Freescale Semiconductor
User's Guide

Document Number: AUG
Rev. L5.0.0_1.0.0-ga, 06/2015

Android™ User's Guide

Contents

1 Overview....1
2 Preparation.... 1
3 Building the Android platform for i.MX....2
4 Running the Android Platform with a Prebuilt Image.... 11
5 Programming Images.... 14
6 Booting.... 17

1 Overview

This document provides the technical information related to the i.MX 6 series SABRE devices:

- Instructions for building from sources or using pre-built images
- Copying the images to a boot media
- Hardware/software configurations for programming the boot media and running the images

This document describes how to configure a Linux® OS build machine and provides the steps to download, patch, and build the software components that create the Android™ system image when working with the sources.

For more information about building the Android platform, see source.android.com/source/building.html.

2 Preparation

2.1 Setting up your computer

To build the Android source files, you will need to use a computer running Linux OS.

You also need to use the 14.04 64-bit version of Ubuntu which are the most tested OS for the Android Lollipop 5.0.0 build.

After installing the computer running Linux OS, you need to check whether you have all the necessary packages installed for an Android build. See "Setting up your machine" on the Android website source.android.com/source/initializing.html.

In addition to the packages requested on the Android website, the following packages are also needed:

```
$ sudo apt-get install uuid uuid-dev
$ sudo apt-get install zlib1g-dev liblz-dev
$ sudo apt-get install liblzo2-2 liblzo2-dev
$ sudo apt-get install lzop
$ sudo apt-get install git-core curl
$ sudo apt-get install u-boot-tools
$ sudo apt-get install mtd-utils
```

NOTE

If you have trouble in installing the JDK in Ubuntu, see community.freescale.com/docs/DOC-98441.

To install git, it is recommended to set teh name and email as follows:

- git config -- global user.name "First Last";
- git config -- gobal user.email "first.last@company.com"

2.2 Unpacking the Android release package

After you have set up a computer running Linux OS, unpack the Android release package by using the following commands:

```
# cd /opt (or any other directory where you placed the android_L5.0.0_1.0.0-
ga_core_source.tar.gz file, but make sure you have permission to access that directory;
otherwise, c_patch cannot work properly)

$ tar xzvf android_L5.0.0_1.0.0-ga_core_source.tar.gz

$ cd android_L5.0.0_1.0.0-ga_core_source/code/

$ tar xzvf l5.0.0_1.0.0-ga.tar.gz
```

3 Building the Android platform for i.MX

3.1 Getting Android source code (Android/kernel/U-Boot)

The Android source code is maintained as more than 100 gits in the Android repository (android.googlesource.com).

To get the Android source code from Google repo, follow the steps below:

```
$ cd ~
$ mkdir myandroid
$ mkdir bin
$ cd myandroid
$ curl http://commondatastorage.googleapis.com/git-repo-downloads/repo > ~/bin/repo
$ chmod a+x ~/bin/repo
$ ~/bin/repo init -u https://android.googlesource.com/platform/manifest -b android-5.0.2_r1
$ ~/bin/repo sync # this command loads most needed repos. Therefore, it can take several
hours to load.
$ cd ~/myandroid/prebuilts/gcc/linux-x86/arm
$ git clone https://android.googlesource.com/platform/prebuilts/gcc/linux-x86/arm/arm-
eabi-4.6
$ cd arm-eabi-4.6
$ git checkout android-4.4.3_r1
```

NOTE

Because the default GCC toolchain from android-5.0.2_r1 release is GCC4.8, which may have issues on ARM Linux kernel compiling, the GCC 4.0 from android-4.4.3_r1 is used to compile the Linux Kernel and U-Boot.

Get kernel source code from Freescale open source git:

```
$ cd myandroid

$ git clone git://git.freescale.com/imx/linux-2.6-imx.git kernel_imx # the kernel repo is
large. Therefore, this process can take a while.

$ cd kernel_imx

$ git checkout l5.0.0_1.0.0-ga
```

NOTE

If you are behind a proxy, use socksify to set socks proxy for git protocol.

If you use U-Boot as your bootloader, then you can clone the U-Boot git repository from Freescale open source git:

```
$ cd myandroid/bootable

$ cd bootloader

$ git clone git://git.freescale.com/imx/uboot-imx.git uboot-imx

$ cd uboot-imx

$ git checkout l5.0.0_1.0.0-ga
```

3.2 Patch code for i.MX

Apply all the i.MX Android patches by performing the following steps:

1. Assume you have unzipped the i.MX Android release package to `/opt/android_L5.0.0_1.0.0-ga_source`.

   ```
   $ cd ~/myandroid
   $ source /opt/android_L5.0.0_1.0.0-ga_core_source/code/l5.0.0_1.0.0-ga/and_patch.sh
   $ help
   ```
2. You should see that the "c_patch" function is available.

   ```
   $ c_patch /opt/android_L5.0.0_1.0.0-ga_core_source/code/l5.0.0_1.0.0-ga
   imx_l5.0.0_1.0.0-ga
   ```

   Here, "/opt/android_L5.0.0_1.0.0-ga_source/code/L5.0.0_1.0.0-ga" is the location of the patches, which is the directory created when you unzip the release package.

   "imx_L5.0.0_1.0.0-ga" is the branch which will be created automatically for you to hold all patches (only in those existing Google gits).

   Choose any branch name instead of "imx_L5.0.0_1.0.0-ga".
3. If everything is OK, "c_patch" generates the following output to indicate the successful patch:

   ```
   ************************************************************
   Success: Now you can build the Android code for FSL i.MX platform
   ************************************************************
   ```

NOTE

The patch script (and_patch.sh) requires some basic utilities like `awk/sed`. If they are not available on the computer running Linux OS, install them first.

3.3 Patch Freescale extended features code

Besides the core features code package, Freescale also provides extended features code packages. These extended features code packages bring more features. The following tables list the extended features code packages.

Table 1. Extended features code package

Package name	Package Description
android_L5.0.0_1.0.0-ga_omxplayer_source.tar.gz	Source code package, binaries and scripts for using omx player
android_L5.0.0_1.0.0-ga_wfdsink_source.tar.gz	Source code package to add Freescale WIFI Sink support into the core features code package

To apply android_L5.0.0_1.0.0-ga_omxplayer_source.tar.gz patches, carry out the following commands:

```
$ cp android_L5.0.0_1.0.0-ga_omxplayer_source.tar.gz ~/myandroid
$ cd ~/myandroid
$ tar xzvf android_L5.0.0_1.0.0-ga_omxplayer_source.tar.gz
```

After unpacking this package, the current build mode is “full”, which means omxplayer feature will be built into the Android platform.

Use switch_build_to.sh to switch between “core” mode and “full” mode.

```
$ ./switch_build_to.sh          (display current mode)
$ ./switch_build_to.sh mode     (mode shall be "core" or "full")
$ ./clean_obj_before_building.sh (clean related binary in the out directory of previous
Android build)
```

NOTE

Run this script after running “lunch” and before running “make”.

