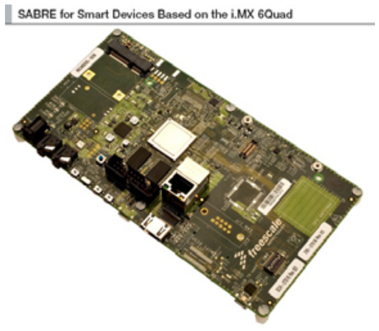
WL18xx First Time Getting Started Guide (IMX6)

Contents

Hardware

- i.MX SABRE board (imx6qsabresd): See This Link (http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=RDIMX6SABREBRD)



i.MX SABRE board

- **TI** WiLink™ WL18xx WLAN SDIO/BT UART adapter Board: wl18xxcom82sdmmc Adapter kit (http://www.ti.com/tool/wl18xxcom82sdmmc)

WiLink™ WL18xx SDIO Board

- **TI** - WL1835MOD COM8 Kit (http://www.ti.com/tool/wl1835modcom8b),* or **TI** WL1837MOD COM8 Kit (http://www.ti.com/tool/wl1837modcom8i)

WL1835MOD COM8 Kit

WL1837MOD COM8 Kit

- SD Card (2GB minimum)
- 5V Power supply
- microUSB cable

Preparing the Environment

Configure your Host PC - Serial Terminal

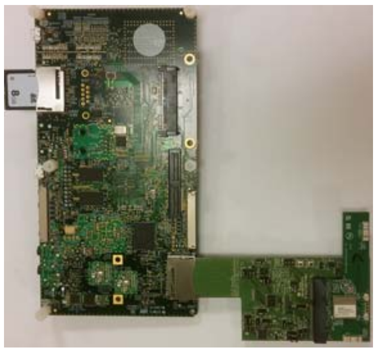
In order to communicate with the SABRE board and run the WLAN/BT demos, a serial port terminal program must be installed on your host PC. For Windows, you could use Teraterm or PuTTY. For Linux, we recommend Minicom.
Open PC's serial port terminal program and use the following settings:

```
Baud rate: 115200
Data : 8 bit
Parity: None
Stop: 1 bit
Flow control: none
Transmit delay msec/char: 1
Transmit delay msec/line: 1
```

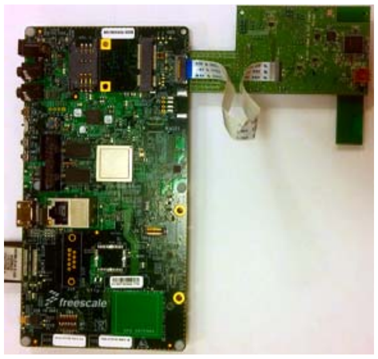
Setup your SABRE board

This section will walk you through setting-up your SABRE board for use with the WiLink8 Demos:

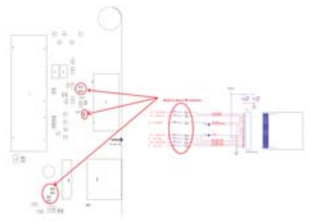
- Plug the WL18xx SDIO Board into the **SD2** port of the SABRE board, and connect provided "flat cable" between the adapter **J13** FPC connector and SABRE **J13** connector



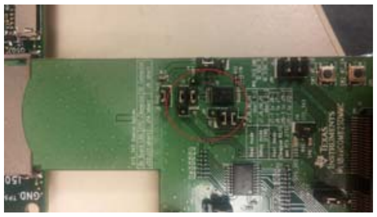
SABRE setup Bottom view


SABRE setup Top view

```
Important Note: when purchased the J13 port on the SABRE is not routed to the host processor.
In order to use J13, the resistors R209,210,211,212,213,214 and 215 need to populated on the SABRE board.
```


Addon Resistor location

```
The adapter board comes with [default (http://www.ti.com/lit/ug/swru398/swru398.pdf)] jumpers assembly to work with a host.
One exception: J10, J11 jumpers should be placed as in the picture below for the HCI UART CTS/RTS to function correctly with SABRE board.
```


J8, J10 Jumper Settings

- Plug the WL18xxMOD COM8 Kit into the WL18xx SDIO Board.
- Insert the SD card loaded with the latest image to the **SD3** port of the SABRE board.
- Plug the USB cable into your host PC and plug it to the SABRE board USB port **J509**.
- Plug the 5V Power supply into the SABRE board power socket and power on the EVM.

