

i.MX Android

User Guide

R14 Beta
June 11, 2012

This document is intended to show how to use the release package. It describes how to setup the build environment, patch the source code, and build total android system. Additionally, this document shows how load and run the Android OS.

Index

[i.MX Android](#)

[1. Preparation](#)

[1.1 PC Setup](#)

[1.2 Unpack i.MX Android Release Package](#)

[1.3 Run Android with Prebuilt Image](#)

[2. Build Android for i.MX](#)

[2.1 Get Android Source Code \(Android/Kernel/uboot\)](#)

[2.2 Patch Code for i.MX](#)

[2.3 Build Android Image](#)

[2.3.1 User Build mode](#)

[2.4 Build Uboot Images](#)

[2.5 Build Kernel Image](#)

[3. Download Images](#)

[3.1 System on MMC/SD](#)

[3.1.1 Create Ramdisk](#)

[3.1.2 Storage Partitions](#)

[3.1.3 Download Images with dd utility](#)

[3.2 System on NFS](#)

[3.2.1 Setup the NFS root](#)

[4. Boot Configurations](#)

[4.1 Boot from eMMC](#)

[4.2 Boot from External SD Card](#)

[MX53 SMD SD \(slot1\) Boot](#)

[4.3 Boot from NFS](#)

[4.4 U-boot Environment](#)

[4.5 Kernel command line \(bootargs\)](#)

1. Preparation

1.1 PC Setup

To build the Android source files, you will need to use Linux PC. It's recommended that you use the 10.10 or 11.04 64-bit version of Ubuntu. It is also recommended to use a quad core or better, to decrease your build times.

To setup your Linux PC, you need check whether you have all necessary packages installed for building Android. Refer to "Setting up your machine" on the Android web site <http://source.android.com/source/initializing.html>.

In addition, there are a few other packages that must be installed on your Linux PC as well. If you are using Ubuntu, run the following command:

```
$ sudo apt-get install gawk sed u-boot-tools
```

1.2 Unpack i.MX Android Release Package

After you setup a Linux PC, unpack the FSL i.MX Android Release Package using the following commands:

```
$ cd /opt (or any other directory you like)
$ tar xzvf imx-android-rl4-beta.tar.gz
$ cd imx-android-rl4-beta/code
$ tar xzvf rl4-beta.tar.gz
```

1.3 Run Android with Prebuilt Image

To try Android before building any code, use the prebuilt images in image directory and go to "Download Images".

For MX53 SMD platform:

- **Uboot image**
 - imx53_smd/u-boot-no-padding.bin
 - imx53_smd/u-boot.bin (with padding)
- **Kernel image**
 - imx53_smd/uImage
- **Images used for eMMC and SD**

- Ramdisk: imx53_smd/uramdisk.img
- Recovery: imx53_smd/recovery.img
- System root: imx53_smd/system.img
- **Images for NFS**
imx53_smd/android_fs.tar.gz

2. Build Android for i.MX

There several steps required to build Android for the i.MX53 SMD. The next sections describe how to

- Get the source code
- Apply the iMX patches
- Build Android
- Build U-Boot
- Build the Kernel

2.1 Get Android Source Code (Android/Kernel/uboot)

The Android source code is consists of more than 100 individual git repositories from the Android repository at android.google.com. To get the Android source code from the Google repository, follow these steps:

```
Assumes you had unzipped i.MX Android release package to /opt/imx-android-
r14-beta/.

$ cd ~
$ mkdir myandroid
$ cd myandroid
$ curl https://dl-ssl.google.com/dl/googlesource/git-repo/repo > ./repo
$ chmod a+x ./repo

$ ./repo init -u https://android.google.com/platform/manifest -b
android-4.0.4_r1.1

$ cp /opt/imx-android-r14-beta/code/r14-beta/default.xml .repo/manifests/
default.xml
(To avoid downloading unnecessary gits from Google repo and download some
gits from Google repo which are not included in default manifest)
$ ./repo sync

After synchronizing the code, do the following:
$ cd myandroid/external
$ git clone git://android.git.linaro.org/platform/external/alsa-lib.git

$ cd myandroid/external
$ git clone git://android.git.linaro.org/platform/external/alsa-utils.git
```

```
$ cd myandroid/hardware
$ git clone git://android.git.linaro.org/platform/hardware/alsa_sound.git
```

Get R14 Beta kernel source code from freescale's opensource git:

```
$ cd myandroid
$ git clone git://git.freescale.com/imx/linux-2.6-imx.git kernel_imx
$ cd kernel_imx
$ git checkout imx-android-r14-beta
```

NOTE: If you're behind proxy, use socksify to set socks proxy for git protocol. See the i.MX Android FAQ.

