君正®

JZ4780 Android4.1 开发指南

Date: Sep. 2013

北京君正集成电路股份有限公司
Ingenic Semiconductor Co., Ltd.

君正®

JZ4780 Android4.1 开发指南

Release history

Date	Revision	Change
Feb. 2013	1.01	First release
Jul. 2013	1.02	修改下载源码方式 增加 Fastboot 章节，内容待完善 增加 V1.3.2 版本烧录工具的使用以及如何更新驱动 增加板级配置部分
Sep.2013	1.03	更新 git 源码下载地址和公司地址

Disclaimer

This documentation is provided for use with Ingenic products. No license to Ingenic property rights is granted. Ingenic assumes no liability, provides no warranty either expressed or implied relating to the usage, or intellectual property right infringement except as provided for by Ingenic Terms and Conditions of Sale.

Ingenic products are not designed for and should not be used in any medical or life sustaining or supporting equipment.

All information in this document should be treated as preliminary. Ingenic may make changes to this document without notice. Anyone relying on this documentation should contact Ingenic for the current documentation and errata.

北京君正集成电路股份有限公司
地址：北京市海淀区东北旺西路中关村软件园二期君正总部大楼
电话:(86-10)56345000
传真:(86-10)56345001
Http: //www.ingenic.cn

目录

1 概述

本文基于君正 JZ4780（32 位的 RISC）处理器、 Android4.1 系统的开发板，编写文档，以供您搭建适用于 JZ4780、Android4.1 系统的开发环境。

该文档主要介绍了以下几个方面的内容：

1) 基于 64 位 PC 机准备的基础开发环境；
2) 搭建编译环境；
3) 下载源码；
4) 编译；
5) 如何烧录；

2 建立开发环境

开始基于君正开发板搭建开发环境，需要准备一台 Linux 机器，我们推荐使用 64 位的 Ubuntu 12.04 系统的 PC 机

2.1 准备 PC 上的基础环境

1) 准备一台环境干净的 64 位 Ubuntu（推荐版本是 12.04）系统的 PC 机；
2) 可以如下命令来使 64 位的系统能够搭建君正 32 位处理器的开发环境：
   apt-get install cpp-4.6 g++-4.6 gcc-4.6 gcc-4.6-multilib gcc g++ cpp gcc-multilib g++-4.6-multilib git-core gnupg flex bison gperf build-essential zip curl libc6-dev libncurses5-dev:i386 x11proto-core-dev libx11-dev:i386 libreadline6-dev:i386 libgl1-mesa-glx:i386 libgl1-mesa-dev g++-multilib mingw32 tofrodos python-markdown libxml2-utils xsltproc zlib1g-dev:i386 la32-libs gawk qt4-dev-tools ncurses*;
3) 君正 4780 的 android 开发环境与 google 官方完全兼容，更详细的编译环境建立方法，请参考 android 的官方网站：http://source.android.com/source/initializing.html

2.2 安装 JDK

默认的 Ubuntu 环境中预装的 JAVA 环境是 openjdk，而 Android 编译必须用 oracle(sun)的 JAVA。请将 Ubuntu 上自带的 openjdk 先完全卸载，随后安装 JAVA 官网上下载的最新的对应于 64 位机器的 JDK 安装包，我们推荐安装 jdk-6u37-linux-x64 的版本；

2.3 额外的编译环境

当标准的 android 编译环境准备好之后，有几个额外的准备步骤，这些可能有别于官方的标准环境。编译 Android 的 Linux 内核，需要安装我们的交叉编译链工具：**mips-4.3-mxu.tar.bz2**
将 mips-4.3-mxu.tar.bz2 解压到 Ubuntu 的/opt 下面，然后将其添加到环境变量 PATH 中，以便任意 shell 都能访问到。

2.4 下载源码

君正 4780Android 可以提供完整的代码，请通过正确的渠道获得，下载的方式可通过 GIT。

1. 下载 repo 脚本；
   $ mkdir android-ingenic
   $ cd android-ingenic
   $ wget http://git.ingenic.cn:8082/bj/repo
   $ chmod +x repo
2. 下载 Android 源码；
   $ ./repo init -u http://git.ingenic.cn/android/platform/manifest -b dev-ing-jb-local-4780 -m outside.xml
   $ ./repo sync
   $ ./repo forall -c "git reset --hard ingenic-COMMONV1.0-20130403"

注意：“ingenic-COMMONV1.0-20130403”是打 TAG 的 Android 源码，最新信息请关注君正官网：http://www.ingenic.cn。

3 编译

3.1 代码目录说明

将代码的压缩包解开，会得到以下的一个目录结构：

```
yliu@ubuntu:/mnt/sdd1/gitwork/android-41$ ls
abi         cts             external      kernel             ndk         sdk
bionic      dalvik          frameworks    libcore            packages    system
bootable    development     gdk           libnativehelper    pdk
build       device          hardware      Makefile           prebuilts
```

其中：

1) Bootloader 的源码，位于：bootable/bootloader/xboot
2) Android 的 linux 内核源码，位于：kernel
3) 完整 android 的源码。

3.2 编译系统文件

以 4780 小开发板 grus 为例，进入到工程根目录，执行编译命令：

user@ubuntu:~/android-41$ **./build/scripts/mbuild_nand allimg grus grus_nand**

以 4780 的 warrior 板为例：

user@ubuntu:~/android-41$ **./build/scripts/mbuild_nand allimg warrior warrior_nand**

整个编译过程需要持续几个小时，具体的时间取决于主机的性能情况。编译完成后，会在 out/target/product/grus/或者是 out/target/product/warrior/目录下生成可以烧录到开发板的文件：

system.img.bin （适用于 USB 烧录，对应“USBBurnTool”工具）

system.img （适用于 SD 卡烧录，对应“SDBurnTool”工具）

boot.img

3.3 编译 X-boot

进入到 bootable/bootloader/xboot/目录下，执行命令：

user@ubuntu:~/android-41/bootable/bootloader/xboot$ **make clean; make grus_nand_config; make**，或者是

user@ubuntu:~/android-41/bootable/bootloader/xboot$ **make clean; make warrior_nand_config; make**

编译完成后，会在当前目录下生成 x-boot-nand.bin 文件。

当成功生产 x-boot-nand.bin，boot.img 和 system.img.bin，就可以进行烧录。

3.4 准备烧录文件

我们用到下面的几个文件：

- x-boot-nand.bin
- boot.img
- system.img （或者 system.img.bin）

烧录方法可参见 USBBurnTool/SDBurnTool 的用户指南或者是快速指南。

4 FastBoot 烧录

4.1 进入 Fastboot 模式

按住“esc”键，进行一次 reset 操作，直到看到 fastboot 的 logo（绿色的 Android 机器人出现，并且，屏幕有文字打印）

4.2 擦除和烧录镜像文件

5 烧录程序到开发板

5.1 USB 烧录工具烧录

5.1.1 使用 V1.3.2 版本及以后的烧录工具

5.1.1.1 更新驱动

以使用 V1.3.2 版本的烧录工具为例：将 USBBurnTool_V1.3.2_20130626.rar 解压到您的工作目录；

1) 按住“SW5”键，再按住“SW7”RESET 键，进入烧录 boot 模式；
2) 鼠标右键“我的电脑”—>“管理”—>“设备管理器”—>“Ingenic Usb Boot Class”—>“Usb Boot Device”—>“更新驱动程序”；

