
i.MX28 Linux BSP

User's Guide

Document Number: 924-76408
Rev. 10.12.01
01/2011



How to Reach Us:

Home Page:

www.freescale.com

Web Support:

<http://www.freescale.com/support>

USA/Europe or Locations Not Listed:

Freescale Semiconductor
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:

Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:

Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064, Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:

Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:

Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals", must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale, the Freescale logo, CodeWarrior, ColdFire, PowerQUICC, StarCore, and Symphony are trademarks of Freescale Semiconductor, Inc. Reg. U.S. Pat. & Tm. Off. CoreNet, QorIQ, QUICC Engine, and VortiQa are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. ARM is the registered trademark of ARM Limited.

© 2010. Freescale Semiconductor, Inc.

About This Book	iv
Audience	iv
Organization.....	iv
References.....	iv
Chapter 1 Introduction.....	1-1
1.1 Flash Boot Loader.....	1-1
1.2 Boot Stream	1-1
1.3 Linux Kernel and Driver.....	1-2
1.4 Root File System.....	1-4
Chapter 2 Building the Linux Platform	2-5
2.1.1 Sudores.....	2-5
2.1.2 Update to the latest packages	2-5
2.1.3 Install host packages needed by LTIB.....	2-6
2.2 Installing and Building LTIB.....	2-7
2.2.1 Install LTIB.....	2-7
2.2.2 Configure and build	2-7
2.2.3 LTIB configuration	2-11
2.3 Rootfs over NFS	2-14
2.3.1 Setting Network Parameters.....	2-14
2.3.2 Setting rootfs.....	2-16
2.4 Setting up the Linux Host	2-16
2.5 Building the Manufacturing Firmware (To Be Supported)	2-19
Chapter 3 Configuring the Target Hardware.....	3-20
3.1 External Cabling	3-20
3.2 Board Configuration	3-20
3.2.1 Boot Modes.....	3-20
Chapter 4 Creating Boot Stream Image.....	4-22
4.1 Setting Up the Kernel Command Line	4-22
4.2 Building the boot stream image.....	4-22

Chapter 5 Booting the Target Hardware 5-23

5.1 Target Preparation.....	5-23
5.1.1 Setting Up kernel command line.....	5-23
5.1.2 Rebuilding the Linux image.....	5-25
5.1.3 Writing the Boot Stream and rootfs to a Boot Medium.....	5-25
5.2 Target Boot	5-27
5.2.1 USB Boot	5-27
5.2.2 MMC/NAND Boot	5-28
5.2.3 Network Boot.....	5-28

About This Book

This document explains how to build and install the Freescale Linux board support package (BSP) to the i.MX28-EVK board.

Refer to the *i.MX Family Linux Software Development Kit Reference Manual* (to be released), for more information about installing the BSP and toolchain for the board, building zImage and root file system.

Audience

This document is intended for software, hardware, and system engineers who are planning to use the product. This document can also be used by any person who wants to understand more about the product.

Organization

This document contains the following chapters.

- [Chapter 1](#) Introduction
- [Chapter 2](#) Building the Linux Platform
- [Chapter 3](#) Configuring the Target Hardware
- [Chapter 4](#) Creating Boot Stream Image
- [Chapter 5](#) Booting the Target Hardware

References

- [i.MX Family Linux Software Development Kit Reference Manual](#)

Chapter 1

Introduction

The i.MX Linux BSP is a collection of binary, code, and support files that can be used to create a Linux kernel image and a root file system for the i.MX board.

1.1 Flash Boot Loader

When the i.MX28 is reset, it executes the (Read-only memory) ROM. There is no alternative - no other code is permitted to handle the reset exception. The ROM reads the boot mode pins to detect the boot source (USB, SD/MMC, NAND flash, and so on.) and negotiates with that source in a device-dependent way to retrieve a “boot stream”. A boot stream is an executable collection of bytes in the Safe Boot (SB) format.

1.2 Boot Stream

The boot stream is an important concept for i.MX28.

The Linux release provides two boot stream images as follows:

- Linux kernel boot stream
- U-boot boot stream

Refer to [Chapter 4](#) for more details about creating these boot stream images.

A boot stream contains instructions that cause it to function as an extended boot loader for the ROM. Such a boot stream starts with a Load command that instructs the ROM to copy the executable into memory. A final Jump command instructs the ROM to transfer control to the executable that is loaded.

Another important command is Call. This command instructs the ROM to make a function call to a given address and continue processing the boot stream when the control returns. A Call command is usually preceded by a Load command that copies into memory the function to be called. Collectively, the Load command, the associated executable and the Call command are referred as a bootlet.

Figure 1 shows a boot stream that contains two bootlets, followed by the main executable:



Figure 1. i.MX28 Boot Stream Outline

Each bootlet is an executable that built separately, for a specific purpose, and may or may not know anything about the bootlets that precede or follow it.

The boot stream can instruct the ROM to call any number of executables before the final Jump, depending on the system needs. The i.MX28 Linux BSP boot streams contain the following bootlets:

- power_prep — This bootlet configures the power supply.
- boot_prep — This bootlet configures the clocks and SDRAM.
- linux_prep — This bootlet prepares to boot Linux

Figure 2 shows a kernel boot stream constructed with the i.MX28 Linux BSP:

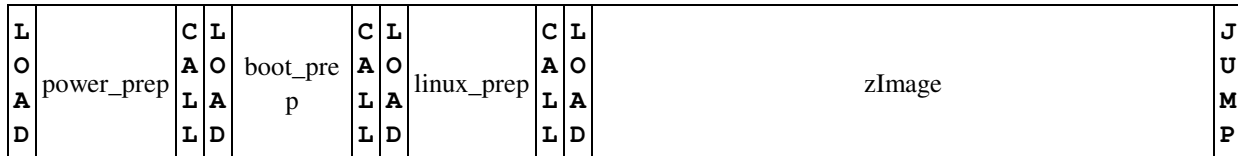


Figure 2. Example of i.MX28 Boot Stream Loading Linux Kernel

Figure 3 shows a U-Boot boot stream:



Figure 3. Example of i.MX28 Boot Stream Loading U-Boot

Refer to Chapter 4 for details about creating a boot stream image.

1.3 Linux Kernel and Driver

At the time of this writing, the latest BSP release is L2.6.35_10.12.01.

The Freescale BSP (L2.6.35_10.12.01_SDK_source_bundle.tar.gz) contains the Freescale Linux 2.6.31 kernel, driver source code and toolchain.

The Freescale BSP images (L2.6.35_10.12.01_SDK_images_MX28.tar.gz) contains a prebuilt kernel image for uboot, prebuild images with bootlets for uboot and linux kernel and filesystem (rootfs-mm.ext2)

Download Prebuild Images from Freescale site

<http://www.freescale.com/imx28tools>

Click on “Linux Binary Demo Files”. Type the following commands

```
$ cd ~
$ mkdir imx28
$ cd imx28
$ mkdir images
$ cd images
$ cp <location of Linux Binary Demo Files
    >/L2.6.XX_XX.XX.XX_SDK_images_MX28.tar.gz .
$ tar -zxvf L2.6.XX_XX.XX.XX_SDK_images_MX28.tar.gz
```

The following files comes into the tar file

- imx28/images/uImage (Kernel Image for uboot)
- imx28/images/imx28_ivt_linux.sb (combine boot steam and kernel image, HAB ENABLED boot stream)
- imx28/images/imx28_ivt_uboot.sb (combine boot steam and uboot image, HAB ENABLED boot stream)
- images/rootfs-mm.ext2 (RAM file system)

If kernel images were compiled (see [chapter 2](#)) files are in

- <ltib directory>/rootfs/boot/uImage (Kernel Image for uboot)
- <ltib directory>/rootfs/boot/imx28_ivt_linux.sb (combine boot steam and kernel image, HAB ENABLED boot stream)
- <ltib directory>/rootfs/boot/imx28_linux.sb (combine boot steam and kernel image, HAB DISABLED boot stream)
- <ltib directory>/rootfs/boot/imx28_ivt_uboot.sb (combine boot steam and uboot image, HAB ENABLED boot stream)
- <ltib directory>/rootfs/boot/imx28_uboot.sb (combine boot steam and uboot image, HAB DISABLED boot stream)
- <ltib directory>/rootfs.jffs2 (file system for flash memories, by default ltib generates this file)

-
- <ltib directory>/rootfs.ext2.gz (file system for RAM. Needs to enable in ltib config menu).

