



# Hi3519AV100 SDK 安装及升级使用说明

文档版本 00B03  
发布日期 2018-08-08

版权所有 © 深圳市海思半导体有限公司 2018。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](mailto:support@hisilicon.com)



# 前言

## 概述

本文为 Hi3519AV100 SDK 的安装及升级使用说明，方便使用者能快速在 Hi3519AV100 DEMB 板上搭建好 SDK 运行环境。

## 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
Hi3519A	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 00B03 (2018-08-08)

第 3 次临时版本发布。

1.6、4.4 小节涉及修改

2.1.2 小节步骤 3 涉及修改



## 文档版本 00B02 (2018-07-06)

第 2 次临时版本发布。

1.4、1.6、4.4、2.1.2 及 5.2 小节涉及修改

将 FPGA 板改为 DEMO 板

## 文档版本 00B01 (2018-05-15)

第 1 次临时版本发布。



# 目 录

前 言.....	i
1 首次安装 SDK.....	1
1.1 Hi3519AV100 SDK 包位置.....	1
1.2 解压缩 SDK 包.....	1
1.3 展开 SDK 包内容.....	1
1.4 在 linux 服务器上安装交叉编译器.....	1
1.5 编译 osdrv.....	2
1.6 SDK 目录介绍.....	2
2 安装、升级 Hi3519AV100 DEMO 板开发环境.....	5
2.1 烧写 uboot、kernel、fs.....	5
2.1.1 准备工作.....	5
2.1.2 操作步骤.....	5
3 开发前环境准备.....	7
3.1 管脚复用.....	7
3.2 连接串口.....	7
3.3 NFS 环境.....	7
4 使用 SDK 和 DEMO 板进行开发.....	8
4.1 开启 Linux 下的网络.....	8
4.2 使用 NFS 文件系统进行开发.....	8
4.3 开启 telnet 服务.....	8
4.4 运行 MPP 业务.....	9
5 地址空间分配与使用.....	10
5.1 DDR 内存管理说明.....	10
5.2 DEMO 板 DDR 内存管理示意.....	10



# 1 首次安装 SDK

如果您已安装过 SDK，可以直接参看 [2 安装、升级 Hi3519AV100 DEMO 板开发环境](#)。

## 1.1 Hi3519AV100 SDK 包位置

在"Hi3519AV100\*\*\*/01.software/board"目录下，您可以看到一个 Hi3519AV100\_SDK\_Vx.x.x.x.tgz 的文件，该文件就是 Hi3519AV100 的软件开发包。

## 1.2 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 Hi3519AV100\_SDK\_Vx.x.x.x.tgz，解压缩该文件，可以得到一个 Hi3519AV100\_SDK\_Vx.x.x.x 目录。

## 1.3 展开 SDK 包内容

返回 Hi3519AV100\_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。

## 1.4 在 linux 服务器上安装交叉编译器

在发布包 Hi3519A 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 himix200 交叉编译器：



解压 `tar -xzf arm-himix200-linux.tgz`, 运行 `chmod +x arm-himix200-linux.install`, 然后运行 `./arm-himix200-linux.install` 即可。

2) 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。

## 1.5 编译 osdrv

参见 `osdrv` 目录下 `readme`

## 1.6 SDK 目录介绍

Hi3519AV100\_SDK\_Vx.x.x.x 目录结构如下:

<code>-- smp</code>	<code>#smp 目录</code>
<code>   -- a53_linux</code>	
<code>     -- drv</code>	<code># drv 目录</code>
<code>     -- extdrv</code>	<code># 板级外围驱动源代码</code>
<code>     -- interdrv</code>	<code># mipi,cipher 等驱动源代码</code>
<code>   -- mpp</code>	<code># 存放单核媒体处理平台的目录</code>
<code>     -- component</code>	<code># mpp 组件</code>
<code>        -- isp</code>	<code># isp 相关组件</code>
<code>     -- init</code>	<code># 内核模块的初始化源代码</code>
<code>     -- obj</code>	<code># 内核模块的 obj 文件</code>
<code>     -- include</code>	<code># 头文件</code>
<code>     -- ko</code>	<code># 内核 ko 模块</code>
<code>     -- lib</code>	<code># 用户态 lib 库</code>
<code>     -- sample</code>	<code># 样例源代码</code>
<code>     -- tools</code>	<code># 媒体处理相关工具</code>
<code>     -- cfg.mak</code>	<code># mpp 配置文件</code>
<code>     -- Makefile.param</code>	<code># mpp 全局编译选项</code>
<code>     -- Makefile.linux.param</code>	<code># mpp linux 编译选项</code>
<code>   -- osal</code>	<code># 存放操作系统适配层的头文件和源文件的目录</code>
<code>     -- include</code>	<code># 存放操作系统适配层的头文件的目录</code>
<code>     -- linux</code>	<code># 存放 linux 系统适配层的源文件的目录</code>