To apply android_L5.0.0_1.0.0-ga_wfdsink_source.tar.gz patches, carry out the following commands:

```
$ cd /opt (or any other directory you like)
$ tar xzvf android_L5.0.0_1.0.0-ga_wfdsink_source.tar.gz
$ cp -a android_L5.0.0_1.0.0-ga_wfdsink_source/wfd-proprietary ~/myandroid/device/
```

3.4 Building Android images

Building the Android image is performed when the source code has been downloaded (Section 3.1) and patched (Section 3.2 and Section 3.3 if needed).

Two commands are executed to build: one is **lunch <buildName-buildType>** to set up the build configuration, and the other is **make** to start the build process.

The build configuration command **lunch** can be issued with an argument Build Name – Build Type string, such as **lunch sabresd_6dq-user**, or can be issued without the argument presenting a menu of selection.

The Build Name is the Android device name found directory ~/myandroid/device/fsl/. The following table lists the Freescale build names.

Table 2. Build names

Build name	Description
evk_6sl	i.MX 6SoloLite Eval Kit
sabreauto_6q	i.MX 6Quad Auto Infotainment
sabresd_6dq	i.MX 6DualQuad SABRE Board and SABRE Platform
sabresd_6sx	i.MX 6SoloX SABRE Board
sabreauto_6sx	i.MX 6SoloX Auto Infotainment

The build type is used to specify what debug options are provided in the final image. The following table lists the build types.

Table 3. Build types

Build type	Description
user	Production ready image, no debug
userdebug	Provides image with root access and debug, similar to "user"
eng	Development image with debug tools

Android build steps are as follows:

1. Change to the top level build directory.

   ```
   $ cd ~/myandroid
   ```
2. Set up the environment for building. This only configures the current terminal.

   ```
   $ source build/envsetup.sh
   ```
3. Execute the Android **lunch** command. In this example, the setup is for the production image of i.MX 6DualQuad SABRE Board/Platform device.

   ```
   $ lunch sabresd_6dq-user
   ```
4. Execute the **make** command to generate the image.

   ```
   $ make 2>&1 | tee build-log.txt
   ```

When the **make** command is complete, the build-log.txt file contains the execution output. Please check for any errors.

For BUILD_ID & BUILD_NUMBER, add a buildspec.mk in your ~/myandroid directory. For details, see the *Android Frequently Asked Questions* (AFAQ).

For i.MX 6Dual/6Quad SABRE-SD and i.MX 6DualLite SABRE-SD boards, we use the same build configuration. The two boards share the same kernel/system/recovery images with the exception of the U-Boot image. The following outputs are generated by default in myandroid/out/target/product/sabresd_6dq:

- root/: root file system (including init, init.rc). Mounted at /
- system/: Android system binary/libraries. Mounted at /system.
- data/: Android data area. Mounted at /data.
- recovery/: root file system when booting in "recovery" mode. Not used directly.
- boot-imx6q.img: composite image for i.MX 6Dual/6Quad SABRE-SD. which includes the kernel zImage, ramdisk, board's device tree binary, and boot parameters.
- boot-imx6dl.img: composite image for i.MX 6DualLite SABRE-SD, which includes the kernel zImage, ramdisk, board's device tree binary, and boot parameters.

- boot-imx6q-ldo.img: a composite image for i.MX 6DualQuad 1.2GHZ SABRE-SD, which includes the kernel zImage, ramdisk, board's device tree binary, and boot parameters.
- ramdisk.img: ramdisk image generated from "root/". Not used directly.
- system.img: EXT4 image generated from "system/". It can be programmed to "SYSTEM" partition on SD/eMMC card with "dd".
- recovery-imx6q.img: EXT4 image for i.MX 6Dual/6Quad SABRE-SD, which is generated from "recovery/". Can be programmed to the "RECOVERY" partition on SD/eMMC card with "dd".
- recovery-imx6q-ldo.img: EXT4 image for i.MX 6Dual/6Quad 1.2GHZ SABRE-SD, which is generated from "recovery/". Can be programmed to "RECOVERY" partition on SD/eMMC card with "dd".
- recovery-imx6dl.img: EXT4 image for i.MX 6DualLite SABRE-SD, which is generated from "recovery/". Can be programmed to the "RECOVERY" partition on SD/eMMC card with "dd".
- u-boot-imx6q.imx: U-Boot image with no padding for i.MX 6Dual/6Quad SABRE-SD.
- u-boot-imx6dl.imx: U-Boot image with no padding for i.MX 6DualLite SABRE-SD.

NOTE

- To build the U-Boot image separately, see Building U-Boot images.
- To build the kernel uImage separately, see Building a kernel image.
- To build boot.img, see Building boot.img.

The default build is configured for internal eMMC boot storage. See Building Android image for the SD card on the SABRE-SD Board for the configuration steps to make SD card in Slot 3 as the boot storage.

3.4.1 Configuration examples of building Freescale devices

The following table shows examples of using the `lunch` command to set up different Freescale devices. After the desired Freescale device is set up, the `make` command is used to start the build.

Table 4. Freescale device lunch examples

Build name	Description
i.MX 6Quad SABRE-SD Board and Platform	$ lunch sabresd_6dq-user
i.MX 6Quad SABRE-AI Board	$ lunch sabreauto_6dq-user
i.MX 6SoloLite EVK Board	$ lunch evk_6sl-user
i.MX 6SoloX SABRE-SD Board	$ lunch sabresd_6sx-user
i.MX 6SoloX SABRE-AI Board	$ lunch sabreauto_6sx-user

After the `lunch` command is executed, the **make** command is issued:

```
$ make 2>&1 | tee build-log.txt
```

3.4.2 User build mode

A production release Android system image is created by using the **user** Build Type. For configuration options, see Table "Build types" in Section Building Android images.

The notable differences between the **user** and **eng** build types are as follows:

- Limited Android System image access for security reasons.
- Lack of debugging tools.
- Installation modules tagged with user.

- APK's and tools according to product definition files, which are found in PRODUCT_PACKAGES in the sources folder ~/myandroid/device/fsl/imx6/imx6.mk. To add customized packages, add the package MODULE_NAME or PACKAGE_NAME to this list.
- The properties are set as: `ro.secure=1` and `ro.debuggable=0`.
- `adb` is disabled by default.

The Freescale development tool options are shown below from: freescale.com/imx6tools.

