
Installation Guide for MV6410 (Linux 2.6)



MicroVision.,Cd.Ltd

Document Information

Version	1.0
File Name	Mv6410 Linux 2.6 Guide.doc
Date	2009.1.30.
Satus	Working

Revision History

Date	Version	Update Descriptions	Editor
2009.1.30.	V1.0	First Edition	Jongill Wee

MV6410-LCD

Author: MicroVision. Co., Ltd.

Publisher: MicroVision. Co., Ltd.

Tel: +82-2-3283-0101

Fax: +82-2-3283-0160

E-mail: sale@microvision.co.kr

Homepage: www.microvision.co.kr, www.mvtool.co.kr

#610 Hanshin IT Tower 235, Guro3-dong, Guro-gu, Seoul, Korea

MicroVision. Co., Ltd.

Copyright ©1991 MicroVision Co., Ltd. ,

Contents.....

- 1. Outline 4/39
- 2. Spec. 5/39
 - 2.1. An Arrangement Plan 5/39
 - 2.2. Packages 6/39
 - 2.3. H/W Lists 7/39
- 3. Doing Configuration Boot of mode 8/39
- 4. Doing Configuration of Environment 11/39
- 5. Installing Toolchain 12/39
- 6. Setting Up TFTP Server 14/39
- 7. u-boot 15/39
- 8. Kernel Compilation 18/39
- 9. Root file System 22/39
- 10. Setting for downloading Host PC to Board 23/39
- 11. Downloading 24/39
- 12. Application (Camera, USB WIFI, TV OUT, S-Video) 36/39

1. Outline

The MV6410 is a 16/32-bit RISC microprocessor, which is designed to provide a cost-effective, low-power capabilities, high performance Application Processor solution for mobile phones and general applications. To provide optimized H/W performance for the 2.5G & 3G communication services, the MV6410 adopts 64/32-bit internal bus architecture. The 64/32-bit internal bus architecture is composed of AXI, AHB and APB buses. It also includes many powerful hardware accelerators for tasks such as motion video processing, audio processing, 2D graphics, display manipulation and scaling. An integrated Multi Format Codec (MFC) supports encoding and decoding of MPEG4/H.263/H.264 and decoding of VC1. This H/W Encoder/Decoder supports real-time video conferencing and TV out for both NTSC and PAL mode. Graphic 3D (hereinafter 3D Engine) is a 3D Graphics Hardware Accelerator which can accelerate OpenGL ES 1.1 & 2.0 rendering. This 3D Engine includes two programmable shaders: one vertex shader and one pixel shader.

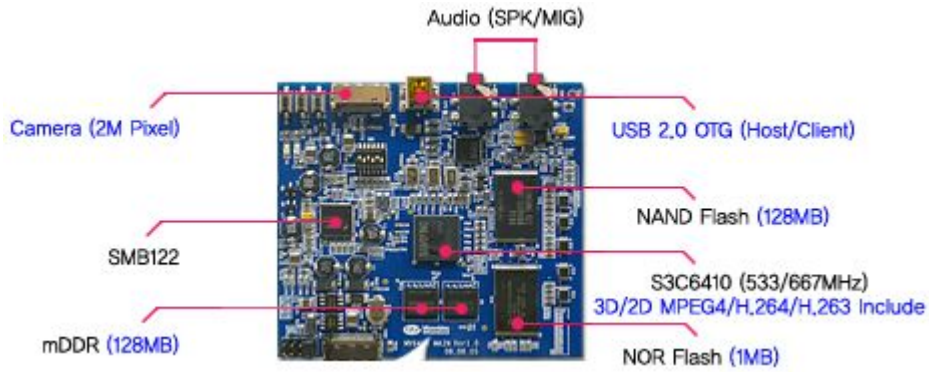


The MV6410 has an optimized interface to external memory. This optimized interface to external memory is capable of sustaining the high memory bandwidths required in high-end communication services. The memory system has dual external memory ports, DRAM and Flash/ROM. The DRAM port can be configured to support mobile DDR, DDR, mobile SDRAM and SDRAM. The Flash/ROM port supports NOR-Flash, NAND-Flash, OneNAND, CF and ROM type external memory.

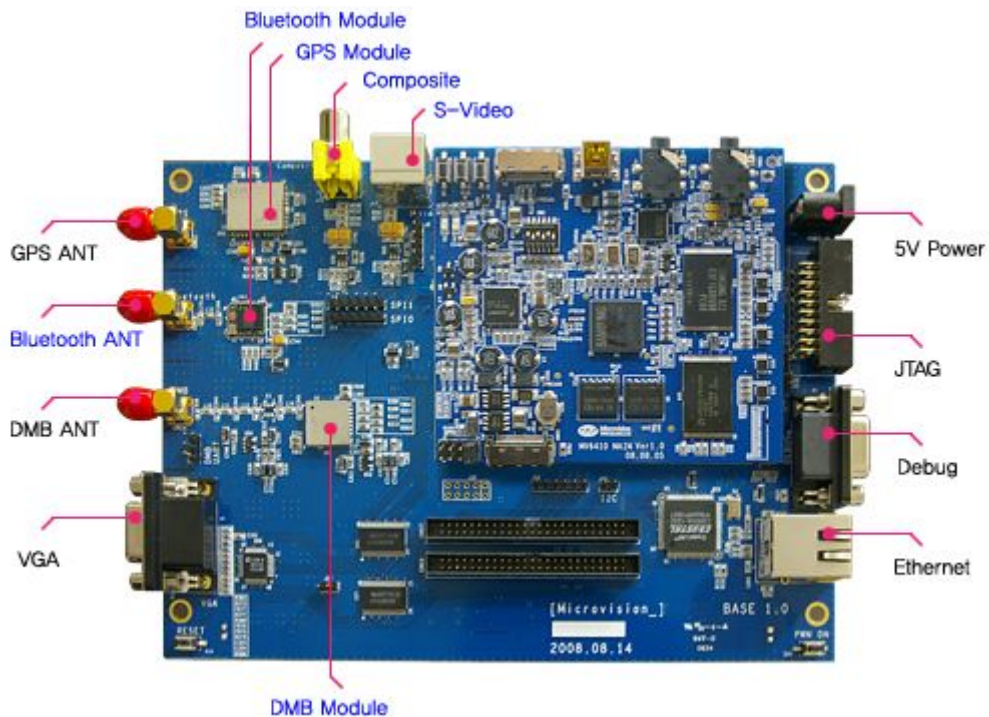
To reduce total system cost and enhance overall functionality, the MV6410 includes many hardware peripherals such as a Camera Interface, TFT 16-bit true color LCD controller, System Manager (power management & etc.), 4-channel UART, 32-channel DMA, 5-channel 32bit Timers with 2PWM output, General Purpose I/O Ports, I2S-Bus interface, I2C-BUS interface, USB Host, USB OTG Device operating at high speed (480Mbps), 3-channel SD/MMC Host Controller and PLLs for clock generation. The ARM subsystem is based on the ARM1176JZF-S core. It includes separate 16KB Instruction and 16KB data caches, 16KB Instruction and 16KB Data TCM. It also includes a full MMU to handle virtual memory management. The ARM1176JZF-S is a single chip MCU, which includes support for JAVA acceleration. The ARM1176JZF-S includes a dedicated vector floating point coprocessor allowing efficient implementation of various encryption schemes as well as high quality 3D graphics applications. The MV6410 adopts the de-facto standard AMBA bus architecture. These powerful, industry standard features allow the MV6410 to support many of the industry standard Operating Systems.

2. Spec.

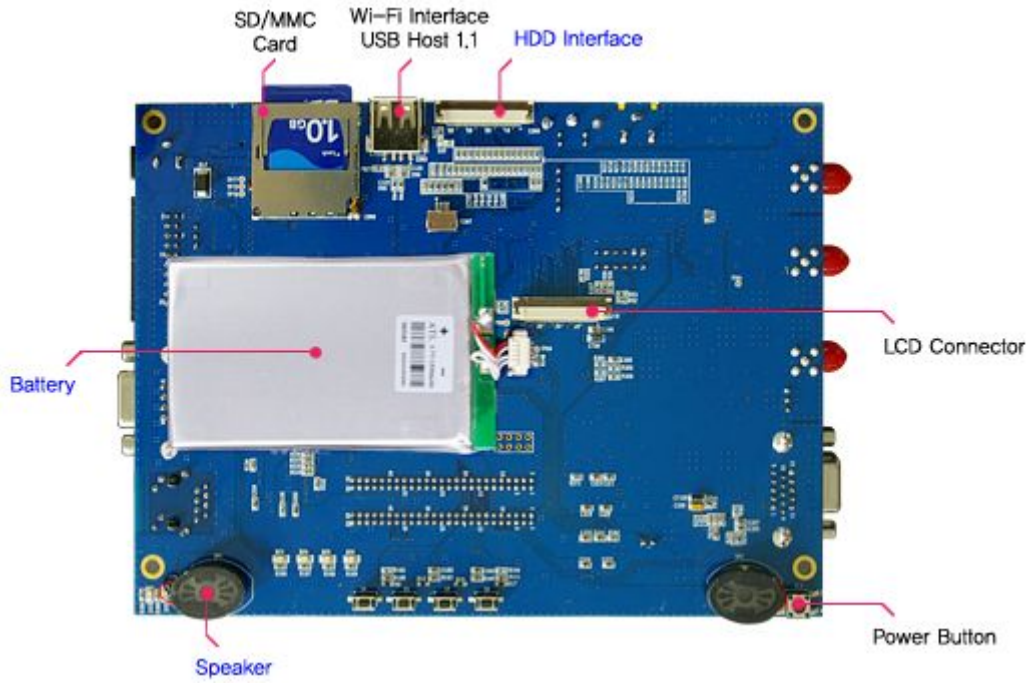
2.1 An Arrangement Plan



- MAIN BOARD -



- BASE BOARD, Front -



- BASE BOARD, Back -

2.2. Packages

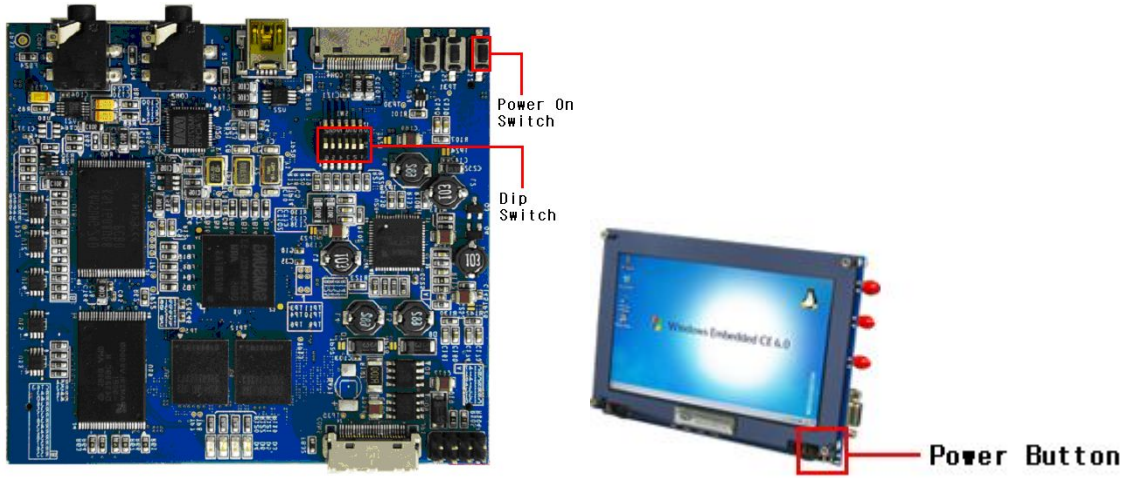


- Package Lists -

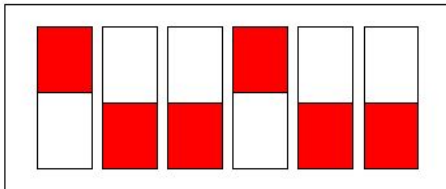
2.3. H/W Lists

ITEM	Spec.	Description
CPU	S3C6410X	ARM1176JZF-S CORE (533/667MHz) UFP / SIMD
SDRAM	SAMSUNG mDDR	128MByte, 32bit access, clock speed 266MHz
NAND FLASH	SAMSUNG NAND	LARGE BLOCK 1Gb (128MByte)
NOR FLASH	AMD	NORFLASH 8Mb (1MByte), 16Bit (LIMIT 64Kbit)
AUDIO CODEC	WM9713	STEREO 400mW, MIC IN, HEAD SET
GRAPHIC	3D/2D	OpenGL 3D / 2D GRAPHIC CONTROLLER
USB	2PORT	USB 2.0 OTG (HOST, CLIENT), USB HOST 1.1
UART	4PORT	UART0: DEBUG, UART1: Bluetooth, UART2: GPS, UART3: RXD, TXD, GND
VIDEO, (TV-OUT)	2PORT	COMPOSITE
DMB	GAON DMB	GAON GDM-801, 1Ch SoC DAB/DMB Module, RF + BB : T3500
CAMERA	MICRON 2M PIXEL	AUTO FOCUS , PREVIEW & SNAP SHOT
ETHERNET	CS8900	10 BASE-T, LINK LED
SD/MMC	1PORT	SD Host & High Speed Multi-Media Card Interface
HDD	1PORT	FPC/FFC Connector Interface
BLUETOOTH	UART1	BLUETOOTH TECHNOLOGY BASE ON CSR BC04-ROM
GPS	GAON GPS MODULE	GAON GGM-150E, SiEF Star III, GPS Engine Without Antenna Engine Type, Size 15X13X27
LCD	7" WVGA	800X480 16bit (Data enable mode)
Battery (Option)	1530mAH @3.7V	FULL OPERATION 1 HOUR
CHARGER,PMIC	SMB122	AC & BATTERY SWITCHING 기능. PROTECTION, AC PRESENT, CC-CV

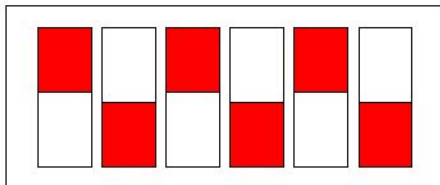
3. Doing Configuration Boot of mode.



