i.MX28 Linux BSP

User's Guide

Document Number: 926-77208
Rev. 10.05.02
05/2010

Contents

About This Book

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

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

Audience

This document is intended for software, hardware, and system engineers who are planning to use the product and for anyone 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

1. *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 Boot Stream

When the iMX28 comes out of reset, it begins executing the ROM. There is no alternative - no other code is permitted to handle the reset exception. The ROM reads the boot mode pins to discover the boot source (USB, SD/MMC, NAND Flash, etc.) and negotiates with that source in a device-dependent way to retrieve a "boot stream." A boot stream is a stream of bytes in Safe Boot (SB) format.

This boot stream starts with a "Load" command that instructs the ROM to copy the executable into memory. The final "Jump" command instructs the ROM to transfer control to the executable that it just loaded.

Another very important command is "Call." This command tells the ROM to make a function call to a given address and then continue processing the boot stream when 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 to as a "bootlet".

Here's a schematic representation of a boot stream that contains two bootlets, followed by the main executable:

LOAD	Bootlet Executable #1	CALL	LOAD	Bootlet Executable #2	CALL	LOAD	Main Executable	JUMP

Figure 1-1 i.MX28 Boot stream outline

Each bootlet is an executable that has been 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 needs of the system. The i.MX28 Linux BSP boot streams contain the following bootlets:

power_prep — This bootlet configures up the power supply.

boot_prep — This bootlet configures up the clocks and sdram.

linux_prep — This bootlet prepares to boot Linux

Here's a schematic representation of a boot stream constructed with the i.MX28 Linux BSP:

LOAD	power_prep	CALL	LOAD	boot_prep	CALL	LOAD	linux_prep	CALL	LOAD	zImage	JUMP

Figure 1-2 An example of i.MX28 Boot stream loading Linux Kernel

Another example for U-Boot boot stream:

LOAD	power_prep	CALL	LOAD	boot_prep	CALL	LOAD	U-Boot	JUMP

Figure 1-3 An example of i.MX28 Boot stream loading U-Boot

For details about how to create boot stream image, please refer to “Creating Boot Stream Image” chapter.

1.2 Flash Boot Loader

The boot stream is an important concept for imx28, which can be regarded as a boot loader.

The Linux release provides two boot stream images:

- Linux kernel boot stream
- U-boot boot stream

Please see “Creating Boot Stream Image” chapter for detail.

There’s no RedBoot support for imx28.

1.3 Linux Kernel and Driver

The Freescale BSP contains the Freescale Linux 2.6.31 kernel, driver source code, and a pre-built kernel image. You can obtain the kernel image from the following location:

```
imx28/zImage
```

```
imx28/uImage
```

```
imx28/imx28_linux.sb (combine boot steam and kernel image)

imx28/imx28_uboot.sb (combine boot steam and uboot image)
```

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:

```
imx28/rootfs.jffs2

imx28/rootfs.ext2.gz
```

Chapter 2
Building the Linux Platform

This chapter explains how to set up the build environment, install and build LTIB, set rootfs for NFS, and set up the host environment.

2.1 Setting Up the Build Environment

Setting up the build environment includes installing Linux OS and LTIB.

2.1.1 Install Linux OS using Linux Builder

Install a Linux distribution such as Fedora 4/5, RedHat or ubuntu 8.0 on one computer.

2.2 Installing and Building LTIB

To install and build LTIB, use these steps.

NOTE

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

1. Install the LTIB package not as root:

```
tar zxf <ltib_release>.tar.gz
./<ltib_release>/install
```

   This command installs LTIB to your directory.

2. Build LTIB:

```
cd <your LTIB directory>
./ltib -m config
```

   Select platform to “Freescale iMX reference boards” and exit, saving the changes. At the next menu, select **mx28** as platform type and a package profile (min/test/fsl gnome profiles are tested). Exit saving changes.

   Then run the following command:

```
./ltib
```

   When complete, you can obtain:

The kernel image from `rootfs/boot/uImage` and `rootfs/boot/zImage`.

