0xFA520000
211  gEfiMdePkgTokenSpaceGuid.PcdUartDefaultBaudRate|9600
212  gArmTokenSpaceGuid.PcdVFPEnabled|0
213  gHwTokenSpaceGuid.PcdUartClkInHz|200000000 You, 2 months ago • [manife
214  gHwTokenSpaceGuid.PcdSerialPortSendDelay|500000
215
```

2. 修改DDR主频，如[图3-2](#)所示。

mpu_hi309a_solution/src/BIOS/BIOS/HwPkg/UEFI/Products/Network/script/
hi309a/fw_hi309a.pcfx

在里面先找到对应的board_id，然后修改对应配置项里的ddrfreqm。



图 3-2 BIOS 切换 DDR 主频图

```
38 --- #CAPP_BT_1
39 --- @board_id : 0xFFFD {
40 ---   boardtype=0xFFFD
41 ---
42 ---   #cpu频率: 单位M 1600=0x640; 2000=0x7D0; 2200=0x898
43 ---   cpufreqm=0x898
44 ---
45 ---   #串口波特率: 9600=0x2580; 38400=0x9600; 115200=0x1C200
46 ---   baudrate=0x1C200
47 ---
48 ---   #DDR参数配置 0xC代表2133
49 ---   ddrfreqm = 0xC
50 ---   #DDR in rank mirror使能配置
51 ---   inrankmirroren = 1
52 ---
53 ---   kernelloaderaddr = 0xA80000
54 ---   kerneladdr = 0x20
55 ---   kernelsize = 0x1FFC000
56 ---   bootcmdloaderaddr = 0x6000000
57 ---   bootcmdaddr = 0x10000
58 ---   bootcmdsize = 0x10000
59 ---   initrdloaderaddr = 0x6020000
60 ---   initrdaddr = 0x10080
61 ---   initrdsize = 0x4000000
62 ---
63 ---   #MTI调试网口硬件配置,如果没有enable参数或者enable=0,则网口不初始化,其他参数不用配置
64 ---   #mode(0:MII 1:RGMII 2:RGMI) speed(#0:10M 1:100M 2:1000M, only RGMII support 1000M)
65 ---   #duplex(#0:MMAC_DUPLEX_HALF 1:MMAC_DUPLEX_FULL) neg_mode(#0:MMAC_NEG_DISABLE 1:define MMAC_NEG_ENABLE)
66 ---   miidrv.enable = 1
67 ---   miidrv.mode = 2
68 ---   miidrv.speed = 2
69 ---   miidrv.duplex = 1
70 ---   miidrv.neg_mode = 1
71 --- }
```

参考频率如[图3-3](#)所示。



图 3-3 DDR 频率参考图

```
DDR_FREQ_1333,  
DDR_FREQ_1400,  
DDR_FREQ_1600,  
DDR_FREQ_1800,  
DDR_FREQ_1866,  
DDR_FREQ_2000, // 10  
DDR_FREQ_2100,  
DDR_FREQ_2133,  
DDR_FREQ_2200,  
DDR_FREQ_2400,  
DDR_FREQ_2600,  
DDR_FREQ_2666,  
DDR_FREQ_2800,  
DDR_FREQ_2933,  
DDR_FREQ_3000,  
DDR_FREQ_3200, // 20  
DDR_FREQ_3400,  
DDR_FREQ_3600,  
DDR_FREQ_3800,  
DDR_FREQ_4000,  
DDR_FREQ_4200,  
DDR_FREQ_4400,  
DDR_FREQ_4600,  
DDR_FREQ_4800,  
DDR_FREQ_5000,  
DDR_FREQ_5200, // 30  
DDR_FREQ_5400,  
DDR_FREQ_5600,  
DDR_FREQ_5800,  
DDR_FREQ_6000,  
DDR_FREQ_6200,  
DDR_FREQ_6400,  
DDR_FREQ_6800,  
DDR_FREQ_7200,  
DDR_FREQ_7600,  
DDR_FREQ_8000, // 40  
DDR_FREQ_8400,  
DDR_FREQ_8800,  
DDR_FREQ_MAX  
DDR_FREQUENCY_INDEX;
```

步骤2 参考“[BIOS编译](#)”进行编译，再参考“[Hi309a升级](#)”的“[步骤1~步骤3](#)”升级为修改后的最新BIOS。

步骤3 参考“[统一镜像构建](#)”完成编译。

步骤4 重新上下电至BIOS，网口上传镜像，在串口输入如下命令。