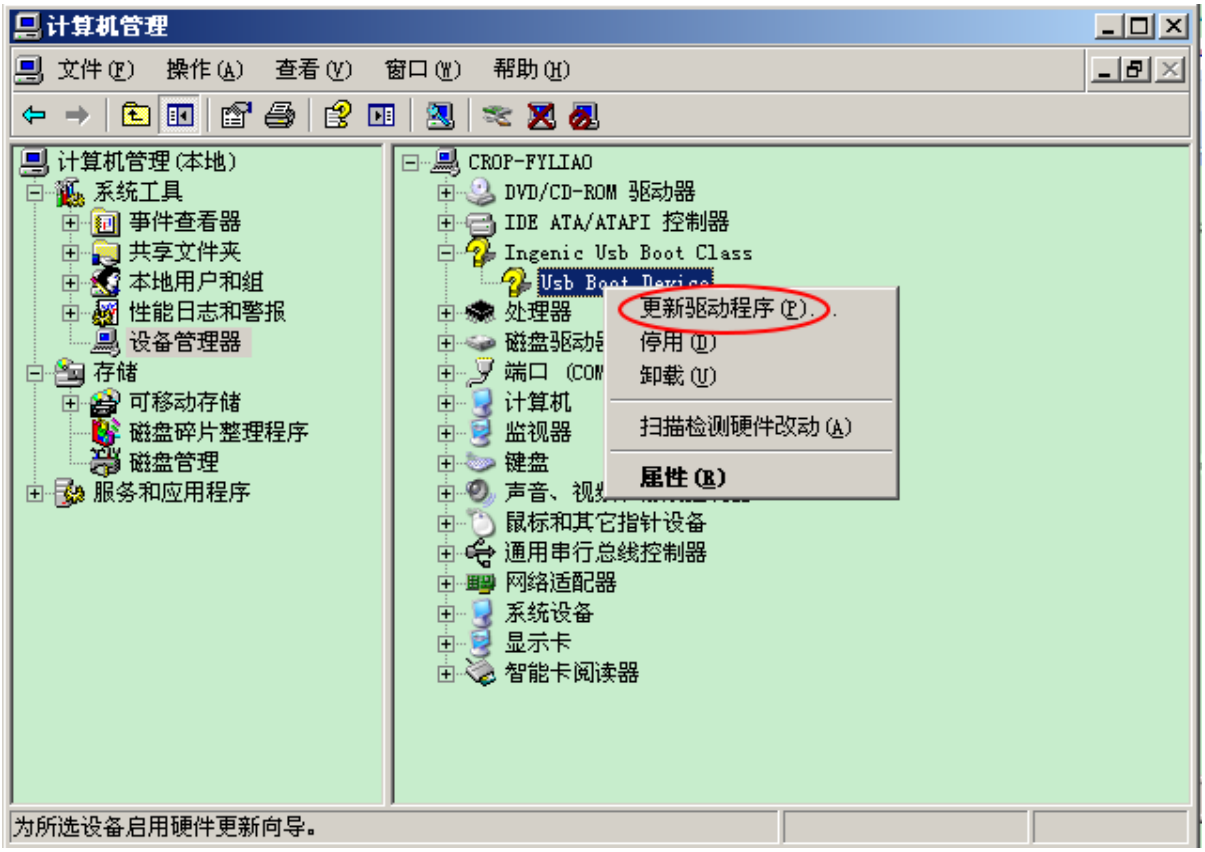

图 5-5

3) 勾选“否，暂时不”后，点击“下一步”；

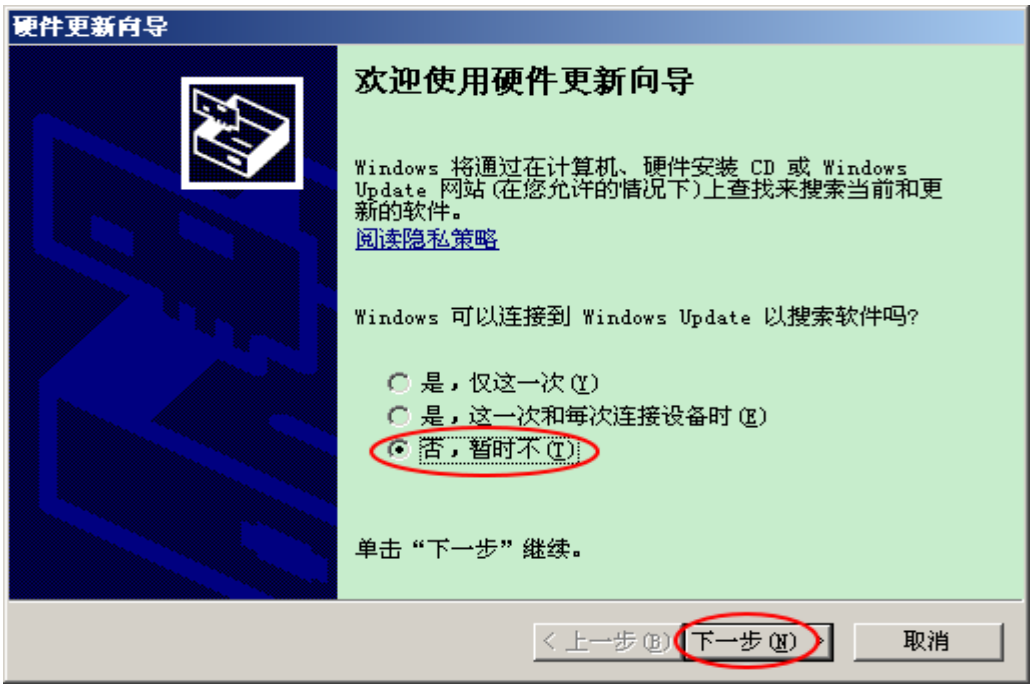

图 5-6

4) 勾选“从列表或指定位置安装（高级）（S）”，再点击“下一步”；

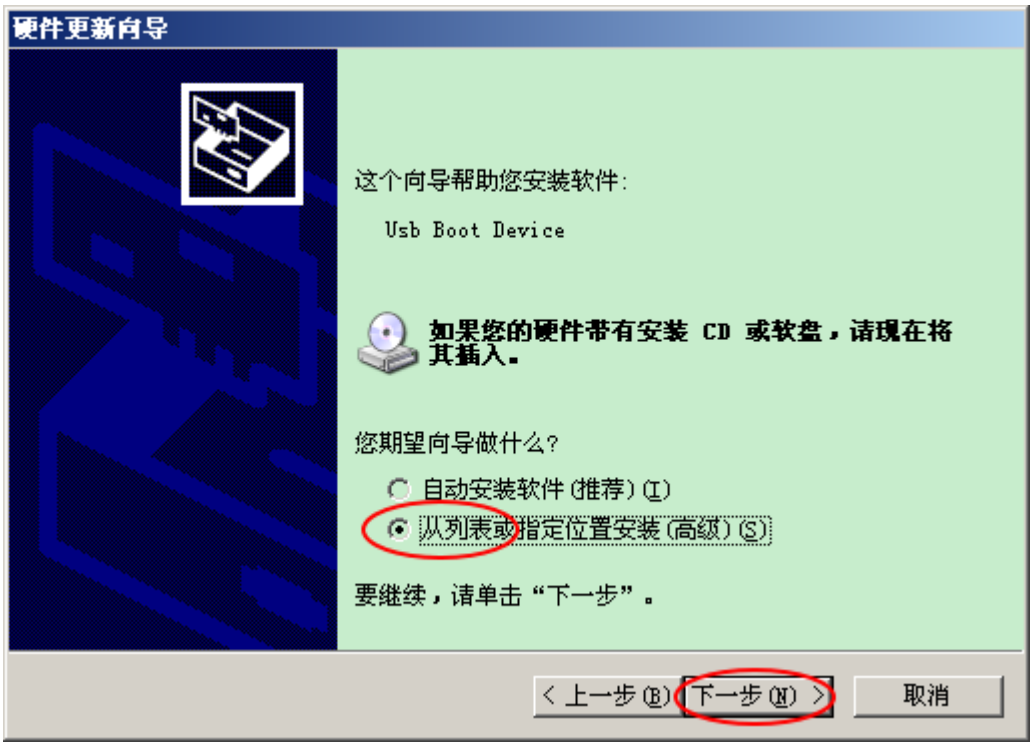

图 5-7

5) 勾选“在搜索中包括这个位置”，点击“浏览”指向“USBBurnTool_V1.3.2_20130626”；

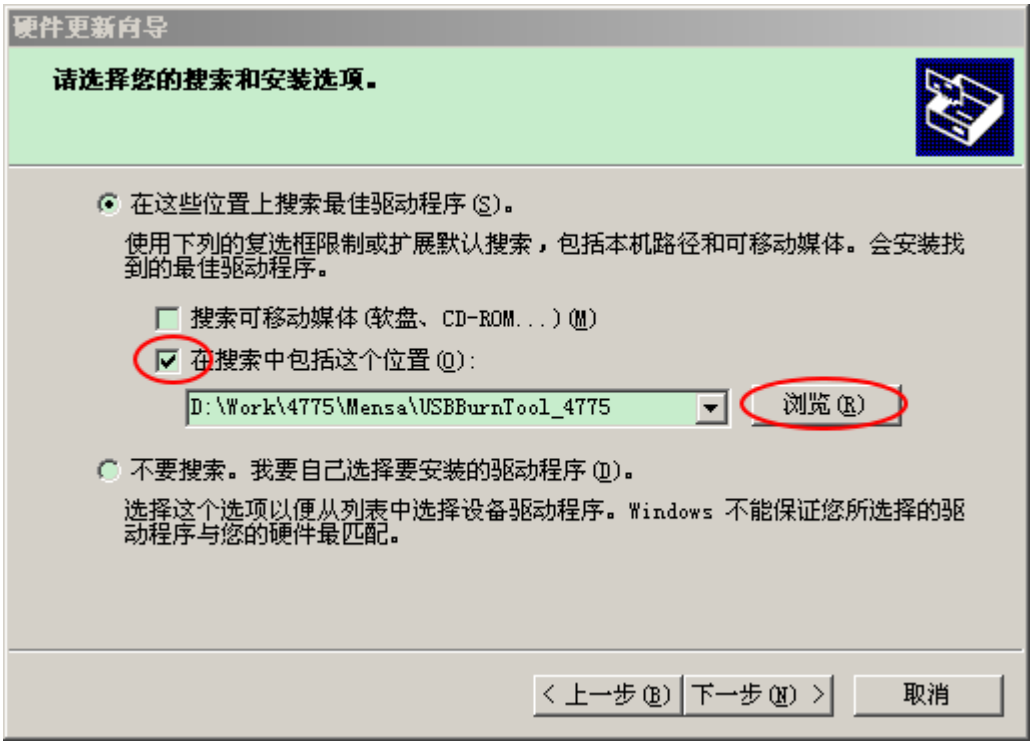

图 5-8

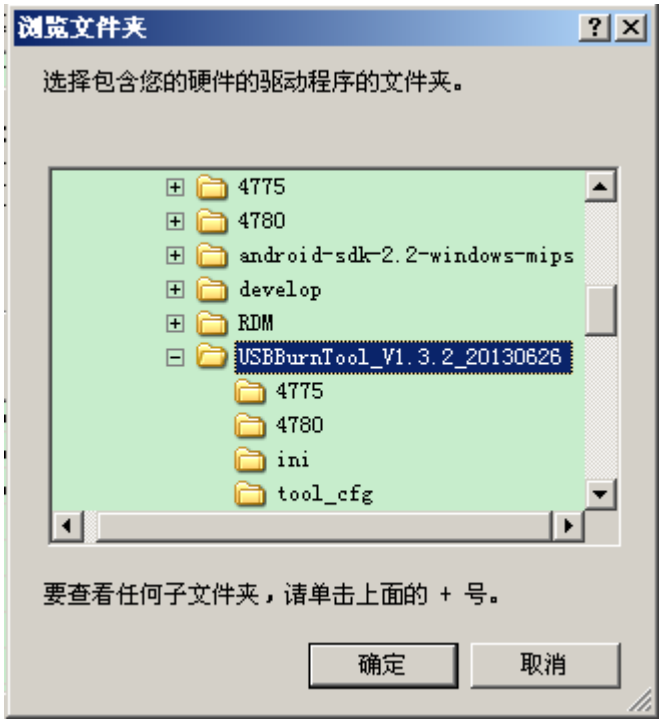

图 5-9

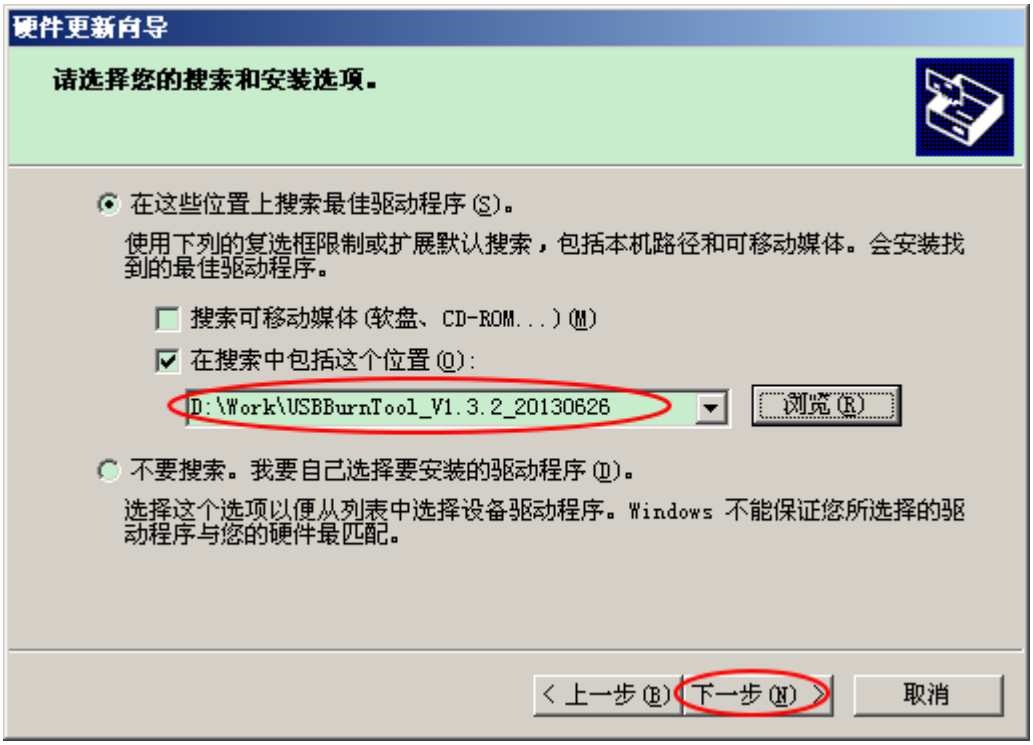

图 5-10

6) 选中 V1.3.2 版本烧录工具的文件夹路径，点击“下一步”;

图 5-11

图 5-12

操作至如图 5-12，驱动更新完毕。有时电脑上会提示需要“重启计算机”，按照提示操作即可。下面让我们来检查驱动更新是否真正成功：

鼠标右键“我的电脑”—>“管理”—>“设备管理器”—>“Ingenic Usb Boot Class”—>“Usb Boot Device”—>“属性”—>“驱动程序”；