The SB file of bootlets and kernel image from `rootfs/boot/imx28_linux.sb`.

The SB file of bootlets and uboot image from `rootfs/boot/imx28_uboot.sb`.

The Jffs2 rootfs image from `rootfs.jffs2`

Change option under Target Image Generation menu from JFFS2 to EXT2 by "ltib –c", then you can get the EXT2 rootfs image from `rootfs.ext2.gz`

NOTE

You must set the network parameters in LTIB in order to boot via NFS:

```
./ltib -c
```

Set the network parameters in the following path:

Target System Configuration

Options--->
Network setup
IP address
netmask
broadcast address
gateway address
nameserver IP address

2.3 Setting rootfs for NFS

There are two ways to set the rootfs for NFS on this package.

- 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

Method 1: Using the rootfs package in the distribution

Use the following commands to set the `rootfs` directory for NFS (you must be the root user for this operation):

```
mkdir /mnt/rootfs
cp imx28/rootfs.ext2.gz  /tools
cd /tools
gunzip rootfs.ext2.gz
mount -o loop -t ext2 rootfs.ext2 /mnt/rootfs
cp -r /mnt/rootfs .
export ROOTFS_DIR=/tools/rootfs
```

Method 2: Using the rootfs created after the kernel build

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

```
%export ROOTFS_DIR=/<your LTIB directory>/rootfs
```

Other methods: For other ways to make a file system image file, see the *i.MX Family Linux Software Development Kit Reference Manual.*

2.4 Setting up the Linux Host

To set up the Linux host system, use these steps.

3. Turn off the firewall, to enable the `tftp` to work:

   ```
     iptables -F
   ```

   OR, at the command line, type:

   ```
   setup
   ```

4. Install the **tftp** server.

5. Install the **nfs** server.

6. Create the `tftboot` directory.

   The kernel images and anything that needs to be uploaded by `tftp` (such as the **zImage** kernel image) will be stored in this directory:

   ```
     mkdir /tftpboot
   ```

7. Edit `/etc/xinetd.d/tftp` to enable tftp as follows:

   ```
   {
   disable  =  no
   socket_type = dgram.
   protocol = udp.
   wait = yes
   user = root
   server = /usr/sbin/in.tftpd.
   server_args = /tftpboot
   }
   ```

8. Run the following command on your Linux host machine:

   ```
   vi /etc/exports
   ```

   add this line in the file: `/tools/rootfs *(rw,sync,no_root_squash)`

9. Restart the **nfs** and **tftp** servers on your host:

   ```
   /etc/init.d/xinetd restart
   /etc/init.d/nfs restart
   ```

10. Copy zImage and rootfs.jffs2 in the release package or LTIB to the tftp directory.

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

or

```
cp /<your LTIB directory>/rootfs/boot/zImage /tftpboot
cp /<your LTIB directory>/rootfs.jffs2 /tftpboot
```

NOTE

These instructions specify using an **nfs** server. Some Linux systems use **nfsserver**, rather than **nfs**. Use these instructions for either server type.

NOTE

A Windows tftp program "tftp.zip" is located under LTIB release package "Common/" folder. You can install it in Windows OS to setup Windows tftp server for downloading images.

2.5 Build Manufacturing Firmware (to be supported)

Please refer 2.2 Installing and Building LTIB to setup ltib environment.

Configure Firmware build profile

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

Choose correct item as below:

--- Choose the platform type

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

--- Choose the packages profile

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

After ltib complete build. Updater.sb will be created

Chapter 3 Configuring the Target Hardware

This chapter details all hardware specific configuration necessary to prepare the iMX28-EVK development board for use with Linux.

3.1 External Cabling

- Plug the Linux host straight serial console cable into the UART DSUB9 connector on the iMX28-EVK.
- Plug the Linux host USB A to mini-B USB cable into the mini-B USB connector on the iMX28-EVK.

3.2 Board Configuration

EVK board uses switch S2 to select boot mode, B0, B1, B2, B3 are labeled on the PIN location of switch.

