



openEuler+Baremetal

混合部署方案编译运行指南

文档版本 00B01

发布日期 2023-10-31

版权所有 © 海思技术有限公司2024。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Hi3093芯片的进行混合部署开发的程序员而写，目的是供您在开发过程中查阅混合部署所需环境以及编译步骤。

产品版本

与本文档相对应的产品版本如下。

产品名称	产品版本
Hi3093	-

读者对象

本文档（本指南）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。



符号	说明
 须知	用于传递设备或环境安全警示信息。如不可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。

日期	版本	修订说明
2023-10-31	00B01	第1次临时版本发布。



目录

前言.....	i
1 概述.....	1
1.1 概述.....	1
2 镜像制作.....	2
2.1 docker 编译构建.....	2
2.2 制作 openEuler+Baremetal 镜像.....	2
2.3 分模块编译.....	3
2.3.1 源码准备.....	3
2.3.2 编译环境准备.....	3
2.3.3 编译 Hi3093 SDK.....	4
2.3.3.1 Hi3093 SDK 编译准备.....	4
2.3.3.2 Hi3093 SDK 编译.....	4
2.3.4 uboot 编译.....	4
2.3.5 制作 uImage.....	5
2.3.6 mcs 编译.....	5
2.3.7 Baremetal 编译.....	5
2.3.8 文件系统的制作.....	6
3 混合部署升级指南.....	7
3.1 混合部署升级指南.....	7



插图目录

图 3-1 文件传输示意图.....	7
图 3-2 openEuler 侧输出示意图.....	8



1 概述

1.1 概述

混合部署框架提供了以下功能：

- openEuler文件系统的编译制作。
- 支持Baremetal以及Baremetal用例编译。
- 支持混合部署中2核使用openEuler，2核使用Baremetal的场景。

用户可通过以上提供的三个功能实现混合部署框架的方案。



2 镜像制作

2.1 docker 编译构建

参考文档：[容器环境下的快速构建指导 — openEuler Embedded在线文档 1.0.0 文档 \(gitee.io\)](#)。

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，我们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。

2.2 制作 openEuler+Baremetal 镜像

编译环境：python 3.4+，OpenSSL 1.1.1f、CMake 3.20.5+。

步骤1 下载release代码包mpu_solution.tgz并解压。

步骤2 将[2.1 docker编译构建](#)docker容器里面路径/usr1/openeuler/src/yocto-embedded-tools/config/arm64目录下的defconfig-kernel和编译生成的openeuler-image-qemu-aarch64-*.rootfs.cpio.gz、zImage拷贝到mpu_solution/custom/open_euler目录下。

步骤3 执行编译脚本。

```
cd mpu_solution/  
./build.sh
```

出现编译选项提示后，输入“oe_bm”。

步骤4 编译生成的文件在mpu_solution/build/build_hi3093/output目录下。

- **u-boot_rsa_4096.bin**: uboot。
- **Hi3093_ext4fs_cms.bin**: 文件系统签名。
- **Hi3093_ext4fs.img**: 文件系统，里面包含uImage和设备树。

----结束



⚠ 注意

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-22.03-LTS-SP1版本，用户可通过修改build.sh指定openEuler版本，openEuler系统部分的网络安全问题由用户负责。

须知

- 需要使用者具有sudo权限的账号去执行build.sh脚本。
- 执行build.sh过程中可能会让输入账户密码。
- 脚本会创建~/hi3093_tool目录，里面存放编译所依赖的工具和编译链。
- 如果有额外需要打包到二级文件系统的ko、bin、so、脚本等文件，需要将文件放到mpu_solution/custom/usr对应目录，然后再执行编译脚本。
- 如果有需要在开机启动时执行的脚本，将对应脚本文件放到mpu_solution/custom目录下，并命名为hi3093_product.sh，然后再执行编译脚本，系统启动时将会自动调用此脚本。
- mpu_solution/build/build_fs/cpio_fs保存了文件系统压缩前的文件及目录结构。

2.3 分模块编译

2.3.1 源码准备

步骤1 下载release代码包mpu_solution.tgz并解压。

步骤2 将**2.1 docker编译构建** docker容器里面路径/usr1/openeuler/src/yocto-embedded-tools/config/arm64目录下的defconfig-kernel和编译生成的openeuler-image-qemu-aarch64-*.rootfs.cpio.gz、zImage拷贝到mpu_solution/custom/open_euler目录下。

----结束

2.3.2 编译环境准备

步骤1 从以下路径下载openEuler-22.03-LTS-SP1交叉编译链到~/hi3093_tool目录。

http://121.36.84.172/dailybuild/EBS-openEuler-22.03-LTS-SP1/embedded_img/arm64/aarch64-std/openeuler-glibc-x86_64-openeuler-image-aarch64-qemu-aarch64-toolchain-22.03.sh

