HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Reconfigurable & embedded Digital Systems

Prof. Yann Thoma

Laboratoire de Programmation Temps Réel

semestre automne 2019 - 2020

Laboratoire 4 : Watchdog

Temps à disposition : 4 périodes

Objectifs

En cas de surcharge une application devrait être capable de supprimer des tâches afin de ne pas mettre à mal le système. Dans ce contexte, ce laboratoire vise à développer un système de watchdog.

Le système sera intégré sur une carte DE1-SoC.

Etape 1 : Réalisation du watchdog

Le concept à développer est celui du canari. Votre système devra comporter une tâche dite *canari* qui devra indiquer son bon fonctionnement de manière régulière. Une autre tâche *watchdog* devra quant à elle vérifier que le canari est toujours en vie, et ce également de manière régulière, et si ce n'est pas le cas, elle devra supprimer les autres tâches du programme en cours.

Reprenez le fichier watchdog.c, et implémentez le comportement suivant :

1. Le canari : Une tâche périodique de très basse priorité Pr_b et de période P_b signale son activité à chaque période.
2. Le watchdog : Une tâche périodique de très haute priorité Pr_h et de période P_h vérifie, à chaque période, si le canari est encore en vie. Si tel n'est pas le cas, les tâches responsables de la surcharge doivent être retirées.
3. Toutes les tâches du programme doivent avoir une priorité Pr_x respectant $Pr_b < Pr_x < Pr_h$
4. Les périodes doivent respecter la relation suivante : $P_h > P_b$
5. En réalité Xenomai intègre un watchdog qui détruit le programme s'il surcharge le CPU pour plus de 4 secondes. Le choix des périodes du watchdog et du canari devront donc prendre ce fait en compte.

Etape 2 : Test du watchdog

1. Reprenez le code de intro_watchdog, et modifiez la charge CPU afin qu'elle dépasse 1 après un certain temps d'exécution.
   — La fonction `busy_cpu()` vous permet de simuler du temps d'exécution.
2. Lancez votre application et vérifiez que le watchdog fonctionne correctement.

Travail à effectuer

Vous devez commenter votre code et commencer votre documentation par une explication de vos choix architecturaux. La documentation proposée est écrite en Doxygen. Vous disposez, dans le répertoire ressources, d'un fichier `Doxyfile`. Vous pouvez donc utiliser la commande `doxygen`, lancée dans ce répertoire, pour générer la documentation. Le code sera à rendre par cyberlearn, en créant une archive contenant le répertoire `ressources`.

Compilation et exécution du labo watchdog sur Xenomai 3/De1-SoC

Mise en route de la carte DE1-SoC

La carte uSD fournie comprend Linux 4.19.55 patché pour Xenomai 3. La version 4.19.55 est la plus récente supportée par Xenomai.

Démarrer la board et se loguer avec root/root au travers d'une liaison série, après avoir physiquement connecté la carte via le lien USB (connecteur micro-USB sur la carte).

```
$ picocom -b 115200 /dev/ttyUSB0
```

Afin de déterminer si le noyau Cobalt de Xenomai est bien installé, il suffit de regarder dans les logs de Linux :

```
$ dmesg | grep -i -e xenomai -e i-pipe
```

et on devrait obtenir :

```
[    0.000000] L2C: I-pipe: revision >= L310-r3p2 detected, forcing WA.
[    0.000000] L2C: I-pipe: write-allocate enabled, induces high latencies.
[    0.000000] I-pipe, 200.000 MHz timer
[    0.000000] I-pipe, 100.000 MHz clocksource, wrap in 42949 ms
[    0.004841] I-pipe, 200.000 MHz timer
[    0.142907] [Xenomai] scheduling class idle registered.
[    0.142925] [Xenomai] scheduling class rt registered.
[    0.143063] I-pipe: head domain Xenomai registered.
[    0.146542] [Xenomai] Cobalt v3.1-devel [DEBUG]
```

Les bibliothèques partagées, fichiers d'inclusion et utilitaires relatifs à Xenomai se trouvent dans le dossier `/usr/xenomai`.