Table 5. Development tool options

	i.MX 6Quad	i.MX 6Dual	i.MX 6DualLite	i.MX 6Solo	i.MX 6SoloX	i.MX 6SoloLite
SABRE Board for Smart Devices	*	Uses i.MX 6Quad	-	-	*	-
SABRE Platform for Smart Devices	*	Uses i.MX 6Quad	*	Uses i.MX 6DualLite	-	-
SABRE for Automotive Infotainment	*	Uses i.MX 6Quad	*	Uses i.MX 6DualLite	-	-
i.MX 6SoloLite Evaluation Kit	-	-	-	-	-	*

* Supported through the superset device (i.MX 6Quad is superset to i.MX 6Dual, i.MX 6DualLite is superset to i.MX 6Solo)

Table 6. Android system image production build method 1

Freescale development tool	Description	Image build command
SABRE Board/Platform for Smart Devices	i.MX 6Dual/Quad	$ make PRODUCT-sabresd_6dq-user 2>&1 | tee build-log.txt
	i.MX 6SoloX	$ make PRODUCT-sabresd_6sx-user 2>&1 | tee build-log.txt
SABRE for Automotive Infotainment	i.MX 6Dual/Quad , i.MX 6DualLite	$ make PRODUCT-sabreauto_6q-user 2>&1 | tee build-log.txt
	i.MX 6SoloX	$ make PRODUCT-sabreauto_6sx-user 2>&1 | tee build-log.txt
SoloLite Evaluation Kit	i.MX 6SoloLite	$ make PRODUCT-evk_6sl-user 2>&1 | tee build-log.txt

Another method to create Android System Images is setting the environment and then issuing the `make` command. To set up the environment, execute the script `~/myandroid/build/envsetup.sh`. The `lunch` command and configuration argument for the Freescale Development Tool is then executed. To start the build, use the `make` command. The following table lists the `lunch` configuration values.

Table 7. Android system image production build method 2

Freescale development tool	Description	Lunch configuration
SABRE Board/Platform for Smart Devices	i.MX 6Dual/Quad	sabresd_6dq-user
	i.MX 6SoloX	sabresd_6sx-user
SABRE for Automotive Infotainment	i.MX 6Dual/Quad , i.MX 6DualLite	sabreauto_6q-user

Table continues on the next page...

Table 7. Android system image production build method 2 (continued)

Freescale development tool	Description	Lunch configuration
	i.MX 6SoloX	sabreauto_6sx-user
SoloLite Evaluation Kit	i.MX 6SoloLite	evk_6sl-user

An example for the SABRE Board for Smart Devices i.MX 6Dual/Quad is:

```
$ cd ~/myandroid
$ source build/envsetup.sh
$ lunch sabresd_6dq-user
$ make
```

To create Android over-the-air, OTA, and package, the following make target is specified:

```
$ make otapackage
```

For more Android building information, see source.android.com/source/building.html.

3.4.3 Building Android image for the SD card on the SABRE-SD Board

The default configuration in the source code package takes internal eMMC as the boot storage. It can be changed to make the SD card in SD Slot 3 as the boot storage as follows:

```
remove out/target/product/sabresd_6dq/root directory and boot*.img
remove /out/target/product/sabresd_6dq/recovery directory and recovery*.img
remove /out/target/product/sabresd_6dq/system directory and system*.img
$ make BUILD_TARGET_DEVICE=sd
```

Follow Section 3.4 to build the images.

3.4.4 Building Android images for NAND on the SABRE-AI board

i.MX 6Quad/DualLite/Solo SABRE-AI platform:

The default configuration in the source code package takes SD card as the boot storage for i.MX 6 series SABRE-AI boards.

The default setting can be changed to make the NAND Flash in U19 be the boot storage as shown below:

```
$ make PRODUCT-sabreauto_6q-user BUILD_TARGET_FS=ubifs
```

i.MX 6SoloX SABRE-AI platform:

The default configuration in the source code package takes SD as the boot storage for SABRE-AI. The default setting can be changed to make the NAND Flash in U19 to be the boot storage as shown below:

```
$ make PRODUCT-sabreauto_6sx-user BUILD_TARGET_FS=ubifs
```

3.5 Building U-Boot images

```
$ cd ~/myandroid/bootable/bootloader/uboot-imx
$ export ARCH=arm
```

```
$ export CROSS_COMPILE=~/myandroid/prebuilts/gcc/linux-x86/arm/arm-eabi-4.6/bin/arm-eabi-
$ make distclean
```

For i.MX 6Quad SABRE-SD:

```
$ make mx6qsabresdandroid_config
$ make
```

For i.MX 6DualLite SABRE-SD:

```
$ make mx6dlsabresdandroid_config
$ make
```

For i.MX 6Solo SABRE-SD:

```
$ make mx6solosabresdandroid_config
```

For i.MX 6Quad SABRE-AI SD:

```
$ make mx6qsabreautoandroid_config
$ make
```

For i.MX 6Quad SABRE-AI NAND:

```
$ make mx6qsabreautoandroid_nand_config
$ make
```

For i.MX 6DualLite SABRE-AI SD:

```
$ make mx6dlsabreautoandroid_config
$ make
```

For i.MX 6DualLite SABRE-AI NAND:

```
$ make mx6dlsabreautoandroid_nand_config
$ make
```

For i.MX 6Solo SABRE-AI SD:

```
$ make mx6solosabreautoandroid_config
$ make
```

For i.MX 6Solo SABRE-AI NAND:

```
$ make mx6solosabresdandroid_nand_config
```

For i.MX 6SoloLite EVK SD:

```
$ make mx6slevkandroid_config
```

For i.MX 6SoloX SABRE-SD SD:

```
$ make mx6sxsabresdandroid_config
```

For i.MX 6SoloX SABRE-AI SD:

```
$ make mx6sxsabreautoandroid_config
```

For i.MX 6SoloX SABRE-AI NAND:

```
$ make mx6sxsabreautoandroid_nand_config
$ make
```

"u-boot.imx" is generated if you have a successful build.

NOTE

Any image that should be loaded by U-Boot must have an unique image head. For example, data must be added at the head of the loaded image to tell U-Boot about the image (i.e., it's a kernel, or ramfs) and how to load the image (i.e., load/execute address). Before loading any image into RAM by U-Boot, you need a tool to add this information and generate a new image which can be recognized by U-Boot. The tool is delivered together with U-Boot. After you set up U-Boot using the steps outlined above, you can find the tool (mkimage) under tools/. The process of using mkimage to generate an image (for example, kernel image and ramfs image), which is to be loaded by U-Boot, is outlined in the subsequent sections of this document.

3.6 Building a kernel image

Kernel image is built while building the Android root file system.

If you do not need to build the kernel image, skip this section.

To run the Android platform using NFS or from SD, build the kernel with the default configuration as follows:

Assume you had already built U-Boot. mkimage was generated under myandroid/bootable/bootloader/uboot-imx/tools/ and it is in your PATH.

```
$ export PATH=~/myandroid/bootable/bootloader/uboot-imx/tools:$PATH
$ cd ~/myandroid/kernel_imx
$ echo $ARCH && echo $CROSS_COMPILE
```

Make sure that you have those two environment variables set. If the two variables are not set, set them as follows:

```
$ export ARCH=arm
$ export CROSS_COMPILE=~/myandroid/prebuilts/gcc/linux-x86/arm/arm-eabi-4.6/bin/arm-eabi-
# Generate ".config" according to default config file under arch/arm/configs.
# To build the kernel image for i.MX 6Dual/Quad, 6DualLite, 6Solo, 6SoloLite, and 6SoloX
$ make imx_v7_android_defconfig

# To build the kernel image for i.MX 6Dual/Quad, 6DualLite, and 6Solo
$ make uImage LOADADDR=0x10008000

# To build the kernel image for i.MX 6SoloLite
$ make uImage LOADADDR=0x80008000

# To build the kernel image for i.MX 6SoloX
$ make uImage LOADADDR=0x80008000
```

The kernel images are found in the folders: ~/myandroid/kernel_imx/arch/arm/boot/zImage and ~/myandroid/kernel_imx/arch/arm/boot/uImage.