步骤2 执行如下命令。

```
sh openeuler-glibc-x86_64-openeuler-image-aarch64-qemu-aarch64-toolchain-22.03.sh -d /home/${USER}/hi3093_tool/toolchain/ -y \
```

步骤3 配置openEuler交叉编译环境，执行如下命令。

```
./home/${USER}/hi3093_tool/toolchain/environment-setup-aarch64-openeuler-linux
```

----结束



须知

- 在配置openEuler交叉编译环境时，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。

2.3.3 编译 Hi3093 SDK

2.3.3.1 Hi3093 SDK 编译准备

步骤1 修改编译脚本权限。

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

步骤2 从社区下载openEuler-22.03-LTS-SP1对应的内核放到mpu_solution/open_source目录下，并将目录名重命名成linux5.10，详细命令如下。

```
cd mpu_solution/open_source  
wget --no-check-certificate https://gitee.com/openeuler/kernel/repository/archive/openEuler-22.03-LTS-SP1.zip  
unzip openEuler-22.03-LTS-SP1.zip  
mv kernel-openEuler-22.03-LTS-SP1/ linux5.10
```

步骤3 配置编译时所需要的环境。

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

步骤4 编译内核。

1. 更换内核默认配置。

```
cd mpu_solution  
cp custom/open_euler/defconfig-kernel open_source/linux5.10/arch/arm64/configs/defconfig
```

2. 打patch。

```
cd mpu_solution/open_source/linux5.10  
patch -p1 < ../../src/patches/openEuler/kernel-22.03-lts-sp1-mmc.patch
```

3. 编译内核。

```
cd mpu_solution/open_source/linux5.10  
make ARCH=arm64 defconfig  
make ARCH=arm64 CROSS_COMPILE=aarch64-openeuler-linux-gnu- -j32
```

----结束

2.3.3.2 Hi3093 SDK 编译

执行如下命令编译SDK。

```
cd mpu_solution/build/build_sdk/  
./build_sdk.sh
```

SDK编译完成后会打包生成两个文件：hi3093sdk.tar.gz，securec.tar.gz存放在mpu_solution/build/output目录下。

2.3.4 uboot 编译

编译生成u-boot，执行如下命令。

```
cd mpu_solution/build/build_uboot  
./build_uboot.sh openeuler
```



命令执行完成后，在mpu_solution/build/output目录下生成u-boot_rsa_4096.bin文件。

2.3.5 制作 uImage

描述：用于openEuler的镜像的转换，zImage转换成uImage。步骤如下：

步骤1 执行第2.3.4 uboot编译的uboot制作的步骤后，将在mpu_solution/custom/open_euler目录生成mkimage可执行文件。

步骤2 将openEuler编译出的zImage制作成uImage，执行如下命令。

```
cd mpu_solution/custom/open_euler
./mkimage -A arm -O linux -T kernel -C none -a 0x98280000 -e 0x98280040 -n linux-5.10.0 -d zImage
uImage \
```

----结束

执行完成后，在mpu_solution/custom/open_euler目录下生成uImage文件。

2.3.6 mcs 编译

概述：混合关键性系统(MCS, Mixed Criticality System)，实现在一颗片上系统中部署多个OS，同时提供Linux的服务管理能力以及实时OS带来的高实时、高可靠的关键能力。软件架构如下所示。

- mcs_km: 提供OpenAMP所需内核模块，支持Client OS启动、专用中断收发、管理保留内存等功能。
- rpmsg_pty_demo: 提供OpenAMP用户态程序Linux端样例，支持在Linux上通过pty设备访问Client OS
- library: 提供OpenAMP样例必需的模块remoteproc、virtio、rpmsg、openamp。

具体编译步骤如下：

步骤1 完成2.2 制作OpenEuler+Baremetal镜像的编译后，如果需要重新打开shell窗口运行mcs模块，则需要执行如下步骤。

```
cd mpu_solution/build
source ~/hi3093_tool/toolchain/environment-setup-aarch64-openeuler-linux
source build_exports
source build_prepare.sh
```

步骤2 进入build/build_hi3093目录，执行如下命令编译对应的openAMP与Libmetal。

```
cd mpu_solution/build/build_hi3093
sh ./build_openamp.sh
```

步骤3 进入build/build_baremetal目录，执行如下命令。

```
cd mpu_solution/build/build_baremetal
sh ./build_mcs.sh
```

步骤4 生成文件：mcs_km.ko和rpmsg_main位于mpu_solution/build/build_baremetal/output/mcs_rpmsg。

----结束

2.3.7 Baremetal 编译

步骤1 进入build/build_baremetal目录，执行如下命令。

```
cd mpu_solution/build/build_baremetal
sh ./build_baremetal_2with2.sh
sh ./build_sample.sh oe_bm
```



