Systèmes d'exploitation et environnements d'exécution embarqués

Rapport de laboratoire

Master HES-SO

Émilie Gsponer, Grégory Emery

3 mars 2016
version 1.0

Table des matières

1 Introduction REPTAR

1.1 Mise en place de l'environnement, utilisation de git

a) Donnée : Il faut tout d'abord récupérer le dépôt étudiant pour les laboratoires SEEE à l'aide de la commande suivante (via une fenêtre de terminal) :

```
$ git clone firstname.lastname@eigit.heig-vd.ch:/home2/reds/seee/seee_student
```

Travail réalisé : Nous n'avions pas les droits d'accès pour le dépôt git, nous l'avons donc téléchargé, puis extrait depuis le lien : https://drive.switch.ch/index.php/s/TbHxQZtmO9IVdkb. Le dossier seee_student a ensuite été placé dans : /home/redsuser/

b) Donnée : Lancez Eclipse et ouvrez le workspace seee_student. Vous devriez obtenir la liste des projets (à gauche). Chaque projet a un lien symbolique dans la racine du workspace.

Travail réalisé : En introduisant le path du dossier seee_student comme workspace au lancement d'Eclipse, nous obtenons la liste de projets suivante :

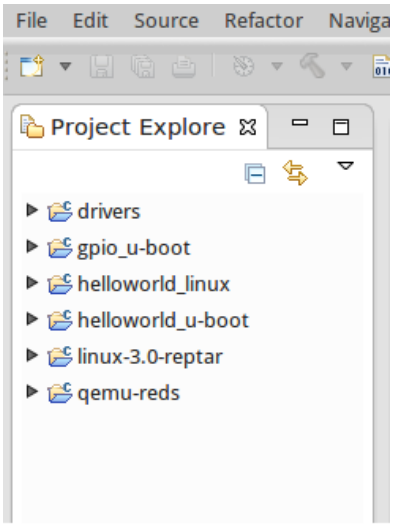

FIGURE 1 – Liste des projets

c) Donnée : Compilez maintenant l'émulateur Qemu. Dans une fenêtre de terminal, lancez la commande suivante à partir de votre répertoire seee_student :

```
$ make qemu
```

Travail réalisé : Vu que nous n'avons pas téléchargé le dossier de projets depuis git, il faut nettoyer le contenu du dossier avec clean ou distclean avant de pourvoir utiliser qemu. Le make qemu prend quelques instants.

```
redsuser@vm-reds-2015s2:~/seee_student$ make clean
redsuser@vm-reds-2015s2:~/seee_student$ make qemu
...
make[1]: Leaving directory '/home/redsuser/seee_student/qemu-reds'
redsuser@vm-reds-2015s2:~/seee_student$
```

1.2 Démarrage de Qemu

a) Donnée : Depuis Eclipse, lancez le debugger avec la configuration de debug « qemu-reds Debug ». Dans la fenêtre Console, vous pourrez entrer directement des commandes de U-boot (tapez help par exemple).

Travail réalisé :

