TP1 MI11 Linux Xenomai

Semestre printemps 2017

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

Table des matières

Travail préalable à la séance de TP

Relire le cours sur Linux embarqué et sur Linux temps réel disponible sur le moodle MI11. Regarder la documentation de Xenomai :
www.xenomai.org/documentation/xenomai-2.6/html/api/index.html
Et lire notamment la documentation sur l'API native, que vous utiliserez pour ce TP :
www.xenomai.org/documentation/xenomai-2.6/html/api/group__native.html

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*) .

Cependant, le démarrage ne se fera plus par le réseau sur le noyau et le système de fichier cross-compilés lors du premier TP sur Linux embarqué. En effet, vous devez utiliser un noyau avec Xenomai. Un effet de bord (non désiré) du noyau avec Xenomai est la perte du driver réseau. Le noyau et le système de fichiers sont donc installés pour cette séance sur une carte SD. **Attention, les fichiers présents sur la carte SD seront effacés entre les deux séances de TP sur Xenomai ; n'y stockez pas d'informations.**
Afin de garder une certaine souplesse, une interface réseau est tout de même disponible via la fonctionnalité *on the go* de l'USB. Pour cela, il faut brancher le câble micro USB entre la cible et le PC. La cible a pour IP 192.168.7.2.

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

Exercice 1 : Tâches

Dans cet exercice, nous allons créer une application simple de type « Hello World » sous Xenomai. L'objectif principal sera de manipuler les tâches temps réel et d'analyser leur fonctionnement.

Reprenez le code du programme « Hello World » du TP sur Linux embarqué et adaptez le pour afficher le message à intervalle régulier (une fois par seconde par exemple). Cross-compilez ce code, téléchargez-le et exécutez-le sur la carte Devkit8600.

Question 1.1 : Ce code s'exécute-t-il de façon temps réel ? Comment le vérifier (regarder le fichier de statistiques de Xenomai)?

Créez maintenant une tâche temps réel Xenomai (avec l'API native) qui se chargera d'exécuter le *printf* et le *sleep*. Afin de cross-compiler ce programme, il faut indiquer à gcc où se trouvent les *headers* de Xenomai (option *-I*) et quelles librairies utiliser (option *-l*). Vous aurez besoin des librairies *native* et *xenomai*.

Question 1.2 : Donnez la ligne de commande utilisée pour la cross-compilation ainsi que le code du programme.

Question 1.3 : Le code est-il vraiment temps réel et pourquoi? Que donne le fichier de statistiques de Xenomai (ne pas interpréter ce résultat pour le moment)?

Remplacez maintenant l'appel à la fonction *sleep* par son équivalent sous Xenomai.

Question 1.4 :Donnez le code du programme et les statistiques de Xenomai (ne pas interpréter ce résultat pour le moment).

Remplacez maintenant l'appel à la fonction *printf* par son équivalent sous Xenomai.

Question 1.5 :Donnez le code du programme et les statistiques de Xenomai. Interprétez maintenant les résultats des différentes statistiques que vous avez relevées.

Exercice 2 : Synchronisation

Créez un programme lançant deux tâches Xenomai qui afficheront chacune une partie d'un message (chaque tâche ne doit rien faire d'autre).

Question 2.1 : Donnez le code du programme et le résultat.

Question 2.2 : Quelle est l'influence de la priorité des tâches ? Comment faire pour afficher le message dans l'ordre ou le désordre ? Justifiez.

Afin d'améliorer le comportement de votre programme, utilisez un sémaphore qui bloquera chacune des tâches dès le début. Les tâches devront être libérées après avoir été toutes lancées.

Question 2.3 : A quelle valeur faut-il initialiser le sémaphore ?

Question 2.4 : Quelle est l'influence du paramètre mode *utilisé à la création du sémaphore* ?

Question 2.5 : Donnez le code du programme et le résultat. Expliquez le fonctionnement du programme.

Nous allons maintenant faire l'affichage du message en boucle (une fois par seconde), grâce à une troisième tâche qui servira de métronome :
- modifiez les deux tâches d'affichage pour faire l'affichage en boucle
- ajoutez la troisième tâche qui synchronisera les deux autres et réalisera l'attente

Question 2.6 : Donnez le code du programme et le résultat. Expliquez le fonctionnement du programme.

Question 2.7 : Regardez le fichier de statistiques et du scheduler de xenomai. Quelles sont les différentes informations ? Vous pouvez bloquer l'avancement de votre programme en utilisant la fonction getchar par exemple.

Exercice 3 : Latence

Dans cet exercice, nous allons nous intéresser à la latence de Xenomai, et la comparer avec les résultats du TP précédent.
Il faut donc réécrire un programme réalisant 10 000 fois une attente de 1ms, en utilisant Xenomai.

Question 3.1 : Donnez le code du programme.

Ajoutez la mesure des latences minimum, maximum et moyenne. Veillez à utiliser les fonctions Xenomai pour la prise de temps et à utiliser le ficher entête correspondant.

Question 3.2 : Donnez le code du programme et les résultats obtenus. Que pouvez vous en conclure ?

Question 3.3 : Chargez le CPU et donnez les résultats obtenus. Que pouvez vous en conclure ?