Table of Contents

Setup

In our example we are using a host PC running CERN CentOS 7 for x86_64 and a ZCU102 evaluation board with a Xilinx Zynq Ultrascale+ MPSoC (ZynqMP) or an ATLAS MUCTPI V1 with a Xilinx Zynq SoC (Zynq). The host PC and the ZynqMP or Zynq are connected in a private network. In this network the IP address of the host is 192.168.1.1 and the one of the ZynqMP or Zynq is 192.168.1.10 . In addition, the host PC is in the CERN public network and has full internet access.

1 Create a/aarch64 or armv7hl root file system

Cross install a root file system from scratch using dnf for both aarch64 and armv7hl.
In our case the filesystem will be located at `$ROOTFS_PATH=/home/ppapageo/NFS`.

1.1 Install dnf

```
sudo yum install dnf
```

1.2 Install qemu

Get the qemu static executable. Qemu has to exist in the same directory in respect to the host and target filesystems. In our case we copy it to the `/usr/local/bin` the host and target root file system.

```
export ROOTFS_PATH=/your/target/path

#aarch64
wget https://github.com/multiarch/qemu-user-static/releases/download/v4.0.0/qemu-aarch64-static
chmod +x qemu-aarch64-static
mkdir -p $ROOTFS_PATH/usr/local/bin
cp -a qemu-aarch64-static $ROOTFS_PATH/usr/local/bin
sudo cp -a qemu-aarch64-static /usr/local/bin

#armv7hl
wget https://github.com/multiarch/qemu-user-static/releases/download/v4.0.0/qemu-arm-static
chmod +x qemu-arm-static
mkdir -p $ROOTFS_PATH/usr/local/bin
cp -a qemu-arm-static $ROOTFS_PATH/usr/local/bin
sudo cp -a qemu-arm-static /usr/local/bin
```

We also need inform the binfmt_misc to use the static qemu:

```
#aarch64
sudo vim /etc/binfmt.d/qemu-aarch64.conf
##clear the file and add the following line:
:qemu-aarch64:M::\x7fELF\x02\x01\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x02\x00\xb7\x00:\xff\xff

#armv7hl
sudo vim /etc/binfmt.d/qemu-arm.conf
##clear the file and add the following line:
:qemu-arm:M::\x7fELF\x01\x01\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x02\x00\x28\x00:\xff\xff\xff
```

Restart and verify:

```
sudo systemctl restart systemd-binfmt.service

#aarch64
sudo cat /proc/sys/fs/binfmt_misc/qemu-aarch64

#armv7hl
sudo cat /proc/sys/fs/binfmt_misc/qemu-arm
```

1.3 Install the root file system

Download and run the mkrootfs python script.
Add any extra packages in a file and pass it to the `--extra` option. (e.g. `extra_rpms.txt`)

```
git clone ssh://git@gitlab.cern.ch:7999/system-on-chip/centos-rootfs.git
cd centos-rootfs

#aarch64
sudo python mkrootfs.py --root=$ROOTFS_PATH --arch=aarch64 --extra=extra_rpms.txt

#armv7hl
sudo python mkrootfs.py --root=$ROOTFS_PATH --arch=armv7hl --extra=extra_rpms.txt
```

Special thanks to Matthias Wittgen, SLAC, who authored the original version of this script.

2 Boot the ZynqMP with the/aarch64 root file system

2.1 Boot the ZynqMP

Make sure you have configured the network to accept NFS through the firewall.
Prepare the SD card (we are using yocto for the preparation of the boot files).
In the uEnv.txt file on the SD card, add the nfs boot options in the bootargs; change the IP address and root filesystem path accordingly:

```
bootargs=earlycon clk_ignore_unused root=/dev/nfs rootfstype=nfs ip=192.168.1.10:::255.255.255.0:
```

2.2 Set up an ntp server

In order for the clocks of the host PC and the ZynqMP to be synchronized, an ntp server and client setup is needed. We use a chrony daemon.