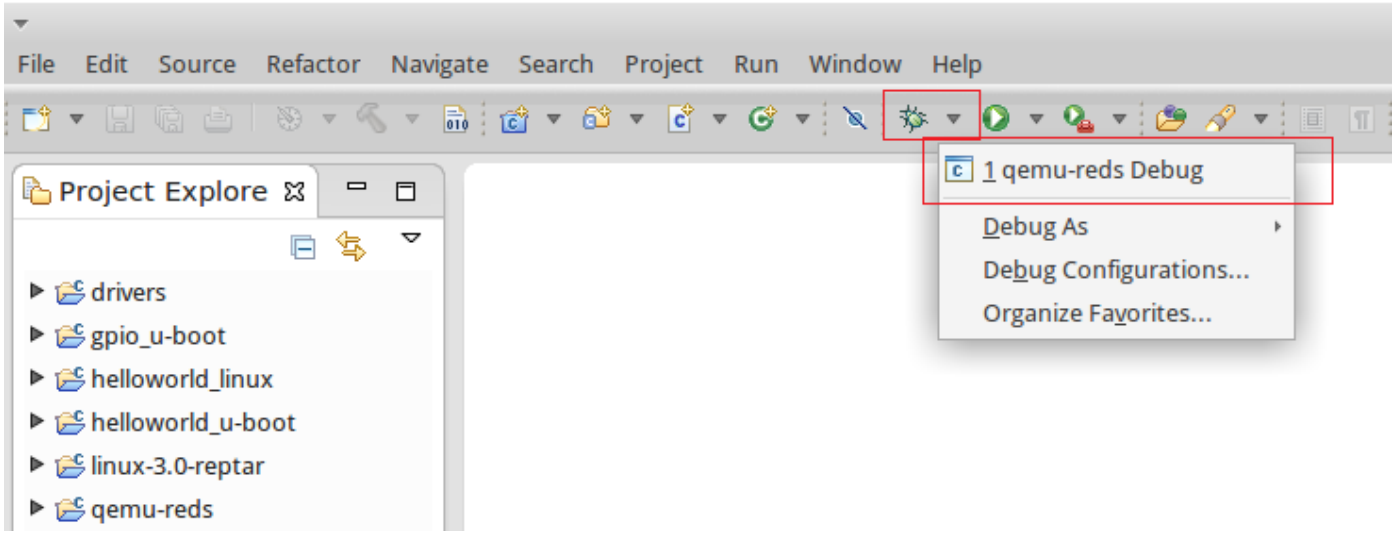

FIGURE 2 – Lancement d'Eclipse en mode Debug

Remarque : Après le lancement du Debug, il faut changer d'onglet en haut à droite en choisissant *Debug* pour avoir la console. Ce changement d'onglet ne se fait pas automatiquement.

