Document Number:FAQs
Rev jb4.2.2_1.0.0-GA, 05/2013

Android Frequently Asked Questions

1 How do I configure the build information?

For every build, you should define a BUILD ID and BUILD NUMBER. We use our release version as BUILD ID and build user date as BUILD NUMBER.

You can create a buildspec.mk file under your ~/myandroid directory. Android will read this file to get the build information, such as BUILD_ID, BUILD_NUMBER, and default build product, etc.

1. Copy the default buildspec.mk

   ```
   $ cd ~/myandroid
   $  cp build/buildspec.mk.default
   buildspec.mk
   ```
2. Change the buidspec.mk, to add BUILD_ID & BUILD_NUMBER. You can add the following lines in buildspec.mk:

   ```
   BUILD_NUMBER := $(USER).$(shell date +%Y%m
   %d.%H%M)
   BUILD_ID :=RXX.XX
   ```

   The BUILD_ID is your software version id. You can change this to your current release number.
3. (Optional) You can also change the TARGET_PRODUCT & TARGET_BUILD_VARIANT in this file to change the default build product.

For example:

```
TARGET_PRODUCT:=sabresd_6dq
TARGET_BUILD_VARIANT:=user
```

4. After these changes, you can directly call `make` to build your Android.

2 How do I download the Android source code behind a firewall?

If you have an HTTPS proxy and your firewall supports socks, perform the following steps.

```
* Install Dante - a socks client
$ sudo apt-get install dante-client
* Configure Dante by adding below lines into /etc/dante.conf
    route {
        from: 0.0.0.0/0 to: .  via: DNS_OR_IP_OF_YOUR_SOCKS_SERVER port =
PORT_OF_YOUR_SOCKS_SERVER
        proxyprotocol: socks_v5
              }       resolveprotocol: fake
* Set environment variable for http proxy and socks
$ export https_proxy=...
$ export SOCKS_USER=...
$ export SOCKS_PASSWD=...
* Download Android code from google
$ curl https://dl-ssl.google.com/dl/googlesource/git-repo/repo > ~/bin/repo
$ chmod a+x ~/bin/repo
$ repo init -u https://android.googlesource.com/platform/manifest -b android-4.2.2_r1
$ cp /opt/jb4.2.2_1.0.0-ga/code/jb4.2.2_1.0.0-ga/default.xml .repo/manifests/default.xml
$ socksify ~/bin/repo sync
```

3 How do I use ADB over Ethernet?

From JB4.2.2, security ADB is enabled default (ro.adb.security is set to 1). In security ADB mode, the ADB over ethernet is not allowed. For more details please see How do I enable and disable security ADB?. To use the ADB over ethernet, there are two steps to follow:

1. Disable the security ADB by changing ADB security setting in init.rc #Delete or comment below line, and then rebuilt the boot.img. setprop ro.adb.security 1.
2. Keep the board connecting with usb to the PC, and enable the ‘usb debuging’ from Setting->Developer Options and then follow the steps below to set up ADB over ethernet.

On the Linux PC, assuming you had built Android code or had installed Android SDK), complete the following actions to use ADB over ethernet:

```
 $ ping IP_OF_YOUR_BOARD (run "netcfg" on board to get IP address)
 $ export ADBHOST=IP_OF_YOUR_BOARD
 "adb" is a host tool created during Android build.It's under out/host/linux-x86/bin/. Make
sure you set path properly.
 $ adb kill-server (Not sure why this step is needed. Just re-start adb daemon on board.)
 $ adb shell
```

Please set the ADB port properly on the device:

```
$ setprop service.adb.tcp.port 5555
```

After setting up the ADB listener port, please re-enable the USB debug function in the Settings application.

3.1 How do I set up a computer to support ADB?

In this release, Google vendor ID and product ID are used for all Android gadget functions.

The user can download the latest Android SDK package and use ADB tool to test ADB function.

On the Windows computer, install Google extra win USB driver, contained in the SDK package, when Windows finds your device.

On the Linux computer, add the following rules for udev rule file: /etc/udev/rules.d/51-android.rules

```
SUBSYSTEM=="usb", SYSFS{idVendor}=="18d1", MODE="0777"
SUBSYSTEM=="usb|usb_device", ATTR{idVendor}=="18d1", MODE="0666", GROUP="plugdev"
```

3.2 How do I enable USB tethering?

The USB tethering feature is supported in this release.