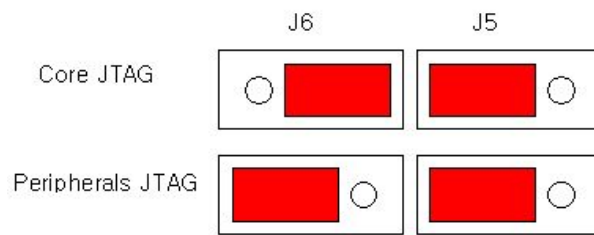
▶ NAND Flash Boot Mode, DIP Switch



▶ NOR Flash Boot Mode, DIP Switch

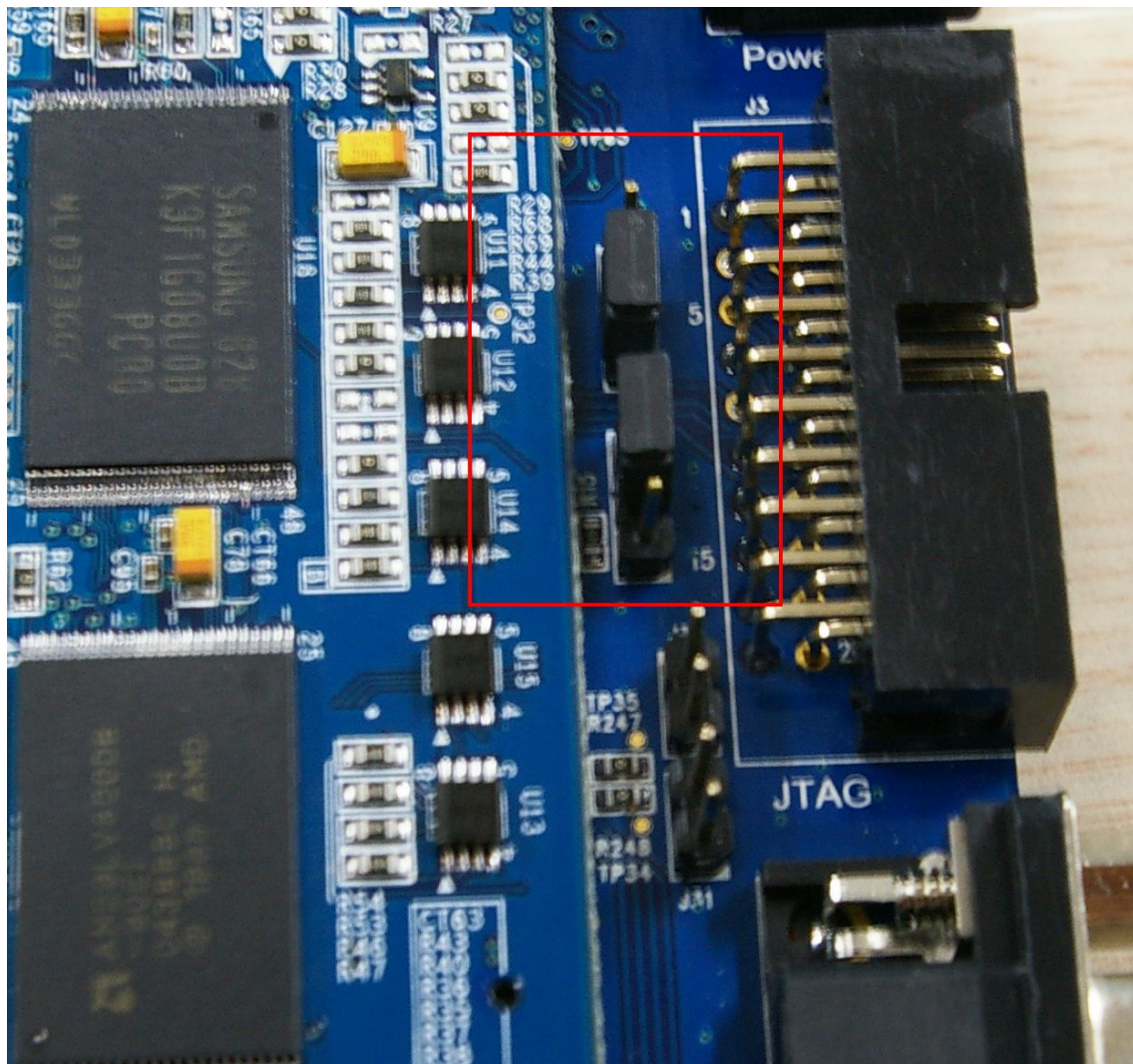


► JUMP Mode

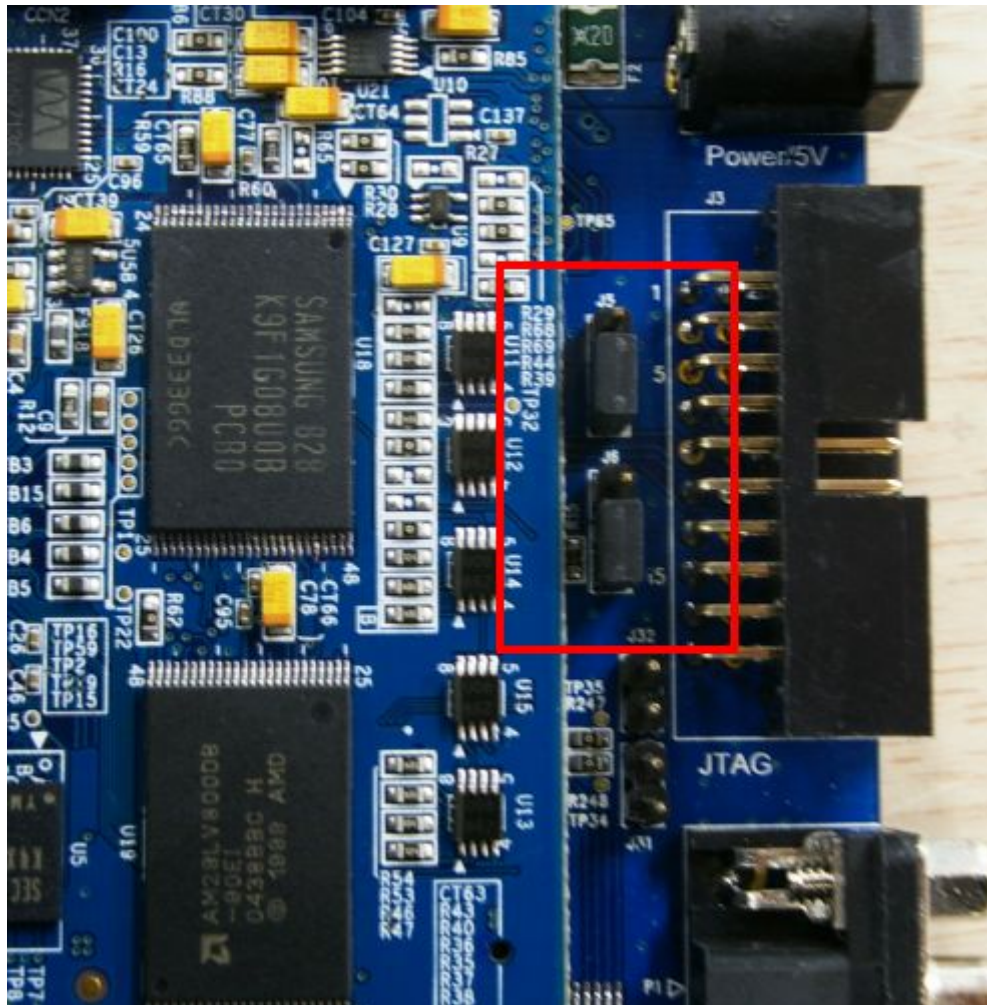


To Do Configuration JTAG

When you use JTAG Emulator.



When you use JTAG Dongle.



4. Doing Configuration of Environment

Overview

Group	File name	Description
Gcc	cross-4.3.1-eabi-armv6-mv20081010.tar	Toolchain Compilation
Source	s3c-u-boot-1.1.6.tar	Boot-loader
	s3c-linux-2.6.21.tar	Kernel
Image	u-boot.bin	Boot-loader Image
	zImage	Kernel Image
	rootfs_mv6410.cramfs	Root filesystem with Qtopia - gcc 2.2.0

After made folder `mv6410` through the command of `mkdir`, Copy it to the working directory `/mv6410`

```
[root@localhost mv6410]# ls
rootfs_mv6410.cramfs
s3c-linux-2.6.21.tar.gz
s3c-u-boot-1.1.6.tar
4.3.1-eabi-armv6-mv20081010.tar.gz
[root@localhost mv6410]#
```

5. Installing Toolchain

First, Copy in CD Sources\Linux\toolchain “4.3.1-eabi-armv6-mv20081010.tar.gz” to in your linux of Host PC `/usr/local/arm` before making `/usr/local/arm` through the command ‘`mkdir`’.

Follow the commands

```
[root@localhost mv6410]# mkdir -p /usr/local/arm <- Making folder 'arm'
[root@localhost mv6410]# tar xvf 4.3.1-eabi-armv6-mv20081010.tar
[root@localhost mv6410]# mv 4.3.1 /usr/local/arm/
[root@localhost mv6410]# export PATH=$PATH:/usr/local/arm/ 4.3.1-eabi-
armv6/usr/bin/arm-linux-
```

For use GCC before You have to set up “`bash_profile`”.

Please make sure! “`bash_profile`”

Follow the commands

```
[root@localhost mv6410]# vi ~/.bash_profile
PATH=$PATH:$HOME/bin:/usr/local/arm/4.3.1-eabi-armv6/usr/bin/
LD_LIBRARY_PATH=/usr/local/arm/4.3.1-eabi-armv6/gmp/lib:/usr/local/arm/4.3.1-eabi-armv6/mpfr/lib
export PATH LD_LIBRARY_PATH
unset USERNAME
LANG=en
```

My linux of system environment.

```
PATH=$PATH:$HOME/bin:/usr/local/arm/4.3.1-eabi-armv6/usr/bin
LD_LIBRARY_PATH=/usr/local/arm/4.3.1-eabi-armv6/gmp/lib:/usr/local/arm/4.3.1-eabi-armv6/mpfr/lib
export PATH LD_LIBRARY_PATH
unset USERNAME
LANG=en
```

Thereafter version 4.3.x . You have to setup your PATH of environment both gmp and mpfr.

For example :

```
LD_LIBRARY_PATH=/usr/local/arm/4.3.1-eabi-armv6/gmp/lib:/usr/local/arm/4.3.1-eabi-armv6/mpfr/lib
export PATH LD_LIBRARY_PATH
```

Warning! Don't modify a name "LD_LIBRARY_PATH"

After exit, You also have to apply to command a **"source ~/.bash_profile"** then You can use GCC 4.3.1.