<code>--dsp_liteos</code>	# dsp 目录
<code>  -- dsp</code>	# dsp 目录
<code>      -- dsp0</code>	# dsp0 目录
<code>         -- algo</code>	# 帧级算子
<code>            -- include</code>	#帧级算子头文件
<code>               -- ext</code>	#帧级算子对外头文件
<code>               -- int</code>	#帧级算子对内头文件
<code>            -- src</code>	#帧级算子源文件
<code>         -- dccs</code>	#核间通信库
<code>         -- include</code>	#公共头文件
<code>         -- lib</code>	#Tile 级算子库
<code>            -- include</code>	#Tile 级算子头文件
<code>         -- liteos</code>	#Liteos
<code>            -- dsp0_ldscripts</code>	#dsp0 链接配置目录
<code>               -- ldscripts</code>	#连接脚本
<code>            -- ipcm</code>	#ipcm 库
<code>            -- liteos</code>	#liteos 库
<code>         -- runtime</code>	# runtime
<code>            -- bin</code>	#可执行目录
<code>            -- obj</code>	#runtime obj 目录
<code>         -- tm</code>	# tile manager
<code>            -- include</code>	#头文件
<code>            -- src</code>	#源文件
<code>-- osdrv</code>	# 存放操作系统及相关驱动的目录
<code>  -- component</code>	# 组件源代码
<code>  -- opensource</code>	# opensource 源代码
<code>      -- busybox</code>	# busybox 源代码
<code>      -- kernel</code>	# linux 内核源代码
<code>      -- uboot</code>	# uboot 源代码
<code>  -- platform</code>	# 平台文件
<code>  -- pub</code>	# 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
-- package              # 存放 SDK 各种压缩包的目录
-- drv.tgz               # drv 压缩包
-- mpp_smp_linux.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
-- osdrv.tgz             # linux 内核/uboot/rootfs/tools 源码压缩包
                        和 rootfs 的压缩包
-- scripts              # 存放 shell 脚本的目录
-- sdk.cleanup           # SDK 清理脚本
-- sdk.unpack            # SDK 展开脚本
```



# 2 安装、升级 Hi3519AV100 DEMO 板开发环境

本章节以 Hi3519AV100 DEMO 板为例，介绍烧写 u-boot、内核以及文件系统的方法。

## 2.1 烧写 u-boot、kernel、fs

### 2.1.1 准备工作

首先，请阅读文档《Hi3519AV100 Demo 单板用户指南》，了解 Hi3519AV100 DEMO 板硬件的功能、结构、接口等信息。

1. 如果您拿到的单板没有 u-boot，就需要使用 HiTool 工具进行烧写。HiTool 工具位置放在 Hi3519A\*\*\*/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 u-boot，可以按照以下步骤使用网口烧写 u-boot、kernel 及 rootfs 到 Flash 中。

本章所有的烧写操作都是在串口上进行，烧写到 SPI NOR Flash 上。

### 2.1.2 操作步骤

#### 步骤 1. 配置 tftp 服务器

可以使用任意的 tftp 服务器，将 package/smp\_image\_uclibc\_xxx(或 image\_uclibc\_xxx)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 serverip（即 tftp 服务器的 ip）、ipaddr（单板 ip）和 ethaddr（单板的 MAC 地址）。

