Chapter 1

Utilisation basique de Buildroot

Objectifs:

- Télécharger Buildroot
- Configurer un système minimal avec Buildroot pour la Raspberry Pi 3
- Lancer la compilation
- Flasher et tester le système généré

1.1 Setup

Comme spécifié dans la documentation Buildroot[1], vous avez besoin de certains paquets/dépendances :

```
sudo apt install sed make binutils gcc g++ bash patch \
  gzip bzip2 perl tar cpio python unzip rsync wget libncurses-dev
```

Lisez les spécificités de votre carte RPi3 sur différents sites notamment le site de la Raspberry Fondation

1.2 Télécharger Buildroot

Comme nous allons réaliser du développement Buildroot, on va télécharger les sources depuis le répertoire Git :

```
git clone git://git.busybox.net/buildroot
```

Aller dans le nouveau dossier `buildroot` créé.
Nous allons utiliser la branche *2019.08* de cette release.

```
git checkout 2019.08
```

[1] https://buildroot.org/downloads/manual/manual.html#requirement-mandatory

1.3 Configuration de Buildroot

Si vous regardez le dossier `configs/`, vous verrez qu'il y a un fichier `raspberrypi3_defconfig`, qui est une configuration pour construire un système pour une Raspberry Pi 3. Faites donc un:

```
make raspberrypi3_defconfig
```

Mais comme on souhaite apprendre Buildroot, on va regarder la configuration un peu plus en détails. Démarrez l'utilitaire de configuration Buildroot :

```
make menuconfig
```

Il est aussi possible d'utiliser les autres outils comme `nconfig`, `xconfig` ou `gconfig`.
Maintenant, regardons les points importants Buildroot:

- Menu `Target Options`
  - `Target Architecture` : `ARM (little endian)`
  - `Target Architecture Variant` : D'après le site web de la RPi3, ils utilisent un CPU Broadcom BCM2837 qui est basé sur ARM `Cortex-A53`
- Menu `Build options` : Regardez notamment le lieu où sauver la configuration Buildroot
- Menu `Toolchain`
  - Par défaut, Buildroot compile sa propre toolchain. Ca prend un peu de temps même si pour `ARMv8` il y a une toolchain pré-buildée fournie par ARM.
- Menu `Kernel`
  - Par défaut, le kernel le plus récent dans la release Buildroot est utilisé. Ici, il s'agit d'un kernel spécifique, sur github d'où le `Custom Tarballs`
  - Faites attention à la configuration kernel utilisé `Using an in-tree defconfig file` avec `bcm2709` comme configuration. Correspond au ficher `bcm2709_defconfig` dans le kernel: https://github.com/raspberrypi/linux/blob/rpi-4.19.y/arch/arm/configs/bcm2709_defconfig
  - Sur ARM, beaucoup de plateformes utilisent un *Device Tree* pour décrire le hardware. Vous pouvez regarder son DT : https://github.com/raspberrypi/linux/blob/rpi-4.19.y/arch/arm/boot/dts/bcm2710-rpi-3-b.dts,
- Menu `Target packages`. Surement le plus important car c'est le menu qui permet d'installer les paquets souhaités dans le rootfs final. On verra dans un autre lab un exemple.
- Menu `Filesystem images`. Pour savoir quel type d'image on veut générer.
- Menu `Bootloaders`
  - Le plus populaire pour ARM est *U-Boot*. Pour la RPi3 n'a pas besoin d'un bootloader pour executer et charger le kernel donc pas besoin!

1.4 Compilation

Simplement lancez `make` ou redirigez la sortie avec la commande `tee`:

```
make 2>&1 | tee build.log
```


1.5 Flasher la carte SD

Une fois la compilation terminée, on va pouvoir flasher notre système sur notre carte SD. Mais avant, encore faut-il l'identifier ! Regardez la sortie de `cat /proc/partitions` et chercher votre carte SD. En géneral, un lecteur de carte SD donnera `mmcblk0`, alors qu'un lecteur externe USB, ça sera du type `sdX` (i.e`sdb`, `sdc`, etc.). **Attention : `/dev/sda` est généralement le disque dur de votre laptop!**

Si votre carte SD est `/dev/mmcblk0`, les partitions seront `/dev/mmcblk0p1`, `/dev/mmcblk0p2`, etc. Si c'est `/dev/sdc`, les partitions seront `/dev/sdc1`, `/dev/sdc2`, etc.

