

NXP LPC32xx Linux Quickstart on the Phytex 3250 board

Version history

<u>Release</u>	<u>Date</u>	<u>Comments</u>
1.00	01/12/2009	Initial release
1.01	2/12/2009	Minor fixes from feedback comments

Table of contents

1	Purpose of this quickstart guide	4
1.1	<i>Copyrights and limitations</i>	4
1.2	<i>Where to start</i>	4
1.3	<i>Host system requirements</i>	4
1.3.1	<i>Additional host machine software requirements</i>	5
1.4	<i>Target board requirements</i>	5
1.5	<i>Development environment requirements</i>	5
1.6	<i>Basic required Linux skills</i>	5
2	Getting and installing LTIB	6
3	Configuring LTIB and the build process	7
3.1	<i>Selecting the LTIB platform</i>	7
3.2	<i>Setting up the platform and build environment</i>	7
3.2.1	<i>Target library and toolchain selection</i>	7
3.2.2	<i>u-boot</i>	7
3.2.3	<i>Linux kernel</i>	7
3.2.4	<i>Root filesystem package selection</i>	8
3.2.5	<i>Target system configuration</i>	8
3.2.6	<i>Target image generation setup</i>	8
3.3	<i>Starting the build cycle</i>	9
3.3.1	<i>Selecting board variants</i>	9
4	Deploying Linux to your Phytec board using NFS	10
4.1	<i>Environment setup</i>	10
4.2	<i>u-boot</i>	10
4.2.1	<i>Installing u-boot on the Phytec board</i>	10
4.2.2	<i>u-boot configuration</i>	11
4.3	<i>Host side configuration for Linux</i>	12
4.3.1	<i>Setting up host TFTP to server the kernel image</i>	12
4.3.2	<i>Setting up host NFS server</i>	13
5	Linux boot	13
5.1	<i>U-boot environment setup</i>	13
5.2	<i>Linux boot</i>	14

1 Purpose of this quickstart guide

This quickstart guide explains the steps necessary to get Linux up and running on the Phytex 3250 board using LTIB. This guide is meant for users new to Linux or as a method to get a Linux image up and running fast.

From www.bitshrine.org, LTIB is explained as:

“The LTIB (Linux Target Image Builder) project is a simple tool that can be used to develop and deploy BSPs (Board Support Packages) for various target platforms. Using this tool a user will be able to develop a GNU/Linux image for their target platform.”

LTIB handles a lot of the work of building a complete Linux system, such as setting up the root filesystem, package selection, system init configuration, etc. Advanced users that prefer not to use LTIB may be interested in just getting the necessary kernel and bootloader patches and starting from there – please see the LPC32xx Linux User’s guide for the locations of those patches.

The steps covered include the following:

- Getting and installing LTIB
- Configuring LTIB and the build process
- Deploying Linux to your Phytex board using NFS

From start to finish, the process can take anywhere from a few hours to a day depending on factors such as host machine speed, internet connection speed, current host system configuration, etc.

1.1 Copyrights and limitations

The LPC32xx BSP is provided free of charge and with no support from NXP. Portions of the BSP are copyrighted by NXP Semiconductors.

1.2 Where to start

The sections in this guide are meant to be followed in sequential order starting with Section 2. Prior to that, the requirements for the host, target, and development environment should be reviewed in Sections 1.2 to 1.4.

1.3 Host system requirements

To develop Linux for the LPC32xx, a host PC running the Linux operating system is needed. Because of the many variations of Linux releases and supported packages, it is unknown if the tools included with the BSP will work correctly on a specific release of Linux. LTIB and the supporting tools have been tested with the Fedora 9 Linux release. Although other releases may work fine, they are currently untested.

The NXP VFP GCC4 toolchain used by LTIB has only been tested on Fedora Core 9 on an i386 based system.

Your host machine will also need a connection to the internet to download packages for the target system. Packages are downloaded and cached on the host machine as needed.