```
setenv serverip xx.xx.xx.xx
```

```
setenv ipaddr xx.xx.xx.xx
```

```
setenv ethaddr xx:xx:xx:xx:xx:xx
```



```
setenv netmask xx.xx.xx.xx  
setenv gatewayip xx.xx.xx.xx  
ping serverip, 确保网络畅通。
```

### 步骤 3. SMP 版本烧写映像文件到 SPI Nand Flash

#### 1. 地址空间说明

1M   4M   32M
----- ----- -----
boot   kernel   rootfs

以下的操作均基于图示的地址空间分配，您也可以根据实际情况进行调整。

#### 2. 烧写 u-boot

```
mw.b 42000000 ff 80000  
tftp 0x42000000 u-boot-hi3519av100.bin  
nand erase 0 80000;nand write 42000000 0 80000
```

#### 3. 烧写内核

```
mw.b 42000000 ff 400000  
tftp 42000000 uImage_hi3519av100_smp  
nand erase 100000 400000;nand write 42000000 100000 400000
```

#### 4. 烧写文件系统

```
mw.b 42000000 ff 2000000  
tftp 42000000 rootfs_hi3519av100_2k_4bit.yaffs2  
nand erase 500000 2000000;nand write.yaffs 42000000 500000 0xafeb00 (0xafeb00 为  
实际文件大小)
```

#### 5. 设置启动参数（注意 linux-4.9.y kernel 默认文件系统只读，需要在 bootargs 中加入 rw 选项，文件系统才可读写）

```
setenv bootargs 'mem=256M console=ttyAMA0,115200 clk_ignore_unused  
root=/dev/mtdblock2 rw rootfstype=yaffs2  
mtdparts=hinand:1M(boot),4M(kernel),32M(rootfs)'  
setenv bootcmd 'nand read 0x42000000 100000 400000; bootm 0x42000000'  
saveenv
```

#### 6. 重启系统

```
reset
```

----结束



# 3 开发前环境准备

## 3.1 管脚复用

无

## 3.2 连接串口

通过 DEMO 板的串口连接到 CPU。

## 3.3 NFS 环境

通过 DEMO 板的网口连接 NFS。



# 4 使用 SDK 和 DEMO 板进行开发

## 4.1 开启 Linux 下的网络

步骤 1. 设置网络

```
ifconfig eth0 hw ether xx:xx:xx:xx:xx:xx;  
ifconfig eth0 xx.xx.xx.xx netmask xx.xx.xx.xx;  
route add default gw xx.xx.xx.xx
```

步骤 2. 然后 ping 一下其他机器，如无意外，网络将能正常工作。

----结束

## 4.2 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,wsiz=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

## 4.3 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。



## 4.4 运行 MPP 业务

步骤 1. 在串口上，进入 mpp/ko 目录，加载驱动，例：

```
cd mpp/ko
./load3519av100 -i -sensor0 imx334
```

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
```

----结束



# 5 地址空间分配与使用

## 5.1 DDR 内存管理说明

- 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。
- OS 内存起始地址为 0x32000000，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。
- 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。
- 请注意任何区域的内存划分都不能重叠。

## 5.2 DEMO 板 DDR 内存管理示意

以容量为 2G Bytes 的 DDR 内存为例，以下为根据本文档和 SDK 默认配置得到的内存管理示意图：

DDR:

----- -----		0x20000000	# Memory managed by DSP.
32M	DSP		
----- -----		0x22000000	# Memory managed by OS.
256M	Single OS		
----- -----		0x32000000	# Memory managed by MMZ.
1760M	MMZ		
----- -----		0xA0000000	

注意：

- 用户在配置启动参数时需要设置 OS 的管理内存为 256M，“setenv bootargs 'mem=256M ...’”。



- load3519av100 脚本默认则 MMZ 管理的内存地址从 0x32000000 起始，大小为 1760M，用户可以根据实际使用情况自行修改。
- 任何用途的内存区域地址空间都不能重叠。
- 如果有特殊应用，可以自行修改 load3519av100 脚本，进行 mmz 区域划分，如  
“insmod hi\_osal.ko mmz\_allocator=hisi  
mmz=anonymous,0,0x32000000,30M;jpeg,0,0x33E00000,2M”。