Integration with Freesacale Linux BSP

To build and integrate the WiLink8 related software components into your own file system, you need to start by getting the official imx linux kernel source. The kernel source code is available at the following git:

```
git.freescale.com/git/cgit.cgi/imx/linux-2.6-imx.git
```

fetching and building the imx kernel source code

1. create a working directory on you linux host 2. clone the git to this location 3. Checkout the release tag (currently **rel_imx_3.14.28_1.0.0_ga**)

```
mkdir ~/imx
cd ~/imx
git clone http://git.freescale.com/git/cgit.cgi/imx/linux-2.6-imx.git
git checkout rel_imx_3.14.28_1.0.0_ga
```

Patching the kernel for WiLink8 build

The WiLink8 drivers are built out of tree using the build script provided here (http://processors.wiki.ti.com/index.php/WL18xx_System_Build_Scripts). These scripts now also provide the option to apply all the patched required to the i.Mx kernel to add WiLink8 Wifi and Bluetooth as well as patching the device tree files for the EVK and SABRE boards to enable them. So initial step is to download the script (step 1 on link) and then modify setup-env for your configuration (step 2). They key item here is to add KERNEL_VARIANT to setup-env in order to get the script to patch the **rel_imx_3.14.28_1.0.0_ga** release.

```
...
export KERNEL_VARIANT=imx-3.14.28
...
```

Next step is to apply the required patches to the kernel

```
./build_wl18xx.sh patch_kernel
```

Now the kernel can be built with these patches applied as follows from inside the kernel source directory:

```
ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- make imx_v7_defconfig
ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- make zImage modules dtbs
```

Installing build artifacts into the file system

Assuming your sdcard has two partions (**boot** and **rootfs**) and is mounted at **/media/boot** and **/media/rootfs** the following commands can be used to install all build artifacts into your sdcard:

- from the kernel source directory run the following command to install the kernel modules into your file system

```
ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- make modules_install INSTALL_MOD_PATH=/media/rootfs
```

- copy the zImage and .dtb file for the SABRE board to your boot partition

```
cp arch/arm/boot/zImage /media/boot
cp arch arm/boot/dts/imx6q-sabresd.dtb /media/boot
```

Building WiLink8 related artifacts

Now the kernel has been built the next stage is to do a **full** build of the WiLink8 related components using the build script. This is done by following step 3 (download code) and step 4 (build drivers) from WL18xx System Build Scripts (http://processors.wiki.ti.com/index.php/WL18xx_System_Build_Scripts) page.

Installing WiLink8 build artifacts into the rootfs partition of your sdcard

Once a full **./build_wl18xx.sh** has been completed, a tar file containing all build artifacts is created inside the **outputs** directory inside the **build-utilites** directory. This tar file is called **fs_skeleton.tbz2**
extract the content of this tar file on top of your **rootfs** sdcard partition. See the following example:

```
cd /media/rootfs
sudo tar xf ~/build-utilities/outputs/fs_skeleton.tbz2
depmod -a
```

Required Connection for HW Integration design

NOTE The connection is based on the i.MX Sabre board with MCIMX6Q6AVT10AC device

Following pin to pin connections are required in order to integrate smoothly WL8 TI module with IMX6Q device.

IMX6Q Line name	IMX6Q pin	WL8 module pin	WL8 Line name
KEY_ROW4	V5	51	**HCI UART CTS**
KEY_COL4	T6	50	**HCI UART RTS**
KEY_ROW1	U6	52	**HCI UART TX**
KEY_COL1	U7	53	**HCI UART RX**
KEY_ROW6	T1	41	**BT EN**
KEY_ROW0	V6	40	**WL EN**
KEY_COL0	W5	14	**WL IRQ**
SD2_DATA3	B22	13	**WL SDIO D3**
SD2_DATA2	A23	12	**WL SDIO D2**
SD2_DATA1	E20	11	**WL SDIO D1**
SD2_DATA0	A22	10	**WL SDIO D0**
SD2_CLK	C21	8	**WL SDIO CLK**
SD2_CMD	F19	6	**WL SDIO CMD**

Running the Demos

Boot the Board

To boot the board, simply apply power to the board by plugging in the 5V Power Supply. If the USB cable is plugged in and the serial port is configured correctly, you should see the output on your serial terminal(will take 1-2 min to load). You will see the board reach u-boot, and then automatically boot into the Linux kernel.