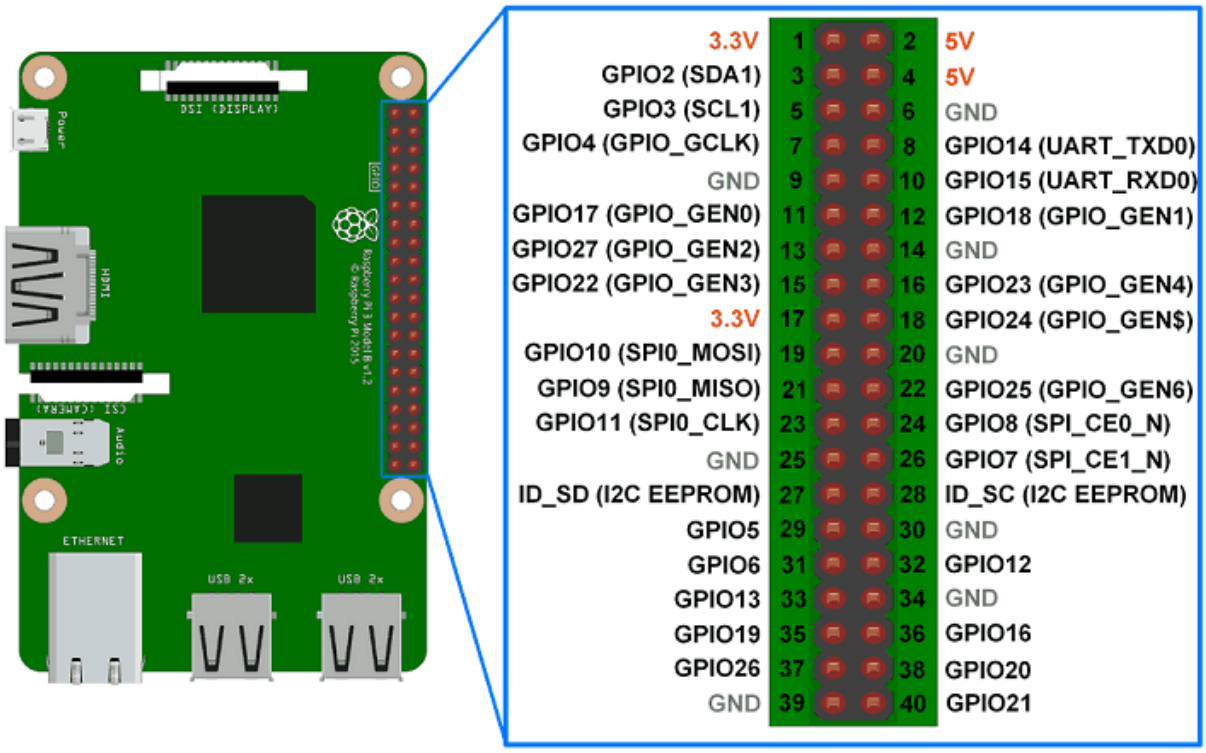
Pour flasher votre système, suivez la commande indiquée dans le `README` de la RPi3 : https://git.buildroot.net/buildroot/tree/board/raspberrypi/readme.txt?h=2019.08:

```
sudo dd if=output/images/sdcard.img of=/dev/mmcblk0 bs=1M
```

Ou `sdc` ou `sdb` au lieu de `mmcblk0` si besoin.

1.6 Démarrage du système

Insérez la carte SD dans la RPi3.
Branchez la série selon les pins de votre RPi3 :



Allumez via USB votre board et vous devriez voir le système booter !

Le login est `root` et profitez pour explorer le système. Lancer `ps` pour regarder combien de processus sont en cours d'exécution, qu'est-ce que Buildroot a généré dans `/bin`, `/lib`, `/usr` et `/etc`.

1.7 Explorer le log de build

De retour sur votre machine de build, regardez la sortie du log `build.log`. Buildroot est un peu verbose mais il affichera chaque message important d'un prefixe `>>>`. Pour avoir une idée générale de ce qui a été fait, vous pouvez lancer:

```
grep ">>>" build.log
```

Vous voyez les différents paquets être téléchargés, extraits, patchés, configurés compilés et installés.