1.3.1 Additional host machine software requirements

As LTIB is installing itself, it may fail due to missing host system packages. If this occurs, carefully read the LTIB error messages and install any other packages on your host machine required by LTIB.

1.4 Target board requirements

A Phytex PhyCore 3250 board is required to run the generated bootloader and kernel images. The board is available from Phytex at www.phytex.com. An optional LCD module is available for the 3250 board to support graphical and touchscreen applications.

1.5 Development environment requirements

The target boot configuration is setup to support DHCP. A DHCP server should be available on the network where your Linux target board is plugged in. If a DHCP server isn't available, enter a manual IP configuration for the target network configuration in step 3.2.5.

An SD or MMC card is also needed to initially setup the bootloader on the Phytex board,

1.6 Basic required Linux skills

Skills such as TFTP boot configuration, basic network setup, mounting and using a storage card (such as SD/MMC), and installing new host system packages are a few of the skills assumed to be known by the user in this guide.

There are already a lot of resources on the internet explaining these types of things so they won't be covered here in this guide.

2 Getting and installing LTIB

Getting and installing LTIB is easy! Open a web browser on your host machine and go to the www.bitshrine.org website. Look for the section for LTIB installation. Follow the directions there to install LTIB. *It is highly recommended to use the netinstall script to install LTIB or use a direct CVS download (explained in the LTIB FAQ). Using a fixed snapshot image may not get you the latest files.*

LTIB will begin installing by downloading files and packages over the internet. If any errors occurred during installation, correct the error before continuing. Most errors are due to missing or out-of-date packages on the host machine.

Once LTIB has finished installing, run LTIB per the website instructions. A menu should appear requesting selection of the platform. Continue to the next section: Configuring LTIB and the build process.

3 Configuring LTIB and the build process

3.1 Selecting the LTIB platform

Once LTIB has been installed and executed for the first time, a menu will appear requesting which platform to use with LTIB, Select the *Phytec 3250 board with the NXP LPC32xx SoC* option. Choose the Exit option and save the configuration to continue to the *LTIB:NXP LPC3XXX on the Phytec 3250 board* menu.

3.2 Setting up the platform and build environment

Prior to building the Linux system, several options need to be setup at the LTIB settings menu. *The default configuration included with LTIB should actually have most of these configurations options pre-setup, but make sure to verify that the default settings are correct.*

3.2.1 Target library and toolchain selection

Select the following options in the LTIB settings menu to setup the target library and cross-toolchain.

LTIB settings menu option	Value to select or enter
Target C library type	glibc
C library package	from toolchain only
Toolchain	gcc-4.3.2-glibc-2.7 (VFP with soft-float)
Enter any CFLAGS for gcc/g++	-fsigned-char -O3 -msoft-float -mfpu=vfp

The target library and cross-toolchain configuration setup is complete. When LTIB starts the build process, the toolchain will be downloaded and installed prior to building code.

3.2.2 u-boot

Select the following options in the LTIB settings menu to build u-boot.

LTIB settings menu option	Value to select or enter
Bootloader choice	u-boot 1.3.3 for the Phytec 3250 board
u-boot flags	()

u-boot will download and build for the target hardware.

3.2.3 Linux kernel

Select the following options in the LTIB settings menu for the Linux kernel.

LTIB settings menu option	Value to select or enter
Kernel	Linux 2.6.27.8 for LPC3250/Phytec 3250

Always rebuild the kernel	[*]
Procedure cscope index	[]
Include kernel headers	[]
Configure the kernel	[*]
Leave the kernel sources after building	[*]

3.2.4 Root filesystem package selection

Select the “Package list - - - >” option from the menu to enter the package list submenu. Make sure the following packages are selected in the menu with the following options.

Package selection menu	Value to select or enter
Busybox	[*]
Device nodes - - - >	Static device files
Module dependencies	[*]
Skeleton base files	[*]

3.2.5 Target system configuration

Select the “Target system configuration - - - >” option from the menu to enter the target configuration submenu. Select the options as shown below.

