i.MX 6Dual/6Quad SABRE-SD Linux User's Guide

Document Number: IMX6DQLXUGSSD
Rev L3.0.35_4.1.0, 09/2013

freescale™

Contents

Section number **Title** **Page**

Chapter 1
About This Book

Chapter 2
Introduction

Chapter 3
Building the Linux Platform

Chapter 4
How to Boot the i.MX 6Dual/6Quad SABRE-SD Board

Chapter 1
About This Book

This document explains how to build and install the Freescale Linux BSP, where BSP stands for Board Support Package, on the i.MX 6Dual/6Quad SABRE-SD board and platform. All steps needed to get the i.MX 6Dual/6Quad SABRE-SD board and platform running are detailed, including board dip switch settings, steps to download an OS image through the manufacturing (MFG) tool, and instructions on configuring and using the U-Boot bootloader.

1.1 Audience

This information 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.

Chapter 2
Introduction

The i.MX 6Dual/6Quad SABRE-SD Linux BSP is a collection of binary, source code, and support files that can be used to create U-Boot boot loader, Linux kernel image, and a root file system for i.MX 6Dual/6Quad SABRE-SD development systems (MCIMX6Q-SDB and MCIMX6Q-SDP). For the steps on how to run and configure new Ubuntu rootfs, see How to Set Up Ubuntu Rootfs.

2.1 Boot Loader

The i.MX 6Dual/6Quad SABRE-SD Linux delivery package contains the following U-Boot bootloader binary:

L3.0.35_4.1.0_130816_images_MX6/u-boot-mx6q-sabresd.bin

This bootloader supports SD/MMC, eMMC4.3, eMMC4.4, eMMC4.41, SATA

2.2 Linux Kernel Image

This Freescale i.MX BSP contains a pre-built kernel image based on the 3.0.35 version of the Linux kernel. The i.MX 6Dual/6Quad SABRE-SD kernel image is located at the following path:

L3.0.35_4.1.0_130816_images_MX6/uImage

2.3 Gnome Mobile Root File System

The root file system package provides busybox, common libraries, and other fundamental elements.

The i.MX 6Dual/6Quad SABRE-SD BSP package contains the following rootfs file system:

L3.0.35_4.1.0_130816_images_MX6/rootfs.ext2.gz

The rootfs.ext2.gz file system includes Freescale specific libraries and gnome GUI. It can be mounted as NFS (see Setting rootfs for NFS), or its contents can be stored on a boot media such as Secure Digital (SD) card.

2.4 Ubuntu Demo Rootfs

An Ubuntu demo rootfs (11.10 Oneiric) with demo applications is provided for demo purpose.

The login credentials are User: linaro and Password: linaro.

Chapter 3
Building the Linux Platform

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

Note that not all of the steps are required for every boot mode. The only required steps are in Setting Up the Linux Host and Installing and Building LTIB.

3.1 Setting Up the Linux Host

See the *Setting Up a Linux Host for LTIB Builds on Ubuntu 9.04* document included in the release package to set up the Linux host server.

3.2 Installing and Building LTIB

To install and build LTIB, follow the steps below:

NOTE

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 previously-installed packages from /opt/freescale/pkgs/.
2. Install the LTIB package, not as root, in a location such as /home/user/:

```
tar zxvf L3.0.35_4.1.0_130816_source.tar.gz

./L3.0.35_4.1.0_130816_source/install
```

   This command installs LTIB to your directory.

3. Build LTIB:

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

4. Select platform to **Freescale i.MX reference boards** and exit, saving the changes. At the next menu, select platform type as **imx6q** and package profile. Exit and save the new configuration. Only the profiles of **Min profile, FSL gnome release packages**, and **mfg firmware profile** pass build tests.

NOTE

You can use the ./ltib -m selectype command to change the profile after the first selection.

5. To build U-Boot for i.MX 6Dual/6Quad SABRE-SD board, select "Choose your board for U-Boot" as "mx6q_sabresd". Please note this option is only for U-Boot. For the kernel image, the current default kernel configuration builds a single image that works for all i.MX 6 boards except i.MX 6SoloLite boards.

```
--- Choose your board
       board (mx6q_sabresd)  --->
```

7. Close the configuration screen to save the changes.
8. Run the following command:

```
./ltib
```

When this procedure is completed, the kernel image and the U-Boot images are located at: rootfs/boot/

9. Some other useful LTIB commands are:

```
./ltib -help

/* List the packages in LTIB */
./ltib –m listpkgs

/* Get the source code of one package. The source code will be extracted to <ltib folder>/
rpm/BUILD/  */
./ltib -m prep -p <package name>

/* This command is used to build the source code of <package name>. If you modify the source
code, you can rebuild the source code by this command  */
./ltib -m scbuild -p <package name>

/* Install one package to rootfs */
./ltib -m scdeploy -p <package name>
```

3.3 Setting rootfs for NFS

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

- Using the ext2 format rootfs package 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
mkdir /tools
cp imx6s/rootfs.ext2.gz  /tools
cd /tools
gunzip rootfs.ext2.gz
mount -o loop -t ext2 rootfs.ext2 /mnt/rootfs
cp -a /mnt/rootfs .
```

NOTE

In some Linux distributions (such as Fedora), the user needs to make sure that the contents inside /tools/rootfs have the proper permission for user access. Since the mount command is made as root, the content shows as restricted access after the command cp -a /mnt/rootfs, which may prevent the NFS mount from working correctly.

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