The upstream device can be WIFI or Ethernet. USB tethering can be enabled in the Settings UI after your OTG USB cable is connected to computer: Settings -> WIRELESS & NETWORKS -> More.. -> Tethering & portable hotspot -> USB tethering. In the meantime, make sure you have disabled ADB.

On the Linux computer, when USB tethering is enabled, you can easily get an USB network device. The IP and DNS server is automatically configured.

On Windows computer, when you have connected the board with the computer and you can see an unknown device named "Android" in the device manager, you have to manually install the tethering driver by the tetherxp.inf file in android_jb4.2.2_1.0.0-ga_tools.tar.gz. After it is successfully installed, you will see "Android USB RNDIS device" in the device manager. By this time, you can use USB rndis network device to access the network.

4 How do I use MTP?

The Media Transfer Protocol is a set of custom extensions to the Picture Transfer Protocol (PTP).

Whereas PTP was designed for downloading photographs from digital cameras, Media Transfer Protocol supports the transfer of music files on digital audio players and media files on portable media players, as well as personal information on personal digital assistants.

Starting with version 4.0, Android supports MTP as a default protocol transfer files with computer, instead of the USB Mass Storage. In this release, as suggested by Google, we disabled the UMS and enabled MTP.

NOTE

Ensure that you disable the USB Tethering when using MTP. In Windows XP, you cannot make MTP work with ADB enabled. In Windows 7, in theory MTP can work together with ADB, but it is also found that some hosts with Windows 7 fail to support it.

When connecting the board to the computer by USB cable, an USB icon will be shown in the notification bar. Then, you can click on the notification area, and select "Connected as a media device" to launch the USB computer connection option UI. There, MTP and PTP can be chosen as current transfer protocol. You can also launch the option UI by Settings -> Storage -> MENU -> USB computer connection.

MTP on Windows XP

Windows XP supports PTP protocol by default. In order to support MTP protocol, you must install Windows Media Player (Version >= 10). When connecting to a computer, you can see MTP devices in Windows Explorer. Since Windows XP only supports copy/paste files in the explorer, you cannot directly open the files in MTP device.

MTP on Windows 7

Windows 7 supports MTP (PTP) protocol by default. When connecting to a computer, you can see MTP devices in Windows Explorer. You can perform any operations just as you would on your hard disk.

MTP on Ubuntu

Ubuntu supports PTP protocol by default. To support MTP protocol, you must install the following packages: libmtp, mtptools by using the following command:

```
$ sudo apt-get install mtp-tools
```

If your default libmtp version is not 1.1.1 (current latest libmtp on ubuntu is 1.1.0), you must upgrade it manually by:

```
$ sudo apt-get install libusb-dev
$ wget http://downloads.sourceforge.net/project/libmtp/libmtp/1.1.1/libmtp-1.1.1.tar.gz
$ tar -xvf libmtp-1.1.1.tar.gz
$ cd libmtp-1.1.1
$ ./configure --prefix=/usr
$ make
$ sudo make install
```

After you have done the steps outlined above, you can transfer the files between the computer and the device by using the following commands:

- mtp-detect finds current connected MTP device.
- mtp-files lists all the files on MTP device.
- mtp-getfile gets the files on MTP device by file ID listed by mtp-files.
- mtp-sendfile puts files onto MTP device.

There's an alternative GUI application, called gMtp, which makes it easier to access MTP device instead of using the commands above. You can install it by using the following command:

```
$ sudo apt-get install gmtp
```

After installation, you can launch gmtp and access MTP device in the file explorer.

5 How do I enter the Android recovery mode manually?

This process only works on i.MX 6Quad SD v1 board.

Press "**VOLUME** -" and "**Power**" to enter Recovery mode. This check is in u-boot.git board support file, where you can change your preferred combo keys.

Also, you can input this command in the kernel:

```
# reboot recovery            # the board reset to recovery mode.
```

to enter recovery mode.

5.1 How do I use Android fastboot?

Fastboot is a feature which can be used to download images from Windows/Linux computer to the target storage device.

This feature is released by Google in the Android SDK package, which can be downloaded from Android official site. Freescale Android release implements part of the fastboot commands in U-Boot such as: flash, reboot, getvar.