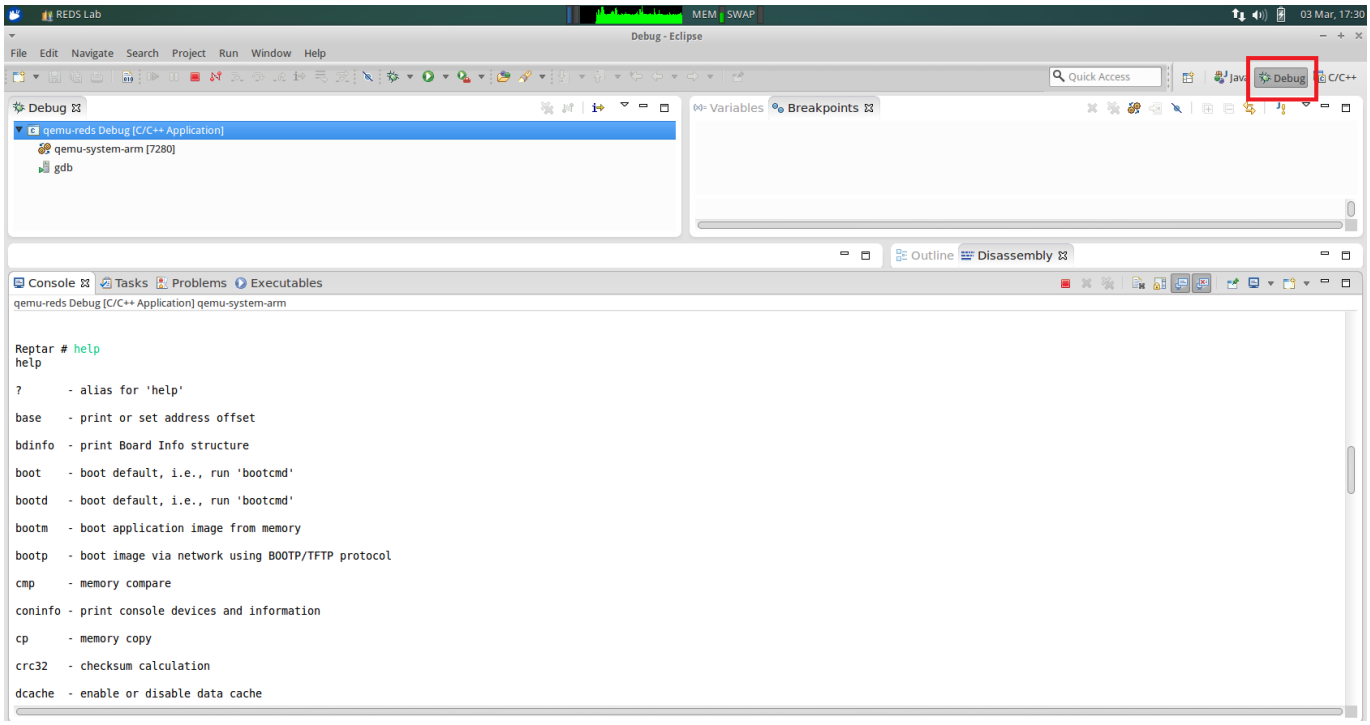

FIGURE 3 – Command help dans l'U-boot

En interrompant le programme avec le bouton *suspend*, on obtient la vue assembleur ci-dessous. L'environnement essaie d'ouvrir le fichier ppoll.c, on est donc en attente d'un événement. Le programme est interrompu après un syscall, il compare deux valeurs ce qui confirme qu'il est effectivement en attente d'événements. L'adresse comparée est sur 64bits, cela nous indique que l'on est sur l'environnement émulé de la machine hôte. Sur le Reptar, l'adresse serait 32bits.

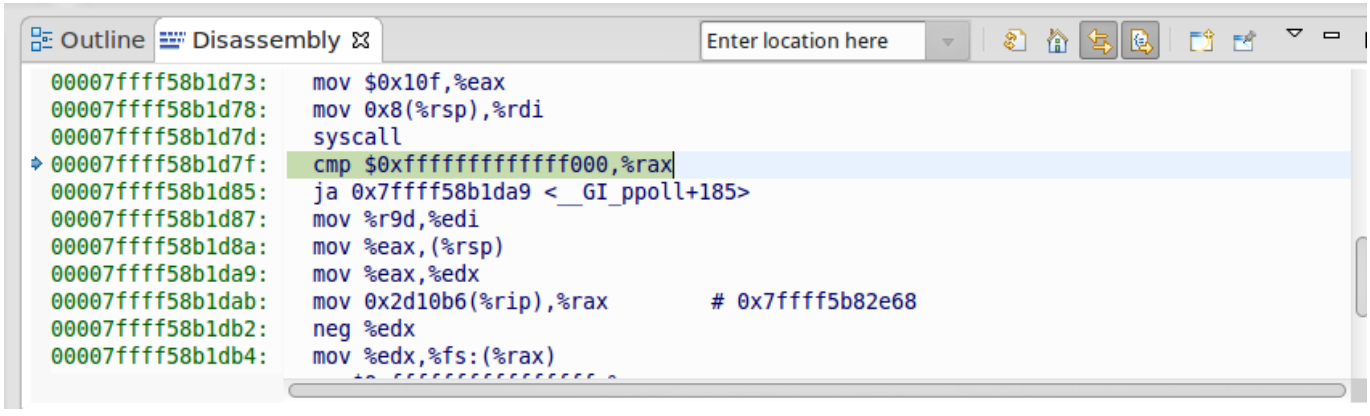

FIGURE 4 – Command help dans l'U-boot

c) Donnée : Stoppez l'exécution, et dans une fenêtre de commande, démarrez qemu à l'aide du script stf (en tapant ./stf) dans le répertoire racine. Vous arrivez dans U-boot.