```
%vi /etc/exports
       edit this file by adding the export directory, for example:
       /tools/rootfs *(rw,no_root_squash)
       save and exit
%exportfs -a
```

3.4 Copying Images to TFTP Server

To use the TFTP server to download the image, copy the kernel image in the release package or LTIB to the TFTP directory.

For example:

```
cp imx6/uImage <tftp folder>
```

or

```
cp /<LTIB directory>/rootfs/boot/uImage <tftp folder>
```

3.5 How to Generate No-Padding U-Boot

To generate no-padding U-Boot, run the following command:

```
sudo dd if=u-Boot-mx6q-sabresd.bin of=u-Boot-mx6q-sabresd-no-padding.bin bs=512 skip=2
```

3.6 How to Generate uImage from a zImage

To generate a uImage with LTIB, in the kernel source code, change the build target from "zImage" to "uImage".

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

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

```
mkimage -A arm -O linux -T kernel -C none -a 0x10800000 -e 0x10800000 -n
"Linux-$(KERNELRELEASE)" -d zImage uImage
```

Note: Replace KERNELRELEASE with the appropriate kernel version for your image. For example, 3.0.35-151-xxxx.

3.7 How to Build U-Boot and Kernel in Standalone Environment

To build U-Boot in a standalone environment, perform the following actions in the root folder of U-Boot sources:

```
./ltib -m prep -p u-boot

cd rpm/BUILD/u-boot-2009.08

make ARCH=arm
CROSS_COMPILE=/opt/freescale/usr/local/gcc-4.6.2-glibc-2.13-linaro-multilib-2011.12/fsl-
linaro-toolchain/bin/arm-none-linux-gnueabi- distclean

make ARCH=arm
CROSS_COMPILE=/opt/freescale/usr/local/gcc-4.6.2-glibc-2.13-linaro-multilib-2011.12/fsl-
linaro-toolchain/bin/arm-none-linux-gnueabi- mx6q_sabresd_config

make ARCH=arm
CROSS_COMPILE=/opt/freescale/usr/local/gcc-4.6.2-glibc-2.13-linaro-multilib-2011.12/fsl-
linaro-toolchain/bin/arm-none-linux-gnueabi-
```

To build the kernel in the standalone environment, do the following:

```
cd <ltib folder>/rpm/BUILD/linux

make ARCH=arm
CROSS_COMPILE=/opt/freescale/usr/local/gcc-4.6.2-glibc-2.13-linaro-multilib-2011.12/fsl-
linaro-toolchain/bin/arm-none-linux-gnueabi- imx6_defconfig
```

```
make ARCH=arm
CROSS_COMPILE=/opt/freescale/usr/local/gcc-4.6.2-glibc-2.13-linaro-multilib-2011.12/fsl-
linaro-toolchain/bin/arm-none-linux-gnueabi- uImage
```

3.8 How to Set Up Ubuntu Rootfs

To set up ubuntu rootfs, perform the following operations:

1. Follow the instructions of Using a Linux Host to Set Up an SD/MMC Card to create an SD card with a valid U-Boot, kernel, and root file system partition without any content and with format only.
2. Mount the SD card and decompress the root file system files. The SD device name in your Linux OS may be different. Use the dmesg command to obtain it. In this example, it is /dev/sdb1:

   ```
   sudo mount /dev/sdb1 /mnt/hd
   cd /mnt/hd
   sudo tar --numeric-owner -xzvf /<path>/oneiric.tgz
   ```

   NOTE

   The option "--numeric-owner" may be unavailable if you use busybox. Because this option is mandatory, make sure that you use the full blown version of tar.

3. Boot with oneiric rootfs. The password is **linaro** in case you cannot log in with the **linaro** user name and without password.
4. Install the FSL packages on the target. You should already have copied all the *.deb files that come along with the released BSP to the target board. The deb files of each release can be located in the demo image package:

   ```
   sudo dpkg \--force-architecture \-i *.deb
   sudo depmod
   ```

5. Flush data to the SD card and reboot:

   ```
   sync
   sudo halt
   ```

6. Oneiric demo rootfs comes from the Linaro release. It can be downloaded from https://wiki.linaro.org/Boards/MX6QSabreLite. Then apply the following changes:

   To reserve the DMA buffer for video playback, ensure that the /proc/sys/vm/lowmem_reserve_ratio value is 1. This setting can be added into /etc/rc.local:

   ```
   echo 1 1 > /proc/sys/vm/lowmem_reserve_ratio
   ```

3.9 Building Manufacturing Firmware

Keep the LTIB environment clean before you begin to configure and build the firmware. Then, set up the LTIB environment and configure the file ./config/platform/imx/imx6q_updater.cf to specify the target firmware.

The following table provides the detailed information.

Table 3-1. Configuration for specific target

Target firmware	Configuration needed
i.MX 6Quad ARM2 board SD/eMMC	CONFIG_PKG_U_BOOT_CONFIG_TYPE="mx6q_arm2_mfg_config" CONFIG_BOARD_MX6Q_ARM2=y
i.MX 6Quad Sabre Lite board SD/ eMMC	CONFIG_PKG_U_BOOT_CONFIG_TYPE="mx6q_sabrelite_mfg_config" CONFIG_BOARD_MX6Q_SABRELITE=y
i.MX 6Quad Sabre-AI board SD/ eMMC	CONFIG_PKG_U_BOOT_CONFIG_TYPE="mx6q_sabreauto_mfg_config" CONFIG_BOARD_MX6Q_SABREAUTO=y
i.MX 6Quad Sabre-AI board SPI-NOR	CONFIG_PKG_U_BOOT_CONFIG_TYPE="mx6q_sabreauto_spi-nor_mfg_config" CONFIG_BOARD_MX6Q_SABREAUTO_SPI-NOR=y
i.MX 6Quad Sabre-AI board WEIM NOR	CONFIG_PKG_U_BOOT_CONFIG_TYPE="mx6q_sabreauto_weimnor_mfg_config" CONFIG_BOARD_MX6Q_SABREAUTO_WEIMNOR=y
i.MX 6Quad Sabre-SD board SD/ eMMC	CONFIG_PKG_U_BOOT_CONFIG_TYPE="mx6q_sabresd_mfg_config" CONFIG_BOARD_MX6Q_SABRESD=y

Run the following command line to generate the manufacturing firmware.

