Instructions with comments about how to get a working EMP emulator

Important information

- The power voltage for the EMP test board is 12 volts.
- When Almalinux is installed, don’t power off the EMP test board by only pushing the power off button; there is an Almalinux operating system running in the board, so tunr it off in that way may cause errors in the OS. Instead use commands like:
  - systemctl poweroff

Step 1: Create, configure and build the Petalinux project

Petalinux is a tool provided by AMD Xilinix for creating and deploying embedded Linux solutions on AMD processing systems. In this project, it is used to create bootables for the processing system of the MPSoC module, which contain the kernel configuration, kernel modules, U-Boot configuration and the Device Tree for the respective hardware platform.

In order to boot the PS of the Zynq into the CentosStream 8 based epos (EmP Operating System) root file system, it requires two things:

1. A FAT32 partition on a SD card with a minimum size of 1 GiB for Petalinux boot files
2. A NFS or a second (EXT4) partition on the SD card for the root filesystem (see Setup EMP development environment & reference project)

From this point, please follow the instructions in the following document:
https://docs.google.com/document/d/1PANLbLp4xcehAxknSIsHNosREiN9XPTcb73Bz7ummsU/edit#heading=h.z8zwc2e3eegp

Some caveats about the documentation:

1. In section “EXT4 (SD/EMMC/SATA/USB):
    1. The path to the root file must be /dev/mmcblk1p2, instead of the one shown in the image; the difference is the 1 instead of 0 in the path.
2. In section “Boot arguments”:
    1. The string to enter here is:

        ```
        setenv bootargs earlycon console=ttyPS0,115200 clk_ignore_unused
        root=/dev/mmcblk1p2 ip=dhcp rw uio_pdrv_genirq.of_id=generic-uio
        ```

        The previous command is setting the Petalinux kernel boot. Because of that, the argument "console" means the serial port ttyPS0 in the SoM; not in your PC. In the same way, the value for argument "root" is a path inside the SoM LINUX, not in your own PC.

Step 2: Build your EPOS root filesystem for EMP

The instructions in this section heavily rely on the following sources:

- https://docs.google.com/document/d/1PANLbLp4xcehAxknSIsHNosREiN9XPTcb73Bz7ummsU/edit#heading=h.z8zwc2e3eegp
- https://gitlab.cern.ch/emci-emp/epos/-/tree/master?ref_type=heads

In particular, the Gitlab’s instructions are followed in great detail.

Comments about Gitlab’s instructions:

1. In section “Creating epos root filesystem for EMP” (link: https://gitlab.cern.ch/emci-emp/epos/-/tree/master/epos-rootfs), there are some commands that ask you to modify a file in your computer; the commands are:

    ```
    1. sudo nano -w /etc/binfmt.d/qemu-aarch64.conf
    2. :qemu-aarch64:M::\x7fELF\x02\x01\x01\x00\x00\x00\x00\x00\x00\x00\x00\
       x00\x02\x00\xb7\x00:\xff\xff\xff\xff\xff\xff\xff\x00\xff\xff\xff\xff\
       xff\xff\xff\xff\xfe\xff\xff\xff:/usr/local/bin/qemu-aarch64-static:
    3. $ sudo systemctl restart systemd-binfmt.service
    ```
2. The explanation about what are doing those commands is
    1. When you modify file “/etc/binfmt.d/qemu-aarch64.conf” in your PC, you are asking QEMU to emulate the architecture Aarch64.

        1. QEMU stands for Quick EMUlator, and it is a versatile tool that can perform both system-level emulation and user-level emulation.
    2. This is useful, because for building the Operative System and load them in the SD card, some scripts and tools written for Aarch64 must be runned in your machine, so this change will allow to execute them; the problem that your machine has a different architecture than Aarch64 is eliminated with this modification.
    3. The line that starts with ":qemu-aarch64:M::" is the line that especifically tells Linux to use QEMU to run binaries with architecture Aarch64.
2. In the same previous section, the python command must be executed in the following path:

```
1. <your_project_dir>/epos/epos-rootfs/
```

3. WATCH OUT: If you are running the installation in a Ubuntu machine, you must install the dnf package manager (which is originally for Fedora or RedHat). In order to make that, you must follow the commands:

```
1. sudo apt update
2. sudo apt upgrade
3. sudo apt install dnf
```

Step 3: Connect to EMP test board through ssh

The EMP/EMCI team already registered our EMP test board in the CERN devices portal; it is registered under Renzo Barraza's name. Because of that, it is not neccessary to connect to the board through USB cable, as if the EMP test board is connected to Ethernet through RJ-45 cable, you can access it from any where using the command ssh.