Target system configuration option	Value to select or enter
Target hostname	nxp
Boot up with a tty and login	[]
Load these modules at boot	()
Start networking	[*]
Network setup - - - >	[]
Set the system time at startup	[]
Start syslogd/klogd	[*]
Start inetd	[]

Network setup - - - > menu option	Value to select or enter
enable Ethernet 0	[*]
Interface	[eth0]
Get network parameters using DHCP	[*]
enable Ethernet 1	[]
enable Ethernet 2	[]
enable Ethernet 3	[]
DHCP client startup	udhcpc -b -i

3.2.6 Target image generation setup

Select the “Target image generation - - - >” option from the menu to enter the target image generation submenu. Select the options as shown below.

Target image generation option	Value to select or enter
Target image - - - >	NFS only
Run a command after building the target image	[]
Read-only root filesystem	[]

3.3 Starting the build cycle

After all the LTIB settings have been configured, select Exit from the LTIB settings menu and save the new configuration. LTIB will now start gathering the necessary packages from the internet and installing and building them. Although this process is mostly automated, there may be one or more situations where input is needed from the user before continuing.

One of these occurrences is when the Linux kernel build is started. Prior to the kernel being built, a kernel configuration menu will appear. Although a pre-built kernel has been setup and selected for you with most of the peripherals enabled for the Phytex 3250 board, a few options need to be verified and possibly changed prior to building the kernel.

3.3.1 Selecting board variants

There are several different versions of the Phytex 3250 board hardware. These versions can be identified by examining the part and version numbers of the CPU module, carrier board, and LCD module.

The hardware will have version numbers as follows:

Phytex hardware	Part and version number
CPU module	1304.0 or 1304.1
Carrier board	1305.0, 1305.1, 1305.2, or 1305.3
LCD module	1307.0 or 1307.1

In the Linux kernel configuration menu, go to the System Type - - - > submenu and then the LPC32xx implementations - - - > submenu. Go into each revisions submenu and select the matching board version ID for each piece of hardware. Make sure only one option is selected in each revision category. When the revisions on the submenus match the hardware revisions, exit the menu by continuously selecting Exit. Be sure to save the config when asked. Once you exit the menu, the build process will complete.

At this point, the build may continue for minutes or hours. When the build is complete, continue to the next step.

4 Deploying Linux to your Phytex board using NFS

Regardless of how Linux is deployed to the target board, a bootloader needs to be installed. u-boot installation and configuration is explained in this step, as well as NFS boot setup for the target. Finally, host side configuration is explained and the Linux is booted on the board.

4.1 Environment setup

To setup the Phytex hardware, start by plugging in the CPU module into the carrier board. If an LCD module is available, plug it into the LCD connector on the carrier board. Connect an Ethernet cable between the target board and an ethernet switch, with Ethernet switch also connected to the host machine and a DHCP server (if configured for DHCP) available somewhere on the network. Connect a serial port between the bottom serial connector on the Phytex board to the host PC's serial port.

4.2 u-boot

4.2.1 Installing u-boot on the Phytex board

Locate the u-boot.bin file that was built with LTIB. This file is usually located in the rootfs/boot directory where LTIB was installed. Copy the u-boot.bin file to the root directory of an SD/MMC card. Once u-boot.bin has been copied, plug the SD/MMC card into the SD/MMC slot of the Phytex 3250 board. Start *minicom* on the host system and configure the minicom serial port for 115.2K, 8 data bits, no parity, and 1 stop bit.

Power on the board and the “Phytex 3250 board” message should appear with a prompt waiting for input. *If the message and prompt do not appear, a bootloader may not be installed on the board. See the documentation included with the board or the CDL documentation on NXP's website for information on how to reflash the bootloader. (<http://www.standardics.nxp.com/support/software/lpc32xx.cdl.drivers/>).*

At the prompt, type “load blk u-boot.bin raw 0x83fc0000” and press enter. The u-boot image should transfer from the SD card to the Phytex's memory at address 0x83fc0000. Then enter the “nsave” command at the prompt to save the image into NAND FLASH. Enter “aboot flash raw 0x83fc0000” to setup the board to automatically boot u-boot.bin everytime the board is reset or powered on. Last of all, change the prompt and boot timeout to 2 seconds by entering “prompt linux> 2” at the prompt.