NOTE

If the HAB_DISABLE bit
HW_OCOTP_ROM7:0x8002C210:bit11 is “1”, then use
boot streams without the name “ivt”.

If the HAB_DISABLE bit
HW_OCOTP_ROM7:0x8002C210:bit11 is “0”, then use
boot streams with the name “ivt”.

For more information about HAB module refer to [i.MX28 Reference Manual](#)

1.4 Root File System

The root file system package provides busybox, common libraries, and other fundamental elements. The Linux BSP contains the original root file system package as follows:

If image is from prebuild images

- imx28/rootfs.ext2

If compiled images

- <ltib directory>/rootfs.jffs2 (file system for flash memories, by default ltib generates this file)
- <ltib directory>/rootfs.ext2.gz (file system for RAM, needs to enable in ltib config menu)

Chapter 2

Building the Linux Platform

This chapter explains how to setup the build environment, install and build the Linux Target Image Builder (LTIB), set rootfs for the Network file system (NFS), and setup the host environment. There is script which configures your host with all tools necessary. Install Linux OS using Linux Builder

Install a Linux distribution Ubuntu 10.04 or Ubuntu 10.10 on your system. (Download the ISO file from ubuntu site burn the cd and boot from it. Follow the instructions on the screen)

NOTE

There is a script to make the rest of the setup for you ltib on your host machine (the ubuntu installation.)

Ubuntu_LTIB_setup.sh which comes with the
L2.6.35_10.12.01_SDK_scripts.tar.gz that comes into
L2.6.35_10.12.01_SDK_source_bundle.tar.gz.

Refer to **Ubuntu_LTIB_setup_README.pdf** on the same location for its usage.

2.1.1 Sudoers

Run 'sudo visudo' so you can edit the sudoer's file. Add the following line to the end of the sudoers file. This is needed for people to be able to use LTIB. This assumes that all your developers have administrator priviledges on this host. If that is not the case, a similar line can be added for each user. Suppose your user is user1

```
user1 ALL = NOPASSWD: /usr/bin/rpm, /opt/freescale/ltib/usr/bin/rpm
```

2.1.2 Update to the latest packages

- Open up System -> Administration -> Update Manager
- Click on Settings
- Open the Updates Tab
- Set 'Release upgrade' to 'Never'. That makes the option to upgrade to Karmic go away.
- Close the settings dialog box.

- Click on 'Check' to check for upgraded packages. It will look for packages that are upgraded from the version that is installed on your box.
- Choose to install the upgrades. This will take a while on a freshly installed box.

2.1.3 Install host packages needed by LTIB

This document assumes you are using Ubuntu. Not a requirement, but the packages may be named differently and the method of installing them may be different.

```
#Packages for ltib
$ sudo apt-get install build-essential
    $ sudo apt-get install zlib1g-dev
    $ sudo apt-get install rpm
    $ sudo apt-get install bison$sudo apt-get install m4$sudo
apt-get install liblzo2-2
    $sudo apt-get install liblzo2-dev
    $ sudo apt-get install uuid
    $ sudo apt-get install libuuid1

$ sudo apt-get install gettext
$sudo apt-get install libgtk2.0-dev
$sudo apt-get install ncurses-dev
$sudo apt-get install libdbus-glib-1-dev
$sudo aptitude -y install libdbus-glib-1-dev liborbit2-dev intltool
$sudo aptitude -y install ccache ncurses-dev zlib1g zlib1g-dev libtool
$sudo aptitude -y install uuid-dev liblzo2-dev
$sudo aptitude -y install tcl dpkg
# Packages required for 64-bit Ubuntu
# Do "uname -a" and see if the word "x86_64" shows up. # if " uname -
a|grep -sq 'x86_64'"
    $sudo aptitude -y install ia32-libs libc6-dev-i386 lib32z1

# The following recommended for Linux development.
# They are not required by LTIB.
$sudo aptitude -y install gparted emacs22-nox openssh-server
$sudo aptitude -y install nfs-common nfs-kernel-server lintian
$sudo aptitude -y install git-core git-doc git-email git-gui gitk
$sudo aptitude -y install diffstat indent tofrodos fakeroot doxygen
    uboot-mkimage

$sudo aptitude -y install sendmail mailutils meld atftpd sharutils
$sudo aptitude -y install manpages-dev manpages-posix manpages-posix-dev
    linux-doc
$sudo aptitude -y install vnc4server xvnc4viewer
```

2.2 Installing and Building LTIB

At the time of this document writing the latest BSP release is L2.6.31_0.08.01. 'XX' will be used through this document to specify the changes resources versions. If you are using newer versions change the 'XX' for the corresponding characters on your file.

2.2.1 Install LTIB

To install and build LTIB, perform the following steps:

NOTE

In some Linux systems, the following procedure must be performed with root permissions. However, these instructions are for performing the procedure not as root.

1. Install the LTIB package not as root:

```
$ cd ~
$ mkdir imx28 #if not already created
$ cd imx28
$ cp <location of Linux Binary Demo Files
    >/L2.6.XX_XX.XX.XX_SDK_images_MX28.tar.gz .
$ tar zxf L2.6.XX_XX.XX.XX_SDK_source_bundle.tar.gz
# It will uncompress three files
# 1 L2.6.XX_XX.XX.XX_SDK_source.tar.gz
# 2 L2.6.XX_XX.XX.XX_SDK_scripts.tar.gz
# 3 L2.6.XX_XX.XX.XX_SDK_docs.tar.gz
# uncompress source file
$tar zxf L2.6.XX_XX.XX.XX_SDK_source.tar.gz
$ cd L2.6.XX_XX.XX.XX_SDK_source
$ ./install
```

The installer will ask you to read and accept a EULA and then ask you for the install directory. For this example, choose directory '~/'imx28'.

After the installation, there will be a new directory named "ltib" under the install directory you specified. This new directory contains the actual ltib Perl script, config files, package specs, and utilities.

From this point on, the LTIB install directory will be referred to as "~/'imx28/ltib" in example command lines, assuming you installed LTIB into your home directory.

