EMBEDDED LINUX

Init process

Brief
Now that we know how U-Boot loads the Linux kernel, let's see which process is started first and how we can change it.

Author
dimitri.boudier@ensicaen.fr
hugo.descoubes@ensicaen.fr

Summary
Analyze the first process and change it to a custom application.

Reminder
/!\ Understand all commands before running them! /!\

DEFAULT INIT PROCESS

Start the BBB, wait for Debian to be ready and log in. Look at the running processes.

```
ls /proc
ps -ef
```

- What does the `/proc` directory contain? What does the `ps` command do? What is the first process to be executed?

Default init application vector (symbolic link)

```
stat /sbin/init
```

- What is the `/sbin/init` file?

Display the main memory map of init process.

```
cat /proc/1/maps
```

- Those are the memory segments for the init process. What segments do you see?

As no customisation has been performed yet (U-boot, Linux, Debian), the distribution starts with the default init process. You can execute all the previous commands on a Ubuntu computer and you will have the same results.

CREATE A STATIC APPLICATION

You will develop a static application, with no need of a distribution whatsoever. First, let's build the application for your BBB but from your **computer**.

Make sure your `CC` environment variable is set: do you see your *standard* GCC or the *Linaro* GCC?.

```
${CC}gcc --version
```

Build three different executables from the very same C source file.

```
cd ${DISCOPATH}/apps
cat hello.c

gcc hello.c -o hello_x64
gcc -static hello.c -o hello_static_x64
${CC}gcc -static hello.c -o hello_static_arm
```

Execute each file, one at a time. They're echo programs, just try it.

```
./hello_x64
./hello_static_x64
./hello_static_arm
```

- Can you run each file?

Let's understand why you can't run the third one.

```
readelf -h hello_x64
readelf -h hello_static_x64
readelf -h hello_static_arm
```

- What does `readelf -h` do? What is an ELF file?

- What are the main differences between those files? So, what can't you run the third one (for now)?

Finally, you have to know what is the `-static` option passed to GCC. Look at the files size.

```
ls -l
```

- Can this help you to suggest what is the `-static` option passed to GCC? Why is it necessary for a BBB application?

- Compiling a static application is usually not useful when using a Linux-based distribution. But now we want to run a distro-less application. Why is building a static application necessary in this case?

BBB APPLICATION WITH NO DISTRO

As we want to run a distribution-less application, we will remove the Debian filesystem from the SD card.

Plug the SD card into your computer and erase the Debian filesystem in it.

```
lsblk -f
sudo rm -rf ${MEDIA}/rootfs/
```

Woopsie! By deleting the Debian filesystem, we also removed the kernel image and the device tree binary. Let's restore them.

```
cd ${DISCOPATH}
export kernel_version=4.14.198-bone-rt-r40

sudo mkdir -p ${MEDIA}/rootfs/boot/
sudo cp -v ./deploy/zImage ${MEDIA}/rootfs/boot/vmlinuz-${kernel_version}
sudo cp -v ./deploy/am335x-boneblack.dtb ${MEDIA}/rootfs/boot/
```

We also have to restore the `uEnv.txt` file. But first you will edit it so that the *bootargs* specify the name of the application we want to be executed first.

```
nano ./misc/uEnv.txt
```

> before : optargs=init=/lib/systemd/systemd
> now : optargs=init=/hello_static_arm

Now copy both the `uEnv.txt` file and the application into the SD card partition. Sync and unmount.

```
sudo cp -v ./misc/uEnv.txt ${MEDIA}/rootfs/
sudo cp -v ./apps/hello_static_arm ${MEDIA}/rootfs/
sync
sudo umount ${MEDIA}/rootfs
```

Now boot the BBB and analyze the U-Boot and kernels output during the booting phase.

- Which program is running? Is the shell running?

While keeping the power supply, remove the SD card from the BBB.

- What kind of message do you see? What is sending those messages?

- Is the application still running? Why's that?

BINGO - If you have a good understanding of this exercise ... you can now understand the difference between a kernel (Operating System) such as Linux (XNU, Hurd, Minix, etc) and a GNU\Linux distribution of services (applications and libraries) such as Ubuntu (Debian, Redhat, Fedora, etc).

RESTORE PREVIOUS WORK

If you want to perform some tests with this configuration, go on.

But if you want to perform some other tests on the BBB or go to the next chapter, you must deploy a full Debian distro onto the SD card. To do so, run your `disco/script/sdcard_deploy.sh`.