When prompted for a login, use root.

Demo Guide

The demos for both BT & WLAN is available on the main WL18xx page

Integration with Freescale android Lollipop (L5.0.0_1.0.0-ga) Release

The following sections describe the process for integrating all the wl18xx related artifact on top of the official Android "Lollipop" (L5.0.0_1.0.0-ga) from Freescale.
The process is composed of two parts:

- Obtaining the official **L5.0.0_1.0.0-ga** from the Freescale website and building it from sources for the sabresd_6dq platform
- Integrating the wl18xx related components for having both Wifi and Bluetooth into the android repository and building the updated images

Obtaing and building the official android relese from Freescale

- start by obtaining the officcal android L5.0.0_1.0.0-ga Release from the freescale web site:

```
Freescale Android 5.0.0 Lollipop Source Code (http://www.freescale.com/webapp/Download?colCode=IMX6-L500-100-ANDROID-SOURCE-BSP&appType=license&location=null&Parent_nodeId=1337699
481071706174845&Parent_pageType=product)
```

- Download the documentation package from the following link:

```
Freescale Android 5.0.0 BSP Documentation (http://www.freescale.com/webapp/Download?colCode=IMX6_L500_100_ANDROID_DOCS&Parent_nodeId=1337699481071706174845&Parent_pageType=product
&Parent_nodeId=1337699481071706174845&Parent_pageType=product&Parent_nodeId=1337699481071706174845&Parent_pageType=product&Parent_nodeId=1337699481071706174845&Parent_pageType=pro
duct)
```

- Extract the documentation package and follow sections 1-3 of the **Android_User's_Guide.pdf** document from the documentation package above for setting up your build environment, fetching and building the official **L5.0.0_1.0.0-ga** Android image for the SD card on the SABRE-SD Board
- Follow section 5.1 for preapring an SDcard for the sabre-sd platform
- Boot the sabre platform and make sure the original L5.0.0_1.0.0-ga image is functional

Integration of the wl18xx related packages into the imx android repo

Note: the folwing steps should be done **only** after the previous section has completed ok and you have a working android image

- Using the **same** build setup that was used for fetching and building the above images, please follow the section below for adding wl18xx related components into the freescale android repository

```
cd $MYDROID/hardware/ti
git clone git://git.omapzoom.org/platform/hardware/ti/wpan.git
cd wpan
git checkout 34aa262bef0ee3b02ae7c8d030557a1a6472c3be
cd $MYDROID/hardware/ti
git clone git://git.omapzoom.org/platform/hardware/ti/wlan.git
cd wlan
git checkout 95400695306cba64700d97bb7e446b3b5e871a37
cd $MYDROID/device
mkdir ti; cd ti
git clone git://git.omapzoom.org/device/ti/proprietary-open.git
cd proprietary-open
git checkout 29bd91158e93bcde70c194be4dacc4af2c01c04c
cd $MYDROID/external
git clone git://git.omapzoom.org/platform/external/crda.git
cd crda
git checkout c49a083c96fe60682ddf2ba9cddc9003b5564058
```

Patching the netd and core gits

- Use the commands below for applying the following patches on top of the android repo

```
cd $MYDROID/system/netd
curl -k "http://review.omapzoom.org/gitweb?p=platform/system/netd.git;a=patch;h=9e9554cb9b059c159586638cd700922d8b3532ef" | git apply
cd $MYDROID/system/core
curl -k "http://review.omapzoom.org/gitweb?p=platform/system/core.git;a=patch;h=0e55183d86e75a1c6fcabb5b5e94f0d0e9873034" | git apply
```