2.2.2 Configure and build

To configure and build your LTIB installation the first time, run this command from the ~/'imx28/ltib directory:

```
$ ./ltib
```

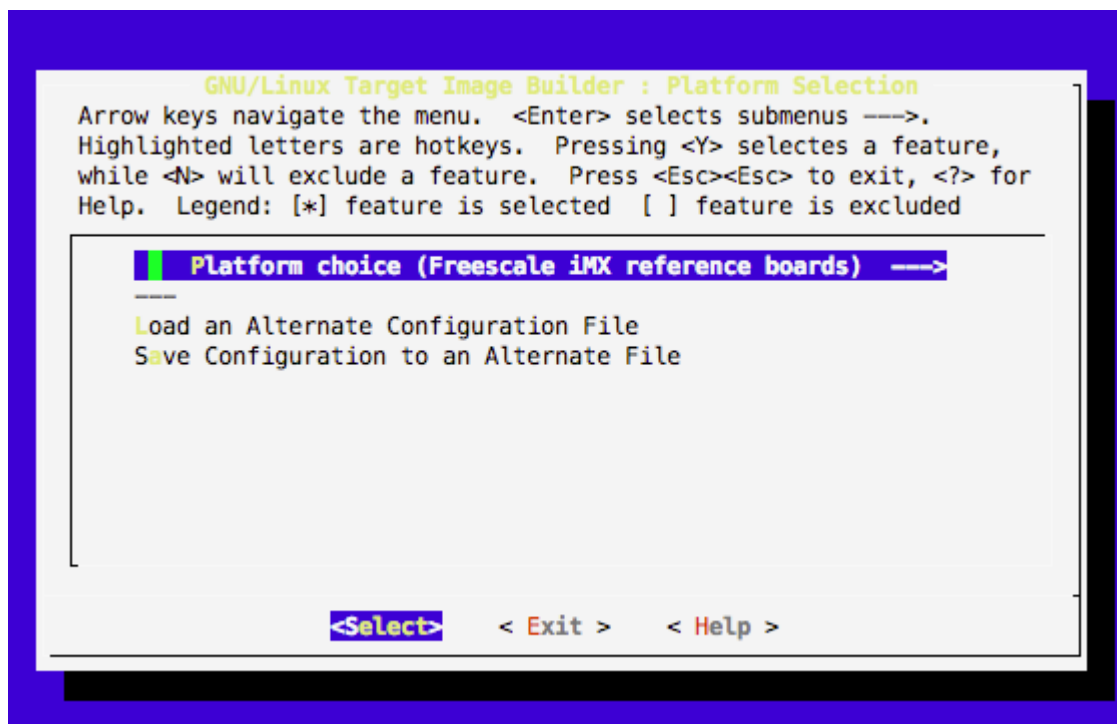
This command invokes LTIB with its default behavior of performing a build. Since the installation has not yet been configured, LTIB will present the curses configuration screen before building. Then, when you exit the configuration screen, LTIB will build the target image. Also, the very first time you run LTIB on a system, it will build and install a number of host tools that it places under /opt/freescale.

To ask LTIB to show the configuration screen without building afterwards, run this command:

```
$ ./ltib -m config
```

2.2.2.1 First time configuration

The first time you configure a given LTIB installation, you will be asked to select the platform. The first screen that appears looks like this:

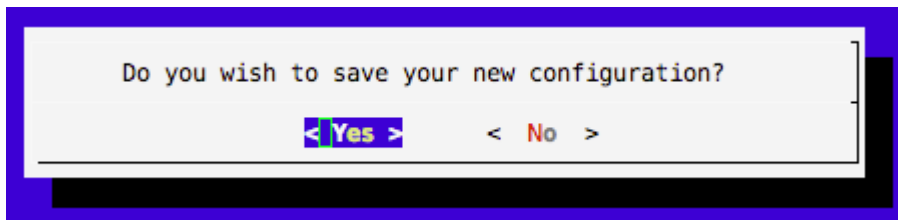


Because the i.MX BSP contains only the i.MX platform, you don't need to change anything on this screen.

On all of the curses configuration screens for both LTIB and the kernel (they use the same mconf system), there are a standard set of keys that are important to know. The bottom of each main screen shows **<Select>** and **<Exit>** items. The right and left arrows move between these two items.

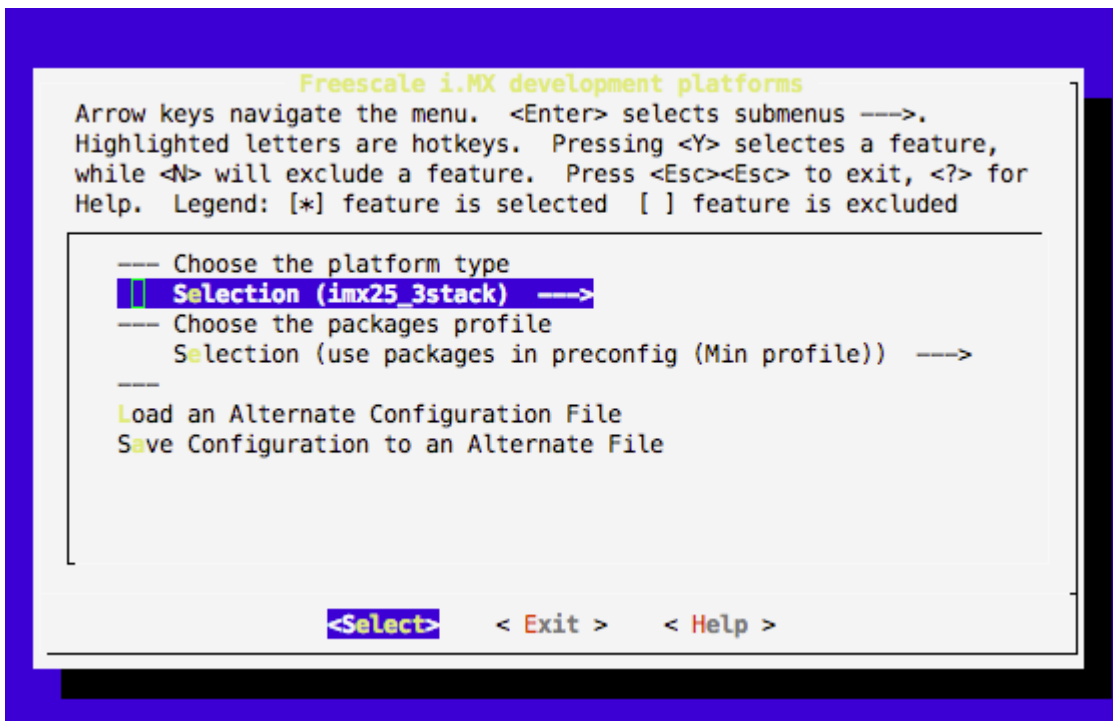
When **<Select>** is highlighted, the up and down arrows move between items in the main list. Pressing return on any item with an arrow to the right will open up a sub-screen. Some sub-screens appear as windows on top of the current screen, and some appear as their own screens.

When **<Exit>** is selected, you can hit the return key to exit the current screen. On most main screens, exiting will bring up the following dialog asking if you want to save changes.



Here, use the right and left arrow keys to select between **<Yes>** and **<No>**. Press return to activate your selection. Selecting **<No>** will cause LTIB not to continue to the following step.