Before using fastboot, Android SDK must be installed on the host, and the target board must boot up to bootloader. Also, before using fastboot, U-Boot must be downloaded to the MMC/SD device with all the partitions created and formatted. Setup the correct board dip switches to boot up the board with U-Boot.

NOTE

The size of images downloaded by fastboot must be < 320MB.

Target side:

- Power on the board with USB otg connected.
- Press any key to enter the U-Boot shell.
- Select the correct device to do fastboot image download by command:
  - SD/MMC card
  - U-Boot > setenv fastboot_dev mmc3
  - Run the fastboot command:

```
U-Boot > fastboot
fastboot is in init......USB Mini b cable Connected!
fastboot initialized
USB_SUSPEND
USB_RESET
USB_RESET
```

Or launch the quick fastboot by "fastboot q" command

```
U-Boot > fastboot q
fastboot is in init......flash target is MMC:1
USB Mini b cable Connected!
```

Or you can input this command in the kernel:

```
# reboot fastboot              # the board reset to fastboot mode.
```

All commands can enter fastboot mode.

Note:

1. On host computer, it will prompt you that a new device was found and that you need to install the driver. Please install it.

2. The quick fastboot is a new implementation for fastboot utility. It increases the image download speed from computer to board up to about 28MB/s, compared to the previous 1MB/s. It only supports download and flash command now, that implies it supports image download.

Host side:

- Enter the Android SDK tools directory and find the fastboot utility (fastboot.exe on Windows, fastboot on Linux).
- Copy all downloaded images to the "images" folder.
- Run the following commands to flash the SD or eMMC:

```
$ fastboot flash bootloader images\u-boot-no-padding.bin
$ fastboot flash boot images\boot.img
$ fastboot flash system images\system.img
$ fastboot flash recovery images\recovery.img
$ fastboot reboot
```

Run the following commands to flash NAND

```
$ fastboot flash boot images\boot.img
$ fastboot flash androod_root images\android_root.img
$ fastboot flash recovery images\recovery.img
$ fastboot reboot
```

NOTE

Fastboot does not support flashing the bootloader to NAND storage.

6 What is the key mapping of the USB keyboard?

The default DELL USB keyboard key mapping is defined as shown below.

Key	Act as
ESC	BACK
F1	MENU
F2	SOFT_RIGHT
F3	CALL
F4	ENDCALL
F5	ENDCALL
F8	HOME
F9	DPAD_CENTER
UP	DPAD_UP
DOWN	DPAD_DOWN
BACK	DEL
ENTER	ENTER

7 How do I generate uramdisk.img?

The following steps generate a RAMDISK image recognized by U-Boot.

NOTE

Uramdisk is not used anymore.

```
Assume you had already built uboot. mkimage was generated under
myandroid/bootable/bootloader/uboot-imx/tools/
$ cd myandroid/out/target/product/sabresd_6q
$ ~/myandroid/bootable/bootloader/uboot-imx/tools/mkimage
-A arm -O linux -T ramdisk -C none -a 0x10308000 -n "Android Root Filesystem" -d ./
ramdisk.img
./uramdisk.img
```

7.1 How do I generate boot.img?

```
$ mkbootimg --kernel <kernel, zImage> --ramdisk < ramdisk> --base < baseaddr> --cmdline
<kernel
command line> --board < board name > -o <output>
$ cd myandroid
$ out/host/linux-x86/bin/mkbootimg --kernel kernel_imx/arch/arm/boot/zImage --ramdisk
ramdisk.img --base 0x10800000 --cmdline "console=ttymxc0,115200 init=/init rw video=mxcfb0
vmalloc=400M" --board mx6q_sabrelite -o boot.img
```

NOTE

If you want to extract and edit the zImage and ramdisk in the boot.img, see http://android-dls.com/wiki/index.php?title=HOWTO:_Unpack%2C_Edit%2C_and_Re-Pack_Boot_Images.

7.2 How do I change the boot command line in boot.img?

After using boot.img, we stored the default kernel boot command line inside this image.

It will package together during Android build.

You can change this by changing BOARD_KERNEL_CMDLINE which is defined in android/{product}/BoardConfig.mk file.

NOTE

Replace {product} with your product, eg, sabresd_6q.

7.3 How do I customize the boot animation?

The user can create his/her own boot animation for his/her device.

Android provides an easy way to replace its default boot animation by putting bootanimation.zip file into /system/media/.