Remove the SD/MMC card. Power cycle or reset the board and u-boot.bin should boot after a few seconds and the u-boot prompt will appear. *If you want, you can keep the SD/MMC card plugged into the board, but it isn't needed during boot of u-boot anymore.*

The sequence is shown below (your output may differ slightly):

```
5
Phytec LPC3250 board
Build date: Dec  4 2008 09:45:09

phy3250>load blk u-boot.bin raw 0x83fc0000
File loaded successfully

phy3250>nsave

phy3250>aboot flash raw 0x83fc0000
Autoboot configuration updated

phy3250>prompt linux> 2

linux>5
Phytec LPC3250 board
Build date: Dec  4 2008 09:45:09
Autoboot in progress, press any key to stop..

U-Boot 1.3.3 (Dec 10 2008 - 10:39:25)

DRAM:  64 MB
NAND:  64 MiB
In:     serial
Out:    serial
Err:    serial
Hit any key to stop autoboot:  0
uboot>
```

u-boot has been successfully installed on the board and now needs to be configured.

4.2.2 u-boot configuration

Before u-boot will boot Linux, the u-boot environment needs to be setup to load and start the Linux kernel. Linux kernel command line parameters also need to be setup to mount the root filesystem over the network using NFS.

4.2.2.1 Booting the kernel image via TFTP

To setup u-boot to boot the kernel via TFTP over the network, first determine the network IP address of the host machine. *For this guide, the example address of 192.168.1.100 is used, but yours may differ.* Once you have the host IP address, enter the following commands at the u-boot prompt to setup TFTP boot.

```
setenv bootfile uImage
setenv bootcmd 'tftpboot; bootm'
setenv bootdelay 1
setenv fileaddr 80100000
setenv loadaddr 0x80100000
setenv serverip 192.168.1.100
saveenv
```

At this point, the target is ready, but the host machine still needs to be setup. *When the target boots and gets its IP address from DHCP, it will be in the same IP address range as the host, ie. 192.168.1.109. Gateway, netmask, and IP addresses will be setup automatically with DHCP but they can also be manually set by setting the appropriate u-boot environment variables and replacing the dhcp command with tftpboot.*

4.2.2.2 U-boot Linux kernel command line parameters

When booting the kernel, a command line is typically passed to the kernel to dynamically configure some options. The kernel will be configured via the command line to mount the root filesystem using NFS, with console I/O on ttyS0.

Enter the following command at the u-boot prompt to setup how the Linux kernel boots.
Setenv bootargs 'console=ttyS0,115200n81 root=/dev/nfs rw
nfsroot=192.168.1.100:<LTIB-PATH>/rootfs ip=dhcp init=/sbin/init'

The IP address '192.168.1.100' should be replaced with your host machine IP address, while the <LTIB-PATH> value should be replaced with the path to the LTIB generated root filesystem on your host system. For example, the path where LTIB was installed might be /home/user/ltib.

Once the bootargs environment variable is setup, type in the command 'saveenv' to store the bootargs value in FLASH. At this point, if everything is set up correctly, the target system is ready to boot and only the host system needs to be setup. Power off the target board for now.

4.3 Host side configuration for Linux

In the previous step, the target board was setup to boot the kernel image from the host machine using tftp. After the kernel has been booted, the root filesystem is mounted on the target using NFS.

4.3.1 Setting up host TFTP to server the kernel image

It's beyond the scope of this guide to explain how to setup a tftp server on your host machine. However, there are many good websites for explaining how to setup a tftp server.