Exit the platform selection screen and save the configuration when asked. This will bring up another sub-platform selection screen, shown here:

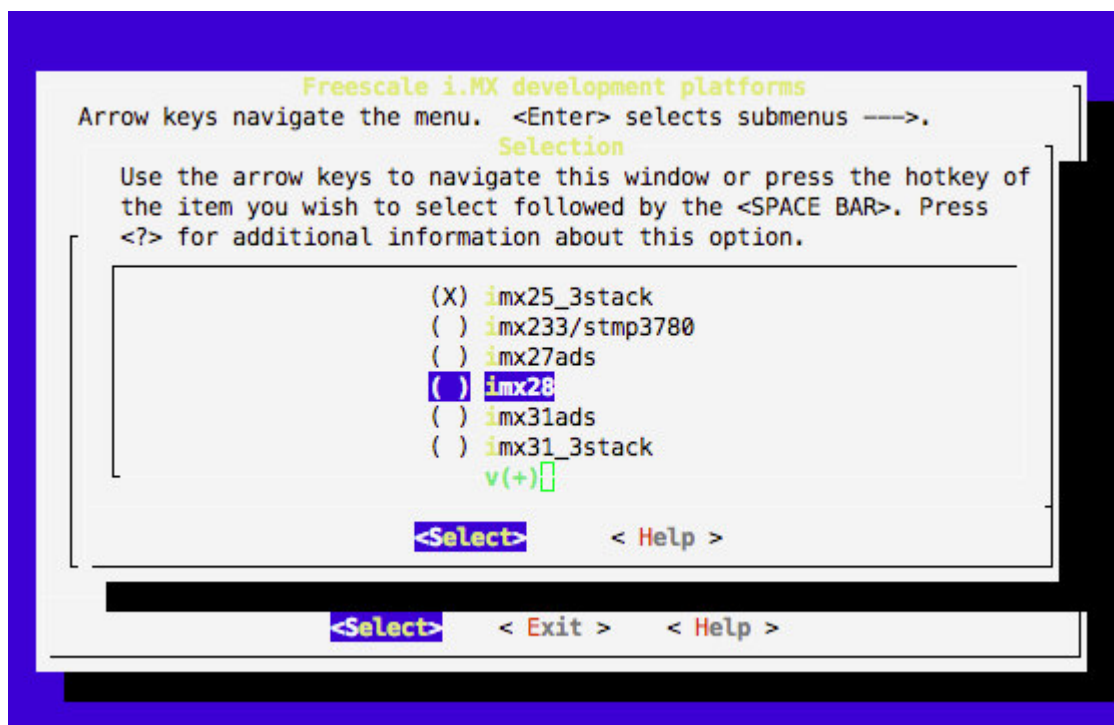


The first option available on this screen is used to choose the sub-platform type. For the i.MX platform, the sub-platform is the specific chip family combined with the board type. For the i.MX28, this is the EVK board. The second option on this screen, under the **Choose the packages profile** item, lets you select the default set of packages. Individual packages can be enabled and disabled later, as you need.

If you need to change the sub-platform later, use the command:

```
$ ./ltib -m selectype
```

Move the cursor down to the Selection item below the Choose the platform type line and press return to open up a platform selection menu, shown here:



Select the imx28 platform and press return to return to the main screen.

Exit the sub-platform selection screen and save the configuration. This will bring you to the main LTIB configuration screen ([section 2.2.3](#)).

2.2.3 LTIB configuration

The screen below shows the full contents of the main LTIB configuration screen for the i.MX28. If you come from section below this window will show up automatically.

To get into this window type on your "~/imx28/ltib" directory:

```
$./ltib -m config
```

```

                                     Freescale iMX28 Based Boards
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selects a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help. Legend: [*] feature is selected [ ] feature is excluded

(iMX28) Platform
--- LTIB settings
    System features --->
--- Choose the target C library type
    Target C library type (glibc) --->
    C library package (from toolchain only) --->
    Toolchain component options --->
--- Toolchain selection.
    Toolchain (ARMv5te gcc-4.1.2,Multi-lib,gnueabi/glibc-2.5-nptl-3)
(-O2 -fsigned-char -msoft-float) Enter any CFLAGS for gcc/g++
--- Choose your bootloader for U-Boot
    bootloader (u-boot) --->
--- Choose your board
--- Choose your Kernel
    kernel (Linux 2.6.31-imx) --->
[ ] Always rebuild the kernel
[ ] Produce cscope index
[*] Include kernel headers
[ ] Configure the kernel
[*] Leave the sources after building
[ ] Build mfg firmware (NEW)
--- Package selection
    Package list --->
--- Target System Configuration
    Options --->
--- Target Image Generation
    Options --->
v(+)

<Select> < Exit > < Help >
```

Most items in the main configuration screen can be ignored unless you have a specific reason to modify them. Two exceptions are items related to the kernel: **Always rebuild the kernel** and **Configure the kernel**. If you enable Configure the kernel, then the kernel configuration screen will be shown the next time the kernel is built.

Another important items on the main screen are **Options** for **Target System Configuration** and **Target Image Generation** and **Package List**.

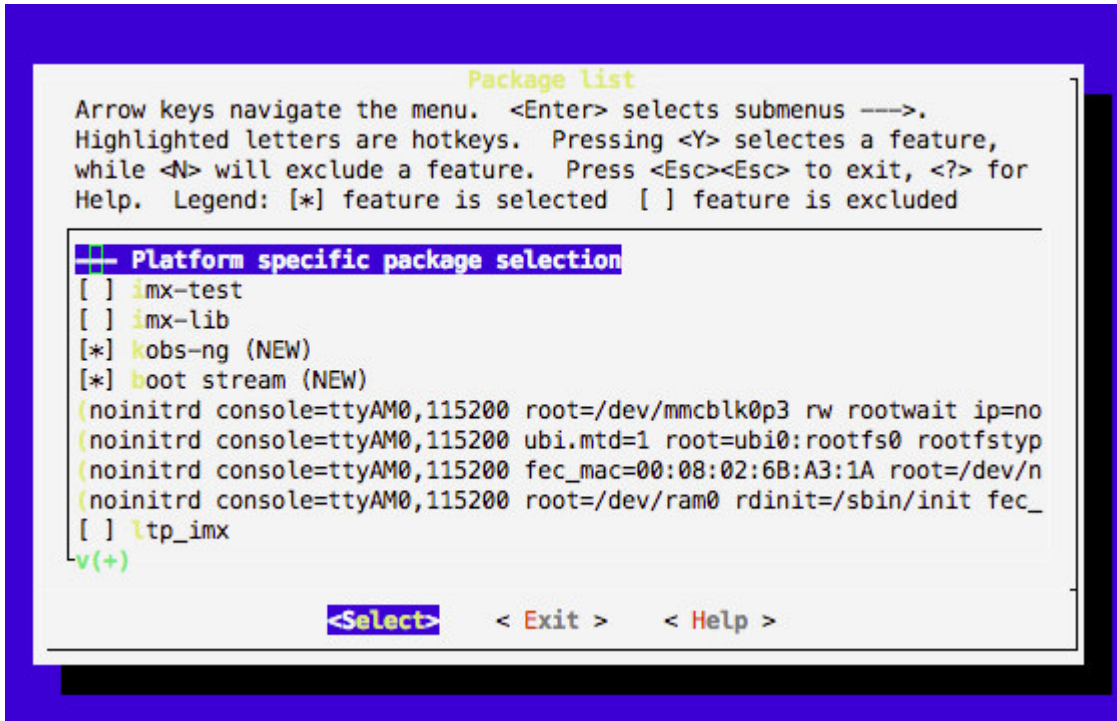
```
Options
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selects a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help. Legend: [*] feature is selected [ ] feature is excluded

(mx28evk) target hostname
[*] boot up with a tty and login
  (:::respawn:/sbin/getty -L ttyAM0 115200 vt100) Enter your inittab sta
() load these modules at boot
[*] start udevd
[ ] start networking
[ ] set the system time at startup
[ ] start syslogd/klogd
[*] start inetd
() Enter command line arguments for inetd startup
v(+)
```

Options for Target System Configuration

Note

You do not actually have to enable start networking in order to boot from NFS or using networking on the target.