- To create your own bootanimation .zip file, see http://www.addictivetips.com/mobile/how-to-change-customize-create-android-boot-animation-guide/.
- How to install the boot animation
  - On the host, use adb to download the bootanimation.zip file into the device, for example:

    ```
    $ adb push ~/Downloads/bootanimation.zip /mnt/sdcard
    ```
  - On the device, remount the /system to writable, and copy the file:

    ```
    $ mount -t ext4 -o rw,remount /dev/block/mmcblk0p5 /system
    $ busybox cp /mnt/sdcard/bootanimation.zip /system/media/
    $ mount -t ext4 -o ro,remount /dev/block/mmcblk0p5 /system
    ```

8 Why cannot certain APKs run on a device without a modem?

Some games require the TelephonyManager to return a device ID by getDeviceId() method of TelephonyManager, which is actually to return the mobile IMEI code with modem connected, but **null** without a modem.

They do not check the return value of getDeviceId(). Therefore, you can probably use null as device id without doing NULL as shown below:

```
TelephonyManager localTelephonyManager =
(TelephonyManager)oAppMain.getSystemService("phone");
      String str;
      if (localTelephonyManager != null)
      {
        nativeAddProperty("IMEI", localTelephonyManager.getDeviceId());
        Object[] arrayOfObject = new Object[1];
```

```
        arrayOfObject[0] = Integer.valueOf(Integer.parseInt(Build.VERSION.SDK));
        nativeAddProperty("DeviceID", String.format("%d", arrayOfObject));
        str = oAppMain.getClass().getPackage().getName();
        nativeAddProperty("Identifier", str);
      }
```

On the platform, when there is no modem connected, the TelephonyManager.getDeviceId() will return null. So, the JNI calling from here nativeAddProperty crashes in the dalvik JNI module, which will try to get strings from a null pointer.

For the tablet customer who does not have a modem connected, a workaround may need to be applied into framework/ base.git:

```
diff --git a/telephony/java/android/telephony/TelephonyManager.java
b/telephony/java/android/telephony/TelephonyMa
index db78e2e..82cf059 100755
--- a/telephony/java/android/telephony/TelephonyManager.java
+++ b/telephony/java/android/telephony/TelephonyManager.java
@@ -192,6 +192,9 @@ public class TelephonyManager {
      *   {@link android.Manifest.permission#READ_PHONE_STATE READ_PHONE_STATE}
      */
     public String getDeviceId() {
+       String s = "2222222222";
+       return s;
+       /*
         try {
             return getSubscriberInfo().getDeviceId();
         } catch (RemoteException ex) {
@@ -199,6 +202,7 @@ public class TelephonyManager {
         } catch (NullPointerException ex) {
             return null;
         }
+       */
     }
```

To verify your IMEI hard code, start the phone application and dial *#06#. It will show the IMEI code.

9 How do I enable or disable the bus frequency feature?

The Bus Frequency driver is used to slow down DDR, AHB, and AXI bus frequency in the SoC when the IPs which need high bus frequency are not working.

This saves the power consumption in Android earlysuspend mode significantly (playing audio with screen off). In this release, the bus frequency driver is **enabled** by default. If you want to enable or disable it, perform the following command in the console:

```
Disable:
$ echo 0 > /sys/devices/platform/imx_busfreq.0/enable
Enable:
$ echo 1 > /sys/devices/platform/imx_busfreq.0/enable
```

Note that if you are using Ethernet, the up operation will enable the FEC clock and force bus frequency to be high. That means you cannot go into low bus mode anymore, regardless whether the Ethernet cable is plugged or unplugged. Therefore, if you want the system to go into the low bus mode, you must do 'netcfg eth0 down' to shut down the FEC manually. If you want to use FEC again, do 'netcfg eth0 up' manually. When FEC is shut down with clock gated, the PHY cannot detect your cable in/out events.

10 How to set networking proxy for Wi-Fi?

To configure the proxy settings for a Wi-Fi network, do as follows:

- Tap and hold a network from the list of added Wi-Fi networks.
- Select "Modify Network".
- Choose "Show advanced options".
- If no proxy settings are present in the network, you have to - Tap "None", Select "Manual" from the menu that opens.
- Enter the proxy settings provided by the network administrator.
- Finally tap on the button denoted as "Save"

11 How do I enable 512MB DDR memory in U-Boot?

The default DDR memory configuration is "1GB" on i.MX 6Dual/6Quad and i.MX 6DualLite board.

