

Android-3G 出厂程序烧写手册 V1.0

北京博创兴盛科技有限公司

2011-07-18

目 录

出厂程序烧写内容如下:	3
烧写出厂程序的软硬件环境	3
1、 烧写 BOOTLOADER (NORFLASH 烧写步骤 1)	10
2、 烧写 BOOTLOADER (并口 JTAG 烧写步骤 2)	3
3、 烧写 BOOTLOADER (网口烧写步骤 3)	10
4、 烧写内核 KERNEL	20
5、 烧写 CRAMFS 文件系统	22
6、 烧写 ANDROID 文件系统	23
7、 触摸屏校正	34

Android 出厂程序烧写手册 V1.0

出厂程序烧写内容如下：

- ◆ 更新 BOOTLOADER 文件(普通用户无需更改)；
- ◆ 使用 tftp 软件烧写内核映像；
- ◆ 使用 tftp 软件烧写 cramfs 文件系统；
- ◆ 使用 U 盘烧写 Android 文件系统；

注：出厂烧写文件镜像及工具存放在光盘的 IMG 文件夹下，且 u-boot 一般用户无需更改，该烧写步骤可以略过。

烧写出厂程序的软硬件环境

- ◆ 软件：超级终端、TFTP32.EXE、SJF6410.exe
- ◆ 驱动：GIVEIO 驱动
- ◆ 硬件：ANDROID-3G 平台、12V 电源线、串口线、网线、JTAG 并口线

1、 烧写 BOOTLOADER (NORFLASH 烧写步骤 1)

☆ 连线：

将产品附带串口线一端连接到 PC 机端串口，另一端连接到 ANDROID-3G 开发板串口 0 (RS232-0 即开发板右侧起靠近电源的串口) 上。

将产品附带网线连接 PC 机与 ANDROID-3G 开发板上端网口 (使用 DM9000 网卡)。

☆ 跳线：

将 ANDROID-3G 核心板上跳线设置成 NORFLASH 烧写模式，如下：

OM4 OM3 OM2 OM1 :0 1 0 1

注：核心板跳线图以 PCB 底板图片为准，不要按照编码开关器件自带编号设置 (即跳线表以 “OM4 OM3 OM2 OM1” 编号为准，不要按照 “4 3 2 1” 设置。核心板外侧为 0，内侧为 1)。
默认出厂已经跳到 NANDFLASH 模式。

(1) 利用 WindowsXP 系统自带串口程序“超级终端”连接串口, 监视控制开发板信息, 或使用光盘 TOOLS 目录下附带的 Xmanager 软件连接串口皆可。

以下以使用 Xmanager 软件安装后的 X-SHELL 工具为例, 通过该软件登陆 ANDROID-3G 串口终端。打开 X-SHELL 软件, 新建串口终端, Method 选择 SERIAL 方式, 在 Setup 中配置串口为串口 0 (根据具体硬件决定), 波特率 115200, 等如图:

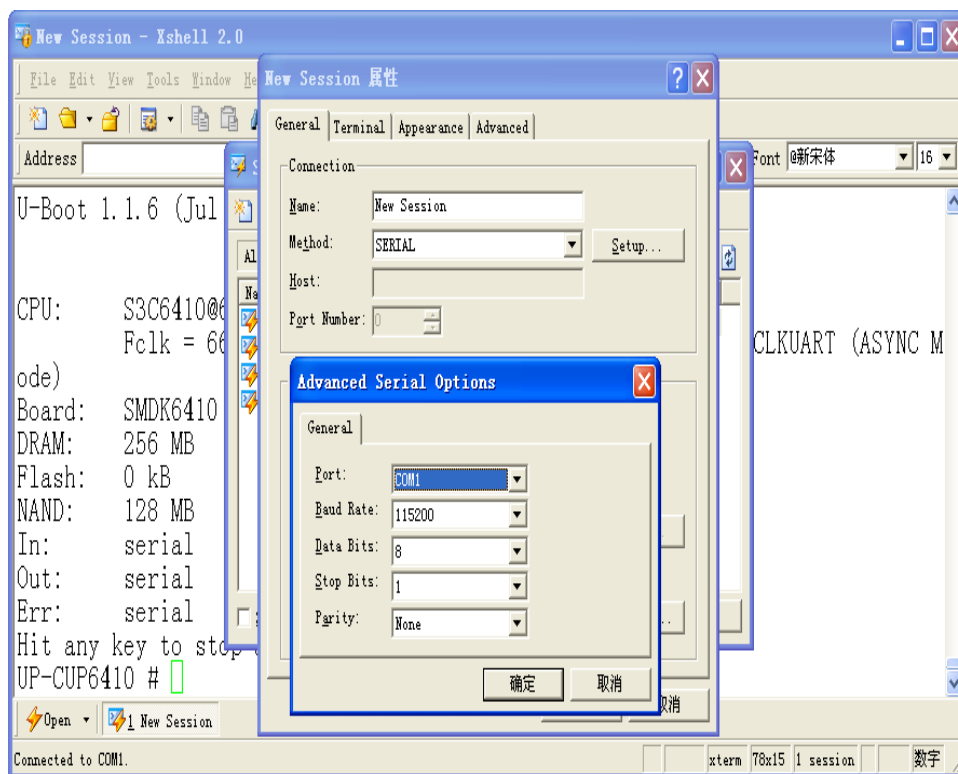


图 1

(2) 确定连接后, 连接电源, 插好网线。按下 ANDROID-3G 开发板右下角 POWER 电源键, 系统上电。X-SHELL 终端进入 ANDROID-3G 开发板的 U-BOOT 功能界面, 按下回车, 进入 U-BOOT 界面如图 2:

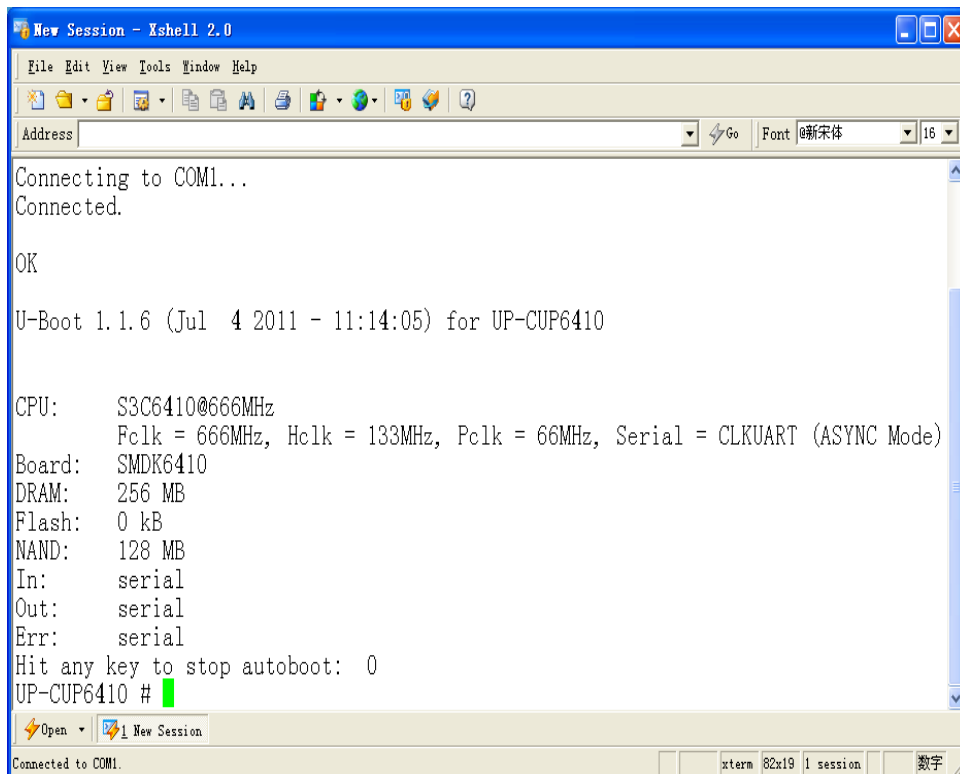


图 2

(3) 更新 U-BOOT, 在 PC 机端打开 TFTP32.exe 软件(在光盘/IMG/目录下, 注意 TFTP32 搜索目录, 如在 IMG 目录下打开则无需特殊设置)如图 3、图 4:

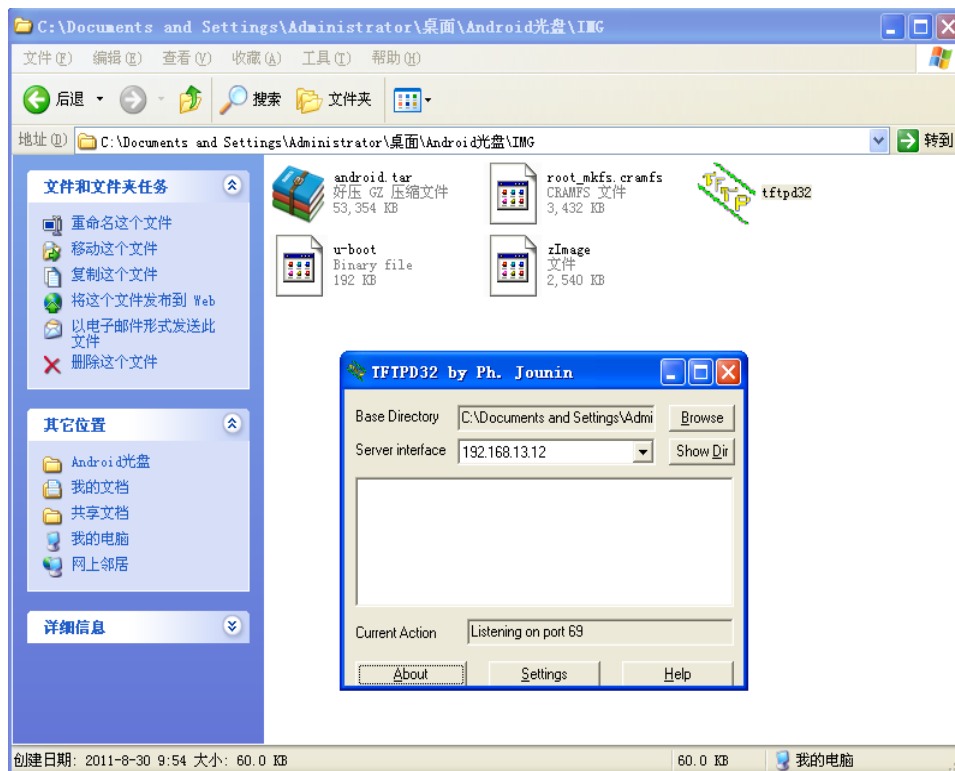


图 3

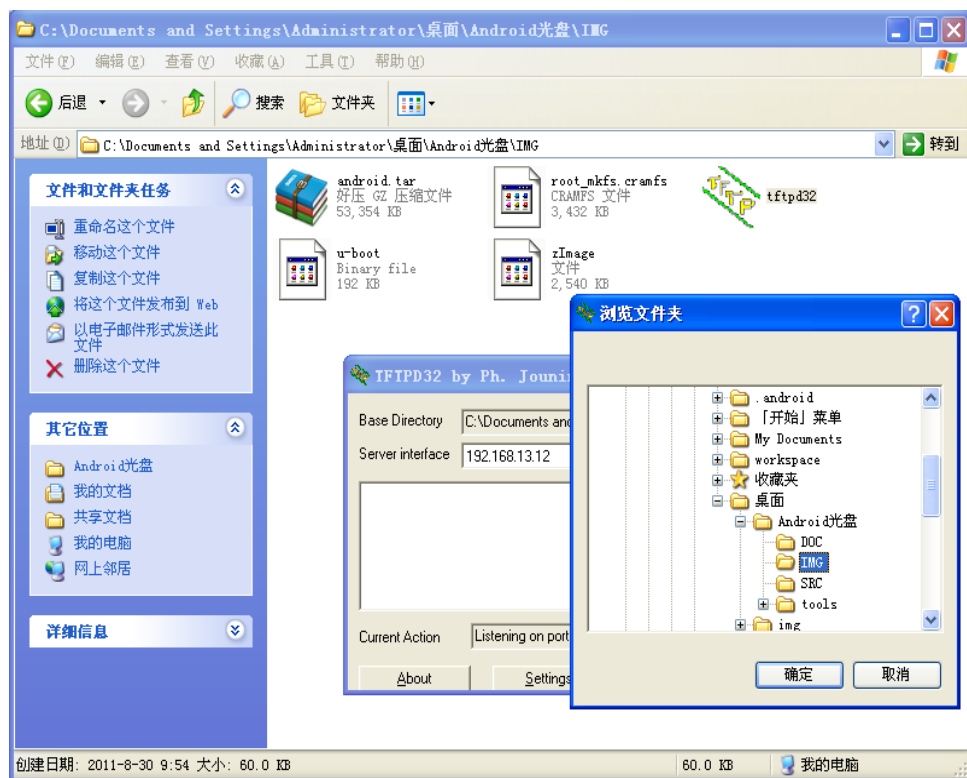


图 4

(4) 在 U-BOOT 终端设置网络 IP:

◆ 首先执行擦除 NANDFLASH 命令: `nand scrub`

注意: `nand scrub` 一定要首先执行, 否则在 u-boot 中无法保存网络设置参数, 并且 Linux 启动的过程中会出现 Nand Flash 的 ECC 校验错误

