i.MX50 ARM2 Linux
User's Guide

Document Number: 09-7520-USRGD-ZCH70
Rev. 10.09.00
09/2010

freescale™
semiconductor

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.MX50 ARM2 board including board dip switch settings for image download and all kinds of boot mode, the steps to download image through sb_loader, u-Boot as well as the boot commands for each boot mode.

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.

References

1. Manufacturing Tool Quick Start Manual.doc
2. ltib_build_host_setup.pdf

Chapter 1
Introduction

The i.MX50 ARM2 Linux BSP is a collection of binary, source code, and support files that can be used to create a Linux kernel image and a root file system for i.MX50 ARM2 board. This document is for general Linux platform.

1.1 Boot Loader

The i.MX50 ARM2 Linux delivery package contains U-boot bootloader binaries.

The default `<Version>_images_MX5X/u-boot-mx50-mddr.bin` and `<Version>_images_MX5X/u-boot-mx50-lpddr2.bin` in the release package supports MMC/SD boot for MX50 ARM2 mDDR and LPDDR2 board separately.

1.2 Linux Kernel image

This Freescale i.MX BSP contains the Freescale Linux 2.6.31 ARM2 kernel, driver source code, and a pre-built kernel image. The i.MX50 ARM2 kernel image is found at the following location:

`uImage` - uImage is used together with U-Boot.

1.3 Root File System

The root file system package provides busybox, common libraries, and other fundamental elements. The i.MX50 ARM2 BSP package contains the following rootfs file system:

`<Version>_images_MX5X/rootfs.ext2.gz`

`rootfs.ext2.gz` file system includes Freescale specific libraries and gnome GUI. It can be mounted as NFS or the source of the storage of rootfs.

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 Linux Host

Setting up the Linux Host includes installing Linux OS and setting up TFTP/NFS server

2.1.1 Install Linux OS using Linux Builder

To build with LTIB or to program images to an MMC/SD card it is necessary to setup a Ubuntu 9.04 Linux host server as detailed in ltib_build_host_setup.pdf.

2.1.2 Linux Host Configuration

This section describes TFTP and NFS server setup. For the detailed setting requirements refer to the help menu in your host machine. The example below describes how to set up the servers on RedHat:

1. Turn off the firewall, to enable the `tftp` to work:
   ```
   iptables -F
   ```

or type at the command line:
   ```
   setup
   ```

2. Install the tftp server.
3. Install the nfs server.
4. Create the `tftboot` directory:
   ```
   mkdir /tftpboot
   ```

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

5. 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
}
```

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

```
vi /etc/exports
```

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

7. Restart the nfs and tftp servers on your host:

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

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.

2.2 Installing and Building LTIB

To install and build LTIB, follow the steps below:

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”.

To run LTIB, some host packages are needed. If any error related to a host package is raised, install the host package.

1. Remove all packages on `/opt/freescale/pkgs/` which are installed before.
2. Install the LTIB package not as root:

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

This command installs LTIB to your directory.

3. Build LTIB:

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

4. Select platform to **Freescale iMX reference boards** and exit, saving the changes. At the next menu, select platform type as **imx5x** and package profile. Exit and save changes. Please note that only **Min profile** and **FSL gnome release packages** are tested by default.

5. To build U-Boot for MX50 ARM2 board, Select “Choose your board” as “MX50_ARM2”. Please note this option is only for U-Boot. For kernel image, current default kernel configuration can build the same images for all i.MX5 parts boards.