Options for Target Image Generation

Note

The package list screen is also where you set the kernel command line options for the i.MX28 when booting Linux directly. When using U-boot, you set the kernel command line in U-boot itself.

As shown in the above screen, there are four kernel command lines listed under the **boot stream package**. The first command line is the default, while the other three are optional ones selected by pressing certain keys during system boot. To edit a command line, move the cursor over it and press return. Refer to [section 5.1.1](#) to see the options about what to edit on those lines.

Finally, you can modify settings for individual packages and choose which packages to install by selecting the **Package list** item in the main screen.

If this is your first time you run ltib the compilation will start automatically. If not, to start the compilation run the following command from your "~/imx28/ltib" directory :

```
$ ./ltib
```

When complete, LTIB will have produced the following in subdirectories of ltib:

- The kernel images roofs/boot/uImage and roofs/boot/zImage.

- The SB files of bootlets and kernel images `rootfs/boot/imx28_linux.sb`, `rootfs/boot/imx28_ivt_linux.sb`.
- The SB files of bootlets and uboot images `rootfs/boot/imx28_uboot.sb`, `rootfs/boot/imx28_ivt_linux.sb`.
- The Jffs2 rootfs image `rootfs.jffs2`

NOTE

See section 1.3 to see details on those files

NOTE

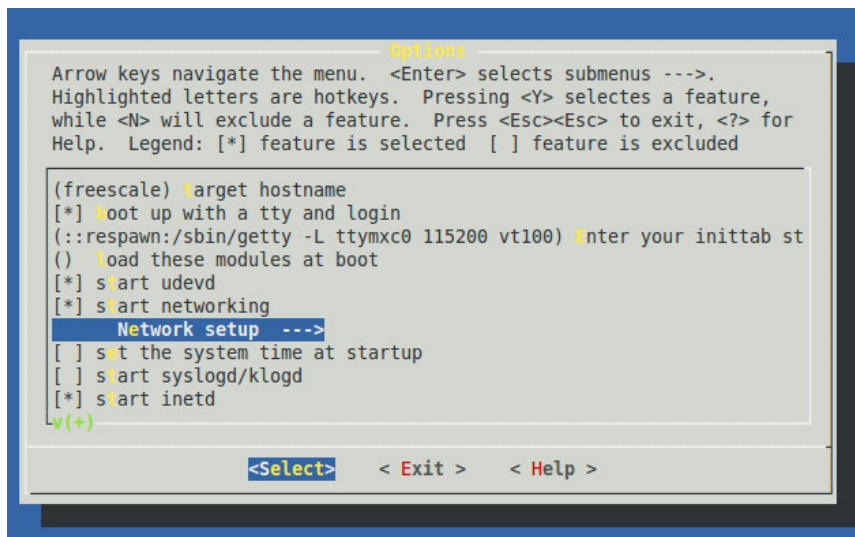
If you want an ext2 file system, execute `./ltib -c`, and change the option under the LTIB “Target Image Generation” menu from “JFFS2” to “EXT2”. After rebuilding the EXT2 rootfs image can be found in `rootfs.ext2.gz`

2.3 Rootfs over NFS

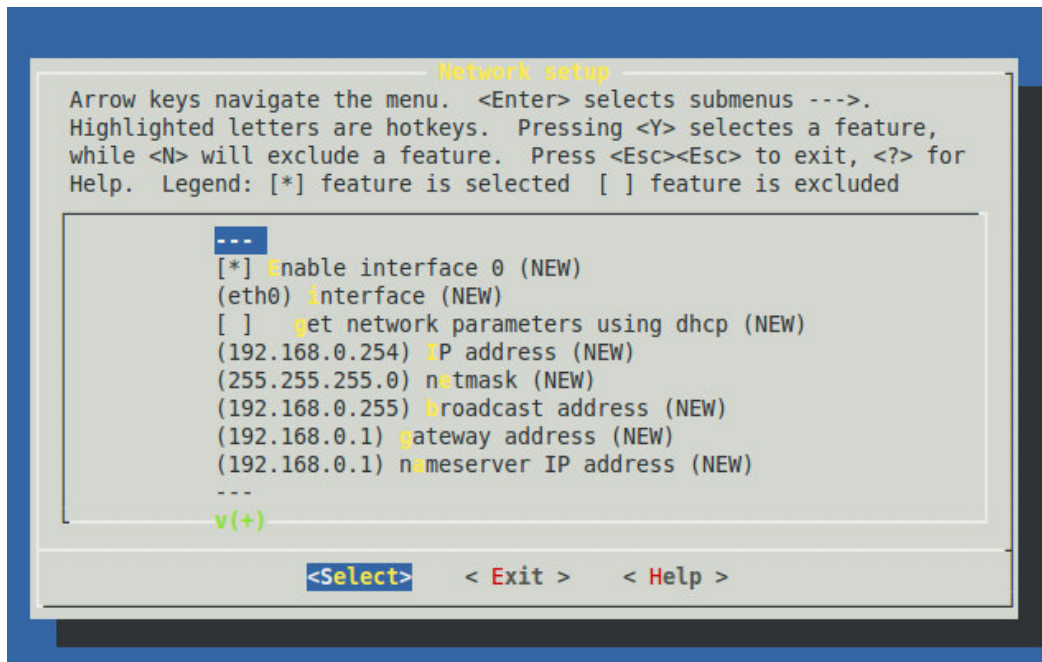
2.3.1 Setting Network Parameters

Setup network parameters in LTIB to boot using the NFS. To set these parameters, execute:

```
$ ./ltib -c
```



Enable networking by pressing space on **start networking** on **Options for Target System Configuration** (see [section 2.2.3](#))



Enable **eth0 interface** by pressing space. Configure the network parameters. Configuration is defaulted to private network class C (192.168.0.0/16 (255.255.255.0)), but you can choose any network configuration. Choose to use either dhcp to obtain the ip from a router or set static ip (as defaulted class c).

IP Address: Board IP address for interface Ethernet 1 (eth0).

netmask: network mask for this ip family segment.

broadcast address: used to define the broadcast segment. (not necessary to configure)

gateway address: default gateway address , usually configured to be your host machine.

nameserver ip : If configured on your network , the DNS to resolve URL addresses. Usually configured to be your host machine (not necessary to configure)

2.3.2 Setting rootfs

There are two places in the BSP to get the rootfs for NFS:

- Use the ext2 format rootfs package already provided in the distribution; or
- Use the rootfs that is created after making the build of the kernel

2.3.2.1 Using the rootfs Package in the Distribution

Use the following commands to set the `rootfs` directory for NFS (When using the prebuild images). Refer to section [2.4](#) to see how to export (make visible) the directory for nfs server.

```
$sudo mkdir /mnt/rootfs
$sudo mkdir /tools
$sudo chmod 777 /tools
$sudo chown nobody /tools
$cp imx28/rootfs.ext2.gz /tools
$cd /tools
$gunzip rootfs.ext2.gz
$sudo mount -o loop -t ext2 rootfs.ext2 /mnt/rootfs
$sudo cp -rp /mnt/rootfs .
```

2.3.2.2 Using the rootfs Created After the kernel Build

Instead of using the `rootfs.ext2.gz`, use the root file system in `<your LTIB directory>`.

