

根据之前的分析，接下来我们需要集中精力来学习一下UBoot的有关内容。当然了，无论是从Flash启动还是从SD启动，UBoot总是要事先准备好的。其实板子出厂的时候，Flash里已经烧好了UBoot、Kernel和文件系统，所以这部分我们就简单来，直接利用Flash中的UBoot，来通过网络启动我们的kernel和文件系统。如果板子烧成了裸机程序，没有UBoot的话，利用烧写了UBoot的SD卡启动板子也是OK的，而且利用UBoot还可以把UBoot烧写到Flash里，但是这部分内容就不是我们这次的重点，就暂且不展开详解了。

在PC上打开串口工具，然后启动开发板，在看到提示信息的时候随便拍拍键盘，启动过程就暂停了，进入了UBoot的提示符。UBoot可以支持许多的命令，可以用于读写传输数据什么的。



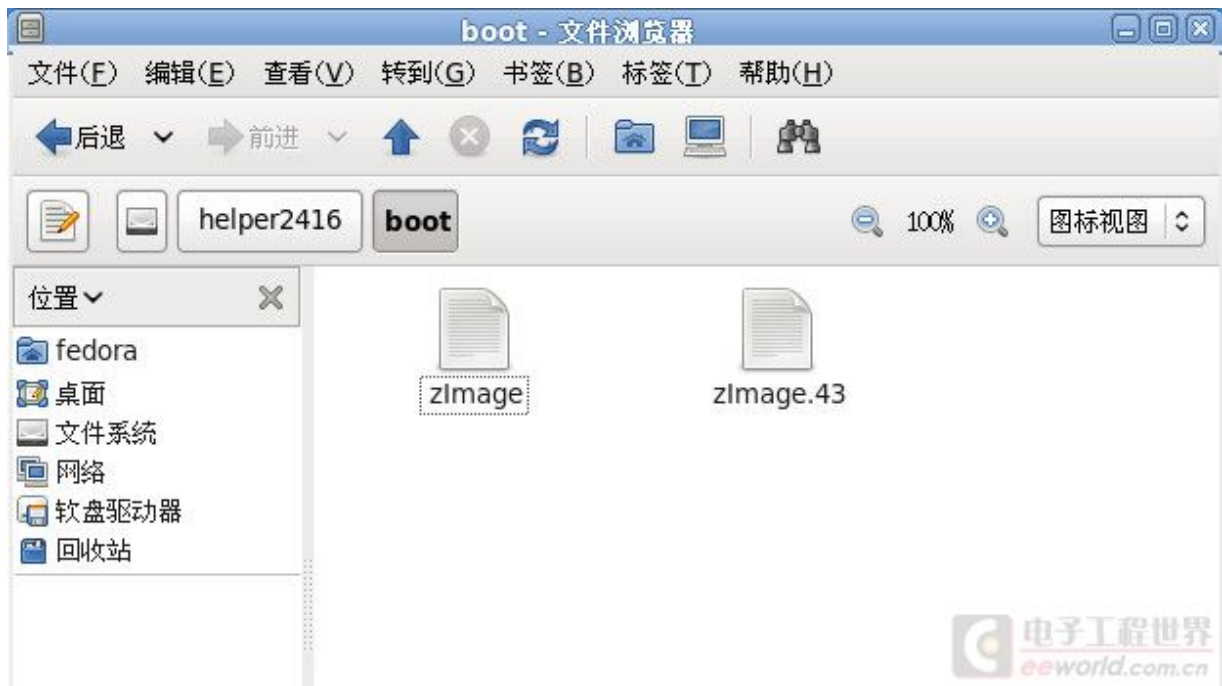
```
fedora@localhost:~  
文件(F) 编辑(E) 查看(V) 终端(T) 帮助(H)  
OK  
U-Boot 1.3.4 (Mar 21 2014 - 14:13:41) for SMDK2416  
  
CPU: S3C2416@534MHz  
Fclk = 534MHz, Hclk = 133MHz, Pclk = 66MHz  
Board: SMDK2416 DDR2  
DRAM: 64 MB  
Flash: 1 MB  
NAND: 256 MB  
In: serial  
Out: serial  
Err: serial  
smc911x: MAC 00:40:5c:26:0a:5b  
Hit any key to stop autoboot: 0  
Helper2416 #  
Helper2416 #
```

The screenshot shows a terminal window titled 'fedora@localhost:~'. The menu bar includes '文件(F)', '编辑(E)', '查看(V)', '终端(T)', and '帮助(H)'. The output shows the U-Boot version '1.3.4' and its configuration for 'SMDK2416'. It lists hardware details: CPU 'S3C2416@534MHz', clocks 'Fclk = 534MHz, Hclk = 133MHz, Pclk = 66MHz', board 'SMDK2416 DDR2', and memory/storage 'DRAM: 64 MB', 'Flash: 1 MB', 'NAND: 256 MB'. It also shows serial port settings 'In: serial', 'Out: serial', 'Err: serial' and a MAC address 'smc911x: MAC 00:40:5c:26:0a:5b'. The prompt 'Hit any key to stop autoboot: 0' is shown, followed by the U-Boot prompt 'Helper2416 #' which appears twice.