直接输入 'y' 确认, 再按回车键, 如图 5

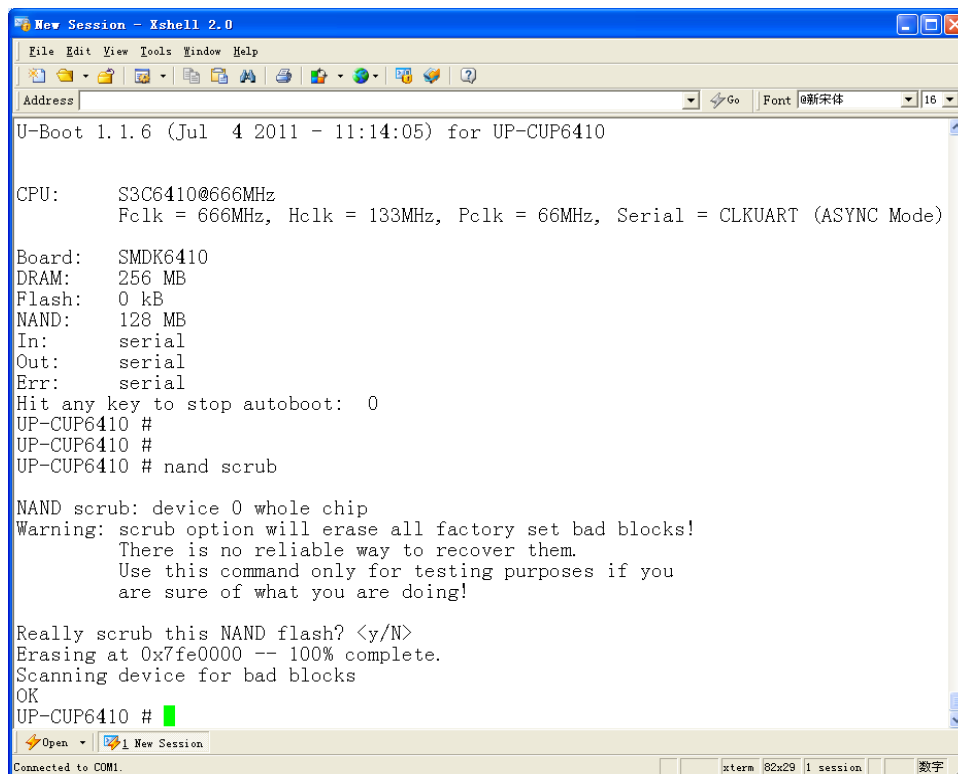


图 5

- ◆ 设置主机 IP 地址：setenv serverip 192.168.13.12

serverip 为运行 TFTP32 软件系统 IP 地址，通常为 WindowsXP 系统 IP。

- ◆ 设置目标板 IP 地址：setenv ipaddr 192.168.13.15

ipaddr 为 ARM 设备 IP 地址(此地址只在 U-BOOT 中有效)。

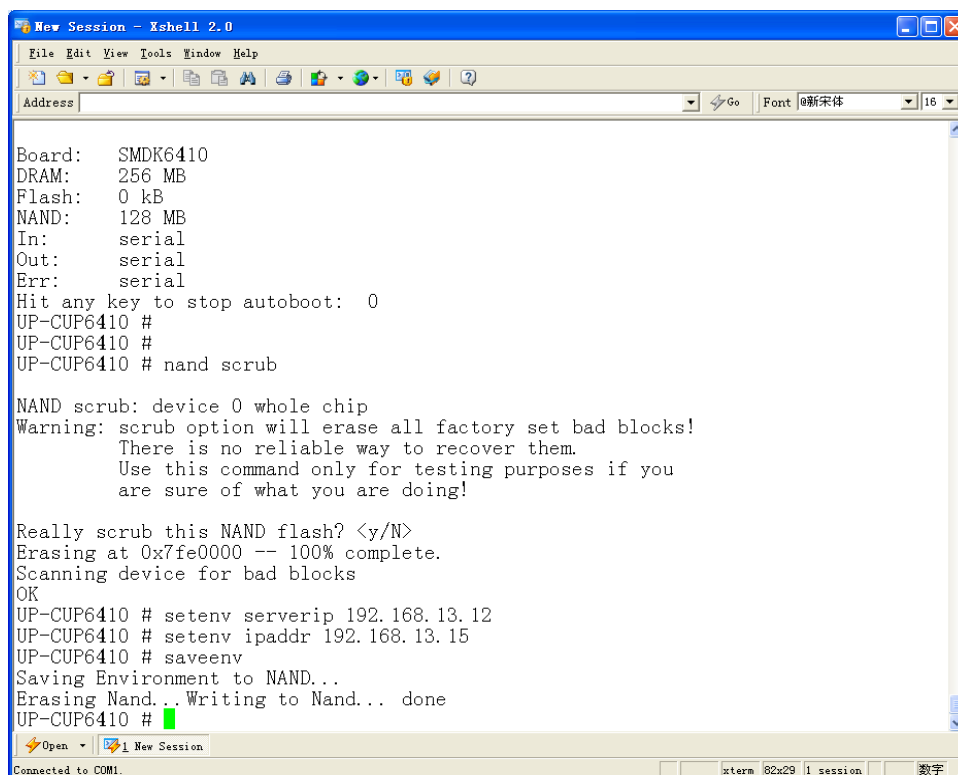


图 6

开始下载 u-boot.bin:

```
# tftp c0008000 u-boot.bin
```

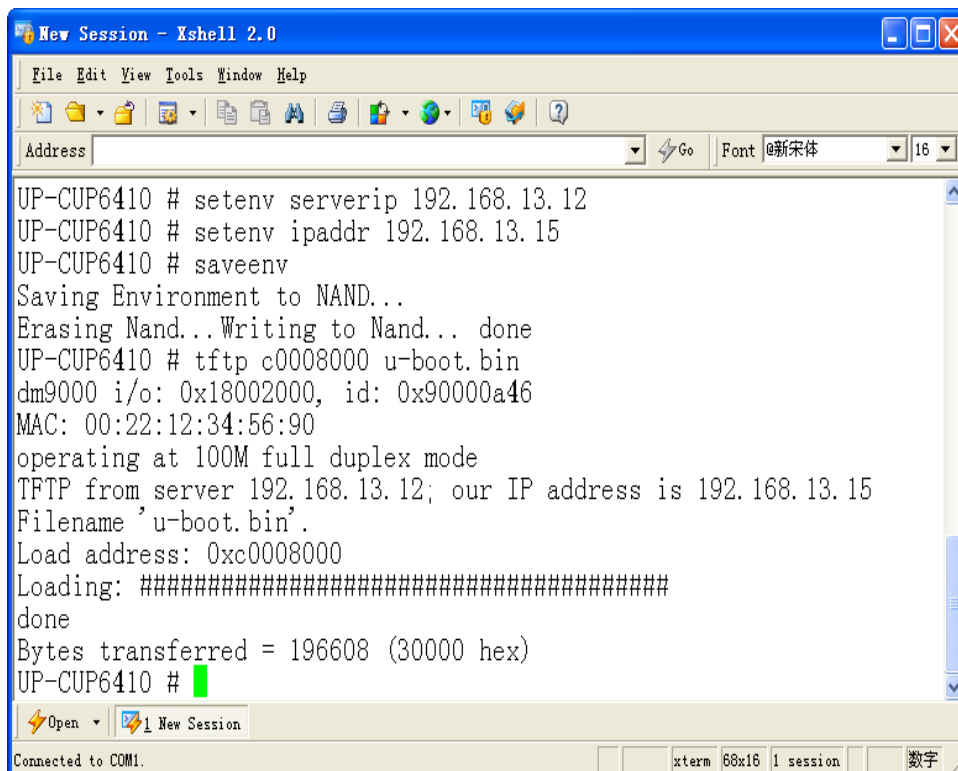
将 u-boot.bin 文件通过 TFTP32 软件的网络功能下载到 ANDROID-3G 的 SDRAM 中，地址为 c0008000。

```
# nand erase 0 40000
```

擦除 NANDFLASH 上 0 地址开始大小为 0x40000 的空间。

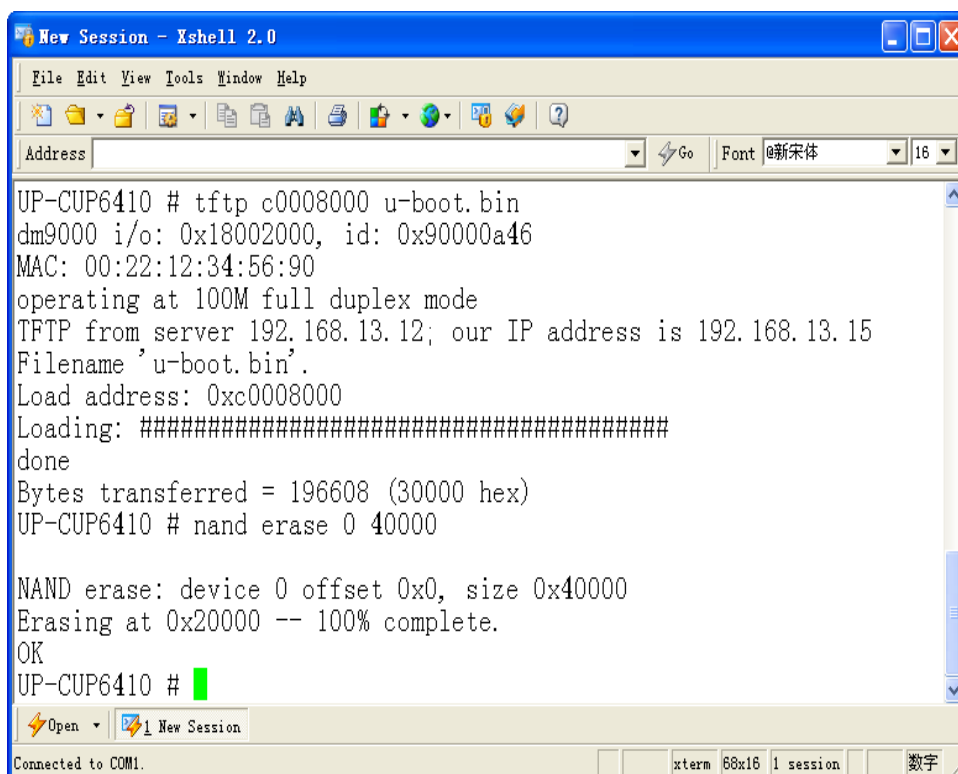
```
# nand write c0008000 0 40000
```

向 NANDFLASH 写入从 SDRAM 上 c0008000 地址处的文件，写入到 NANDFLASH 上 0 地址开始处 0x40000 大小的内容。如图 8、图 9:



```
New Session - Xshell 2.0
File Edit View Tools Window Help
Address [ ] Go Font @新宋体 16
UP-CUP6410 # setenv serverip 192.168.13.12
UP-CUP6410 # setenv ipaddr 192.168.13.15
UP-CUP6410 # saveenv
Saving Environment to NAND...
Erasing Nand...Writing to Nand... done
UP-CUP6410 # tftp c0008000 u-boot.bin
dm9000 i/o: 0x18002000, id: 0x90000a46
MAC: 00:22:12:34:56:90
operating at 100M full duplex mode
TFTP from server 192.168.13.12; our IP address is 192.168.13.15
Filename 'u-boot.bin'.
Load address: 0xc0008000
Loading: #####
done
Bytes transferred = 196608 (30000 hex)
UP-CUP6410 #
```

图 7



```
New Session - Xshell 2.0
File Edit View Tools Window Help
Address [ ] Go Font @新宋体 16
UP-CUP6410 # tftp c0008000 u-boot.bin
dm9000 i/o: 0x18002000, id: 0x90000a46
MAC: 00:22:12:34:56:90
operating at 100M full duplex mode
TFTP from server 192.168.13.12; our IP address is 192.168.13.15
Filename 'u-boot.bin'.
Load address: 0xc0008000
Loading: #####
done
Bytes transferred = 196608 (30000 hex)
UP-CUP6410 # nand erase 0 40000

NAND erase: device 0 offset 0x0, size 0x40000
Erasing at 0x20000 -- 100% complete.
OK
UP-CUP6410 #
```

图 8

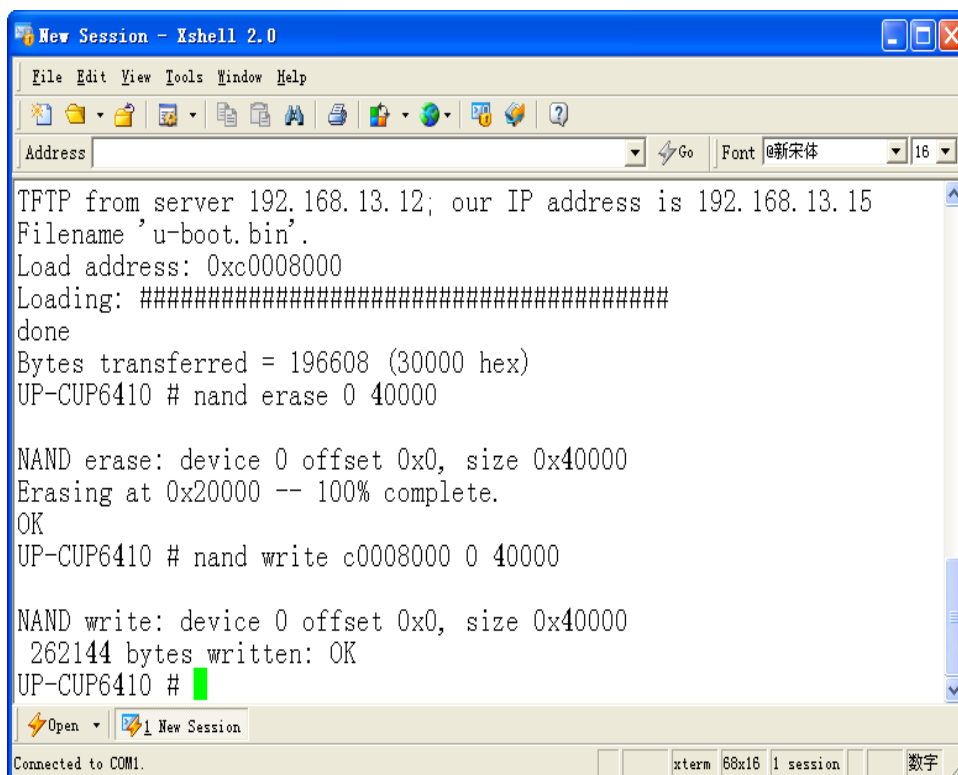


图 9

备注：此方法通过 NORFLASH 启动 u-boot 烧写 NANDFLASH u-boot, 仍需按照下面步骤 3 方法，通过网线重新烧写 u-boot 文件，方可正常启动系统。

2、 烧写 BOOTLOADER (并口 JTAG 烧写步骤 2) (选用)

☆ 连线：

将产品附带并口线一端连接到 PC 机上并口端，另一端并口连接到 UP-JTAG 端。将 UP-JTAG 另一端 20P 连接到 ANDROID-3G 面板背面 SD 卡插槽一侧的 14P 插槽当中。

☆ 跳线：

将 ANDROID-3G 核心板上跳线设置成 NANDFLASH 烧写模式，如下：

OM4 OM3 OM2 OM1 :0 0 1 0

备注：跳线模式出厂已经默认跳到 NANDFLASH 模式，用户确认后，无需更改。

注：此种方法适合在开发板为裸机情况下初次烧写 BOOTLOADER 文件内容到 NANDFLASH

☆ 安装 giveio 并口驱动：

Android-3G 出厂程序烧写手册

在使用 PC 机并口烧写内容前，首先要安装并口 GIVEIO 驱动，这样 JTAG 才能正确识别开发板硬件。

- 1) 把并口线插到 PC 机的并口，并把并口与 Android 开发板 JTAG 相连。
- 2) 把整个 GIVEIO 目录(在光盘/工具软件/flashvivi 目录下)拷贝到 C:\WINDOWS 下，并把该目录下的 giveio.sys 文件拷贝到 c:/windows/system32/drivers 下。
- 3) 在控制面板里，选添加硬件>下一步>选一是我已经连接了此硬件>下一步>选中一添加新的硬件设备>下一步>选中安装我手动从列表选择的硬件>下一步>选择一显示所有设备>选择一从磁盘安装-浏览，指定驱动为 C:\WINDOWS\GIVEIO\giveio.inf 文件，点击确定，安装好驱动。

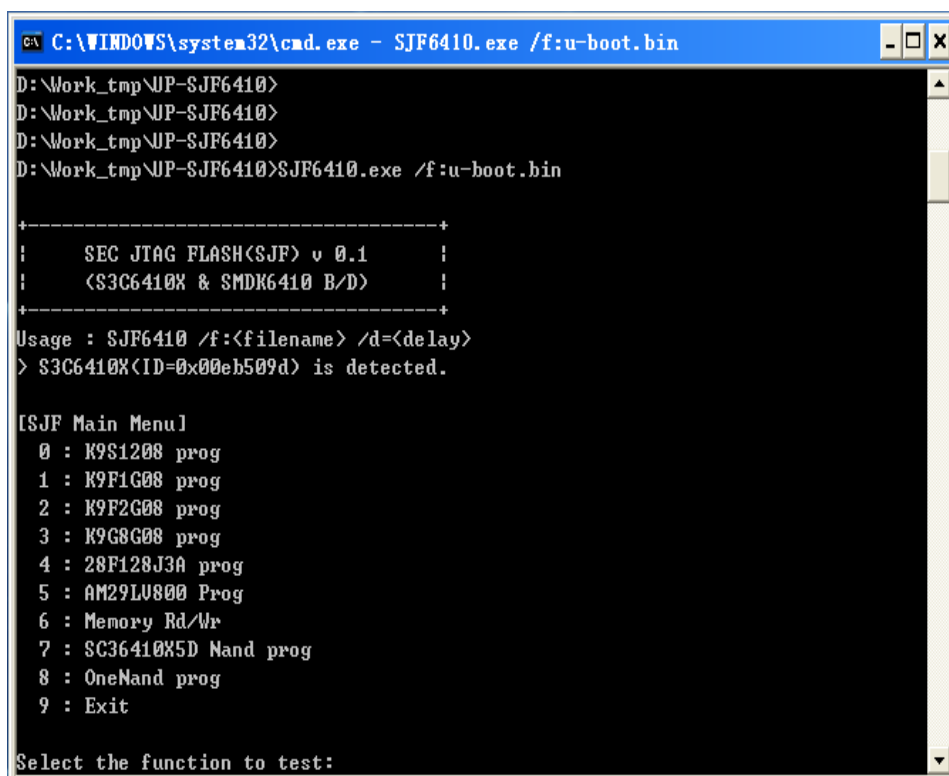
☆ 烧写：

连接电源，按下 ANDROID-3G 开发板右下角 POWER 电源键，系统上电。

进入 PC 机端 MS-DOS 命令行，使用 SJF6410.EXE 软件烧写 u-boot.bin 文件。

例如笔者将 SJF6410.EXE 及 u-boot.bin 等烧写文件放在 PC 机的 D:\work_tmp\UP-SJF6410\文件夹下。

- (1) 使用 SJF6410.exe /f:u-boot.bin 命令格式烧写 u-boot.bin 文件如下图所示：



```

C:\WINDOWS\system32\cmd.exe - SJF6410.exe /f:u-boot.bin
D:\Work_tmp\UP-SJF6410>
D:\Work_tmp\UP-SJF6410>
D:\Work_tmp\UP-SJF6410>
D:\Work_tmp\UP-SJF6410>SJF6410.exe /f:u-boot.bin

+-----+
|      SEC JTAG FLASH(SJF) v 0.1      |
|      (S3C6410X & SMDK6410 B/D)      |
+-----+

Usage : SJF6410 /f:<filename> /d=<delay>
> S3C6410X(ID=0x00eb509d) is detected.

[ SJF Main Menu ]
0 : K9S1208 prog
1 : K9F1G08 prog
2 : K9F2G08 prog
3 : K9G8G08 prog
4 : 28F128J3A prog
5 : AM29LV800 Prog
6 : Memory Rd/Wr
7 : SC36410X5D Nand prog
8 : OneNand prog
9 : Exit

Select the function to test:
  