Follow the commands

```
[root@localhost mv6410]# source ~/.bash_profile
```

6. Setting Up TFTP Server

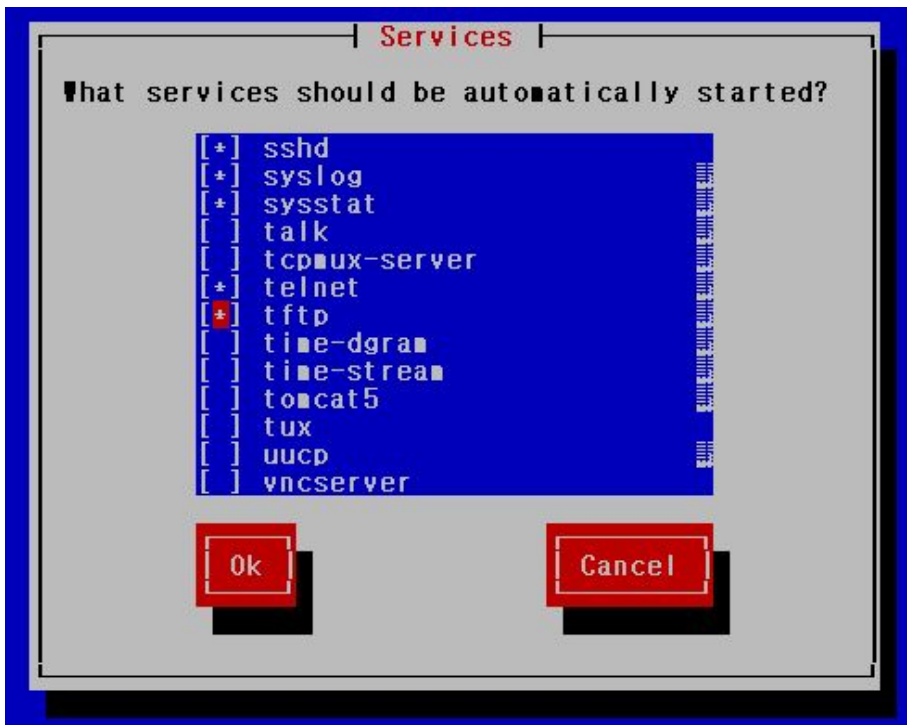
Follow the command

```
[root@localhost mv6410]# setup
```

Choose one from “System services”



Choose one from “tftp”



Click “OK”. Finally “quit” setup utility and execute Follow the command

```
[root@localhost mv6410]# service xinetd restart
```

7. u-boot

- ▶ Unzipping

Unzipping “s3c-u-boot-1.1.6_rel_080612.gz”

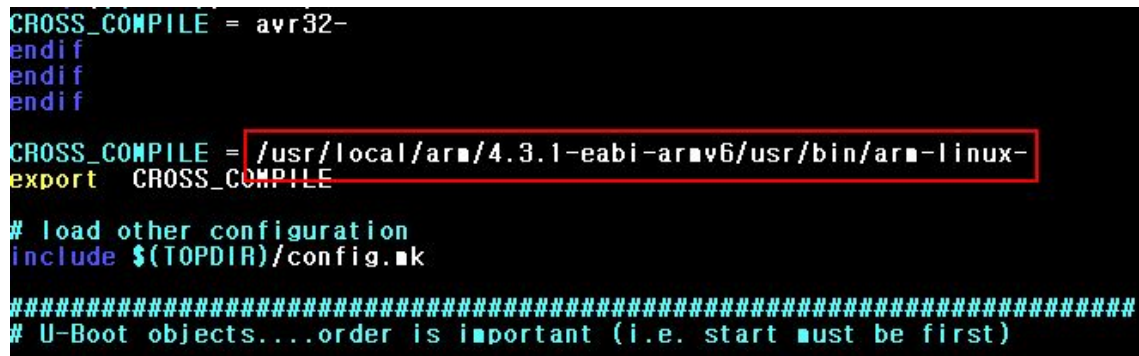
Follow the command

```
[root@localhost mv6410]# tar xvf s3c-u-boot-1.1.6_rel_080612.gz
[root@localhost mv6410]# cd s3c-u-boot-1.1.6
[root@localhost s3c-uboot-1.1.6]#
```

- ▶ To modify cross compile path.

Follow the command

```
[root@localhost s3c-uboot-1.1.6]# vi Makefile
```



```
CROSS_COMPILE = avr32-
endif
endif
endif

CROSS_COMPILE = /usr/local/arm/4.3.1-eabi-armv6/usr/bin/arm-linux-
export CROSS_COMPILE

# load other configuration
include $(TOPDIR)/config.mk

#####
# U-Boot objects...order is important (i.e. start must be first)
```

After modify , `CROSS_COMPILE = /usr/local/arm/4.3.1-eabi-armv6/usr/bin/arm-linux-` , Getting out.

- ▶ Setting IP Address by u-boot

Follow the command

```
[root@localhost s3c-u-boot-1.1.6]# vi include/configs/smdk6410.h
```

```
#define CONFIG_ETHADDR      00:40:5c:26:0a:5b
#define CONFIG_NETMASK      255.255.255.0
#define CONFIG_IPADDR       192.168.0.165
#define CONFIG_SERVERIP     192.168.0.160
#define CONFIG_GATEWAYIP    192.168.0.254

#define CONFIG_ZERO_BOOTDELAY_CHECK
```

<code>#define CONFIG_ETHADDR</code>	MAC ADDRESS
<code>#define CONFIG_NETMASK</code>	SUBNETMASK
<code>#define CONFIG_IPADDR</code>	Target Board PC
<code>#define CONFIG_SERVERIP</code>	HOST PC

- ▶ Doing Compile u-boot

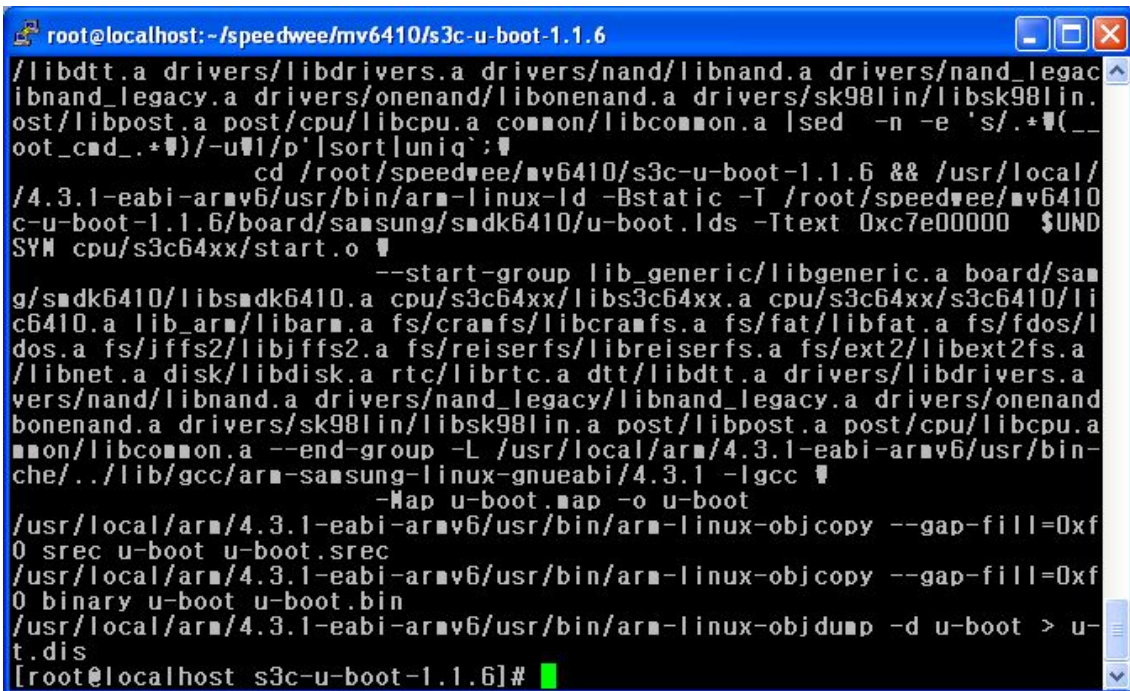
Follow the command

```
[root@localhost s3c-uboot-1.1.6]# make smdk6410_config
```

Configuring for smdk6410 board...

```
[root@localhost s3c-uboot-1.1.6]# make
```

The make done that made u-boot in folder “s3c-uboot-1.1.6”



```
root@localhost: ~/speedwee/mv6410/s3c-u-boot-1.1.6
/libdtt.a drivers/libdrivers.a drivers/nand/libnand.a drivers/nand_legacy/
libnand_legacy.a drivers/onenand/libonenand.a drivers/sk98lin/libsk98lin.
ost/libpost.a post/cpu/libcpu.a common/libcommon.a lsd -n -e 's/.*\(_\
oot_cmd_.*\)/-u\1/p'|sort|uniq`;
      cd /root/speedwee/mv6410/s3c-u-boot-1.1.6 && /usr/local/
/4.3.1-eabi-armv6/usr/bin/arm-linux-ld -Bstatic -T /root/speedwee/mv6410
c-u-boot-1.1.6/board/samsung/smdk6410/u-boot.lds -Ttext 0xc7e00000 $UND
SYM cpu/s3c64xx/start.o
      --start-group lib_generic/libgeneric.a board/sa
g/smdk6410/lib_smdk6410.a cpu/s3c64xx/lib_s3c64xx.a cpu/s3c64xx/s3c6410/li
c6410.a lib_arm/libarm.a fs/cramfs/libcramfs.a fs/fat/libfat.a fs/fdos/l
dos.a fs/jffs2/libjffs2.a fs/reiserfs/libreiserfs.a fs/ext2/libext2fs.a
/libnet.a disk/libdisk.a rtc/librtc.a dtb/libdtt.a drivers/libdrivers.a
vers/nand/libnand.a drivers/nand_legacy/libnand_legacy.a drivers/onenand
bonenand.a drivers/sk98lin/libsk98lin.a post/libpost.a post/cpu/libcpu.a
mon/libcommon.a --end-group -L /usr/local/arm/4.3.1-eabi-armv6/usr/bin-
che/./lib/gcc/arm-samsung-linux-gnueabi/4.3.1 -lgcc
      -Map u-boot.map -o u-boot
/usr/local/arm/4.3.1-eabi-armv6/usr/bin/arm-linux-objcopy --gap-fill=0xf
0 srec u-boot u-boot.srec
/usr/local/arm/4.3.1-eabi-armv6/usr/bin/arm-linux-objcopy --gap-fill=0xf
0 binary u-boot u-boot.bin
/usr/local/arm/4.3.1-eabi-armv6/usr/bin/arm-linux-objdump -d u-boot > u-
t.dis
[root@localhost s3c-uboot-1.1.6]#
```

After make, Copy u-boot.bin to “/tftpboot”

Follow the command

```
[root@localhost s3c-uboot-1.1.6]# cp u-boot /tftpboot
```

8. Kernel Compilation

- ▶ Unzipping

Unzipping “s3c-linux-2.6.21.tar.gz”

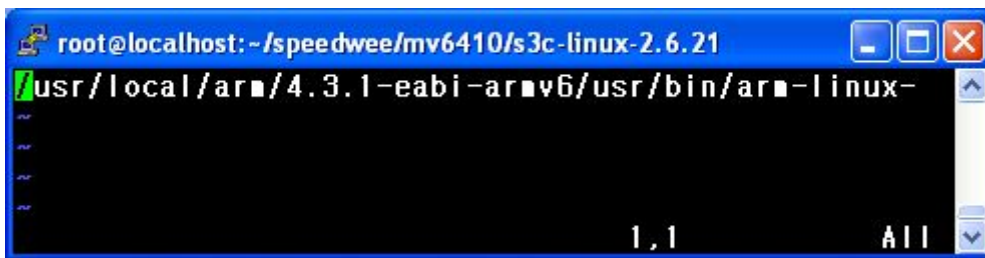
Follow the command

```
[root@localhost mv6410]# tar xvf s3c-linux-2.6.21.tar.gz
```

- ▶ To modify cross compile path.

Follow the command

```
[root@localhost test]# cd s3c-linux-2.6.21  
[root@localhost s3c-linux-2.6.21]# vi .cross_compile
```



After modify , “/usr/local/arm/4.3.1-eabi-armv6/usr/bin/arm-linux-“, Getting out

► Kernel patch

Doing not to get mismatch (null-point check) with Kernel, When you are compileing.

```
[root@localhost s3c-linux-2.6.21]# cat kernel-4.3.1.patch | patch -p1
```

```
KBUILD
[root@localhost s3c-linux-2.6.21]# cat kernel-4.3.1.patch | patch -p1
patching file Makefile
Reversed (or previously applied) patch detected! Assume -R? [n]
[root@localhost s3c-linux-2.6.21]# cat kernel-4.3.1.patch
--- s3c-linux-2.6.21/Makefile 2008-04-29 14:40:26.000000000 +0900
+++ s3c-linux-2.6.21/Makefile 2008-07-05 12:24:01.000000000 +0900
@@ -486,9 +486,9 @@
 all: vmlinux

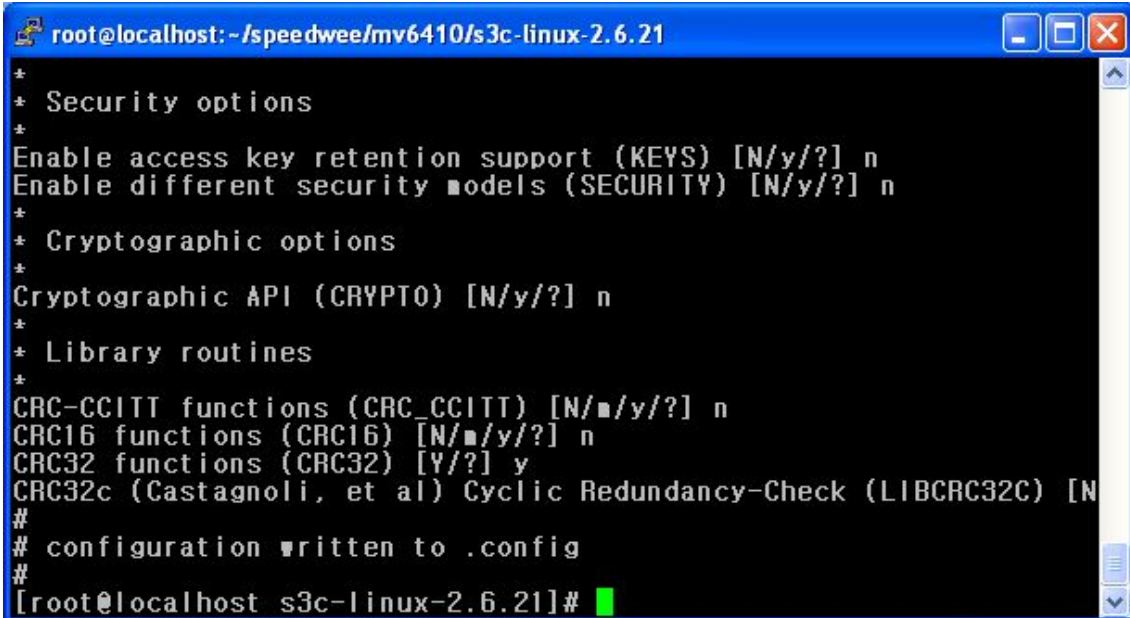
ifdef CONFIG_CC_OPTIMIZE_FOR_SIZE
-CFLAGS += -Os
+CFLAGS += -Os -fno-delete-null-pointer-checks
else
-CFLAGS += -O2
+CFLAGS += -O2 -fno-delete-null-pointer-checks
endif

include $(srctree)/arch/$(ARCH)/Makefile
[root@localhost s3c-linux-2.6.21]#
```

