Hi3559A/C V100 SDK 安装及升级使用说明

文档版本　00B04

发布日期　2018-05-15

商标声明

 、HISILICON、海思和其他海思商标均为深圳市海思半导体有限公司的商标。

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。

深圳市海思半导体有限公司

地址：深圳市龙岗区坂田华为基地华为电气生产中心　　邮编：518129

网址：http://www.hisilicon.com

客户服务电话：+86-755-28788858

客户服务传真：+86-755-28357515

客户服务邮箱：support@hisilicon.com

前 言

概述

本文为 Hi3559A/C V100 SDK 的安装及升级使用说明，方便使用者能快速在 Hi3559A V100 或 Hi3559C V100 DEMB 板上搭建好 SDK 运行环境。

说明

本文以 Hi3559AV100 为例，未有特殊说明，Hi3559CV100 与 Hi3559AV100 内容一致。

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
Hi3559A	V100
Hi3559C	V100

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

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。

文档版本 00B04 (2018-05-15)

第 4 次临时版本发布

1.2.3、2.4 和 6.2 小节涉及修改

文档版本 00B03 (2018-04-04)

第 3 次临时版本发布

新增第 1 章内容

文档版本 00B02 (2018-02-10)

1.6 和 2.2.1 小节涉及修改。

添加 2.2.2 小节。

文档版本 00B01 (2018-01-15)

第 1 次临时版本发布。

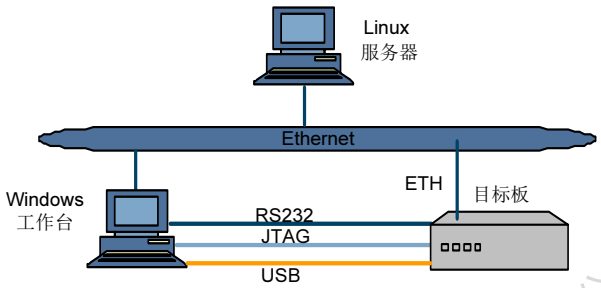
目 录

1 开发环境搭建

1.1 概述

一个典型的嵌入式开发环境通常包括 Linux 服务器、Windows 工作台和目标板，三者同处于一个网络中，以 Hi3559AV100DMEB 为例，典型开发环境如图 1-1 所示。

图1-1 Hi3559AV100 典型开发环境



在 Linux 服务器上建立交叉编译环境，为软件开发提供交叉编译服务。

Windows 工作台和 Linux 服务器共享程序，并安装超级终端，通过网口远程登录到 Linux 服务器，进行交叉编译。

Windows 工作台通过串口、网口和 USB 口与 Hi3559AV100 单板连接，可将编译后的镜像文件烧写到 Hi3559AV100 单板，并调试运行 Hi3559AV100 单板程序。

说明

开发环境中使用了 Windows 工作台，实际上很多工作也可以在 Linux 服务器上完成，如使用 minicom 代替超级终端等，用户可自行选择。

1.2 Linux 服务器开发环境搭建

推荐用户使用 64 位 Linux 服务器，本开发包在 32 位 Linux 服务器、较老版本的 Linux 服务器、偏冷门的 Linux 服务器上可能存在未知的兼容性问题。

推荐的硬件配置如下：

- CPU Intel(R) Xeon(R) CPU E5-2450 0 @ 2.10GHz 或更好 CPU
- DDR: >= 16GB
- Hard disk >= 600GB
- Gigabit Ethernet
- OS: Ubuntu 14.04 64bit

1.2.1 发布包使用的 Linux Server 版本

本示例安装如下版本 Linux 系统，本开发包默认均以此 Linux 系统进行编译：

```
ubuntu14.04_64bit_server_R5
Linux version 3.13.0-24-generic (buildd@xxx) (gcc version 4.8.2 (Ubuntu 4.8.2-19ubuntu1)) #46-Ubuntu SMP Thu Apr 10 19:11:08 UTC 2014
```