```
--- Choose your board

        board (mx50_arm2)  --->
```

6. Run the following command:

```
./ltib
```

   When this procedure is completed, the kernel image is located at rootfs/boot/uImage.

7. Input the following command to get LTIB command help:

```
./ltib -help
/* Get the source code of one package */
./ltib -m prep -p <package name>
/* Build one package */
./ltib -m scbuild -p <package name>
/* Install one package to rootfs */
./ltib -m scdeploy -p <package name>
```

2.3 Setting rootfs for NFS

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

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

Use the following commands to set the `rootfs` directory for NFS using the `rootfs.ext2.gz` package already included in the distribution (you must be the root user for this operation):

```
mkdir /mnt/rootfs
cp imx5x/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
```

NOTE

In some Linux distributions (such as Fedora) the user needs to make sure that the contents inside `/tools/rootfs` has the proper permission for user access. Since the mount command is made as root, the content shows as restricted access after the command `cp -r /mnt/rootfs`, this may cause the NFS not been able to get mounted.

To use the root file system created in the LTIB directory after the kernel build, use the command:

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

2.4 Copying images to TFTP server

To use tftp server to download the image, copy the kernel image in the release package or LTIB to the tftp directory. For example:

```
cp imx5x/uImage /tftpboot
```

Or:

```
cp /<LTIB directory>/rootfs/boot/uImage /tftpboot
```

2.5 How to generate no-padding U-Boot

To generate no-padding U-Boot, run:

```
sudo dd if=u-boot-MX50.bin of=u-boot-MX50-no-padding.bin bs=512 skip=2
```

2.6 How to generate uImage

In kernel source code, change build target from “zImage” to “uImage”.

If you want to generate uImage from zImage you built, you can generate a “uImage" based on the above zImage as below:

- Build u-boot package to get “mkimage” tool under rpm/BUILD/u-boot-<version>/ tools/mkimage.
- Copy mkimage to /usr/bin/
- Run the below command:

  mkimage -A arm -O linux -T kernel -C none -a 0x70008000 -e 0x70008000 -n "Linux-<kernel_version>" -d zImage uImage

  Note: Replace kernel version for your image. For example, 2.6.31-151-xxxx.

2.7 Build Manufacturing Firmware

Please refer section 2-2 to setup LTIB environment.

Configure Firmware build profile

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

Choose correct item as below:

```
--- Choose the platform type

      Selection (imx5x)  --->

--- Choose the packages profile

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

In “Freescale iMX5x Based Boards” section, choose the board information as the following:

```
--- Choose your board

       board (mx50_arm2)  --->
```

After ltib complete build, **initramfs.cpio.gz.uboot** is generated under ltib root folder. **u-boot.bin** and **uImage** for MFG tool are generated under `rootfs/boot/`.

Chapter 3 Boot configuration switch

The boot modes of the i.MX50 ARM2 board are controlled by the boot configuration DIP switch on the main board. The following sections just lists basic boot setup configurations. For more different combinations for the boot modes, see i.MX50 HW guide.

Note: eSDHC Slot 1 is SD only socket at CPU board. eSDHC Slot 2 is MS/SD combo socket at CPU board.

3.1 Configuration setup for USB HID downloading

Table 3-1 shows the boot switch settings to allow programming via MfgTool or sb_loader to download image by USB HID.

Table 3.1 Main board switch setup for USB HID downloading

Switch	D1	D2	D3	D4	D5	D6	D7	D8
SW2	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
SW3	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
SW4	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF

.

3.2 Configuration switch setup for SD Slot1 boot

Table 3-2 shows the boot switch settings for SD Slot 1 boot

Table 3.2 Main board switch setup for SD Slot1 boot

Switch	D1	D2	D3	D4	D5	D6	D7	D8
SW2	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
SW3	OFF	OFF	ON	OFF	OFF	OFF	OFF	OFF
SW4	OFF	ON	OFF	OFF	ON	OFF	OFF	OFF

Chapter 4 Flash memory map

This chapter describes the software layout in MMC/SD cards. It can be useful to understand the later sections about image download.

4.1 MMC/SD flash memory map

MMC/SD flash scheme is different from NAND and NOR flash which are deployed in the BSP software. MMC/SD flash must keep the first sector (512 bytes) as MBR (Master Boot Record) in order to use MMC/SD as the rootfs.

At boot up, MBR is executed to look up the partition table to determine which partition to use for booting. Bootloader should be at the end of MBR. Kernel Image and rootfs can be put any address after bootloader.

MBR can be generated through `fdisk` command when creating partitions in MMC/SD cards in the Linux Host server.