If you want to find the data to connect to the Trenz test board, follow the next steps:

1. Go to the following link: https://network.cern.ch/sc/fcgi/sc.fcgi?Action=SelectForDisplay
2. In the form, look the field associated with "Search by name of the responsible, main user or LanDB manager", and in the field "SURNAME" put: "Barraza Altamirano"
    1. If you only put "Barraza" it won't work.
3. Push search button, and in the resulting list, select the device called "TRENZ-TGC - TRENZ BASEBOARD ZU4EG LINUX "

However, the only way to evaluate if the OS is correctly loaded is using the USB connection. For that, follow the next instructions:

1. Plug your SD card with the AlmaLinux loadded in it.
2. Connect your USB cable to the board TE0790-03 board that is closer to the main Trenz SoM; the other part of the cable must be connected to your computer.
3. Power the EMP text board with 12 V DC.
4. Open the Putty program and follow the settings recommended in the Google Doc instructions; if communication fails, try with the second serial port
5. Push the button closer to the TE0790-03 board were you connected your USB cable.
    1. You will know that the button was correctly pushed because the board's fan must start.

After those steps, the starting of Almalinux should appear in your putty screen. After the starting ends, the Almalinux last messages should be the following:

```
AlmaLinux 9.4 (Shamrock Pampas Cat)
Kernel 6.1.30-xilinx-v2023.2 on an aarch64

localhost login:
```

To enter the system, you must use the following credentials:

- User: root
- Password: EMPaina

After that, you will be insede the machine with root privileges. It is highly recommended to create your own user. Just as an example, here I present a possible user to create; the user is called "tgc_cms" and its password is "saphir":

```
useradd -m tgc_cms
echo "tgc_cms:saphir" | chpasswd
```

Additionally, you must give sudo privileges to your user, for that use the following commands (assuming your created user is called tgc_cms):

```
sudo usermod -aG wheel tgc_cms
```

Now that you have your user set, you can use the command “ifconfig” to know under which direction you can connect to the Board through Ethernet.

As the EMP test board is registered at CERN, you can access to ir by ssh with the following command in your machine connected to CERN network (in this command we assume that the user to which you are trying to connect to is “tgc_cms”.

```
ssh tgc_cms@trenz-tgc
```

Alternatively, you can use:

```
ssh tgc_cms@128.141.43.81
```

Now you can check if your user realy have sudo privileges. For that, enter into your user using ssh and in console type: “sudo whoami”; it should return “root”.

Step 4: Network connection check

In the following steps, it is a must for the Trenz test board to have access to Internet. However, sometimes the network connection is not automatically set during EPOS installation. To check that network connection is correctly set, follow these steps:

1. Check if you have access to internet by using the command:
    1. `ping -c 4 www.google.com`
        1. This command will try to reach the site www.google.com 4 times. If that connection fails, you know that you have no internet access.
2. If you have internet access, you are ready and can skip the following steps in this section. But if you don’t have internet access, execute the following command:
    1. `nmcli con show eth0`
        1. In the returned message, look for the value of “ipv4.method”; if it is “auto”, you are good and can skip the following steps in this section, if it is “manual” you must follow the next steps.
3. Use the following command to change the previous value from “manual” to “auto”
    1. `sudo nmcli con mod eth0 ipv4.method auto`
4. Restart the network manager by using the command:
    1. `sudo systemctl restart NetworkManager`

Step 5: Load neccesary libraries

Check that –reseaserver is correctly detected

First, you must install git in your EMP test board. For that, use the following command:

```
sudo dnf install git
```

If you find a problem trying to use this command, check if the error message says something like this:

```
        [tgc_cms@trenz-tgc ~]$ sudo dnf install git
Unable to detect release version (use '--releasever' to specify release version)
AlmaLinux $releasever - CERN
487 kB/s |  26 kB     00:00
AlmaLinux $releasever - AppStream
11 MB/s | 9.8 MB     00:00
AlmaLinux $releasever - BaseOS
12 MB/s | 8.1 MB     00:00
AlmaLinux $releasever - CRB
8.7 MB/s | 2.2 MB     00:00
AlmaLinux $releasever - Extras
308 kB/s |  20 kB     00:00
Extra Packages for Enterprise Linux $releasever - aarch64
755 kB/s | 8.8 kB     00:00
Errors during downloading metadata for repository 'epel':
  - Status code: 404 for http://linuxsoft.cern.ch//epel/$releasever/Everything/aarch64/repodata/repomd.xml (IP:
188.185.29.10)
Error: Failed to download metadata for repo 'epel': Cannot download repomd.xml: Cannot download repodata/repomd.xml:
All mirrors were tried
```