► Kernel Compile

Follow the command

```
[root@localhost s3c-linux-2.6.21]# make mv6410_defconfig
```

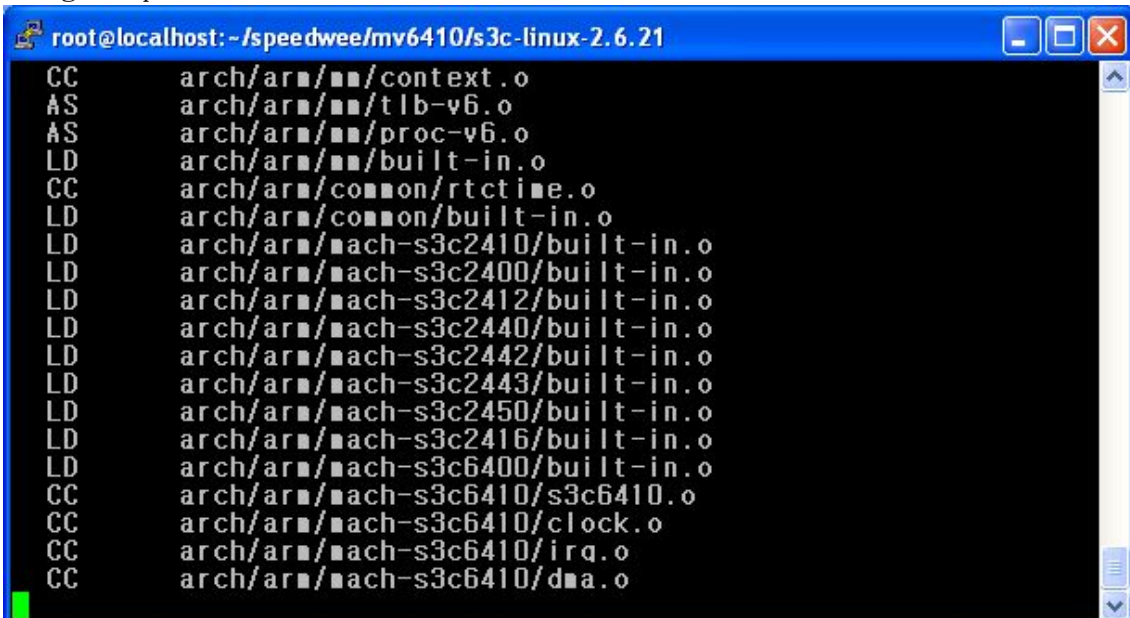


```
root@localhost: ~/speedwee/mv6410/s3c-linux-2.6.21
*
* Security options
*
Enable access key retention support (KEYS) [N/y/?] n
Enable different security models (SECURITY) [N/y/?] n
*
* Cryptographic options
*
Cryptographic API (CRYPTO) [N/y/?] n
*
* Library routines
*
CRC-CCITT functions (CRC_CCITT) [N/m/y/?] n
CRC16 functions (CRC16) [N/m/y/?] n
CRC32 functions (CRC32) [Y/?] y
CRC32c (Castagnoli, et al) Cyclic Redundancy-Check (LIBCRC32C) [N
#
# configuration written to .config
#
[root@localhost s3c-linux-2.6.21]#
```

Follow the command

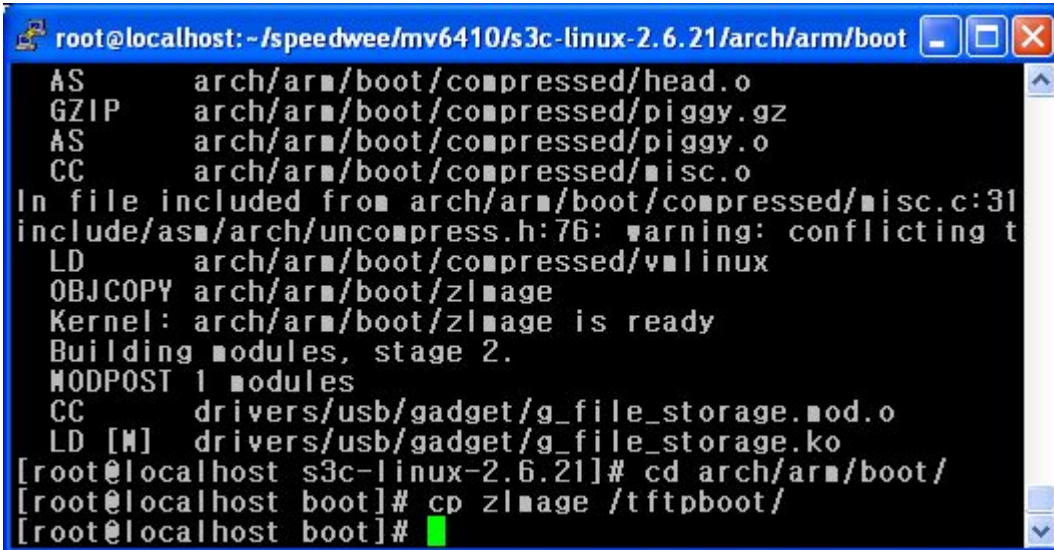
```
[root@localhost s3c-linux-2.6.21]# make
```

Being Compile



```
root@localhost: ~/speedwee/mv6410/s3c-linux-2.6.21
CC      arch/arm/mm/context.o
AS      arch/arm/mm/tlb-v6.o
AS      arch/arm/mm/proc-v6.o
LD      arch/arm/mm/built-in.o
CC      arch/arm/common/rtctime.o
LD      arch/arm/common/built-in.o
LD      arch/arm/mach-s3c2410/built-in.o
LD      arch/arm/mach-s3c2400/built-in.o
LD      arch/arm/mach-s3c2412/built-in.o
LD      arch/arm/mach-s3c2440/built-in.o
LD      arch/arm/mach-s3c2442/built-in.o
LD      arch/arm/mach-s3c2443/built-in.o
LD      arch/arm/mach-s3c2450/built-in.o
LD      arch/arm/mach-s3c2416/built-in.o
LD      arch/arm/mach-s3c6400/built-in.o
CC      arch/arm/mach-s3c6410/s3c6410.o
CC      arch/arm/mach-s3c6410/clock.o
CC      arch/arm/mach-s3c6410/irq.o
CC      arch/arm/mach-s3c6410/dma.o
```

Copy in folder arch/arm/boot/ "zImage" "to /tftpboot"



```
root@localhost: - /speedwee/mv6410/s3c-linux-2.6.21/arch/arm/boot
AS      arch/arm/boot/compressed/head.o
GZIP    arch/arm/boot/compressed/piggy.gz
AS      arch/arm/boot/compressed/piggy.o
CC      arch/arm/boot/compressed/misc.o
In file included from arch/arm/boot/compressed/misc.c:31:
include/asm/arch/uncompress.h:76: warning: conflicting t
LD      arch/arm/boot/compressed/vmlinux
OBJCOPY arch/arm/boot/zImage
Kernel: arch/arm/boot/zImage is ready
Building modules, stage 2.
MODPOST 1 modules
CC      drivers/usb/gadget/g_file_storage.mod.o
LD [M]  drivers/usb/gadget/g_file_storage.ko
[root@localhost s3c-linux-2.6.21]# cd arch/arm/boot/
[root@localhost boot]# cp zImage /tftpboot/
[root@localhost boot]#
```

Follow the command

```
[root@localhost boot]# cp zImage /tftpboot
```

9. Root file System

Root filesystem is composed of Cramfs (Compressed ROM file system).

To port the Root File System onto the target board easily, copy the root file system to `'/tftpboot/'` directory.

Follow the command

```
[root@localhost mv6410]# cp rootfs_qtopia.cramfs /tftpboot/
```

11. Setting for downloading Host PC to Board

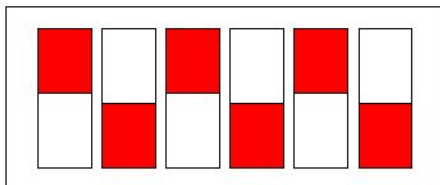
In this chapter, you can understand how to download u-boot.bin zImage and file system Please the following window appears on your screen.

First, you have to set up environment such as Board with Host PC and then Connect USB 2.0 Device with your Host PC to download through USB and also UART for monitoring.



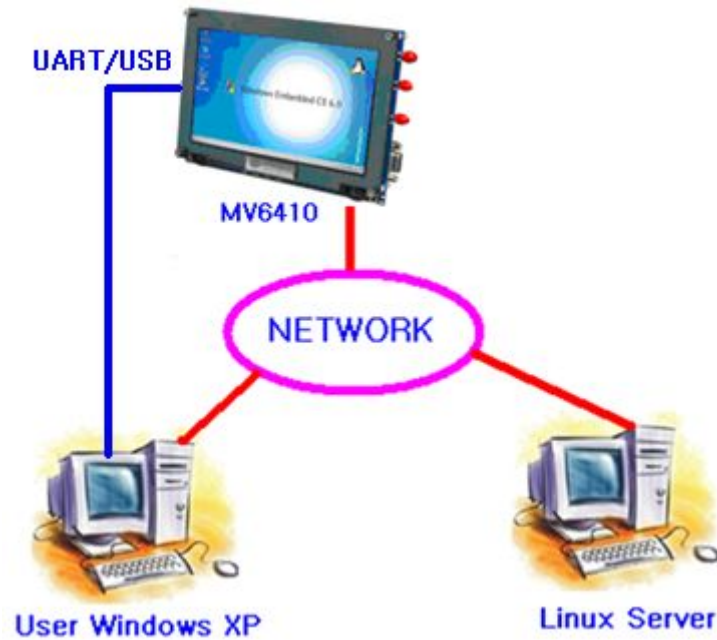
To do configuration NOR Flash through the Dipswitch.

► NOR Flash Boot Mode, DIP Switch



11. Downloading

The ways are between DNW and Minicom through TFTP form Linux of Server. I tell you about DNW through TFTP.



DDR, NAND Space

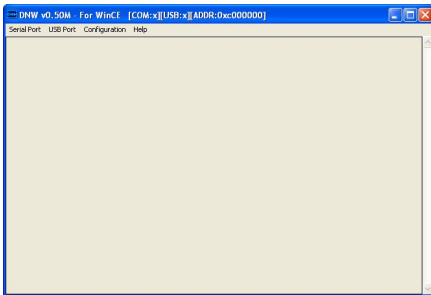
DDR space Virtual Address only uses by u-boot

BANK	Physical Address	Virtual Address	Size
0	0x30000000 ~ 0x33FFFFFF	0xC0000000 ~ 0xC3FFFFFF	64 MB
1	0x38000000 ~ 0x3BFFFFFF	0xC8000000 ~ 0xCBFFFFFF	64 MB

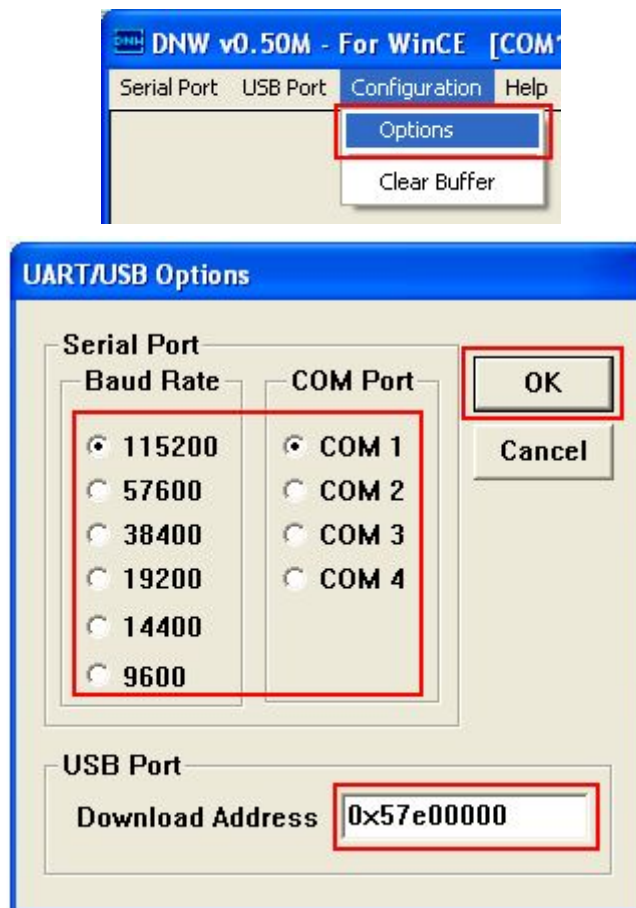
NAND FLASH Space

NAND block No.	NAND block offset	Size	Contents
0x0 (0x0 ~ 0xF)	0x0 (0x0 ~ 0x3FFFF)	256Kb	U-Boot Bootloader
0x10 (0x10 ~ 0x7F)	0x40000 (0x40000 ~ 0x1FFFFFF)	1.75MB	Linux Kernel
0x80 (0x80 ~ 0xC7F)	0x200000 (0x200000 ~ 0x31FFFFFF)	48MB	Root file system
0xC80 (0xC80 ~ end)	0x3200000 (0x3200000 ~ end)	(NAND 50 MB)	Extra area

Run DNW in WSourcesWDNW



On the Configuration menu, click Options to set the UART/USB options. The following window appears on your screen. Select Baud Rate and COM Port as shown in figure "UART/USB options", enter the download address as 0x57e00000 and then click OK button.