```
watchdog -close  
mac -s 1  
ifconfig eth0 192.168.0.11 255.255.255.0 192.168.0.1  
ftp -get 0xa80000 eth0 192.168.0.10 zImage wftpd的账号 wftpd的密码
```

图 3-4 zImage 升级图

```
set eth0 IpAddr: 192.168.0.12 Mask: 255.255.255.0 Gateway: 192.168.0.1  
shell> ftp -get 0xa80000 eth0 192.168.0.10 zImage admin admin  
The local IP address is: 192.168.0.12  
Connected to the ftp server.  
Download file from server.  
File Download Successfully, Size:0x732B58  
Ftp get zImage from 192.168.0.10 via eth0 success, size is 0x732B58
```

```
ftp -get 0x6000000 eth0 192.168.0.10 hi309a**.dtb wftpd的账号 wftpd的密码
```

图 3-5 设备树升级图

```
shell> ftp -get 0x6000000 eth0 192.168.0.10 hi309a.dtb admin admin  
The local IP address is: 192.168.0.11  
Connected to the ftp server.  
Download file from server.  
File Download Successfully, Size:0xB3A8  
Ftp get hi309a.dtb from 192.168.0.10 via eth0 to 0x6000000 success, size is 0xB3A8
```

```
ftp -get 0x20000000 eth0 192.168.0.10 initrd_boot.cpio.gz wftpd的账号 wftpd的密码
```

图 3-6 文件系统升级图

```
shell> ftp -get 0x20000000 eth0 192.168.0.10 initrd_boot.cpio.gz admin admin  
The local IP address is: 192.168.0.12  
Connected to the ftp server.  
Download file from server.  
File Download Successfully, Size:0x4664526  
Ftp get initrd_boot.cpio.gz from 192.168.0.10 via eth0 success, size is 0x4664526
```

boot -b zImage的加载地址 zImage的大小 文件系统的加载地址 0x20000000 dtb设备树加载地址 设备树的大小

例如：

```
boot -b 0xA80000 0x1EBDA00 0x20000000 0x6000000 0x6000000 0x10000
```

步骤5 执行以下命令插入eMMC的ko：

```
cd /lib/udrivers/  
insmod hi309a_emmc.ko
```

步骤6 首次使用需对eMMC进行分区（如已分区请跳到**步骤7**），请务必跳过前3121512个sector，这一块用于BIOS自动启动读取镜像和grub启动分区，可以参考以下命令，从1.5G位置开始创建了一个2G大小的分区（实际大小可以自行设置）：

1. 对eMMC进行分区：fdisk /dev/mmcb1k0

图 3-7 eMMC 分区图

```
# fdisk /dev/mmcb1k0
```

2. 创建新分区：n

图 3-8 新分区图

```
Device contains neither a valid DOS partition table, nor Sun, SGI, OSF or GPI disklabel
Building a new DOS disklabel. Changes will remain in memory only,
until you decide to write them. After that the previous content
won't be recoverable.

The number of cylinders for this disk is set to 1888000.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): n
```

3. 创建主分区：p

图 3-9 主分区图

```
Partition type
  p   primary partition (1-4)
  e   extended
p
```

4. 创建p2分区：2（注：p1分区预留为grub启动分区）

图 3-10 p2 分区图

```
Partition number (1-4): 2
```

5. 设定p2分区起始sector为1.5G处：3121512

图 3-11 p2 分区起始大小图

```
First sector (16-61071359, default 16): 3121512
```

6. 设定p2分区终止sector为2G处：+2G

图 3-12 p2 分区结束大小图

```
Last sector or +size{K,M,G,T} (3121512-61071359, default 61071359): +2G
```

7. 保存分区：w

图 3-13 保存分区图

```
Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table
[ 231.104042] mmcblk0: p1 p2
```



步骤7 格式化p2分区为ext4格式:

```
umount /dev/mmcbk0p2  
mkfs.ext4 /dev/mmcbk0p2
```

步骤8 进入“mpu_solution/build/build_fs”目录下执行“./build_hi309a_cpio.sh -f ext4” (systemd引导方式执行“./build_hi309a_cpio.sh -f ext4 -i systemd”)进行“initrd_boot_emmc.cpio.gz”和“Hi309a_ext4.img”镜像的编译。

步骤9 单板配置网络，与主机处于同一网段，使用mobaxterm或其他工具传输编译出的“Hi309a_ext4.img”至单板“/tmp”目录。

步骤10 将“Hi309a_ext4.img”写入到p2分区。