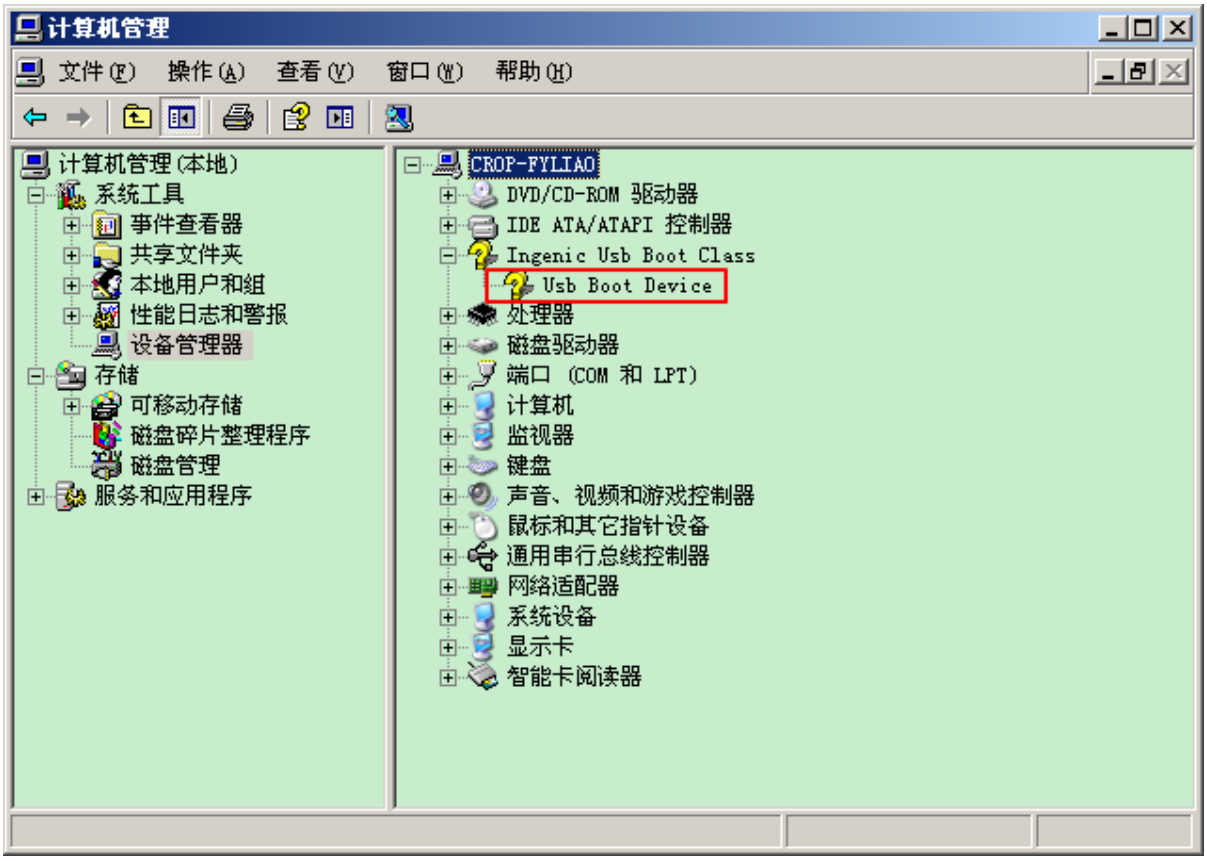

图 5-13

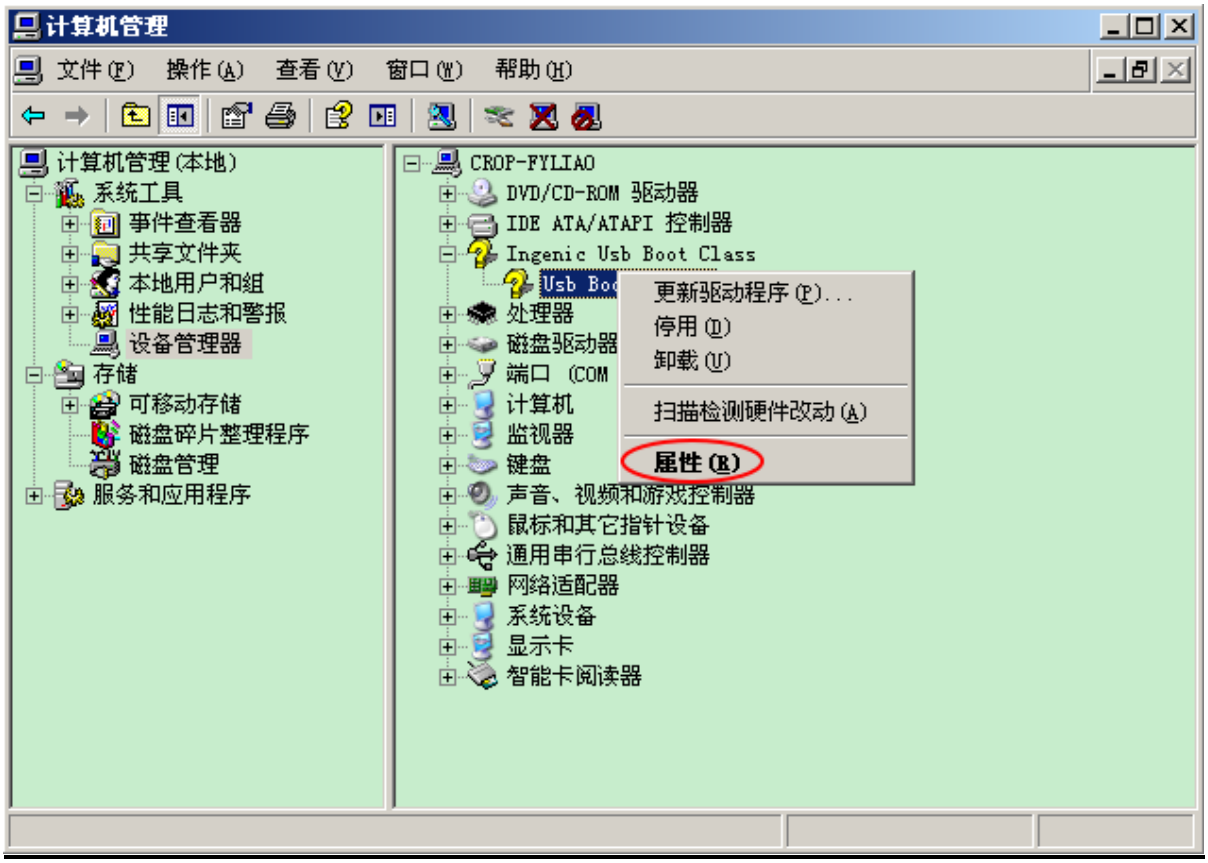

图 5-14

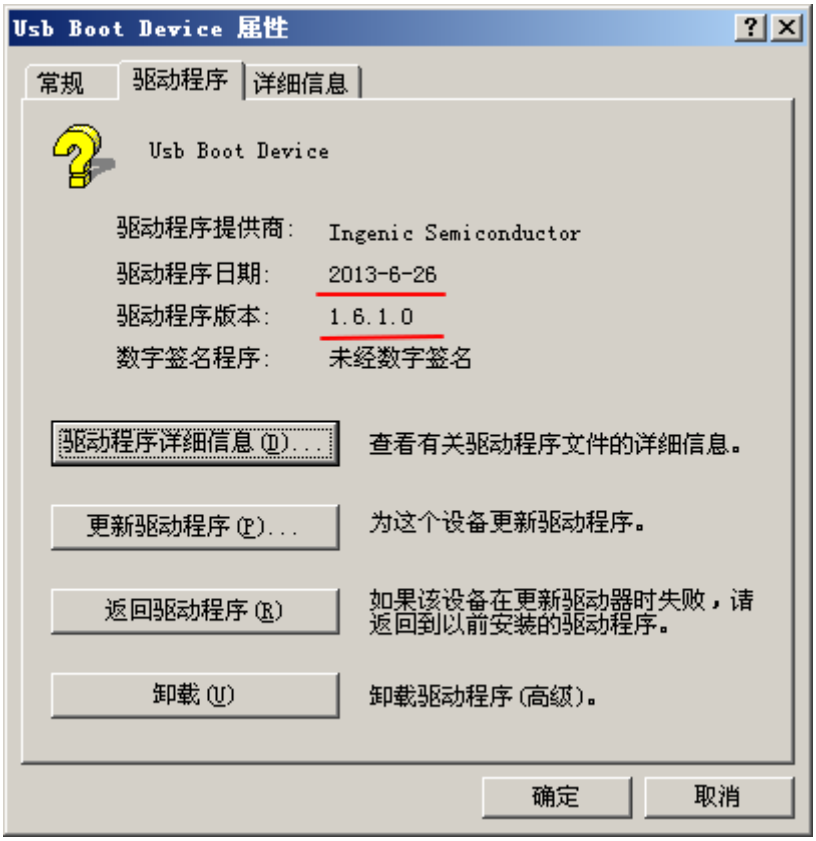

图 5-15

Reset 开发板后，即可进入下一阶段；

5.1.1.2 配置并烧录

1) 进入“USBBurnTool_V1.3.2_20130626”文件夹，双击“USBBurnTool.exe”打开烧录工具；

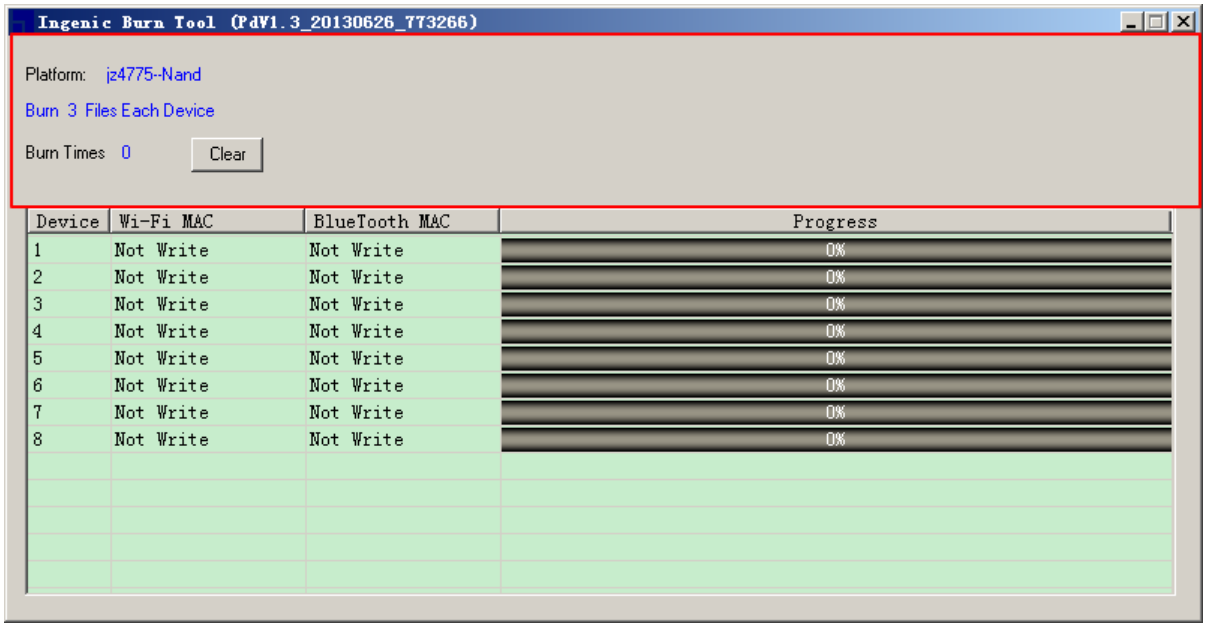

图 5-16

2) 鼠标右键单击弹出窗口“Platform”所在区域，在弹出菜单选项中选择“Config”选项，如图 5-17 所示窗口：

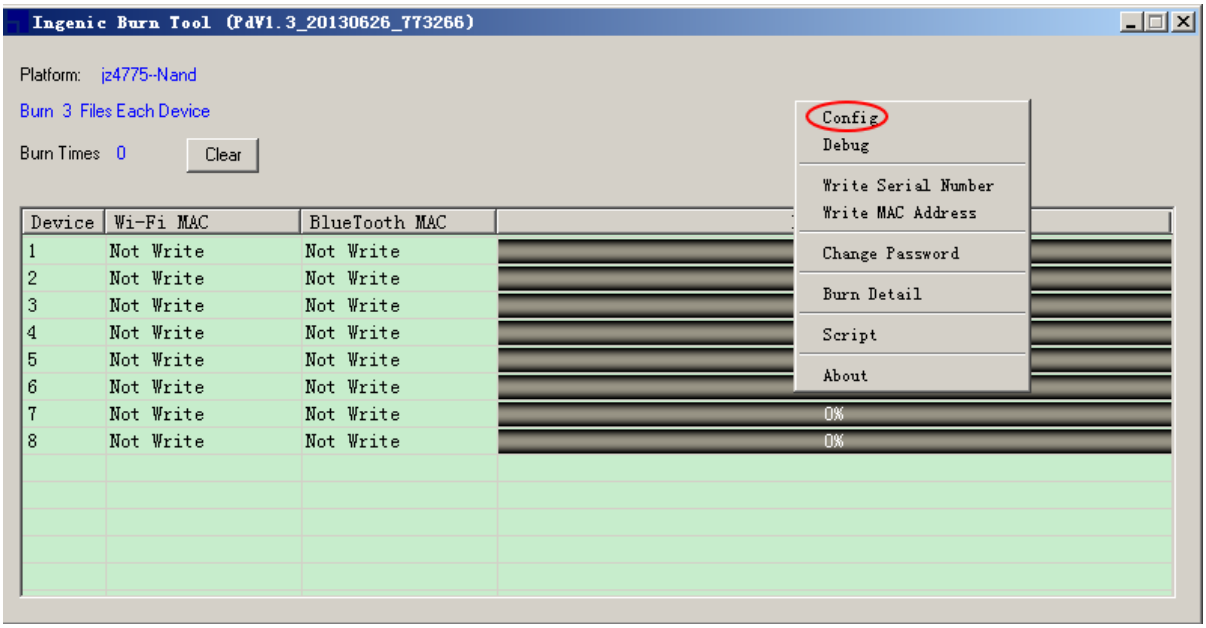

图 5-17

3) 在新弹出的窗口点击“Import Config”；

