Getting Started with the Source Code for iMX8 Boards

Diego Dorta

Version 0.3, Dec 18, 2017

Table of Contents

Source Codes for iMX8 Boards

This document was meant to guide users that want to build U-Boot and Linux Kernel for i.MX8 boards manually from the source codes. The next section presents three basic topics:

- How to compile the **U-Boot** for i.MX8QM LPDDR4 ARM2 and i.MX8MQ EVK Board.
- How to compile the **Kernel** for i.MX8QM LPDDR4 ARM2 and i.MX8MQ EVK Board.
- How to integrate them with a root file system generated by the Yocto Project.

The iMX8 boards described below use the same procedure for generating the Kernel and root file system. However, the U-Boot is different for each board and the steps to compile U-Boot for these boards can be found on section 2.

To create a full image, choose the corresponding U-Boot step for the board and then follow the Kernel and rootfs steps:

1. Toolchain

There are two toolchains to cross compile the source code: 4.9.51-mx8-beta Yocto (aarch64-toolchain-4.9.51-mx8-beta) or Ubuntu toolchain.

1.1. Yocto Toolchain

Go to the following link and download `fsl-imx-internal-xwayland-glibc-x86_64-fsl-image-gui-aarch64-toolchain-4.9.51-mx8-beta` package (1.5GB):

```
http://shlinux12.ap.freescale.net/Yocto_SDK/4.9.x-1.0.0-GCC6.2/aarch64/
```

After downloading it, change the permission to execute the file:

```
$ sudo chmod +x fsl-imx-internal-xwayland-glibc-x86_64-fsl-image-gui-aarch64-toolchain-4.9.51-mx8-beta.sh
```

Execute the file:

```
$ ./fsl-imx-internal-xwayland-glibc-x86_64-fsl-image-gui-aarch64-toolchain-4.9.51-mx8-beta.sh
```

To prepare the toolchain, just execute the following command line:

```
$ source /opt/fsl-imx-internal-xwayland/4.9.51-mx8-beta/environment-setup-aarch64-poky-linux
```

1.2. Ubuntu Toolchain

Install the Ubuntu toolchain ***gcc-aarch64-linux-gnu*** GNU C compiler for the arm64 architecture:

```
$ sudo apt-get install gcc-aarch64-linux-gnu
```

Export the environment variables:

```
$ export ARCH=arm64
$ export CROSS_COMPILE=/usr/bin/aarch64-linux-gnu-
```

NOTE This guide was tested with both cross compiler packages.

2. U-Boot

2.1. U-Boot for i.MX8QM LPDDR4 ARM2

To compile and flash the U-Boot into a SD card, follow the next **six** steps:

2.1.1. Mkimage Step

Create a folder to organize the files:

```
$ mkdir mx8
$ cd mx8/
```

Download the mkimage from:

```
$ git clone https://bitbucket.sw.nxp.com/scm/imx/imx-mkimage.git
$ cd imx-mkimage/
$ git checkout imx_4.9.51_imx8_beta1
```

NOTE Commit ID: e131af1031fff50106777c2338c41bf58c0b86a1

Once it is downloaded, just follow the next steps.

2.1.2. U-Boot Step

Download the U-Boot from:

```
$ git clone https://bitbucket.sw.nxp.com/scm/imx/uboot-imx.git
$ cd uboot-imx/
$ git checkout imx_v2017.03_4.9.51_imx8_beta1
```

NOTE Commit ID: c8e6eb2179512ee87c8faa08bbe826ead40ab702

To compile the U-Boot:

```
$ unset LDFLAGS
$ make -j8 imx8qm_lpddr4_arm2_defconfig
$ make -j8
```

The result of this compilation is an `u-boot.bin` file.

Copy the `u-boot.bin` to the mkimage iMX8QM folder:

```
$ cp u-boot.bin ../imx-mkimage/iMX8QM/
```

IMPORTANT Use `unset LDFLAGS` if the Yocto toolchain is used.

2.1.3. ATF (Arm Trust Firmware) Step

Download the ATF from:

```
$ git clone https://bitbucket.sw.nxp.com/scm/imx/arm-trusted-firmware.git
$ cd arm-trusted-firmware/
$ git checkout imx_4.9.11_imx8_alpha
```

NOTE Commit ID: fb892584c0f5f4897884f9b4f823c5af7282cfb3

Compile the ATF by running:

```
$ make PLAT=imx8qm bl31
```

The result is a `bl31.bin` file.

Copy the `bl31.bin` to the mkimage iMX8QM folder:

```
$ cp build/imx8qm/release/bl31.bin ../imx-mkimage/iMX8QM/
```

2.1.4. SC Firmware Step

NOTE This section needs some improvements. Currently it is not possible to compile the firmware, so the pre-built firmware is used instead.

