

## Partie I - Chaîne de compilation

Christophe BLAESS

Avril 2010

*La petite carte IGEPv2 disponible depuis quelques mois chez ISEE (voir <http://www.igep-platform.com/>) est une plateforme performante, économique et très complète pour s'initier au développement Linux embarqué. Il existe déjà plusieurs distributions spécifiques pour cette carte, mais nous étudions dans cette série d'article comment construire son propre système en partant de zéro..."*

### Téléchargement et installation des sources

Dans ce premier article nous allons examiner la création d'une *cross-toolchain* (chaîne de compilation croisée) permettant de travailler sur un PC pour produire du code sur la carte cible.

Cette étape est primordiale à tout développement sur la carte. Il faut d'abord s'assurer que le processeur de la cible soit supporté par le compilateur GCC. C'est le cas, car le processeur *Texas Instruments OMAP 3* est bâti autour d'un coeur Arm classique (Cortex A8).

Plutôt que de nous lancer dans une compilation fastidieuse de GCC avec de nombreuses lignes de commandes rébarbatives, je vous propose de faire appel à un outil sensiblement plus convivial : **buildroot**.

Cet utilitaire permet en théorie de construire un système Linux complet, prêt à *booter*, à partir de fichiers de configuration prédéfinis. Toutefois nous limiterons ici son usage à la génération d'une *toolchain* adaptée à notre processeur.

Commençons par télécharger une version récente de **buildroot** :

```
$ mkdir Projet_Igep
$ cd Projet_Igep/
$ wget http://buildroot.uclibc.org/downloads/buildroot-2010.02.tar.bz2
[...]
Longueur: 3640302 (3,5M) [application/x-tar]
Sauvegarde en : «buildroot-2010.02.tar.bz2»
100%[=====]
3 640 302612K/s ds 7,0s
(510 KB/s) - «buildroot-2010.02.tar.bz2» sauvegardé
$
```

Et décompressons le fichier :

```
$ tar -xjf buildroot-2010.02.tar.bz2
$ cd buildroot-2010.02
```

## Configuration et compilation de *Buildroot*

*Buildroot* possède de nombreuses options de compilation, et – comme c'est l'usage dans le domaine de Linux embarqué – elles sont enregistrées dans un fichier `.config`. Nous vous proposons un fichier `.config` déjà préparé pour la compilation de *Buildroot* sur <http://www.logilin.fr/files/articles/config-buildroot>.

```
$ cp config-buildroot .config
$ make menuconfig
```

Observez la configuration proposée, vous remarquerez que nous n'avons inclus la compilation d'aucun utilitaire pour la cible (sauf *gdbserver*) ni noyau. Dans le menu "*Build options*", vous trouverez une entrée "*Toolchain and header file location?*". La chaîne par défaut est "`${HOME}/cross/arm-linux`". C'est l'emplacement où la *toolchain* sera installée à l'issue de la compilation. Attention, le chemin doit être définitif, il ne faut pas essayer de déplacer ensuite la *toolchain*. Si vous désirez l'installer dans un répertoire système (par exemple `/opt/cross/arm-linux`, il vous faudra exécuter la compilation avec les droits *root*).

Quittez l'interface de configuration en sauvant les modifications, puis lancez la compilation ainsi :

```
$ make
```

La chaîne doit se compiler sans erreur. Si ce n'est pas le cas, vérifiez la liaison Internet (en ligne de commande avec `wget` et `ftp`), et la présence des outils de compilation natifs (chaîne `gcc`).

A l'issue de cette (assez longue) compilation, pendant laquelle on observe des phases de téléchargement des sources des *packages* nécessaires, on trouvera dans le répertoire indiqué plus haut une arborescence :

```
$ cd ~/cross/arm-linux/
$ ls
bin      lib      usr
$
```

Le répertoire `bin` est généralement vide (sauf si vous avez ajouté d'autres applications lors du "`make menuconfig`").

Le répertoire `lib` contient les bibliothèques qui sont nécessaires pour le bon fonctionnement du système cible. On y trouve entre autres :

- la bibliothèque **uclibc** (une bibliothèque C, au même titre que la *Glibc*, mais très allégée pour les systèmes embarqués),
- l'éditeur de liens dynamiques `ld-uclibc.so`,
- les bibliothèques supplémentaires `libm` (mathématiques), `libpthread` (*Posix Threads*), `libcrypt` (utilisée pour les applications réseau sécurisées), `librt` (*Real-Time* contenant les extensions de la norme *Posix.1b*), `libdl` (pour charger dynamiquement des bibliothèques supplémentaires)...

Le répertoire `usr/bin` contient la chaîne de compilation, dont :

- `arm-linux-ar` (archivageur pour créer des bibliothèques),
- `arm-linux-as` (assembleur),
- `arm-linux-gcc` (compilateur),
- `arm-linux-gdb` (débugueur),
- `arm-linux-ld` (éditeur de liens),
- etc...

### Test de la chaîne de compilation croisée

Nous invoquerons donc toujours notre compilateur ainsi :

```
$ ~/cross/arm-linux/usr/bin/arm-linux-gcc
```

Ceci peut paraître fastidieux, mais nous verrons dans un prochain article comment installer un environnement de développement où la commutation entre un compilateur et l'autre se fait très aisément.

Testons notre *cross-toolchain* en compilant une application minimale :

```
// hello.c

#include <stdio.h>
#include <time.h>

int main(void)
{
    fprintf ("Bonjour, il est %s\n",
            ctime(time(NULL)));
    return 0;
}
```

```
$ ~/cross/arm-linux/usr/bin/arm-linux-gcc hello.c -Wall -o hello
$ file hello
hello: ELF 32-bit LSB executable, ARM, version 1 (SYSV), dynamically
linked (uses shared libs), not stripped
$
```

Nous voyons bien que le fichier est compilé pour un processeur Arm. Si nous essayons de l'exécuter sur un PC, la commande échoue :

```
$ ./hello
bash: ./hello: cannot execute binary file
$
```

Notre chaîne de compilation est prête. Nous pouvons vérifier avec l'utilitaire *qemu*

l'exécution correcte de notre application de test :

```
$ qemu-arm-L ~/cross/arm-linux/./hello
Bonjour, il est Sun May9 08:44:40 2010
$
```

L'option -L précise le répertoire servant de racine pour trouver les bibliothèques système.

Dans le prochain article de cette série, nous nous attaquerons à un gros morceau : la compilation du noyau Linux...

Christophe BLAESS

*Retrouvez toutes ces manipulations dans notre formation  
« Linux Temps-Réel et Embarqué »  
sur <http://www.logilin.fr/>*