Patching the sabresd_sdq platform for enabling wl18xx

- Use the commands below for applying the following patch on to of the **device/fsl** git for activating wl18xx support

```
cd $MYDROID/device/fsl
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga/device-fsl-patches/0001-sabresd_6dq-add-wilink8-
platform-support.patch" | git apply
```

Adding wilink8 related support to the android kernel

- Apply the kernel patches drom the following link "kernel_imx" directory.

https://git.ti.com/wilink8-wlan/build-utilites/trees/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga

The following commands can be used for this action

```
cd $MYDROID/kernel_imx
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga/0001-imx6q-sabresd-add-support-for-wilink8-wlan-
and-bluet.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga/0002-Bluetooth-Add-tty-HCI-driver.patch" | git
apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga/0003-imx_v7_defconfig-enable-Wilink8-related-
switches.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga/0004-st_kim-do-not-use-debugfs-functions-if-not-
enabled.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga/0005-st_kim-allow-suspend-if-callback-is-not-
registered.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga/0006-btwilink-add-minimal-device-tree-
support.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga/0007-ti-st-add-device-tree-support.patch" | git
apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga/0008-imx6sl-evk-add-support-for-wilink8-wlan-and-
bluetoot.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga/0009-imx6-decrease-wilink8-sdio-pins-drive-
strength.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga/0010-mmc-Add-SDIO-function-devicetree-subnode-
parsing.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga/0011-imx6q-sabresd-update-wilink8-entries-for-
r8.6.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga/0012-imx6sl-evk-update-wilink8-entries-for-
r8.6.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga/0013-imx_v7_android_defconfig-enable-Wilink8-
related-swit.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga/0014-drivers-misc-ti-st-fix-debugfs-creation-
error-handli.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.10.53-android-5.0.0_1.0.0-ga/0015-drivers-misc-ti-st-fix-null-pointer-
exception-in-st_.patch" | git apply
```

- Rebuild the android kernel after applying the kernel patches using the following sequence:

```
cd $MYDROID/kernel_imx
make imx_v7_android_defconfig
make uImage LOADADDR=0x10008000
```

Building the wl18xx related module with the updated kernel

The wl18xx related drivers are built as modules with the android kernel using **backports**

Use the following sequence for building the modules with the android kernel

```
export ARCH=arm
export CROSS_COMPILE=${MYDROID}/prebuilts/gcc/linux-x86/arm/arm-eabi-4.6/bin/arm-eabi-
export KERNEL_DIR=${YOUR_PATH}/kernel_imx/
export KLIB=${KERNEL_DIR}
export KLIB_BUILD=${KERNEL_DIR}
cd ${MYDROID}/hardware/ti/wlan/mac80211/compat_wl18xx/
make defconfig-wl18xx
make
```

Installing the compiled modules into the android file system

Use the following sequence for copying the compiled drivers (.ko) into the android image system aread

Note: The modules are installed into **/system/lib/modules** and are loaded from **init.rc** when the android image is booting

```
cd $OUT/system/lib/
mkdir modules;cd modules
cp -fp ${MYDROID}/hardware/ti/wlan/mac80211/compat_wl18xx/compat/compat.ko .
cp -fp ${MYDROID}/hardware/ti/wlan/mac80211/compat_wl18xx/net/wireless/cfg80211.ko .
cp -fp ${MYDROID}/hardware/ti/wlan/mac80211/compat_wl18xx/net/mac80211/mac80211.ko .
cp -fp ${MYDROID}/hardware/ti/wlan/mac80211/compat_wl18xx/drivers/net/wireless/ti/wl18xx/wl18xx.ko .
cp -fp ${MYDROID}/hardware/ti/wlan/mac80211/compat_wl18xx/drivers/net/wireless/ti/wlcore/wlcore.ko .
cp -fp ${MYDROID}/hardware/ti/wlan/mac80211/compat_wl18xx/drivers/net/wireless/ti/wlcore/wlcore_sdio.ko .
```