As a general setup procedure, a tftp file area is setup by editing the /etc/xinet.d/tftp file. Several import flags are shown below:

```
disable      = no
server_args  = -s /tftpboot/
```

The /tftpboot/ directory is the specified directory where your files are stored for the tftp server. If the tftp file needs to be changed, be sure to restart the tftp server (or enable it if it is stopped). Either copy your uImage file or link to it in the /tftpboot/ directory.

When the target board powers up, it will request the file 'uImage' from the host via tftp. Setup up your tftp server to download the image to the target board when requested.

4.3.2 Setting up host NFS server

Like tftp, it is beyond the scope of this guide to setup the NFS server. As a general setup procedure, the `/etc/exports` file needs to be edited to add the LTIB rootfs directory for NFS. The change will appear similar to the entry below:

```
<LTIB-PATH>/rootfs 192.168.1.* (rw,no_root_squash,no_subtree_check,sync)
```

Your LTIB-PATH and IP address may be different depending on your setup. After the edits are made, you will need to restart the NFS server (or enable it if it is disabled).

The host NFS server needs to be made aware of the LTIB root filesystem path. Once this is setup, the target board and host system are ready.

5 Linux boot

If everything has been setup correctly, the board should boot u-boot, download and boot the Linux kernel via tftp, and then mount the root filesystem via NFS. To start this procedure, simply power on the board. After Linux boot is complete, a shell prompt should appear on the minicom terminal where you can execute programs. The output of the boot sequence is similar to the output shown in section 5.2.

5.1 U-boot environment setup

The following example shows the u-boot environment setup for my target machine. This was generated with the u-boot `printenv` command. *Note the underlined bootargs variable extends across multiple lines.*

```
u-boot> printenv
baudrate=115200
ethaddr=DE:AD:BE:EF:00:01
gatewayip=192.168.1.1
ipaddr=192.168.1.193
bootdelay=1
loadaddr=0x80100000
bootfile=uImage
filesize=17CCF8
fileaddr=80100000
gatewayip=192.168.1.1
netmask=255.255.255.0
ipaddr=192.168.1.193
serverip=192.168.1.51
bootcmd=dhcp; bootm
bootargs=console=ttyS0,115200n8 root=/dev/nfs rw
nfsroot=192.168.1.51:/home/usb10132/dev/ltib/rootfs ip=dhcp
stdin=serial
stdout=serial
stderr=serial
```