```

上图显示正确检测到 S3C6410X CPU。

- (2) 选择 ‘1’ K9F1G08 烧写 NANDFLASH，如图所示：

```
C:\WINDOWS\system32\cmd.exe - SJF6410.exe /f:u-boot.bin

D:\Work_tmp\JP-SJF6410>
D:\Work_tmp\JP-SJF6410>
D:\Work_tmp\JP-SJF6410>
D:\Work_tmp\JP-SJF6410>SJF6410.exe /f:u-boot.bin

+-----+
|      SEC JTAG FLASH(SJF) v 0.1      |
|      (S3C6410X & SMDK6410 B/D)      |
+-----+

Usage : SJF6410 /f:<filename> /d=<delay>
> S3C6410X(ID=0x00eh509d) is detected.

[ SJF Main Menu ]
0 : K9S1208 prog
1 : K9F1G08 prog
2 : K9F2G08 prog
3 : K9G8G08 prog
4 : 28F128J3A prog
5 : AM29LV800 Prog
6 : Memory Rd/Wr
7 : SC36410X5D Nand prog
8 : OneNand prog
9 : Exit

Select the function to test:1
```

(3) 输入 ‘0’ ,FLASH 编程。如图所示:

```
C:\WINDOWS\system32\cmd.exe - SJF6410.exe /f:u-boot.bin

Usage : SJF6410 /f:<filename> /d=<delay>
> S3C6410X(ID=0x00eh509d) is detected.

[ SJF Main Menu ]
0 : K9S1208 prog
1 : K9F1G08 prog
2 : K9F2G08 prog
3 : K9G8G08 prog
4 : 28F128J3A prog
5 : AM29LV800 Prog
6 : Memory Rd/Wr
7 : SC36410X5D Nand prog
8 : OneNand prog
9 : Exit

Select the function to test:1

[K9F1G08 NAND Flash JTAG Programmer]
K9F1G08U is detected. ID=0xecf1

0 : K9F1G08 Program
1 : K9F1G08 Print blkPage
2 : Exit

Select the function to test : 0
```

(4) 输入 ‘0’ ,开始向 NANDFLASH 中烧写程序如图所示:

```
C:\WINDOWS\system32\cmd.exe - SJF6410.exe /f:u-boot.bin

3 : K9G8G08 prog
4 : 28F128J3A prog
5 : AM29LV800 Prog
6 : Memory Rd/Wr
7 : SC36410X5D Nand prog
8 : OneNand prog
9 : Exit

Select the function to test:1

[K9F1G08 NAND Flash JTAG Programmer]
K9F1G08U is detected. ID=0xecf1

0 : K9F1G08 Program
1 : K9F1G08 Print blkPage
2 : Exit

Select the function to test : 0

[K9F1G08 NAND Flash Writing Program]

Source size:0h~33ffffh

Available target block number : 0~4096
Input target block number : 0
```

(5) 此过程烧写时间较长，PC 机最好不要切换到其他程序界面，大约 20 分钟左右烧写完毕。输入 ‘2’ 退出。如图所示：

```
C:\WINDOWS\system32\cmd.exe

0 : K9F1G08 Program
1 : K9F1G08 Print blkPage
2 : Exit

Select the function to test : 0

[K9F1G08 NAND Flash Writing Program]

Source size:0h~33ffffh

Available target block number : 0~4096
Input target block number : 0
blockIndex = 0
..block #0 erase done
page programing : #0 ~ #104
page program done

0 : K9F1G08 Program
1 : K9F1G08 Print blkPage
2 : Exit

Select the function to test : 2

D:\Work_tmp\JP-SJF6410>
```

如上所示，并口烧写 u-boot 完毕， ANDROID-3G 开发板断电，拔出并口线。

备注：此时仍需按照下面步骤 3 方法，通过网线重新烧写 u-boot 文件，方可正常启动系统。

3、 烧写 BOOTLOADER (网口烧写步骤 3)

☆ 连线:

将产品附带串口线一端连接到 PC 机端串口, 另一端连接到 ANDROID-3G 开发板串口 0 (RS232-0 即开发板右侧起靠近电源的串口) 上。

将产品附带网线连接 PC 机与 ANDROID-3G 开发板上端网口 (使用 DM9000 网卡)。

☆ 跳线:

将 ANDROID-3G 核心板上跳线设置成 NANDFLASH 烧写模式, 如下:

OM4 OM3 OM2 OM1 :0 0 1 0

注: 核心板跳线图以 PCB 底板图片为准, 不要按照编码开关器件自带编号设置 (即跳线表以 “OM4 OM3 OM2 OM1” 编号为准, 不要按照 “4 3 2 1” 设置。核心板外侧为 0, 内侧为 1)。

默认出厂已经跳到 NANDFLASH 模式。

(1) 利用 WindowsXP 系统自带串口程序“超级终端”连接串口, 监视控制开发板信息, 或使用光盘 TOOLS 目录下附带的 Xmanager 软件连接串口皆可。

以下以使用 Xmanager 软件安装后的 X-SHELL 工具为例, 通过该软件登陆 ANDROID-3G 串口终端。打开 X-SHELL 软件, 新建串口终端, Method 选择 SERIAL 方式, 在 Setup 中配置串口为串口 0 (根据具体硬件决定), 波特率 115200, 等如图:

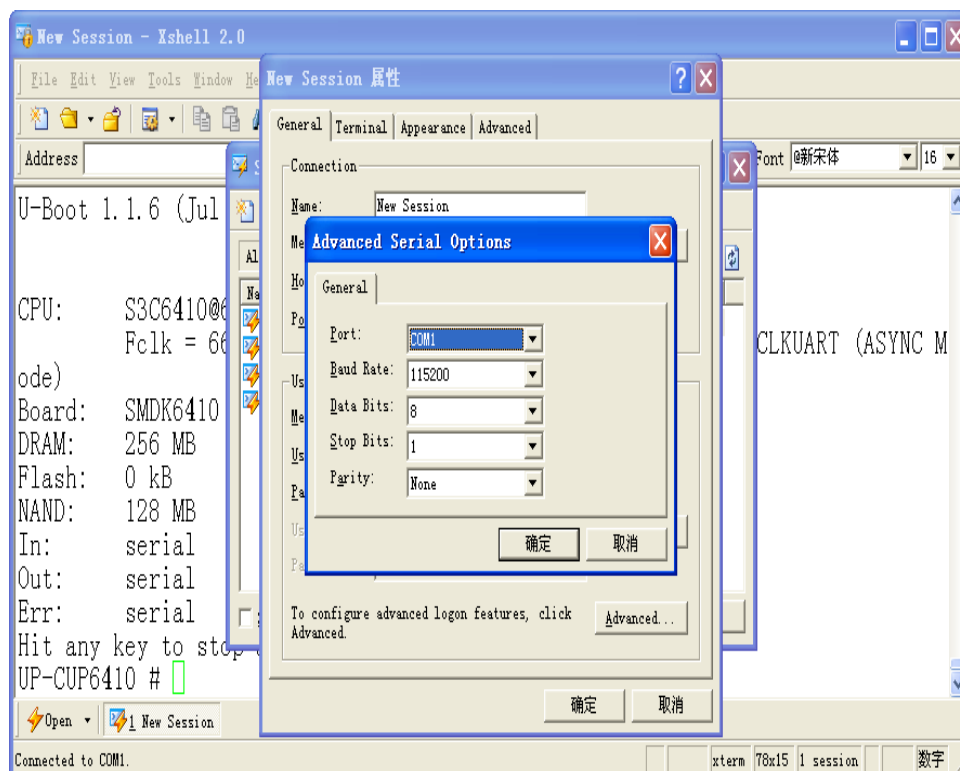


图 9

(2) 确定连接后, 连接电源, 插好网线。按下 ANDROID-3G 开发板右下角 POWER 电源键, 系统上电。

X-SHELL 终端进入 ANDROID-3G 开发板的 U-BOOT 功能界面, 按下回车, 进入 U-BOOT 界面, 如图:

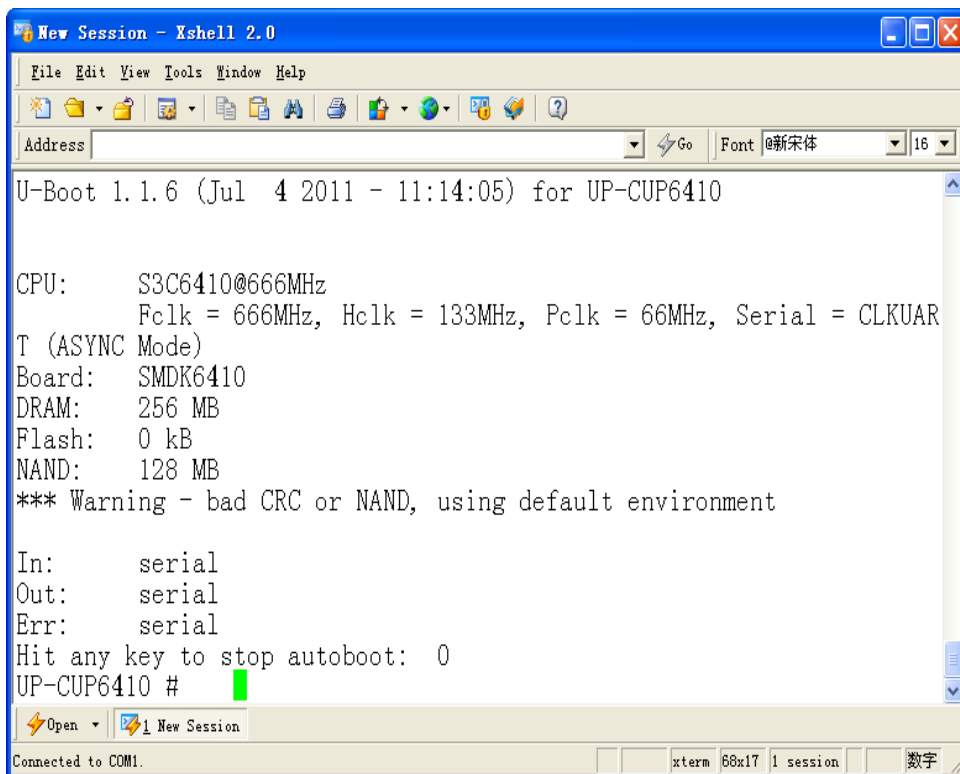


图 10

(3) 更新 U-BOOT, 在 PC 机端打开 TFTP32.exe 软件(在光盘 IMG/目录下, 注意 TFTP32 搜索目录, 如在 IMG 目录下打开则无需特殊设置)如图 11 和图 12:

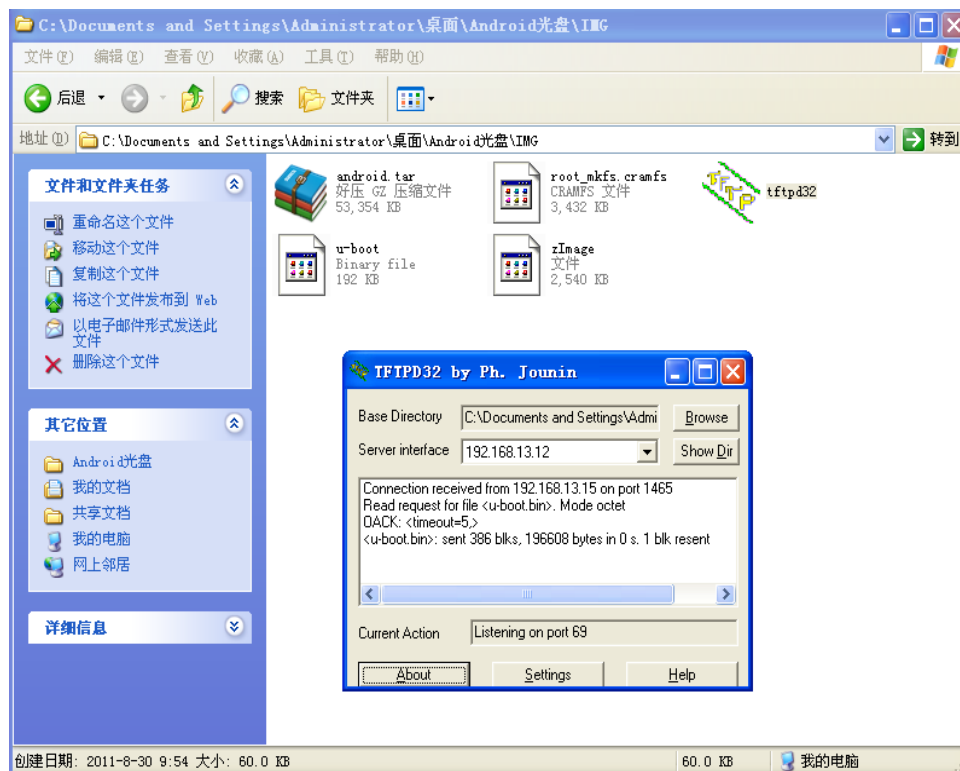


图 11

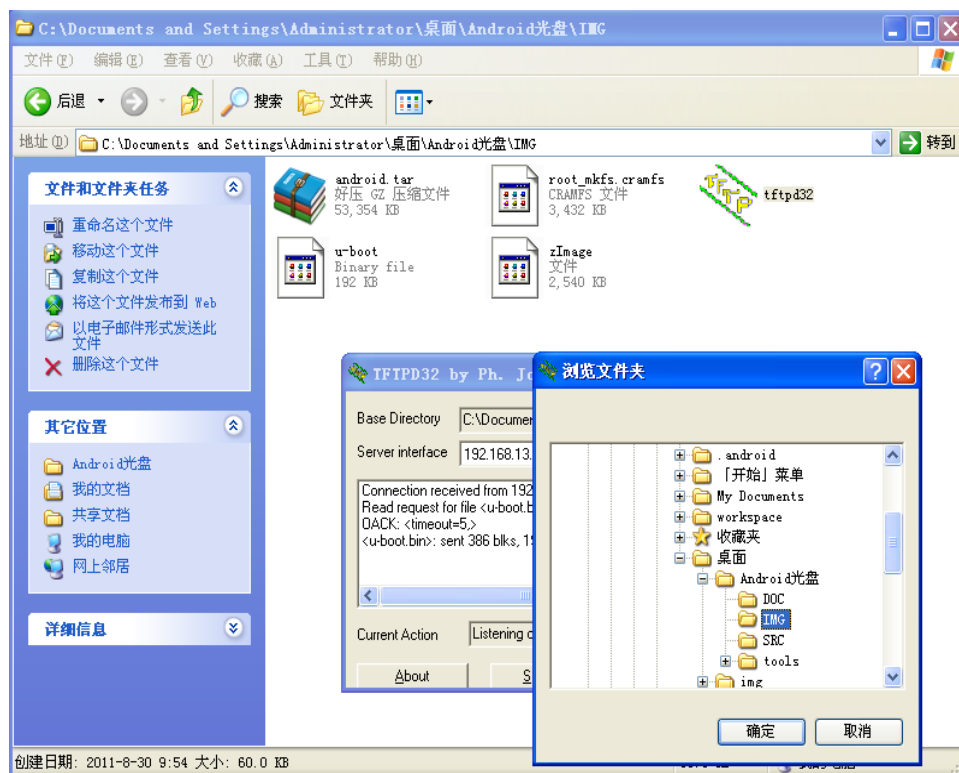


图 12

(4) 在 U-BOOT 终端设置网络 IP:

◆ 首先执行擦除 NANDFLASH 命令: `nand scrub`

注意: `nand scrub` 一定要首先执行, 否则在 u-boot 中无法保存网络设置参数, 并且 Linux 启动的过程中会出现 Nand Flash 的 ECC 校验错误。