```
cd /tmp  
dd if=Hi309a_ext4.img of=/dev/mmcbk0p2 bs=1M count=390
```

步骤11 重新上下电至BIOS，参考以下命令将新生成的“initrd_boot_emmc.cpio.gz”、“zImage”和“hi309a*.dtb”传输到内存。

```
watchdog -close  
mac -s 1  
ifconfig eth0 192.168.0.11 255.255.255.0 192.168.0.1  
ftp -get 0xa80000 eth0 192.168.0.10 zImage wftpd的账号 wftpd的密码  
ftp -get 0x6000000 eth0 192.168.0.10 hi309a*.dtb wftpd的账号 wftpd的密码  
ftp -get 0x20000000 eth0 192.168.0.10 initrd_boot_emmc.cpio.gz wftpd的账号 wftpd的密码
```

步骤12 将内存中的三个镜像写到eMMC，参考以下命令：

```
emmc -i  
emmc -w 0x20 0x1EBDF00 0xA80000  
emmc -w 0x10000 0x20000 0x6000000  
emmc -w 0x10100 0x4000000 0x20000000
```

须知

内存地址由**步骤11**使用的内存地址决定，文件大小根据实际情况设定，并在设置p1分区时避开对应位置，例如这里最后一条eMMC写命令为写不超过64M的文件系统，可根据实际情况进行修改，BIOS的默认参数支持小于80M的initrd，如需扩大请修改“src/BIOS/BIOS/HwPkg/UEFI/Products/Network/script/hi309a/fw_hi309a.pcf”中的对应参数。

步骤13 重新上下电即可自动autoboot至内核，根文件系统挂载在p2分区。

----结束

3.2 升级方式

后续eMMC镜像升级方式可以参考“[Hi309a首次使用](#)”的“[步骤7](#)”至“[步骤13](#)”；也可以参考以下步骤进行一键升级。

步骤1 参考“[Flash启动使用指南](#)”的“[步骤1](#)”至“[步骤3](#)”将镜像上传至单板（在Flash已有镜像的情况下可以直接执行“[Flash启动使用指南](#)”中的“[步骤5](#)”）。

步骤2 参考“[Flash启动使用指南](#)”的“[步骤6](#)”启动至内核。

步骤3 输入账号密码进入到Linux命令行，执行以下命令插入eMMC ko：

```
cd /lib/udrivers  
insmod hi309a_emmc.ko
```



步骤4 使用MobaXterm或其它工具将镜像“Hi309a_ext4.img”、“initrd_boot_emmc.cpio.gz”、“zImage”和“hi309a.dtb”上传至单板“/tmp”目录。

步骤5 在根目录下执行“./hi309a_update_emmc.sh”，脚本执行过程中需要输入一次y，进行eMMC p1分区格式化确认，脚本执行完即为升级完成，重新上下电或reboot等待自动从eMMC启动即可。

----结束



4 Flash 启动使用指南

步骤1 参考[统一镜像构建](#)完成编译后，在“build/build_fs”目录下执行“./build_hi309a_cpio.sh -f mini”进行镜像构建。

步骤2 按住“ctrl + T”进入到BIOS命令行。

步骤3 通过调试网口将镜像“initrd_boot_mini.cpio.gz”、“zImage”、“hi309a**.dtb”上传到单板，命令如下。

```
watchdog -close
mac -s 1
ifconfig eth0 192.168.0.11 255.255.255.0 192.168.0.1
ftp -get 0xa80000 eth0 192.168.0.10 zImage wftpd的账号 wftpd的密码
ftp -get 0x6000000 eth0 192.168.0.10 hi309a**.dtb wftpd的账号 wftpd的密码
ftp -get 0x20000000 eth0 192.168.0.10 initrd_boot_mini.cpio.gz wftpd的账号 wftpd的密码
```

步骤4 将镜像从内存写至Flash，命令如下。