The Freescale Android platform recommends to adopt 1GB memory by default. If you want to enable the 512MB memory, you can refer to the following modifications:

```
~/myandroid/bootable/bootloader/uboot-imx/include/configs/mx6dl_sabresd.h
~/myandroid/bootable/bootloader/uboot-imx/include/configs/mx6q_sabresd.h

-#define PHYS_SDRAM_1_SIZE      (1u * 1024 * 1024 * 1024)
+#define PHYS_SDRAM_1_SIZE      (1u * 512 * 1024 * 1024)

~/myandroid/bootable/bootloader/uboot-imx/include/configs/mx6dl_sabresd_android.h
~/myandroid/bootable/bootloader/uboot-imx/include/configs/mx6q_sabresd_android.h

-               "splashimage=0x30000000\0"                              \
+               "splashimage=0x1D000000\0"                              \

~/myandroid/device/fsl/sabresd_6dq/BoardConfig.mk (change the default boot commands)
-BOARD_KERNEL_CMDLINE := console=ttymxc0,115200 init=/init video=mxcfb0 video=mxcfb1:off
video=mxcfb2:off fbmem=10M fb0base=0x27b00000 vmalloc=400M androidboot.console=ttymxc0
+BOARD_KERNEL_CMDLINE := console=ttymxc0,115200 init=/init video=mxcfb0 video=mxcfb1:off
video=mxcfb2:off fbmem=10M fb0base=0x17b00000 gpumem=96M vmalloc=400M
androidboot.console=ttymxc0
```

NOTE

To run Android platform with 512MB DDR memory, you have to configure GPU memory as the smaller value which can allow more free memory in the system. Meanwhile, the fb0base should also be adjusted. When you build the software, ensure to include “~/myandroid/device/fsl/sabresd_6dq/BoardConfig.mk” changes, which will build-in the boot commands in your boot.img. If you use the release image, set the correct boot command manually as follows:

```
setenv bootargs console=ttymxc0,115200 init=/init video=mxcfb0
video=mxcfb1:off video=mxcfb2:off fbmem=10M fb0base=0x17b00000
gpumem=96M vmalloc=400M androidboot.console=ttymxc0
```

12 How do I run the image on i.MX 6Solo board?

The user can adopt the correct bootloader u-boot.bin so that the image can run on i.MX 6Solo board.

One emulation configuration for i.MX 6Solo has been provided in the default U-Boot delivery (mx6solo_sabresd_android_config) to ensure that DDR is 32 bit. The user can refer to this configuration and follow the instructions in User Guide 'Build U-Boot Image’ to build the i.MX 6Solo u-boot.bin.

```
For i.MX6solo SabreSD:
make mx6solo_sabresd_android_config
```

NOTE

To emulate single core, the user can add "nosmp" to the boot command line to disable SMP:

```
setenv bootargs console=ttymxc0,115200 init=/init nosmp
video=mxcfb0 video=mxcfb1:off video=mxcfb2:off gpumem=96M
fbmem=10M fb0base=0x17b00000 vmalloc=400M
androidboot.console=ttymxc0
```

The user can also disable the configuration "CONFIG_SMP" in the kernel.

13 How do I enable waking up system by using the USB HID device?

The Android framework supports wake up of the whole system by the external input device such as USB, BT, or HID.

If the user wants to enable this feature, the user needs to enable the USB remote wake up feature in kernel by default. Applying the following patches to the kernel enables USB remote wake up for OTG port and all the connected devices:

```
diff --git a/arch/arm/plat-mxc/usb_common.c b/arch/arm/plat-mxc/usb_common.c
index 97d963a..bc10b88 100755
--- a/arch/arm/plat-mxc/usb_common.c
+++ b/arch/arm/plat-mxc/usb_common.c
@@ -484,7 +484,7 @@ static int usb_register_remote_wakeup(struct platform_device *pdev)
            return -ENODEV;
   }
   irq = res->start;
-   device_set_wakeup_capable(&pdev->dev, true);
+   device_init_wakeup(&pdev->dev, true);
   enable_irq_wake(irq);
   return 0;
diff --git a/drivers/usb/core/hub.c b/drivers/usb/core/hub.c
index 55a63e4..b7da7b0 100644
--- a/drivers/usb/core/hub.c
+++ b/drivers/usb/core/hub.c
@@ -1546,7 +1546,7 @@ void usb_set_device_state(struct usb_device *udev,
            recursively_mark_NOTATTACHED(udev);
   spin_unlock_irqrestore(&device_state_lock, flags);
   if (wakeup >= 0)
-           device_set_wakeup_capable(&udev->dev, wakeup);
+           device_init_wakeup(&udev->dev, wakeup);
}
EXPORT_SYMBOL_GPL(usb_set_device_state);
```