直接输入 'y' 确认, 再按回车键, 如图 13

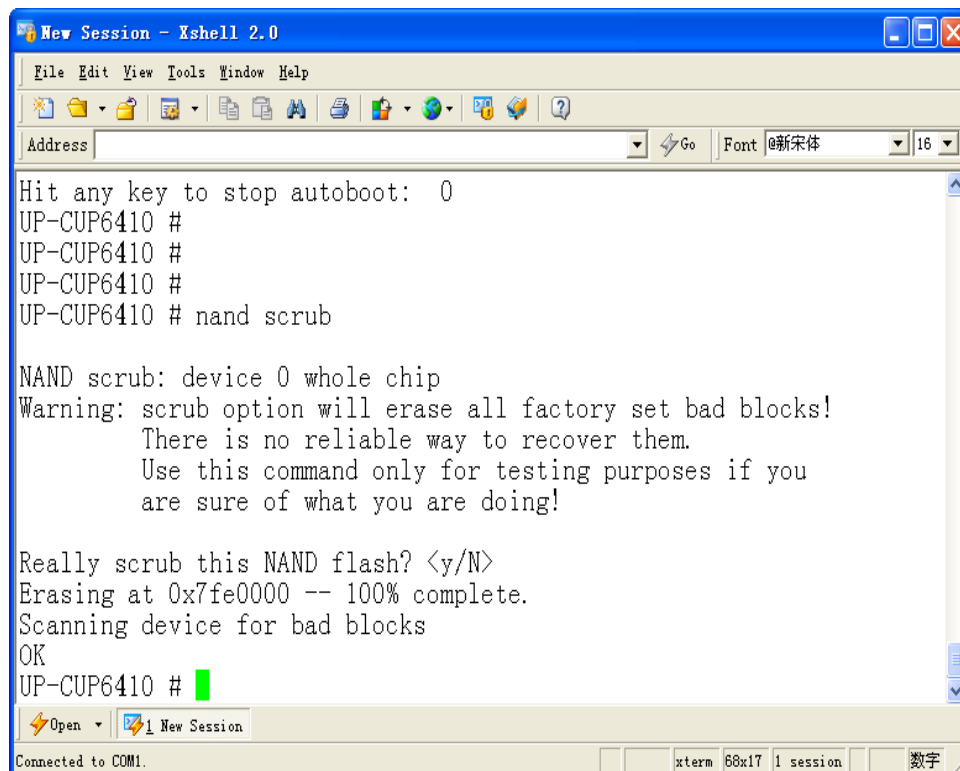


图 13

备注：全部擦出 NANDFLASH 之后，此时 ANDROID-3G 系统务必不要掉电或重启，否则 uboot 会消失，仍需重新使用 JTAG 烧写或 NORFLASH 烧写。

执行 nand scrub 命令后接着设置网络 IP 地址：

- ◆ 设置主机 IP 地址：setenv serverip 192.168.13.12
serverip 为运行 TFTP32 软件系统 IP 地址，通常为 WindowsXP 系统 IP。
- ◆ 设置目标板 IP 地址：setenv ipaddr 192.168.13.15
ipaddr 为 ARM 设备 IP 地址(此地址只在 U-BOOT 中有效)。
- ◆ 保存 IP 地址：saveenv
saveenv 保存设置后，系统重启 UBOOT 中设置的网络 IP 仍保留。

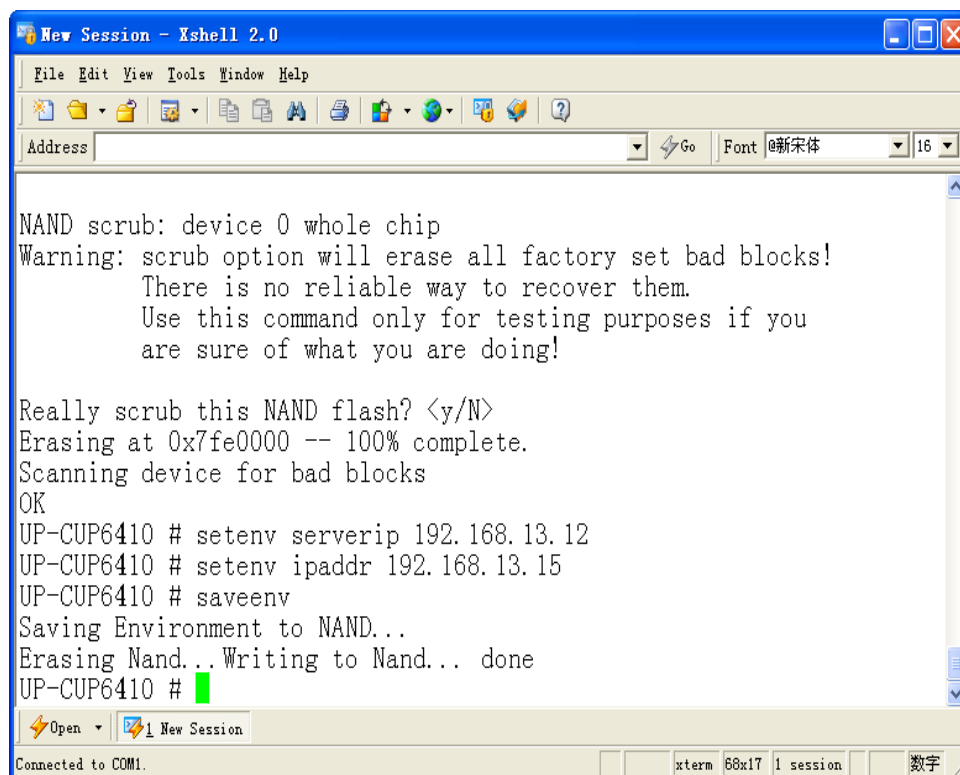


图 14

※ 使用 `nand scrub` 命令请谨慎！该命令会将系统 NANDFLASH 全部格式化，包括出厂烧写进去的 UBOOT 也将擦除掉，因此执行完该命令后，系统千万不要掉电，利用 RAM 中的 UBOOT 将 `uboot.bin` 重新烧写进 NANDFLASH。

开始下载 `u-boot.bin`：

```
# tftp c0008000 u-boot.bin
```

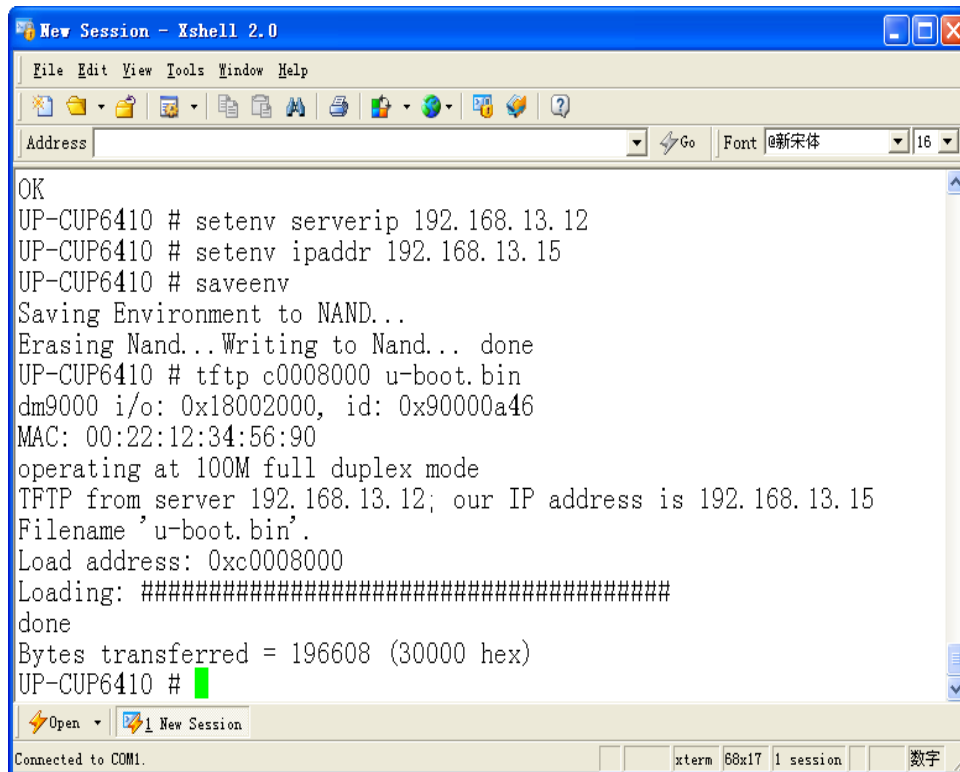
将 `u-boot.bin` 文件通过 TFTP32 软件的网络功能下载到 ANDROID-3G 的 SDRAM 中，地址为 `c0008000`。

```
# nand erase 0 40000
```

擦除 NANDFLASH 上 0 地址开始大小为 `0x40000` 的空间。

```
# nand write c0008000 0 40000
```

向 NANDFLASH 写入从 SDRAM 上 `c0008000` 地址处的文件，写入到 NANDFLASH 上 0 地址开始处 `0x40000` 大小的内容。如图 15 和图 16：

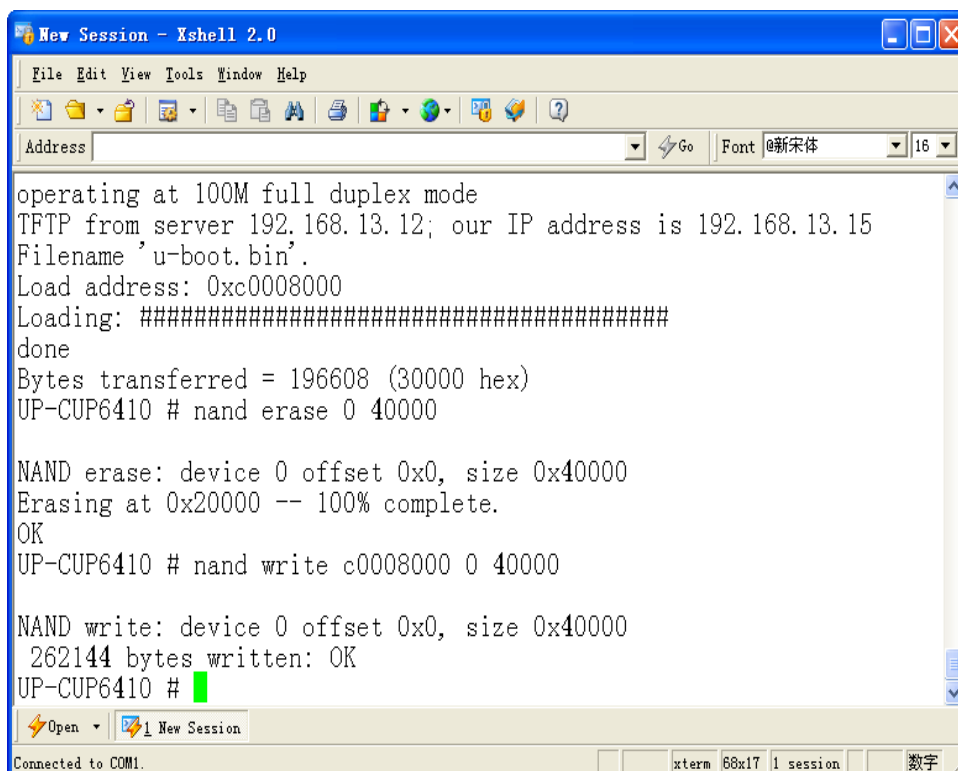


```

New Session - Xshell 2.0
File Edit View Tools Window Help
Address
OK
UP-CUP6410 # setenv serverip 192.168.13.12
UP-CUP6410 # setenv ipaddr 192.168.13.15
UP-CUP6410 # saveenv
Saving Environment to NAND...
Erasing Nand...Writing to Nand... done
UP-CUP6410 # tftp c0008000 u-boot.bin
dm9000 i/o: 0x18002000, id: 0x90000a46
MAC: 00:22:12:34:56:90
operating at 100M full duplex mode
TFTP from server 192.168.13.12; our IP address is 192.168.13.15
Filename 'u-boot.bin'.
Load address: 0xc0008000
Loading: #####
done
Bytes transferred = 196608 (30000 hex)
UP-CUP6410 #

```

图 15



```

New Session - Xshell 2.0
File Edit View Tools Window Help
Address
operating at 100M full duplex mode
TFTP from server 192.168.13.12; our IP address is 192.168.13.15
Filename 'u-boot.bin'.
Load address: 0xc0008000
Loading: #####
done
Bytes transferred = 196608 (30000 hex)
UP-CUP6410 # nand erase 0 40000

NAND erase: device 0 offset 0x0, size 0x40000
Erasing at 0x20000 -- 100% complete.
OK
UP-CUP6410 # nand write c0008000 0 40000

NAND write: device 0 offset 0x0, size 0x40000
262144 bytes written: OK
UP-CUP6410 #

```

图 16

配置 u-boot:

```
#setenv bootcmd nand read c0008000 40000 3c0000\;bootm c0008000
```

U-BOOT 更新完毕！

4、 烧写内核 Kernel

☆ 连线：

将产品附带串口线一端连接到 PC 机端串口，另一端连接到 ANDROID-3G 开发板串口 0 (RS232-0 即开发板右侧起靠近电源的串口) 上。

将产品附带网线连接 PC 机与 ANDROID-3G 开发板上端网口 (使用 DM9000 网卡)。

☆ 跳线：

将 ANDROID-3G 核心板上跳线设置成 NANDFLASH 烧写模式，默认出厂已经设置好。如下：

```
# OM4 OM3 OM2 OM1 :0 0 1 0
```

☆ 烧写：

经过上述烧写过程后，我们就可以直接从 ANDROID-3G 的 NANDFLASH 中启动 U-BOOT，并利用 U-BOOT 的 TFTP 网络功能快速下载系统的内核与文件系统等内容了。

连接电源，插好网线。按下 ANDROID-3G 开发板右上角 POWER 电源键，系统上电。

在 PC 机端打开 TFTP32 软件软件，注意该软件搜索目录为产品光盘的 IMG 目录即与要烧写的文件目录一致，并打开串口终端软件如超级终端、Xshell 、或是 DNW 进入 ANDROID-3G 的 U-BOOT 串口终端，迅速按下回车键后，进入 u-boot 界面。

配置 IP 地址后，执行以下命令，如图所示：

```
# tftp c0008000 zImage
```

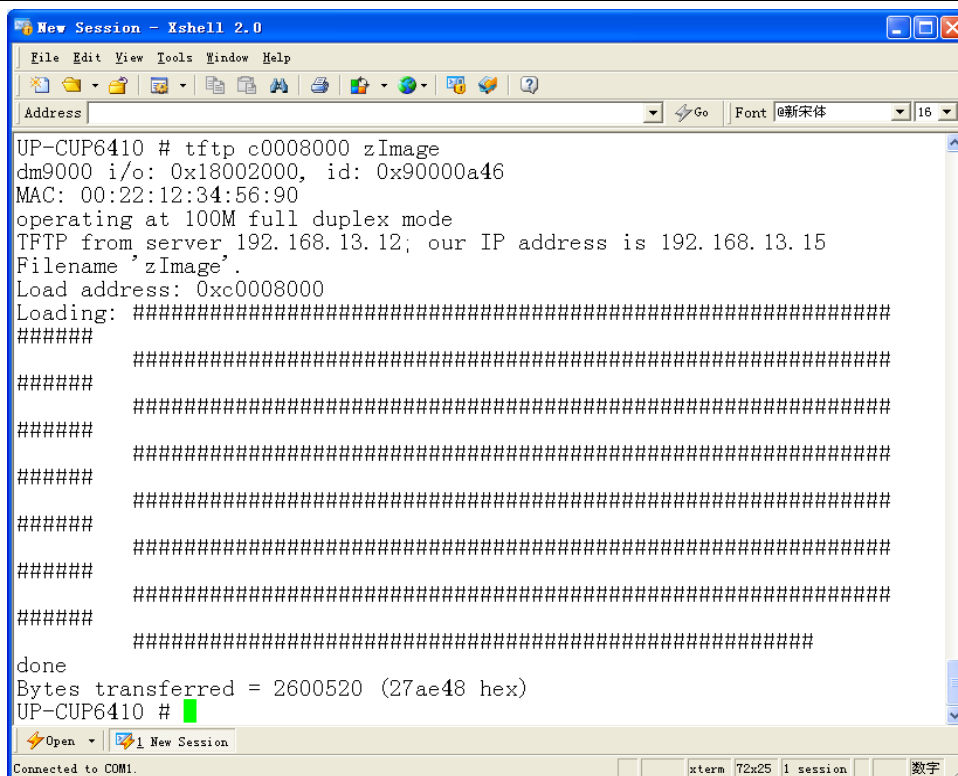
利用 TFTP32 软件将内核文件 zImage 烧写如 SDRAM 地址为 c0008000

```
# nand erase 40000 3c0000
```