Next, Checkout U-Boot source code from the main repository:

```
$ cd myandroid/bootable/bootloader
$ git clone git://git.denx.de/u-boot.git uboot-imx
```

2.2 Patch Code for i.MX

Apply all i.MX Android patches using following steps:

```
Assume you had unzipped i.MX Android release package to /opt/imx-android-r14-beta/.
```

```
$ cd ~/myandroid
$ . /opt/imx-android-r14-beta/code/r14-beta/and_patch.sh
$ help
```

```
Now you should see "c_patch" function is available for you
$ c_patch /opt/imx-android-r14-beta/code/r14-beta imx_r14-beta
```

```
Here "/opt/imx-android-r14-beta/code/r14-beta" is the location of the
patches (i.e. directory created when you unzip release package)
"imx_r14-beta" is the branch which will be created automatically for you to
hold all patches (only in those existing Google gits).
You can choose any branch name you like instead of "imx_r14-beta".
If everything is OK, "c_patch" will generate the following output to
indicate successful patch:
```

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

2.3 Build Android Image

After applying all i.MX patches, build the U-Boot, kernel and Android image using following

steps:

```
$ cd ~/myandroid
$ source build/envsetup.sh
```

For MX53 SMD platform:

```
$ lunch imx53_smd-user
```

```
$ make
```

To build faster on a multi-core Linux PC use the `-j` switch. For example, to build with 4 make jobs:

```
$ make -j4
```

For MX53 SMD build, the following outputs are generated as default under `myandroid/out/target/product/imx53_smd`:

- **root/** : root file system (including init, init.rc, etc). 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.
- **boot.img**: a composite image which includes the kernel zImage, ramdisk, and boot paramters.
- **ramdisk.img**: Ramdisk image generated from "root/".
- **system.img**: EXT4 image generated from "system/".
- **userdata.img**: EXT4 image generated from "data/".
- **recovery.img**: EXT4 image generated from "recovery/".
- **u-boot.bin**: uboot image with padding.
- **uImage**: kernel image with uboot header.

Note:

- Make sure the `mkimage` is a valid commands in your build machine. If not, please follow the below commands to have it installed:

```
$sudo apt-get install uboot-mkimage
```

- To build the U-Boot image separately, please refer the section 2.4
- To build the kernel uImage separately, please refer the section 2.5

2.3.1 User Build mode

For a production release, the android image should be built in user mode. The user mode build has the following differences from the eng mode build:

- It will have limited access to increase security
- Many debug utilities are not included.
- It will install modules tagged with user, and those application packages (APKs) and tools according to product definition. The product definition **PRODUCT_PACKAGES** is defined in **device/fsl/imx5x/imx5x.mk**
- Properties are set as follows `ro.secure=1` and `ro.debuggable=0`.
- Android Debug Bridge (ADB) is disabled by default.

If you need to add your customized package, add the package's `MODULES_NAME` or `PACKAGE_NAME` to the **PRODUCT_PACKAGES** list the in **device/fsl/imx5x/imx5x.mk**.