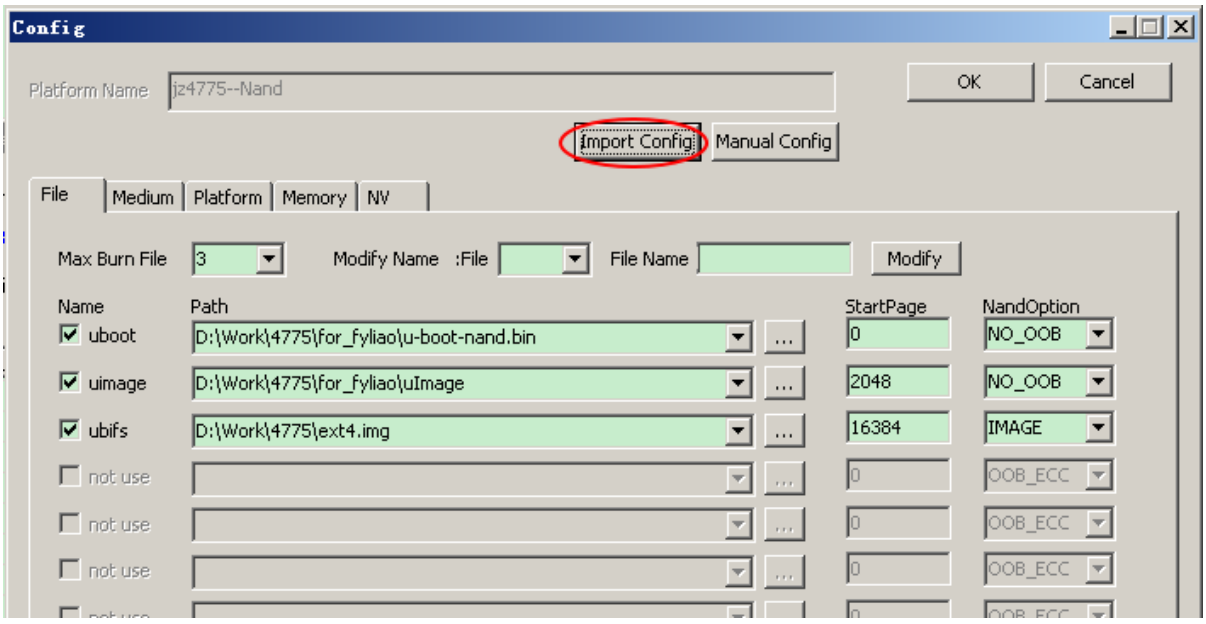

图 5-18

4) 在 ini 文件夹里双击选择“Jz4780-Nand.ini”；

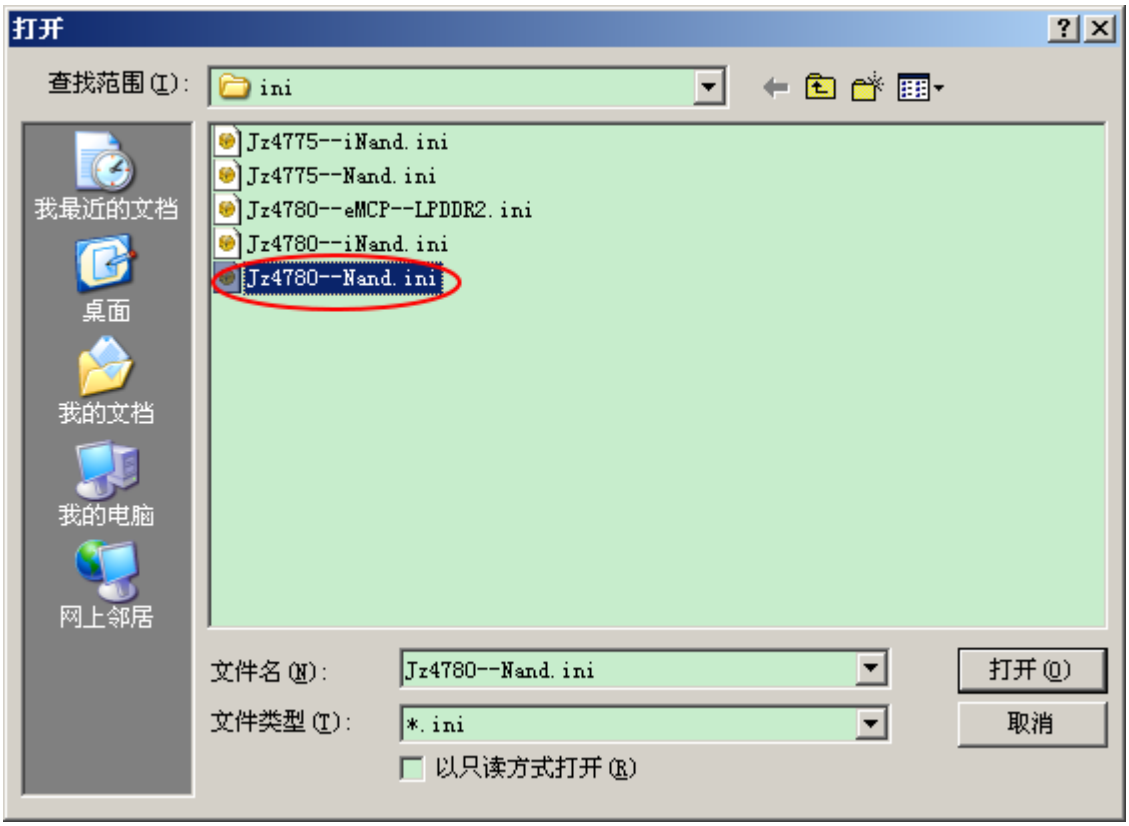

图 5-19

5) Platform Name 显示为“Jz4780-Nand”；

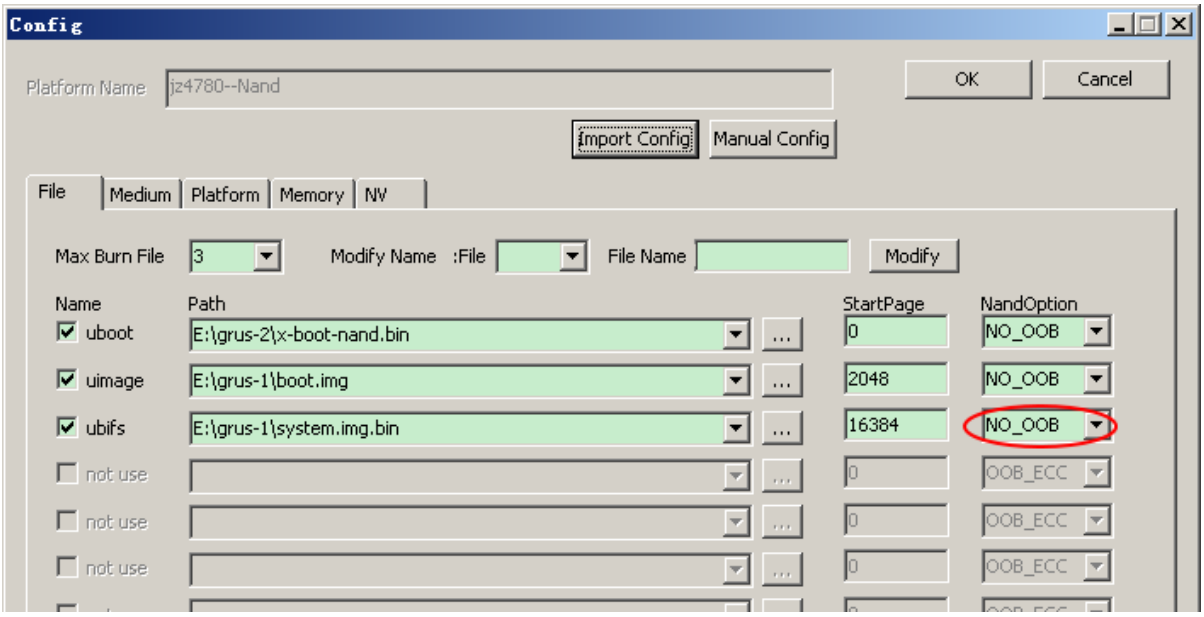

图 5-20

6) Ubifs 对应的 NandOption 在下拉菜单中选择“IMAGE”；

7) Ubifs 选择 system.img 文件；

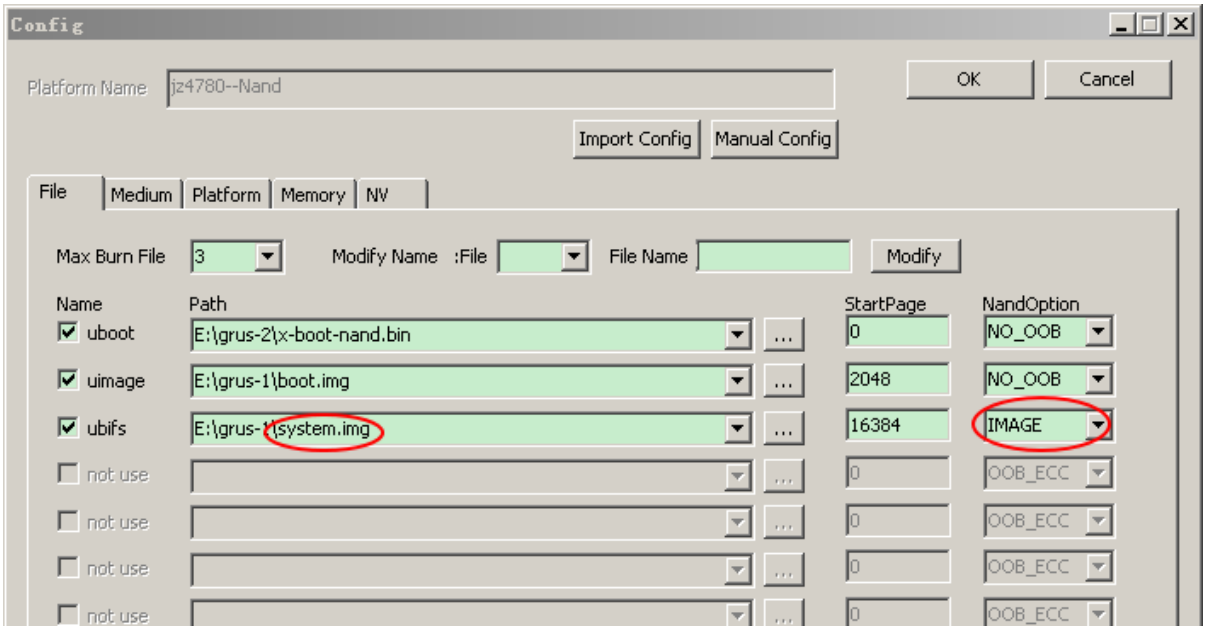

图 5-21

8) 确认 Medium—>System Partition Size 对应的大小是否 512MB

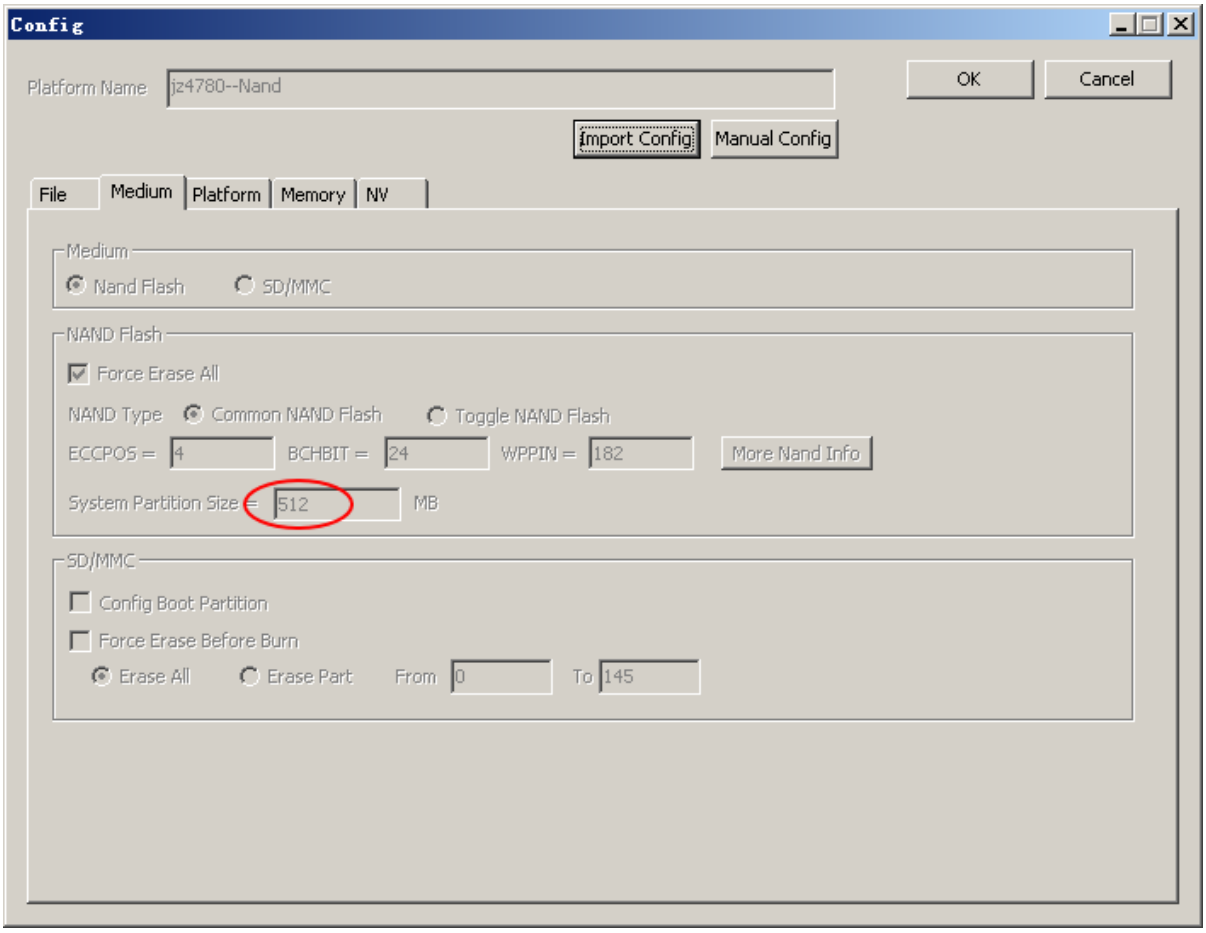

图 5-22

9) 如果答案为否，需要点击“Manual Config”，再对该值进行修改：

