INSA Lyon
4ème année
Spécialité Informatique

Gestion des ressources physiques des systèmes informatiques

Projet

Kevin Marquet

Novembre 2013

1 Description du projet

L'objectif général de ce projet est d'illustrer les notions vues en cours associées à l'ordonnancement, aux systèmes temps-réel. Vous illustrerez ces notions à la fois au-dessus de Linux et dans votre mini OS et dans les deux cas, on s'intéresse à l'exécution *sur Raspberry Pi*.

Sur chacun des systèmes, vous devrez tester différentes politiques d'ordonnancement permettant de mettre en évidence :

— le besoin de synchronization entre processus ;
— le besoin d'ordonnancement à priorités ;
— l'impact de la politique d'ordonnancement sur les performances ;
— les différents critères de choix des algorithmes d'ordonnancement ;
— le fait que les processus sont bornés en temps ou en I/O.

Concrètement, vous devrez :

— Terminer votre mini OS (aller jusqu'au bout du sujet précédent) ;
— Implémenter, dans votre mini OS, différentes politiques d'ordonnancement différentes de roiund-robin, dont au moins une à priorités ;
— Implémenter un ensemble de processus fonctionnant au-dessus de votre mini OS, permettant d'illustrer les différences entre les différences d'ordonnancement. Un de ces processus doit être un processus multimédia (lecture de musique, vidéo ou autre) et doit ramer avec certaines politiques d'ordonnancement, pas avec d'autres.
— Implémenter un ensemble de processus fonctionnant au-dessus de Linux, permettant d'illustrer les différences entre l'ordonnanceur `USER` et l'ordonnanceur `RT`. Encore une fois, un de ces processus doit être un processus multimédia (lecture de musique, vidéo ou autre) et doit ramer avec uen politique d'ordonnancement, pas avec l'autre.
— Préparer exposé et démonstrations (voir plus bas).

2 Suggestions d'applications

Voici quelques exemples d'applications que vous pouvez utiliser ou implémenter pour illustrer les concepts vus en cours. Gardez à l'esprit que vous cherchez à illustrer ces concepts, et pas "juste" à programmer un jeu vidéo...

2.1 Les philosophes

Le programme donné dans le sujet d'implémentation du mini OS est un bon exemple, bien que ce soit un exemple jouet, permettant de tester vos primitives de synchronisation.

2.2 Clignotage de la LED

Les fonctions `led_on()` et `led_off()` permettent d'allumer et éteindre la LED. Assurez-vous que votre mini OS fonctionne avec deux processus, l'un éteignant la LED régulièrement, l'autre l'allumant.

2.3 Lecteur MIDI

On vous fournit en ligne une archive comprenant un lecteur midi fonctionnel pour votre mini OS.

Également, on vous fournit une archive `pmidi.tgz` comprenant les sources d'un lecteur midi fonctionnant au dessus de Linux. Pour compiler, *make* dans le répertoire `src`.

2.4 Lecteur WAV

On vous fournit en ligne un lecteur de .wav (archive `wav_player.tgz`), fonctionnant uniquement au-dessus de Linux (désolé, pas eu le temps de coder le lecteur .wav pour le mini-OS...). Attention, un décodeur lecteur wav, ça ne demande pas beaucoup de ressources, donc pas forcément utile pour vous.

2.5 Un clavier pour votre mini-OS

Le tutoriel `http://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/os/input01.html` expose comment récupérer les appuis de touches par `polling`.

2.6 Synthétiseur de son

Dans votre mini OS, vous pouvez jouer un son différent selon la touche du clavier. Sous Linux, c'est aussi possible : il existe plein de logiciel libre disponible.

2.7 Jeux : casse-briques, tétris etc.

C'est pas compliqué, si vous utilisez les bonnes librairies (ne tous cas au-dessus de Linux...). Et parfais pour illustrer problèmes de latence, de synchronisation (surtout si vous jouez de la musique en même temps), de performances... mais ça demande peut-être un peu de boulot. En même temps, vous êtes 6 :)

2.8 Lecture vidéo

Au-dessus de Linux, vous devez pouvoir vous en sortir. Au-dessus de votre mini OS, c'est moins clair : nous on n'a pas pris le temps d'écrire ou récupérer un driver vidéo, mais n'hésitez pas !

3 Déroulement et rendu

Le projet dure 5 séances. Une bonne partie de la dernière séance sera consacrée à l'évaluation. Cette évaluation prendra la forme d'un exposé oral pendant lequel vous présenterez votre projet à l'aide de quelques slides et de démonstrateurs. Bien que ces critères soient susceptibles d'évoluer, cet oral sera noté de la manière suivante :

- Mise en évidence de la bonne synchronization de processus + démo d'un problème (par exemple les philosophes) résolu : 3 points ;
- Mise en évidence d'une problématique d'ordonnancement (exemple : le lecteur de musique rame) dans votre mini-OS : 3 points ;
- Choix de la politique d'ordonnancement incluant analyse performances, temps moyens : 5 points. Dans cette partie, vous détaillerez
  - une analyse analytique de vos algorithmes : *waiting time*, *response time* etc.
  - une analyse expérimentale de vos algorithmes : latence observée etc.
- Mise en évidence d'une problématique d'ordonnancement sous Linux : 3 pts ;

— Comparaison entre les comportements sous Linux : 3 pts. Quels avantages à l'un et l'autre des OS ? Quels inconvénients ?
— Mise en évidence du compromis CPU / requêtes I/O de vos applications : 2 points ;
— Clarté de la présentation : 1 pt.