Building the final android image

Use the follwing seuence for rebuilding the android images that now includes all the added wl18xx releated components

```
cd $OUT
rm *.img
rm obj/PACKAGING/systemimage_intermediates/system.img
rm -fr root/
rm -fr recovery/
```

```
cd $MYDROID
make BUILD_TARGET_DEVICE=sd
```

Program the sabre platform SD Card with the updated android image that includes the wl18xx support

Note: The following sequence assumes that the android image is booting from SD Card Use the following sequence for updating the SD Card that is already running the standard **L5.0.0_1.0.0-ga** image:

- Power off the sabre platform and remove the SD Card mounted into the **SD3** port
- Plug the SD Card into your Linux host and check its mount point (/dev/sd**<x>**) using **mount**
- Assuming the SD Card has been mounted as **/dev/sdc** use the below sequence to upate the system and boot images:

```
cd $OUT
sudo umount /dev/sdc1
sudo umount /dev/sdc5
sudo dd if=boot.img of=/dev/sdc1
sudo dd if=system.img of=/dev/sdc5
sync
```

- Eject the SD Card from the linux host
- Pluig the SD Card into the sabre platform SD3 port
- Power on the platform and wait for the android image to boot
- You should now be able to got into the **settings** page and enable wifi and bluetooth

Integration with Freescale android Lollipop (L5.1.1_2.1.0-ga) Release

The following sections describes the process for integrating all the wl18xx related artifact on top of the official Android "Lollipop" (L5.2.2_2.1.0-ga) from NXP.
The process is composed of two parts:

- Obtaining the official **L5.1.1_2.1.0-ga** from the NXP website and building it from sources for the sabresd_6dq platform
- Integrating the wl18xx related components for having Wifi, into the android repository and building the updated images

Obtaing and building the official android relese from NXP

- start by obtaining the officcal android L5.1.1_2.1.0-ga Release from the NXP web site:

```
NXP Android 5.1.1 Lollipop Source Code (https://www.nxp.com/webapp/sps/download/license.jsp?colCode=IMX6_L5.1_2.1.0_AND_SOURCE_BSP&appType=file1&location=null&DOWNLOAD_ID=null)
```

- Download the documentation package from the following link:

```
NXP Android 5.1.1 BSP Documentation (https://www.nxp.com/webapp/Download?colCode=IMX6_L5.1_2.1.0_ANDROID-DOCS&location=null&fpsp=1&WT_TYPE=Supporting%20Information&WT_VENDOR=FREES
CALE&WT_FILE_FORMAT=gz&WT_ASSET=Documentation&fileExt=.gz&Parent_nodeId=1265411638783721675357&Parent_pageType=product)
```

- Extract the documentation package and follow sections 1-3 of the **Android_User's_Guide.pdf** document from the documentation package above for setting up your build environment, fetching and building the official **L5.1.1_2.1.0-ga** Android image for the SD card on the SABRE-SD Board
- Follow section 5.1 for preapring an SDcard for the sabre-sd platform
- Boot the sabre platform and make sure the original L5.1.1_2.1.0-ga image is functional

Integration of the wl18xx related packages into the android repo

Note: the folwing steps should be done **only** after the previous section has completed ok and you have a working android image

- Using the **same** build setup that was used for fetching and building the above images, please follow the section below for adding wl18xx related components into the android repository

```
cd $MYDROID/hardware/ti
git clone git://git.omapzoom.org/platform/hardware/ti/wpan.git
cd wpan
git checkout 34aa262bef0ee3b02ae7c8d030557a1a6472c3be
cd $MYDROID/hardware/ti
git clone git://git.omapzoom.org/platform/hardware/ti/wlan.git
cd wlan
git checkout 95400695306cba64700d97bb7e446b3b5e871a37
cd $MYDROID/device
mkdir ti; cd ti
git clone git://git.omapzoom.org/device/ti/proprietary-open.git
cd proprietary-open
git checkout 29bd91158e93bcde70c194be4dacc4af2c01c04c
cd $MYDROID/external
git clone git://git.omapzoom.org/platform/external/crda.git
cd crda
git checkout c49a083c96fe60682ddf2ba9cddc9003b5564058
cd $MYDROID/external/wpa_supplicant_8
git checkout lollipop-mr1-release
```

