2 - INTERRUPTION HORLOGE

1. CHOIX DU QUANTUM

On considère un système en temps partagé et on suppose, pour simplifier, qu'à chaque passage sur le processeur une tâche utilise la totalité de son quantum de temps. On suppose par contre qu'il y a un overhead de s millisecondes à chaque commutation.

1.1.

Quel est le surcoût en temps lié aux commutations ?

1.2.

On prend un quantum de 100 ms et un overhead de 1 ms. On affecte le temps de commutation en fin de tâche à la tâche qui est retirée du processeur.

Quelle est la durée totale (temps CPU consommé, en incluant les commutations) d'une tâche demandant 20 ms de calcul ? D'une tâche demandant 500 ms de calcul ?

Quel est le pourcentage de temps passé par chacune de ces tâches en commutations ?

1.3.

Quels sont les critères de choix du quantum?

2. GESTION D'UNE INTERRUPTION HORLOGE

On considère le processeur JB007 qui dispose d'un registre temporisateur Rtempo, décrémenté automatiquement toutes les micro-secondes *en mode usager* et *en mode système.* Lorsque sa valeur atteint 0, Rtempo provoque une interruption (interruption horloge). Ce registre est initialisé automatiquement à une valeur RTMAX : celle où tous les bits sont à 1, sauf le bit de poids fort qui est à 0. La représentation utilisée est une représentation en complément à 2.

L'interruption horloge est utilisée à la fois pour tenir à jour l'heure universelle dans un mot RHU, pour instaurer un tour de rôle avec quantum d'exécution et pour comptabiliser le temps CT utilisé par une tâche. Lorsque cette variable CT dépasse une valeur CTMAX, la tâche est détruite.

2.1.

Quelle doit être la taille minimale de Rtempo si l'intervalle entre deux interruptions horloge est de 10 ms ?

2.2.

Pourquoi décrémente-t-on Rtempo même en mode système, et pourquoi continue-t-on à le décrémenter lorsqu'il atteint 0 ?

Le système gère une table TDT des descripteurs de tâches : TDT[i] contient l'ensemble des informations relatives à la tâche i. On considère pour l'instant que la valeur QX du quantum est telle que QX = RTMAX.

2.3.

Quelles sont les opérations concernant la gestion du temps que doit effectuer le système la première fois qu'il charge la tâche i pour l'exécuter ?

Le système dispose d'une variable ITE qui contient l'indice de la tâche élue. On suppose qu'une tâche élue se voit toujours attribuer un quantum entier, même si elle avait précédemment perdu le processeur en ayant utilisé partiellement un quantum de temps (suite à une demande d'E/S par exemple).

2.4.

Donnez l'algorithme de la routine de traitement de l'interruption horloge. Cette routine provoque une nouvelle élection.

2.5.

Pourquoi est-il important que la durée de traitement de l'interruption horloge ne soit pas trop longue ? Quelle doit être la durée maximum de traitement pour conserver une gestion du temps cohérente ?

2.6.

Quels sont les avantages et les inconvénients de toujours initialiser le registre Rtempo à la même valeur ?

3. QUANTUM ET TICK

Sous Unix, un tick correspond au passage à 0 du registre temporisateur. Certaines tâches de l'interruption horloge sont effectuées à chaque tick, alors que d'autres ne sont gérées que tous les N ticks, N dépendant de la tâche à traiter. Par exemple, la commutation de tâches est traitée toutes les 100 ms alors que l'intervalle entre 2 ticks est de 10 ms.

Par contre, la mise à jour de l'heure universelle, du temps utilisé par la tâche ou le test des alarmes à déclencher (réveil de tâche, réémission de paquets, …) est effectué à chaque tick.

3.1.

Modifiez la routine de traitement de l'interruption horloge pour gérer une commutation toutes les 100 ms avec un intervalle entre 2 ticks de 10 ms.

Chaque fois qu'un processus perd le processeur, le système réajuste sa priorité en appliquant la formule :

$$p\_pri = p\_user + p\_cpu + p\_nice$$

où p_pri est la priorité de la tâche, p_user la priorité de base d'une tâche utilisateur, p_cpu le temps utilisé par la tâche, comptabilisé en nombre de ticks, et p_nice le paramètre d'une commande nice effectuée par l'utilisateur.

3.2.

Modifiez le traitement précédent pour prendre en compte ce nouveau mécanisme.

3.3.

Lors du positionnement d'une alarme, l'instant de déclenchement peut être choisi à la micro-seconde près. Quelle est, en réalité, la précision du déclenchement ?

3.4.

Comment peut-on mesurer le temps passé par la tâche en mode utilisateur et en mode système ?