Please refer to following table:

B3	**B2**	**B1**	**B0**	**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)

See hardware user manual doc for EVK board detail configuration.

NOTE

MX28 evk board need to hardware rework for booting from SD1. Please check detail from evk hardware user guide.

Chapter 4
Creating Boot Stream Image

The iMX28 SoC contains a built-in ROM firmware capable of loading and executing binary images in special format from different locations including 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 how the ROM firmware should handle these bootlets (e.g. load a bootlet to On-Chip RAM and run it from there).

For kernel configuration and building, please see the *i.MX Family Linux Software Development Kit Reference Manual.*

4.1 Setting the kernel command line

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

```
./ltib –p boot_stream.spec -f
```

The output boot stream images will be under rootfs/boot/ directory, named imx28_linux.sb and imx28_uboot.sb.

Chapter 5
Booting the Target Hardware

This chapter will assist the user in booting the IMX28 development board for the first time. There are a number of ways to boot Linux kernel on the IMX28:

- 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 IMX28 built-in ROM firmware. The ROM code reads the boot stream image containing Linux kernel from various 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 via USB or MMC card or NAND flash.

The Linux SDK provides two boot stream images:

- Linux kernel boot stream
- U-boot boot stream

Refer to “Creating Boot Stream Image” chapter to see how to generate a new boot stream image.

The following chapters describe how to prepare and boot the Linux kernel in each of the supported boot modes.

5.1 Target Preparation

5.1.1 Setting kernel command line

Kernel boot command line could be set in boot steam configuration ltib menu under package list → boot steam. These command line options include a default command line and three command lines selected by key presses during system start up. If the command line configuration file has less than four command lines then those entries are replaced by the following default command line string:

```
console=ttyAM0,115200
```

In order 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 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'          IP address of Ubuntu Linux host
' Target IP'        IP address assigned to the IMX28 development board
```

- ENET MAC address:

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

Where:
' xx:xx:xx:xx:xx:xx  '  MAC address of ENET of EVK board.
```

- gpmi or ssp1 selection for hw confliction

```
gpmi/ssp1

Where:
gpmi: initialize gpmi device
ssp1: initialize ssp1 (SD1) device
```

There are four preset command lines which allow booting the kernel with root file system located on MMC card, NAND flash and NFS boot:

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

```
noinitrd console=ttyAM0,115200 root=/dev/mmcblk0p3 rw rootwait
            fec_mac=00:08:02:6B:A3:1A ip=dhcp gpmi
```

- Alternative command line 1 for NFS (press KEY1 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 ip-dhcp rw gpmi
```

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

```
noinitrd console=ttyAM0,115200 root=/dev/mtdblock1  rootfstype=jffs2 rw rootwait
            fec_mac=00:08:02:6B:A3:1A gpmi
```

- Alternative command line 3 for ramdisk(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 default command lines are modified it is necessary to rebuild the release to get the Linux kernel boot stream image with updated command. Usually in LTIB, issue the command:

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

5.1.3 Writing the boot stream and rootfs to a boot media

MMC boot

- Default kernel command line use 3 primary partition.

1. FAT

2. Boot stream partition of type 0x53 (OnTrack DM6 Aux3)
3. linux rootfs such as ext2.

Create a third primary partition on MMC card in addition to the mandatory one containing the boot stream image. Below is an example output of fdisk -l showing a properly configured MMC/SD card (all steps are done in host PC):

```
fdisk -l /dev/sdb

Disk /dev/sdb: 1967 MB, 1967128576 bytes
61 heads, 62 sectors/track, 1015 cylinders
Units = cylinders of 3782 * 512 = 1936384 bytes
Disk identifier: 0x00000000

Device Boot      Start      End      Blocks   Id        System
/dev/sdb1   *        1       974    1840896    b  W95 FAT32
/dev/sdb2          974       977      5339+  53  OnTrack DM6 Aux3
/dev/sdb3          977      1016      74572  83  Linux
```

- Create a MMC partition image containing the boot stream in target by sdimage which is in uuc package in ltib.