说明

ubuntu 系统安装步骤略。

1.2.2 网络环境搭建

请用户自行配置网络，并安装 nfs，samba，ssh 等网络组件。

1.2.3 软件包安装

操作系统安装好后，且用户已自行配置好网络环境，则可继续如下步骤完成相关软件包的安装：

步骤 1. 配置默认使用 bash

执行 sudo dpkg-reconfigure dash 选择 no

步骤 2. 安装软件包

执行：sudo apt-get install make libc6:i386 lib32z1 lib32stdc++6 zlib1g-dev libncurses5-dev ncurses-term libncursesw5-dev g++ u-boot-tools:i386 texinfo texlive gawk libssl-dev openssl bc

步骤 3. 创建/etc/ld.so.preload 文件，并执行 echo "" > /etc/ld.so.preload，以解决 64bit linux server 上某些第三方库编译失败的问题。

----结束

2 首次安装 SDK

如果您已安装过 SDK，可以直接参考 3 “安装、升级 Hi3559AV100 DEMO 板开发环境”。

2.1 Hi3559AV100 SDK 包位置

在"Hi3559AV100***/01.software/board"目录下，您可以看到一个 Hi3559AV100_SDK_Vx.x.x.x.tgz 的文件，该文件就是 Hi3559AV100 的软件开发包。

2.2 解压缩 SDK 包

在 linux 服务器上（或者一台装有 linux 的 PC 上，主流的 linux 发行版本均可以），使用命令：tar -zxf Hi3559AV100_SDK_Vx.x.x.x.tgz，解压缩该文件，可以得到一个 Hi3559AV100_SDK_Vx.x.x.x 目录。

2.3 展开 SDK 包内容

返回 Hi3559AV100_SDK_Vx.x.x.x 目录，运行./sdk.unpack(请用 root 或 sudo 权限执行)将会展开 SDK 包打包压缩存放的内容，请按照提示完成操作。

如果您需要通过 WINDOWS 操作系统中转拷贝 SDK 包，请先运行./sdk.cleanup，收起 SDK 包的内容，拷贝到新的目录后再展开。

2.4 在 linux 服务器上安装交叉编译器

在发布包 Hi3559A V100R001C02SPCxxx.rar 所在的目录中下载工具链文件。

注意：安装交叉编译器需要有 sudo 权限或者 root 权限。

1) 安装 aarch64 交叉编译器：

解压 tar –xzf aarch64-himix100-linux.tgz，运行 chmod +x aarch64-himix100-linux.install，然后运行./aarch64-himix100-linux.install 即可。

2) 安装 arm-none-eabi 交叉编译器：

解压 tar –xzf gcc-arm-none-eabi-4_9-2015q3.tgz，参考其中的 readme 下载工具链包，运行 chmod +x gcc-arm-none-eabi-4_9-2015q3.install，然后运行./gcc-arm-none-eabi-4_9-2015q3.install 即可。