```
./ltib --profile config/platform/imx/updater.profile --preconfig config/platform/imx/
imx6q_updater.cf --continue --batch
```

After LTIB has completed the build, **initramfs.cpio.gz.uboot** is generated under the ltib root folder. The **u-boot.bin** and **uImage** for MFG tool are generated under rootfs/boot/.

Chapter 4
How to Boot the i.MX 6Dual/6Quad SABRE-SD Board

The boot modes of the i.MX 6Dual/6Quad SABRE-SD board are controlled by the boot configuration DIP switches on the board. To locate the boot configuration switches, see SABRE-SDP Quick Start Guide (SABRESDP_IMX6_QSG), which is available at http://www.freescale.com/files/32bit/doc/quick_start_guide/SABRESDP_IMX6_QSG.pdf. The following sections list basic boot setup configurations only.

4.1 How to Enter Serial Download Mode for MFGTool

No dedicated boot dips are reserved for serial download mode on i.MX 6 SABRE-SD boards. Therefore, a tricky method is used to enter serial download mode. For example, set boot mode as SD3 boot (SW6 dip 2, 7 **on** others are **off**). Do not insert SD card into SD3 slot, and power on the board. After "HID- Compliant device" is detected, it means that the board has entered serial download mode. Insert the SD card into SD3 slot. Another way to do this is to configure an invalid boot switch setting, for example, set all the dips of SW6 to off.

4.2 How to Boot from SD Card from Slot2

The following table shows the dip settings for SD2 boot (J500). The SD2 slot (J500) sits besides the LVDS1 connection on the back of the board.

Table 4-1. Boot switch setup for SD2 boot (J500)

Switch	D1	D2	D3	D4	D5	D6	D7	D8
SW6	ON	OFF	OFF	OFF	OFF	OFF	ON	OFF

4.3 How to Boot From SD Card from Slot3

The following table shows the dip settings for SD3 boot (J507). The SD3 slot (J507) is located between the HDMI and UART ports.

Table 4-2. Boot switch setup for SD3 boot (J507)

Switch	D1	D2	D3	D4	D5	D6	D7	D8
SW6	OFF	ON	OFF	OFF	OFF	OFF	ON	OFF

4.4 How to Boot From eMMC4.4

Table below shows the boot switch settings to boot from eMMC4.4 (SDIN5C2-8G) through the SD4 slot. The Sandisk SDIN5C2-8G (U512) is located besides the LVDS1 connection on the back of the board.

Table 4-3. Boot switch setup for SD4 boot

Switch	D1	D2	D3	D4	D5	D6	D7	D8
SW6	ON	ON	OFF	ON	OFF	ON	ON	OFF

4.5 How to Boot From SATA

Table below shows the boot switch settings to boot from SATA.

Table 4-4. Boot switch setup for SATA boot

Switch	D1	D2	D3	D4	D5	D6	D7	D8
SW6	OFF	OFF	OFF	OFF	OFF	ON	OFF	OFF

Chapter 5
Flash Memory Map

This chapter describes the software layout in MMC/SD cards.

This information may be useful for understanding subsequent sections about image download.