Environment size: 424/16380 bytes
uboot>

5.2 Linux boot

```
5
Phytec LPC3250 board
Build date: Dec  4 2008 09:45:09
Autoboot in progress, press any key to stop..

U-Boot 1.3.3 (Jan  8 2009 - 19:06:25)

DRAM:  64 MB
NAND:  64 MiB
In:     serial
Out:    serial
Err:    serial
Hit any key to stop autoboot:  0
      HW MAC address:  00:50:C2:8C:ED:B9
ENET:auto-negotiation complete
ENET:Link status up
ENET:FULL DUPLEX
ENET:100MBase
BOOTP broadcast 1
DHCP client bound to address 192.168.1.62
TFTP from server 192.168.1.51; our IP address is 192.168.1.62
Filename 'uImage'.
Load address: 0x80100000
Loading:
#####
#####
done
Bytes transferred = 1695024 (19dd30 hex)
## Booting kernel from Legacy Image at 80100000 ...
   Image Name:   Linux Kernel Image
   Image Type:   ARM Linux Kernel Image (uncompressed)
   Data Size:    1694960 Bytes =  1.6 MB
   Load Address: 80008000
   Entry Point:  80008000
   Verifying Checksum ... OK
   Loading Kernel Image ... OK
OK

Starting kernel ...

Uncompressing
Linux.....
..... done, booting the kernel.
Linux version 2.6.27.8 (usb10132@fed_kv.WORKGROUP) (gcc version 3.4.5)
#18 PREEMPT Thu Jan 8 18:32:36 PST 2009
CPU: ARM926EJ-S [41069264] revision 4 (ARMv5TEJ), cr=00053177
Machine: Phytec 3250 board with the LPC3250 Microcontroller
Memory policy: ECC disabled, Data cache writeback
CPU0: D VIVT write-back cache
©2006-2007 NXP Semiconductors. All rights reserved.
```

```
CPU0: I cache: 32768 bytes, associativity 4, 32 byte lines, 256 sets
CPU0: D cache: 32768 bytes, associativity 4, 32 byte lines, 256 sets
Built 1 zonelists in Zone order, mobility grouping on. Total pages:
16256
Kernel command line: console=ttyS0,115200n8 root=/dev/nfs rw
nfsroot=192.168.1.51:/home/usb10132/dev/ltib/rootfs ip=dhcp
PID hash table entries: 256 (order: 8, 1024 bytes)
Console: colour dummy device 80x30
Dentry cache hash table entries: 8192 (order: 3, 32768 bytes)
Inode-cache hash table entries: 4096 (order: 2, 16384 bytes)
Memory: 64MB = 64MB total
Memory: 61340KB available (3168K code, 236K data, 104K init)
Calibrating delay loop... 132.71 BogoMIPS (lpj=663552)
Mount-cache hash table entries: 512
CPU: Testing write buffer coherency: ok
net_namespace: 288 bytes
NET: Registered protocol family 16
Hardware descriptor info:
  DRAM config word: 0x00000008
  syscfg word:      0x00000000
  fieldval word:    0x000a3250
  MAC address:      <6>00:50:c2:8c:ed:b9
LPC32XX DMA driver
SCSI subsystem initialized
usbcore: registered new interface driver usbfs
usbcore: registered new interface driver hub
usbcore: registered new device driver usb
NET: Registered protocol family 2
IP route cache hash table entries: 1024 (order: 0, 4096 bytes)
TCP established hash table entries: 2048 (order: 2, 16384 bytes)
TCP bind hash table entries: 2048 (order: 1, 8192 bytes)
TCP: Hash tables configured (established 2048 bind 2048)
TCP reno registered
NET: Registered protocol family 1
NetWinder Floating Point Emulator V0.97 (double precision)
JFFS2 version 2.2. (NAND) © 2001-2006 Red Hat, Inc.
msgmni has been set to 119
io scheduler noop registered
io scheduler anticipatory registered
io scheduler deadline registered
io scheduler cfq registered (default)
CLCD: Phytex LCD hardware, QVGA portrait display
Console: switching to colour frame buffer device 30x40
Serial: 8250/16550 driver4 ports, IRQ sharing disabled
serial8250.0: ttyS0 at MMIO 0x40090000 (irq = 9) is a 16550A
console [ttyS0] enabled
serial8250.0: ttyS1 at MMIO 0x40080000 (irq = 7) is a 16550A
serial8250.0: ttyS2 at MMIO 0x40088000 (irq = 8) is a 16550A