Get the firmware from the NXP Alpha release:

```
https://www.nxp.com/webapp/Download?colCode=L4.9.11_IMX8QM_SOURCE_Alpha&appType=licens
e&Parent_nodeId=1454108179417712493862&Parent_pageType=product
```

Copy the `imx-sc-firmware-0.1.bin` to the previous created mx8 folder, then:

```
$ chmod +x imx-sc-firmware-0.1.bin
$ ./imx-sc-firmware-0.1.bin
```

Copy the firmware into the mkimage iMX8QM folder:

```
$ cp imx-sc-firmware-0.1/mx8qm-scfw-tcm.bin ../imx-mkimage/iMX8QM/
```

Then, rename the firmware to *scfw_tcm.bin*:

```
$ cd imx-mkimage/iMX8QM/
$ mv mx8qm-scfw-tcm.bin scfw_tcm.bin
```

2.1.5. Compiling

Check the files: `bl31.bin`, `scfm_tcm.bin`, `u-boot.bin`:

iMX8QM

```
$ cd imx-mkimage/
$ tree iMX8QM/

 ├──── bl31.bin
 ├──── expand_c_define.sh
 ├──── head.hash
 ├──── imx8qm_dcd_1.6GHz.cfg
 ├──── imx8qm_dcd_800MHz.cfg
 ├──── imx8qm_dcd.cfg
 ├──── lib
 │    ├──── ddrc_mem_map.h
 │    └──── ddr_phy_mem_map.h
 ├──── scfw_tcm.bin
 ├──── soc.mak
 ├──── u-boot-atf.bin
 ├──── u-boot.bin
 └──── u-boot-hash.bin

1 directory, 14 files
```

NOTE Open a new terminal to generate `flash.bin` file:

```
$ cd imx-mkimage/
$ make clean
$ make SOC=iMX8QM flash
```

2.1.6. Flashing

To flash the binary into the SD card, just:

```
$ sudo dd if=iMX8QM/flash.bin of=/dev/<your device> bs=1k seek=33 status=progress &&
sync
```

2.2. U-Boot for i.MX8MQ EVK Board

2.2.1. Mkimage Step

Create a folder to organize the files:

```
$ mkdir mx8
$ cd mx8/
```

Download the mkimage from:

```
$ git clone https://bitbucket.sw.nxp.com/scm/imx/imx-mkimage.git
$ cd imx-mkimage/
$ git checkout imx_4.9.51_imx8_beta1
```

NOTE Commit ID: e131af1031fff50106777c2338c41bf58c0b86a1

Once it is downloaded, just follow the next steps.

2.2.2. U-Boot Step

Download the U-Boot from:

```
$ git clone https://bitbucket.sw.nxp.com/scm/imx/uboot-imx.git
$ cd uboot-imx/
$ git checkout imx_v2017.03_4.9.51_imx8_beta1
```

NOTE Commit ID: c8e6eb2179512ee87c8faa08bbe826ead40ab702

To compile the U-Boot:

```
$ unset LDFLAGS
$ make -j8 imx8mq_evk_defconfig
$ make -j8
```

Copy the `u-boot.bin`, `u-boot-spl.bin`, `u-boot-nodtb.bin`, `fsl-imx8mq-evk.dtb` to the mkimage iMX8QM folder:

```
$ cp spl/u-boot-spl.bin ../imx-mkimage/iMX8M/
$ cp u-boot.bin ../imx-mkimage/iMX8M/
$ cp u-boot-nodtb.bin ../imx-mkimage/iMX8M/
$ cp arch/arm/dts/fsl-imx8mq-evk.dtb ../imx-mkimage/iMX8M/
```

IMPORTANT Use `unset LDFLAGS` if the Yocto toolchain is used.

2.2.3. ATF (Arm Trust Firmware) Step

Download the ATF from:

```
$ git clone https://bitbucket.sw.nxp.com/scm/imx/arm-trusted-firmware.git
$ cd arm-trusted-firmware/
$ git checkout imx_4.9.51_imx8_beta1
```

NOTE Commit ID: a4388010f06ceb8fa9a33391bff47dabd027dbb1

Compile the ATF by running:

```
$ make PLAT=imx8mq bl31
```

This builds `bl31.bin` file under `build/imx8mq/release/bl31.bin`.

Copy the `bl31.bin` to the mkimage iMX8M folder:

```
$ cp build/imx8mq/release/bl31.bin ../imx-mkimage/iMX8M/
```

2.2.4. Firmware iMX Step

Download the file `imx-4.9.51-mx8qm_beta1-external-mirror_2017-11-29.tar.gz` from:

```
http://yb2.am.freescale.net/build-
output/Linux_IMX_MX8_BETA/79/build_log/imx_rc_archive/external-mirror/
```

Follow the steps:

```
$ tar -xf imx-4.9.51-mx8qm_beta1-external-mirror_2017-11-29.tar.gz
$ chmod +x firmware-imx-7.1.bin
$ ./firmware-imx-7.1.bin
```