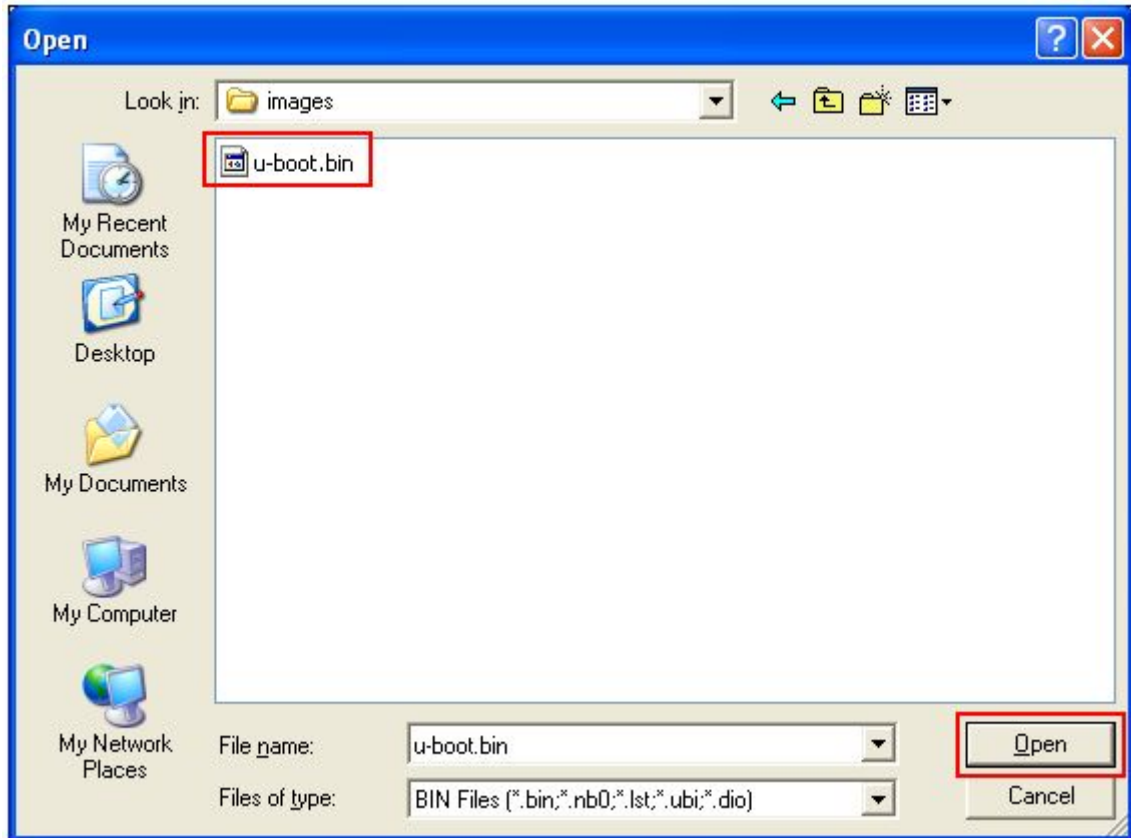
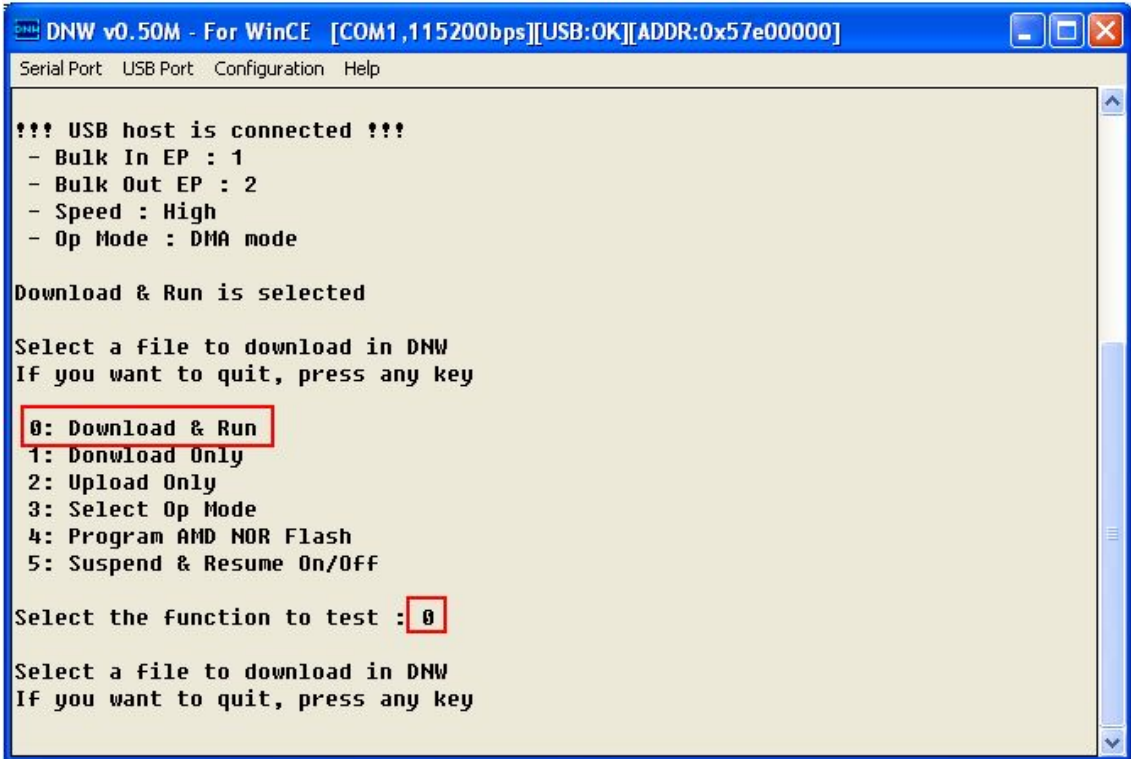


<UART/USB options>

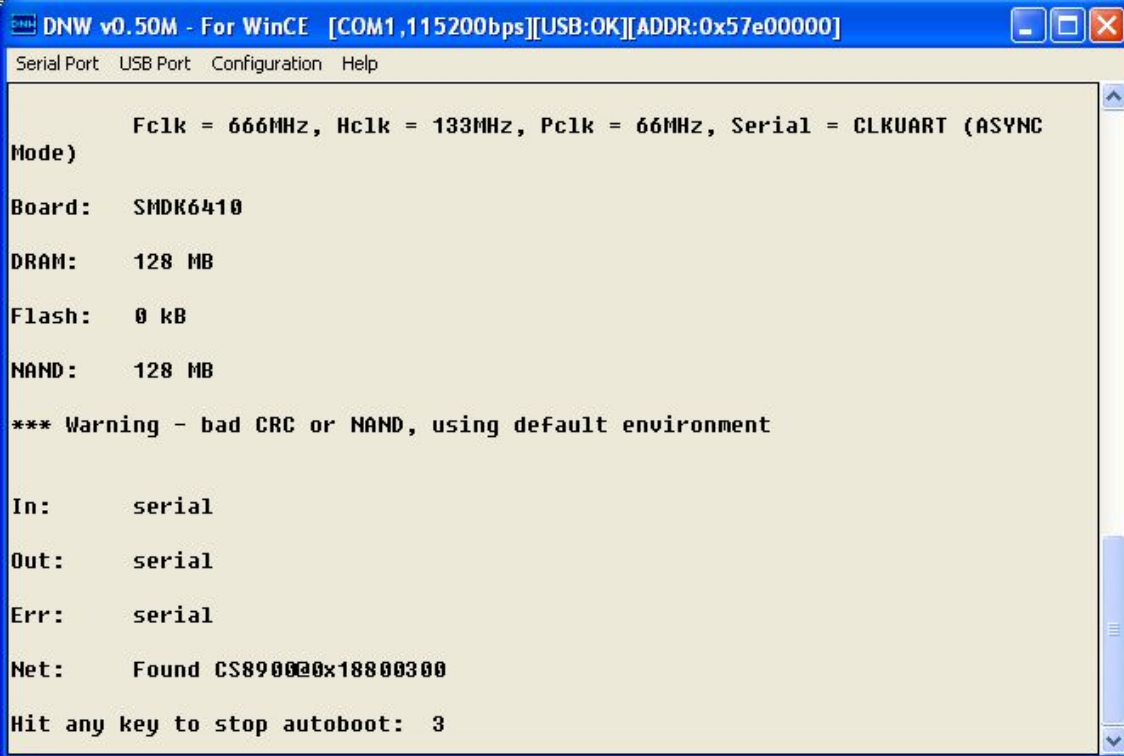
On the Serial Port menu, click Connect. Switch ON the reference board and then press any key and then install the USB driver in WSourcesWDNW driver directory.



Enter “0”, On the USB Port menu, click Transmit and the following window appears on your screen. Select u-boot.bin file from XWSources\Linux\Wimages directory and then click Open button.



As soon as u-boot.bin download is over, the following messages appear in the DNW window. Please hit the SPACE BAR key to view the current Ethernet Boot Loader Configuration. Configure the Ethernet Boot loader as follows by entering the respective options.



```
DNW v0.50M - For WinCE [COM1,115200bps][USB:OK][ADDR:0x57e00000]
Serial Port  USB Port  Configuration  Help

      Fc1k = 666MHz, Hc1k = 133MHz, Pc1k = 66MHz, Serial = CLKUART (ASYNC
Mode)
Board:  SMDK6410
DRAM:   128 MB
Flash:  0 kB
NAND:   128 MB

*** Warning - bad CRC or NAND, using default environment

In:     serial
Out:    serial
Err:    serial
Net:    Found CS8900@0x18800300
Hit any key to stop autoboot: 3
```