As you can see in the previous error, the problem is that Almalinux is not replacing “$releaserver” by the release number. Because of that, you must explicitly specify the version you want to install. Use the option “--releaserver 9” at the end of the previous command, that is:

```
sudo dnf install git --releasever 9
```

Install libraries libuio and boost-program-options

To be able to run binaries that are necessary for the clock setting, you must install some libraries before. To install library libuio, follow the instructions in section “libuio - UserspaceIO helper library”.

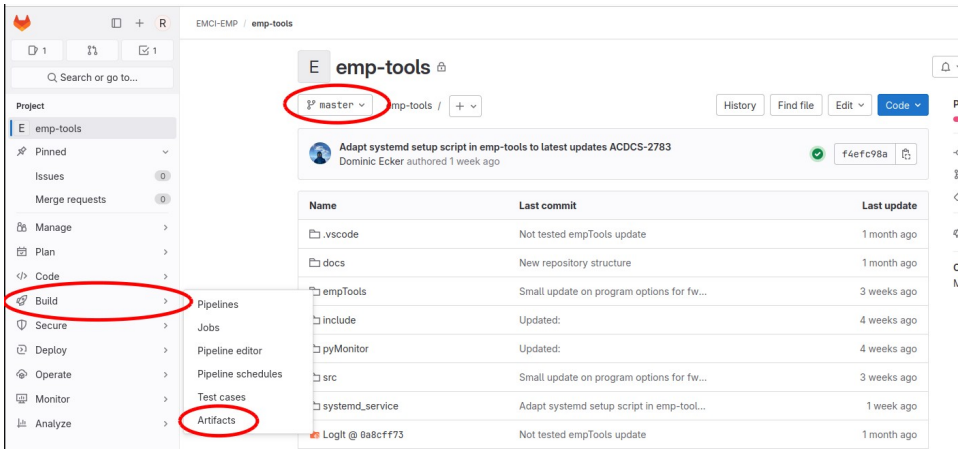
To install library “boost-program-options”, use the command:

```
sudo yum install boost-program-options
```

Additionally, if in the following steps there are libraries missing, use “sudo yum install <MISSING_LIB_NAME>”, or “sudo dnf install <MISSING_LIB_NAME>” commands to get them.

Step 6: Setting the clocks

In this state, you already can load a firmware to your board, but probably it won’t run, as the clocks neccessary are not confurated. There is an executable that will set the clock, and you must run it every time you start the board. However, in the following command we will use that script to be automatiacally start the clocks every time the EMP text board is booted.

1. In the Giltab EMP_tools, go to branch the master branch
2. In the left side column, select the option “Artifacts”, which is under menu “Build”. An image of it can be seen here:
    1. 

3. After ckicling on “Artifacts”, in the displayed page, click in the drop-down menu related with the lastest commit.
4. In this drop-down menu, you should see at least the following files:
    1. artifacts.zip
    2. metadata.zip

3. job.log

5. Download artifacts.zip.
6. Unzip the downloaded file, and copy its resulting folder into your EMP test board; you can use the command “scp”.
    1. You can place the folder in every path of your EMP test board.

Now that you have the files in your EMP test board, apply the following commands to always start the clock when booting your EMP test board:

1. Clone the git repo: https://gitlab.cern.ch/emci-emp/emp-tools/-/tree/master?ref_type=heads
    1. Indeed, the used part of the repo is just the directory “systemd_service”, but to avoid complications, the entire repository was copied.
2. In the directory generated when unzip artifacts.zip, and that was copied to your EMP test board, you should find 2 subdirectories: bin_EMP and bin_TRENZ. For the board present at SAPHIR’s lab at 2024/07/03, our file of interest is inside directory bin_TRENZ; that is because the board we have is not the EMP, but a test board developed around a Trenz board.
3. Apply the executable file into path “/usr/bin/”. For that, use the following command:

```
1. sudo cp ./bin_TRENZ/clkProgrammer /usr/bin/clkProgrammer
```

4. Enable permissions:

```
1. sudo chmod +x /usr/bin/clkProgrammer
```

5. Go to the cloned repository at step 1, in particular to the path:

```
1. ./emp-tools/systemd_service
```

6. Copy the file “systemd-programclock.service” into necessary paths:

```
1. sudo cp systemd-programclock.service /lib/systemd/system/systemd-
   programclock.service
2. sudo cp systemd-programclock.service /etc/systemd/system/systemd-
   programclock.service
```

7. Enable permissions for copied file:

```
1. sudo chmod 644 /etc/systemd/system/systemd-programclock.service
```

8. Set the automatic clock when booting the system:

```
1. sudo systemctl start systemd-programclock
2. sudo systemctl enable systemd-programclock
3. sudo systemctl status systemd-programclock
```

In the following, you can always use the command “sudo systemctl status systemd-programclock” yo check the clock status, and if it is running.