Refer to section [2.4](#) to see how to export (make visible) the directory for nfs server.

Refer to the *i.MX Family Linux Software Development Kit Reference Manual*, for other ways to create a file system image file.

2.4 Setting up the Linux Host

To set up the Linux host system, perform the following steps:

1. Install the **tftp** server.

Type the following commands:

```
$ sudo apt-get install tftpd tftp
$ sudo mkdir /tftproot
$ sudo chmod -R 777 /tftproot
$ sudo chown nobody /tftproot
```

2. Install the **nfs** server.

Type the following commands:

```
$ sudo apt-get install nfs-kernel-server nfs-common
```

3. Edit with your preferred editor `/etc/xinetd.d/tftp` to enable tftp as follows (if it doesn't exist create it. Remember to run command with sudo user or root privileges.):

```
service tftp
{
  protocol = udp
  port = 69
  socket_type = dgram
  wait = yes
  user = nobody
  server = /usr/sbin/in.tftpd
  server_args = /tftpboot
  disable = no
}
```

4. Edit with your preferred editor `/etc/exports` to enable nfs sharing as follows (if it doesn't exist create it. Remember to run command with sudo user or root privileges.)

add this line in the file if you are using prebuild images

```
/tools/rootfs *(rw, sync, no_root_squash)
```

add this line in the file if you are using compiled images

```
~/imx28/ltib/rootfs *(rw, sync, no_root_squash)
```

5. Restart the **nfs** and **tftp** servers on the host:

```
$ /etc/init.d/xinetd restart
$ /etc/init.d/nfs-kernel-server start      #if is first time
$ /etc/init.d/nfs-kernel-server restart   #if already running
```

6. Copy `uImage` and `rootfs.jffs2` in the release package or LTIB to the tftp directory.

If you are using precompiled images

```
$ cp ~/imx28/zImage /tftpboot
$ cp ~/imx28/rootfs.jffs2 /tftpboot
```

If you are using compiled images

```
$ cp ~/imx28/ltib/rootfs/boot/uImage /tftpboot
$ cp ~/imx28/ltib/rootfs.jffs2 /tftpboot
```

If you make changes to file system or kernel after you compile you will need to copy again those files to /tftpboot directory or create a symbolic link to reference them from tftpboot directory.

To create symbolic links

```
$ ln -s ~/imx28/ltib/rootfs/boot/uImage /tftpboot/uImage
$ ln -s ~/imx28/ltib/rootfs/rootfs.jffs2 /tftpboot/rootfs.jffs2
```

7. Testing tftp and nfs servers locally

To test TFTP installation

```
# create a file on the /tftpboot
$ echo hello > /tftpboot/hello.txt
# change to your home directory
$ cd
# get the file hello.txt from the server
$ tftp localhost #open tftp prompt
tftp> get hello.txt
Received 6 bytes in 0.0 seconds
tftp> quit
$ #return to linux prompt
```

To test nfs server

```
# change to home directory
$ cd
# make directory to test
$ mkdir nfstest
# mount directory
$ sudo mount -t nfs 127.0.0.1:/opt/nfsroot nfstest
# should be able to see the files into your shared directories
$ ls nfstest
$ sudo umount nfstest # umount directory
```

NOTE

A Windows tftp program “tftp.zip” is available in the LTIB release package `Common/` folder. This tftp program can be installed in the Windows OS to provide a Windows tftp server for downloading images.

2.5 Building the Manufacturing Firmware (To Be Supported)

Refer to Section 2.2 Installing and Building LTIB, to setup the ltib environment.

Configure the firmware build profile

```
$ ./ltib -selectype
```

Choose the following:

--- Choose the platform type

Selection (**imx28**) --->

--- Choose the packages profile

Selection (**mfg firmware profile**) --->

Save and exit.

After ltib completes the build, Updater.sb and updater_ivt.sb will have been created.

Chapter 3

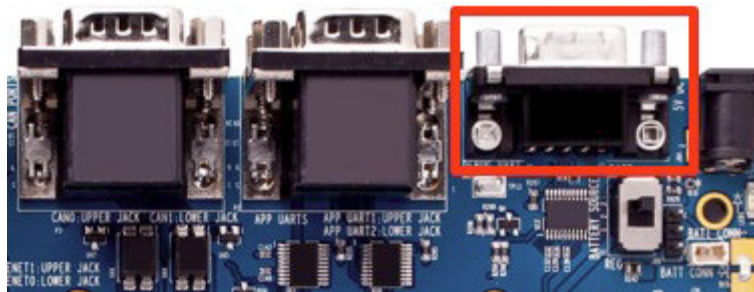
Configuring the Target Hardware

This chapter details all hardware-specific configuration necessary to prepare the i.MX28-EVK development board for use with Linux.

3.1 External Cabling

Perform the following to setup external cabling:

- Plug the Linux host straight serial console cable into the UART DSUB9 connector on the i.MX28-EVK. The debug UART is the rightmost DB9 connector when viewing the EVK from top. Connect the other cable end to your Host Computer. You can use either a standard serial port or a USB-to-serial interface. Open serial terminal (minicom for linux or HyperTerminal for windows) and configure port to 115200 baud, 8N1.



- Plug the Linux host USB A to mini-B USB cable into the mini-B USB connector on the i.MX28-EVK, or connect the power Cable.

3.2 Board Configuration

3.2.1 Boot Modes

The EVK board uses DIP switch S2 to select boot mode. Bits B0, B1, B2, and B3 are labeled on the board silkscreen. [Table 1](#) gives the boot mode values.

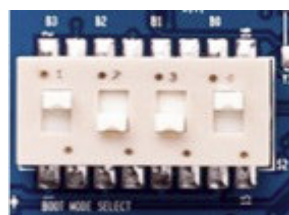


Table 1. Boot Mode Values

B3 /DIP1	B2/DIP2	B1/DIP3	B0 /DIP4	Boot Mode
0	0	0	0	USB0
0	1	0	0	GPMI (NAND)
1	0	0	1	SSP0(SD0)
1	0	1	0	SSP1(SD1)

Refer to the *Hardware User Manual* for detailed EVK board configuration.

NOTE

The i.MX28 EVK board needs hardware rework for booting from SD1, Refer to the *EVK Hardware User Guide*.

Chapter 4

Creating Boot Stream Image

The i.MX28 SoC contains built-in ROM firmware that is capable of loading and executing binary images in special format from different locations, including an MMC/SD card and NAND flash. Such a binary image is called a “boot stream” and consists of a number of smaller bootable images (bootlets) and instructions for the ROM firmware to handle these bootlets (for example, load a bootlet to on-chip RAM and run it).

Refer to the *i.MX Family Linux Software Development Kit Reference Manual*, for kernel configuration and building.

4.1 Setting Up the Kernel Command Line

In LTIB, run the following command, then choose “Package list” and then set default and alternative kernel command lines under the “boot stream” option:

```
$. /ltib -m config
```

4.2 Building the boot stream image

In LTIB, to build a new Linux Kernel and U-Boot boot stream image, give the command:

```
$ ./ltib -p boot_stream.spec -f
```

The output boot stream images are available in `rootfs/boot/` directory, named `imx28_linux.sb` and `imx28_uboot.sb` for HAB-disabled images, and `imx28_ivt_linux.sb` and `imx28_ivt_uboot.sb` for HAB-enabled images. The i.MX28 EVK is shipped with HAB enabled by default.