3.7 Building boot.img

As outlined in Running the Android Platform with a Prebuilt Image, we use boot.img and booti as default commands to boot, not the uramdisk and uImage we used before.

Use this command to generate boot.img under Android environment:

```
# Boot image for the i.MX 6Dual/6Quad SABRE-SD board
$ cd ~/myandroid
$ source build/envsetup.sh
$ lunch sabresd_6dq-user
$ make bootimage

# Boot image for the i.MX 6Dual/6Quad SABRE-AI board
```

```
$ source build/envsetup.sh
$ lunch sabreauto_6q-user
$ make bootimage

# Boot image for the i.MX 6SoloLite EVK board
$ source build/envsetup.sh
$ lunch evk_6sl-user
$ make bootimage

# Boot image for the i.MX 6SoloX SABRE-SD board
$ source build/envsetup.sh
$ lunch sabresd_6sx-user
$ make bootimage

# Boot image for the i.MX 6SoloX SABRE-AI board
$ source build/envsetup.sh
$ lunch sabreauto_6sx-user
$ make bootimage
```

4 Running the Android Platform with a Prebuilt Image

To test the Android platform before building any code, use the prebuilt images from the following packages and go to "Download Images" and "Boot".

Table 8. Image packages

Image package	Description
android_L5.0.0_1.0.0-ga_core_image_6qsabresd.tar.gz	The table below shows the prebuilt image for i.MX 6Dual/6Quad SABRE-SD board and platform and i.MX 6Solo/6DualLite SABRE-SD platform, which has basic Android features.
android_L5.0.0_1.0.0-ga_core_image_6qsabreauto.tar.gz	The table below shows the prebuilt image for i.MX 6Dual/6Quad and i.MX 6Solo/6DualLite SABRE-AI platform, which has basic Android features.
android_L5.0.0_1.0.0-ga_core_image_6slevk.tar.gz	The table below shows the prebuilt image for the i.MX 6SoloLite EVK board, which has basic Android features.
android_L5.0.0_1.0.0-ga_core_image_6sxsabresd.tar.gz	Prebuilt image for the i.MX 6SoloX Sabre-SD board, which has basic Android features.
android_L5.0.0_1.0.0-ga_core_image_6sxsabreauto.tar.gz	Prebuilt image for i.MX 6SoloX SABRE-AI board, which has basic Android features.
android_L5.0.0_1.0.0-ga_full_image_6qsabresd.tar.gz	The table below shows the prebuilt image for i.MX 6Dual/Quad and i.MX 6DualLite SABRE-SD board, which includes Freescale extended features. For more information and details about the Freescale Extended Feature Packages available for this release, see the *Android™ Release Notes* (ARN).
android_L5.0.0_1.0.0-ga_full_image_6qsabreauto.tar.gz	The table below shows the prebuilt image for i.MX 6Dual/Quad and i.MX 6DualLite/Solo SabreAI board, which includes Freescale extended features. For more information and details about the Freescale Extended Feature Packages available for this release, see the *Android™ Release Notes* (ARN).
android_L5.0.0_1.0.0-ga_full_image_6slevk.tar.gz	Prebuilt image for the i.MX 6SoloLite EVK board, which includes Freescale extended features.
android_L5.0.0_1.0.0-ga_full_image_6sxsabresd.tar.gz	Prebuilt image for the i.MX 6SoloX SABRE-SD board, which includes Freescale extended features.
android_L5.0.0_1.0.0-ga_full_image_6sxsabreauto.tar.gz	Prebuilt image for i.MX 6SoloX SABRE-AI board, which includes Freescale extended features.

The following tables list the detailed contents of android_L5.0.0_1.0.0-ga_core_image_6qsabresd.tar.gz and android_L5.0.0_1.0.0-ga_full_image_6qsabresd.tar.gz image packages.

The table below shows the prebuilt images to support the system boot from eMMC on the i.MX 6Dual/6Quad SABRE-SD board and platform and i.MX 6Solo/6DualLite SABRE-SD platform.

Table 9. Images for i.MX 6 SABRE-SD board and platform eMMC boot

SABRE-SD eMMC image	Description
u-boot-imx6q.imx	The bootloader (with padding) for i.MX 6Dual/6Quad SABRE-SD board and platform
u-boot-imx6dl.imx	The bootloader (with padding) for i.MX 6Solo/6DualLite SABRE-SD platform
eMMC/boot-imx6dl.img eMMC/boot-imx6q.img eMMC/boot-imx6q-ldo.img	Boot Image for eMMC
eMMC/system.img	System Boot Image
eMMC/recovery-imx6dl.img eMMC/recovery-imx6q.img eMMC/recovery-imx6q-ldo.img	Recovery Image

The table below shows the prebuilt images to support the system boot from SD on the i.MX 6Dual/6Quad and i.MX 6Solo/6DualLite SABRE-SD boards.

Table 10. Images for SABRE-SD SD

SABRE-SD SD image	Description
u-boot-imx6q.imx	The bootloader (with padding) for the i.MX 6Dual/Quad SABRE-SD board
u-boot-imx6dl.imx	The bootloader (with padding) for the i.MX 6DualLite SABRE-SD board
SD/boot-imx6dl.img SD/boot-imx6q.img SD/boot-imx6q-ldo.img	Boot image for SD
SD/system.img	System boot image
SD/recovery-imx6dl.img SD/recovery-imx6q.img SD/recovery-imx6q-ldo.img	Recovery image

The following tables list the detailed contents of android_L5.0.0_1.0.0-ga_core_image_6qsabreauto.tar.gz and android_L5.0.0_1.0.0-ga_full_image_6qsabreauto.tar.gz image packages:

The table below shows the prebuilt images to support the system boot from SD on the i.MX 6Dual/6Quad and i.MX 6Solo/6DualLite SABRE-AI boards.

Table 11. Images for i.MX 6 SABRE-AI SD boot

SABRE-AI SD image	Description
u-boot-imx6q.imx	The bootloader (with padding) for i.MX 6Dual/6Quad SABRE-AI SD boot
u-boot-imx6dl.imx	The bootloader (with padding) for i.MX 6DualLite SABRE-AI SD boot
SD/boot-imx6q.img SD/boot-imx6dl.img	Boot Image for SD
SD/system.img	System Boot Image
SD/recovery-imx6q.img SD/recovery-imx6dl.img	Recovery Image

The table below shows the prebuilt images to support the system boot from NAND on the i.MX 6 series SABRE-AI board.