Step 7: Load firmware to Trenz board

Step 7.1: Build the example project in your computer

To load the Trenz test board's firmware, you must follow the instructions in section "Prepare the Firmware to be deployed" from the guide in: https://docs.google.com/document/d/1PANLbLp4xcehAxknSIsHNosREiN9XPTcb73Bz7ummsU/edit .

The Gitlab repository on which you will be working on is: https://gitlab.cern.ch/emci-emp/emp-firmware

However, there are outdated instructions in that guide, they are:

1. In step 2, instead of sourcing script "emp_assembly.tcl" you must source the script "emp-firmware-2023.2.tcl".
2. Something that is not mentioned in the Google Doc's instructions, but that it is mentioned in the EMP-firmware repo, is that you must update some polarities in a module depending on your board version. Follow the instructions in the emp-firmware gitlab to do so.
    1. For the Trenz test board we had in the lab at 2024/07/01, this changes means opening the modeule "transceivers_TE0807$_0$", and change the following fields to the following values:
        1. MGT RX Polarity Inv: 0xFFF
        1. MGT TX Polarity Inv: 0x000

Step 7.2: Generate firmware files and load them into your Trenz Board.

The way of loading a firmware to the EMP test board is by transferring generate the bitstream of your Vivado project, process it uwing a python script given by EMP/EMCI team, and transferring the resulting files (.dtbo file mainly) to the directory /lib/firmware/ inside the EMP test board Linux system.

For that Python script to work, first you must have availablo tools that are provided by Vivado and by Petalinux. To load the tools provided by Petalinux you must, in your computer (not in the EMP test board):

1. Load Petalinux by executing this command:

   ```
   1. source <YOUR_PETALINUX_DIR>/2023.2/settings.sh
   ```
2. Execute the following command:

   ```
   1. export PATH=$PATH:<YOUR_PETALINUX_DIR>/tools/xsct/bin/
   ```

Now that Petalinux was set, you must enable a tool present at Vivado. For this you have 2 options: one is to source the file “settings64.sh” that is inside Vivado install directory, the second is to install the tool dtc in your EMP test board; we will use this later one by applying the command:

- `sudo dnf install dtc`

Now that we have the needed packages, let’s clone the repository where it is the python script that will create the necessary file received by EMP test board from your binary file generated by Vivado; clone the following repository in your PC (not in the EMP test board):

- https://gitlab.cern.ch/emci-emp/epos/-/tree/master/etools/xsa-to-overlays/xsa-to-overlays

After this step, open the Vivado 2023.2 project where your design is, and export it; include the bitstream in your exportation. After that, the produced bitstream file (file .xsa) must be put in the same directory that tha python script “emp-xsa-to-overlays.py”. Then run the following command to get the output files from the python script:

- `./emp-xsa-to-overlays.py <YOUR_BITSTREAM_FILE>.xsa`

3 questions should appear in console; answer the 3 of them with “yes”. After that, a new directory called “output” must have been created. Go inside that directory, and copy the files .dtbo & .bit.bin into this directory of your Trenz test board:

- `/lib/firmware/`

Then go to your home directory in your Trenz test board, and clone the repository etools; this repository:

- `https://gitlab.cern.ch/emci-emp/epos/-/tree/master/etools?ref_type=heads`

After cloning it, inside of it, you will find the script “loadFirmare.sh”.

To activate the firmware you just loaded into directory lib/firmware/ , you must execute the script loadFirmare.sh, but giving as its argument the name of your .dtbo file. For example, if your file is called blah_blah.dtbo, you must run the following command to get the firmware running in your Trenx test board:

- `sudo ./loadFirmare.sh blah_blah.dtbo`

Sudo is needed.

Reference

- This document is heavily based on the following instructions created by EMP/EMCI team: https://docs.google.com/document/d/1PANLbLp4xcehAxknSIsHNosREiN9XPTcb73Bz7ummsU/edit