To build user mode images:
eg:

```
$ . build/envsetup
$ lunch imx53_smd-user
$ make
```

For More android building information please refer to <http://source.android.com/source/building.html>

2.4 Build Uboot Images

To build U-Boot run the following commands:

```
$ cd ~/myandroid/bootable/bootloader/uboot-imx
$ export ARCH=arm
$ export CROSS_COMPILE=~myandroid/prebuilt/linux-x86/toolchain/arm-eabi-4.4.3/bin/arm-eabi-
$ make distclean
$ make mx53_smd_android_ics_config
$ make
```

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

The above u-boot.bin has 1024KB padding at the head of file, for example first executable instruction is at the offset 1KB. If you want to generate a no-padding image, you need do below dd command in host.

```
$ sudo dd if=./u-boot.bin of=./u-boot-no-padding.bin bs=1024 skip=1; sync
```

Usually this no-padding uboot image is used in the SD card, for example, program this no-padding uboot image into 1KB offset of SD card so that we do not overwrite the MBR (including partition table) within first 512B on the SD card.

2.5 Build Kernel Image

Kernel image will be built when building the android root file system. So you can jump to next section if you do not care how to build kernel image. To build the kernel with the default configuration run the following commands:

```
$ cd ~/myandroid/kernel_imx
$ export ARCH=arm
```

```
$ export CROSS_COMPILE=~/.myandroid/prebuilt/linux-x86/toolchain/arm-eabi-4.4.3/bin/arm-eabi-
$ make imx5_android_defconfig

Generate ".
config" according to default config file under arch/arm/configs.
$ make -j4 uImage
```

After the kernel build completes, the kernel image is located in `~/myandroid/kernel_imx/arch/arm/boot/uImage`.

3. Download Images

The MX53 SMD Android can be configured to boot from three different sources:

1. Boot from internal eMMC
2. Boot from external SD card
3. Boot from NFS (networking)

Before boot, you should program the bootloader, kernel, ramdisk, and rootfs images into the main storage device (internal eMMC or external SD card) or unpack the NFS root filesystem into the NFS server root.

For MX53 SMD platform, following download methods are supported:

- MFGTool to download all images to eMMC card
- Using dd command to download all images to SD card

3.1 System on MMC/SD

Below images are needed to create an android system on MMC/SD/TF:

- u-boot: u-boot.bin or u-boot-no-padding.bin
- kernel: uImage
- ramdisk: uramdisk.img
- Android system root image: system.img
- Recovery root image: recovery.img

You can use the images from the release package or build the images yourself.

3.1.1 Create Ramdisk

The ramdisk generated by the build must be converted to a format that is known to U-boot.

```
# change directory to the build output directory.
$ cd ~/.myandroid/out/target/product/imx53_smd
```



```
# make uramdisk.img
$ mkimage -A arm -O linux -T ramdisk -C none -a 0x70408000 -n "Android Root
Filesystem" -d ./ramdisk.img ./uramdisk.img
```

3.1.2 Storage Partitions

The layout of the MMC/SD/TF card for Android system is showed in below.

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

The SYSTEM partition is used to put the 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	0	10MB	N/A	bootloader/kernel/ uramdisk images
Primary 1	MEDIA	10MB	User Defined	VFAT. Mount as /sdcards	Media file storage
Primary 2	SYSTEM	follow MEDIA	>= 200MB	EXT4. Mount as /system (with read only)	Android system bin/ libs (system.img)
Logic 5 (Extended 3)	DATA	follow SYSTEM	> 200MB	EXT4. Mount as /data	Android data (e.g. installed app)
Logic 6 (Extended 3)	CACHE	follow DATA	> 10MB	EXT4. Mount as /cache	Android cache
Primary 4	RECOVERY	follow CACHE	> 20MB	EXT4. Mount as / in recovery	Root file system for recovery mode (recovery.img)

				mode	
--	--	--	--	------	--

To create these partitions in internal eMMC, use MFG tool described in the "Quick Start Guide".

To create these partitions on an external SD card use fdisk utility on Linux PC.

After creating the partitions by fdisk, format each file systems by the following commands:

```
$ mkfs.vfat /dev/sdx1
$ mkfs.ext4 /dev/sdx2 -O ^extent -L system
$ mkfs.ext4 /dev/sdx4 -O ^extent -L recovery
$ mkfs.ext4 /dev/sdx5 -O ^extent -L data
$ mkfs.ext4 /dev/sdx6 -O ^extent -L cache
```

Note: /dev/sdxN, the x is the disk index from 'a' to 'z', that may be different on each Linux PC.

