EMBEDDED LINUX

Visit of the Linux kernel tree

Brief
Fact: Linux is a monolithic modular kernel.
Questions: How do we choose which services should be built or not? How do we choose which services should be part of the kernel monolith, or which should be configured as modules?
Statement: We'll visit the Linux sources to understand how to build a custom kernel image.

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

Resources

- Kernel repository : https://www.kernel.org/
- Kernel map : http://www.makelinux.net/kernel_map/
- French kernel contributor (Docs) : https://bootlin.com/

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

LINUX KERNEL VERSIONNING

Go and check out the sources content.

```
cd ${DISCOPATH}/kernel/bb-kernel/KERNEL
ls
```

- What do you see here?

>

- Have you looked at hidden files?

>

Use the graphical Git repository browser to observe kernel evolutions and versions. Look at some commits by Robert C. Nelson.

```
gitk
```

- Why can't we see the history before v2.6.12-rc2?

>

BOOT FILES - Linux Kernel Image, Device Tree Binaries and first calling procedures

Observe Linux kernel compressed (zImage) and uncompressed (Image) images.

```
ls arch/arm/boot
```

- What is their size?

Observe the architecture-specific Linux entry point (`head.S` file and `_start:` label).

```
cat arch/arm/boot/compressed/head.S | grep -A 14 _start:
gedit arch/arm/boot/compressed/head.S &
```

- What seems to be done here? How can you tell this is an architecture-specific file?

Observe the generic Linux entry point (`main.c` and `start_kernel()` function).

```
cat init/main.c | grep -A 108 start_kernel
gedit init/main.c &
```

- What seems to be done here? Is it that different from a MCU `main()` function?

How can you tell this is an generic file? Have you noticed that the `head.S` and the `main.c` are not in the same folders at all?

Observe all of the default device tree sources.

```
ls arch/arm/boot/dts
```

- What are the `dts`, `dtsi` and `dtb` extensions?

You see that all ARM CPUs have their specific device tree sources here. As a matter of fact, the BBB CPU is an AM335x SoC device tree sources.

- What command can you use to filter out any file that does not contain `am335x`? Do you remember the device tree binary we regularly use?

CONFIGURATION FILE - .config file

Observe the Linux kernel `.config` configuration file. It is used by the several Makefiles on Linux files system tree to build the kernel image.

```
cat .config
```

- What do the `CONFIG_xxx` represent? What do `=y`, `=m` and `is not set` mean?

- For instance, what is the value of the `CONFIG_8250_SERIAL` field?

Now open the default x86 kernel configuration interface (with `menuconfig`, using the ncurse library).

```
make menuconfig
> Exit
> Do not save!
```

- What do `[*]`, `[ ]` and `<M>` mean? Is there any link with the previous question?

However, we should not open the x86 interface but the ARM kernel configuration interface!

```
make ARCH=arm menuconfig
```

As an example let's search the 8250 on the menuconfig interface. The 8250 is a historic UART peripheral on Intel computer technology. In the menuconfig interface, type

```
/
8250_OMAP
[Enter]
```

- Write down the first result. **We'll need it later**.

>

- Briefly explain each field of the first result.

>

The symbol `SERIAL_8250_OMAP` is defined at *see previous answer*. Open this file or display its content.

```
cat drivers/tty/serial/8250/Kconfig
cat drivers/tty/serial/8250/Kconfig | grep -A 15 8250_OMAP
```

- What is the purpose of a `Kconfig` file?

>

Let's make things clear for now.

- What is the link between the `Kconfig` file and the `make menuconfig` interface?

>

- What is the link between the `.config` file and the `make menuconfig` interface? Why should you use the latter instead of just editing the `.config` file?

>

BUILDING FILES - Makefile

Still using the 8250 UART as an example, observe all the 8250 drivers technologies supported by Linux.

```
ls -l drivers/tty/serial/8250
```

And now just the ones supported by TI OMAP SoC solutions.

```
ls -l drivers/tty/serial/8250 | grep omap
```

Finally, observe the local Makefile.

```
cat drivers/tty/serial/8250/Makefile
```

- What is the use of each line in a `.config` file (note that they all start with `CONFIG_`)? How does their value (`=y`, `=m`, `is not set`) impact the compilation process?

>

You might have noticed that many folders in the kernel sources contain their own Makefile (and their own Kconfig too). All sub-Makefiles are all called by a "master" Makefile at the root folder.

```
cat Makefile
```

LINKING FILES - Linker Scripts and Memory Map

List all the kernel linker scripts for ARM architecture.

```
find . -iname *.lds | grep arm
```

Observe the content of this specific linker script.

```
cat arch/arm/boot/compressed/vmlinux.lds
```

- Do you remember the what is the purpose of a linker script? Describe this one.

Observe the kernel physical memory map.

```
cat System.map
```

CONCLUSION

The aim of this document is to comprehend how services can be built into the kernel or built as external modules.

- Write here what you understood. You shall talk about `.config`, `Kconfig` and `Makefile` files.