Table 12. Images for i.MX 6 SABRE-AI NAND boot

SABRE-AI NAND image	Description
u-boot-imx6q-nand.imx	The bootloader (with padding) for i.MX 6Dual/6Quad SABRE-AI NAND boot
u-boot-imx6dl-nand.imx	The bootloader (with padding) for i.MX 6DualLite SABRE-AI NAND boot
NAND/boot-imx6q-nand.img NAND/boot-imx6dl-nand.img	Boot Image for NAND
NAND/android_root.img	System Boot Image
NAND/recovery-imx6q-nand.img NAND/recovery-imx6dl-nand.img	Recovery Image

The following tables list the detailed contents of android_L5.0.0_1.0.0-ga_core_image_6slevk.tar.gz image package.

The table below shows the prebuilt images to support the system boot from SD on the i.MX 6SoloLite EVK board.

Table 13. Images for i.MX 6SoloLite EVK SD

EVK SD image	Description
u-boot-imx6sl.imx	Bootloader (with padding) for the i.MX 6SoloLite EVK board
SD/boot-imx6sl.img	Boot image for SD
SD/system.img	System boot image
SD/recovery-imx6sl.img	Recovery image

The following tables list the detailed contents of the android_L5.0.0_1.0.0-ga_core_image_6sxsabresd.tar.gz image package.

The table below shows the prebuilt images to support the system boot from SD on the i.MX 6SoloX SABRE-SD board.

Table 14. Images for i.MX 6SoloX SABRE-SD SD

i.MX 6SoloX SABRE-SD SD image	Description
u-boot-imx6sx.imx	Bootloader (with padding) for the i.MX 6SoloX SABRE-SD board
SD/boot-imx6sx.img	Boot image for the i.MX 6SoloX SABRE-SD board
SD/system.img	System boot image
SD/recovery-imx6sx.img	Recovery image for the i.MX 6SoloX SABRE-SD board

The following tables list the detailed contents of android_L5.0.0_1.0.0-ga_core_image_6sxsabreauto.tar.gz and android_L5.0.0_1.0.0-ga_full_image_6sxsabreauto.tar.gz image packages.

The table below shows the prebuilt images to support the system boot from SD on the i.MX 6SoloX SABRE-AI boards.

Table 15. Images for i.MX 6SoloX SABRE-AI SD

i.MX 6SoloX SABRE-AI SD image	Description
u-boot-imx6sx.imx	Bootloader (with padding) for i.MX 6SoloX SABRE-AI SD boot.
SD/boot-imx6sx.img	Boot image for SD.
SD/system.img	System boot Image.
SD/recovery-imx6sx.img	Recovery image.

The table below shows the prebuilt images to support the system boot from NAND on the i.MX 6SoloX SABRE-AI boards.

Table 16. Images for i.MX 6SoloX SABRE-AI NAND

i.MX 6SoloX SABRE-AI NAND image	Description
u-boot-imx6sx-nand.imx	Bootloader (with padding) for i.MX 6SoloX NAND boot
NAND/boot-imx6sx.img	Boot image for NAND
NAND/android_root.img	System Boot image
NAND/recovery-imx6sx.img	Recovery image

NOTE

boot.img is an Android image that stores zImage and ramdisk together. It also stores other information such as the kernel boot command line, machine name, e.g., This information can be configured in corresponding board's BoardConfig.mk. If you use boot.img, you do not need uImage and uramdisk.img.

5 Programming Images

The images from the Freescale prebuilt release package or created from source code contain the U-Boot boot loader, system image, and recovery image. At a minium, the storage devices on the Freescale development system (MMC/SD or NAND) must be programmed with the U-Boot boot loader. The i.MX 6 series boot process determines what storage device to access based on the switch settings. When the boot loader is loaded and begins execution, the U-Boot environment space is then read to determine how to proceed with the boot process. For U-Boot environment settings, see Section Booting.

The following download methods can be used to write the Android System Image:

- MFGTool to download all images to MMC/SD card and NAND storage.
- Using dd command to download all images to MMC/SD card.

5.1 System on MMC/SD

The images needed to create an Android system on MMC/SD are listed below:

- U-Boot image: u-boot.imx
- boot image: boot.img
- Android system root image: system.img
- Recovery root image: recovery.img

5.1.1 Storage partitions

The layout of the MMC/SD/TF card for Android system is shown below:

- [Partition type/index] is which defined in the MBR.
- [Name] is only meaningful in the Android platform. Ignore it when creating these partitions.
- [Start Offset] shows where partition is started, unit in MB.

The SYSTEM partition is used to put the built Android system image. The DATA is used to put applications' unpacked codes/data, system configuration database, etc. In normal boot mode, the root file system is mounted from uramdisk. In recovery mode, the root file system is mounted from the RECOVERY partition.

Partition type/index	Name	Start offset	Size	File system	Content
N/A	BOOT Loader	1 KB	1 MB	N/A	bootloader
Primary 1	Boot	8 MB	8 MB	boot.img format, kernel + ramdisk	boot.img
Primary 2	Recovery	Follow Boot	8 MB	boot.img format, kernel + ramdisk	recovery.img
Logic 5 (Extended 3)	SYSTEM	Follow Recovery	512 MB	EXT4. Mount as / system	Android system files under / system/ dir.
Logic 6 (Extended 3)	CACHE	Follow SYSTEM.	512 MB	EXT4. Mount as / cache.	Android cache for image store of OTA.
Logic 7 (Extended 3)	Device	follow CACHE	8 MB	Ext4. Mount at /vender.	To Store MAC address files.
Logic 8 (Extended 3)	Misc	Follow Device	6 MB	N/A	For recovery store bootloader message, reserve.
Logic 9 (Extended 3)	DATAFOOTER	Follow Misc	2 MB	N/A	For crypto footer of DATA partition encryption.
Primary 4	DATA	Follow Misc.	Total - Other images	EXT4. Mount at / data.	Application data storage for the system application and for internal media partition in /mnt/sdcard/ dir.

There is a shell script mksdcard.sh.tar in MFGTool-Dir\Profiles\Linux\OS Firmware.

The script below can be used to partition and program an SD card as shown in the partition table above:

```
$ cd ~/myandroid/
$ sudo chmod +x ./device/fsl/common/tools/fsl-sdcard-partition.sh
$ sudo ./device/fsl/common/tools/fsl-sdcard-partition.sh -f <soc_name> /dev/sdX
# <soc_name> can be as imx6q, imx6dl, imx6sl, and imx6sx.
```

NOTE

- The minimum size of SD card is 2GB.
- /dev/sdxN, the x is the disk index from 'a' to 'z'. That may be different on each computer running Linux OS.
- Unmount all the SD card partitions before running the script.

5.1.2 Downloading images with MFGTool

MFGTool can be used to download all images into a target device. It is a quick and easy tool for downloading images. See the *Android™ Quick Start Guide* (AQSUG) for detailed description of MFGTool.

5.1.3 Downloading images with dd utility

The Linux utility "dd" on the computer running Linux OS can be used to download the images into the MMC/SD/TF card.

Before downloading, ensure that your MMC/SD/TF card partitions are created as described in Storage partitions.