5.1 MMC/SD/SATA Memory Map

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

Upon boot up, the MBR is executed to look up the partition table to determine which partition to use for booting. The bootloader should be after the MBR. The kernel image and rootfs may be stored at any address after bootloader.

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

Chapter 6
Downloading Images by Using MFGTool

This chapter describes the procedure for using the MFGTool to download images to the different devices.

6.1 Installing the MFGTool

Unzip Mfgtools-Rel-4.1.0_130816_MX6Q_UPDATER.tar.gz

6.2 Usage

The MFGTool V2 follows the instructions in "Profiles\MX6Q Linux Update\OS Firmware\ucl2.xml" to execute program operations. The user must read and update ucl2.xml to understand the operations before using the MFGTool V2.

Additionally, the user must correctly configure the cfg.ini and UICfg.ini. For example, if only a board would be supported, "PortMgrDlg=1" should be set in UICfg.ini; if four boards would be supported, "PortMgrDlg=4" should be set. Incorrect configuration will cause MFGTool V2 to malfunction.

Follow these instructions to use the i.MX 6Dual/6Quad SABRE-SD MFGTool V2:

- Connect a USB cable from a PC to the USB OTG port on the board.
- Connect a USB cable from OTG-to-UART port to PC for console output.
- Open a Terminal emulator program. Refer to Setting Up the Terminal .
- Set boot pin to Mfgtool mode. Refer to How to Enter Serial Download Mode for MFGTool.
- The default profile of the manufacturing tool 2 assumes your file system to be packed and compressed using bzip2 algorithm, an example can be found in the MFGTool release package in the folder Profiles\MX6Q Linux Update\OS Firmware\files. To create this file, you can run the following commands as a root user on Linux OS. You can also modify profile to support other formats.

- >cd *your_rootfs_dir*
- >tar cjf ../rootfs.tar.bz2 *

- You can specify your images in two ways: One way is by editing "Profiles\MX6Q Linux Update\OS Firmware\ucl2.xml" to modify the file path or flash operations to accommodate your needs. After the modification is completed, save the changes and exit. The other way is by replacing the files in "Profiles\MX6Q Linux Update\OS Firmware\files" directory with your files.

NOTE

You will find u-boot-<board>.bin and uImage binaries in "Profiles\MX6Q Linux Update\OS Firmware" folder. Those files are manufacturing firmware loaded by tools and shouldn't be changed unless you need to adapt it for your own board.

- Before running "MfgTool2.exe", see Manufacturing Tool V2 Quick Start Guide to know how to configure Mfgtool2, it can be found at Document\V2 directory in the MFGTOOL release package.