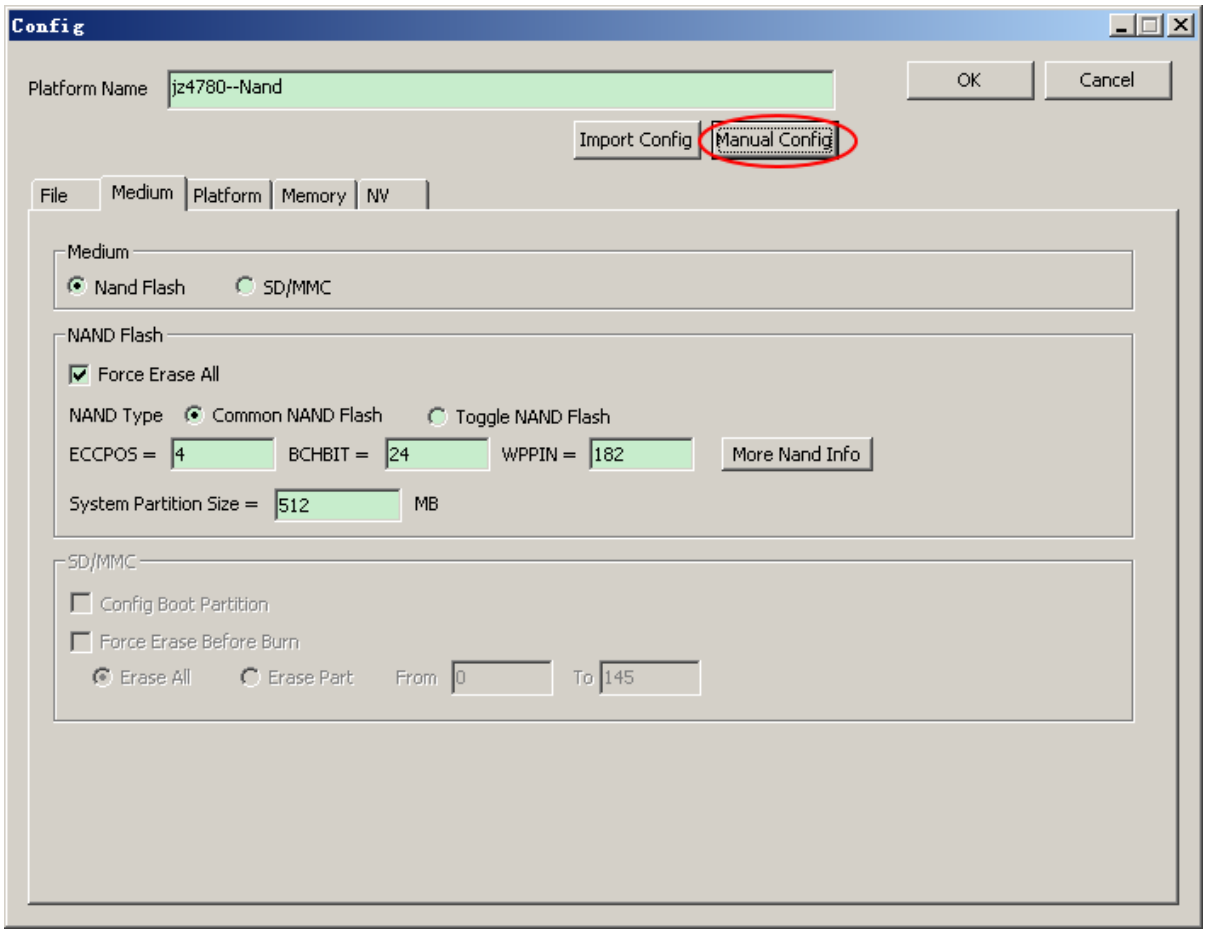

图 5-23

10) 确认各项配置以及镜像文件选择正确，点击“Cancel”退出到烧录界面；

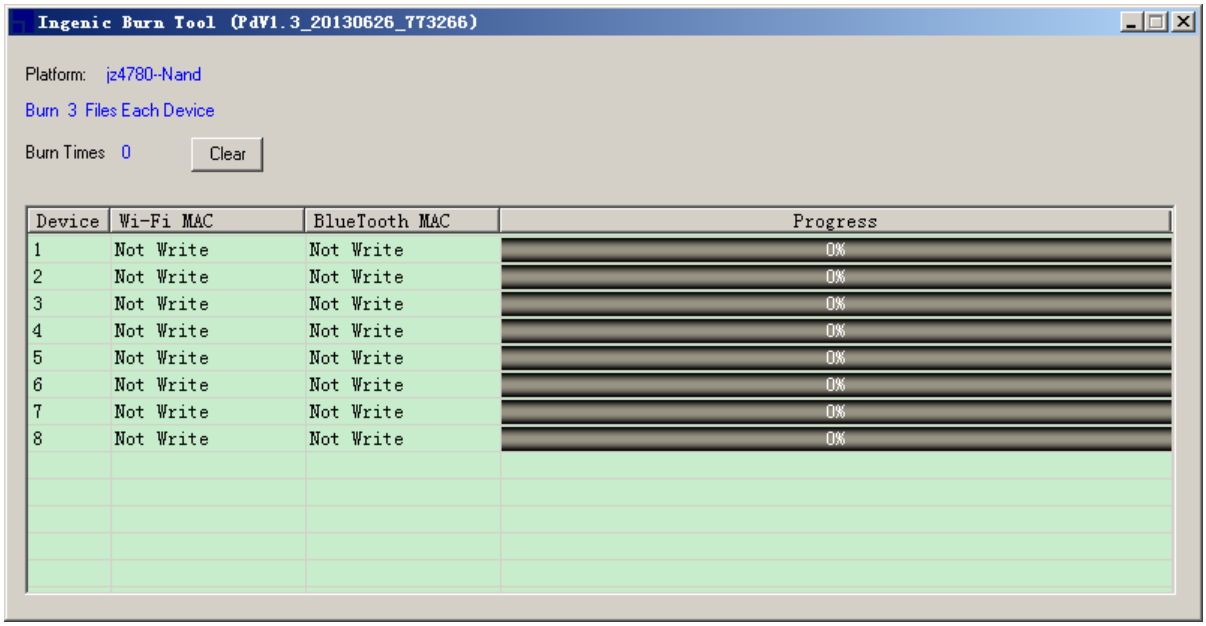

图 5-24

11) 按住“SW5”，再按“SW7”，进入烧录模式：

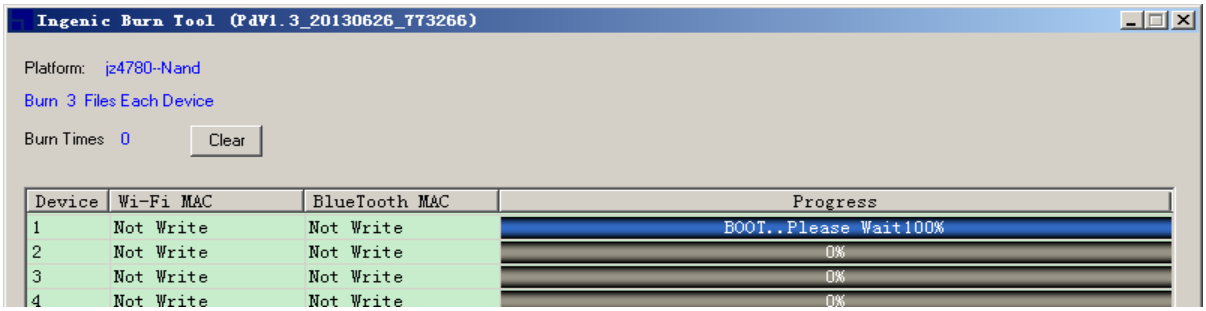

图 5-25

12) 显示“Successful”后，烧录完成；

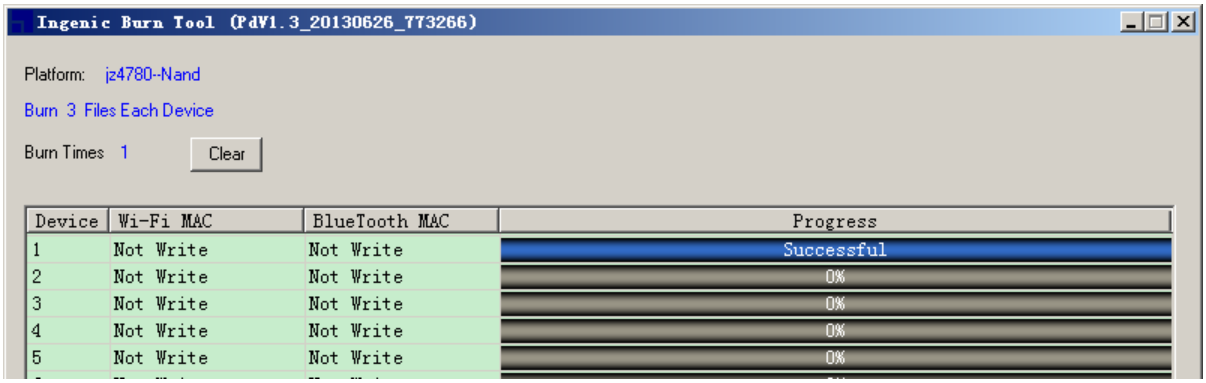

图 5-26

5.2 SD 卡烧录

5.2.1 制作一个可用的卡烧录工具

a. **准备一张 2G 大小的 SD 卡**

将这张 SD 卡插入读卡器，并将该读卡器播放 PC 端的 USB 插槽。

b. **格式化 SD 卡**

以 FAT32 格式化你的 SD 卡。