第一件事，能够使UBoot加载NFS中的Kernel。

首先要了解的就是Kernel是怎么被使用起来的。对于我们现在使用UBoot来说，实际上也就是把Kernel文件加载到内存的某个位置，然后跳转过去运行。那么第一件事也就是把NFS的内核文件加载到内存里啦。

准备工作，把/helper2416/boot目录下的zImage.43复制为zImage文件，



接下来需要用到的命令如下：

1. `nfs c0008000 192.168.168.168:/helper2416/boot/zImage`

复制代码

这就是说，把后面NFS网络地址上的文件，加载到内存c0008000的位置上了。

```
fedora@localhost:~  
文件(F) 编辑(E) 查看(V) 终端(T) 帮助(H)  
Flash: 1 MB  
NAND: 256 MB  
In: serial  
Out: serial  
Err: serial  
smc911x: MAC 00: 40: 5c: 26: 0a: 5b  
Hit any key to stop autoboot: 0  
Helper2416 # nfs c0008000 192.168.168.168:/helper2416/boot/zImage  
smc911x: initializing  
smc911x: detected LAN9220 controller  
smc911x: phy initialized  
smc911x: MAC 00: 40: 5c: 26: 0a: 5b  
File transfer via NFS from server 192.168.168.168; our IP address is 192.1680  
Filename '/helper2416/boot/zImage'.  
Load address: 0xc0008000  
Loading: #####  
#####  
#####  
#####  
#####  
#####  
done  
Bytes transferred = 1923264 (0x1d58c0)  
Helper2416 #
```

别急，你别看我执行成功了，你执行可能就失败啦，如下信息——

```
1. Helper2416 # nfs c0008000 192.168.168.168:/helper2416/boot/test
```

```
smc911x: initializing  
smc911x: detected LAN9220 controller  
smc911x: phy initialized  
smc911x: MAC 00:40:5c:26:0a:5b  
File transfer via NFS from server 192.168.168.168; our IP address is 192.168.0.1  
Filename '/helper2416/boot/test'.  
Load address: 0xc0008000  
Loading: *  
ARP Retry count exceeded; starting again  
复制代码
```

为啥呢？因为在加载文件以前，我们还需要正确配置自己的开发板的IP地址。  
这就说到另外一个事了，设置参数，也可以认为自定义命令吧。

在UBoot中可以使用set命令或setenv命令设置环境参数和自定义的命令。

那么，废话少说，直接上有营养的，开发板上的UBoot已经设置好了一些参数，我们需要设置开发板的IP地址：

```
1. set ipaddr 192.168.168.100
```

```
set netmask 255.255.255.0
```

```
set serverip 192.168.168.168
```

复制代码

设置完成以后，可以通过print命令或者printenv命令来观察设定值是否正确。

```
netmask=255.255.255.0  
ipaddr=192.168.168.100  
serverip=192.168.168.168
```

设定OK，这次执行nfs命令吧，应该就成功了吧。

成功以后，执行如下命令，就是跳转到指定位置去运行啦：

```
1. bootm c0008000
```

复制代码

但是因为还没有配置文件系统从NFS加载，所以现在跳转还为时过早。

启动内核以前，一定要设置好内核参数。

在UBoot中，去设置bootargs就对啦

对于我们使用君益兴的板子来说，把bootargs设置成这样：

```
1. set bootargs console=ttySAC0,115200 root=/dev/nfs
```

```
nfsroot=192.168.168.168:/helper2416/rootfs,tcp,nolock,rsiz=1024,wsiz=1024 rw
```

```
ip=192.168.168.100 init=/linuxrc
```

复制代码

里面的重要内容一看就懂咯，控制台输出不用说，就那样设置，根文件系统利用nfs，加载自指定IP的指定路径，稍微有些读写参数，还要设定本机IP和init程序。

好了，有了以上的内容就足够了。所以我把ipaddr、netmask、serverip、bootargs按照上面设定好，然后再定义一条命令：

```
1. set bootcmd nfs c0008000 192.168.168.168:/helper2416/boot/zImage \; bootm  
c0008000
```