3) 执行 source /etc/profile，安装交叉编译器的脚本配置的环境变量就可以生效了，或者请重新登陆也可。

2.5 编译 osdrv

参见 osdrv 目录下 readme。

2.6 SDK 目录介绍

Hi3559AV100_SDK_Vx.x.x.x 目录结构如下：

```
|-- drv                              # drv 目录
|    |-- extdrv                      #  板级外围驱动源代码
|    |-- interdrv                    #  片内其它模块 mipi,cipher 等驱动源代码
|-- image_glibc_multi-core_arm64     #  单 Linux 方案的烧写镜像文件
|-- image_glibc_big-little_arm64     # Linux+Huawei LiteOS 双系统方案的烧写镜像文件
|-- mpp                              #  存放单核媒体处理平台的目录
|    |-- component                   # mpp 组件
|    |    |-- isp                    # isp firmware 源代码
|    |    |-- gpu                    # gpu 驱动源代码
|    |    |-- pci                    # pciv 驱动源代码
|    |-- out                         #  发布文件
|    |    |-- linux                  # linux 发布文件
|    |    |    |-- multi-core        #  在 A73MP 单 Linux 方案上运行的驱动文件
|    |    |    |    |-- init         #  内核模块的初始化源代码
|    |    |    |    |-- obj          #  内核模块的 obj 文件
|    |    |    |    |-- include      #  头文件
|    |    |    |    |-- ko           #  内核 ko 模块
```

```
|   |   |   |   |-- lib             # 用户态 lib 库
|   |   |   |-- big-little          # 在 Linux+LiteOS 双系统方案中 A73MP Linux 上运行的驱动文件
|   |   |   |   |-- init            # 内核模块的初始化源代码
|   |   |   |   |-- obj             # 内核模块的 obj 文件
|   |   |   |   |-- include         # 头文件
|   |   |   |   |-- ko              # 内核 ko 模块
|   |   |   |   |-- lib             # 用户态 lib 库
|   |   |-- liteos                  # 内核模块的 obj 文件
|   |   |   |-- single              # 在 Linux+Huawei LiteOS 双系统方案中 A53UP 的 Huawei LiteOS 上运行的驱动文件
|   |   |   |-- dsp                 # 在 DSP 的 Huawei LiteOS 上运行的驱动文件
|   |-- sample                      # 各种业务的样例源代码
|   |-- tools                       # 媒体处理相关工具
|   |-- cfg.mak                     # mpp 配置文件
|   |-- Makefile.param              # mpp 全局编译选项
|   |-- Makefile.linux.param        # mpp linux 编译选项
|   |-- Makefile.liteos.param       # mpp liteos 编译选项
|-- osdrv                           # 存放操作系统及相关驱动的目录
|   |-- components                  # 组件源代码
|   |   |-- ipcm                    # 核间通信驱动源码
|   |   |-- pcie_mcc                # pcie_mcc 驱动源码
|   |-- opensource                  # opensource 源代码
|   |   |-- arm-trusted-firmware    # ATF 源代码
|   |   |-- busybox                 # busybox 源代码
|   |   |-- kernel                  # linux 内核的 patch
|   |   |-- uboot                   # uboot 源代码
|   |-- platform                    # 平台文件
|   |   |-- liteos_a53              # A53UP 上运行的 Huawei LiteOS 操作系统
|   |   |-- liteos_m7               # M7 上运行的 Huawei LiteOS 操作系统
|   |-- pub                         # 编译好的镜像、工具、drv 驱动等
|   |-- tools                       # 工具源代码
```

```
|     |-- readme_cn.txt          # osdrv 中文使用说明
|     |-- readme_en.txt          # osdrv 英文使用说明
|     |-- Makefile               # osdrv Makefile
|-- osal                         # 存放操作系统适配层的头文件和源文件的目录
|     |-- include                # 存放操作系统适配层的头文件的目录
|     |-- linux                  # 存放 linux 系统适配层的源文件的目录
|     |-- liteos                 # 存放 Huawei LiteOS 系统适配层的源文件的目录
|-- package                      # 存放 SDK 各种压缩包的目录
|     |-- drv.tgz                # drv 压缩包
|     |-- mpp.tgz                # 媒体处理平台软件压缩包
|     |-- osal.tgz               # 操作系统适配层源码压缩包
|     |-- hisyslink.tgz          # 核间通信模块压缩包
|     |-- osdrv.tgz              # linux 内核/uboot/rootfs/tools 源码压缩包
|-- scripts                      # 存放 shell 脚本的目录
|-- sdk.cleanup                  # SDK 清理脚本
|-- sdk.unpack                   # SDK 展开脚本
```

3 安装、升级 Hi3559AV100 DEMO 板开发环境

本章节以 Hi3559AV100 DEMO 板为例，介绍烧写 u-boot、内核以及文件系统的方法。

3.1 准备工作

首先，请阅读文档《Hi3559AV100 Demo 单板用户指南》或《Hi3559CV100 Demo 单板使用指南》，了解 Hi3559AV100 DEMO 板硬件的功能、结构、接口等信息。