Dans le dossier `bin`, on retrouve les utilitaires `dohell` et `latency` afin de tester les capacités RT du système.

Dans le dossier `lib`, on y trouve la bibliothèque partagée pour accéder à l'API `alchemy` (anciennement `native`).

Compilation

Nous allons développer en Xenomai 3.

Les API ayant changé significativement entre les versions 2 et 3 de Xenomai, se référer au document suivant afin de voir les correspondances entre les versions 2 et 3, les fonctionnalités ajoutées ou supprimées :

`https://xenomai.org/documentation/xenomai-3/html/MIGRATION/index.html`

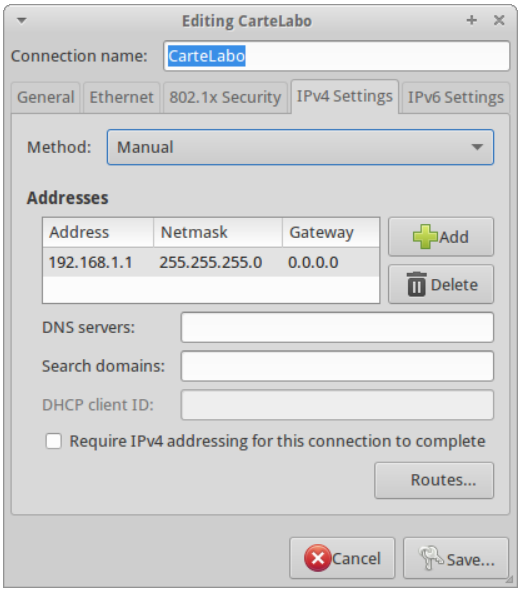
En résumé, le changement majeur qui nous intéresse est la nouvelle API `alchemy` (anciennement `native`). Les interfaces sont en général identiques, mais certaines fonctionnalités sont devenues obsolètes. Ici se trouve la documentation de l'API `alchemy` :

`https://xenomai.org/documentation/xenomai-3/html/xeno3prm/group__alchemy.html`

La plateforme cible étant basée sur ARM, il est nécessaire de cross-compiler le labo watchdog et de le linker avec les bibliothèques Xenomai elles aussi cross-compilées. Dans l'archive du labo watchdog, le dossier `libxenomai` contient tous les fichiers d'inclusions et bibliothèques nécessaires à la cross-compilation de watchdog. Il contient tout simplement une copie du contenu de `/usr/xenomai` qui se trouve sur la De1-SoC.

Exécution

Connecter la machine hôte et la De1-SoC avec un câble ethernet. Considérons le sous-réseau '192.168.1.0'. Configurer l'interface ethernet de la machine hôte avec une adresse fixe ex : '192.168.1.1'. Ouvrez l'édition des connexions réseau et appliquez la configuration à CarteLabo :



Configurer l'adresse IP de la De1-SoC :

```
$ ifconfig eth0 192.168.1.3 netmask 255.255.255.0 up
```

Transférer l'exécutable par ssh sur la De1-SoC :

```
$ scp intro_watchdog root@192.168.1.3:/home/root
```

Une fois l'executable transféré, il suffit de l'exécuter de la manière suivante :

```
$ LD_LIBRARY_PATH=/usr/xenomai/lib ./intro_watchdog
```

Bien sûr, il est possible d'exporter de manière persistante `/usr/xenomai/lib` dans `LD_LIBRARY_PATH` :

```
$ export LD_LIBRARY_PATH=/usr/xenomai/lib
```

Pour l'exécution sur la carte il est possible de rester sur le port série, mais à partir du moment où vous avez défini correctement les adresses IPs et que `scp` a fonctionné, vous pouvez également vous connecte en ssh.