复制代码

这样，执行saveenv命令就能把前面的设置都保存到Flash中，以后就不需要每次都设置一遍了。

大功告成，启个动吧！

可是不幸的是，启动过程似乎卡住了。



```
fedora@localhost:~  
文件(F) 编辑(E) 查看(V) 终端(T) 帮助(H)  
mmc1: SDHCI controller on samsung-hsmmc [s3c-sdhci.1] using ADMA  
usbcore: registered new interface driver usbhid  
usbhid: USB HID core driver  
asoc: wm8731-hifi <-> s3c24xx-iis mapping ok  
ALSA device list:  
#0: wm8731  
TCP cubic registered  
NET: Registered protocol family 17  
can: controller area network core (rev 20090105 abi 8)  
NET: Registered protocol family 29  
can: raw protocol (rev 20090105)  
can: broadcast manager protocol (rev 20090105 t)  
Registering the dns_resolver key type  
s3c-rtc s3c2410-rtc: setting system clock to 2000-01-01 00:20:52 UTC (946686052)  
smc911x smc911x: eth0: link down  
smc911x smc911x: eth0: link up, 100Mbps, full-duplex, lpa 0xCDE1  
IP-Config: Guessing netmask 255.255.255.0  
IP-Config: Complete:  
    device=eth0, addr=192.168.168.100, mask=255.255.255.0, gw=255.255.255.255,  
    host=192.168.168.100, domain=, nis-domain=(none),  
    bootserver=255.255.255.255, rootserver=192.168.168.168, rootpath=  
VFS: Mounted root (nfs filesystem) on device 0:12.  
Freeing init memory: 272K
```

之后显示

```
1. [31/Dec/1999:16:21:10 +0000] boa: server version Boa/0.94.13
```

```
[31/Dec/1999:16:21:10 +0000] boa: server built Mar 26 2009 at 15:28:42.
```

```
[31/Dec/1999:16:21:10 +0000] boa: starting server pid=71, port 80
```

复制代码

过了好一阵子，最后显示

```
1. nfs: server 192.168.168.168 not responding, still trying
```

复制代码

好吧，只能一步一步查。不说废话了，以下是我发现的线索.....

首先在FC上ping开发板，一开始是通的，到上述LOG boa信息之后就ping不通了，看来是这里的问题.....

Boa据说是http服务器，所以观察文件系统中的/etc/init.d/rcS，在httpd启动之后，有两条脚本命令：

```
1. /sbin/ifconfig lo 127.0.0.1
```

```
/etc/init.d/ifconfig-eth0
```