Patching the netd and core gits

- Use the commands below for applying the following patches on top of the android repo

```
cd $MYDROID/system/netd
curl -k "http://review.omapzoom.org/gitweb?p=platform/system/netd.git;a=patch;h=88e4873ecc9dda720ecff22341cf45636a812570" | git apply
cd $MYDROID/system/core
curl -k "http://review.omapzoom.org/gitweb?p=platform/system/core.git;a=patch;h=0e55183d86e75a1c6fcabb5b5e94f0d0e9873034" | git apply
```

Patching the sabresd_sdq platform for enabling wl18xx wlan

- Use the commands below for applying the following patch on to of the **device/fsl** git for activating wl18xx support

```
cd $MYDROID/device/fsl
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.14.52-android-5.1.1_2.1.0-ga/device-fsl-patches/0001-sabresd_6dq-add-wilink8-
wlan-platform-support.patch" | git apply
```

Patching the external/sepolicy git for enabling wl18xx kernel modules loading

- Use the commands below for applying the following patch on to of the **external/sepolicy**

```
cd $MYDROID/external/sepolicy
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.14.52-android-5.1.1_2.1.0-ga/external-sepolicy-patches/0001-sepolicy-enable-
loading-of-modules.patch" | git apply
```

Adding wilink8 related support to the android kernel

- Apply the kernel patches drom the following link "kernel_imx" directory.

https://git.ti.com/wilink8-wlan/build-utilites/trees/master/patches/kernel_patches/imx-3.14.52-android-5.1.1_2.1.0-ga

The following commands can be used for this action

```
cd $MYDROID/kernel_imx
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.14.52-android-5.1.1_2.1.0-ga/0001-imx6q-sabresd-add-support-for-wilink8-wlan-
and-bluet.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.14.52-android-5.1.1_2.1.0-ga/0002-imx6sl-evk-add-support-for-wilink8-wlan-and-
bluetoot.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.14.52-android-5.1.1_2.1.0-ga/0003-Bluetooth-Add-tty-HCI-driver.patch" | git
apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.14.52-android-5.1.1_2.1.0-ga/0004-imx_v7_defconfig-enable-Wilink8-related-
switches.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.14.52-android-5.1.1_2.1.0-ga/0005-st_kim-do-not-use-debugfs-functions-if-not-
enabled.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.14.52-android-5.1.1_2.1.0-ga/0006-st_kim-allow-suspend-if-callback-is-not-
registered.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.14.52-android-5.1.1_2.1.0-ga/0007-btwilink-add-minimal-device-tree-
support.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.14.52-android-5.1.1_2.1.0-ga/0008-ti-st-add-device-tree-support.patch" | git
apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.14.52-android-5.1.1_2.1.0-ga/0009-mmc-Add-SDIO-function-devicetree-subnode-
parsing.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.14.52-android-5.1.1_2.1.0-ga/0010-imx_v7_android_defconfig-enable-Wilink8-
related-swit.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.14.52-android-5.1.1_2.1.0-ga/0011-drivers-misc-ti-st-fix-debugfs-creation-
error-handli.patch" | git apply
curl -k "http://git.ti.com/wilink8-wlan/build-utilites/blobs/raw/master/patches/kernel_patches/imx-3.14.52-android-5.1.1_2.1.0-ga/0012-drivers-misc-ti-st-fix-null-pointer-
exception-in-st_.patch" | git apply
```

- Rebuild the android kernel after applying the kernel patches using the following sequence:

```
cd $MYDROID/kernel_imx
make imx_v7_android_defconfig
make uImage LOADADDR=0x10008000
```

Building the wl18xx related module with the updated kernel

The wl18xx related drivers are built as modules with the android kernel using **backports**

Use the following sequence for building the modules with the android kernel

```
export ARCH=arm
export CROSS_COMPILE=${MYDROID}/prebuilts/gcc/linux-x86/arm/arm-eabi-4.6/bin/arm-eabi-
export KERNEL_DIR=${YOUR_PATH}/kernel_imx/
export KLIB=${KERNEL_DIR}
export KLIB_BUILD=${KERNEL_DIR}
cd ${MYDROID}/hardware/ti/wlan/mac80211/compat_wl18xx/
make defconfig-wl18xx
make
```