然后，请阅读文档《多核 使用指南》，了解 Hi3559AV100 上的多核分布、业务部署、调试方法等信息。

可以按照以下操作来烧写 u-boot、内核以及文件系统，以下操作均使用网络来更新。

1. 如果您拿到的单板没有 uboot，就需要使用 HiTool 工具进行烧写。HiTool 工具位置放在 Hi3559A***/01.software/pc/HiTool，使用说明请参见该目录下的《HiBurn 工具使用指南》。
2. 如果您拿到的单板中已经有 uboot，可以按照以下步骤使用网口烧写 uboot、kernel 及 rootfs 到 Flash 中。

本章所有的烧写操作都在 DEMO 板上的串口座上进行。烧写到 SPI NAND Flash 上。

3.2 操作步骤

3.2.1 单系统 Linux 方案烧写步骤

步骤 1. 配置 tftp 服务器

可以使用任意的 tftp 服务器，将发布包 image_glibc_multi-core_arm64 目录下的相关文件拷贝到 tftp 服务器目录下。

步骤 2. 参数配置

单板上电后，敲任意键进入 u-boot。设置 ipaddr（单板 ip）、ethaddr（单板的 MAC 地址）和 serverip（即 tftp 服务器的 ip）。

setenv ipaddr 10.67.208.170

setenv ethaddr 00:10:ab:20:81:70

setenv netmask 255.255.254.0

setenv gatewayip 10.67.208.1

setenv serverip 10.67.209.239

ping serverip，确保网络畅通。

以上为举例，IP 以实际为准。

步骤 3. 烧写 multi-core 版本映像文件到 SPI NAND

注意：单 Linux 方案要烧写 image_glibc_multi-core_arm64 目录中的镜像文件！

1. 地址空间说明

1M	9M	16M
uboot	kernel	rootfs

以下操作基于图示的地址空间分配，也可以根据实际情况进行调整。

2. 拨码选择主 CPU

通过拨码开关 SW1.4 设置选择主 CPU：

0：从 A53MP Core0 启动；

1：从 A53UP 启动。

3. 烧写 u-boot

mw.b 0x44000000 0xff 0x100000

tftp 0x44000000 u-boot-hi3559av100.bin

nand erase 0x0 0x100000

nand write 0x44000000 0x0 0x100000

4. 烧写内核

mw.b 0x44000000 0xff 0x900000

tftp 0x44000000 uImage_hi3559av100_multi-core

nand erase 0x100000 0x900000

nand write 0x44000000 0x100000 0x900000

5. 烧写文件系统

mw.b 0x44000000 0xff 0x1000000

tftp 0x44000000 rootfs_hi3559av100_2k_24bit.yaffs2

nand erase 0xA00000 0x1000000

nand write.yaffs 0x44000000 0xA00000 0xcfeb00 （0xcfeb00 为 rootfs 文件实际大小）

6. 设置启动参数

setenv bootargs 'mem=512M console=ttyAMA0,115200 root=/dev/mtdblock2 rw rootfstype=yaffs2 mtdparts=hinand:1M(boot),9M(kernel),16M(rootfs)'

setenv bootcmd 'nand read 0x44000000 0x100000 0x900000;bootm 0x44000000'

saveenv

7. 重启系统

reset

----结束

3.2.2 双系统 Linux+Huawei LiteOS 方案烧写步骤

步骤 1. 准备 Huawei LiteOS 端的镜像

Huawei LiteOS 端的烧写镜像是由客户在 Huawei LiteOS 端的业务代码编译生成的。如果是初次调试环境，可以用发布包中的 sample 编译出来的镜像，A53 端的镜像建议使用 mpp/sample/vio/ sample_vio.bin；M7 端的镜像建议使用 osdrv/ /platform/liteos_m7/liteos/sample/sample_osdrv sample.bin。

步骤 2. 配置 tftp 服务器

可以使用任意的 tftp 服务器，将发布包 image_glibc_big-little_arm64 目录下的相关文件拷贝到 tftp 服务器目录下。将上面步骤 1 准备的 Huawei LiteOS 镜像也拷贝到 tftp 服务器目录下。