Chapter 5

Booting the Target Hardware

This chapter explains how to boot the i.MX28 development board for the first time. Linux kernel can be booted on the i.MX28 using the following ways:

- Boot from USB
- Boot from MMC/SD card
- Boot from NAND flash
- Boot from Ethernet (network boot)

All boot modes except network boot are supported by the i.MX28 built-in ROM firmware. The ROM code reads the boot stream image containing the Linux kernel from the first three sources. Network boot of the Linux kernel is performed by the U-Boot boot loader. U-Boot is loaded and started by the ROM firmware through the USB or MMC card or NAND flash.

The Linux SDK provides two boot stream images as follows:

- Linux kernel boot stream
- U-boot boot stream

Refer to [Chapter 4](#) for more details on how to generate a new boot stream image.

5.1 Target Preparation

The sections describe how to prepare and boot the Linux kernel in each boot mode (refer to [section 3.2.1](#) for boot modes).

5.1.1 Setting Up kernel command line

The kernel boot command line can be set in LTIB in the “boot steam” configuration menu under “Package list” -> “boot steam”. These command line options include a default command line and three command lines selected by key-press during system start up. If the command line configuration file has less than four command lines, then the unused entries are replaced by the following default command line string:

```
console=ttyAM0,115200
```

To select the location of the root file system, the “root” command line variable must be configured. There may also be a need to set additional command line options based on the root file system storage type:

- Root file system located on a MMC card partition:

```
root=/dev/mmcblk0p[N] rw rootwait
```

Where N is the number of the MMC card primary partition containing the root file system

- Root file system located on a NAND flash (Jffs2):

```
root=/dev/mtdblock1 rootfstype=jffs2
```

- Root file system on NFS over Ethernet link:

```
ip=dhcp/off/[Target IP] root=/dev/nfs nfsroot=/tools/rootfs
```

Where:

Host IP is the IP address of Ubuntu Linux host

Target IP is the IP address assigned to the i.MX28 development board

- ENET MAC address

Where:

'xx:xx:xx:xx:xx:xx' MAC address of ENET of the EVK board.

NOTE

On the i.MX28 EVK, the MAC address is stored in OTP fuses that have been pre-programmed. If a different the MAC address is to be used, then the following option can be added:

```
fec_mac=xx:xx:xx:xx:xx:x
```

- gpmi or ssp1 selection

```
gpmi/ssp1
```

Where:

gpmi: initialize gpmi (i.e. NAND) interface

ssp1: initialize ssp1 (i.e. SD Card 1) interface

Either a NAND device can be used, or an SD device on SD slot 1 can be used, but not both. This is due to pin-sharing on the i.MX28. SD slot 0 is unaffected by this choice.

5.1.1.1 LTIB boot stream configurations

There are four preset command lines which allow booting the kernel with the root file system located on SD/MMC card, NFS, NAND flash, or RAM disk:

- Default command line for SD (no key press during booting):

```
noinitrd console=ttyAM0,115200 root=/dev/mmcblk0p3 rw rootwait gpmi
```

- Alternative command line 1 for NAND (press KEY1 during booting):

```
noinitrd console=ttyAM0,115200 ubi.mtd=1 root=ubi0:rootfs0 rootfstype=ubifs
rw gpmi
```

- Alternative command line 2 for NFS (press KEY2 during booting):

```
noinitrd console=ttyAM0,115200 fec_mac=00:08:02:6B:A3:1A root=/dev/nfs
nfsroot=10.193.100.213:/data/rootfs_home/rootfs_mx28 rw ip=dhcp rootwait gpmi
```

- Alternative command line 3 for RAM disk (press KEY3 during booting)

```
noinitrd console=ttyAM0,115200 root=/dev/ram0 rdinit=/sbin/init
fec_mac=00:08:02:6B:A3:1A gpmi
```

5.1.2 Rebuilding the Linux image

If the default command lines are modified, then it is necessary to rebuild the release to get the Linux kernel boot stream image with those updated command lines. In LTIB, issue the command:

```
./ltib -p boot_stream.spec -f
```

5.1.3 Writing the Boot Stream and rootfs to a Boot Medium

This section describes how to write the boot stream and rootfs to a boot medium.

5.1.3.1 MMC Boot

The first step is to put a boot image onto your SD/MMC card using the method described below.

The BSP includes the **mk_mx28_sd** (into **L2.6.XX.XX.XX.SDK_scripts.tar.gz** file) shell script that makes it very easy to place the build output onto an SD/MMC card. The script will write the selected boot stream, the rootfs, and a 32MB FAT32 partition to the card.

The **mk_mx28_sd** script should always be run from the LTIB install directory. It knows how to find the boot stream files and rootfs from this directory.

Before running the script, you need to determine the device for your SD/MMC card. By default, the script will not execute with the card already mounted, to prevent any possibility of overwriting a non-removable volume such as a hard drive. So, if you don't already know the device for your card, you must insert the card and determine the device, then remove the card before running the script.

Before inserting the card, run this command to see all available SCSI devices:

```
$ ls /dev/sd?
```

Now insert the card, and run the above command again. The SD/MMC card will be the new **/dev/sdX** device in the output (where the X is a lower-case letter in alphabetic sequence). This is the device name that you pass to **mk_mx28_sd**. Before running the script, be sure to remove the card. Once you already know your card's device, updating the card contents with a new build will be much faster.

To place Linux onto the SD/MMC card:

```
$ cd ~/ltib
$ mk_mx28_sd /dev/sdX
```

To place U-boot onto the SD/MMC card, add the `-u` option to the command line:

```
$ cd ~/ltib
$ mk_mx28_sd -u /dev/sdX
```

After you invoke the script, it will present a prompt for you to insert your SD/MMC card. Once you have done so, type "yes" to have the script continue and format the card.

Set the boot mode on the i.MX28 EVK board for MMC/SD on SSP0 (refer to [section 3.2.1](#))

As an alternative, the Windows tools called `cfimager` can create the partitions, and write the boot stream and rootfs.

```
Cfimager.exe -a -f imx28_linux.sb -e rootfs.ext2 --dual_boot -d <mass storage disk, no “:”, such as H>
```

NOTE

The default rootfs file system released for SD is EXT2 format. EXT2 is not a journaling file system. Any disruption to the file system while syncing can cause a file system error, such as power lost, kernel panic and so on. To avoid such errors, either follow the normal power sequence or use an EXT3 file system. You can convert an EXT2 filesystem to an EXT3 filesystem by using the command `tune2fs -j rootfs.ext2`.

5.1.3.2 NAND Boot

A boot stream image cannot be burned to NAND flash from the Linux host. It is necessary to first load the kernel from an MMC card or network boot. Then, after Linux is running on the board, it is possible to burn the boot stream image to NAND using the “kobs-ng” tool:

- Copy the boot stream to the root file system. For example, in the case of an NFS root, use:

```
cp <where the boostream lives>/iMX28_linux.sb /tools/rootfs
```

- Boot the target and log into it:
- On the target, burn the boot stream image to the flash:

```
#flash_eraseall /dev/mtd0
#kobs-ng init imx28_linux.sb
```

- Copy the jffs2 image to current root file system.
For example, in the case of NFS root, run the following command on the Linux host. Note that the jffs2 image must match the type of flash device in use.

```
cp rootfs.jffs2 /tools/rootfs
```

- On the target, erase the MTD 1 partition:

```
flash_eraseall /dev/mtd1
```

- On the target, burn the jffs2 image from the rootfs to the flash:

```
nandwrite /dev/mtd1 rootfs.jffs2
```

5.1.3.3 Network Boot

Linux kernel network boot is implemented using the U-boot boot loader that is part of the Linux release for Freescale i.MX28. The U-boot boot stream is loaded from SD or USB by sb_loader.exe tool. Refer to [USB Boot](#) and [Network Boot](#) for more details.