3.1.3 Download Images with dd utility

The linux utility "dd" on Linux PC can be used to download the images into the SD card. Before downloading, make sure your SD card's partitions are created as 3.1.1 described. And all the partitions can be recognized by Linux PC. To download all the images into the card, please use the commands below:

```
# Change directory to the build output directory.
$ cd ~/myandroid/out/target/product/imx53_smd

# Optionally erase the U-Boot environment. This is required when
# changing the Android release that is programed in SD/eMMC.
$ sudo dd if=/dev/zero of=/dev/sdx bs=1024 seek=1 count=1023; sync

Download the uboot image:
$ sudo dd if=u-boot.bin of=/dev/sdx bs=1K skip=1 seek=1; sync
Or If you're using no padding uboot image:
$ sudo dd if=u-boot-no-padding.bin of=/dev/sdx bs=1K seek=1; sync
Download the kernel image:
$ sudo dd if=uImage of=/dev/sdx bs=1M seek=1; sync

Download the initramfs image:
# sudo dd if=uramdisk.img of=/dev/sdx bs=1M seek=6; sync

Download the android system root image:
# sudo dd if=system.img of=/dev/sdx2; sync

Download the android recovery image:
```

```
# sudo dd if=recovery.img of=/dev/sdx4; sync
```

3.2 System on NFS

To run the MX53 SMD system from an NFS root filesystem, you must have a PC which has a NFS Server and the NFS root directory is properly exported, e.g. /opt/nfsroot for NFS root. Setting up the NFS server is not covered in this document.

3.2.1 Setup the NFS root

After you setup the NFS server, please put the kernel image and the Android root file system (decompress and untar android_fs.tar.gz) into a directory that is exported by the NFS server.

For kernel image, use uImage

- If you are using a prebuilt image, make sure you pick up the correct uImage (see "Prebuilt image for using uboot")
- If you are building your own image, make sure you generated a uImage (see "Generate uImage to be loaded by u-boot").

Copy uImage to the TFTP server root directory. For example:

```
$ cp your_uImage /opt/nfsroot/
```

Setup the Android file system: (take SabreSD as example)

- If you are using a prebuilt image, unzip the android zip file (see "Prebuilt image for using uboot") to the NFS server root. For example:

```
$ cd /opt/imx-android-r14-beta/image/imx53_smd/NFS
$ tar xzvf ./android_fs.tar.gz
$ rm -rf /opt/nfsroot/android_fs
$ cp -r android_fs /opt/nfsroot
```

- If you built out your own Android image, copy the generated Android files to the NFS root manually. For example:

```
$ cd myandroid
$ rm -rf /opt/nfsroot/android_fs
$ cp -r out/target/product/imx53_smd/root/* /opt/nfsroot/android_fs
$ cp -r out/target/product/imx53_smd/system/* /opt/nfsroot/android_fs/
system/
```

NOTE: Since the NFS uses system and cache folder under /opt/nfsroot/, comment out the mount system and cache lines in /opt/nfsroot/init.rc.

4. Boot Configurations

4.1 Boot from eMMC

For MX53 SMD, the default U-Boot environment is setup to boot from eMMC. So after a fresh install of the u-Boot. If the boot select switches (SW26 and SW28) are set correctly, the board will boot without making changing to the U-Boot environment.

MX53 SMD eMMC(SD slot3) Boot

Switch	D1	D2	D3	D4	D5	D6	D7	D8
SW26	OFF	ON	OFF	OFF	ON	ON	ON	OFF
SW28	OFF	OFF	OFF	OFF	OFF	OFF	OFF	ON

To setup the U-Boot environment to boot from eMMC run the following commands:

```
U-Boot > setenv bootcmd 'run bootcmd_eMMC'
U-Boot > saveenv [Save the environments]
```

On the first boot, it will take longer for the Android UI to come up.

4.2 Boot from External SD Card

Because the default U-Boot environment is setup to boot from eMMC, the U-Boot environment as well as the boot switches must be configured as shown below to boot from and external SD card.