14 How to configure the logical display density

The Android UI framework defines a set of standard logical densities to help the application developers target application resources. Device implementations MUST report one of the following logical Android framework densities:

- 120 dpi, known as 'ldpi'
- 160 dpi, known as 'mdpi'
- 213 dpi, known as 'tvdpi'
- 240 dpi, known as 'hdpi'

- 320 dpi, known as 'xhdpi'
- 480 dpi, known as 'xxhdpi'

Device implementations SHOULD define the standard Android framework density that is numerically closest to the physical density of the screen, unless that logical density pushes the reported screen size below the minimum supported. To configure the logical display density for framework, you must define the following line in the init.freescale.rc:

```
setprop ro.sf.lcd_density <density>
```

15 How do I enable and disable security ADB?

Android JB 4.2.2 introduces public key authentication policy when connecting to PC host via ADB.

This feature will keep ADB offline until the device and PC pass authentication. With security ADB enabled, Android SDK Platform-tools in PC host side higher than revision 16.0.2 is needed to make a valid ADB connection. Otherwise, PC can't connect the device over ADB(the status will always be offline). With latest Platform-tools, the device will pop-up a dialog to remind you to allow ADB connect or not when connecting the device with PC through ADB. You need select "OK" to allow PC connect your devices. If select "Cancel", the ADB connect will be rejected. For running CTS, you need to select the "Always allow from this computer". Otherwise, the device will still need your confirmation in each reboot. And to disable it please delete this line or comment it out as below:

```
  #setprop ro.adb.secure 1
```

16 How do I enable BT on imx6 Sabresd Rev C?

Android JB 4.2.2 introduces bluedroid infrastructure instead of bluez.

To enable BT on imx6 Sabresd RevC, please refer to below document on freescale community. https://community.freescale.com/docs/DOC-94235

17 How do I change BT MAC address on i. MX 6 Sabre SD Rev C board?

If you enable BT on i.MX 6 Sabre SD Rev C board and find that certain issues are caused by the fact that the same MAC address is shared by two BT peers, you could do the following to change BT MAC address.

1. cat /system/etc/firmware/ar3k/1020200/ar3kbdaddr.pst

   Check the original MAC address. The result looks like this: 1260417f0300(12:60:41:7f:03:00).

2. Mount -w -o remount /system

   Change system file system as writable.

3. echo "*******" > /system/etc/firmware/ar3k/1020200/ar3kbdaddr.pst
4. It is changed.

18 How do I configure rear and front camera?

Property "back_camera_name" and "front_camera_name" are used to configure the camera being used as rear camera or front camera. The name should be either v4l2_dbg_chip_ident.match.name returned from v4l2's IOCTL VIDIOC_DBG_G_CHIP_IDENT or v4l2_capability.driver returned from v4l2's IOCTL VIDIOC_QUERYCAP. Camera HAL will go through all the V4L2 device present in system. Camera HAL will choose the first matched name in property setting as the corresponding camera. Comma is used as a delimiter of different camera names among multiple camera selections.

Below is an example been set in myandroid/device/fsl/sabresd_6dq/init.rc. setprop back_camera_name ov5640_mipi setprop front_camera_name uvc,ov5642_camera,ov5640_camera.

media_profiles.xml in /system/etc is used to configure the parameters been used in recording video and taking picture. Freescale provide several media profile examples which help customers to make the parameters align with their camera module's capability and their device definition.

