i.MX23 Linux BSP

User’s Guide

Document Number: 926-77208
Rev. 10.05.03
05/2010

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

Contents

About This Book

This document explains how to build and install the Freescale Linux BSP to the i.MX23-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*.

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 iMX23 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:

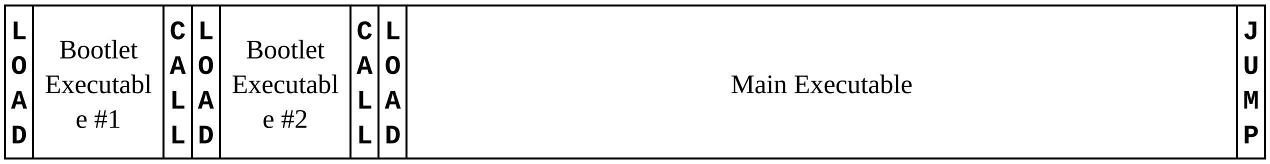

Figure 1-1 i.MX23 Boot stream outline

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.MX23 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.MX23 Linux BSP:

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

Figure 1-2 An example of i.MX23 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.MX23 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 imx233, which can be regarded as a boot loader.

The Linux SDK 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 imx233.

1.3 Linux Kernel and Driver

The Freescale BSP contains the Freescale Linux 2.6.28 kernel, driver source code, and a pre-built kernel image. You can obtain the kernel image from the following location:

```
L2.6.31_10.05.02_ER_images_MX233/zImage
```

```
L2.6.31_10.05.02_ER_images_MX233/imx23_linux.sb
```

```
L2.6.31_10.05.02_ER_images_MX233/imx23_uboot.sb
```

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:

```
imx233/rootfs.jffs2
```

```
imx233/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 platform type and package profile. Exit saving changes.

   Then run the following command:

   ```
   ./ltib
   ```

   When complete, you can obtain the kernel image from `rootfs/boot/zImage`.

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 imx233/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 imx233/zImage /tftpboot
    cp imx233/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

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 (**imx233/stmp3780**) --->

--- Choose the packages profile

Selection (**Updater profile for mx233**) --->

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 iMX233-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 iMX233-EVK.
- Plug the Linux host USB A to mini-B USB cable into the mini-B USB connector on the iMX233-EVK.

3.2 Board Configuration

iMX233-EVK SoC OTP bits (Default have burned)

There's a tool called BitBurner to burn the certain bits. Please refer to BitBurner\378x_mmc_boot_readme.txt file in the software package.

The following OTP bits should be set in order to enable MMC/SD boot:

- To enable SD/MMC MBR support in the iMX233 SoC ROM:

  - HW_OCOTP_ROM0[3]: 1

- To configure the IMX233 SoC ROM to use PWM3 pin for SD/MMC power gate control:

  - HW_OCOTP_ROM0[20]: 0

  - HW_COOTP_ROM0[21]: 1

iMX233-EVk Board

- Switches:

  - J3: Left, Choose fake battery

- S14: On
- S22: On
- S36:

USB	0000
NAND	0010
MMC	1001

LMS430 LCD card

- Jumper settings:
  - J8: 1-2
  - J6: 1-2
  - J5: 1-2
  - J4 1-2

NOTE

There is a conflict between any SSP device plugged into SSP1 connector and the MMC/SD card slot located on the CPU daughter card. If support for two or more conflicting devices is enabled in Linux kernel, the conflict will be detected and only one device will be enabled.

Chapter 4
Creating Boot Stream Image

The iMX23 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 imx23_linux.sb and imx23_uboot.sb.

Chapter 5
Booting the Target Hardware

This chapter will assist the user in booting the IMX23 development board for the first time. There are a number of ways to boot Linux kernel on the IMX23:

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

All boot modes except network boot are supported by the IMX23 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 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

Refer to the “Creating Boot Stream Image” chapter for details how to set up kernel command lines. 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 set the SSP port to a particular function, set the ssp1 and ssp2 command line variables:
  - SSP1 port in SPI mode:

    ```
    ssp1=spi1
    ```
  - SSP1 port in MMC/SD mode:

    ```
    ssp1=mmc
    ```

NOTE

The sspX option values are mutually exclusive. It is not possible to configure a SSP port for two different functions simultaneously.

- 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 (UBI volume):