c. **修改配置文件 Configuration.txt**

修改配置文件 Configuration.txt，设置路径指向制作卡烧录工具的文件“2G-warrior-unmd5-burner.fw”，并保证起始地址为“0”。

例如：我将“2G-warrior-unmd5-burner.fw”放在 D 盘根目录，则配置文件的内容如图 5-25 所示。

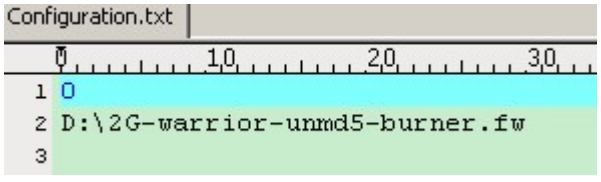

图 5-25

d. **打开烧录 SD 卡的工具**

打开 **SDBoot.exe**：

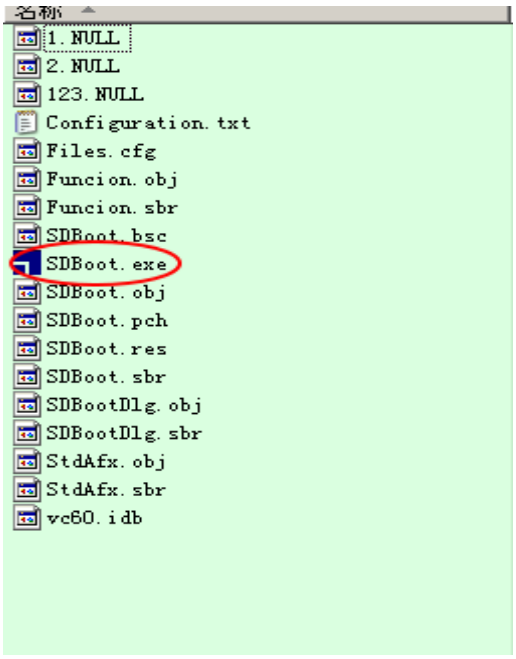

图 5-26

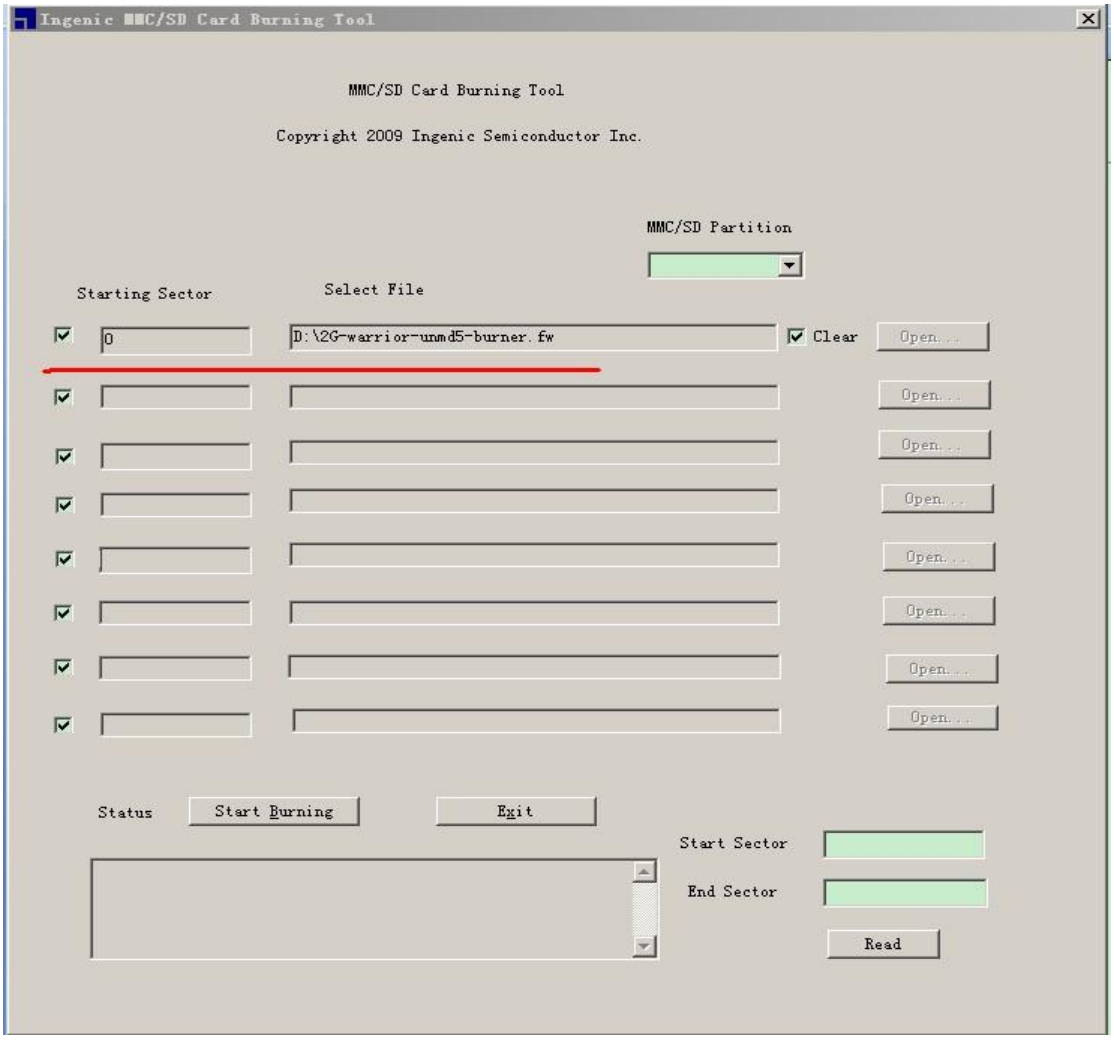

图 5-27

注意：如果你的 PC 使用的是 Windows Vista and Windows 7 操作系统，需要鼠标右键“**SDBoot.exe**”，选择“以管理员身份运行”后再打开该工具，方可操作成功。