步骤 3. 参数配置

单板上电后，敲任意键进入 u-boot。设置 ipaddr（单板 ip）、ethaddr（单板的 MAC 地址）和 serverip（即 tftp 服务器的 ip）。

setenv ipaddr 10.67.208.170

setenv ethaddr 00:10:ab:20:81:70

setenv netmask 255.255.254.0

setenv gatewayip 10.67.208.1

setenv serverip 10.67.209.239

ping serverip，确保网络畅通。

以上为举例，IP 以实际为准。

步骤 4. 烧写 single+big.little 版本映像文件到 SPI NAND

注意：双系统方案要烧写 image_glibc_big-little_arm64 目录中的镜像文件！

1. 地址空间说明

1M	9M	32M	16M	1M
uboot	linux kernel	rootfs	A53UP LiteOS	M7 LiteOS

以下操作基于图示的地址空间分配，也可以根据实际情况进行调整。

2. 拨码选择主 CPU

   通过拨码开关 SW1.4 设置选择主 CPU：

   0：从 A53MP Core0 启动；

   1：从 A53UP 启动。

   需要拨码到 0，从 A53MP 启动。

3. 烧写 u-boot

   mw.b 0x52000000 0xff 0x100000

   tftp 0x52000000 u-boot-hi3559av100.bin

   nand erase 0 0x100000

   nand write 0x52000000 0 0x100000

4. 烧写 Linux 内核

   mw.b 0x52000000 0xff 0x1000000

   tftp 0x52000000 uImage_hi3559av100_big-little

   nand erase 0x100000 0x900000

   nand write 0x52000000 0x100000 0x900000

5. 烧写文件系统

   mw.b 0x52000000 0xff 0x2000000

   tftp 0x52000000 rootfs_hi3559av100_2k_24bit.yaffs2

   (注：此示例 Flash 为 2KPage 24bitECC，应以实际使用器件选择相应 rootfs)

   nand erase 0xA00000 0x2000000

   nand write.yaffs 0x52000000 0xA00000 0xd32c00

   (注：nand write.yaffs 的最后一个参数(size)，应该以实际下载的 rootfs 长度为准

   如打印：“Bytes transferred = 13839360 (d32c00 hex)”，则为：0xd32c00)

6. 烧写 A53UP Huawei LiteOS

   mw.b 0x52000000 0xff 0x1000000

   tftp 0x52000000 sample_vio.bin

   nand erase 0x2A00000 0x1000000

   nand write 0x52000000 0x2A00000 0x1000000

7. 烧写 Cortex-M7 Huawei LiteOS (此步选做)

   mw.b 0x52000000 0xff 0x100000

   tftp 0x52000000 sample.bin

   nand erase 0x3A00000 0x100000

   nand write 0x52000000 0x3A00000 0x100000

8. 设置启动参数

   setenv bootargs 'mem=512M console=ttyAMA0,115200 clk_ignore_unused rw root=/dev/mtdblock2 rootfstype=yaffs2 mtdparts=hinand:1M(boot),9M(kernel),32M(rootfs)'; sa

   不启动 Cortex-M7:

   setenv bootcmd 'nand read 0x45000000 0x2A00000 0x1000000; go_a53up 0x45000000; nand read 0x52000000 0x100000 0x900000;bootm 0x52000000'; sa

   带 Cortex-M7 启动:

   setenv bootcmd 'nand read 0x45000000 0x2A00000 0x1000000; go_a53up 0x45000000; config_m7; nand read 0x52000000 0x3a00000 0x100000; cp.b 0x52000000 0x19000000 0x100000; go_m7; nand read 0x52000000 0x100000 0x900000;bootm 0x52000000'; sa