```
flash -w 0 0x1000000 0x20000000 0x2800000
flash -w 0 0x3800000 0xa80000 0x700000
flash -w 0 0x3F00000 0x6000000 0x20000
```

须知

这里的initrd参数设计为小于40M，zImage设计为小于7M，dtb设计为小于1M，如超出可根据实际情况重新设计，三者总和理论最大值为Flash大小-BIOS大小，当前为48M。

步骤5 重新上下电至BIOS命令行，从Flash读取镜像至内存后启动，命令如下。

```
flash -r 0 0x1000000 0x20000000 0x2800000
flash -r 0 0x3800000 0xa80000 0x700000
flash -r 0 0x3F00000 0x6000000 0x20000
```

步骤6 执行以下命令启动内核。

```
boot -b 0xA80000 0x1EBDA00 0x20000000 0x6000000 0x6000000 0x20000
```

----结束



须知

由于systemd引导方式的文件系统较大，暂不支持通过Flash启动。



5 USB 启动使用指南

步骤1 编译和更新支持USB启动的BIOS镜像。

参考“[BIOS编译](#)”进行编译，再参考“[Hi309a升级](#)”的“[步骤1~步骤3](#)”升级为修改后的最新BIOS。

步骤2 将适配单板的镜像“initrd_boot.cpio.gz”、“zImage”、“hi309a.dtb”拷贝到U盘的第一个分区。

提供如下步骤作为参考：

1. 通过网络方式进入单板Linux系统。
2. 加载USB驱动。
3. 格式化u盘的第一个分区为ext4文件系统，如果没分区，先分区，执行如下命令。
`mkfs.ext4 -b 1024 /dev/sda1` 也有可能是/dev/sdx1
4. 执行mount命令挂载文件系统。
5. 通过调试网口将适配单板的镜像“initrd_boot.cpio.gz”、“zImage”、“hi309a.dtb”上传到单板，**注意：和单板匹配的镜像。**
6. 拷贝文件到U盘。

须知

可以通过其他方式拷贝文件到U盘，目前只支持ext4文件系统。

步骤3 插入U盘，断电重启单板，中途不要有其他操作，BIOS自动从U盘里加载系统。

----结束

6 GRUB 引导启动使用指南

步骤1 参考[Hi309a升级](#)章节启动进入到内存文件系统中，执行根目录下的“emmc_autopart.sh”自动分区脚本，该脚本会创建3个分区：p1分区为fat32格式，大小为500M，用于存放启动相关文件及grub程序；p2和p3分区格式为ext4，大小为2G，用于存放文件系统。

须知

emmc_autopart.sh脚本只需执行一次。

步骤2 进入到“/mnt/mmcbk0p1”目录执行vi grub.cfg可以看到默认提供的grub启动配置文件grub.cfg如[图6-1](#)所示，其中OS0_type_tempfs为内存文件系统启动（此方式目前支持的文件系统大小为90M，不通过grub启动的方式内存文件系统大小不受影响），OS1_type_emmc_fs为使用临时文件系统启动切换到/dev/mmcbk0p2分区上的文件系统，OS2_custom_define供用户自定义设置。此配置文件仅作参考，用户可以自定义修改启动配置文件。

图 6-1 参考 grub 启动配置文件