- Execute "MfgTool2.exe" and power on the board. If this is the first time connecting an i.MX 6 board with the MFGTool V2, system will automatically install HID driver for you.
- There is no configuration Menu in MFGTool V2. The configuration is done by parsing ucl2.xml and cfg.ini. After the board has been detected, you will see "HID-compliant device" in the notification bar as shown in the figure below.

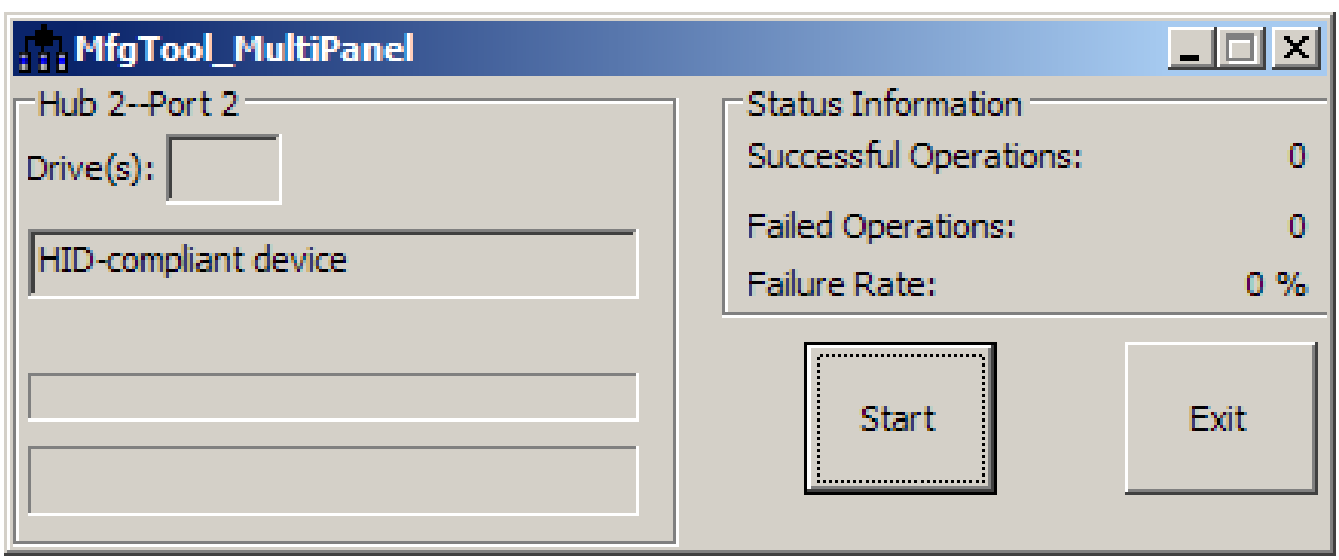

Figure 6-1. Programming SD with Manufacturing Tool – Board Detection

- Start the downloading process by pressing the Start-button. You will see the green progress bar indicating the current task completion as well as the blue progress bar

indicating the whole task completion as shown in the figure below. When you see "Done" in the notification bar, press the Stop-button to finish.

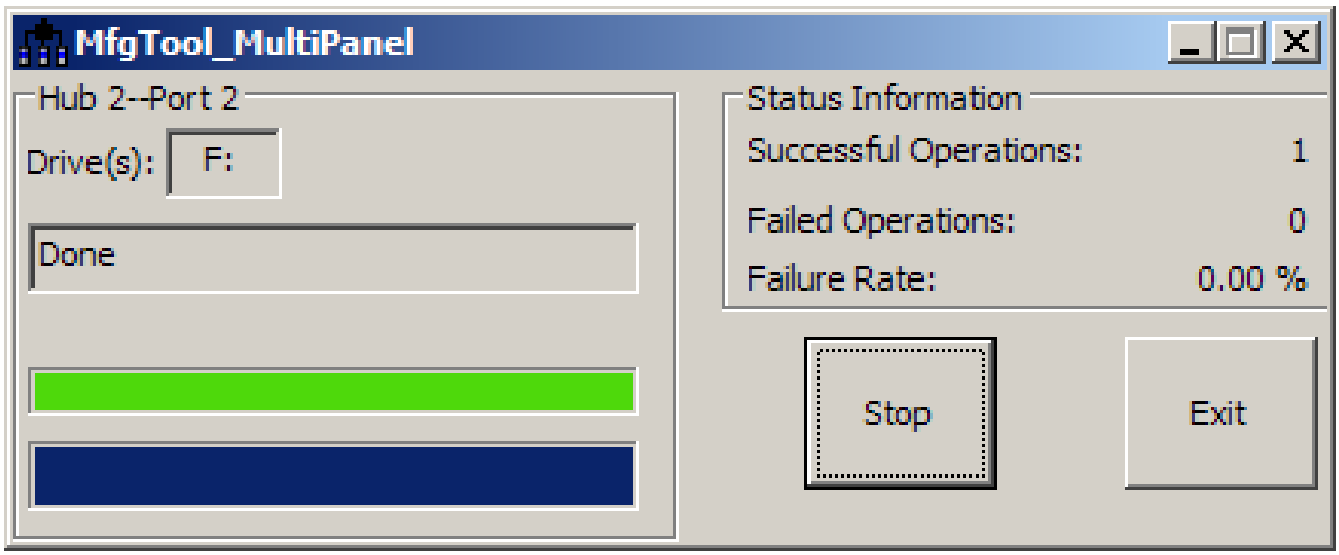

Figure 6-2. Programming SD with Manufacturing Tool – Image downloading

NOTE

The manufacturing tool may sometimes report an error message when it is downloading the file system in an SD card. This can be caused by insufficient space in the SD card due to a small partition size. To fix this, unzip the file "Profiles\MX6Q Linux Update\OS Firmware \mksdcard.sh.tar" and then modify the script to increase the size of the partition and create more partitions according to your file system requirements. After the modification is done, tar the script again.

Chapter 7
Downloading Images by Bootloader or NFS

7.1 Setting Up the Terminal

The i.MX 6Dual/6Quad SABRE-SD board can communicate with a host server (Windows or Linux) by connecting J509 (Lower left corner) and the host server through a micro-B USB cable. The USB to serial driver can be found under www.ftdichip.com/Drivers/VCP.htm. Common serial communication programs such as HyperTerminal, Tera Term, or PuTTY can be used. The example below describes the serial terminal setup using HyperTerminal on a Windows host:

1. Connect the target and the Windows PC using a micro-B USB cable.
2. Open HyperTerminal on the Windows PC and select the settings as shown in the figure below.

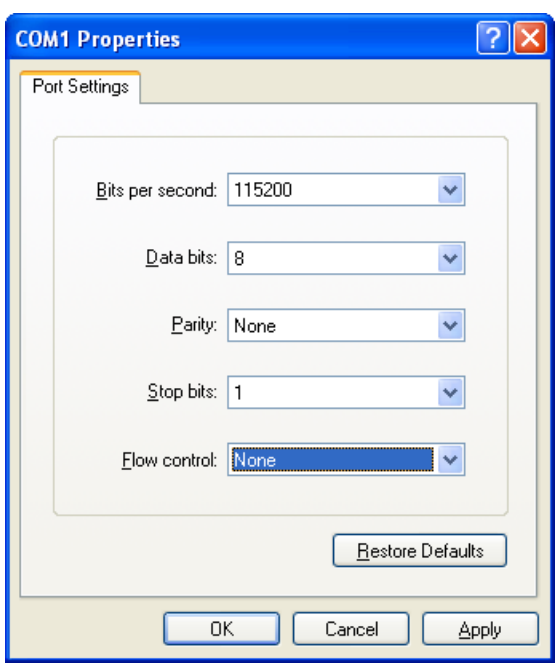

Figure 7-1. HyperTerminal Settings for Terminal Setup

7.2 Downloading Images by U-Boot

The following sections describe how to download images by U-Boot.

7.2.1 MMC/SD on SD3

To enter the U-Boot prompt, press any key before the U-Boot environment variable, "bootdelay", is down counted and before it times out. The default setting is 3 seconds.

1. To clean up the environment variables stored on MMC/SD to their defaults, do the following in U-Boot console:

```
MX6Q SABRESD U-Boot >mmc dev 2
MX6Q SABRESD U-Boot >mmc write 0x10800000 0x600 0x10
```

   or

```
MX6Q SABRESD U-Boot >destroyenv
```

2. Configure U-Boot environment for network communications. Below is an example. The lines preceding with '#' character are comments and have no effect.

```
MX6Q SABRESD U-Boot > setenv bootargs console=ttymxc0,115200
MX6Q SABRESD U-Boot > setenv loadaddr 0x10800000
MX6Q SABRESD U-Boot > setenv serverip 10.192.225.216
MX6Q SABRESD U-Boot > setenv bootfile uImage
### The user can set fake MAC address through ethaddr environment if need
MX6Q SABRESD U-Boot > setenv ethaddr 00:01:02:03:04:05
MX6Q SABRESD U-Boot > saveenv
```

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

```
MX6Q SABRESD U-Boot > dhcp
```

4. Query the information about MMC/SD card in slot 3.

```
MX6Q SABRESD U-Boot >mmc dev 2
MX6Q SABRESD U-Boot >mmcinfo
```

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

```
MX6Q SABRESD U-Boot > help mmc
mmc - MMC sub system
Usage:
mmc read addr blk# cnt
mmc write addr blk# cnt
mmc erase blk# cnt
mmc rescan
mmc part - lists available partition on current mmc device
mmc dev [dev] [part] - show or set current mmc device [partition]
mmc bootpart [dev] [part] - show or set boot partition
mmc list - lists available devices
```

6. Program the kernel uImage located in RAM at ${loadaddr} into the microSD. For example the command to write the image with the size 0x400000 from ${loadaddr} to the offset of 0x100000 of the microSD card. Refer to the following examples for the definition of the mmc Parameters.

```
blk# = (microSD Offset)/(SD block length) = 0x100000/0x200 = 0x800

cnt = (image Size)/(SD block length) = 0x400000/0x200 = 0x2000
```

   This example assumes that the kernel image is less than 0x400000. If the kernel image exceeds 0x400000, increase the image length. After issuing the tftp command, filesize U-Boot environment variable is set with the number of bytes transferred. This can be checked to determine the correct size needed for the calculation. Use U-Boot command printenv to see the value.

```
MX6Q SABRESD U-Boot >mmc dev 2
MX6Q SABRESD U-Boot >mmc write 0x10800000 0x800 0x2000
```

7. Boot up the system through rootfs in SD card through HannStar LVDS:

```
### For LVDS0 connection
MX6Q SABRESD U-Boot > setenv bootargs_mmc 'setenv bootargs ${bootargs} root=/dev/
mmcblk1p1 rootwait rw video=mxcfb0:dev=ldb,LDB-XGA,if=RGB666'
### For LVDS1 connection
MX6Q SABRESD U-Boot > setenv bootargs_mmc 'setenv bootargs ${bootargs} root=/dev/
mmcblk1p1 rootwait rw video=mxcfb0:dev=ldb,LDB-XGA,if=RGB666 ldb=sin1'
MX6Q SABRESD U-Boot >setenv bootcmd_mmc 'run bootargs_base bootargs_mmc;mmc dev 2;mmc
read ${loadaddr} 0x800 0x2000;bootm'
MX6Q SABRESD U-Boot > setenv bootcmd 'run bootcmd_mmc'
MX6Q SABRESD U-Boot > saveenv
```

7.2.2 eMMC4.4 on SDHC4

1. To clean up the environments stored on eMMC4.4, do the following in U-Boot console:

```
MX6Q SABRESD U-Boot > mmc dev 3
MX6Q SABRESD U-Boot > mmc write 0x10800000 0x600 0x10
```

   or

```
 run"destroyenv" command in the console.
```

2. Set correct boot pin config. Power up the board and set the U-Boot environment variables as needed. For example,

```
MX6Q SABRESD U-Boot > setenv bootargs console=ttymxc0,115200
MX6Q SABRESD U-Boot > setenv loadaddr 0x10800000
MX6Q SABRESD U-Boot > setenv serverip 10.192.225.216
MX6Q SABRESD U-Boot > setenv bootfile uImage
### The user can set fake MAC address vi ethaddr environment if need
MX6Q SABRESD U-Boot > setenv ethaddr 00:01:02:03:04:05
MX6Q SABRESD U-Boot > saveenv
```

3. Copy the uImage to TFTP server. Then download it to RAM:

```
MX6Q SABRESD U-Boot > dhcp
```

4. Query the information about MMC/SD card.

```
MX6Q SABRESD U-Boot >
mmc dev 3
mmcinfo
```

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

```
MX6Q SABRESD U-Boot > help mmc

mmc - MMC sub system
Usage:
mmc read addr blk# cnt
mmc write addr blk# cnt
mmc erase blk# cnt
mmc rescan
mmc part - lists available partition on current mmc device
mmc dev [dev] [part] - show or set current mmc device [partition]
mmc bootpart [dev] [part] - show or set boot partition
mmc list - lists available devices
```

6. Program the kernel uImage into eMMC4.4. For example, the command below writes the image with the size 0x400000 from ${loadaddr} to the offset 0x100000 of the MMC/SD card. Here, the following equation applies: 0x800 =0x100000/0x200, 0x2000=0x400000/0x200. The block size of this card is 0x200. This example assumes the kernel image is less than 0x400000 bytes. If the kernel image exceeds 0x400000, enlarge the image length.

```
MX6Q SABRESD U-Boot > mmc dev 3
MX6Q SABRESD U-Boot > mmc write ${loadaddr} 0x800 0x2000
```

7. Boot up the system through RFS in eMMC4.4 through HannStar LVDS:

```
### For LVDS0 connection
MX6Q SABRESD U-Boot > setenv bootargs_mmc 'setenv bootargs ${bootargs}
root=/dev/mmcblk0p1 rootwait rw video=mxcfb0:dev=ldb,LDB-XGA,if=RGB666'
### For LVDS1 connection
MX6Q SABRESD U-Boot > setenv bootargs_mmc 'setenv bootargs ${bootargs}
root=/dev/mmcblk0p1 rootwait rw video=mxcfb0:dev=ldb,LDB-XGA,if=RGB666 ldb=sin1'
MX6Q SABRESD U-Boot > setenv bootcmd_mmc 'run bootargs_base bootargs_mmc;mmc dev 3
;mmc read ${loadaddr}  0x800 0x2000;bootm'
MX6Q SABRESD U-Boot > setenv bootcmd 'run bootcmd_mmc'
MX6Q SABRESD U-Boot > saveenv
```

8. Boot up the system through RFS in eMMC4.4 through CLAA WVGA panel:

```
MX6Q SABRESD U-Boot > setenv bootargs_mmc 'setenv bootargs ${bootargs}
root=/dev/mmcblk0p1 rootwait rw video=mxcfb0:dev=lcd,CLAA-WVGA,if=RGB565 ip=dhcp'
```

9. Boot up the system through RFS in eMMC4.4 through HDMI:

```
MX6Q SABRESD U-Boot > setenv bootargs_mmc 'setenv bootargs ${bootargs}
root=/dev/mmcblk0p1 rootwait rw video=mxcfb1:dev=ldb,LDB-XGA,if=RGB666
```

```
video=mxcfb0:dev=hdmi,1920x1080M@60,if=RGB24 ip=dhcp'
```

10. To program the rootfs to MMC/SD, see Using i.MX 6Dual/6Quad SABRE-SD Board as Host Server to Create rootfs, or Using a Linux Host to Set Up an SD/MMC Card.

7.3 U-Boot Configurations

The U-Boot "print" command can be used to check environment variable values.

The "setenv" command can be used to set environment variable values. See the U-Boot user guide for details.

7.4 Using i.MX 6Dual/6Quad SABRE-SD Board 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.MX 6Dual/6Quad SABRE-SD as Linux Host server to create the rootfs on MMC/SD cardor SATA device. The example below is SD card. Device file node name needs to be changed for SATA device.

1. Boot from NFS or other storage. Check partitions information:

```
root@freescale ~$ cat /proc/partitions
```

2. To create a partition in MMC/SD Slot 3, use the fdisk command in the Linux console:

```
root@freescale ~$ fdisk /dev/mmcblk1
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/mmcblk1: 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
```

3. As described in Flash Memory Map, the rootfs partition should be located after kernel image; the first 0x800000 bytes can be reserved for MBR, bootloader, and kernel sections. From the log shown above, the Units of current MMC/SD card is 32768 bytes. The beginning 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:

```
         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-read mmcblk1 partition table
          p1
```

4. Format the MMC/SD partitions as types ext3 or ext4 type. For example, to use ext3:

```
root@freescale ~$ mkfs.ext3 /dev/mmcblk1
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.
```

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

```
mount -t ext2 -o loop /rootfs.ext2 /mnt/cdrom
cd /mnt
mkdir mmcblk1p1
mount -t ext3 /dev/mmcblk1p1 /mnt/mmcblk1p1/
cp -af /mnt/cdrom/* /mnt/mmcblk1p1/
umount /mnt/mmcblk1p1
umount /mnt/cdrom
```

6. Type sync to write the contents to MMC/SD.
7. Type poweroff to power down the system. Follow the instructions in Running the Image on the Target to boot the image from MMC/SD card.

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 up an i.MX 6Dual/6Quad SABRE-SD board.

8.1 Requirements

An SD/MMC card reader, like a USB card reader, is required. It will be used to transfer the boot loader and kernel 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 such as Ubuntu 9.04.

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/sdb (a block is 1kB large).

NOTE

Make sure that the device node is correct for the SD/MMC card. Otherwise, it may damage your operating system or data on your computer.

8.2 Copying the Boot Loader Image

Enter the following command to copy the U-Boot image to the SD/MMC card:

```
$ sudo dd if=u-boot-mx6q-sabresd.bin  of=/dev/sdb bs=512 seek=2 skip=2 conv=fsync
```

This assumes a pre-built U-Boot image delivered with the BSP or built from the U-Boot source code. If using a non-padded U-Boot image, "skip=2" should be omitted from the above command line. The first 1 KB of the SD/MMC card, that includes the partition table, will be preserved.

8.3 Copying the Kernel Image

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

```
$ sudo dd if=uImage of=/dev/sdb bs=512 seek=2048 conv=fsync
```

This will copy uImage to the media at offset 1 MB (bs x seek = 512 x 2048 = 1MB).

8.4 Copying the Root File System (rootfs)

First, a partition table must be 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 16384 (in sectors of 512 bytes) enter the following command:

```
$ sudo fdisk /dev/sdb
```

NOTE

On most Linux host operating systems, SD card will be mounted automatically upon insertion. Therefore, before running fdisk, please make sure that SD card is unmounted (via 'sudo umount /dev/sdb').

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]
16384   [starting at offset sector #16384, i.e. 8MB, 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/sdb1
Or
$ sudo mkfs.ext4 /dev/sdb1
```

Copy the target file system to the partition:

```
$ mkdir /home/user/mountpoint
$ sudo mount /dev/sdb1 /home/user/mountpoint
```

Extract rootfs package to certain directory: extract rootfs.ext2.gz to /home/user/rootfs for example:

```
$ gunzip rootfs.ext2.gz
$ mount -o loop -t ext2 rootfs.ext2 /home/user/rootfs
```

Assume that the root file system files are located in /home/user/rootfs as in the previous step:

```
$ cd /home/user/rootfs
$ sudo cp -a * /home/user/mountpoint
$ sudo umount /home/user/mountpoint
```

NOTE

Copying the file system takes several minutes depending on the size of your rootfs.

The file system content is now on the media.

Chapter 9
Running the Image on the Target

This chapter explains how to run an image on the target from downloaded device and NFS.

These instructions assume that you have downloaded the kernel image using the instructions in either Downloading Images by Using MFGTool, or Downloading Images by Bootloader or NFS, or Using a Linux Host to Set Up an SD/MMC Card. If you have not setup your Serial Terminal yet, please refer to Setting Up the Terminal .