```
ubi.mtd=1 root=ubi0:rootfs rootfstype=ubifs
```

  - Root file system on NFS over USB link: (You need built-in USB. The default image USB build as module)

```
ip=[Target IP]:[Host IP]::::usb0:off root=/dev/nfs
nfsroot=/tools/rootfs,rsize=1024,wsize=1024


Where:
’ Host IP’        IP address of Ubuntu Linux host
’ Target IP’      IP address assigned to the IMX233 development board
```

  - Root file system on NFS over Ethernet link:

```
ip=[Target IP]:[Host IP]::::eth0:off root=/dev/nfs
nfsroot=/tools/rootfs,rsize=1024,wsize=1024


Where:
’ Host IP’        IP address of Ubuntu Linux host
’ Target IP’      IP address assigned to the IMX233 development board
```

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

- Default command line:

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

- Alternative command line 1 (assigned to Key1 button):

```
noinitrd console=ttyAM0,115200 ssp1=spi1 ubi.mtd=1 root=ubi0:rootfs0 rootfstype=ubifs rw gpmi
```

- Alternative command line 2 (assigned to Key2 button): **User need modify it to set your rootfs path and nfs server address.**

```
noinitrd console=ttyAM0,115200 root=/dev/nfs
         nfsroot=10.193.100.213:/data/rootfs_home/rootfs_mx233 rw ssp1=spi1 ip=dhcp
         rootwait gpmi
```

- Alternative command line 3 (assigned to Key3 button):

```
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 SDK

If default command lines are modified it is necessary to rebuild the SDK 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 to a boot media

MMC boot

- The MMC card must contain a 16MB primary partition of type 0x53 (OnTrack DM6 Aux3). Below is an example output of fdisk -l showing a properly configured partition:

```
fdisk -l /dev/sdd

Disk /dev/sdd: 1014 MB, 1014497280 bytes
64 heads, 32 sectors/track, 967 cylinders
Units = cylinders of 2048 * 512 = 1048576 bytes


Device Boot     Start     End     Blocks   Id       System
/dev/sdd1       1         16      16368    53       OnTrack DM6 Aux3
```

- Create a MMC partition image containing the boot stream:

```
dd if=/dev/zero of=mmc_boot_partition.raw bs=512 count=4

dd if=<where the boostream lives>/iMX233_linux.sb
of=mmc_boot_partition.raw ibs=512 seek=4 conv=sync,notrunc
```

- Write the MMC partition image to the card (assuming the MMC boot partition is /dev/sdd1 as in the example above):

```
fdisk -l /dev/sdd

Disk /dev/sdd: 1014 MB, 1014497280 bytes
64 heads, 32 sectors/track, 967 cylinders
Units = cylinders of 2048 * 512 = 1048576 bytes


Device Boot     Start     End     Blocks   Id       System
/dev/sdd1       1         16      16368    53       OnTrack DM6 Aux3
```

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 STMP3780 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>/iMX23_linux.sb /tools/rootfs
```

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

```
kobs-ng -d iMX23_linux.sb
```

NOTE

If flash is dirty or has unexpected data, you use below command to erase all.

