TP MI11 Prise en main de Linux embarqué
Partie 2

Semestre printemps 2017

Guillaume Sanahuja – guillaume.sanahuja@hds.utc.fr

Table des matières

Travail préalable à la séance de TP.....1
Exercice 1 : Hello World.....1
Exercice 2 : Clignotement des LEDs.....2
Exercice 3 : Boutons poussoirs.....2
Exercice 4 : Charge CPU.....3
Exercice 5 : Touchscreen et PWM (facultatif).....3

Travail préalable à la séance de TP

Relire le cours sur Linux embarqué disponible sur le moodle MI11. Télécharger les ressources techniques sur le site MI11 et prendre connaissance du manuel utilisateur de la carte DevKit8600 (DevKit8600 User Manual.pdf). Avoir compris le précédent TP.

Un compte rendu de TP au format PDF est à rendre sur le moodle de l'UV . Il vous est demandé d'y écrire les réponses aux questions figurant dans les différents exercices. Également, vous pouvez compléter le compte rendu avec toute information que vous jugerez utile, par exemple les manipulations que vous avez réalisées pendant la séance, les résultats que vous avez observés, …

Pour ce TP, vous devez réutiliser la machine virtuelle de la séance précédente (clone de la machine *mi11_devkit8600*) . Vous travaillerez sur ce clone.

Vous devez avoir terminé l'exercice 5 du TP précédent pour pouvoir commencer celui-ci. Si ce n'est pas le cas demander de l'aide au chargé de TP afin de démarrer rapidement ce TP. Vous utiliserez donc le boot par le réseau sur le noyau et le système de fichiers de la séance précédente.

Pour l'ensemble de ce TP, s'assurer qu'aucune carte SD n'est présente dans la cible.

Exercice 1 : Hello World

Maintenant que nous avons pu cross-compiler notre système et le démarrer, nous allons programmer un premier exécutable qui affiche le message « Hello World ! ». Etant donné que le

code sera inclus uniquement dans un fichier source, nous n'avons pas besoin d'utiliser de Makefile.

Créez un fichier *main.c* et éditez le. Ajoutez le code C qui permet d'afficher le message voulu puis compilez votre programme avec la commande suivante :

```
gcc main.c –o main
```

Utilisez la commande *file* pour obtenir des renseignements sur le fichier *main.*

Question 1.1 : Que constatez vous ? Pourquoi ce fichier ne peut-il pas s'exécuter sur la cible ?

Cross-compilez maintenant votre programme avec la commande suivante :

```
arm-poky-linux-gnueabi-gcc main.c -o main
```

Question 1.2 : Que faut-il faire avant de pouvoir lancer cette commande ?

Question 1.3 : Utilisez de nouveau la commande file, que constatez vous ?

Vérifiez ensuite le bon fonctionnement sur la cible.

Question 1.4 : Fournissez le code dans le compte rendu et le résultat obtenu dans le terminal.

Exercice 2 : Clignotement des LEDs

Nous allons maintenant manipuler les périphériques de la cible. Pour que ce soit visuel, nous allons nous attaquer aux traditionnelles LEDs. Trois sont présentes sur la carte de développement, la rouge étant réservée à la fonction POWER. Les deux LEDs vertes sont accessibles et correspondent aux LEDs système et utilisateur.

Question 2.1 : Rappelez comment accéder aux LEDs en manipulant des fichiers.

Dans le terminal, affichez les valeurs de ces fichiers et modifiez-les.

Question 2.2 : Fournissez les 4 lignes de commandes permettant d'allumer et d'éteindre les 2 LEDs.

Créez maintenant un nouveau fichier *led.c* qui devra allumer en alternance la LED utilisateur et la LED système chaque seconde. Compilez ce code avec la chaîne de compilation croisée et testez-la sur la cible. Pour rappel, les entrées sorties étant vu comme des fichiers sous Linux, vous pouvez utiliser les fonctions open, read, write pour accéder aux IO et contrôler les LEDs.

Question 2.3 : Fournissez le code source led.c et faites valider le fonctionnement de l'application par le chargé de TP.

Exercice 3 : Boutons poussoirs

Question 3.1 : Combien y a t-il de boutons poussoirs sur la cible ? Comment y accède t-on ?

Faites un essai de lecture de l'état des boutons en ligne de commande.

Question 3.2 : Donnez les commandes utilisées. Quelles sont les valeurs des différents événements ?

Écrivez maintenant un programme réagissant aux actions sur les boutons, en affichant un message et/ou allumant une LED par exemple. Pour cela, vous devez stocker le résultat de la lecture du fichier dans une structure de type *input_event*, qui contiendra donc les informations liées à l'événement d'une touche.

Question 3.3 : Faites une recherche dans la cross-toolchain pour trouver où est déclarée la struct input_event.

Cross-compilez votre programme et testez le.

Question 3.4 : Fournissez le code source et faites valider le fonctionnement de l'application par le chargé de TP.

Exercice 4 : Charge CPU

Cet exercice va vous permettre de vérifier l'incidence de la charge CPU sur une tâche périodique. Ecrivez donc un programme réalisant 10 000 fois une attente de 1ms. Utilisez les fonctions *gettimeofday* et *timersub* pour mesurer le temps total. Utilisez maintenant la commande *stress* pour charger le CPU.

Question 4.1 : Quels sont les différents temps relevés ?

Améliorez votre programme afin de calculer et afficher les latences minimum, maximum et moyenne.

Question 4.2 : Quels sont les différents temps relevés ? Quelles sont vos conclusions ? Comment pourrait-on améliorer les résultats ?

Question 4.3 : Fournissez le code source de votre programme.

Exercice 5 : Touchscreen et PWM (facultatif)

Le but de cet exercice est d'écrire un programme réagissant au touchscreen afin de changer le rétro éclairage de l'écran.

Question 5.1 : Sachant que le touchscreen est un périphérique d'entrée tout comme les boutons poussoirs, quel est le fichier à manipuler ?

Faites un essai de lecture de l'état du touchscreen en ligne de commande.

Question 5.2 : Quelles sont les caractéristiques du touchscreen ?

Question 5.3 : Quel est le fichier à manipuler pour contrôler le rétro éclairage de l'écran?

Faites un essai de modification du rétro éclairage en ligne de commande.

Question 5.4 : Donnez les commandes utilisées.

Codez maintenant un programme contrôlant graduellement la valeur du rétro éclairage en fonction le zone de l'écran appuyée (éteint à gauche et totalement allumé à droite).

Question 5.5 : Fournissez le code source et faites valider le fonctionnement de l'application par le chargé de TP.