MX53 SMD SD (slot1) Boot

Switch	D1	D2	D3	D4	D5	D6	D7	D8
SW26	OFF	OFF	OFF	OFF	OFF	OFF	ON	OFF
SW28	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF

To setup the U-Boot environment to boot from the external SD slot run the following commands:

```
U-Boot > setenv bootcmd 'run bootcmd_SD'
U-Boot > saveenv [Save the environments]
```

On the first boot, it will take longer for the Android UI to come up.

4.3 Boot from NFS

To setup the u-boot environment for loading kernel and root files system from NFS, U-boot must be installed as in either of the previous sections "Boot form eMMC" or "Boot from SD". Additionally, the MX53 SMD must have an Ethernet network connection. The network connect must have access to the NFS server and a DHCP server.

```
U-Boot > setenv nfskernel /opt/nfsroot/uImage
U-Boot > setenv nfsroot /opt/nfsroot/android_fs [Your rootfs]
U-Boot > setenv serverip 192.168.1.10 [Your TFTP/NFS server ip]
U-Boot > setenv bootcmd 'run bootcmd_NFS' [load kernel from TFTP and boot]
U-Boot > saveenv [Save the environment]
```

After done these settings, reboot the board, let u-boot to run the bootcmd environment to load kernel and run.

On the first boot, it will take longer for the Android UI to come up.

4.4 U-boot Environment

The following list explains some of the common u-boot environment variables.

- ethaddr/fec_addr: Ethernet MAC address of your board. These should normally be read from IIM and not be set.
- serverip: IP address of your NFS server.
- loadaddr/rd_loadaddr: the kernel/initramfs image load addresses in memory.
- nfskernel: the name of image file loaded by "nfs" command.
- bootcmd: the first variable to run after uboot boot.
- bootcmd_SD: default boot command to boot form external SD card.
- bootcmd_eMMC: default boot command to boot from eMMC.
- bootcmd_NFS: default boot command to boot from NFS.
- bootargs_base: default variable used to set the common kernel argument by the boot command.
- bootargs: the kernel command line passed form U-Boot to the kernel. Described in section 4.5.

- dhcp: get ip address by BOOTP protocol.
- bootm: (only work in NFS case) start to run the kernel, for other case, we use booti command.

4.5 Kernel command line (bootargs)

Depend on different booting/usage scenarios, you may need different kernel boot parameters set for bootargs or bootargs_base.

Kernel Parameter	Description	Typical Value	Used When
console	where to output kernel log by printk and console baud rate	console=ttymxc0,115200	All case
init	tell kernel where is the init file	init=/init	All case for Android. "init" in Android is located in "/" instead of in "/sbin"
ip	tell kernel how/whether to get IP address	ip=none or ip=dhcp or ip=static_ip_address	"ip=dhcp" or "ip=static_ip_address" is mandatory in "boot from TFTP/NFS"
nfsroot	where is NFS server/directory	rootfs=ip_address:/opt/nfsroot,v3,tcp	Used in "boot from tftp/NFS" together with "root=/dev/nfs"
root	indicate where is 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	tell kernel/driver which resolution/depth and refresh rate should be used	video=mxcfb0:dev=ldb,LDB-XGA,if=RGB666,bpp=32 video=mxcfb1:dev=si902x_hdmi,1280x720M@60,bpp=32	LVDS connected to DIO HDMI connected to 720p Monitor/TV

pmem	pmem size configure	pmem=128M,64M	pmem=<gpu pmem size>,<vpu pmem size>
fbmem	framebuffer reservation size configure	fbmem=12M,12M	fbmem=<fb0 size>,<fb1 size>
vmalloc	vmalloc virtual range size for kernel	vmalloc=576M	vmalloc=<size>
androidboot.console	The android shell console, should be same as console=	androidboot.console=ttymxc0,115200	If you want to use the default shell's job control, like Ctrl-C to terminate a running process, you must set set this for kernel.
fec_mac	Setup the FEC mac address	fec_mac=00:04:9f:00:ea:d3	On i.MX53 SMD board, the SoC has a MAC address fused in. If for any reason, one does not exist in IIM or this needs to be changed or overridden, this argument can be used.