MaaXBoard OSM93 Yocto Development Guide

REV. LF6.6.3-1.0.0

Copyright Statement:

- The MaaXBoard OSM93 single board computer and its related intellectual property are owned by Avnet.
- Avnet has the copyright of this document and reserves all rights. Any part of the document should not be modified, distributed or duplicated in any approach and form with the written permission issued by Avnet.

Disclaimer:

- Avnet does not take warranty of any kind, either expressed or implied, as to the program source code, software and documents provided along with the products, and including, but not limited to, warranties of fitness for a particular purpose; The entire risk as to the quality or performance of the program is with the user of products.

Regulatory Compliance:

- MaaXBoard OSM93 single board computer has passed the CE, FCC & SRRC certification.

Catalog

Chapter 1 Build with Yocto

1.1 Setup Build Environment

To setup the build environment need:

- Hardware: It is recommended that at least 300GB of disk space and 4GB of RAM
- Software: Ubuntu 64-bit OS, 20.04 LTS version or later LTS version (Ubuntu Desktop or Ubuntu Server version). You could also run the Ubuntu 64-bit OS on virtual machine or in docker container.

The following packages are required for the development environment. The required packages can be installed using the bash script below:

```
$ sudo apt-get update
$ sudo apt-get install -y wget git-core diffstat unzip texinfo gcc-multilib \
build-essential chrpath socat cpio python python3 python3-pip python3-pexpect \
xz-utils debianutils iputils-ping python3-git python3-jinja2 libegl1-mesa libsdl1.2-dev \
pylint3 xterm rsync curl gawk zstd lz4 locales bash-completion
```

Install repo:

```
$ mkdir -p ~/bin
$ curl https://storage.googleapis.com/git-repo-downloads/repo > ~/bin/repo
$ chmod a+x ~/bin/repo
$ export PATH=~/bin:$PATH
```

Set Git configuration:

```
$ git config --global user.name "Your Name"
$ git config --global user.email "you@example.com"
```

1.2 Fetch Source Code

1.2.1 Download meta layers from NXP

```
$ mkdir ~/imx-yocto-bsp
$ cd ~/imx-yocto-bsp
```

```
$ repo init -u https://github.com/nxp-imx/imx-manifest -b imx-linux-nanbield -m imx-6.6.3-1.0.0.xml
```

```
$ repo sync
```

1.2.2 Download MaaXBoard OSM93 Source Code

To download the source code of MaaXBoard OSM93, clone the repository from Github:

```
$ cd ~/imx-yocto-bsp/sources
$ git clone https://github.com/Avnet/meta-maaxboard.git -b nanbield meta-maaxboard
```

1.3 Build

1.3.1 Edit build Configuration

If you want to create a new build folder or set the configuration for the first time, run the command:

```
$ cd ~/imx-yocto-bsp
$ MACHINE=maaxboard-osm93 source sources/meta-maaxboard/tools/maaxboard-setup.sh -b maaxboard-osm93/build
```

If you want to build in an existing build folder, use the following command:

```
$ cd ~/imx-yocto-bsp
$ source sources/poky/oe-init-build-env maaxboard-osm93/build
```

1.3.2 Build

Execute the following command to build a Weston Wayland image:

```
$ bitbake avnet-image-full
```

After the build has successfully completed, the output files are deployed in:

~/imx-yocto-bsp/maaxboard-osm93/build/tmp/deploy/images/maaxboard-osm93/

imx-boot-tagged	Bootloader Image
avnet-image-full-maaxboard-osm93 -xxxx.rootfs.wic	System image, this includes: Linux kernel, DTB and root file system.
Image	Kernel image
maaxboard-osm93.dtb	MaaXBoard OSM93 device tree binary
overlays	MaaXBoard OSM93 device tree overlay binary
avnet-image-full-maaxboard-osm93 -xxxx.rootfs.tar.bz2	System image compressed archive file

Chapter 2 Standalone Build of u-Boot and Kernel

This chapter describes how to build U-boot and Kernel using SDK or ARM GCC in a standalone environment.

2.1 Cross-compile tool chain

The cross-compile tool chain that is used, can be ARM GCC or Yocto SDK.

2.1.1 ARM GCC

Download the tool chain for the A-profile architecture on arm Developer GNU-A Downloads page. It is recommended to use the 10.3 version for this release. You can download the "gcc-arm-10.3-2021.07-x86_64-aarch64-none-linux-gnu.tar.xz ", and decompress the file into a local directory.

```
$ mkdir ~/toolchain
$ tar -xJf gcc-arm-10.3-2021.07-x86_64-aarch64-none-linux-gnu.tar.xz -C ~/toolchain
```

Execute the following command to check that the toolchain can be directly run.

```
$ cd toolchain/gcc-arm-10.3-2021.07-x86_64-aarch64-none-linux-gnu/bin/
$ ./aarch64-none-linux-gnu-gcc -v
```