Copy the files to the mkimage iMX8M folder:

```
$ cd firmware-imx-7.1.bin/
$ cp firmware/hdmi/cadence/hdmi_imx8m.bin ../imx-mkimage/iMX8M/
$ cp firmware/ddr/synopsys/lpddr4_pmu_train_* ../imx-mkimage/iMX8M/
```

2.2.5. Compiling

Check the files: `bl31.bin`, `fsl-imx8mq-evk.dtb`, `hdmi_imx8m.bin`, `u-boot.bin`, `u-boot-nodtb.bin`, `u-boot-spl.bin`, `lpddr4_pmu_train_dmem.bin`, `lpddr4_pmu_train_imem.bin`:

iMX8M

```
$ cd imx-mkimage/
$ tree iMX8M/
├── bl31.bin
├── fsl-imx8mq-evk.dtb
├── hdmi_imx8m.bin
├── lib
│   └── ddr_memory_map.h
├── lpddr4_pmu_train_dmem.bin
├── lpddr4_pmu_train_imem.bin
├── mkimage_imx8.c
├── README
├── soc.mak
├── u-boot.bin
├── u-boot-nodtb.bin
└── u-boot-spl.bin

1 directory, 12 files
```

Open a new terminal to generate `flash.bin` file:

```
$ cd imx-mkimage/
$ make clean
$ make SOC=iMX8M flash_spl_uboot
```

NOTE Or uses: `make SOC=iMX8M flash_hdmi_spl_uboot`

2.2.6. Flashing

To flash the binary into the SD card, just:

```
$ sudo dd if=iMX8M/flash.bin of=/dev/<your device> bs=1k seek=33 status=progress &&
sync
```

2.3. U-Boot for i.MX8QXP LPDDR4 ARM2

2.3.1. Coming soon.

3. Kernel

3.1. Linux iMX

Download linux-imx from:

```
$ git clone https://bitbucket.sw.nxp.com/scm/imx/linux-imx.git
$ cd linux-imx/
$ git checkout imx_4.9.51_imx8_beta1
```

Export the toolchain:

```
$ source /opt/fsl-imx-internal-xwayland/4.9.11-8mq_alpha/environment-setup-aarch64-
poky-linux
$ unset LDFLAGS
```

3.2. Compiling

Compile the Kernel:

```
$ make ARCH=arm64 defconfig
$ make -j8 ARCH=arm64
```

3.3. Copy the binaries to SD card

Copy the `Image` file and `dtb` into your SD card:

```
$ cp arch/arm64/boot/Image /media/${USER}/Boot
$ cp arch/arm64/boot/dts/freescale/<corresponding_dtb_file>.dtb /media/${USER}/Boot
```

4. Root File System

As a quick alternative to create a complete rootfs, use the pre-built rootfs:

4.1. Downloading Files

Download the pre-built images from:

```
http://yb2.am.freescale.net/build-output/Linux_IMX_MX8_BETA/79/fsl-imx-xwayland/
```

4.2. Preparing Files

Uncompress the file by typing:

```
$ bunzip2 fsl-image-validation-imx-imx8qmlpddr4arm2-20171122025529.rootfs.tar.bz2
```

Copy the file into the SD card:

```
$ tar -xf fsl-image-validation-imx-imx8qmlpddr4arm2-20171122025529.rootfs.tar -C
/media/${USER}/rootfs
```

5. Building Root File System with Yocto Project

To get the BSP you need to have `repo utility` installed (only need to do this once).

```
$ mkdir ~/bin
$ curl http://commondatastorage.googleapis.com/git-repo-downloads/repo  > ~/bin/repo
$ chmod a+x ~/bin/repo
$ PATH=${PATH}:~/bin
```

5.1. i.MX8QM LPDDR4 ARM2

Download the `Yocto Project` Environment into your directory:

```
$ mkdir fsl-arm-yocto-bsp
$ cd fsl-arm-yocto-bsp
$ repo init -u https://source.codeaurora.org/external/imx/imx-manifest -b imx-linux-morty -m imx-4.9.51-8qm_beta1.xml
$ repo sync
```

The configuration for i.MX8QM LPDDR4 ARM2 is detailed below:

```
$ MACHINE=imx8qmlpddr4arm2 DISTRO=fsl-imx-wayland source ./fsl-setup-release.sh -b bld-wayland
```

Just run the following command for creating the image (this may take a while):

```
$ bitbake core-image-minimal
```

After this step is completed, flash the image into the SD card:

```
$ cd tmp/deploy/images/imx8qmlpddr4arm2/
$ sudo dd if=core-image-minimal-imx8qmlpddr4arm2-20171212175528.rootfs.sdcard of=/dev/<your_device> status=progress && sync
```

5.2. i.MX8MQ EVK Board

5.2.1. Coming soon.

5.3. i.MX8QXP LPDDR4 ARM2

5.3.1. Coming soon.

Feel free to contact me: diego.dorta@nxp.com