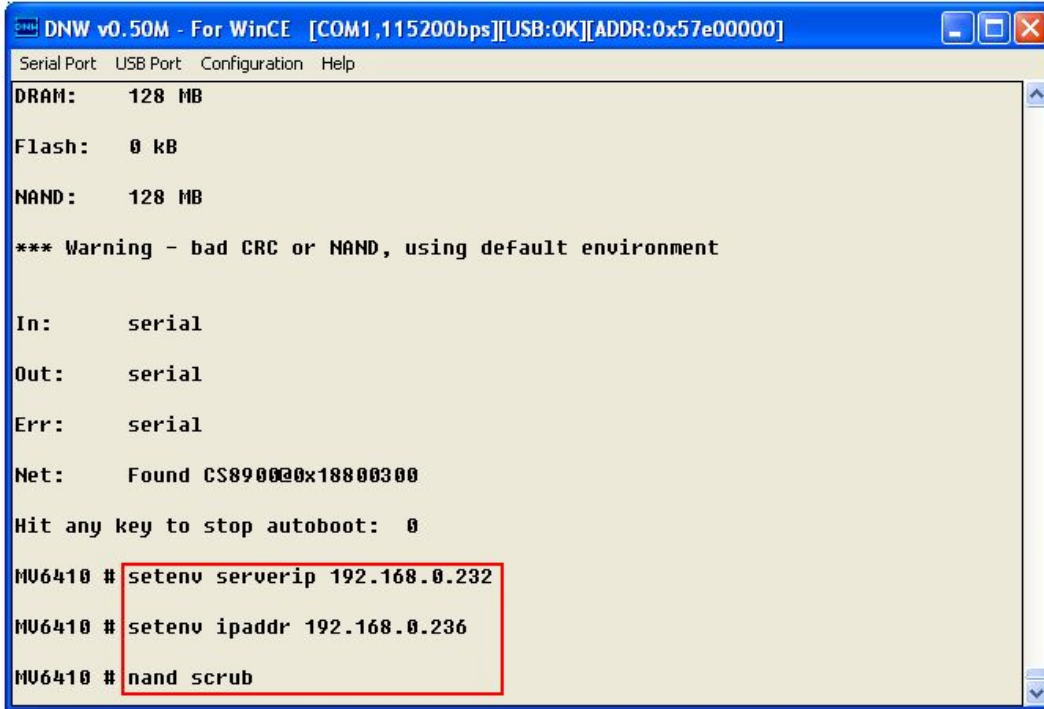
Setting IP address

Linux Server PC for taking in tftpboot

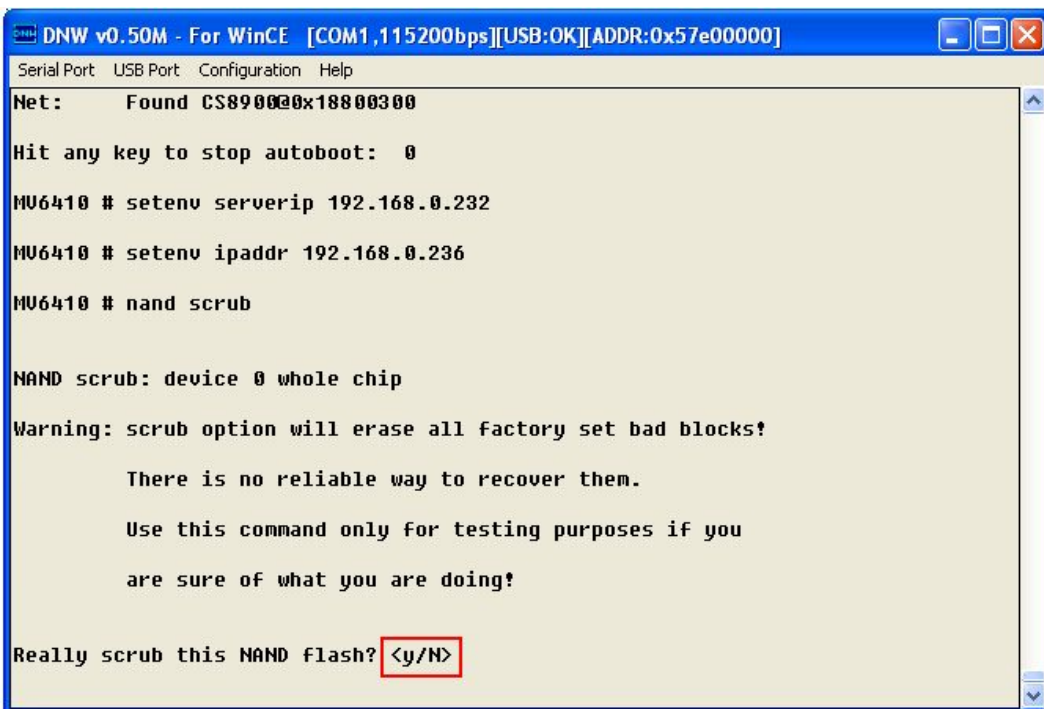
TFTP Server IP : `setenv serverip 192.168.0.232`

Device PC IP : `setenv ipaddr 192.168.0.236`

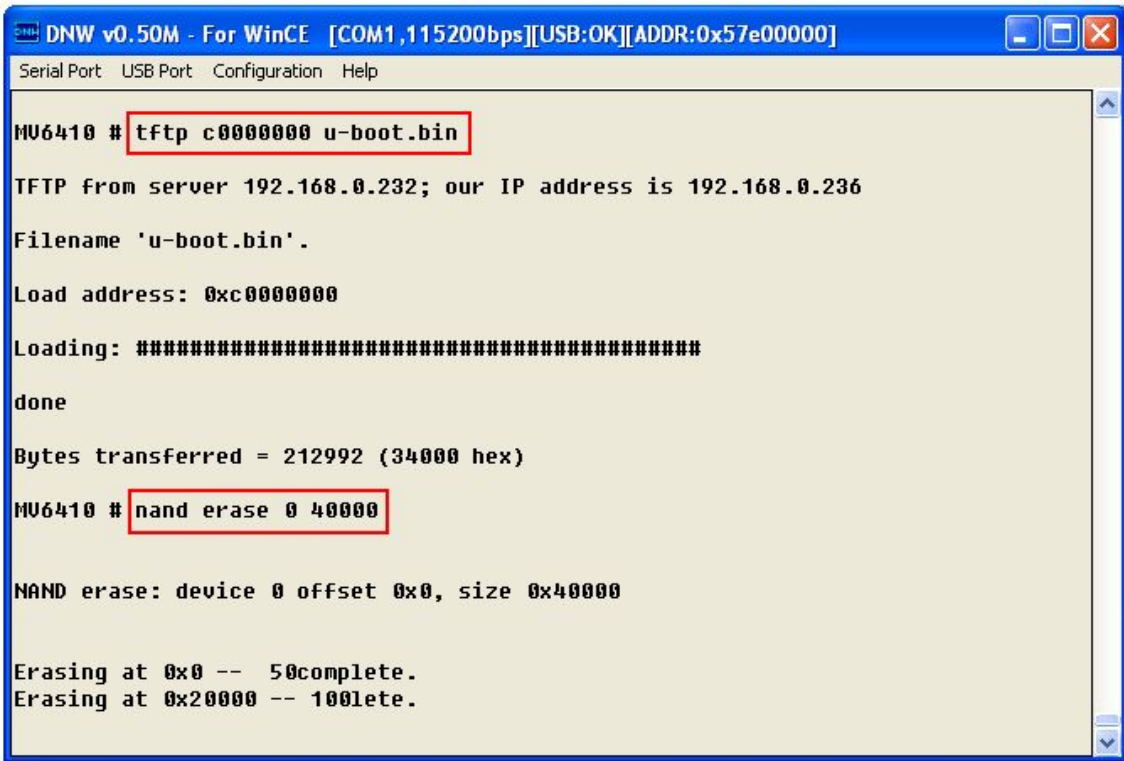
For erasing Bad blocks in NAND : `nand scrub`



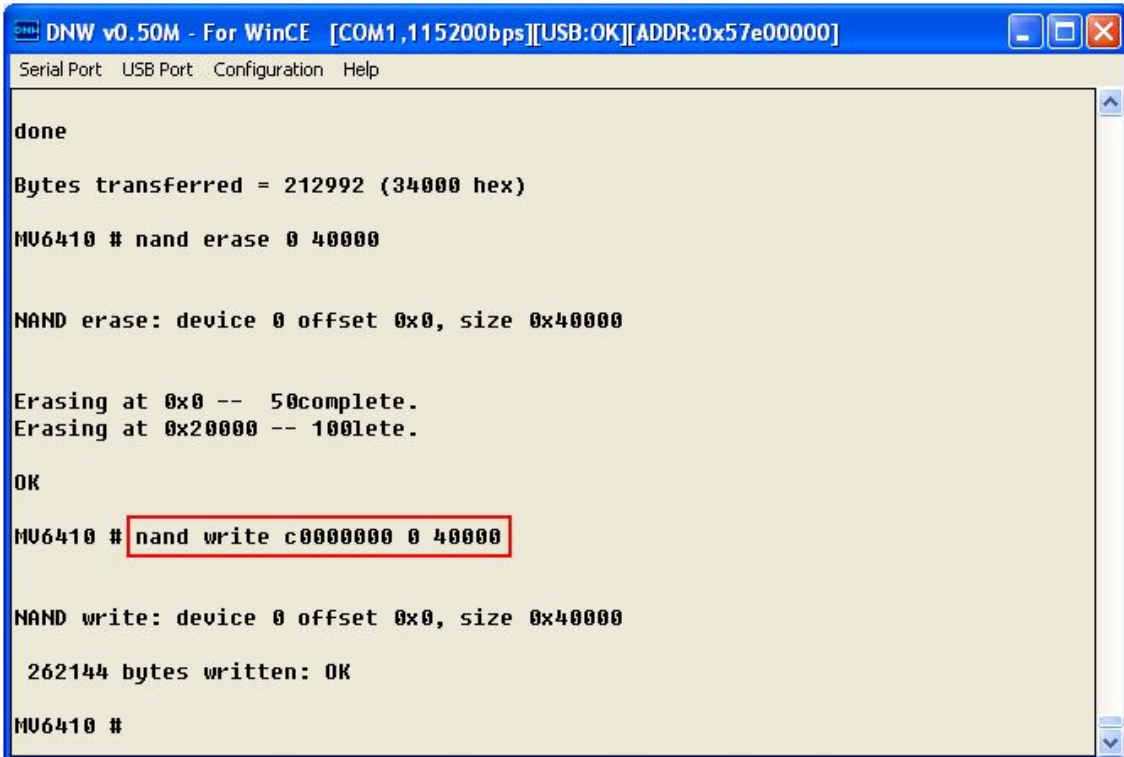
Input 'Y' directly hit 'Enter'



```
tftp c0000000 u-boot.bin  
nand erase 0 40000
```

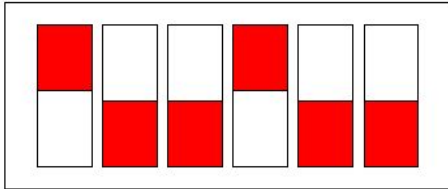


```
nand write c0000000 0 40000
```

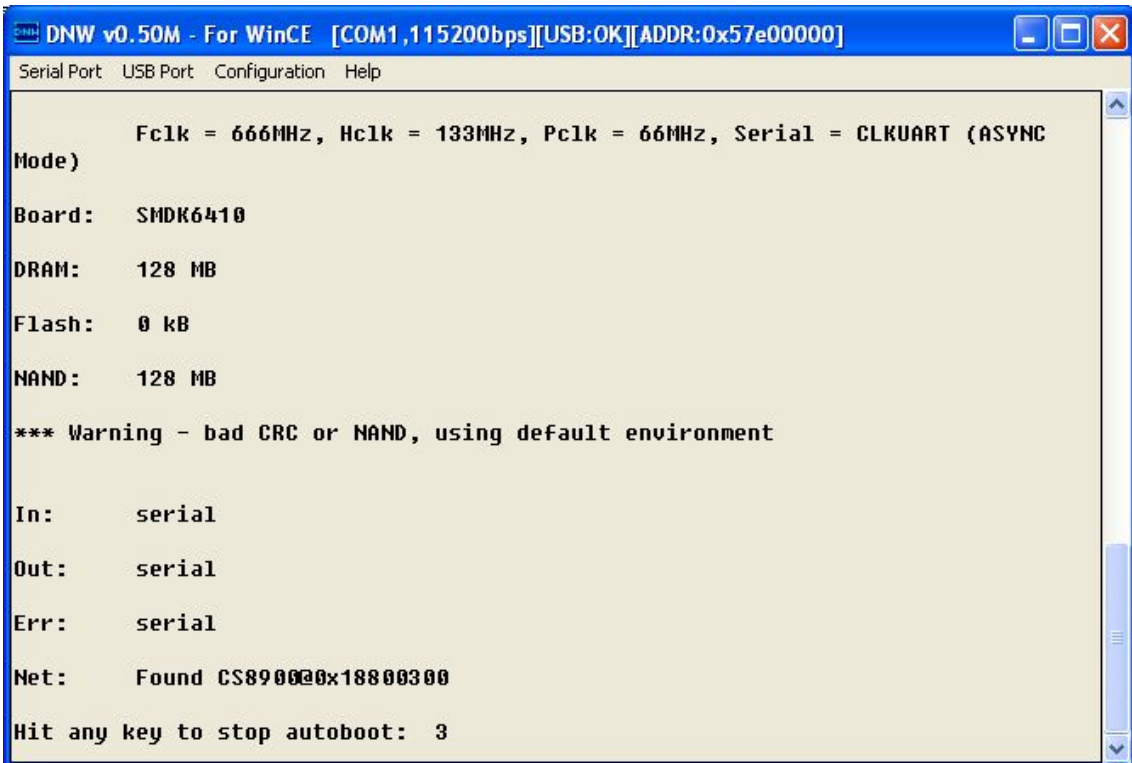


Please reboot After Setting NAND of Boot Mode

► NAND Flash Boot Mode, DIP Switch



As soon as u-boot.bin download is over, the following messages appear in the DNW window. Please hit the SPACE BAR key to view the current Ethernet Boot Loader Configuration. Configure the Ethernet Boot loader as follows by entering the respective options.