```
#echo –n 1 > /sys/bus/platform/devices/gpmi-nfc.0/ignorebad
#flash_eraseall /dev/mtd<N>
```

Network boot

Linux kernel network boot is implemented using the U-boot boot loader that is part of the Linux SDK for Freescale IMX23. The U-boot boot stream is loaded from any boot source supported by the IMX23 ROM. Refer to “MMC boot” and “NAND boot” sections for details on booting from those boot sources.

5.1.4 Preparing a root file system

Root file system on MMC card

- 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:

```
fdisk -l /dev/sdd

Disk /dev/sdd: 1014 MB, 1014497280 bytes
64 heads, 32 sectors/track, 967 cylinders
Units = cylinders of 2048 * 512 = 1048576 bytes


Device Boot     Start    End      Blocks   Id       System
/dev/sdd1       1        100                        FAT
/dev/sdd2       101      16       16368    53       OnTrack DM6 Aux3
/dev/sdd3       117      967      973824   83       Linux
```

- Format the second partition:

```
mkfs.ext2 /dev/sdd3
```

- Mount the second MMC/SD partition:

```
mount /dev/sdd2 /mnt/mmc
```

- Copy the Linux SDK 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 imx23_linux.sb –e rootfs.ext2 –d <mass storage disk, no “:”, such as
            H>
```

Root file system on NFS partition

Refer to “Setting rootfs for NFS” and “Setting up the Linux Host” sections.

NOTE

Note that the imx23 Ethernet conflicts with MMC/SD so MMC cards must be removed from MMC/SD slot on the iMX23 EVK board.

Root file system on NAND flash

A file system image may not be burned to the NAND Flash from the Linux host. The i.MX23 board can be booted using a root file system on MMC or NFS first and then the Flash file system image may be written to the NAND device with the nandwrite tool:

- Enter ltib/rootfs directory.
- tar -jcvf ../rootfs.tar.bz2 .
- Copy rootfs.tar.bz2 to mmc card or NFS root
- Boot board from mmc or NFS root
- Run the following commands

```
flash_eraseall /dev/mtd0
flash_eraseall /dev/mtd1
kobs-ng init  imx23_linux.sb
ubiattach /dev/ubi_ctrl -m 1 -d 0
ubimkvol /dev/ubi0 -n 0 -N rootfs0 -s 256MiB
ubimkvol /dev/ubi0 -n 1 -N rootfs1 -s 256MiB
ubimkvol /dev/ubi0 -n 2 -N data –m
mkdir -p /mnt/ubi0; mount -t ubifs ubi0_0 /mnt/ubi0
mkdir -p /mnt/ubi1; mount -t ubifs ubi0_1 /mnt/ubi1
tar -jxvf -C /mnt/ubi0 rootfs.tar.bz2
tar -jxvf –C /mnt/ubi1 rootfs.tar.bz2
umount /mnt/ubi0
umount /mnt/ubi1
```

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 MMC/NAND boot

- Select a MMC boot mode. Refer to the “Configuring the Target Hardware” chapter for details.
- Press and hold a SW37, SW26 button on iMX233 main board to select one of the alternative command lines instead of default one if needed
- Power on the iMX233 development board
- Press SW2 to turn on board.
- Wait until the kernel starts booting and release the button

5.3.2 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. Refer to the “Configuring the SDK Host Linux Environment” for details how to configure and start TFTP server on Linux host.
- Copy the Linux kernel image to /tftpboot directory:

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

- Boot the U-boot boot stream image from MMC/SD or NAND
- 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:

```
setenv ipaddr [Target IP]
setenv serverip [Host IP]
setenv netmask [Netmask]

Where:
'Host IP'          IP address of the Ubuntu Linux host
'Target IP'        IP address assigned to the IMX233 development board
'Netmask'          IP network mask associated with the Target IP
```

- In order to change the default kernel command line built into U-boot, set the bootargs runtime variable. The default command line is:

  ```
  console=ttyAM0,115200n8 lcd_panel=lms430 ssp1=spi1 ssp2=gpmi enc28j60=0@1,2:3
  ubi.mtd=1 root=ubi0:rootfs rootfstype=ubifs mem=128M
  ```

- Load and start the Linux kernel:

  ```
  boot
  ```