Installing the compiled modules into the android file system

The modules are installed into **/system/lib/modules** as part of the rebuild done below and are loaded from **init.rc** when the android image is booting

Building the final android image

Use the follwing seuence for rebuilding the android images that now includes all the added wl18xx releated components

```
cd $OUT
rm *.img
rm obj/PACKAGING/systemimage_intermediates/system.img
rm -fr root/
rm -fr recovery/
cd $MYDROID
make BUILD_TARGET_DEVICE=sd
```

Program the sabre platform SD Card with the updated android image that includes the wl18xx support

Note: The following sequence assumes that the android image is booting from SD Card Use the following sequence for updating the SD Card that is already running the standard **L5.0.0_1.0.0-ga**

image:

- Power off the sabre platform and remove the SD Card mounted into the **SD3** port
- Plug the SD Card into your Linux host and check its mount point (/dev/sd**<x>**) using **mount**
- Assuming the SD Card has been mounted as **/dev/sdc** use the below sequence to update the system and boot images:

```
cd $OUT
sudo umount /dev/sdc1
sudo umount /dev/sdc5
sudo dd if=boot.img of=/dev/sdc1
sudo dd if=system.img of=/dev/sdc5
sync
```

- Eject the SD Card from the linux host
- Plugthe SD Card into the sabre platform SD3 port
- Power on the platform and wait for the android image to boot
- You should now be able to go into the **settings** page and enable wifi

{{ 1. switchcategory:MultiCore= ■ For technical support on MultiCore devices, please post your questions in the C6000 MultiCore Forum ■ For questions related to the BIOS MultiCore SDK (MCSDK), please use the BIOS Forum Please post only comments related to the article **WL18xx First Time Getting Started Guide (IMX6)** here.	Keystone= ■ For technical support on MultiCore devices, please post your questions in the C6000 MultiCore Forum ■ For questions related to the BIOS MultiCore SDK (MCSDK), please use the BIOS Forum Please post only comments related to the article **WL18xx First Time Getting Started Guide (IMX6)** here.	C2000=*For technical support on the C2000 please post your questions on The C2000 Forum. Please post only comments about the article* ***WL18xx First Time Getting Started Guide (IMX6)*** *here.*	DaVinci=*For technical support on DaVincoplease post your questions on The DaVinci Forum. Please post only comments about the article* ***WL18xx First Time Getting Started Guide (IMX6)*** *here.*	MSP430=*For technical support on MSP430 please post your questions on The MSP430 Forum. Please post only comments about the article* ***WL18xx First Time Getting Started Guide (IMX6)*** *here.*	OMAP35x=*For technical support on OMAP please post your questions on The OMAP Forum. Please post only comments about the article* ***WL18xx First Time Getting Started Guide (IMX6)*** *here.*	OMAPL1=*For technical support on OMAP please post your questions on The OMAP Forum. Please post only comments about the article* ***WL18xx First Time Getting Started Guide (IMX6)*** *here.*	MAVRK=*For technical support on MAVRK please post your questions on The MAVRK Toolbox Forum. Please post only comments about the article* ***WL18xx First Time Getting Started Guide (IMX6)*** *here.*	*For technical support please post your questions at http://e2e.ti.com. Please post only comments about the article* ***WL18xx First Time Getting Started Guide (IMX6)*** *here.* }}

Links

Amplifiers & Linear
Audio
Broadband RF/IF & Digital Radio
Clocks & Timers
Data Converters

DLP & MEMS
High-Reliability
Interface
Logic
Power Management

Processors

- ARM Processors
- Digital Signal Processors (DSP)
- Microcontrollers (MCU)
- OMAP Applications Processors

Switches & Multiplexers
Temperature Sensors & Control ICs
Wireless Connectivity

Retrieved from "https://processors.wiki.ti.com/index.php?title=WL18xx_First_Time_Getting_Started_Guide_(IMX6)&oldid=212997"

This page was last edited on 18 February 2016, at 09:15.

Content is available under Creative Commons Attribution-ShareAlike unless otherwise noted.