To compile a project with ARM GCC, first set the environment with the following commands before building :

```
$ TOOLCHAIN_PATH=$HOME/toolchain/gcc-arm-10.3-2021.07-x86_64-aarch64-none-linux-gnu/bin
$ export PATH=$TOOLCHAIN_PATH:$PATH
$ export ARCH=arm64
$ export CROSS_COMPILE=aarch64-none-linux-gnu-
```

2.1.2 Yocto SDK

Generate an SDK from the Yocto Project build environment with the following command after generating the image in the previous chapter.

```
$ cd ~/imx-yocto-bsp
$ source sources/poky/oe-init-build-env maaxboard-osm93/build
$ bitbake avnet-image-full -c populate_sdk
```

The generated file is: *~/imx-yocto-bsp/maaxboard-osm93/build/tmp/deploy/sdk/*

fsl-imx-xwayland-glibc-x86_64-avnet-image-full-armv8a-maaxboard-osm93-toolchain-6.6-nanbield.sh

and execute this script to install the SDK. The default location is /opt but can be placed anywhere on the host machine.

```
lily@8d4a7d791316:~/imx-yocto-bsp-6.6.3/yocto/nanbield-lf-6.6.3-1.0.0/maaxboard-osm93/build/tmp/deploy/sdk$
./fsl-imx-xwayland-glibc-x86_64-avnet-image-full-armv8a-maaxboard-osm93-toolchain-6.6-nanbield.sh
NXP i.MX Release Distro SDK installer version 6.6-nanbield
==========================================================
Enter target directory for SDK (default: /opt/fsl-imx-xwayland/6.6-nanbield):
You are about to install the SDK to "/opt/fsl-imx-xwayland/6.6-nanbield". Proceed [Y/n]? Y
Extracting SDK.....................................done
Setting it up...done
SDK has been successfully set up and is ready to be used..
```

When using SDK to compile a project, first execute the following command to configure environment variables :

```
$ . /opt/fsl-imx-xwayland/6.6-nanbield/environment-setup-armv8a-poky-linux
```

2.2 Build U-Boot in a standalone environment

2.2.1 Get the source code and firmware

To get the source code of u-boot, imx-atf and imx-mkimage, execute the following commands:

```
$ mkdir tmp
$ cd tmp
$ git clone https://github.com/Avnet/uboot-imx.git -b maaxboard_lf-6.6.3-1.0.0
$ git clone https://github.com/Avnet/imx-atf.git -b maaxboard_lf-6.6.3-1.0.0
$ git clone https://github.com/Avnet/imx-mkimage.git -b maaxboard_lf-6.6.3-1.0.0
```

Download the firmware-imx, decompress and accept NXP EULA when running:

```
$ wget https://www.nxp.com.cn/lgfiles/NMG/MAD/YOCTO/firmware-imx-8.23.bin
$ chmod +x firmware-imx-8.23.bin
$ ./firmware-imx-8.23.bin
```

Execute the ‘ls’ command to view the tmp directory:

```
$ ls tmp
firmware-imx-8.23 firmware-imx-8.23.bin imx-atf imx-mkimage uboot-imx
```

So far, the required source code and firmware have been prepared.

2.2.2 Compile script

Create a bash script in the tmp directory and change the file mode:

```
$ cd tmp
$ touch make_mxosm93_uboot.sh
$ chmod 766 make_mxosm93_uboot.sh
$ vi make_mxosm93_uboot.sh
```

