|  |
| --- |
| Neusoft |
| 镜像制作与烧写 |
|  |
|  |
| **lv-k@neusoft.com** |
| **2010-11-16** |

本文详细叙述内核编译、文件系统镜像制作、内核烧写、文件系统镜像烧写、以及从烧写的内核和文件系统启动的NBL设置方法。

**综述**

步骤比较复杂，具体参见后面的"详细内容"，这里先给出操作过程：

一、,编译内核

$svn checkout http://lv-k@10.1.29.125/svn/sony/trunk/Source/Kernel

$cd Avp13\_Kernel

$ make avp13\_defconfig

$ make

二、制作文件系统镜像

$svn checkout http://lv-k@10.1.29.125/svn/sony/trunk/Source/Rootfs

$cd For\_Release

$./build.sh all

三、设置烧写分区

启动板子后，依次运行：

NBL> sdm init

NBL> diag ethaddr 00:11:12:33:44:55

NBL> sdm create -n -l 0x400000 kernel

NBL> sdm create -n -l 0x800000 initrd\_fat

NBL> fdisk -a sdme::

NBL> mkfs sdme1:uvfat:

NBL> mkdir sdme1:uvfat:/boot

四、设置NBL环境

NBL> set boot\_option "console=ttyS0,115200n8 root=/dev/ram0 ip=10.1.29.xx1::10.1.29.1:255.255.255.0"

NBL> set initrd "sdme1:uvfat:/boot/ramdisk.img"

NBL> set console on

NBL> set myip 10.1.29.xx1

NBL> set gateip 10.1.29.1

NBL> set servip 10.1.29.xx2

五、烧写内核和文件系统,启动系统

将文件系统镜像和内核拷贝到tftp导出目录之后，如下执行：

NBL> cp net:tftp:vmlinux.bin sdmd::

NBL> cp net:tftp:ramdisk.img sdme1:uvfat:/boot/

NBL> boot -r sdmd:raw:vmlinux.bin

至此，内核与文件系统从板子里面正常启动了。

目录

[一、编译内核 4](#_Toc277675972)

[二、制作initrd文件系统镜像 4](#_Toc277675973)

[1,创建空白镜像 4](#_Toc277675974)

[2,将文件系统拷入空白镜像 4](#_Toc277675975)

[3,压缩镜像 4](#_Toc277675976)

[三、准备烧写分区 5](#_Toc277675977)

[1,初始化sdm设备 5](#_Toc277675978)

[2,初始化mac地址 5](#_Toc277675979)

[3,创建内核烧写区域 5](#_Toc277675980)

[4,创建文件系统烧写分区 5](#_Toc277675981)

[四、烧写内核和文件系统 7](#_Toc277675982)

[1,将内核从tftp上拷贝到内核烧写区域 7](#_Toc277675983)

[2,将文件系统镜像从tftp上面拷贝到文件烧写分区 7](#_Toc277675984)

[五、配置uboot环境 8](#_Toc277675985)

[1．配置内核启动参数 8](#_Toc277675986)

[2．配置文件系统镜像位置 8](#_Toc277675987)

[六、从板子启动加载内核并启动文件系统 8](#_Toc277675988)

[七、补充 9](#_Toc277675989)

# 一、编译内核

$svn checkout http://lv-k@10.1.29.125/svn/sony/trunk/Source/Kernel

$cd Avp13\_Kernel

$ make avp13\_defconfig

$ make

注意，

\*编译内核之前，环境变量设置为"export PATH=/usr/local/arm-sony-linux-gnueabi/devel/bin/:$PATH"

\*生成的内核avp13Build/vmlinux.bin

# 二、制作initrd文件系统镜像

## 1,创建空白镜像

#dd if=/dev/zero of=ramdisk bs=1M count=16

#mkfs.ext2 -m0 ramdisk

## 2,将文件系统拷入空白镜像

#mkdir tmpmnt

#mount -o loop ramdisk tmpmnt

#cp -a rootfs/\* tmpmnt

这里，rootfs是文件系统目录,注意，rootfs里面的"dev/ram0"不可少。

## 3,压缩镜像

#umount tmpmnt

#gzip -v9 ramdisk && mv $\_.gz $\_.img

至此，文件系统镜像创建完毕，可以把它拷贝到你的tftp目录,以便烧写:

#cp ramdisk.img /tftpboot

注：

pc版文件系统镜像制作方法略有不同，如下：

(1)（前面方法没有实践）生成pc版本文件系统镜像如下：

#cd rootfs

#find . | cpio --quiet -c -o | gzip -9 -n > ../ramdisk.img

这里，rootfs是适合x86的pc上的文件系统，生成的镜像放在rootfs同级目录下。

(2)运行镜像时需要:

a)#cp ../ramdisk.img /boot

b)修改/boot/grub/grub.conf

修改添加内容参考grub.conf原有内容。

# 三、准备烧写分区

## 1,初始化sdm设备

NBL> sdm init

## 2,初始化mac地址

NBL> diag ethaddr 00:11:12:33:44:55

这个最好不要与别人冲突。

## 3,创建内核烧写区域

NBL> sdm create -n -l 0x400000 kernel

这里，区域名为kernel,大小指定4M.