Chapter 5 Downloading images using Mfg Tools

Currently Manufacturing Tools only supports downloading images onto SD card via Slot1.

Before downloading, the board needs to be set as USB recovery mode, refer to 3.1

Refer to Manufacturing Tools release package for detailed information.

Chapter 6 Download images by bootloader or NFS

6.1 Setup Terminal

The i.MX50 ARM2 board can communicate with a host server (Windows or Linux) via the serial cable. Common serial communication programs such as HyperTerminal, Tera Term or PuTTY can be used. The example below describes the serial terminal setup using HyperTerminal in a Windows host:

1. Connect the target and the Windows PC via a serial cable.
2. Open HyperTerminal on the Windows PC, and select the settings shown in Figure 6-1.

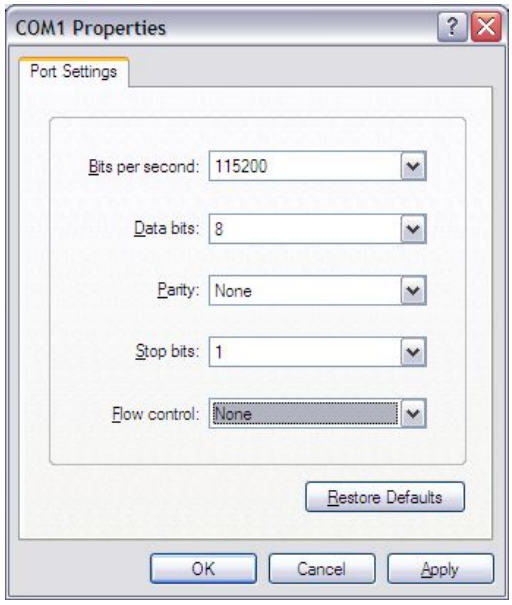

Figure 6 -1 HyperTerminal Settings for terminal setup

3. Set the boot configuration options as described in section 3.2 and power up the target. Status messages and the bootloader prompt are displayed on the terminal screen.

6.2 Download by U-Boot

6.2.1 MMC/SD

- Power on the board from MMC/SD by following the dip setting in Section Dip switch setup for boot modes. Then set environment variables. For example,

```
ARM2 U-Boot > setenv serverip 10.192.225.216
ARM2 U-Boot > setenv fec_addr 00:04:9f:00:ea:d3
ARM2 U-Boot > setenv ethaddr 00:04:9f:00:ea:d3
ARM2 U-Boot > setenv bootfile uImage
ARM2 U-Boot > saveenv
```

- Copy uImage to tftp server. Then download it to RAM:

```
ARM2 U-Boot > dhcp
```

- Query the information about MMC/SD card. Note current version doesn't support 64bit. So the capability information (>=4G) is shown as negative value.

```
ARM2 U-Boot > mmcinfo
Device: FSL_ESDHC
Manufacturer ID: 3
OEM: 5344
Name: SD04G
Tran Speed: 25000000
Rd Block Len: 512
SD version 2.0
High Capacity: Yes
Capacity: 2002647686
Bus Width: 4-bit
```

- Check the usage of "mmc" command. "blk#" is equal to "<the offset of read/write>/<block length of the card>". "cnt" is equal to "<the size of read/write>/<block length of the card>".

```
BBG U-Boot > help mmc
mmc - MMC sub system

Usage:
mmc read <device num> addr blk# cnt
mmc write <device num> addr blk# cnt
mmc rescan <device num>
mmc list - lists available devices
```

- Program the kernel uImage into MMC/SD. For example, the below command writes the image with the size 0x300000 from ${loadaddr} to the offset 0x100000 of the MMC/SD card. Here 0x800 =0x100000/512, 0x1800=0x300000/512. The block size of this card is 512. This example assumes the kernel image is less than 0x300000 bytes. If the kernel image exceeds 0x300000, please enlarge the image length.

```
ARM2 U-Boot >  mmc write 0 ${loadaddr} 0x800 0x1800
```