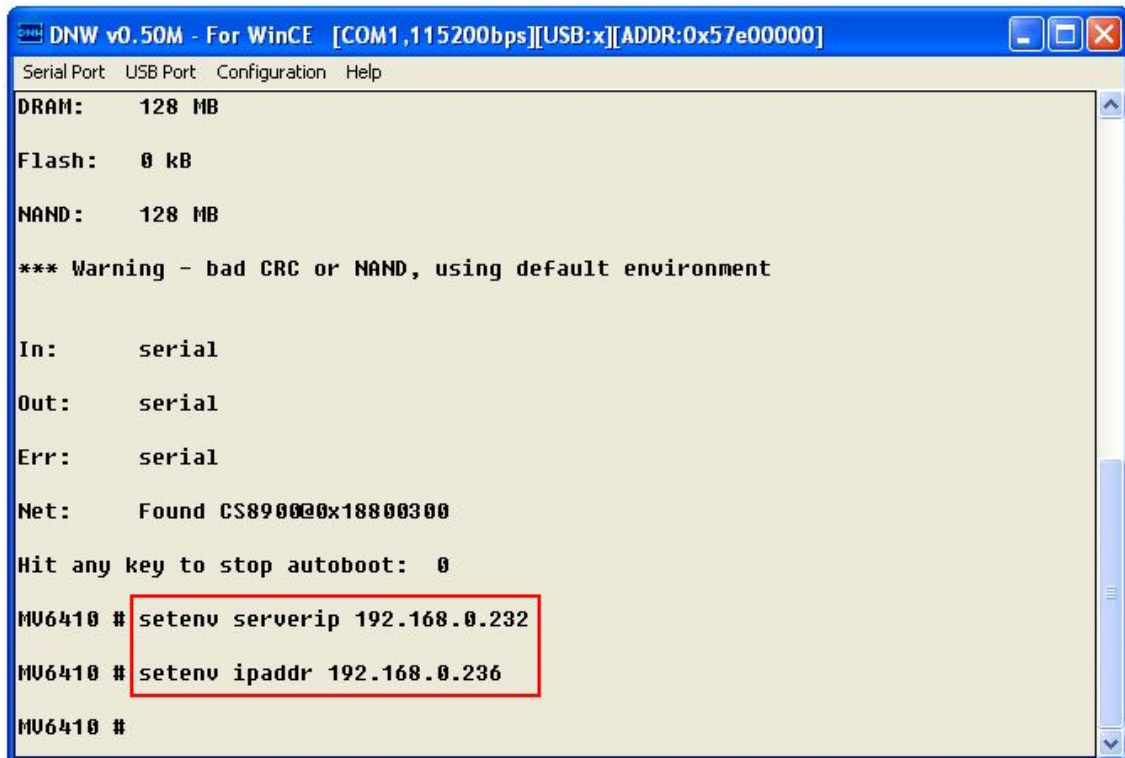


Setting IP address

Linux Server PC for taking in tftpboot

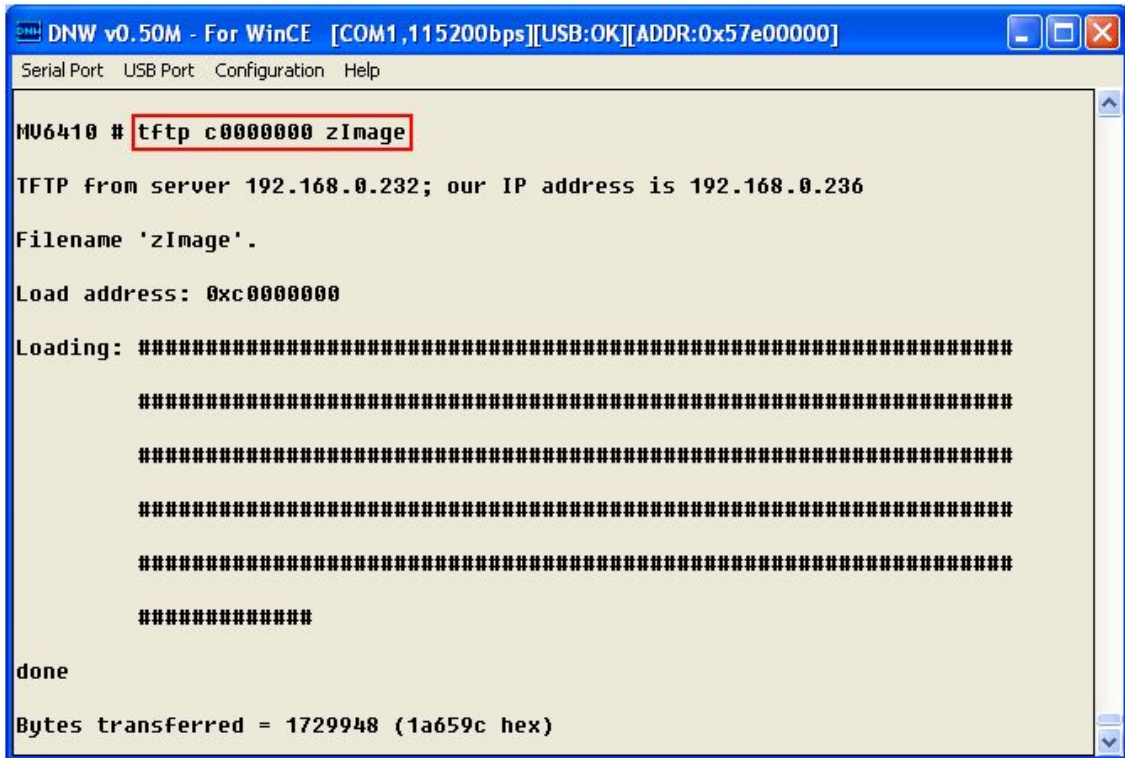
TFTP Server IP : `setenv serverip 192.168.0.232`

Device PC IP : `setenv ipaddr 192.168.0.236`

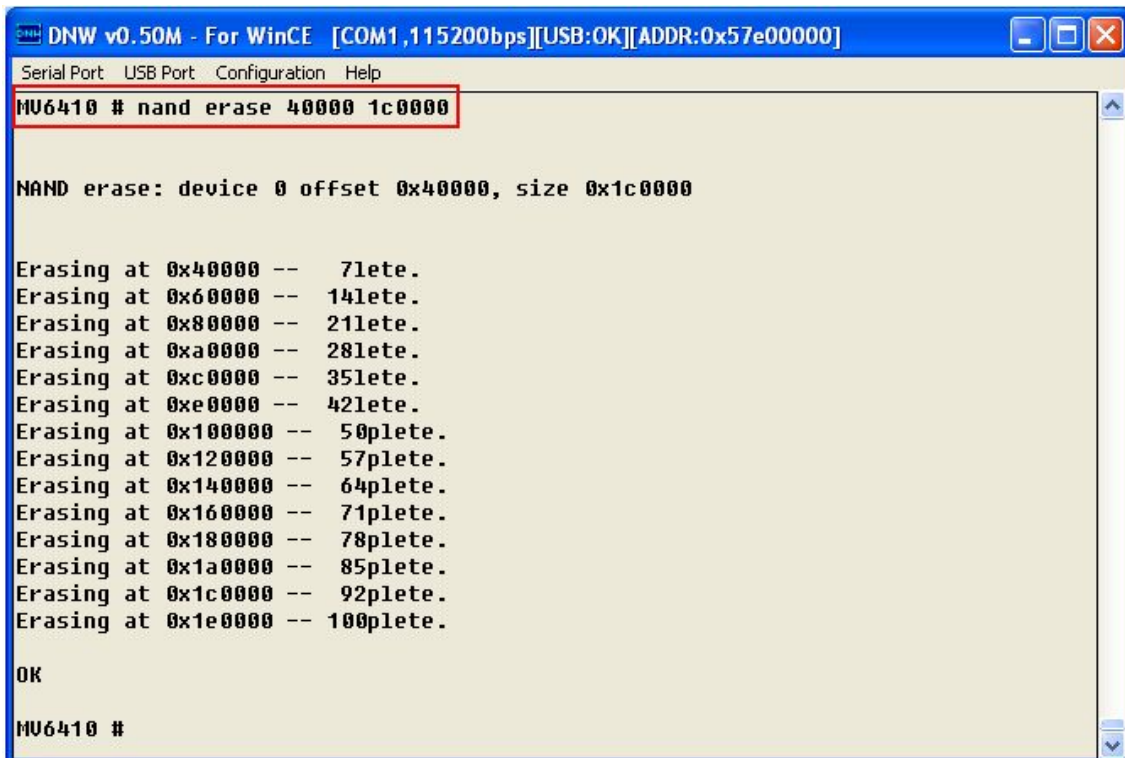


```
DNW v0.50M - For WinCE [COM1,115200bps][USB:x][ADDR:0x57e00000]
Serial Port  USB Port  Configuration  Help
DRAM:      128 MB
Flash:     0 kB
NAND:      128 MB
*** Warning - bad CRC or NAND, using default environment
In:        serial
Out:       serial
Err:       serial
Net:       Found CS89000@0x18800300
Hit any key to stop autoboot: 0
MV6410 # setenv serverip 192.168.0.232
MV6410 # setenv ipaddr 192.168.0.236
MV6410 #
```