2.3 Install additional packages using dnf

Additional packages can be installed with dnf from repositories specified in the .conf file.
For example:

```
cd centos-rootfs

#aarch64
dnf -y -c dnf.conf --forcearch=aarch64 \
        --releasever=7 \
        --repo=centos-base,centos-updates,centos-extras,arm64-epel \
        --installroot=$ROOTFS_PATH \
        install ${your_package}

#armv7hl
dnf -y -c dnf.conf --forcearch=armv7hl \
        --releasever=7 \
        --repo=centos-base,centos-updates,centos-extras,arm-epel \
        --installroot=$ROOTFS_PATH \
        install ${your_package}
```

When running dnf update and some packages fail to install due to scriptlet failures try the same command with the option `--setopt=tsflags=noscripts` .

3 Build and install a specific version of GCC

CentOS usually comes with a version of gcc which might be outdated for other software that you may want to build. As an example, we are building gcc version 8.2 as required by ROOT and the ATLAS TDAQ software.

```
wget https://ftp.gnu.org/gnu/gcc/gcc-8.2.0/gcc-8.2.0.tar.xz
tar -xJf gcc-8.2.0.tar.xz
cd gcc-8.2.0
mkdir build
cd build
../configure
make -j $(($(nproc) + 1))
make install -j $(($(nproc) + 1))
```

4 Download and compile ROOT

Install external dependencies: `cmake3 libX11-devel libXext-devel libXpm-devel libXft-devel redhat-lsb-core python-devel`

```
git clone https://github.com/root-project/root.git ${PREFIX}/ROOT
```

Install a newer version of GCC, e.g. GCC 8.2 (see above).
Root tries to download some packages from the internet. If your board has not internet access (you can give access the board by following this guide), you will have to download them manually from the host side. The target urls are usually located at `${package}/src/${package}-stamp/${package}-urlinfo.txt/-gitinfo.txt`. In this case:

```
export PREFIX=/root/installation/path
wget http://lcgpackages.web.cern.ch/lcgpackages/tarFiles/sources/vdt-0.4.2.tar.gz -P ${PREFIX}/RO
wget http://lcgpackages.web.cern.ch/lcgpackages/tarFiles/sources/openssl-1.0.2o.tar.gz -P ${PREFI
wget http://lcgpackages.web.cern.ch/lcgpackages/tarFiles/sources/tbb2019_U1.tar.gz -P ${PREFIX}/R
```

Comment out the downloading section on the .cmake file:

```
cd ${PREFIX}/ROOT/build/interpreter/llvm/src/tools/cling/tools/plugins/clad/clad-prefix/tmp/
vi clad-gitclone.cmake
git clone https://github.com/vgvassilev/clad.git ${PREFIX}/ROOT/build/interpreter/llvm/src/tools/
```

Now on the ZynqMP side:

```
export CXX=/usr/local/bin/g++
export CC=/usr/local/bin/gcc
export LIBRARY_PATH=/usr/lib64:${LIBRARY_PATH} #the libstdc++ of the locally built gcc is locate
cd build
cmake3 ../
make -j $(($(nproc) + 1))
```

Then you can source and use ROOT:

```
source ${PREFIX}/ROOT/build/bin/thisroot.sh
```

5 Build ATLAS TDAQ

As an example of a more complex software we are building a subset of the ATLAS TDAQ software. The instructions of the process can give you a general idea on how the building can be achieved. For the actual software, access to the ATLAS TDAQ repositories will be required.

6 Cross Compile with sysroot

CrossCompileWithCentOS

7 Miscellaneous

Visit CentOSForZynqMPMisc for hints and tips concerning system administration.

-- PanagiotisPapageorgiou - 2019-05-17

This topic: SystemOnChip > CentOSForZynqMP
Topic revision: r30 - 2019-07-24 - RalfSpiwoks