- Boot up the system through RFS in MMC/SD card:

```
ARM2 U-Boot > setenv bootargs_mmc 'setenv bootargs ${bootargs} console=tty0
                root=/dev/mmcblk0p2 rootwait rw'
ARM2 U-Boot > setenv bootcmd_mmc 'run bootargs_base bootargs_mmc;mmc read 0
                ${loadaddr} 0x800 0x1800;bootm'
ARM2 U-Boot > setenv bootcmd 'run bootcmd_mmc'
ARM2 U-Boot > saveenv
```

- To program the rootfs to MMC/SD, See section "Use i.MX50 as Host server to create rootfs" or section "Using a Linux Host to set up an SD/MMC card".

6.2.2 U-Boot configurations

U-Boot "print" command can be used to check environment variable values. "setenv" command is to set environment variable value. See U-Boot User Guide for the details.

6.3 Use i.MX50 as Host server to create rootfs

Linux provides multiple methods to program images to the storage device. This section describes how to use the i.MX50 ARM2 as Linux Host server to to create the rootfs on MMC/SD card.

1. Boot from NFS or other storage. To create a partition in MMC/SD Slot 0.use the `fdisk` command in the Linux console:

```
root@freescale ~$ fdisk /dev/mmcblk0
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel. Changes will remain in memory only,
until you decide to write them. After that the previous content
won't be recoverable.

The number of cylinders for this disk is set to 124368.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): p

Disk /dev/mmcblk0: 4075 MB, 4075290624 bytes
4 heads, 16 sectors/track, 124368 cylinders
Units = cylinders of 64 * 512 = 32768 bytes

        Device Boot      Start         End      Blocks  Id System
```

2. As described in Chapter 4, the rootfs partition should be located after kernel image; the first 0x800000 bytes can be reserved for MBR, bootloader and kernel sections.

   From the above log, the `Units` of current MMC/SD card is 32768 bytes. The begin cylinder of the first partition can be set as "0x300000/32768 = 96". The last cylinder can be set according to the rootfs size. Create a new partition by typing `n`:

```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-124368, default 1): 96
Last cylinder or +size or +sizeM or +sizeK (96-124368, default 124368):
Using default value 124368

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-r mmcblk0:ead partition table
 p1
```

3. Format the MMC/SD partitions as types ext3 or ext2 type. For example, to use ext3:

```
root@freescale ~$ mkfs.ext3 /dev/mmcblk0p1
mke2fs 1.41.4 (27-Jan-2009)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
248992 inodes, 994184 blocks
49709 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=1019215872
31 block groups
32768 blocks per group, 32768 fragments per group
8032 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736

Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 20 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

4. Copy the rootfs contents into the MMC/SD card (copy the rootfs.ext2 to NFS rootfs)

```
mount -t ext2 -o loop /rootfs.ext2 /mnt/cdrom
cd /mnt
mkdir mmcblk0p1
mount -t ext3 /dev/mmcblk0p1 /mnt/mmcblk0p1/
cp -rf /mnt/cdrom/* /mnt/mmcblk0p1/
umount /mnt/mmcblk0p1
umount /mnt/cdrom
```

5. Type `sync` to write the contents to MMC/SD.
6. Type `poweroff` to power down the system. Follow the instructions in Chapter 7 to boot the image from MMC/SD card.

Chapter 7
Running the Image on the Target

This chapter explains how to run an image on the target from downloaded flash and NFS. These instructions assume that you have downloaded the kernel image using the instructions in Chapter 5 or Chapter 6.

7.1 Run the image from NFS

To boot from NFS, do as the follows:

```
ARM2 U-Boot > setenv nfsroot /data/rootfs_home/rootfs_MX50
ARM2 U-Boot > run bootcmd_net
```

7.2 Run the image from MMC/SD flash

To boot the system from MMC/SD flash follow the steps bellow:

1. Set boot configuration switch as indicated in section 3.2 Configuration switch setup for boot modes. Power on the board.

2. Enter the following commands in the U-Boot prompt:

```
ARM2 U-Boot > setenv bootargs_mmc 'setenv bootargs ${bootargs} console=tty0
                root=/dev/mmcblk0p2 rootwait rw'
ARM2 U-Boot > setenv bootcmd_mmc 'run bootargs_base bootargs_mmc;mmc read 0
                ${loadaddr} 0x800 0x1800;bootm'
ARM2 U-Boot > setenv bootcmd 'run bootcmd_mmc'
```

Chapter 8
Using a Linux Host to set up an SD/MMC card

This chapter describes the steps to prepare an SD/MMC card to boot off an i.MX50 ARM2.

8.1.1 Requirements

An SD/MMC card reader, like a USB card reader, is required. It will be used to transfer the boot loader and kernel (`zImage`) images, to initialize the partition table and copy the root file system. To simplify the instructions, it is assumed that a 4GB SD/MMC card is used.

Any Linux distribution can be used for the following procedure. It is recommended to use a Linux distribution that LTIB has been tested against (like Fedora, Ubuntu, etc).

The Linux kernel running on the Linux host will assign a device node to the SD/MMC card reader. The kernel might decide the device node name or udev rules might be used. In the following instructions, it is assumed that udev is not used.

To identify the device node assigned to the SD/MMC card, enter the command:

```
$ cat /proc/partitions
major minor  #blocks  name
   8     0   78125000 sda
   8     1   75095811 sda1
   8     2          1 sda2
   8     5    3028221 sda5
   8    32  488386584 sdc
   8    33  488386552 sdc1
   8    16    3921920 sdb
   8    18    3905535 sdb1
```

In this example, the device node assigned is `/dev/sdX` (a block is 1kB large)

Please make sure you're using the right device node for MMC/SD, because writing image to wrong device node (for example, which stores system data) could probably crash the operating system.

8.1.2 Copying the boot loader image

Enter the following command to copy the U-Boot image to the SD/MMC card (please note that this operation will delete the partition table present on the media):

```
$ sudo dd if=u-boot-MX50.bin of=/dev/sdX bs=512 && sync && sync
```

To update redboot to another version, please run the following command instead:

```
$ sudo dd if=u-boot-MX50-no-padding.bin of=/dev/sdX bs=512 seek=2 && sync && sync
```

The first 1kB, that includes the partition table, will be preserved.

8.1.3 Copying the kernel image (zImage)

The following command will copy the kernel image to the SD/MMC card

```
$ sudo dd if= uImage of=/dev/sdX bs=512 seek=2048 && sync && sync
```

This will copy the `uImage` to the media at offset 1MB.

8.1.4 Copying the file system (rootfs)

A partition table must be first created. If a partition already exists and it is big enough for the file system you want to deploy, then you can skip this step.

To create a partition, at offset 8192 (in sectors of 512 bytes) enter the following command:

```
$ sudo fdisk /dev/sdX
```

Type the following parameters (each followed by <**ENTER**>):

```
u         [switch the unit to sectors instead of cylinders]
d         [repeat this until no partition is reported by the  'p'  command ]
n         [create a new partition]
p         [create a primary partition]
1         [the first partition]
8192      [starting at offset sector #8192, i.e. 4MB, which leaves enough space for the
kernel, the boot loader and its configuration data]
<enter>   [using the default value will create a partition that spans to the last sector of
the medium]
w         [ this writes the partition table to the medium and fdisk exits]
```

The file system format ext3 or ext4 is a good option for removable media due to the built-in journaling. Run the following command to format the partition:

```
$ sudo mkfs.ext3 /dev/sdX1
Or
$ sudo mkfs.ext4 /dev/sdX1
```

Copy the target file system to the partition:

```
$ mkdir /home/user/mountpoint
$ sudo mount /dev/sdX1 /home/user/mountpoint
```

Let's assume the root file system files are located in `/home/user/rootfs`:

```
$ cd /home/user/rootfs
$ sudo cp - rpa [A-z]* /home/user/mountpoint
$ sudo umount /home/user/mountpoint
```

The file system content is now on the media.