```
sdimage imx28_linux.sb /dev/sdb
```

- Format the second partition:

```
mkfs.ext2 /dev/sdb2
```

- Mount the second MMC/SD partition:

```
mount /dev/sdb2 /mnt/mmc
```

- Copy the Linux release development root file system to the directory where the MMC partition is mounted. You have at least ways to get root file system, please refer to “Setting rootfs for NFS” Section
- Add manual updates to the root file system under /mnt/mmc if needed
- Unmount the MMC partition:

```
umount /mnt/mmc
```

There is a windows tools call “cfimager” can create partition, write boot stream and rootfs

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

NOTE

Default rootfs file system released for SD is EXT2 format. EXT2 is not a journal file system. Any behavior made file system with sync will cause file system error, like power lost, kernel panic etc. Follow normal power sequence or use EXT3 to avoid the file system sync issue.

NAND boot

A boot stream image can not be burned to NAND flash from the Linux host. It is necessary to load the kernel from MMC card or network boot. Once 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 root file system. For example, in case of NFS root use:

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

- Boot the target and log in to it:
- Burn the boot stream image to the flash:

```
#echo –n 1 > /sys/bus/platform/devices/gpmi.0/ignorebad
#flash_eraseall /dev/mtd0
#kobs-ng init imx28_linux.sb
```

- Copy jffs2 image to current root file system.

  For example, in case of NFS root run the following command on the Linux host.

  The image must match the flash device in use.

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

- Boot the target and log in to it
- Erase MTD partition:

```
flash_eraseall /dev/mtd1
```

- Burn the jffs2 image to the flash:

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

Network boot

Linux kernel network boot is implemented using the U-boot boot loader that is part of the Linux release for Freescale IMX28. The U-boot boot stream is loaded from SD or USB by sb_loader.exe tool. See detail from usb boot and network boot session in target boot chapter.

5.2 Host preparation

5.2.1 Root file system on NFS partition

Please refer to “Setting rootfs for NFS” Section.

5.3 Target Boot

5.3.1 USB boot

- Select a USB boot mode (0000). Refer to the “Configuring the Target Hardware” chapter for details.
- Power on the iMX28 development board
- Press power key.
- Press KEY1/2/3 to select alternative boot cmdline, hold on it till bootlets run pass. Press no key for default boot cmdline.
- After Windows recognize USB HID device, run following cmd in Windows concole:

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

5.3.2 MMC/NAND boot

- Select a MMC boot mode (SD:1001, NAND: 0100). Refer to the “Configuring the Target Hardware” chapter for details.
- For SD, Insert SD card with imx28_linux.sb and rootfs.ext2 burned by cfimage.exe into SD slot1. For NAND, boot steam and rootfs image should be already burn to flash.
- Power on the iMX28 development board
- Press power key.
- Press KEY1/2/3 to select alternative boot cmdline, hold on it till bootlets run pass. Press no key for default boot cmdline.
- Bootlet and kernel run.

5.3.3 Network boot

- Connect the target and host using a Ethernet 10 Base-T cable
- Make sure that TFTP server is running on the Linux host.
- Copy the Linux kernel image to /tftpboot directory:

```
cp rootfs/boot/uImage /tftpboot
```

- Make usre rootfs file exists in ROOTFS_DIR folder.
- Insert SD card with imx28_uboot.sb burned by cfimage.exe into SD slot0
- Power on the iMX28 development board
- Press power key. Bootlets and uboot run
- Press enter in the U-boot serial console (e.g., via 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 netmask 255.255.255.0
MX28 U-Boot > setenv bootfile uImage
MX28 U-Boot > setenv loadaddr 0x42000000
MX28 U-Boot > setenv nfsroot [ROOTFS_DIR]
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 >saveevn

Where:
'Host IP'        IP address of the Ubuntu Linux host
'ROOTFS_DIR'     NFS folder path
'MAC address'    MAC address of ethernet
```

- Load and start the Linux kernel:

```
boot
```