e. **使“ MMC/SD Partition”的下拉盘符与您 SD 卡在 PC 上的盘符一致**

图 5-28

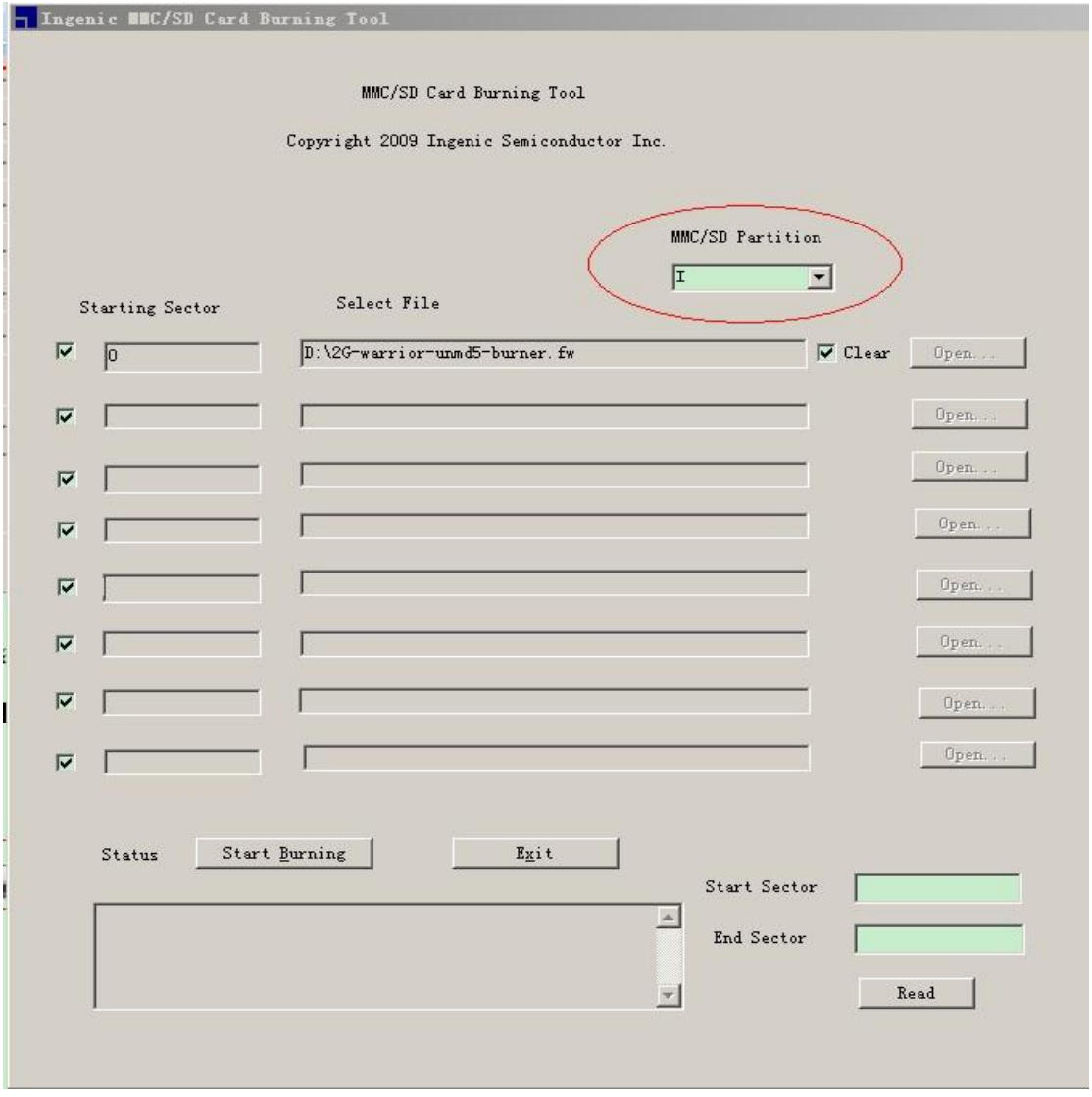

图 5-29

f. **开始烧录**

点击烧录工具界面上的“Start Burning”，您会看到如图 5-31 所示的打印。

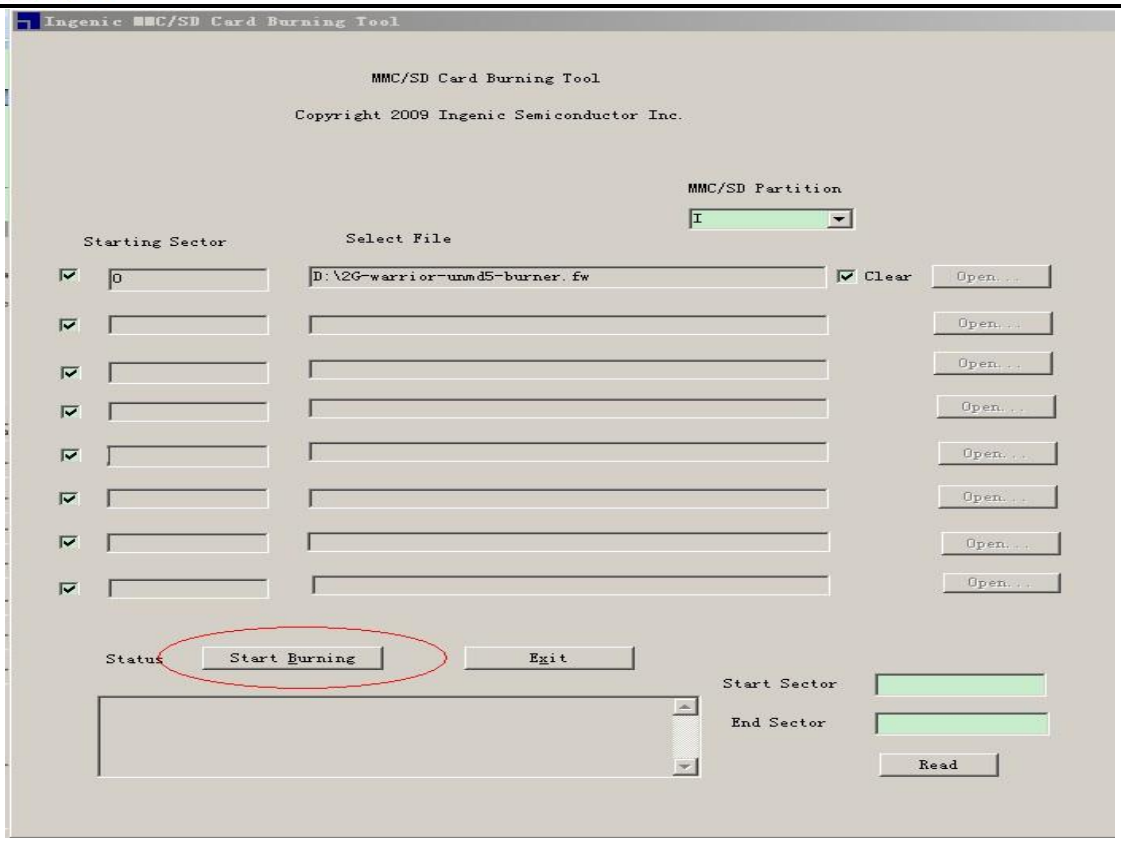

图 5-30

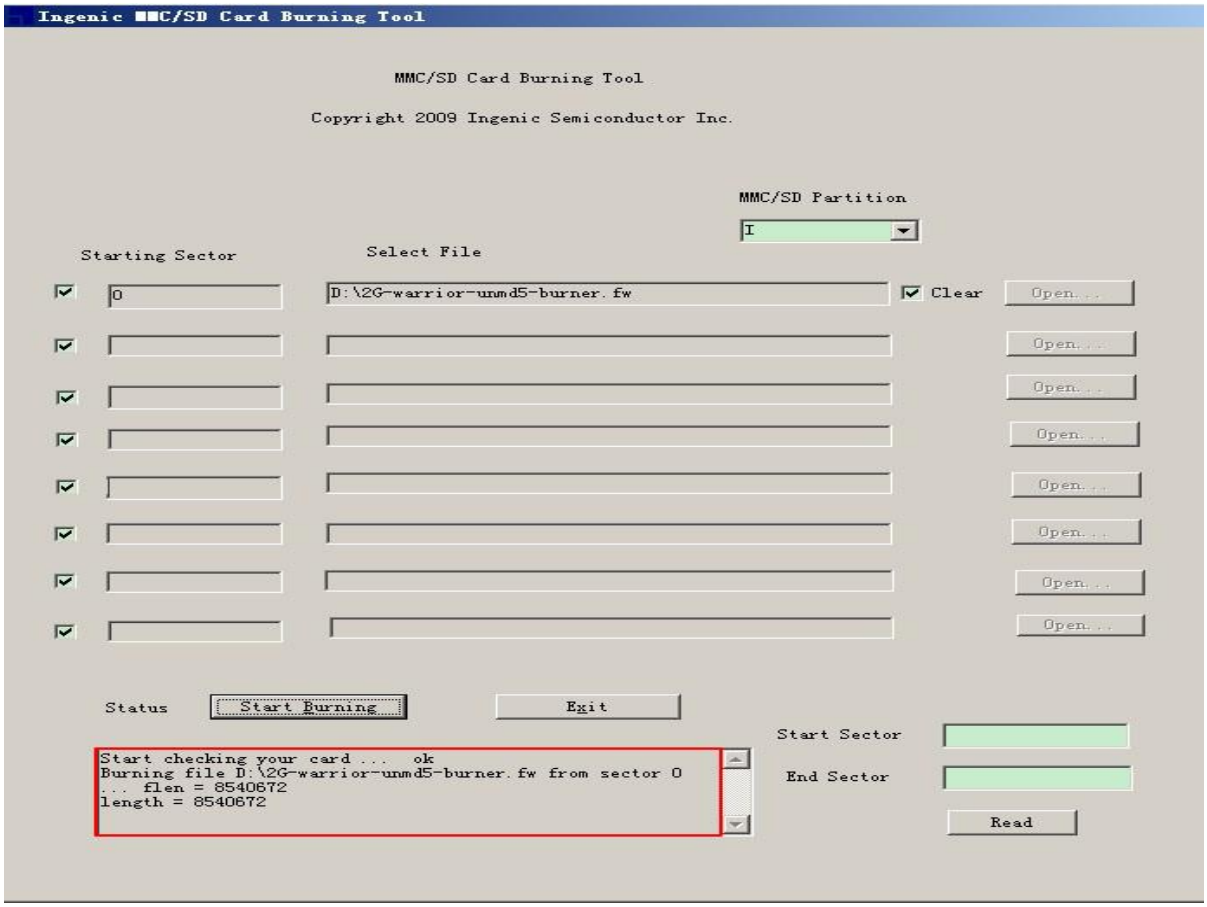

图 5-31

g. **烧录成功**

当烧录成功，会看到如图 5-32 所示的“Write Completed!”窗口。

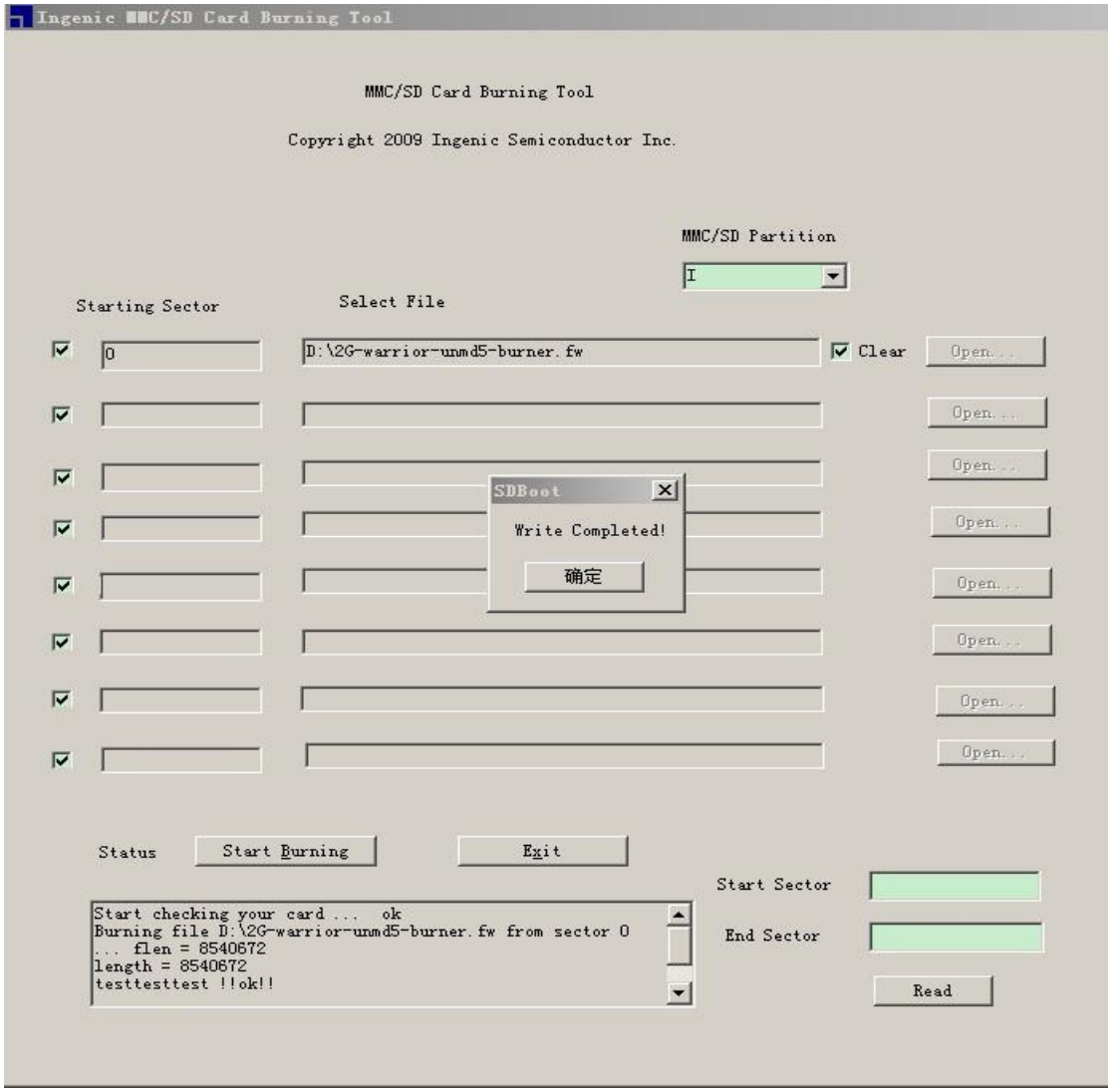

图 5-32

h. **从 PC 上弹出读卡器**

i. **快速格式化 SD 卡**

重新插入读卡器，进行“快速格式化”的操作，现在你读卡器上 2G 的 TF 卡，已经可以烧录 Warrior 板了。

5.2.2 拷贝需要的文件到 SD 卡

a. 配置文件

拷贝“burning_list.txt”“config.txt.”这两个配置文件到 SD 卡；

b. 烧录文件 Binary files

拷贝 3 个烧录文件到 SD 卡，分别为：x-boot-nand.bin、boot.img 和 system.img

5.2.3 烧录开发板

5.2.3.1 进入 SD 卡烧录模式（正常 boot 模式）

a. 接上 5V-3A 的电源；

b. 按住音量减（-）键，reset 一次；

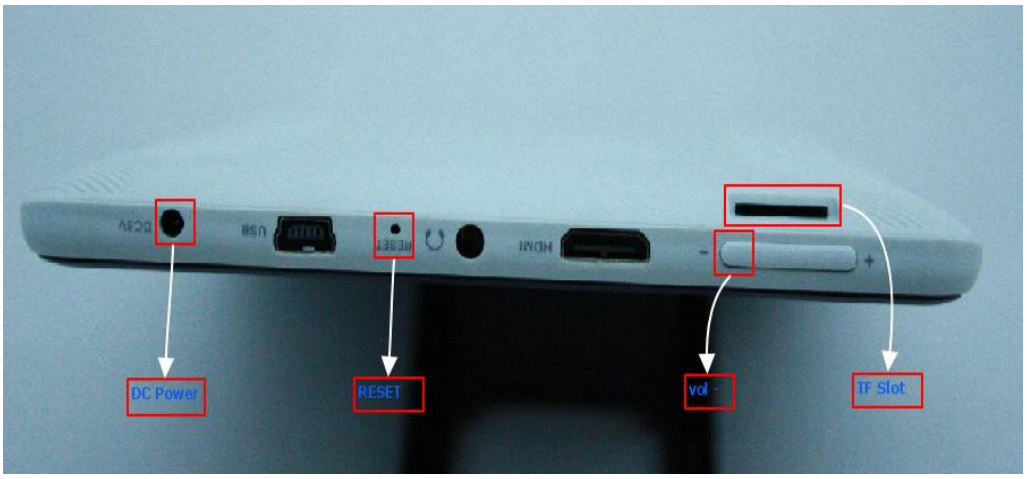

图 5-33

注意： 这里需要一直按住音量减键，直到 warrior 板屏幕上出现 boot 的 logo（绿色的 Android 机器人）

5.2.3.2 成功烧录

整个卡烧录过程中，屏幕上会有一些提示性打印。烧录成功后，warrior 板自动启动。

图 5-34

1 板级配置板级配置

进入 Android 工程目录：user@ubuntu:~/android-41$ ls

abi	dalvik	frameworks	libnativehelper	packages	system
bionic	development	gdk	Makefile	pdk	update
bootable	device	hardware	nda	prebuilt	vendor
build	docs	kernel	ndk	prebuilts	
cts	external	libcore	sdk		

这里分为三部分：

i) Bootloader
ii) Kernel
iii) Android system.

在后面我们将详细介绍如何配置你的产品环境。

1.1 配置 Bootloader

Bootloader 的源码位于 Android 工程的 bootable/bootloader/xboot，

./Makefile	编译信息和配置相关信息，例如：grus_nand_config
./include/configs/grus.h;	板极设置信息：memory, nandflash, CPU frequency, UART, binary, GPIO 等
/boot/board/Makefile	板级的 Makefile/* Makefile of board */
./boot/board/grus/jz4780_grus_special.c	开发板的一些特殊代码，例如控制 LCD 等
./boot/board/grus/mbr.h	MBR 信息，仅针对 emmc-nand 或者是 inand。

1.2 板级 kernel 配置

Linux kernel 的源码位于 Android 工程项目的 kernel 下：android-41/kernel

./arch/mips/configs/grus_release_defconfig	默认的是 grus
./arch/mips/xburst/soc-4780/board/grus	配置板级的目录
./arch/mips/xburst/soc-4780/board/grus/grus-nand.c	
./arch/mips/xburst/soc-4780/board/grus/grus-regulator.c	

6 板级配置

进入 Android 工程目录：user@ubuntu:~/android-41$ ls

abi	dalvik	frameworks	libnativehelper	packages	system
bionic	development	gdk	Makefile	pdk	update
bootable	device	hardware	nda	prebuilt	vendor
build	docs	kernel	ndk	prebuilts	
cts	external	libcore	sdk		

这里分为三部分：

i) Bootloader
ii) Kernel
iii) Android system.

在后面我们将详细介绍如何配置你的产品环境。

6.1 配置 Bootloader

Bootloader 的源码位于 Android 工程的 bootable/bootloader/xboot，