Table 1. Camera Settings

Profile file name	Rear camera	Front camera
media_profiles_1080p.xml	maximum to 1080P, 30FPS and 8Mb/s for recording video	maximum to 720P, 30FPS, and 3Mb/s for recording video
media_profiles_720p.xml	maximum to 720P, 30FPS, and 3Mb/s for recording video	maximum to 720P, 30FPS, and 3Mb/s for recording video
media_profiles_480p.xml	maximum to 480P, 30FPS, and 2Mb/s for recording video	maximum to 480P, 30FPS, and 2Mb/s for recording video
media_profiles_qvga.xml	maximum to QVGA, 15FPS, and 15Mb/s for recording video	maximum to QVGA, 15FPS, and 15Mb/s for recording video

NOTE

Since not all uvc cameras can have 1080P, 30 FPS resolution setting, it is recommended that media_profiles_480p.xml be used for any board's configuration which defines the uvc as rear camera or front camera.

19 How do I configure camera sensor parameters?

Android jb4.2 version needs camera sensor focal length and sensitive element size to calculate view angle when using panorama. The focal length and sensitive element size should be customized based on the camera sensor being used. The default release has the parameters for ov5642/ov5640 as front/back camera.

The Ov5640Csi.*/Ov5640Mipi.*/Ov5642Csi.* in hardware/imx/mx6/libcamera2 are provided to configure sensor. They implement class Ov5640Mipi/Ov5642Csi/Ov5640Csi. For a new camera sensor, a new camera sensor class should be created with the corresponding focal length and sensitive element size as the variables mFocalLength, mPhysicalWidth, and mPhysicalHeight in the class.

Table 2. Camera Sensor Parameters

Parameters	Description
mFocalLength	mFocalLength

Table continues on the next page...

Table 2. Camera Sensor Parameters (continued)

Parameters	Description
mPhysicalWidth	sensitive element width
mPhysicalHeight	sensitive element height

20 How do I configure UBI image for miscellaneous Nand chips?

The default UBI image built for SabreAuto board is configured for the Nand chip MT29F8G08ABABAWP with below attribution:

- Minimum input/output unit size of the flash is 4096
- Logical eraseblock size(LEB) of the UBI volume is 516096
- Physical eraseblock size of the flash chip is 512Kib

Above information can be fetched through below command or related Nand chip's specification:

```
          $ mtdinfo /dev/mtd3 -u
mtd3
Name:                           user
Type:                           nand
Eraseblock size:                262144 bytes, 256.0 KiB
Amount of eraseblocks:          3456 (905969664 bytes, 864.0 MiB)
Minimum input/output unit size: 4096 bytes
Sub-page size:                  4096 bytes
OOB size:                       224 bytes
Character device major/minor:   90:6
Bad blocks are allowed:         true
Device is writable:             true
Default UBI VID header offset:  4096
Default UBI data offset:        8192
Default UBI LEB size:           253952 bytes, 248.0 KiB
Maximum UBI volumes count:      128
```

Below is an example to make the UBI image for the Nand MT29F8G08ABACAWP, which has the following attribution:

```
minimum input/output unit size of the flash is 4096
logical eraseblock size of the UBI volume is 253952
physical eraseblock size of the flash chip is 256Kib
$~/myandroid/device/fsl$ git diff
diff --git a/sabreauto_6q/BoardConfig.mk b/sabreauto_6q/BoardConfig.mk
index b118ec5..f3db72c 100644
--- a/sabreauto_6q/BoardConfig.mk
+++ b/sabreauto_6q/BoardConfig.mk
@@ -85,8 +85,8 @@ BOARD_KERNEL_CMDLINE := console=ttymxc3,115200 init=/init
video=mxcfb0:dev=ldb,b
 ifeq ($(TARGET_USERIMAGES_USE_UBIFS),true)
 #UBI boot command line.
 UBI_ROOT_INI := device/fsl/sabreauto_6q/ubi/ubinize.ini
-TARGET_MKUBIFS_ARGS := -m 4096 -e 516096 -c 4096 -x none
-TARGET_UBIRAW_ARGS := -m 4096 -p 512KiB $(UBI_ROOT_INI)
+TARGET_MKUBIFS_ARGS := -m 4096 -e 253952 -c 4096 -x none
+TARGET_UBIRAW_ARGS := -m 4096 -p 256KiB $(UBI_ROOT_INI)
```

NOTE

This NAND partition table must align with MFGTool's config.
BOARD_KERNEL_CMDLINE += mtdparts=gpmi-nand:16m(bootloader), 16m(bootimg),128m(recovery),-(root) ubi.mtd=3.

Document Number FAQs
Rev jb4.2.2_1.0.0-GA
05/2013