9. 重启系统

   reset

----结束

4 开发前环境准备

4.1 管脚复用

无

4.2 连接串口

通过 DEMO 板上的串口连接主 CPU。

4.3 NFS 环境

通过 DEMO 板上的网口连接 NFS。

5 使用 SDK 和 DEMO 板进行开发

5.1 开启 Linux 下的网络

步骤 1. 设置网络

ifconfig eth0 hw ether 00:10:67:20:81:70;

ifconfig eth0 10.67.208.170 netmask 255.255.254.0;

route add default gw 10.67.208.1

以上为举例，IP 以实际为准。

步骤 2. 然后 ping 一下其他 IP，正常情况下网络将能正常工作。

----结束

5.2 使用 NFS 文件系统进行开发

步骤 1. 在开发阶段，推荐使用 NFS 作为开发环境，可以省去重新制作和烧写根文件系统的工作。

步骤 2. 挂载 NFS 文件系统的操作命令：

mount -t nfs -o nolock -o tcp -o rsize=32768,wsize=32768 xx.xx.xx.xx:/your-nfs-path /mnt

步骤 3. 然后就可以在/mnt 目录下访问服务器上的文件，并进行开发工作。

----结束

5.3 开启 telnet 服务

网络正常后，运行命令 telnetd& 就可以启动单板 telnet 服务，然后才能使用 telnet 登录到单板。

5.4 运行 MPP 业务

步骤 1. 进入 mpp/out/linux/multi-core/ko 目录，加载驱动，例：

```
cd mpp/out/linux/multi-core/ko
./load3559av100_multicore -i -sensor0 imx477 -sensor1 imx477
```

步骤 2. 进入各 sample 目录下执行相应样例程序(sample 需要先在服务器上成功编译过)

```
cd mpp/sample/vio
./sample_vio 0 0
```

----结束

6 地址空间分配与使用

6.1 DDR 内存管理说明

1. 所有 DDR 内存中，一部分由操作系统管理，称为 OS 内存；另一部分由 osal 模块管理，供媒体业务单独使用，称为 MMZ 内存。
2. Single 端的 OS 内存起始地址为 0x42000000，内存大小可通过 bootargs 进行配置，例如第三章中的 setenv bootargs 'mem=256M ... ',表示分配给 Single OS 操作系统内存为 256M，您可以根据实际情况进行调整。
3. MMZ 内存由 osal 内核模块管理（mpp/ko 目录下的 hi_osal.ko），加载 osal 模块时，通过模块参数指定其起始地址及大小, 可在 load 脚本中修改 MMZ 的起始地址 mmz_start 及大小 mmz_size。
4. 请注意任何区域的内存划分都不能重叠。

6.2 DDR 内存分配

参考《Hi3559A/C V100 开发环境用户指南》第 9 章 Hi3559AV100 内存分配。

注意:

1. 任何用途的内存区域地址空间都不能重叠。
2. OS 的管理内存在配置启动参数时设置，默认为 512M，“setenv bootargs 'mem=512M …' ”。
3. MMZ 区域内存在 load 脚本中设定，如果有特殊区域需求，可以单独分区。例如“insmod hi_osal.ko anony=1 mmz_allocator=hisi mmz=anonymous,0,0x64000000,2048M:jpeg,0,0xe4000000,16M || report_error ”。
4. 双系统方案中，每个系统都有自己管理的 MMZ 区域，其中有些区域需要映射到其他系统中作为共享区域使用。在加载 MMZ 驱动时，需要指定要映射的其他系统中的 MMZ 区域地址范围。

   例如，Linux 系统的 load 脚本中“insmod hi_osal.ko anony=1 mmz_allocator=hisi mmz=anonymous,0,0x70000000,512M map_mmz=0x90000000,2816M”。其中 map_mmz 模块参数指定 LiteOS 的 MMZ 区域 0x90000000 开始，2816M 长度的内存可以映射到 Linux 系统中。

同理，Huawei LiteOS 系统的 sdk_init.c 中 media_mem_init 的参数需要指定 Linux 端的 MMZ 内存要映射到 LiteOS 系统的物理地址范围。

5. 双系统方案中，Huawei LiteOS 系统要为 MMZ 区域建立页表，需要将 MMZ 区域的地址范围配置给 HuaweiLiteOS，配置放在 osdrv 根目录下面的 osdrv_mem_cfg.sh 中。