将 NANDFLASH 起始地址为 0x40000 开始处大小为 0x3c0000 的空间擦除

```
# nand write c0008000 40000 3c0000
```

从 SDRAM 的 0xc0000000 地址处，向 NANDFLASH 起始地址为 0x40000 写入大小为 0x3c0000 的文件内容。
如图 17 和图 18：

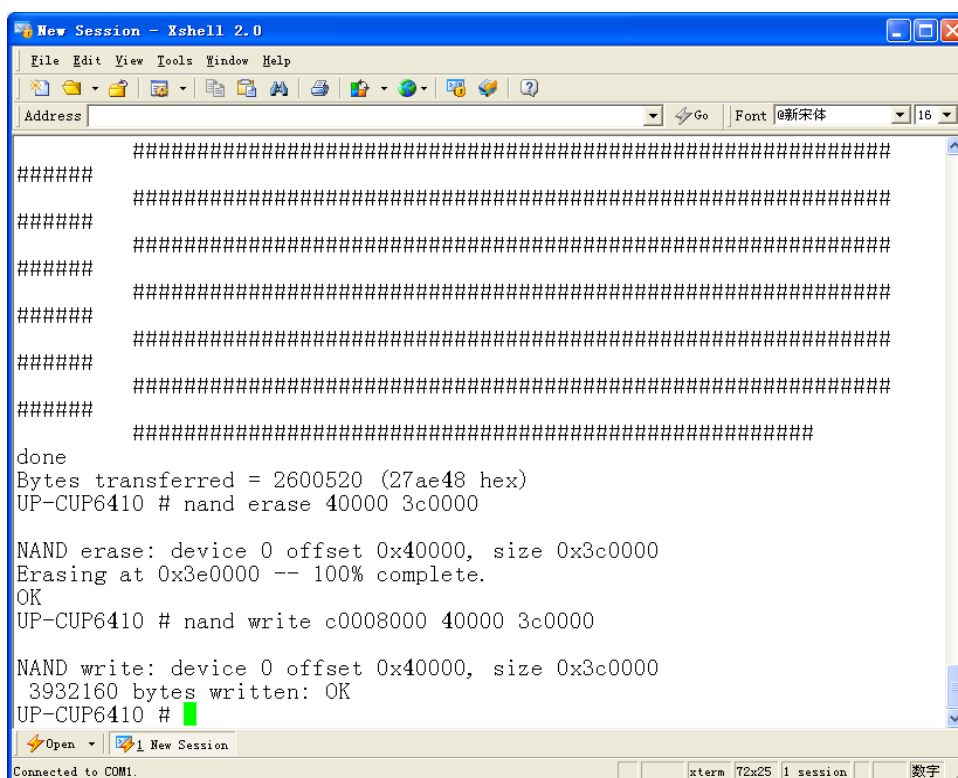


```

New Session - Xshell 2.0
File Edit View Tools Window Help
Address [ ] Go Font @新宋体 16
UP-CUP6410 # tftp c0008000 zImage
dm9000 i/o: 0x18002000, id: 0x90000a46
MAC: 00:22:12:34:56:90
operating at 100M full duplex mode
TFTP from server 192.168.13.12; our IP address is 192.168.13.15
Filename 'zImage'.
Load address: 0xc0008000
Loading: #####
#####
#####
#####
#####
#####
#####
#####
#####
done
Bytes transferred = 2600520 (27ae48 hex)
UP-CUP6410 #

```

图 17



```

New Session - Xshell 2.0
File Edit View Tools Window Help
Address [ ] Go Font @新宋体 16
#####
#####
#####
#####
#####
#####
#####
#####
done
Bytes transferred = 2600520 (27ae48 hex)
UP-CUP6410 # nand erase 40000 3c0000
NAND erase: device 0 offset 0x40000, size 0x3c0000
Erasing at 0x3e0000 -- 100% complete.
OK
UP-CUP6410 # nand write c0008000 40000 3c0000
NAND write: device 0 offset 0x40000, size 0x3c0000
3932160 bytes written: OK
UP-CUP6410 #

```

图 18

内核烧写完毕！

5、 烧写 CRAMFS 文件系统

```
# tftp c0008000 root_mkfs.cramfs
```

将文件系统 root_mkfs.cramfs 下载到 SDRAM 的 0xc0008000 地址处

```
# nand erase 400000 400000
```

将 NANDFLASH 上 0x400000 起始地址处 0x400000 大小的空间擦除

```
# nand write c0008000 400000 400000
```

将 SDRAM 上 0xc0008000 地址开始的内容烧写到 NANDFLASH 的 0x400000 起始地址，大小为 0x400000，如图 19 和图 20：

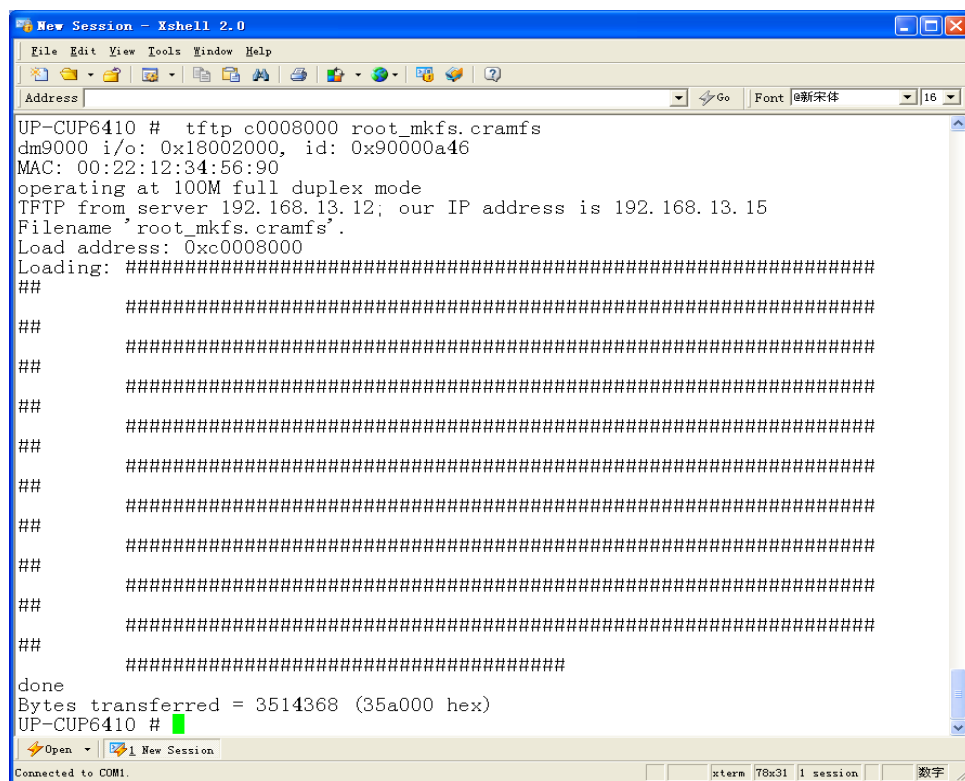


图 19

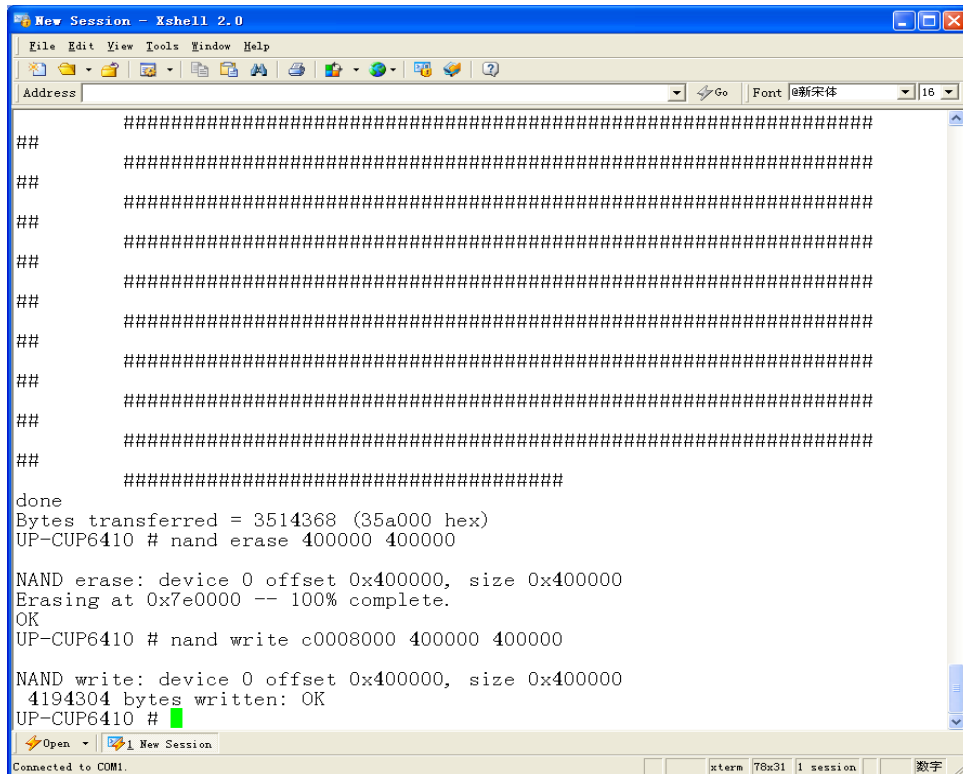


图 20

cramfs 文件系统烧写完毕!

6、 烧写 Android 文件系统

系统烧写 cramfs 文件系统后, 即可利用 cramfs 文件系统烧写 Android 文件系统。

◆ 启动 cramfs 文件系统

进入 u-boot 界面, 执行以下命令:

```
# setenv bootargs noinitrd root=/dev/mtdblock0 console=ttySAC0 init=/linuxrc video=fb:LCD640x480 mem=224M
```

配置启动参数

```
#saveenv
#boot
```

保存配置参数, 启动 cramfs 文件系统, 如图 21:

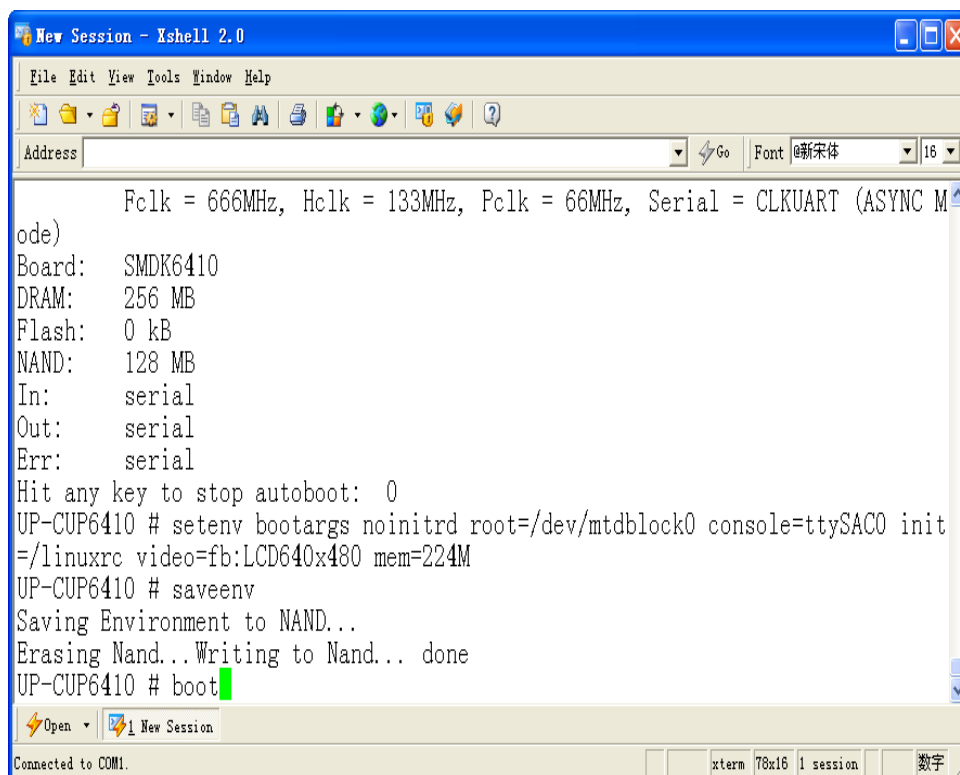


图 21

启动进入 **cramfs** 文件系统，会打印以下启动信息：

```

UP-CUP6410 # boot
NAND read: device 0 offset 0x40000, size 0x3c0000
3932160 bytes read: OK
Boot with zImage

Starting kernel ...

Uncompressing Linux.....
done, booting the kernel.
Linux version 2.6.29 (root@uptech) (gcc version 4.3.2 (Sourcery G++ Lite 2008q3-72) ) #81 PREEMPT Mon Sep 12 01:48:15
CST 2011
CPU: ARMv6-compatible processor [410fb766] revision 6 (ARMv7), cr=00c5387f
CPU: VIPT nonaliasing data cache, VIPT nonaliasing instruction cache
Machine: SMDK6410
Memory policy: ECC disabled, Data cache writeback
CPU S3C6410 (id 0x36410101)
S3C24XX Clocks, (c) 2004 Simtec Electronics
S3C64XX: PLL settings, A=666000000, M=532000000, E=24000000
S3C64XX: HCLK2=266000000, HCLK=133000000, PCLK=66500000
mout_apll: source is fout_apll (1), rate is 666000000
mout_epll: source is fout_epll (1), rate is 24000000
mout_mppll: source is mppll (1), rate is 532000000
mmc_bus: source is mout_epll (0), rate is 24000000

```


Android-3G 出厂程序烧写手册

```
mmc_bus: source is mout_epll (0), rate is 24000000
mmc_bus: source is mout_epll (0), rate is 24000000
usb-bus-host: source is clk_48m (0), rate is 48000000
uclk1: source is dout_mppll (1), rate is 66500000
spi-bus: source is mout_epll (0), rate is 24000000
spi-bus: source is mout_epll (0), rate is 24000000
audio-bus: source is mout_epll (0), rate is 24000000
audio-bus: source is mout_epll (0), rate is 24000000
irda-bus: source is mout_epll (0), rate is 24000000
Built 1 zonelists in Zone order, mobility grouping on.  Total pages: 56896
Kernel command line: noinitrd root=/dev/mtdblock0 console=ttySAC0 init=/linuxrc video=fb:LCD640x480 mem=224M
PID hash table entries: 1024 (order: 10, 4096 bytes)
Console: colour dummy device 80x30
console [ttySAC0] enabled
Dentry cache hash table entries: 32768 (order: 5, 131072 bytes)
Inode-cache hash table entries: 16384 (order: 4, 65536 bytes)
Memory: 224MB = 224MB total
Memory: 221568KB available (4596K code, 849K data, 140K init)
SLUB: Genslabs=10, HWalign=32, Order=0-3, MinObjects=0, CPUs=1, Nodes=1
Calibrating delay loop... 665.19 BogoMIPS (lpj=3325952)
Mount-cache hash table entries: 512
CPU: Testing write buffer coherency: ok
net_namespace: 520 bytes
NET: Registered protocol family 16
S3C6410: Initialising architecture
bio: create slab <bio-0> at 0
SCSI subsystem initialized
usbcore: registered new interface driver usbfs
usbcore: registered new interface driver hub
usbcore: registered new device driver usb
cfg80211: Using static regulatory domain info
cfg80211: Regulatory domain: US
        (start_freq - end_freq @ bandwidth), (max_antenna_gain, max_eirp)
        (2402000 KHz - 2472000 KHz @ 40000 KHz), (600 mBi, 2700 mBm)
        (5170000 KHz - 5190000 KHz @ 40000 KHz), (600 mBi, 2300 mBm)
        (5190000 KHz - 5210000 KHz @ 40000 KHz), (600 mBi, 2300 mBm)
        (5210000 KHz - 5230000 KHz @ 40000 KHz), (600 mBi, 2300 mBm)
        (5230000 KHz - 5330000 KHz @ 40000 KHz), (600 mBi, 2300 mBm)
        (5735000 KHz - 5835000 KHz @ 40000 KHz), (600 mBi, 3000 mBm)
cfg80211: Calling CRDA for country: US
NET: Registered protocol family 2
IP route cache hash table entries: 2048 (order: 1, 8192 bytes)
TCP established hash table entries: 8192 (order: 4, 65536 bytes)
TCP bind hash table entries: 8192 (order: 3, 32768 bytes)
TCP: Hash tables configured (established 8192 bind 8192)
```

```

TCP reno registered
NET: Registered protocol family 1
NetWinder Floating Point Emulator V0.97 (double precision)
ashmem: initialized
yaffs Sep 11 2011 17:31:15 Installing.
msgmni has been set to 433
alg: No test for stdrng (krng)
io scheduler noop registered
io scheduler anticipatory registered
io scheduler deadline registered
io scheduler cfq registered (default)
s3cfs_prlbe*****
info->mem cd882da0
request_mem_region 77100000 100000 s3c-lcd
request_mem_region ok ,add by lyj_uptech
ioremap end,add by lyj_uptech
18000,8000,1000
S3C_VIDINTCON0 = f4600130
writel ok,add it *****
s3cfb_pre_init ok,add by lyj_uptech
s3cfb_set_backlight_power ok,add by lyj_uptech
s3cfb_set_lcd_power ok,add by lyj_uptech
s3cfb_set_backlight_level set ok ,add by lyj_uptech
S3C_LCD clock got enabled :: 133.000 Mhz
LCD TYPE :: LCD640x480 will be initialized
S3CFB_NUM= 4
Window[0] - FB1: map_video_memory: clear ff000000:0012c000
                FB1: map_video_memory: dma=5da00000 cpu=ff000000 size=0012c000
Window[0] - FB2: map_video_memory: clear ff096000:00096000
                FB2: map_video_memory: dma=5da96000 cpu=ff096000 size=00096000
Console: switching to colour frame buffer device 80x30
fb0: s3cfb frame buffer device
S3CFB_NUM= 4
Window[1] - FB1: map_video_memory: clear ff12c000:00096000
                FB1: map_video_memory: dma=5dc00000 cpu=ff12c000 size=00096000
Window[1] - FB2: map_video_memory: clear ff177000:0004b000
                FB2: map_video_memory: dma=5dc4b000 cpu=ff177000 size=0004b000
fb1: s3cfb frame buffer device
S3CFB_NUM= 4
Window[2] - FB1: map_video_memory: clear ff1c2000:00096000
                FB1: map_video_memory: dma=5dd00000 cpu=ff1c2000 size=00096000
fb2: s3cfb frame buffer device
S3CFB_NUM= 4
Window[3] - FB1: map_video_memory: clear ff258000:00096000
                FB1: map_video_memory: dma=5de00000 cpu=ff258000 size=00096000
fb3: s3cfb frame buffer device