All partitions can be recognized by the the computer running Linux OS. To download all images into the card, use the commands below:

```
Download the U-Boot image:
# sudo dd if=u-boot.imx of=/dev/sdx bs=1K seek=1; sync
Download the boot image:
# sudo dd if=boot.img of=/dev/sdx1; sync
Download the Android system root image:
# sudo dd if=system.img of=/dev/sdx5; sync
Download the Android recovery image:
# sudo dd if=recovery.img of=/dev/sdx2; sync
```

5.2 System on NAND for the SABRE-AI board

The images needed to create an Android system on NAND are listed below:

- U-Boot image: u-boot-imx6q-nand.imx, u-boot-imx6dl-nand.imx, u-boot-imx6solo-nand.imx
- Boot image: boot.img
- Android system root image: android_root.img
- Recovery root image: recovery.img

The images can either be obtained from the release package or they can be built from source.

5.2.1 Storage partitions on NAND

The layout of the NAND for Android system is shown below:

- [Partition type/index] is which defined in the MBR.
- [Name] is only meaningful in the Android platform. Ignore it when creating these partitions.
- [Start Offset] shows where partition is started, unit in MB.

The SYSTEM partition is used to put the built Android system image. The DATA is used to put the application's unpacked codes/data, system configuration database, etc. In normal boot mode, the root file system is mounted from uramdisk. In recovery mode, the root file system is mounted from the RECOVERY partition.

Partition type/index	Name	Start offset	Size	File system	Content
MTD0	BOOT Loader	0	64 MB	N/A	bootloader
MTD1	Boot	64 MB	16 MB	boot.img format, kernel + ramdisk	boot.img
MTD2	Recovery	Follow Boot	16 MB	boot.img format, kernel + ramdisk	recovery.img
MTD3(ubi0:system)	SYSTEM	Follow Recovery	360 MB	ubifs. Mount as / system	Android system files under / system/ dir.
MTD3(ubi0:cache)	CACHE	Follow SYSTEM.	512 MB	ubifs. Mount as / cache	Android cache for image store of OTA.
MTD3(ubi0:device)	Vendor	Follow CACHE.	10 MB	ubifs. Mount at / vender	For Store MAC address files.
MTD3(ubi0:data)	Data	follow Vendor	7 GB	ubifs Mount at /vender.	Application data storage for system application. And for internal media partition, in /mnt/sdcard/ dir.

To create storage partitions, use MFGTool as described in the *Android™ Quick Start Guide* (AQSUG).

5.2.2 Downloading images with MFGTool for NAND

MFGTool can be used to download all images into a target device.

It is a quick and easy tool for downloading images. See the *Android™ Quick Start Guide* (AQSUG) for detailed description of MFGTool for NAND.

6 Booting

This chapter describes booting from MMC/SD and NAND.

6.1 Booting from MMC/SD

This section describes how to boot from MMC/SD on the Freescale SABRE and EVK development tool devices:

- i.MX 6DualQuad SABRE-SD board/platform
- i.MX 6DualQuad/6DualLite SABRE-AI board
- i.MX 6SoloLite EVK board
- i.MX 6SoloX SABRE-SD board

6.1.1 Booting from MMC/SD on the i.MX 6DualQuad/6DualLite SABRE-SD board

The following table lists the boot switch settings for different boot methods:

Download mode (MFGTool mode)	(SW6) 00001100 (from 1-8 bit)
eMMC 4-bit (MMC2) boot	(SW6) 11100110 (from 1-8 bit)
eMMC 8-bit (MMC2) boot	(SW6) 11010110 (from 1-8 bit)
SD2 boot	(SW6) 10000010 (from 1-8 bit)
SD3 boot	(SW6) 01000010 (from 1-8 bit)

To boot from eMMC, perform the following operations:

Change the board boot switch to eMMC 4-bit mode and make (SW6) 11100110 (from 1-8 bit). Or change (SW6) 11100110 (from 1-8 bit) for 8-bit boot mode.

The default environment in boot.img is booting from eMMC. If you want to use the default environment in boot.img, use the following command:

```
U-Boot > setenv bootargs
```

To clear the bootargs environment, use the following commands:

```
U-Boot > setenv fastboot_dev mmc2 [eMMC as fastboot deivce]
U-Boot > setenv bootcmd booti mmc2      [Load the boot.img from eMMC]
U-Boot > setenv bootargs console=ttymxc0,115200 init=/init video=mxcfb0:dev=ldb,bpp=32
video=mxcfb1:off video=mxcfb2:off video=mxcfb3:off vmalloc=400M androidboot.console=ttymxc0
consoleblank=0  androidboot.hardware=freescale cma=384M [Optional]
U-Boot > saveenv        #[Save the environments]
```

NOTE

The mmcX value changes depending on the boot mode. These are the correct values:

- eMMC --> mmc2
- SD2 --> mmc3
- SD3 --> mmc1

bootargs env is an optional setting for booti. The boot.img includes a default bootargs, which is used if there is no definition of the bootargs environment.

Some SoCs on SABRE-SD boards do not have MAC address fused. Therefore, if you want to use FEC in U-Boot, set the following environment:

```
U-Boot > setenv ethaddr 00:04:9f:00:ea:d3            #[setup the
MAC address]
U-Boot > setenv fec_addr 00:04:9f:00:ea:d3           $[setup the
MAC address]
```

To boot from the SD card, perform the following operations:

Change the board boot switch to SD3 boot: (SW6) 01000010 (from 1-8 bit). To clear the bootargs env and set up the booting from SD card in SD slot 3, use the following command:

```
U-Boot > setenv fastboot_dev mmc1  [eMMC as fastboot deivce]
U-Boot > setenv bootcmd booti mmc1         [Load the boot.img from SD card]
U-Boot > setenv bootargs console=ttymxc0,115200 init=/init video=mxcfb0:dev=ldb,bpp=32
video=mxcfb1:off video=mxcfb2:off video=mxcfb3:off vmalloc=400M androidboot.console=ttymxc0
consoleblank=0 androidboot.hardware=freescale cma=384M  mtdparts=gpmi-nand:16m(bootloader),
16m(bootimg),16m(recovery),-(root) ubi.mtd=4   #[Optional]
U-Boot > saveenv    #[Save the environments]
```

NOTE

bootargs env is an optional setting for booti. The boot.img includes a default bootargs, which is used if there is no definition of the bootargs environment.

Some SoCs on SABRE-SD boards do not have MAC address fused.

Therefore, if you want to use FEC in U-Boot, set the following environment:

```
U-Boot > setenv ethaddr 00:04:9f:00:ea:d3            #[setup the
MAC address]
U-Boot > setenv fec_addr 00:04:9f:00:ea:d3           $[setup the
MAC address]
```

6.1.2 Booting from MMC/SD on the the i.MX 6DualQuad/6DualLite SABRE-AI board

The following table lists the boot switch settings for different boot methods on i.MX 6 series SABRE-AI boards.