Travail réalisé : Cette partie n'a plus rien avoir avec Eclipse, on peut le fermer et lancer un terminal. Avec la commande *stf* tapée à la racine du répertoire seee_student, on arrive au même point qu'en lançant le Debug dans Eclipse. On peut également essayer la commande *help*

```
redsuser@vm-reds-2015s2:~/seee_student$ ./stf
WARNING: Image format was not specified for 'filesystem/flash' and probing
    guessed raw.
...

Reptar # help
?       - alias for 'help'
base    - print or set address offset
bdinfo  - print Board Info structure
boot    - boot default, i.e., run 'bootcmd'
bootd   - boot default, i.e., run 'bootcmd'
bootm   - boot application image from memory
bootp   - boot image via network usi
...
```

1.3 Tests avec U-boot

a) Donnée : Dans U-boot, listez les variables d'environnement avec la commande printenv. Observez les variables prédéfinies « tftp1, tftp2 et goapp ». Ces variables définissent des commandes U-boot qui peuvent être exécutées à l'aide de la commande run (par exemple run tftp1). La commande go <addr> permet de lancer l'exécution à l'adresse physique <addr>. Vous pouvez définir/modifier vos propres variables et les sauvegarder dans la flash émulée avec la commande saveenv (seulement avec le lancement via stf).

Travail Réalisé :

b) Donnée : La production de l'exécutable helloworld_u-boot s'effectue en tapant la commande make dans le répertoire contenant les sources du programme. Ensuite, vous pouvez transférer le fichier (extension .bin) dans U-boot et exécuter le binaire (aidez-vous des variables d'environnement prédéfinies).

Travail réalisé :

c) Donnée : Testez le debugger dans Eclipse avec le projet helloworld_u-boot. Mettez un breakpoint dans le code source au démarrage du programme, et lancez le debugger avec la configuration de debug « helloworld_u-boot Debug ».

Travail Réalisé :

1.4 Tests avec Linux

a) Donnée : Lancez le script ./deploy qui permettra de déployer le noyau Linux dans la sdcard virtuelle (ignorez l'erreur due à l'absence de certains fichiers).

Travail réalisé :

b) Donnée : Poursuivez ensuite en cross-compilant l'application helloworld pour Linux (via make).

Travail réalisé :

c) Donnée : Copiez l'exécutable dans le rootfs

Travail réalisé :

d) Donnée : Lancez le script stq suivi de la commande boot dans U-boot pour amorcer le démarrage de Linux

Travail réalisé :

e) Donnée : Lancez votre application

Travail réalisé :

f) Donnée : Dans Linux, tapez la commande suivante :

```
$ /usr/share/qt/examples/effects/lighting/lighting -qws &
```

Travail réalisé :

1.5 Tests sur la plate-forme réelle

a) Donnée : Déployez l'application helloworld dans U-boot sur la plate-forme REPTAR avec l'interface réseau. Le transfert peut s'effectuer avec la commande tftp. Il est nécessaire d'exécuter la commande suivante pour mettre à jour les adresses IP et MAC de la plate-forme REPTAR :

```
# run setmac setip
```

Travail réalisé :

b) Donnée : Déployez l'application helloworld dans Linux à l'aide du réseau et de la commande scp.

Travail réalisé :

1.6 Accès aux périphériques REPTAR

a) Donnée : Sur la base de l'exemple gpio_u-boot., vous devez développer une application permettant d'interagir avec les LEDs et les switchs présents sur la carte CPU de la plate-forme REPTAR. Le but de l'application est d'allumer une LED lorsqu'on appuie sur un switch.

1. La LED 0 doit s'allumer lorsqu'on appuie sur le SWITCH 0.
2. La LED 1 s'allume si l'on appuie sur le SWITCH 1.
3. Et ainsi de suite pour les LEDs et switchs 0..3 de la carte CPU.

Le switch numéro 4 sert à quitter l'application. Aidez-vous des fichiers d'en-tête (#include) déjà présents dans le chablon fourni.
L'application gpio_u-boot est à déployer dans U-boot via la commande tftp.

Travail réalisé :