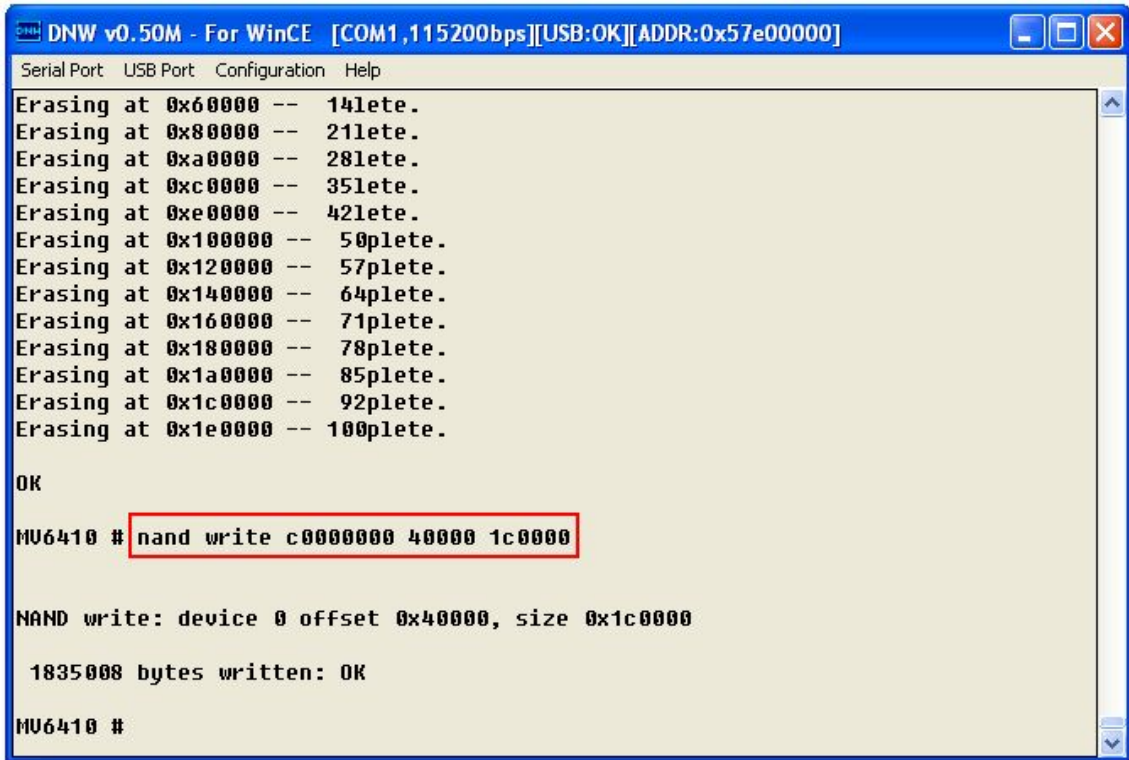
tftp c0000000 zImage



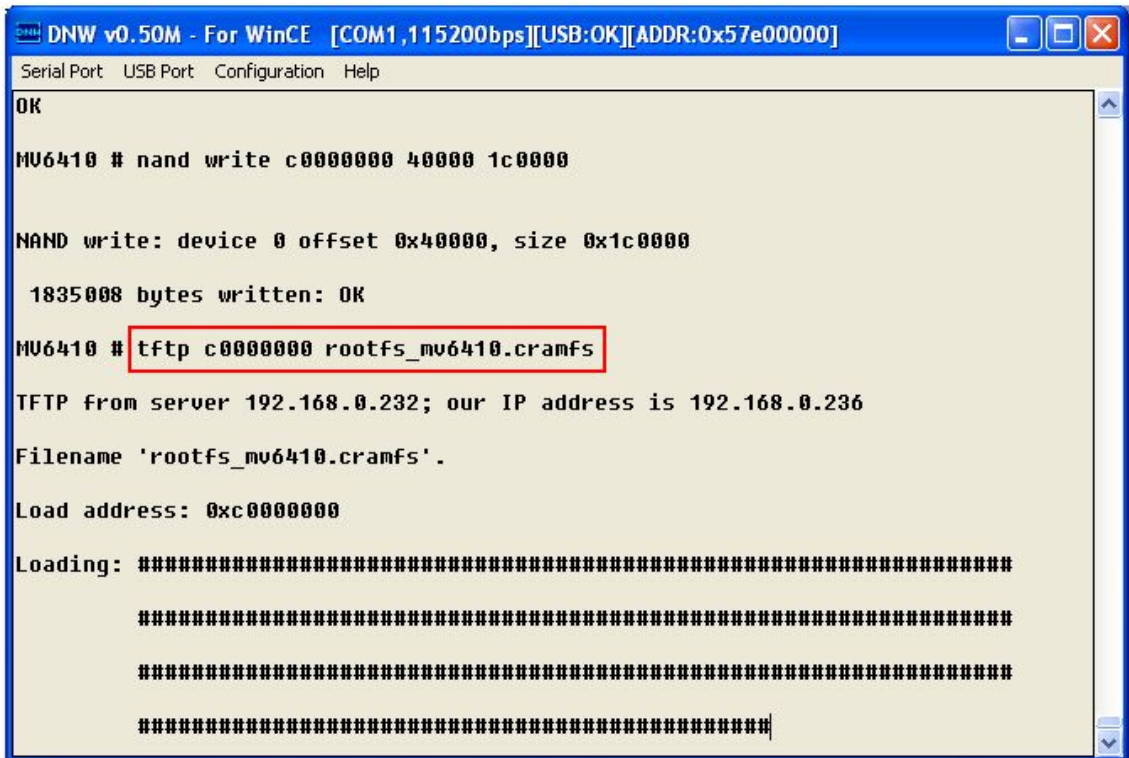
nand erase 40000 1c0000



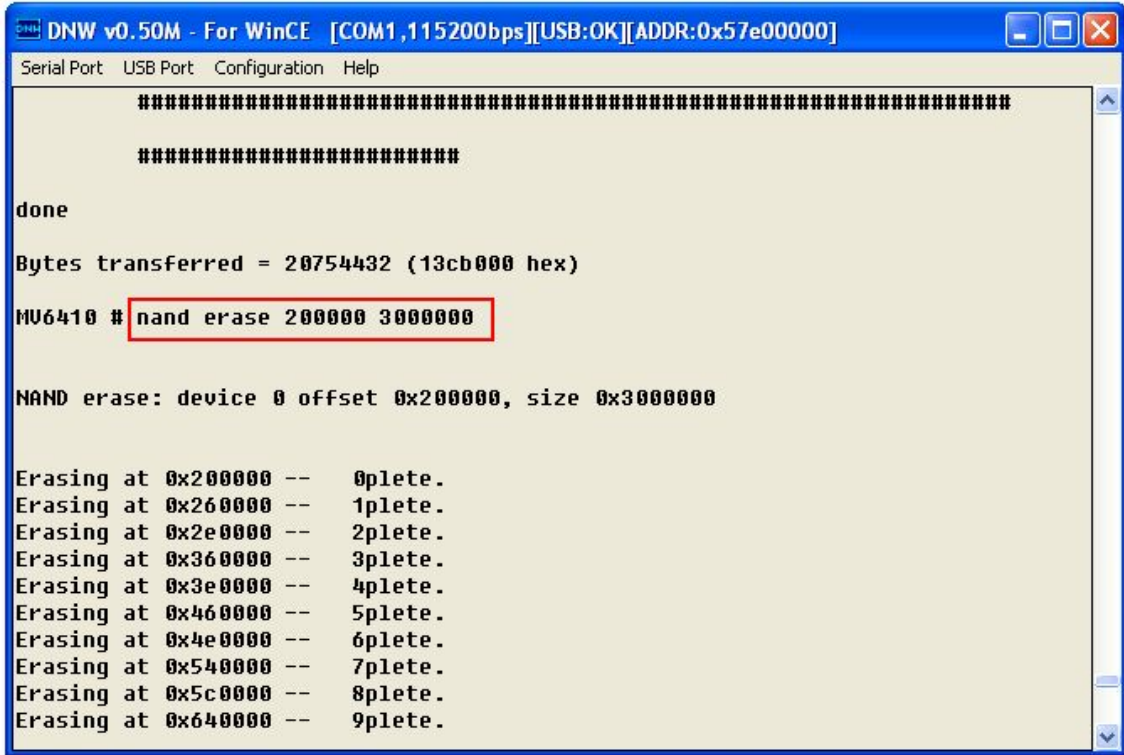
nand write c0000000 40000 1c0000



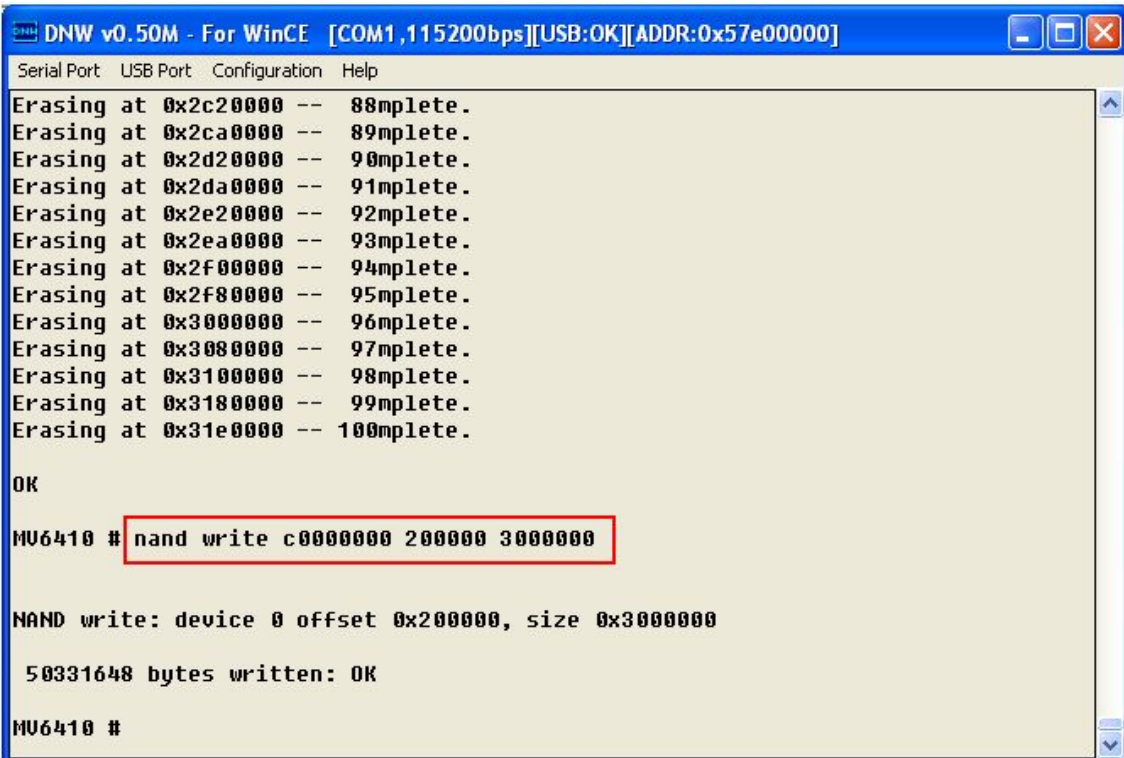
tftp c0000000 rootfs_mv6410.cramfs



nand erase 200000 3000000



nand write c000000 200000 3000000



Please reboot

12. Application (Camera, USB WIFI, TV OUT, S-Video)

```
[root@glibc root]# ll
-rw-r--r--    1 513    root           0 Jan  1 00:00 .bash_history
-rw-r--r--    1 513    root          175 Jan  1 00:00 .bash_logout
-rw-r--r--    1 513    root          161 Jan  1 00:00 .bash_profile
-rw-r--r--    1 513    root         1.7k Jan  1 00:00 .bashrc
-rwxr-xr-x    1 513    root        26.9k Jan  1 00:00 bplay*
lrwxrwxrwx    1 513    root           5 Jan  1 00:00 brec -> bplay*
-rwxr-xr-x    1 513    root        14.9k Jan  1 00:00 fbcam*
-rwxr-xr-x    1 513    root       194.6k Jan  1 00:00 mpg123*
-rw-r--r--    1 513    bin          2.6M Jan  1 00:00 rt73.ko
-rw-r--r--    1 root    root       123.3k Jan  1 00:00 s3c-tvenc.ko
-rw-r--r--    1 root    root        99.0k Jan  1 00:00 s3c-tvscaler.ko
-rwxr-xr-x    1 513    bin          2.6M Jan  1 00:00 tv_test*
[root@glibc root]# _
```

bplay : Player speaker program

brec : Microphone program

fbcam : Camere program

mpg123 : MP3 player

tv_test : TVOUT program

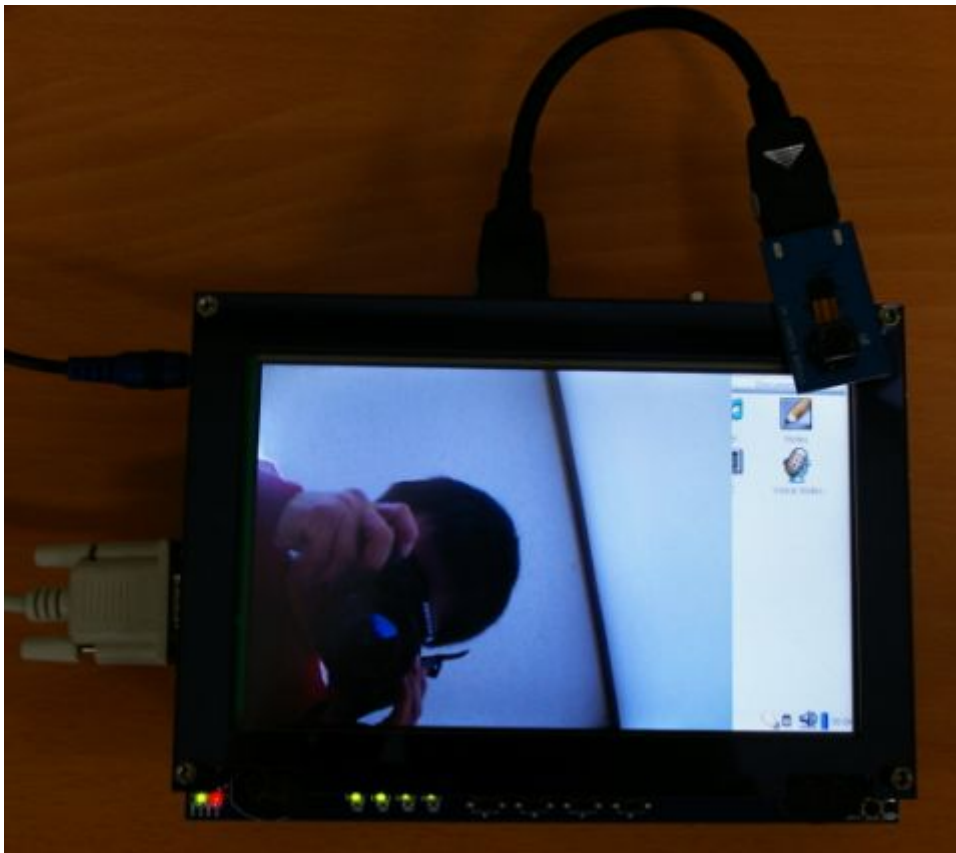
rt73.ko : USB WIFI

1) Camera

Booting linux Beforehand connection camera with main board.

Working #./fbcam

```
[root@glibc root]# ./fbcam
External Camera initialized
Resolution changed back to VGA (640x480) mode (default)
Preview memory required: 0x00320000 bytes
Preview memory required: 0x00258000 bytes
Capturing 640x480x0 "RGBR" images
Images are 0 bytes each
```



2) WIFI

```
# insmod rt73.ko
# ifconfig rausb0 192.168.xxx.xxx
# iwlist scanning
...
```

```
[root@glibc root]# usb 1-1: new full speed USB device
ress 6
usb 1-1: configuration #1 chosen from 1 choice

[root@glibc root]# insmod rt73.ko
idVendor = 0x148f, idProduct = 0x2573
usbcore: registered new interface driver rt73
[root@glibc root]# ifconfig rausb0 192.168.0.236
=> usb_rtusb_open
[root@glibc root]# iwlist scanning
```

Make sure AP list of names.

```
# iwconfig rausb0 essid [Write! Ap of name ]
```

```

Channel:6
Encryption key:off
Bit Rates:36 Mb/s
Cell 06 - Address: 08:10:74:03:BA:8A
Mode:Managed
ESSID:"vipsap3"
Channel:7
Encryption key:on
Bit Rates:1 Mb/s
Cell 07 - Address: 00:13:10:3B:BB:37
Mode:Managed
ESSID:""
Channel:7
Encryption key:on
Bit Rates:108 Mb/s
Cell 08 - Address: 00:1F:1F:17:5C:F8
Mode:Managed
ESSID:"CubicAP-1"
Channel:11
Encryption key:on
Bit Rates:0 kb/s

[root@glibc root]# iwconfig rausb0 essid vipsap3
[root@glibc root]# _
```

2) TVOUT, S-Video

First. Executes both

```
# insmod s3c-tvscaler.ko
```

```
# insmod s3c-tvenc.ko
```

Second Execute your wants of things

```
#!/tv_test 0 -> Composite tv out
```

```
#!/tv_test 0 0 -> Composite tv out
```

```
#!/tv_test 0 1 -> S-Video tv out
```

```
[root@glibc root]# insmod s3c-tvscaler.ko
S3C TV Scaler  Init : Success
Done
[root@glibc root]# insmod s3c-tvenc.ko
S3C TV Encoder  Init : Success
Done
[root@glibc root]# ./tv_test 0 0
Device file opens3c_tvscaler_init:
TV-OUT: VIDIOC_ENUMINPUT : index = 0
TV-OUT: VIDIOC_S_INPUT
TV-OUT: VIDIOC_ENUMOUTPUT : index = 0
TV-OUT: VIDIOC_S_OUTPUT
C: VIDIOC_S_FMT
P: VIDIOC_S_CTRL

pPreBuff = 0x40001000
pPostBuff = 0x40401000
V4L2 APPL : Name of the interface is S3C TV-OUT driveTVENCODER
V4L2 APPL : [0]: IN channel name LCD FIFO_OUT
V4L2 APPL : LCD FIFO INPUT
V4L2 APPL : [0]: OUT channel name TV-OUT
V4L2 APPL : TV-OUT
```

You can see TV OUT form Board