```

Android-3G 出厂程序烧写手册

```
#####Hello, world#####
Serial: 8250/16550 driver, 4 ports, IRQ sharing disabled
s3c6400-uart.0: s3c2410_serial0 at MMIO 0x7f005000 (irq = 16) is a S3C6400/10
s3c6400-uart.1: s3c2410_serial1 at MMIO 0x7f005400 (irq = 20) is a S3C6400/10
s3c6400-uart.2: s3c2410_serial2 at MMIO 0x7f005800 (irq = 24) is a S3C6400/10
s3c6400-uart.3: s3c2410_serial3 at MMIO 0x7f005c00 (irq = 28) is a S3C6400/10
brd: module loaded
loop: module loaded
nbd: registered device at major 43
pmem: 1 init
PPP generic driver version 2.4.2
PPP Deflate Compression module registered
PPP BSD Compression module registered
SLIP: version 0.8.4-NET3.019-NEWTTY (dynamic channels, max=256).
CSLIP: code copyright 1989 Regents of the University of California.
dm9000 Ethernet Driver
eth0 (dm9000): not using net_device_ops yet
eth0: dm9000 at ce878000,ce878002 IRQ 115 MAC: 00:22:12:34:56:90
usbcore: registered new interface driver rt73usb
console [netcon0] enabled
netconsole: network logging started
Linux video capture interface: v2.00
Driver 'sd' needs updating - please use bus_type methods
Driver 'sr' needs updating - please use bus_type methods
S3C NAND Driver, (c) 2008 Samsung Electronics
S3C NAND Driver is using hardware ECC.
NAND device: Manufacturer ID: 0xec, Chip ID: 0xf1 (Samsung NAND 128MiB 3,3V 8-bit)
Creating 2 MTD partitions on "NAND 128MiB 3,3V 8-bit":
0x0000000400000-0x0000000800000 : "cramfs"
0x0000000800000-0x0000000c00000 : "ubifs"
usbmon: debugfs is not available
ohci_hcd: USB 1.1 'Open' Host Controller (OHCI) Driver
s3c2410-ohci s3c2410-ohci: S3C24XX OHCI
s3c2410-ohci s3c2410-ohci: new USB bus registered, assigned bus number 1
s3c2410-ohci s3c2410-ohci: irq 79, io mem 0x74300000
usb usb1: New USB device found, idVendor=1d6b, idProduct=0001
usb usb1: New USB device strings: Mfr=3, Product=2, SerialNumber=1
usb usb1: Product: S3C24XX OHCI
usb usb1: Manufacturer: Linux 2.6.29 ohci_hcd
usb usb1: SerialNumber: s3c24xx
usb usb1: configuration #1 chosen from 1 choice
hub 1-0:1.0: USB hub found
hub 1-0:1.0: 2 ports detected
Initializing USB Mass Storage driver...
usbcore: registered new interface driver usb-storage
```

USB Mass Storage support registered.
usbcore: registered new interface driver usbserial
USB Serial support registered for generic
usbcore: registered new interface driver usbserial_generic
usbserial: USB Serial Driver core
USB Serial support registered for GSM modem (1-port)
usbcore: registered new interface driver option
option: v0.7.2:USB Driver for GSM modems
mice: PS/2 mouse device common for all mice
gpio-keys init
s3c_gpio_kps[i] = 124
s3c_gpio_kps[i] = 124 ok
s3c_gpio_kps[i] = 125
s3c_gpio_kps[i] = 125 ok
s3c_gpio_kps[i] = 126
s3c_gpio_kps[i] = 126 ok
s3c_gpio_kps[i] = 127
s3c_gpio_kps[i] = 127 ok
s3c_gpio_kps[i] = 128
s3c_gpio_kps[i] = 128 ok
interrupt_size = 5
interrupt_size = 5
interrupt_size = 5
input: gpio-keys as /devices/platform/gpio-keys/input/input0
S3C Touchscreen driver, (c) 2008 Samsung Electronics
S3C TouchScreen got loaded successfully : 12 bits
input: S3C TouchScreen as /devices/virtual/input/input1
sdhci: Secure Digital Host Controller Interface driver
sdhci: Copyright(c) Pierre Ossman
gpio_leds init
gpio_leds init successfully!!!
usbcore: registered new interface driver usbhid
usbhid: v2.6:USB HID core driver
logger: created 64K log 'log_main'
logger: created 256K log 'log_events'
logger: created 64K log 'log_radio'
Advanced Linux Sound Architecture Driver Version 1.0.18a.
ALSA device list:
No soundcards found.
TCP cubic registered
NET: Registered protocol family 17
can: controller area network core (rev 20090105 abi 8)
NET: Registered protocol family 29
RPC: Registered udp transport module.
RPC: Registered tcp transport module.
VFP support v0.3: implementor 41 architecture 1 part 20 variant b rev 5

Android-3G 出厂程序烧写手册

```
drivers/rtc/hctosys.c: unable to open rtc device (rtc0)
VFS: Mounted root (cramfs filesystem) readonly on device 31:0.
Freeing init memory: 140K
usb 1-1: new full speed USB device using s3c2410-ohci and address 2
usb 1-1: New USB device found, idVendor=05e3, idProduct=0606
usb 1-1: New USB device strings: Mfr=1, Product=2, SerialNumber=0
usb 1-1: Product: USB Hub 2.0
usb 1-1: Manufacturer:
usb 1-1: configuration #1 chosen from 1 choice
hub 1-1:1.0: USB hub found
hub 1-1:1.0: 4 ports detected
hwclock: can't open '/dev/misc/rtc': No such file or directory
eth0: link down

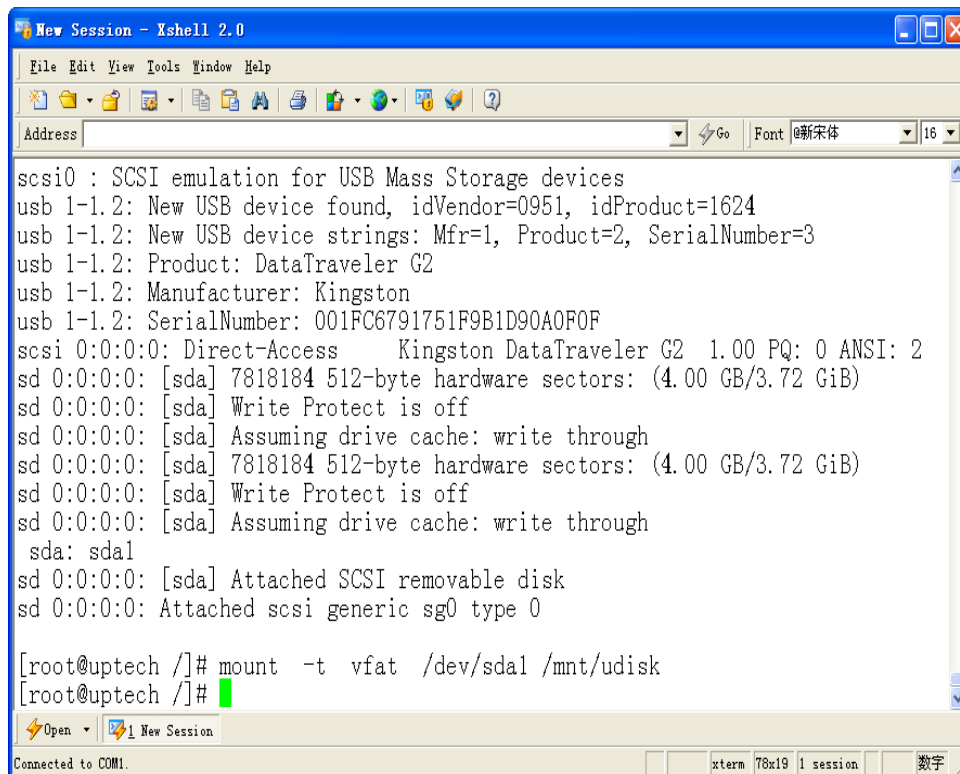
Please press Enter to activate this console. eth0: link up, 100Mbps, full-duplex, lpa 0x45E1

[root@uptech /]#
```

◆ 挂载 U 盘

当设备正确检测到 U 盘并识别后,使用 `mount` 命令根据终端提示将 U 盘分区挂载到 ARM 端 `/mnt/sdcard` 目录, 如图 22:

```
#mount -t vfat /dev/sda1 /mnt/udisk
```



```
New Session - Xshell 2.0
File Edit View Tools Window Help
Address [ ] Go Font @新宋体 16
scsi0 : SCSI emulation for USB Mass Storage devices
usb 1-1.2: New USB device found, idVendor=0951, idProduct=1624
usb 1-1.2: New USB device strings: Mfr=1, Product=2, SerialNumber=3
usb 1-1.2: Product: DataTraveler G2
usb 1-1.2: Manufacturer: Kingston
usb 1-1.2: SerialNumber: 001FC6791751F9B1D90A0F0F
scsi 0:0:0:0: Direct-Access Kingston DataTraveler G2 1.00 PQ: 0 ANSI: 2
sd 0:0:0:0: [sda] 7818184 512-byte hardware sectors: (4.00 GB/3.72 GiB)
sd 0:0:0:0: [sda] Write Protect is off
sd 0:0:0:0: [sda] Assuming drive cache: write through
sd 0:0:0:0: [sda] 7818184 512-byte hardware sectors: (4.00 GB/3.72 GiB)
sd 0:0:0:0: [sda] Write Protect is off
sd 0:0:0:0: [sda] Assuming drive cache: write through
sda: sdal
sd 0:0:0:0: [sda] Attached SCSI removable disk
sd 0:0:0:0: Attached scsi generic sg0 type 0

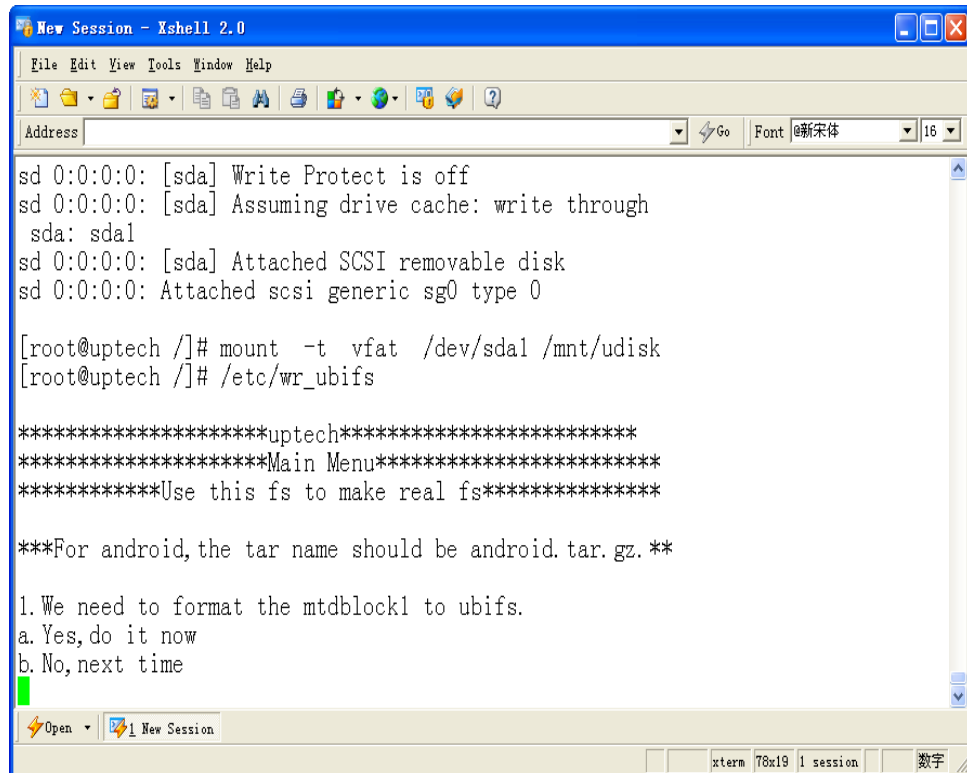
[root@uptech /]# mount -t vfat /dev/sda1 /mnt/udisk
[root@uptech /]#
```

图 22

备注：先将光盘/IMG/下的 android.tar.gz 文件系统镜像复制到 U 盘

◆ 烧写 android 文件系统镜像

1) 执行 wr_ubifs 命令烧写文件系统，如图所示：



```
New Session - Xshell 2.0
File Edit View Tools Window Help
Address [Go] Font @新宋体 [16]
sd 0:0:0:0: [sda] Write Protect is off
sd 0:0:0:0: [sda] Assuming drive cache: write through
sda: sdal
sd 0:0:0:0: [sda] Attached SCSI removable disk
sd 0:0:0:0: Attached scsi generic sg0 type 0

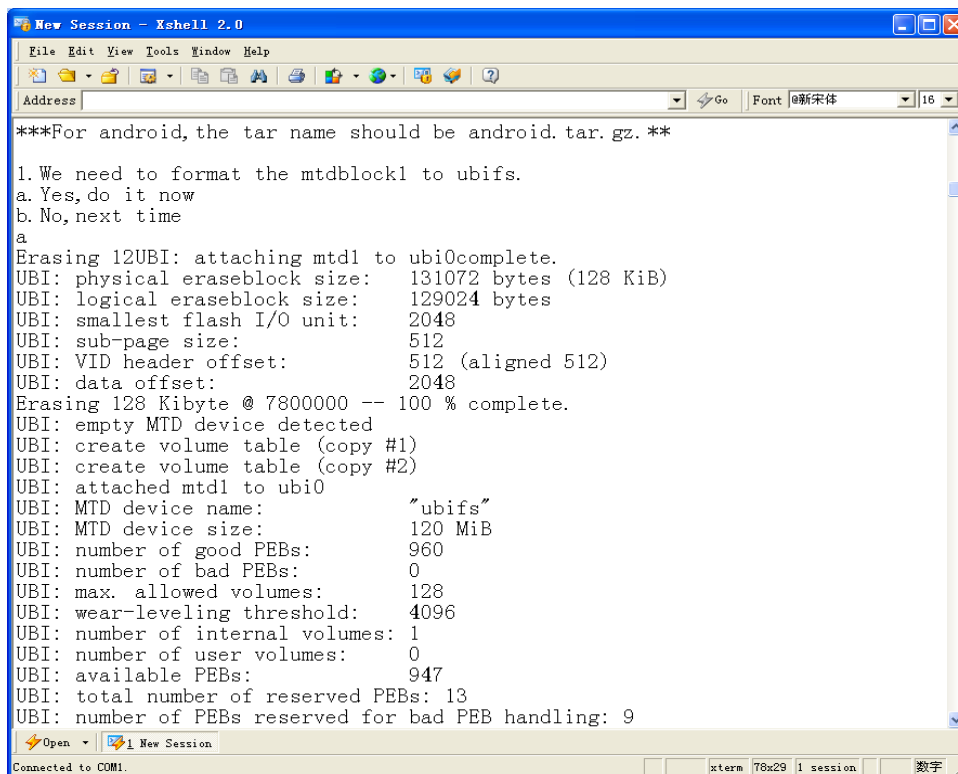
[root@uptech /]# mount -t vfat /dev/sdal /mnt/udisk
[root@uptech /]# /etc/wr_ubifs

*****uptech*****
*****Main Menu*****
*****Use this fs to make real fs*****

***For android, the tar name should be android.tar.gz.**

1.We need to format the mtdblock1 to ubifs.
a.Yes,do it now
b.No,next time
█
```