serial8250.0: ttyS3 at MMIO 0x40098000 (irq = 10) is a 16550A
lpc32xx_hsuart.0: ttyTX0 at MMIO 0x40014000 (irq = 26) is a
lpc32xx_hsuart
lpc32xx_hsuart.0: ttyTX1 at MMIO 0x40018000 (irq = 25) is a
lpc32xx_hsuart
```

```
lpc32xx_hsuart.0: ttyTX2 at MMIO 0x4001c000 (irq = 24) is a
lpc32xx_hsuart
loop: module loaded
LPC32XX_mii_bus: probed
eth0: LPC32XX mac at 0x31060000 irq 29
eth0: attached PHY driver [SMSC LAN8700] (mii_bus:phy_addr=0:00, irq=-
1)
Driver 'sd' needs updating - please use bus_type methods
NAND device: Manufacturer ID: 0x20, Chip ID: 0x36 (ST Micro NAND 64MiB
1,8V 8-bit)
Scanning device for bad blocks
Bad eraseblock 427 at 0x006ac000
Bad eraseblock 830 at 0x00cf8000
Bad eraseblock 1010 at 0x00fc8000
Bad eraseblock 1089 at 0x01104000
Bad eraseblock 1342 at 0x014f8000
Bad eraseblock 1784 at 0x01be0000
Bad eraseblock 3357 at 0x03474000
Creating 4 MTD partitions on "NAND 64MiB 1,8V 8-bit":
0x00000000-0x00168000 : "phy3250-boot"
0x00168000-0x00190000 : "phy3250-ubt-prms"
0x00190000-0x00590000 : "phy3250-kernel"
0x00590000-0x04000000 : "phy3250-rootfs"
at25 spi0.0: 32 KByte at25256a eeprom, pagesize 64
usbmon: debugfs is not available
I2C device at address 0x2c<6>ISP1301 Vendor ID : 0x04cc
ISP1301 Product ID : 0x1301
ISP1301 Version ID : 0x0210
usb-ohci usb-ohci: at 0xf3120000, irq 0
usb-ohci usb-ohci: pnx4008 OHCI
usb-ohci usb-ohci: new USB bus registered, assigned bus number 1
usb-ohci usb-ohci: irq 59, io mem 0xf3120000
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 libusual
input: LPC32xx Touchscreen as /class/input/input0
rtc-lpc32xx rtc-lpc32xx: rtc core: registered rtc-lpc32xx as rtc0
i2c /dev entries driver
PNX4008-WDT: PNX4008 Watchdog Timer: heartbeat 19 sec
mmci-pl18x: DMA buffer(10000 bytes), P:0x839f0000, V:0xffc40000
mmc0: MMCI rev 0 cfg 00 at 0x0000000020098000 irq 15,13
usbcore: registered new interface driver usbhid
usbhid: v2.6:USB HID core driver
Advanced Linux Sound Architecture Driver Version 1.0.17.
ASoC version 0.13.2
UDA1380 Audio Codec 0.6<6>asoc: UDA1380 <-> lpc3xxx-i2s1 mapping ok
ALSA device list:
 #0: LPC3XXX_I2S_UDA1380 (UDA1380)
TCP cubic registered
NET: Registered protocol family 17
```

```
RPC: Registered udp transport module.
RPC: Registered tcp transport module.
ieee80211: 802.11 data/management/control stack, git-1.1.13
ieee80211: Copyright (C) 2004-2005 Intel Corporation
<jketreno@linux.intel.com>
rtc-lpc32xx rtc-lpc32xx: setting system clock to 1970-01-01 00:58:08
UTC (3488)
mmc0: host does not support reading read-only switch. assuming write-
enable.
mmc0: new SD card at address bb08
mmcblk0: mmc0:bb08 SD01G 1006080KiB
  mmcblk0: p1
Sending DHCP requests .., OK
IP-Config: Got DHCP answer from 0.0.0.0, my address is 192.168.1.62
IP-Config: Complete:
    device=eth0, addr=192.168.1.62, mask=255.255.255.0,
gw=192.168.1.1,
    host=192.168.1.62, domain=netscreen-5GT, nis-domain=(none),
    bootserver=0.0.0.0, rootserver=192.168.1.51, rootpath=
Looking up port of RPC 100003/2 on 192.168.1.51
Looking up port of RPC 100005/1 on 192.168.1.51
VFS: Mounted root (nfs filesystem).
Freeing init memory: 104K
init started: BusyBox v1.11.2 ()
starting pid 804, tty '': '/etc/rc.d/rcS'
Mounting /proc and /sys
Setting the hostname to nxp
Mounting filesystems
mount: mounting usbfs on /proc/bus/usb failed: No such file or
directory
Starting syslogd and klogd
Running sysctl
Setting up networking on loopback device:
Setting up networking on eth0:
You need to manually set your nameserver in /etc/resolv.conf
starting pid 842, tty '': '-/bin/sh'
[root@nxp /]#
```