./Makefile	编译信息和配置相关信息，例如：grus_nand_config

./include/configs/grus.h;	板极设置信息：memory, nandflash, CPU frequency, UART, binary, GPIO 等
/boot/board/Makefile	板级的 Makefile/* Makefile of board */
./boot/board/grus/jz4780_grus_special.c	开发板的一些特殊代码，例如控制 LCD 等
./boot/board/grus/mbr.h	MBR 信息，仅针对 emmc-nand 或者是 inand。

6.2 板级 kernel 配置

Linux kernel 的源码位于 Android 工程项目的 kernel 下：android-41/kernel

./arch/mips/configs/grus_release_defconfig	默认的是 grus
./arch/mips/xburst/soc-4780/board/grus	配置板级的目录
./arch/mips/xburst/soc-4780/board/grus/grus-nand.c	
./arch/mips/xburst/soc-4780/board/grus/grus-regulator.c	
./arch/mips/xburst/soc-4780/board/grus/grus-pm.c	
./arch/mips/xburst/soc-4780/board/grus/grus-i2c.c	
./arch/mips/xburst/soc-4780/board/grus/grus-lcd.c	
./arch/mips/xburst/soc-4780/board/grus/grus-sound.c	
./arch/mips/xburst/soc-4780/board/grus/grus-mmc.c	
./arch/mips/xburst/soc-4780/board/grus/grus.h	
./arch/mips/xburst/soc-4780/board/grus/grus-misc.c	

6.3 Android system 的配置

./build/target/product/security/grus.pk8	升级的产品密钥
./build/target/product/security/grus.pem	
./build/target/product/security/grus.x509.pem	
./build/target/board/grus	系统的主要配置

3 总结板级配置板级配置

进入 Android 工程目录：user@ubuntu:~/android-41$ ls

abi	dalvik	frameworks	libnativehelper	packages	system
bionic	development	gdk	Makefile	pdk	update
bootable	device	hardware	nda	prebuilt	vendor
build	docs	kernel	ndk	prebuilts	
cts	external	libcore	sdk		

这里分为三部分：

iv) Bootloader
v) Kernel
vi) Android system.

在后面我们将详细介绍如何配置你的产品环境。

3.1 配置 Bootloader

Bootloader 的源码位于 Android 工程的 bootable/bootloader/xboot，

./Makefile	编译信息和配置相关信息，例如：grus_nand_config
./include/configs/grus.h;	板极设置信息：memory, nandflash, CPU frequency, UART, binary, GPIO 等
/boot/board/Makefile	板级的 Makefile/* Makefile of board */
./boot/board/grus/jz4780_grus_special.c	开发板的一些特殊代码，例如控制 LCD 等
./boot/board/grus/mbr.h	MBR 信息，仅针对 emmc-nand 或者是 inand。

3.2 板级 kernel 配置

Linux kernel 的源码位于 Android 工程项目的 kernel 下：android-41/kernel

./arch/mips/configs/grus_release_defconfig	默认的是 grus
./arch/mips/xburst/soc-4780/board/grus	配置板级的目录
./arch/mips/xburst/soc-4780/board/grus/grus-nand.c	
./arch/mips/xburst/soc-4780/board/grus/grus-regulator.c	

7 总结

Android 开发环境的搭建大同小异，基本上与 Android 官方网站介绍的一致，区别主要在于：

a) 君正的 Android 开发板，同样 64 位的 Linux 系统，需要额外安装兼容 32 位的相应插件；
b) 需要君正提供的交叉编译工具；
c) 君正提供的对应的烧录工具；