9.1 Running the Image from NFS

To boot from NFS, do as follows:

1. Power on the board.
2. Enter the following commands in the U-Boot prompt:

```
MX6Q SABRESD U-Boot > setenv serverip 10.192.225.216
MX6Q SABRESD U-Boot > setenv bootfile uImage
MX6Q SABRESD U-Boot > setenv nfsroot /data/rootfs_home/rootfs_mx6
MX6Q SABRESD U-Boot > setenv bootargs_base 'setenv bootargs console=ttymxc0,115200'
### For LVDS0 connection
MX6Q SABRESD U-Boot > setenv bootargs_nfs 'setenv bootargs ${bootargs} root=/dev/nfs ip=dhcp
nfsroot=${serverip}:${nfsroot},v3,tcp video=mxcfb0:dev=ldb,LDB-XGA,if=RGB666'
### For LVDS1 connection
MX6Q SABRESD U-Boot > setenv bootargs_nfs 'setenv bootargs ${bootargs} root=/dev/nfs ip=dhcp
nfsroot=${serverip}:${nfsroot},v3,tcp video=mxcfb0:dev=ldb,LDB-XGA,if=RGB666 ldb=sin1'
### HDMI
MX6Q SABRESD U-Boot > setenv bootargs_nfs 'setenv bootargs ${bootargs} root=/dev/nfs
ip=dhcp nfsroot=${serverip}:${nfsroot},v3,tcp video=mxcfb1:dev=ldb,LDB-XGA,if=RGB666
video=mxcfb0:dev=hdmi,1920x1080M@60,if=RGB24'

MX6Q SABRESD U-Boot > setenv bootcmd_net 'run bootargs_base bootargs_nfs;bootm'
MX6Q SABRESD U-Boot > setenv bootcmd 'dhcp; run bootcmd_net'
MX6Q SABRESD U-Boot > saveenv
```

NOTE

If the MAC address has not burned into fuse, you must set the MAC address to use network in U-Boot.

```
setenv ethaddr xx:xx:xx:xx:xx:xx
```

9.2 Running the Image from MMC/SD

To boot the system from MMC/SD flash, do as follows:

1. Power on the board.
2. Assume that the kernel image starts from the address 0x100000 byte (the block start address is 0x800). The kernel image size is less than 0x400000 byte. Enter the following commands in the U-Boot prompt:

```
MX6Q SABRESD U-Boot > setenv loadaddr 0x10800000
MX6Q SABRESD U-Boot > setenv bootargs_base 'setenv bootargs console=ttymxc0,115200'
MX6Q SABRESD U-Boot > setenv bootargs_mmc 'setenv bootargs ${bootargs} root=/dev/
mmcblk1p1 rootwait rw video=mxcfb1:dev=ldb,LDB-XGA,if=RGB666 video=mxcfb0:dev=hdmi,
1920x1080M@60,if=RGB24'
MX6Q SABRESD U-Boot > setenv bootcmd_mmc 'run bootargs_base bootargs_mmc;mmc dev 2;mmc
read ${loadaddr} 0x800 0x2000;bootm'
MX6Q SABRESD U-Boot > setenv bootcmd 'run bootcmd_mmc'
MX6Q SABRESD U-Boot > saveenv
MX6Q SABRESD U-Boot > run bootcmd
```

NOTE

The above commands read the kernel image from the SD3

3. To read kernel image from eMMC4.4, update the command line as shown below:

```
MX6Q SABRESD U-Boot >setenv bootcmd_mmc 'run bootargs_base bootargs_mmc;mmc dev 3;mmc
read ${loadaddr} 0x800 0x2000;bootm'
```

```
MX6Q SABRESD U-Boot > setenv bootargs_mmc 'setenv bootargs ${bootargs} root=/dev/
mmcblk0p1 rootwait rw video=mxcfb1:dev=ldb,LDB-XGA,if=RGB666 video=mxcfb0:dev=hdmi,
1920x1080M@60,if=RGB24 ip=dhcp'
```

Chapter 10
Frequently Asked Questions

The following paragraphs address the most commonly asked questions related to Linux BSP.

10.1 How to Fuse in U-Boot

U-Boot contains a tool, imxotp, which is used for fusing.

```
U-Boot > imxotp
imxotp - One-Time Programable sub-system

Usage:
imxotp imxotp read <index>
 - read fuse at 'index'
imxotp blow [--force] <index> <value>
 - blow fuse at 'index' with hex value 'value'
Tips:
'addr' to 'index':
convert 'index' from 'address'
index = (addr - otp_base) / 0x10
eg, addr is 0x021bc410, otp_base is 0x021bc400, the index = 1
```

- '--force' must be present in order to blow the fuse. Command will abort if '--force' is missing.
- index = (addr - otp_base) / 0x10, where the addr is the address of the fuse you want to operate, the otp_base is the base address of the fuse block.
- 'value' should correspond to fuse settings according to the fuse map and desired fuse configuration.

10.2 How to Enable Secure Boot

Freescale i.MX 6 series chips support High Assurance Boot (HAB). The secure boot option is off by default and can be turned on by burning some eFuses. See *i.MX 6 Linux High Assurance Boot (HAB) User Guide* for detailed instructions on how to enable this feature.

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number: IMX6DQLXUGSSD
Rev. L3.0.35_4.1.0
09/2013