## 4,创建文件系统烧写分区

1）创建文件系统烧写所在区域

NBL> sdm create -n -l 0x800000 initrd\_fat

这里，指定文件系统镜像大小8M。

2)在区域上创建一个分区：

在区域上创建一个分区：

NBL> fdisk -a sdme::

操作之后，会输出如下信息：

Num. Boot Start Length SystemID

1 8 16376 0x83 (Linux native (ext2))

3）在分区上创建fat文件系统:

NBL> mkfs sdme1:uvfat:

4)创建一个目录：

NBL> mkdir sdme1:uvfat:/boot

至此，烧写分区创建完毕。这时候，查看当前分区情况如下：

NBL> sdm list

sdma: nblconfig

device : 0

start sect : 0x88

image size : 0x20000 bytes(0x20 sectors)

protect : Yes

sdmb: sdmreg

device : 0

start sect : 0xa8

image size : 0x20000 bytes(0x20 sectors)

protect : Yes

sdmc: sdmlog

device : 0

start sect : 0xc8

image size : 0x20000 bytes(0x20 sectors)

protect : Yes

sdmd: kernel

device : 0

start sect : 0xe8

image size : 0x400000 bytes(0x400 sectors)

protect : No

sdme: initrd\_fat

device : 0

start sect : 0x4e8

image size : 0x800000 bytes(0x800 sectors)

protect : No

其中sdma,sdmb,sdmc是执行"sdm init"之后就有了；而sdmd,sdme在"sdm create -n -l 0x800000 initrd\_fat"之后就有了.

另外,需要注意：

(1)“准备烧写分区”这个步骤只需要进行一次，以后除非重新布局，否则不用再做。

(2)可以在NBL下运行一些文件操作的命令,例如：

\*创建的目录内容：

NBL> ls -l sdme1:uvfat:/boot

输出如下：

drwxrwxrwx root 4096 2000/01/01 00:00:00 ./

drwxrwxrwx root 32768 2000/01/01 00:00:00 ../

这里，时间是固定的"2000/01/01 00:00:00".

(3)可以操作块，例如:

删除刚才创建的区域sdme：

NBL> sdm delete fat

(4)创建的分区号不要超过sdmp.总大小不要超过64M.

从文档"avp13\_nbl\_spec.txt"的第三节"3. Memory map of NBL"看，总物理内存512M，但是NBL只用其中64M，其他被别的微处理器使用了。

# 四、烧写内核和文件系统

## 1,将内核从tftp上拷贝到内核烧写区域

NBL> cp net:tftp:vmlinux.bin sdmd::

这样，内核烧写完毕.实际这个时候如果配置好uboot(即NBL)环境，大家就可以用:

boot -r -c "console=ttyS0,115200n8 root=/dev/nfs nfsroot=10.1.29.44:/home/quietheart/nfs ip=10.1.29.180::10.1.29.1:255.255.255.0" sdmd:raw:vmlinux.bin

从nfs上启动自己的文件系统了。

## 2,将文件系统镜像从tftp上面拷贝到文件烧写分区

NBL> cp net:tftp:ramdisk.img sdme1:uvfat:/boot/

这里，在我的机器上面，拷贝之后输出如下信息：

i8254x: Ethernet address: 00:11:22:33:44:55

# [if\_i8254x.c] i8254x\_set\_phy\_to\_mac():822: Auto-negotiation failed: MII\_GBCR=0x00000200, MII\_GBSR=0x%, MII\_ANAR=

0x%, MII\_ANLPAR=0x%

i8254x: Ethernet address: 00:11:22:33:44:55

# [if\_i8254x.c] i8254x\_set\_phy\_to\_mac():822: Auto-negotiation failed: MII\_GBCR=0x00000200, MII\_GBSR=0x%, MII\_ANAR=

0x%, MII\_ANLPAR=0x%

这里，需要注意文件系统镜像不要太大。最好用"rm"命令先删除"sdme1:uvfat:/boot"中无用的文件。

# 五、配置uboot环境

## 1．配置内核启动参数

NBL> set boot\_option "console=ttyS0,115200n8 root=/dev/ram0 ip=10.1.29.180::10.1.29.1:255.255.255.0"

## 2．配置文件系统镜像位置

NBL> set initrd "sdme1:uvfat:/boot/ramdisk.img"

这里，我的uboot全部配置如下：

NBL> print

console=on

myip=10.1.29.180

ethaddr=0011223344559966

gateip=10.1.29.1

servip=10.1.29.44

boot\_option=console=ttyS0,115200n8 root=/dev/ram0 ip=10.1.29.180::10.1.29.1:255.255.255.0

initrd=sdme1:uvfat:/boot/ramdisk.img

# 六、从板子启动加载内核并启动文件系统

启动系统命令如下：

NBL> boot -r sdmd:raw:vmlinux.bin

至此，内核与文件系统从板子里面正常启动了。

另外，如果烧的内核是压缩版的话，例如vmlinux.bin.gz

那么用

NBL> boot -r -d sdmd:raw:vmlinux.bin.gz

也能启动。

# 七、补充

无。

以上,有问题请联系

Email:lv-k@neusoft.com