```
set timeout=3*M
set default=1*M
^M
menuentry "OS0_type_tempfs" {^M
    set root=(hd0,msdos1)^M
    linux /boot1/Image ro console=ttyS0,115200 vmlalloc=512M earlycon=uart8250,mmio32,0xfa510000 rdinit=/sbin/init maxcpus=8 ramdisk_s
    devicetree /boot1/hi309a_demo_uart0_kernel.dtb^M
    initrd /boot1/initrd_boot.cpio.gz^M
}^M
^M
menuentry "OS1_type_emmc_fs" {^M
    set root=(hd0,msdos1)^M
    linux /boot2/Image ro root=/dev/mmcbk0p2 console=ttyS0,115200 vmlalloc=512M earlycon=uart8250,mmio32,0xfa510000 rdinit=/sbin/init
    devicetree /boot2/hi309a_demo_uart0_kernel.dtb^M
    initrd /boot2/initrd_boot_emmc.cpio.gz^M
}^M
^M
menuentry "OS2_custom_define" {^M
    set root=(hd0,msdos1)^M
    linux /boot3/Image ro root=/dev/mmcbk0p3 console=ttyS0,115200 vmlalloc=512M earlycon=uart8250,mmio32,0xfa510000 rdinit=/sbin/init
    devicetree /boot3/hi309a_demo_uart0_kernel.dtb^M
    initrd /boot3/initrd_boot_emmc.cpio.gz^M
}^M
```

说明

修改grub.cfg的方式:

方式一: 进入系统之后挂载 “/dev/mmcblkp1” 分区, 进入到挂载目录, 使用vi工具编辑 grub.cfg文件, 或者配置网络然后使用mobaxterm或其他传输工具替换 “grub.cfg” 文件。

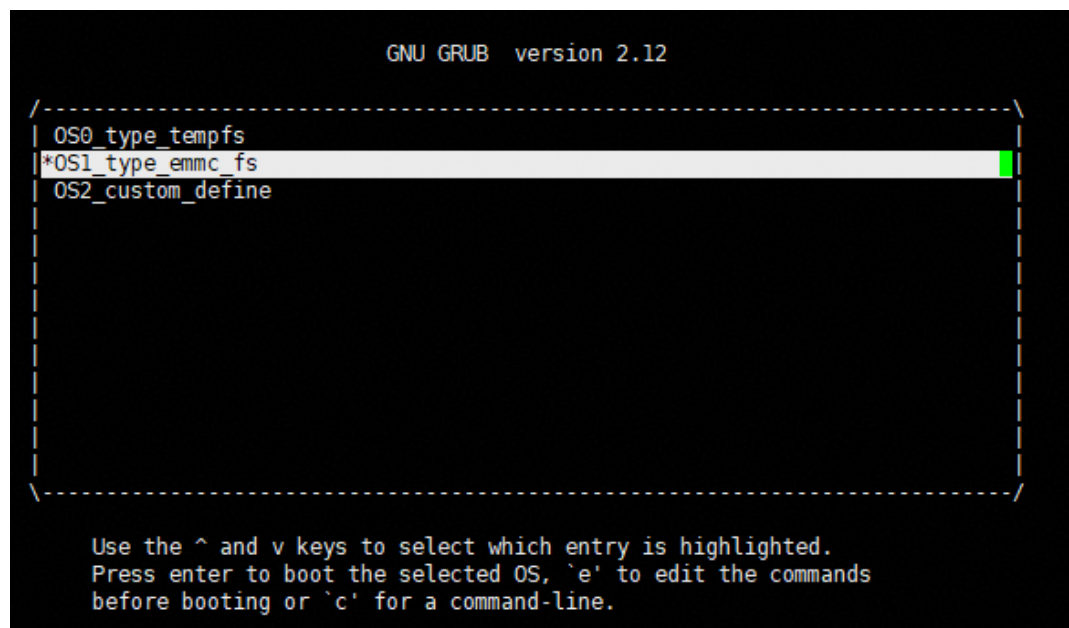
方式二: 在BIOS shell中, 配置网络后通过ftp将grub.cfg加载到内存, 然后执行memdump将文件从内存写入FS0中, 自动分区脚本配置FS0对应/dev/mmcblkp1分区。

```
watchdog -close
mac -s 1
ifconfig eth0 192.168.0.11 255.255.255.0 192.168.0.1
ftp -get 0x20000000 eth0 192.168.0.10 grub.cfg wftpd的账号 wftpd的密码
memdump 0x20000000 0x1000 grub.cfg
```

步骤3 以默认的 “grub.cfg” 为例, 单板配置网络, 与主机处于同一网段, 使用 “mobaxterm” 或其他工具传输编译出的 “Image”、“initrd_boot.cpio.gz” 和 “hi309a_demo_uart0_kernel.dtb” 文件至 “/mnt/mmcblk0p1/boot1” 目录, 传输 “Image”、“initrd_boot_emmc.cpio.gz” 和 “hi309a_demo_uart0_kernel.dtb” 文件至 “/mnt/mmcblk0p1/boot2” 目录, 请参考[Hi309a首次使用的步骤8、步骤9和步骤10](#)将文件系统Hi309a_ext4.img写入emmc的p2分区。同理可将定制化的文件系统存放于p3分区, p3分区已通过自动分区脚本默认实现, 具体ext4类型文件系统内容由客户自定义实现。修改默认系统启动项通过设置grub.cfg中的set_default字段的值来实现, 数值对应OS数值。

步骤4 重新上下电之后可以看到如[图6-2](#)所示菜单, 可通过上下键进行选择, 默认项为 OS1_type_emmc_fs。

图 6-2 grub 启动菜单



----结束