2) 选择 a 格式化磁盘分区，之后烧写文件系统如图所示：



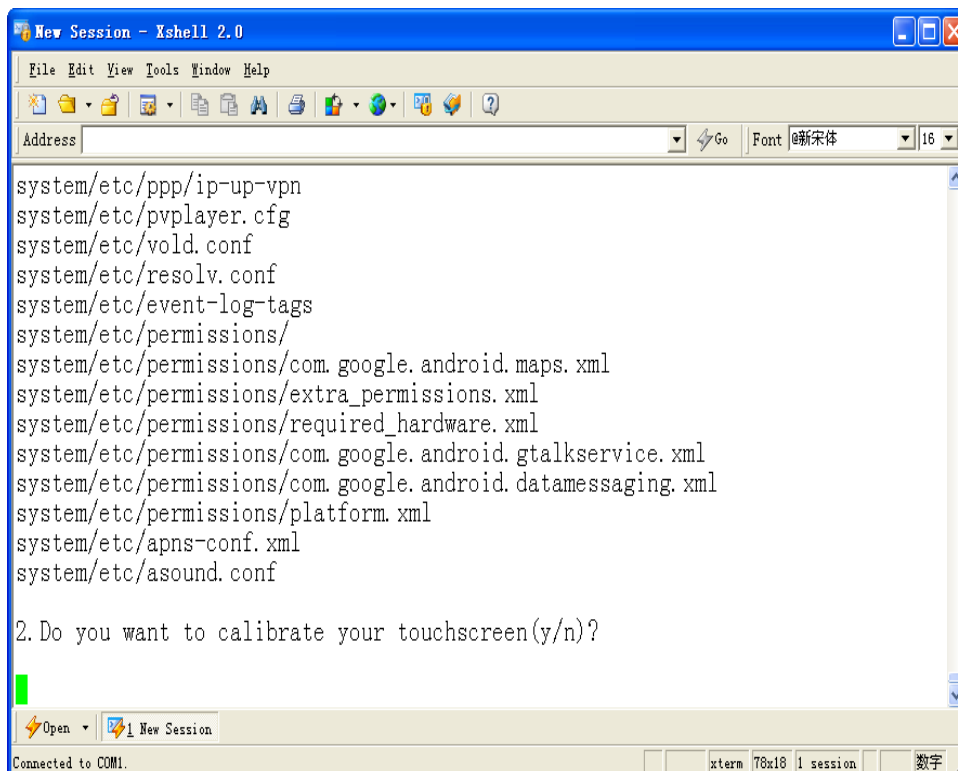
```
New Session - Xshell 2.0
File Edit View Tools Window Help
Address
***For android, the tar name should be android.tar.gz.***

1. We need to format the mtdblock1 to ubifs.
a. Yes, do it now
b. No, next time
a
Erasing 12UBI: attaching mtd1 to ubi0complete.
UBI: physical eraseblock size: 131072 bytes (128 KiB)
UBI: logical eraseblock size: 129024 bytes
UBI: smallest flash I/O unit: 2048
UBI: sub-page size: 512
UBI: VID header offset: 512 (aligned 512)
UBI: data offset: 2048
Erasing 128 Kibyte @ 7800000 -- 100 % complete.
UBI: empty MTD device detected
UBI: create volume table (copy #1)
UBI: create volume table (copy #2)
UBI: attached mtd1 to ubi0
UBI: MTD device name: "ubifs"
UBI: MTD device size: 120 MiB
UBI: number of good PEBs: 960
UBI: number of bad PEBs: 0
UBI: max. allowed volumes: 128
UBI: wear-leveling threshold: 4096
UBI: number of internal volumes: 1
UBI: number of user volumes: 0
UBI: available PEBs: 947
UBI: total number of reserved PEBs: 13
UBI: number of PEBs reserved for bad PEB handling: 9

Open New Session
Connected to COM1. xterm 78x29 1 session 数字
```

备注：在烧写过程中，不要按下 PC 机键盘任意键，否则系统认为是非法输入，不进行触摸屏校正。此时需要手动触摸屏校正，方法见“7 触摸屏校正”。

提示是否校正触摸屏，如图所示：

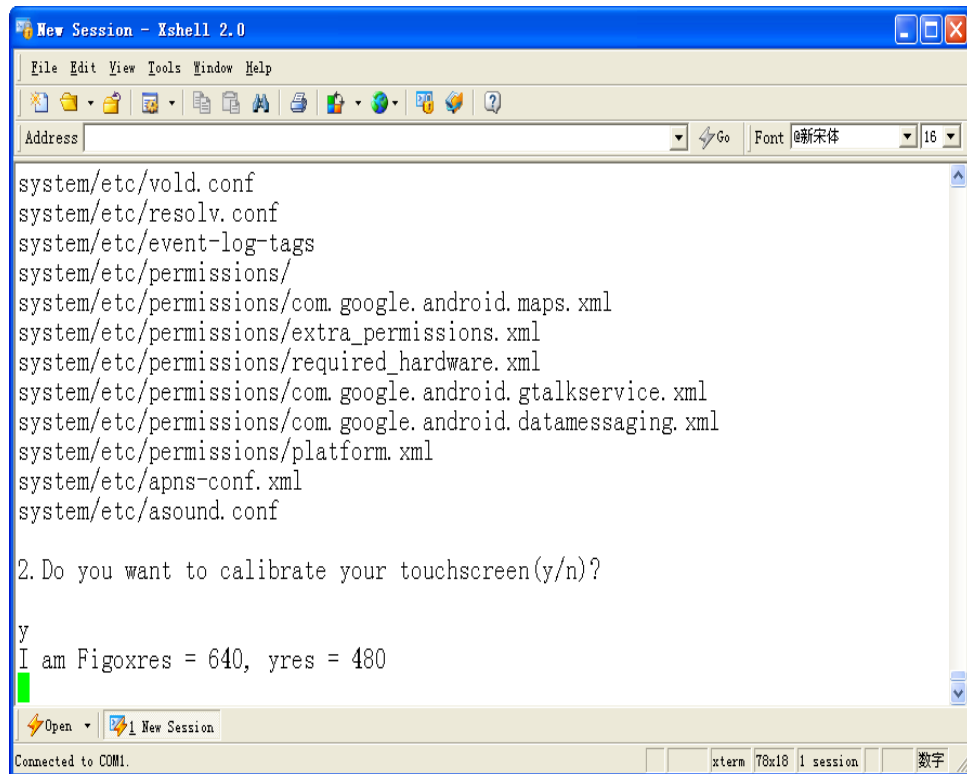


```
New Session - Xshell 2.0
File Edit View Tools Window Help
Address
system/etc/ppp/ip-up-vpn
system/etc/pvplayer.cfg
system/etc/vold.conf
system/etc/resolv.conf
system/etc/event-log-tags
system/etc/permissions/
system/etc/permissions/com.google.android.maps.xml
system/etc/permissions/extra_permissions.xml
system/etc/permissions/required_hardware.xml
system/etc/permissions/com.google.android.gtalkservice.xml
system/etc/permissions/com.google.android.datamessaging.xml
system/etc/permissions/platform.xml
system/etc/apns-conf.xml
system/etc/asound.conf

2. Do you want to calibrate your touchscreen(y/n)?
[Green cursor]

Open New Session
Connected to COM1. xterm 78x18 1 session 数字
```

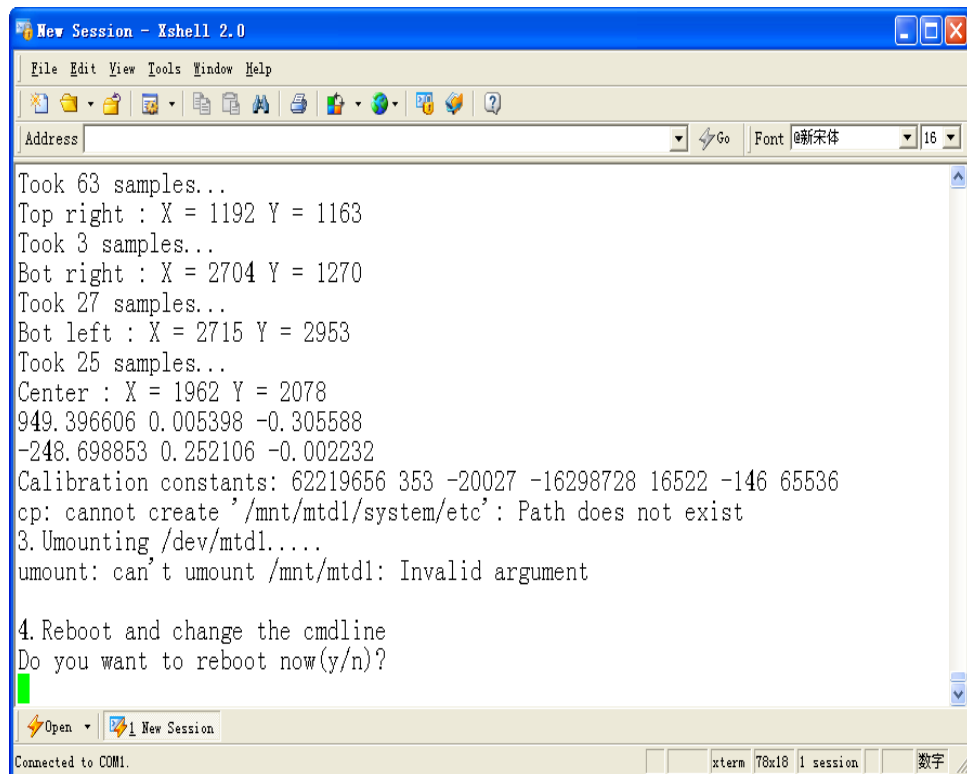
- 3) 选择 y 校正触摸屏，如图所示：



```
New Session - Xshell 2.0
File Edit View Tools Window Help
Address
system/etc/vold.conf
system/etc/resolv.conf
system/etc/event-log-tags
system/etc/permissions/
system/etc/permissions/com.google.android.maps.xml
system/etc/permissions/extra_permissions.xml
system/etc/permissions/required_hardware.xml
system/etc/permissions/com.google.android.gtalkservice.xml
system/etc/permissions/com.google.android.datamessaging.xml
system/etc/permissions/platform.xml
system/etc/apns-conf.xml
system/etc/asound.conf

2. Do you want to calibrate your touchscreen(y/n)?
y
I am Figo xres = 640, yres = 480
Open New Session
Connected to COM1. xterm 78x18 1 session 数字
```

- 4) 按照 LCD 显示器的提示，依次点击触摸屏，如图所示：



```
New Session - Xshell 2.0
File Edit View Tools Window Help
Address
Took 63 samples...
Top right : X = 1192 Y = 1163
Took 3 samples...
Bot right : X = 2704 Y = 1270
Took 27 samples...
Bot left : X = 2715 Y = 2953
Took 25 samples...
Center : X = 1962 Y = 2078
949.396606 0.005398 -0.305588
-248.698853 0.252106 -0.002232
Calibration constants: 62219656 353 -20027 -16298728 16522 -146 65536
cp: cannot create '/mnt/mtd1/system/etc': Path does not exist
3. Umounting /dev/mtd1....
umount: can't umount /mnt/mtd1: Invalid argument

4. Reboot and change the cmdline
Do you want to reboot now(y/n)?
Open New Session
Connected to COM1. xterm 78x18 1 session 数字
```

- 5) 输入 y 重启系统，如图所示：

Android 文件系统烧写完毕！

◆ 启动 Android 文件系统

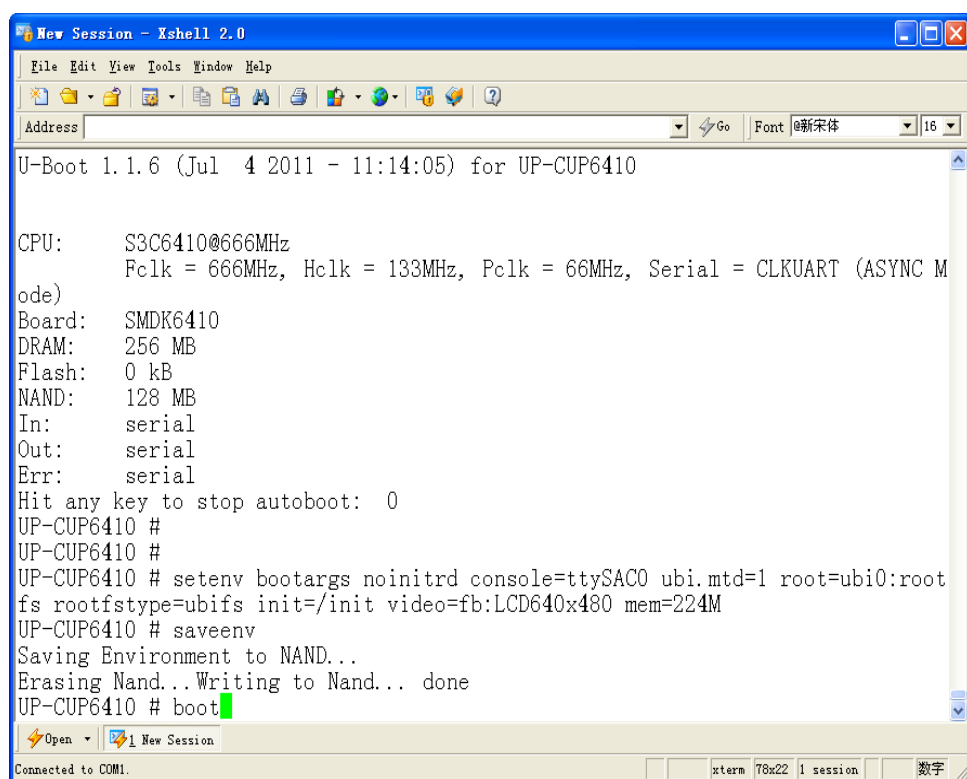
进入 u-boot 界面, 执行以下命令:

```
# setenv bootargs nointrd console=ttySAC0 ubi.mtd=1 root=ubi0:rootfs rootfstype=ubifs init=/init
video=fb:LCD640x480 mem=224M
```

配置启动参数

```
#saveenv
#boot
```

保存配置参数, 启动 Android 文件系统, 如图所示:

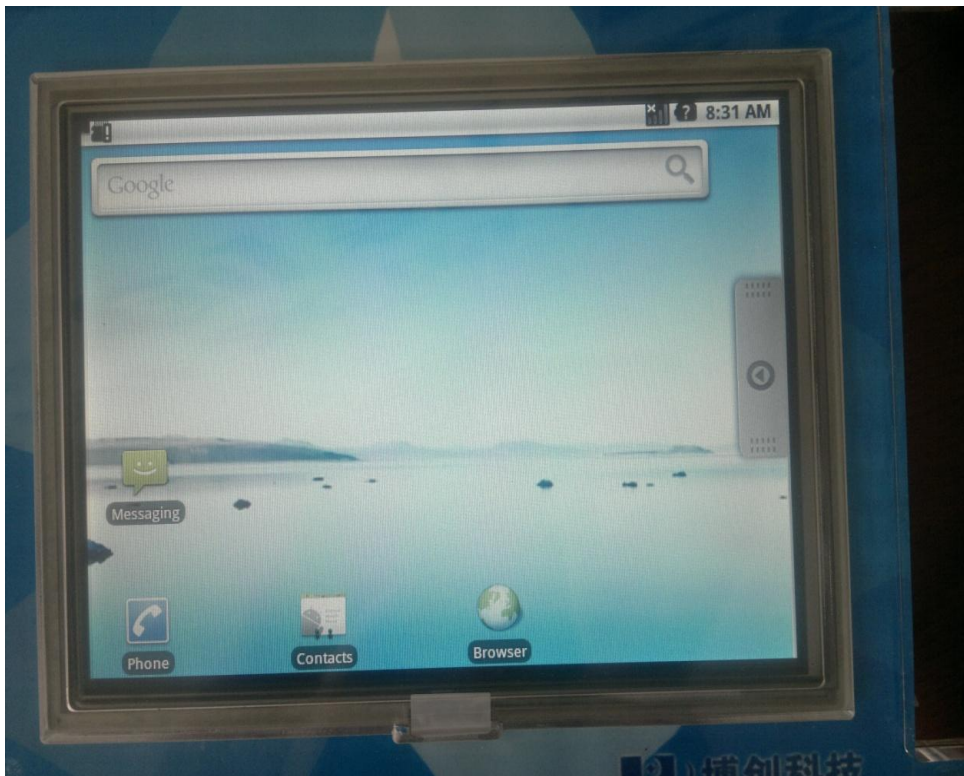


The screenshot shows a terminal window titled "New Session - Yshell 2.0". The terminal output displays the U-Boot version (1.1.6) and board information (UP-CUP6410). It then shows the execution of the 'setenv' command to configure boot parameters, followed by 'saveenv' to save them to NAND. The final step is 'boot', which starts the Android filesystem. The terminal interface includes a menu bar (File, Edit, View, Tools, Window, Help), a toolbar with icons for file operations, and a status bar at the bottom indicating the connection to COM1.

```
New Session - Yshell 2.0
File Edit View Tools Window Help
Address [ ] Go Font @新宋体 16
U-Boot 1.1.6 (Jul  4 2011 - 11:14:05) for UP-CUP6410

CPU:      S3C6410@666MHz
        Felk = 666MHz, Hclk = 133MHz, Pclk = 66MHz, Serial = CLKUART (ASYNC M
ode)
Board:    SMDK6410
DRAM:     256 MB
Flash:    0 kB
NAND:     128 MB
In:       serial
Out:      serial
Err:      serial
Hit any key to stop autoboot:  0
UP-CUP6410 #
UP-CUP6410 #
UP-CUP6410 # setenv bootargs nointrd console=ttySAC0 ubi.mtd=1 root=ubi0:root
fs rootfstype=ubifs init=/init video=fb:LCD640x480 mem=224M
UP-CUP6410 # saveenv
Saving Environment to NAND...
Erasing Nand...Writing to Nand... done
UP-CUP6410 # boot
```

系统启动后即可进入图形界面, 终端显示:



```
New Session - Xshell 2.0
File Edit View Tools Window Help
Address: [ ] Go Font @新宋体 16
#
#
# ls
d
dev
etc
sys
data
init
proc
sbin
default.prop
cache
init.goldfish.rc
init.rc
sdcard
sqlite_stmt_journals
system
init.smdk6410.rc
config
# █
Open New Session
Connected to COM1. xterm 78x22 1 session 数字
```

7、 触摸屏校正

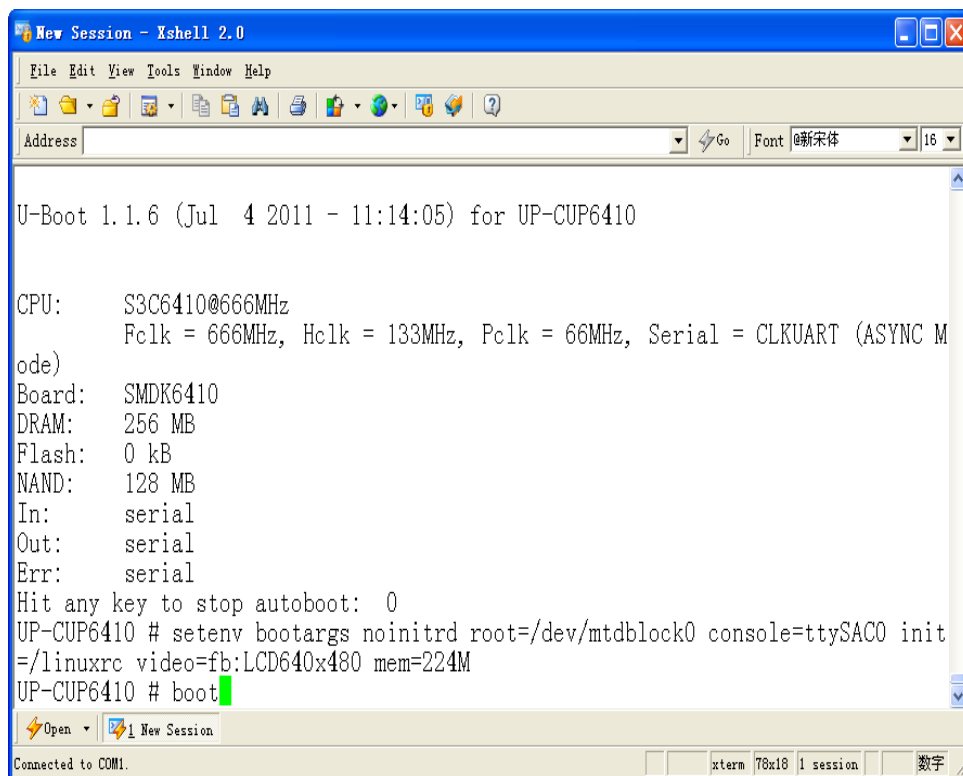
Android 触摸屏校正，在 cramfs 文件系统下进行校正，然后将校正文件复至到 android 文件系统 /system/etc 目录下。

- ◆ 启动 cramfs 文件系统

进入 u-boot 界面, 执行以下命令:

```
# setenv bootargs noinitrd root=/dev/mtdblock0 console=ttySAC0 init=/linuxrc video=fb:LCD640x480 mem=224M
#boot
```

配置启动参数, 启动 cramfs 文件系统, 如图所示:

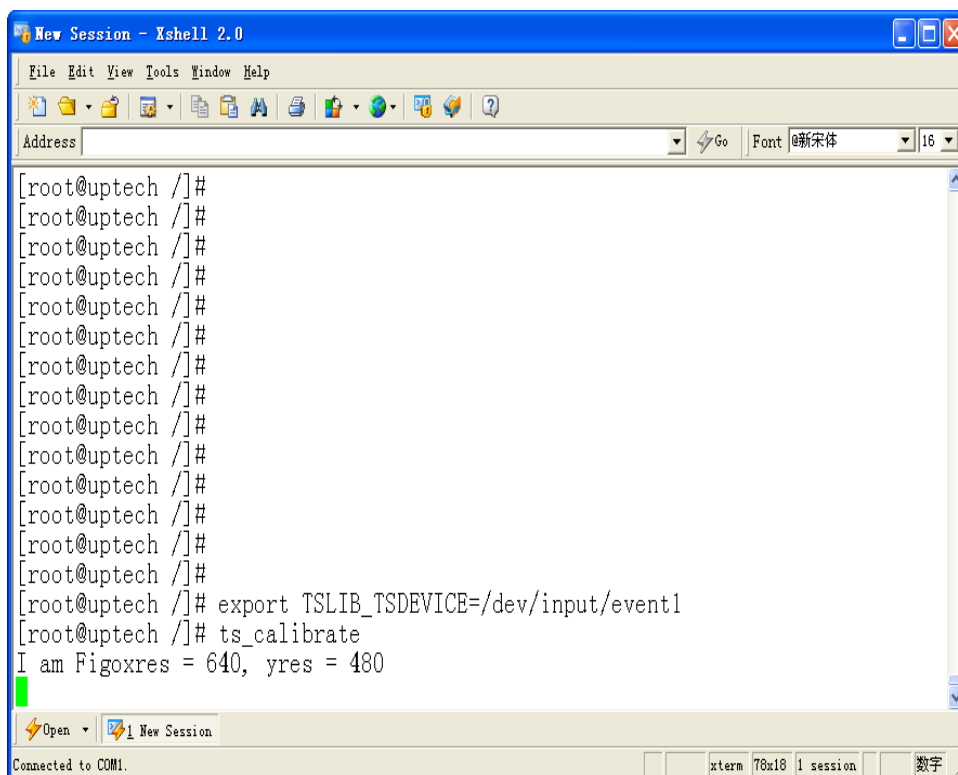


◆ 设置环境变量

```
#export TSLIB_TSDEVICE=/dev/input/event1
```

◆ 执行触摸屏校正程序 ts_calibrate

```
#ts_calibrate
```

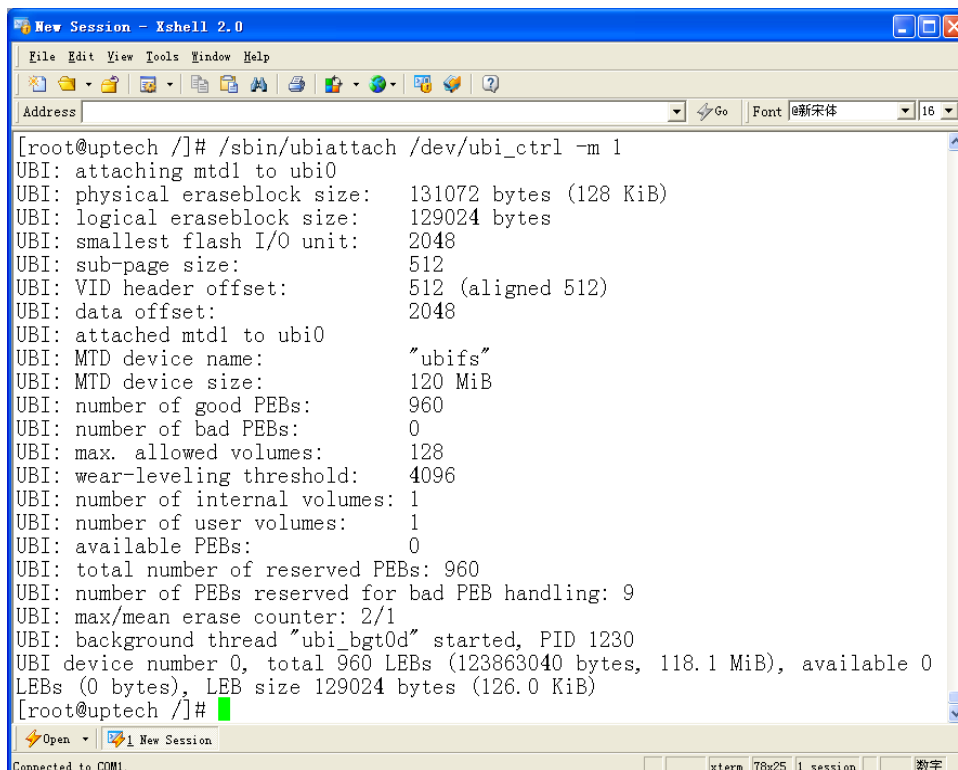


```
New Session - Xshell 2.0
File Edit View Tools Window Help
Address
[root@uptech /]#
[root@uptech /]#
[root@uptech /]#
[root@uptech /]#
[root@uptech /]#
[root@uptech /]#
[root@uptech /]#
[root@uptech /]#
[root@uptech /]#
[root@uptech /]#
[root@uptech /]#
[root@uptech /]#
[root@uptech /]#
[root@uptech /]# export TSLIB_TSDEVICE=/dev/input/event1
[root@uptech /]# ts_calibrate
I am Figoxres = 640, yres = 480
Open New Session
Connected to COM1. xterm 78x18 1 session 数字
```

按照 LCD 显示器的提示，依次点击触摸屏

- ◆ 创建 UBI 设备，并关联 MTD1

```
#/sbin/ubiattach /dev/ubi_ctrl -m 1
```



```
New Session - Xshell 2.0
File Edit View Tools Window Help
Address
[root@uptech /]# /sbin/ubiattach /dev/ubi_ctrl -m 1
UBI: attaching mtd1 to ubi0
UBI: physical eraseblock size: 131072 bytes (128 KiB)
UBI: logical eraseblock size: 129024 bytes
UBI: smallest flash I/O unit: 2048
UBI: sub-page size: 512
UBI: VID header offset: 512 (aligned 512)
UBI: data offset: 2048
UBI: attached mtd1 to ubi0
UBI: MTD device name: "ubifs"
UBI: MTD device size: 120 MiB
UBI: number of good PEBs: 960
UBI: number of bad PEBs: 0
UBI: max. allowed volumes: 128
UBI: wear-leveling threshold: 4096
UBI: number of internal volumes: 1
UBI: number of user volumes: 1
UBI: available PEBs: 0
UBI: total number of reserved PEBs: 960
UBI: number of PEBs reserved for bad PEB handling: 9
UBI: max/mean erase counter: 2/1
UBI: background thread "ubi_bgt0d" started, PID 1230
UBI device number 0, total 960 LEBs (123863040 bytes, 118.1 MiB), available 0
LEBs (0 bytes), LEB size 129024 bytes (126.0 KiB)
[root@uptech /]#
Open New Session
Connected to COM1. xterm 78x25 1 session 数字
```

- ◆ 挂载 ubifs 分区至/mnt/mtd1/目录下

```
#mount -t ubifs ubi0_0 /mnt/mtd1
```

```
New Session - Xshell 2.0
File Edit View Tools Window Help
Address
UBI: MTD device name:      "ubifs"
UBI: MTD device size:      120 MiB
UBI: number of good PEBs:   960
UBI: number of bad PEBs:    0
UBI: max. allowed volumes: 128
UBI: wear-leveling threshold: 4096
UBI: number of internal volumes: 1
UBI: number of user volumes: 1
UBI: available PEBs:        0
UBI: total number of reserved PEBs: 960
UBI: number of PEBs reserved for bad PEB handling: 9
UBI: max/mean erase counter: 2/1
UBI: background thread "ubi_bgt0d" started, PID 1230
UBI device number 0, total 960 LEBs (123863040 bytes, 118.1 MiB), available 0
LEBs (0 bytes), LEB size 129024 bytes (126.0 KiB)
[root@uptech /]# mount -t ubifs ubi0_0 /mnt/mtd1
UBIFS: recovery needed
UBIFS: recovery completed
UBIFS: mounted UBI device 0, volume 0, name "rootfs"
UBIFS: file system size: 120895488 bytes (118062 KiB, 115 MiB, 937 LEBs)
UBIFS: journal size: 6064128 bytes (5922 KiB, 5 MiB, 47 LEBs)
UBIFS: media format: 4 (latest is 4)
UBIFS: default compressor: LZO
UBIFS: reserved for root: 5182151 bytes (5060 KiB)
[root@uptech /]#
```

◆ 复制校正文件至 Android 系统的 `system/etc` 目录下

```
#cp /var/pointercal /mnt/mtd1/system/etc
```

```
New Session - Xshell 2.0
File Edit View Tools Window Help
Address
UBI: MTD device size:      120 MiB
UBI: number of good PEBs:   960
UBI: number of bad PEBs:    0
UBI: max. allowed volumes: 128
UBI: wear-leveling threshold: 4096
UBI: number of internal volumes: 1
UBI: number of user volumes: 1
UBI: available PEBs:        0
UBI: total number of reserved PEBs: 960
UBI: number of PEBs reserved for bad PEB handling: 9
UBI: max/mean erase counter: 2/1
UBI: background thread "ubi_bgt0d" started, PID 1230
UBI device number 0, total 960 LEBs (123863040 bytes, 118.1 MiB), available 0
LEBs (0 bytes), LEB size 129024 bytes (126.0 KiB)
[root@uptech /]# mount -t ubifs ubi0_0 /mnt/mtd1
UBIFS: recovery needed
UBIFS: recovery completed
UBIFS: mounted UBI device 0, volume 0, name "rootfs"
UBIFS: file system size: 120895488 bytes (118062 KiB, 115 MiB, 937 LEBs)
UBIFS: journal size: 6064128 bytes (5922 KiB, 5 MiB, 47 LEBs)
UBIFS: media format: 4 (latest is 4)
UBIFS: default compressor: LZO
UBIFS: reserved for root: 5182151 bytes (5060 KiB)
[root@uptech /]# cp /var/pointercal /mnt/mtd1/system/etc
[root@uptech /]#
```

触摸屏校正完毕！

启动 Android 文件系统见“烧写 Android 文件系统”的“启动 Android 文件系统”部分。