Download mode (MFGTool mode)	(S3) 0101 (from 1-4 bit)
SD on CPU Board	(S1) 0100100000 (from 1-10 bit) (S2) 0010 (from 1-4 bit) (S3) 0010 (from 1-4 bit)

To boot from SD, perform the following operations:

Chang the board boot switch to (S3, S2, S1) 0010, 0010,0100100000 (from 1 bit).

The default environment in boot.img is booting from SD. If you want to use the default environment in boot.img, use the following command:

```
U-Boot > setenv bootargs
```

To clear the bootargs environment, use the following command:

```
U-Boot > setenv bootcmd booti mmc1      #[Load the boot.img from SD]
U-Boot > setenv bootargs console=ttymxc3,115200 init=/init video=mxcfb0:dev=ldb,bpp=32
video=mxcfb1:off video=mxcfb2:off video=mxcfb3:off vmalloc=400M androidboot.console=ttymxc3
consoleblank=0 androidboot.hardware=freescale cma=384M  #[Optional]
U-Boot > saveenv   #[Save the environments]
```

NOTE

bootargs env is an optional setting for booti. The boot.img includes a default bootargs, which is used if there is no definition of the bootargs environment.

Some SoCs on SABRE-AI boards do not have MAC address fused. Therefore, if you want to use FEC in U-Boot, set the following environment:

```
U-Boot > setenv ethaddr 00:04:9f:00:ea:d3            #[setup the
MAC address]
U-Boot > setenv fec_addr 00:04:9f:00:ea:d3           $[setup the
MAC address]
```

6.1.3 Booting from SD on the i.MX 6SoloLite EVK board

Below are the Boot Switch settings to control the boot storage:

Download mode (MFGTool mode)	SW3: 01000000(from 1-8 bit) SW4: 00101100 (from 1-8 bit) SW5: 00000000(from 1-8 bit) Boot_Mode: 10 (from 1-2 bit)
SD boot	SW3: 01000000(from 1-8 bit) SW4: 00101100 (from 1-8 bit) SW5: 00000000(from 1-8 bit) Boot_Mode: 01 (from 1-2 bit)

To boot from SD, perform the following operations:

Chang the board Boot_Mode switch to 01 (from 1-2 bit) and (SW3,4,5) 01000000 0010110000000000 (from 1-8 bit).

The default environment in boot.img is booting from SD. If you want to use default enviroment in boot.img,use the following command:

```
UBoot > setenv bootargs
```

To clear the bootargs environment, use the following command:

```
U-Boot > setenv bootcmd booti mmc1
U-Boot > setenv bootargs console=ttymxc0,115200 init=/init androidboot.console=ttymxc0
consoleblank=0 androidboot.hardware=freescale   #[Optional]
U-Boot > saveenv                        [Save the environments]
```

NOTE

bootargs env is an optional setting for booti. The boot.img file includes a default bootargs, which is used if there is no definition about the bootargs env.

Due to some SoCs on the EVK boards, do not fuse MAC address. You need to set the following environment if you want to use FEC in U-Boot:

```
U-Boot > setenv ethaddr 00:04:9f:00:ea:d3       [setup the MAC
address]
U-Boot > setenv fec_addr 00:04:9f:00:ea:d3      [setup the MAC
address]
```

6.1.4 Booting from SD on the i.MX 6SoloX SABRE-SD board

The following table lists the boot switch settings to control the boot storage:

Download mode (MFGTool mode)	SW10: 00000000 (from 1-8 bit) SW11: 00111000 (from 1-8 bit) SW12: 01000000 (from 1-8 bit) Boot_Mode: 10 (from 1-2 bit)
SD boot	SW3: 00000000 (from 1-8 bit) SW4: 00111000 (from 1-8 bit) SW5: 01000000 (from 1-8 bit) Boot_Mode: 01 (from 1-2 bit)

To boot from SD, perform the following operations:

Chang the board Boot_Mode switch to 01 (from 1-2 bit) and (SW10, 11, 12) 00000000 00111000 01000000 (from 1-8 bit).

The default environment in boot.img is booting from SD. If you want to use the default environment in boot.img, use the following command:

```
U-Boot > setenv bootargs
```

To clear the bootargs environment, use the following command:

```
U-Boot > setenv bootcmd booti mmc2
U-Boot > setenv bootargs console=ttymxc0,115200 init=/init androidboot.console=ttymxc0
consoleblank=0 androidboot.hardware=freescale cma=384M      [Optional]
U-Boot > saveenv                          [Save the environments]
```

NOTE

bootargs env is an optional setting for booti. The boot.img file includes a default bootargs, which is used if there is no definition about the bootargs env.

Due to some SoCs on the SABRE-SD boards, do not fuse MAC addres. You need to set the following environment if you want to use FEC in U-Boot:

```
U-Boot > setenv ethaddr 00:04:9f:00:ea:d3       [setup the MAC
address]
U-Boot > setenv fec_addr 00:04:9f:00:ea:d3      [setup the MAC
address]
```

6.2 Booting from NAND for the SABRE-AI board

6.2.1 Booting from NAND on the i.MX 6DualQuad/6DualLite SABRE-AI board

The following table lists the boot switch settings for NAND boot on the i.MX 6DualQuad/6DualLite SABRE-AI boards.

Download mode (MFGTool mode)	(S3): 0101 (from 1-4 bit)
NAND (Micro 29F8G08ABACA)	(S3): 0010 (from 1-4 bit) (S2): 0001 (from 1-4 bit) (S1): 0001000000 (from 1-10 bit)

Boot from NAND

Change the board boot switch to (S3, S2,S1) 0010, 0001,0001000000 (from 1 bit) on an i.MX 6 series SABRE-AI board.

The default environment in boot.img is booting from NAND. If you want to use the default environment in boot.img, use the following command:

```
U-Boot > setenv bootcmd 'nand read 0x12000000 0x4000000 0x1000000;booti 0x12000000'
#[Load the boot.img from NAND]
U-Boot > setenv bootargs console=ttymxc3,115200 init=/init video=mxcfb0:dev=ldb,bpp=32
video=mxcfb1:off video=mxcfb2:off video=mxcfb3:off vmalloc=400M androidboot.console=ttymxc3
consoleblank=0 androidboot.hardware=freescale cma=384M mtdparts=gpminand: 64m(bootloader),
16m(bootimg),16m(recovery),(root) ubi.mtd=4   [Optional]
U-Boot > saveenv [Save the environments]
```

NOTE

bootargs env is an optional setting for booti. The boot.img includes a default bootargs, which will be used if there is no definition of the bootargs environment.

Some SoCs on SABRE-AI boards do not have MAC address fused.

Therefore, if you want to use FEC in U-Boot, set the following environment:

```
U-Boot > setenv ethaddr 00:04:9f:00:ea:d3            [setup the
MAC address]
U-Boot > setenv fec_addr 00:04:9f:00:ea:d3           [setup the
MAC address]
```

6.2.2 Booting from NAND on the i.MX 6SoloX SABRE-AI board

The following table lists the boot switch settings for NAND boot on the i.MX 6SoloX SABRE-AI boards.