This section describes preparation of the host computer.

For rootfs setup Refer to [section 2.3](#) for more details.

5.2 Target Boot

This section describes how to boot the target EVK.

5.2.1 USB Boot

Perform the following to boot from USB:

- Select the USB boot mode (0000) on the DIP switch. Refer to [Chapter 3](#) for more details.
- the i.MX28 development board.
- Press the power key.
- Press KEY1/2/3 to select an alternative boot cmdline, hold the key until the bootlets have run. To use the default boot command line, do not press any key.
- After Windows recognizes the EVK as a USB HID device, run the following command in the Windows console:

```
z:\sb_loader.exe /f imx28_linux.sb  
z:\sb_loader.exe /f imx28_uboot.sb
```

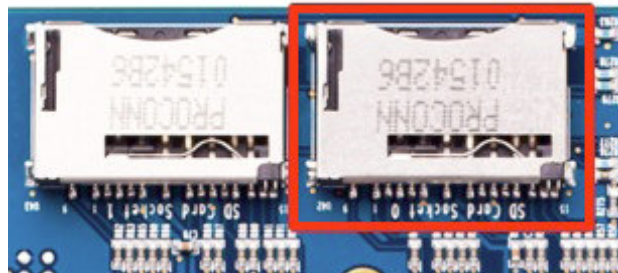
NOTE

Use the “ivt” bootstreams, if the chip is HAB-enabled.

5.2.2 MMC/NAND Boot

Perform the following to boot the MMC/NAND:

- Select a boot mode (SD/MMC:1001, NAND: 0100) using the DIP switch. Refer to [Chapter 3](#) for more details.
- For SD, insert the SD card with i.MX28_linux.sb and rootfs.ext2 burned by cfimage.exe into the SD slot 0. For NAND, the boot stream and rootfs image should be burned into the flash as described previously.



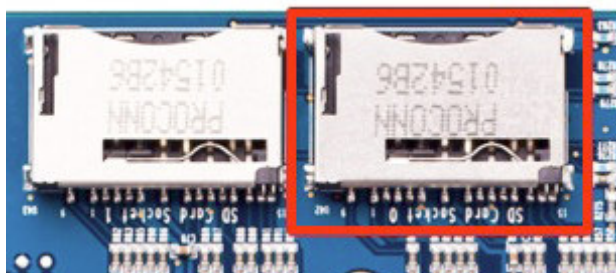
- Apply power to the i.MX28 development board.
- Press the power key.
- Press KEY1/2/3 to select an alternative boot cmdline, hold the key until the bootlets have run. To use the default boot command line, do not press any key.
- The bootlets and kernel will run.

5.2.3 Network Boot

Perform the following to boot from the network:

- Connect the target and host using an Ethernet 10 Base-T cable
- Ensure that the Trivial File Transfer Protocol (TFTP) server is running on the Linux host. (refer to [section 2.4](#))
- Copy the Linux kernel image to the /tftpboot directory:

```
cp ~/imx28/ltib/rootfs/boot/uImage /tftpboot
```
- Ensure that the rootfs file is available in the /etc/exports file in your host. (refer to [section 2.4](#))
- Insert SD card with i.MX28_uboot.sb burnt by cfimage.exe into SD slot0



- Power ON the i.MX28 development board
- Press the power key. Run Bootlets and uboot
- Press enter in the U-boot serial console (for example, using the minicom) to get the U-boot prompt
- Set the U-boot run-time variables :

```

MX28 U-Boot > setenv bootargs 'console=ttyAM0,115200n8'
MX28 U-Boot > setenv bootcmd 'run bootcmd_net'
MX28 U-Boot > setenv bootdelay 2
MX28 U-Boot > setenv baudrate 115200
MX28 U-Boot > setenv serverip [Host IP]
MX28 U-Boot > setenv ipaddr [IP Address]
MX28 U-Boot > setenv netmask [Netmask]
MX28 U-Boot > setenv bootfile uImage
MX28 U-Boot > setenv loadaddr 0x42000000
MX28 U-Boot > setenv nfsroot [Root nfs]
MX28 U-Boot > setenv bootargs_nfs 'setenv bootargs ${bootargs} root=/dev/nfs
ip=dhcp nfsroot=${serverip}:${nfsroot} fec_mac=[MAC address] gpmi'
MX28 U-Boot > setenv bootcmd_net 'run bootargs_nfs; dhcp; bootm'
MX28 U-Boot >setenv ethaddr [MAC address]
MX28 U-Boot >saveenv

```

Yellow marks are the one that you need to configure. Refer to [section 2.3.1](#) to see more details.

Following the section 2.3.1 suggested configuration

[Host IP] = 192.168.0.1

[IP Address] = 192.168.0.254

[NetMask] =255.255.255.0

[Root nfs] =/tools/rootfs if precompiled images are used. /home/youruser/imx28/ltib/rootfs if using self compiled images. (Note that you shouldn't use the ~ char which refer to home on linux environment. Use *youruser* with the corresponding user on your host machine.

NOTE [MAC address] = default with the board or configure your own.

NOTE

On the i.MX28 EVK, the MAC address is stored in OTP fuses that have been pre-programmed. If a differentthe MAC address is to be used, then the following option can be added:

xx:xx:xx:xx:xx:xx (Use hexadecimal values)

NOTE *dhcp* = If you have a dhcp server present on your network (maybe a router) leave as dhcp this configuration. If there is no dhcp modify the bootargs_nfs environment variable to

```
MX28 U-Boot > setenv bootargs_nfs 'setenv bootargs ${bootargs} root=/dev/nfs
ip=[Host IP] nfsroot=${serverip}:${nfsroot} fec_mac=[MAC address] gpmi'
```

Once all variables are set to the correct values, you can save the entire environment to the SD/MMC card:

```
MX28 U-Boot > saveenv
Saving Environment to MMC...
Writing to MMC(0)... done
```

If you have the suggested configuration your environment variables will be as follow. (printenv command lets you see the environment variables)

```
MX28 U-Boot > printenv
ipaddr=192.168.0.254
netmask=255.255.255.0
bootfile="uImage"
loadaddr=0x42000000
bootargs_mmc=setenv bootargs ${bootargs} root=/dev/mmcblk0p3 rw rootwait
ip=dhcp fec_mac=${ethaddr}
bootcmd_mmc=run bootargs_mmc; mmc read 0 ${loadaddr} 100 3000; bootm
ethact=FEC0
bootargs=console=ttyAM0,115200n8
bootcmd=run bootcmd_net
bootdelay=6
baudrate=115200
serverip=10.81.4.117
nfsroot=/tools/rootfs
bootargs_nfs=setenv bootargs ${bootargs} root=/dev/nfs ip=192.168.0.254
nfsroot=${serverip}:${nfsroot} gpmi
bootcmd_net=run bootargs_nfs; bootm
stdin=serial
stdout=serial
stderr=serial
ver=U-Boot 2009.08 (Aug 09 2010 - 18:49:32)
Environment size: 616/130044 bytes
```

To boot into Linux from the U-boot command line, simply run the boot command. Alternatively, you can reset the EVK and wait for it to automatically boot.

```
MX28 U-Boot > boot
```