步骤2 生成的bin文件位于mpu_solution/build/build_baremetal/output/对应目录下。

----结束

2.3.8 文件系统的制作

步骤1 将mpu_solution/custom/open_euler目录下的openeuler-image-qemu-aarch64-*.rootfs.cpio.gz和2.3.5 制作ulmage生成的ulmage放到mpu_solution/build/output目录下。

```
cd mpu_solution
cp custom/open_euler/openeuler-image-qemu-aarch64-*.rootfs.cpio.gz build/output/
cp custom/open_euler/ulmage build/output/
```

步骤2 将mpu_solution/build/output目录下的openeuler-image-qemu-aarch64-*.rootfs.cpio.gz修改为initrd_boot.cpio.gz。

```
cd mpu_solution/build/output
mv openeuler-image-qemu-aarch64-*.rootfs.cpio.gz initrd_boot.cpio.gz
```

步骤3 编译dtb。

```
cd mpu_solution/build/build_rtos
dtc -I dts -O dtb -o hi1711_asic.dtb hi3093_mcs_2with2.dts
```

命令执行完成后，在mpu_solution/build/build_rtos生成hi1711_asic.dtb文件。

步骤4 进入mpu_solution/build/build_fs目录下，使用有sudo权限的账号执行命令。

```
cd mpu_solution/build/build_fs
./build_hi3093_ext4.sh oe_bm
```

步骤5 命令执行完成后，在mpu_solution/build/output目录下生成hi3093_ext4.tar.gz文件（压缩了Hi3093_ext4fs_cms.bin和Hi3093_ext4fs.img文件），在mpu_solution/build/build_fs/cpio_fs保存了文件系统压缩前的文件及目录结构。

----结束

3 混合部署升级指南

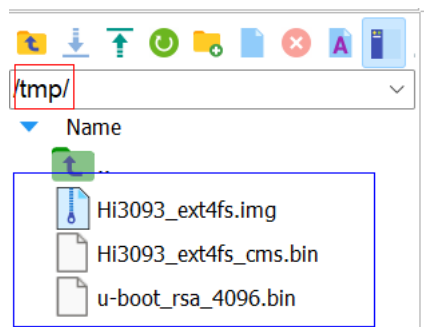
3.1 混合部署升级指南

步骤1 成功进入openEuler系统，执行以下命令。

```
sh /hi3093_init.sh (如果在系统启动阶段已自动执行过此命令，则不用再执行)  
ifconfig eth2 192.168.0.11
```

步骤2 通过MobaXterm建立ssh连接，remote host设为192.168.0.11，把第2.2 制作openEuler+Baremetal镜像中制作完成的Hi3093_ext4fs.img、Hi3093_ext4fs_cms.bin和u-boot_rsa_4096.bin通过拖曳方式放在/tmp目录下，如图3-1所示。

图 3-1 文件传输示意图



步骤3 执行如下命令。

```
cd /tmp  
sh /hi3093_upgrade.sh 4096  
reboot
```

步骤4 升级成功后，进入系统后，进行初始化和网络配置。

```
ifconfig eth2 192.168.0.11
```

步骤5 进入lib/modules目录，执行如下命令。

```
cd /lib/modules  
insmod mcs_km.ko
```

步骤6 将编译好的rpmsg_main、mcs_core2.bin、mcs_core3.bin通过网络传输到单板上的同一文件夹下(不能够放在tmp目录)。



执行如下命令。

```
./rpmsg_main -i 2 -c 2 -t mcs_core2.bin -a 0x93000000 -m mcs_core3.bin -n 0x93100000
```

须知

- rpmsg_main位于mpu_solution/build/build_baremetal/output/mcs_rpmsg中。
- mcs_core2.bin、mcs_core3.bin位于mpu_solution/build/build_baremetal/output/mcs中。
- 改变rpmsg_main和mcs_core2.bin、mcs_core3.bin的执行权限；如果rpmsg_main和mcs_core2.bin、mcs_core3.bin执行权限不够，会导致执行失败。

步骤7 执行完成**步骤6**后，出现如下打印Hello, Barmetal。

图 3-2 openEuler 侧输出示意图

```
qemu-aarch64 /lib/modules # ./rpmsg_main -i 2 -c 2 -t mcs_core2.bin -a 0x93000000 -m mcs_core3.bin -n 0x93100000
cpu:2, path:mcs_core2.bin

Initialize the virtio, virtqueue and rpmsg device
virt add:0x7fa610c000, status_reg:0x7fa610c000, tx:0x7fa6138000, rx:0x7fa6134000, mempool:0x7fa6110000
number of services: 1
start client os
[core2][sample_mcs_run:54]:bm_mcs_create_ept create first success
[core2][sample_mcs_cb:27]:data: aaaaaaaaaa

[core2][sample_mcs_run:60]:mcs prepare to destroy endpoint, receive times = 1
[core2][sample_mcs_run:67]:mcs destroy endpoint success
found matched endpoint, creating shutdown with id:1 in host os
found matched endpoint, creating console with id:2 in host os
The console endpoint is ready
H
e
l
l
o
,
B
a
r
e
m
e
t
a
l
!
```

----结束