Copy the following content into the make_mxosm93_uboot.sh script:

```
#!/bin/bash
PRJ_PATH=`pwd`
export JOBS=`cat /proc/cpuinfo | grep processor | wc -l`
export CROSS_COMPILE=$HOME/toolchain/gcc-arm-10.3-2021.07-x86_64-aarch64-none-linux-gnu/bin/aarch64-none-linux-gnu-
export SRCS="imx-atf uboot-imx imx-mkimage"
MKIMG_BIN_PATH=$PRJ_PATH/imx-mkimage/iMX93
export FMW_PATH=firmware
set -e
function fetch_firmware()
{
	cd $PRJ_PATH

	mkdir -p $FMW_PATH && cd $FMW_PATH
	if [ ! -d firmware-imx-8.23 ] ; then
	   wget https://www.nxp.com.cn/lgfiles/NMG/MAD/YOCTO/firmware-imx-8.23.bin
	   bash firmware-imx-8.23.bin --auto-accept > /dev/null 2>&1
	fi

	if [ ! -d firmware-sentinel-0.11 ] ; then
	   wget https://www.nxp.com/lgfiles/NMG/MAD/YOCTO/firmware-sentinel-0.11.bin
	   bash firmware-sentinel-0.11.bin --auto-accept > /dev/null 2>&1
	fi
	if [ ! -d firmware-upower-1.3.1 ] ; then
	   wget https://www.nxp.com/lgfiles/NMG/MAD/YOCTO/firmware-upower-1.3.1.bin
	   bash firmware-upower-1.3.1.bin --auto-accept > /dev/null 2>&1
	fi
	if [ ! -d meta-maaxboard ] ; then
	   git clone https://github.com/Avnet/meta-maaxboard.git -b nanbield
```

```
    fi
    rm -f *.bin

}
function build_atf()
{
  SRC=imx-atf
  if [ ! -d $SRC ] ; then
    git clone https://github.com/Avnet/$SRC.git -b maaxboard_lf-6.6.3-1.0.0
  fi
  cd $SRC
  make -j${JOBS} CROSS_COMPILE=${CROSS_COMPILE} PLAT=imx93 bl31
  set -x
  cp build/imx93/release/bl31.bin $MKIMG_BIN_PATH
  set +x
  cd $PRJ_PATH
}

function build_uboot()
{
  SRC=uboot-imx

  if [ ! -d $SRC ] ; then
    git clone https://github.com/Avnet/$SRC.git -b maaxboard_lf-6.6.3-1.0.0
  fi

  cd $PRJ_PATH/${SRC}

  if [ ! -f .config ] ; then
    make ARCH=arm maaxboard-osm93_defconfig
  fi
  make -j${JOBS} CROSS_COMPILE=${CROSS_COMPILE} ARCH=arm

  set -x
  cp u-boot.bin $MKIMG_BIN_PATH
  cp u-boot-nodtb.bin $MKIMG_BIN_PATH
  cp spl/u-boot-spl.bin $MKIMG_BIN_PATH
  cp arch/arm/dts/maaxboard-osm93.dtb $MKIMG_BIN_PATH/$IMXBOOT_DTB
```

```
    cp tools/mkimage $MKIMG_BIN_PATH/mkimage_uboot
    set +x
}
function build_imxboot()
{
    SRC=imx-mkimage

    if [ ! -d $SRC ] ; then
        git clone https://github.com/Avnet/$SRC.git -b maaxboard_lf-6.6.3-1.0.0
    fi

    cd $PRJ_PATH
    # copy firmware
    cp $FMW_PATH/firmware-imx-*/firmware/ddr/synopsys/lpddr4_[id]mem_[12]d*.bin
$MKIMG_BIN_PATH
    cp $FMW_PATH/firmware-sentinel-*/mx93a1-ahab-container.img $MKIMG_BIN_PATH

    cd $PRJ_PATH/${SRC}
    # generate bootloader image
    make SOC=iMX93 REV=A1 flash_singleboot

    cp $MKIMG_BIN_PATH/flash.bin u-boot-maaxboard-osm93.imx
    chmod a+x u-boot-maaxboard-osm93.imx
    # copy bootloader image out
    cp u-boot-maaxboard-osm93.imx $PRJ_PATH
}
fetch_firmware
build_atf
build_uboot
build_imxboot
```

Execute the script to build:

```
$ ./make_mxosm93_uboot.sh
$ ls -t
u-boot-maaxboard-osm93.imx  uboot-imx  make_mxosm93_uboot.sh  imx-mkimage  firmware
imx-atf  meta-maaxboard  firmware-upower-1.3.0  firmware-sentinel-0.11  firmware-imx-8.23
```

The boot image for Maaxboard OSM93 is u-boot-maaxboard-OSM93.imx in the current directory.

2.3 Build Kernel in a standalone environment

Get the Linux source code

```
$ git clone https://github.com/Avnet/linux-imx.git -b maaxboard_lf-6.6.3-1.0.0
```

Check that the environment variables are correctly set :

```
$ echo $CROSS_COMPILE $ARCH
```

Build the kernel sources

```
$ cd linux-imx
$ make distclean
$ make maaxboard-osm93_defconfig
$ make -j4
```

Execute the ‘ls’ command to view the Image and dtb files after compilation.

```
$ ls arch/arm64/boot/Image
$ ls arch/arm64/boot/dts/freescale/maaxboard*dtb
arch/arm64/boot/dts/freescale/maaxboard-osm93.dtb
```

Execute the following command to compile the kernel modules, and install the modules to rootfs in the current directory.

```
$ make modules
$ make modules_install INSTALL_MOD_PATH=./rootfs
```

Chapter 3 System power on and boot up

To program the generated new Bootloader and System image files into MaaXBoard OSM93's eMMC memory, or for guidance on power-up MaaXBoard OSM93, the boot-up process, and how to exercise the supported BSP features of MaaXBoard OSM93, please refer to ***MaaXBoard-OSM93-Linux-Yocto-UserManual***.

Chapter 4 Appendix

4.1 Hardware Documents

For the detail hardware introduction, please refer to ***MaaXBoard OSM93 Hardware user manual***.

4.2 Software Documents

MaaXBoard OSM93 supports Yocto Linux, for additional information, please refer to the following documents:

- ***MaaXBoard OSM93 Linux Yocto User Manual***
  - Describes how to boot up MaaXBoard OSM93 and aspects of the BSP functionality
- ***MaaXBoard OSM93 Linux Yocto Development Guide***
  - Detailed guidance on how to rebuild the Linux system image (This document)

4.3 Contact Information

- Product Webpage:

  https://www.avnet.com/wps/portal/us/products/avnet-boards/avnet-board-families/maaxboard/maaxboard-osm93/maaxboard-osm93-board-family