Xenomai 3 – Prise en main – Écrire des tâches temps réels périodiques

Damien Masson / Laurent George – prenom.nom@esiee.fr – bureaux 4206/5255

22 mai 2017

Vous allez utiliser le système d'exploitation Xenomai 3 installé en mode conoyau afin de programmer un système de tâches temps réel périodiques que vous aurez tout d'abord analysé d'un point de vue théorique. Vous allez comparer les traces d'exécution de ce système à la théorie.

1 Qu'est-ce que Xenomai

Vous allez commencer par lire la documentation sur la page `https://xenomai.org/start-here/` (lire les deux premières sections : "what is Xenomai about" ? et "How does Xenomai deliver real-time ?".

Vous êtes ensuite libre de chercher de la documentation pour répondre à ces questions :

- pourquoi Linux ne peut pas être utiliser pour programmer des applications avec des contraintes temps-réel fortes ?
- qu'apporte le patch "Preempt-RT" pour résoudre ce problème ?
- quelles sont ses limites ?
- quelle est l'autre approche proposé par Xenomai 3 (et par d'autre RTOS comme RTAI par exemple) ?

La documentation pour la suite se trouve à l'adresse `https://xenomai.org/documentation/xenomai-3/html/xeno3prm/group__alchemy.html`.

2 Hello RT-World

Si ce n'est pas déjà fait, bootez votre machine sur le noyau Xenomai.

Téléchargez l'archive `http://igm.univ-mlv.fr/~masson/v2/Teachings/IN4R21/tp1/hello.tgz` et décompressez la à l'endroit de votre choix. Étudiez le fichier hello.c, compilez avec la commande make, testez. En cas de soucis, vérifiez que le répertoire /usr/xenomai/lib se trouve bien dans les répertoires de recherche des librairies dynmique (variable `LD_LIBRARY_PATH`). De plus, vous devez avoir les privilèges super-utilisateur pour lancer l'application.

3 Hello Periodic RT-World

Téléchargez l'archive `http://igm.univ-mlv.fr/~masson/v2/Teachings/IN4R21/tp1/period.tgz`, décompressez-la et étudiez le code. Compilez, Testez.

4 Simuler de la charge processeur

Proposez une méthode permettant pour une tâche de générer de la charge processeur, i.e. la tâche cherche à effectuer des opérations actives pendant un temps C. Visualisez la charge générée en affichant dans la console les dates d'exécutions des tâches.

Si vous n'arrivez pas à générer une charge a peu pres constante, voici quelques conseils :

— utilisez une(des) variable(s) "static" pour les opérations qui occupent le processeur
— cherchez comment dire à gcc de ne pas optimiser le code et modifiez le makefile en conséquence

Vous avez également vu au TP précédant une méthode pour occuper le processeur. Testez-la et comparez. Choisissez la meilleure solution

5 Mise en œuvre

Soit les deux systèmes de tâches suivants :

	C_i	T_i	D_i
τ_1	1	4	4
τ_2	2	6	6
τ_3	3	8	8

	C_i	T_i	D_i
τ_1	2	7	7
τ_2	2	11	11
τ_3	5	13	13

— sont-ils ordonnançables ? avec quel(s) algorithme(s) ?
— tracez les chronogrammes théoriques avec les algorithmes à priorités fixes que vous jugerez adéquats
— programmez ces systèmes et relevez les traces d'éxécutions
— comparez

NOTE : si votre machine de tests utilise plusieurs processeurs, il faudra tester le comportement par défaut de Xenomai : utilise-t-il plusieurs processeurs pour les tâches temps réel ? Comment mettre cela en évidence ? Etudiez la documentation pour savoir comment spécifier "l'affinité processeur" des tâches.

Une autre solution peut consister à générer des tâches supplémentaires pour être sur que les tâches étudiées s'exécutent sur le même processeur (puisque votre étude théorique est mono-processeur).