Download mode (MFGTool mode)	S1 (Boot_Mode): 0101 (from 1-4 bit)
NAND (Micro MT29F64G08AFAAA)	S1 (Boot_Mode): 0010 (from 1-4 bit) SW3: 00000000 (from 1-8bit) SW4: 00000001 (from 1-8bit)

Boot from NAND

Change the board boot switch to (S1, S3,S4) 0010, 00000000,00000001 (from 1bit) on an i.MX 6SoloX SABRE-AI board.

The default environment in boot.img is booting from NAND. If you want to use the default environment in boot.img, use the following command:

```
U-Boot > setenv bootcmd 'nand read 0x80800000 0x4000000 0x1000000;booti0x12000000' #[Load
the boot.img from NAND]
U-Boot > setenv bootargs console=ttymxc0,115200 init=/init video=mxcfb0:dev=ldb,bpp=32
video=mxcfb1:off video=mxcfb2:off video=mxcfb3:off vmalloc=400M androidboot.console=ttymxc0
consoleblank=0 androidboot.hardware=freescale cma=384M mtdparts=gpminand: 64m(bootloader),
16m(bootimg),16m(recovery),(root)
ubi.mtd=4 [Optional]
U-Boot > saveenv [Save the environments]
```

NOTE

bootargs env is an optional setting for booti. The boot.img includes a default bootargs, which will be used if there is no definition of the bootargs environment.

Some SoCs on SABRE-AI boards do not have MAC address fused.

Therefore, if you want to use FEC in U-Boot, set the following environment:

```
U-Boot > setenv ethaddr 00:04:9f:00:ea:d3   [setup the MAC
address]
U-Boot > setenv fec_addr 00:04:9f:00:ea:d3  [setup the MAC
address]
```

6.3 Boot-up configurations

This section explains the common U-Boot environments used for MMC/SD boot, and kernel command line.

6.3.1 U-Boot environment

- ethaddr/fec_addr is a MAC address of the board.
- serverip is an IP address of the TFTP/NFS server.
- loadaddr/rd_loadaddr is the kernel/initramfs image load address in memory.
- bootfile is the name of the image file loaded by "dhcp" command, when you are using TFTP to load kernel.

- bootcmd is the first variable to run after U-Boot boot.
- bootargs is the kernel command line, which the bootloader passes to the kernel. As described in Kernel command line (bootargs), bootargs env is optional for booti. boot.img already has bootargs. If you do not define the bootargs env, it will use the default bootargs inside the image. If you have the env, it will be used.

  If you want to use default env in boot.img, use the following command to clear the bootargs env.

  ```
  > setenv bootargs
  ```
- dhcp: get ip address by BOOTP protocol, and load the kernel image ($bootfile env) from TFTP server.
- booti:

  booti command will parse the boot.img header to get the zImage and ramdisk. It will also pass the bootargs as needed (it will only pass bootargs in boot.img when it can't find "bootargs" var in your U-Boot env). To boot from mmcX, do the following:

  ```
  > booti mmcX
  ```

  To read the boot partition (the partition store boot.img, in this instance, mmcblk0p1), the X was the MMC bus number, which is the hardware MMC bus number, in i.MX 6Dual/6Quad SABRE-SD and i.MX 6Solo/6DualLite SABRE-SD boards. eMMC is mmc2.

  You can also add partition ID after mmcX.

  ```
  > booti mmcX boot   #  boot is default
  > booti mmcX recovery      # boot from the recovery partition
  ```

  If you have read the boot.img into memory, use this command to boot from

  ```
  > booti 0xXXXXXXXX
  ```
- bootm (only works in for the NFS) starts running the kernel. For other use cases, use booti command.
- splashimage is the virtual or physical address of bmp file in memory. If MMU is enabled in board configuration file, the address is virtual. Otherwise, it is physical. See README in U-Boot code root tree for details.
- splashpos sets the splash image to a free position, 'x,y', on the screen. x and y should be a positive number, which is used as number of pixel from the left/top. Note that the left and top should not make the image exceed the screen size. Specify 'm,m' for centering the image. Usually, for example, '10,20', '20,m', 'm,m' are all valid settings. See README in U-Boot code root tree for details.
- lvds_num chooses which LVDS interface, 0 or 1, is used to show the splash image. Note that we only support boot splash on LVDS panel. We do not support HDMI or any other display device.

6.3.2 Kernel command line (bootargs)

Depending on the different booting/usage scenarios, you may need different kernel boot parameters set for bootargs.

Kernel parameter	Description	Typical value	Used when
console	Where to output kernel log by printk.	console=ttymxc0,115200	All use cases.
init	Tells kernel where the init file is located.	init=/init	All use cases. "init" in the Android platform is located in "/" instead of in "/sbin".
ip	Tells kernel how/ whether to get an IP address.	ip=none or ip=dhcp	"ip=dhcp" or "ip=static_ip_address" is mandatory in "boot from TFTP/NFS".

Table continues on the next page...

Kernel parameter	Description	Typical value	Used when
		or ip=static_ip_address	
nfsroot	Where the NFS server/directory is located.	rootfs=ip_address:/opt/nfsroot,v3,tcp	Used in "boot from tftp/NFS" together with "root=/dev/nfs".
root	Indicates the location of the root file system.	root=/dev/nfs or root=/dev/mmcblk0p2	Used in "boot from tftp/NFS" (i.e., root=/dev/nfs). Used in "boot from SD" (i.e., root=/dev/mmcblk0p2) if no ramdisk is used for root fs.
video	Tells kernel/driver which resolution/depth and refresh rate should be used, or tells kernel/driver not to register a framebuffer device for a display device.	video=mxcfb0:dev=ldb,LDB-XGA,if=RGB666,bpp=32 or video=mxcfb1:dev=hdmi,1920x1080M@60,if=RGB24,bpp=32 or video=mxcfb2:off	To specify a display framebuffer with: video=mxcfb<0,1,2>:dev=<ldb,hdmi>,<LDB-XGA,xres x yresM@fps>,if=<RGB666,RGB24>,bpp=<16,32> or To disable a display device's framebuffer register with: video=mxcfb<0,1,2>:off
vmalloc	vmalloc virtual range size for kernel.	vmalloc=400M	vmalloc=<size>
androidboot.console	The Android shell console. It should be the same as console=.	androidboot.console=ttymxc0	If you want to use the default shell job control, such as Ctrl+C to terminate a running process, you must set this for the kernel.
fec_mac	Sets up the FEC MAC address.	fec_mac=00:04:9f:00:ea:d3	On SABRE-SD board, the SoC does not have MAC address fused in. If you want to use FEC, assign this parameter to the kernel.
cma	CMA memory size for GPU/VPU physical memory allocation.	cma=384M	It is 256 MB by default.
androidboot.selinux	Disables the SELINUX feature.	androidboot.selinux=disabled	Only for debugging purpose.
androidboot.dm_verity	disable the DM_VERITY feature	androidboot.dm_verity=disabled	Only for debugging purpose.

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number: AUG
Rev. L5.0.0_1.0.0-ga
06/2015