复制代码

不用说，就是ifconfig搞的了，继续，检查这个ifconfig-eth0

检查这个文件，发现里面没什么不对劲的，当然它确实是在设置IP，会导致开发板和NFS的连接断开，但是看起来，从NFS启动的时候应该不会重新设置IP啊。

把脚本中的条件单独拿出来跑一下，原形毕露了，条件判断不正常！

好吧，在网上使劲搜一搜，发现问题根源了。

脚本中的判断

```
1. if grep -q "^/dev/root / nfs " /etc/mtab ; then
```

复制代码

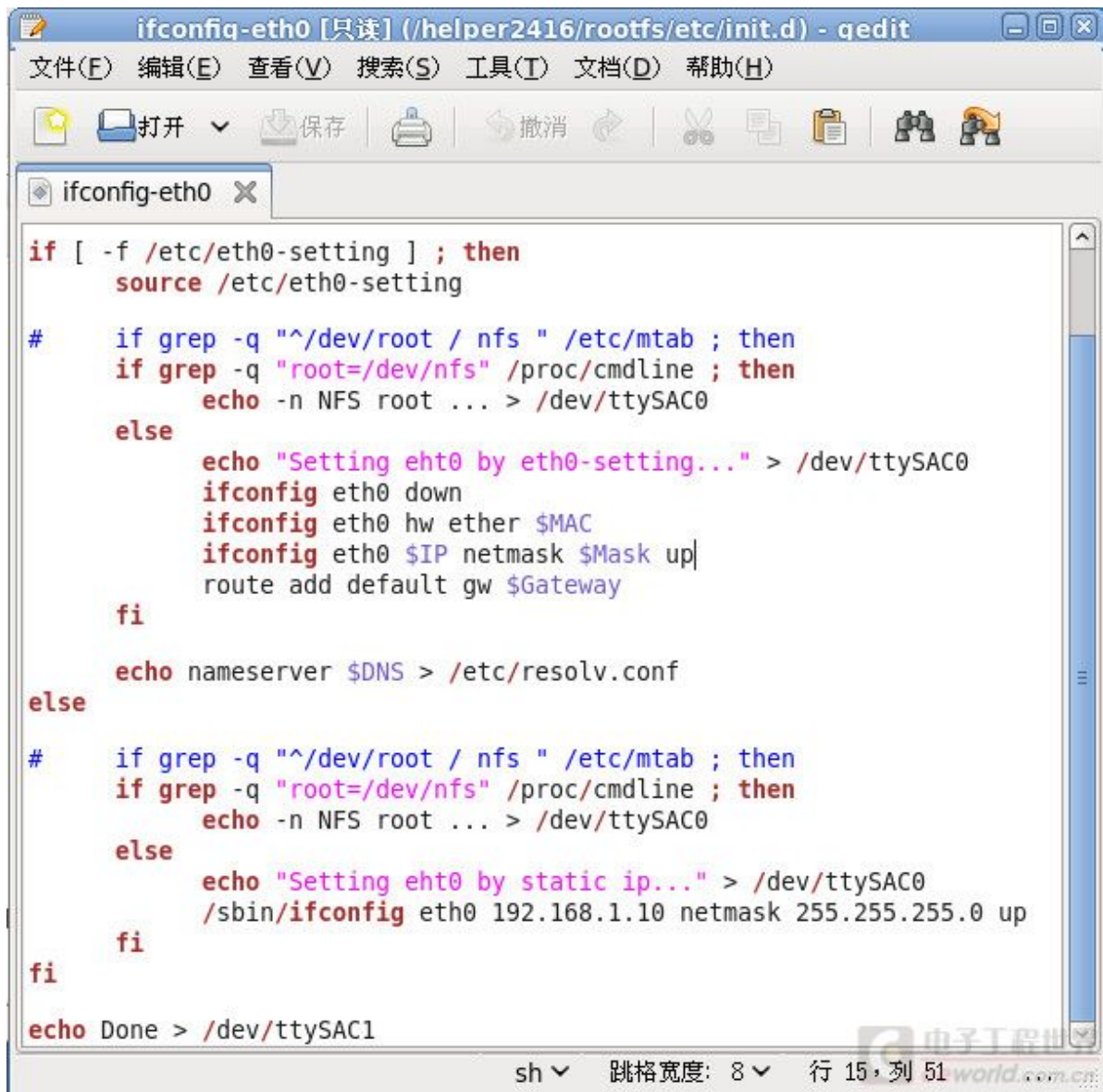
在我们使用的这个版本已经不好用了，需要修改这一行，变更为：

```
1. if grep -q "root=/dev/nfs" /proc/cmdline ; then
```

复制代码

从内核参数上来判断，应该是再准确不过了吧，试着改一下。





The screenshot shows a gedit editor window titled "ifconfig-eth0 [只读] (/helper2416/rootfs/etc/init.d) - gedit". The menu bar includes 文件(F), 编辑(E), 查看(V), 搜索(S), 工具(T), 文档(D), and 帮助(H). The toolbar contains icons for opening, saving, printing, undo, redo, cut, copy, paste, and zoom. The main text area displays the following script:

```
if [ -f /etc/eth0-setting ] ; then
    source /etc/eth0-setting
#
    if grep -q "^/dev/root / nfs " /etc/mtab ; then
        if grep -q "root=/dev/nfs" /proc/cmdline ; then
            echo -n NFS root ... > /dev/ttySAC0
        else
            echo "Setting eht0 by eth0-setting..." > /dev/ttySAC0
            ifconfig eth0 down
            ifconfig eth0 hw ether $MAC
            ifconfig eth0 $IP netmask $Mask up|
            route add default gw $Gateway
        fi
    else
        echo nameserver $DNS > /etc/resolv.conf
else
#
    if grep -q "^/dev/root / nfs " /etc/mtab ; then
        if grep -q "root=/dev/nfs" /proc/cmdline ; then
            echo -n NFS root ... > /dev/ttySAC0
        else
            echo "Setting eht0 by static ip..." > /dev/ttySAC0
            /sbin/ifconfig eth0 192.168.1.10 netmask 255.255.255.0 up
        fi
    fi
echo Done > /dev/ttySAC1
```

The status bar at the bottom shows "sh", "跳格宽度: 8", "行 15, 列 51", and a watermark for "world.com.cn".

再次启动，运行平稳，Qt界面也出现了。



```
fedora@localhost:~
文件(F) 编辑(E) 查看(V) 终端(T) 帮助(H)

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:4451 errors:3 dropped:0 overruns:0 frame:3
TX packets:5851 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:3715395 (3.5 MiB) TX bytes:898478 (877.4 KiB)
Interrupt:59 DMA chan:ff

lo
Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
UP LOOPBACK RUNNING MTU:16436 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

jyxtec login: root
[root@jyxtec /]1 ls
1?[=1?[          home          qtenv          usbdisk
4[4             lib           root           usr
bin            linuxrc        sbin           var
dev            mnt           sdcard         www
etc            opt           sys
g_file_storage. ko  proc         tmp
[root@jyxtec /]2
```