ENSEEIHT - 3SN
Contrôle de systèmes temps réel - 14 janvier 2022
Durée : 1 heure - Tous documents autorisés

Ordonnancement temps réel

Exercice 1

1. Soit la configuration de tâches périodiques indépendantes suivante :
   — $T_1 : C = 2, D = 3, P = 3$
   — $T_2 : C = 2, D = 8, P = 8$
   — $T_3 : C = 1, D = 24, P = 24$
   Est-elle ordonnançable avec un algorithme à priorités statiques ? Si ce n'est pas le cas, l'est-elle avec un algorithme à priorités dynamiques ?
2. Même question pour la configuration de tâches périodiques indépendantes suivante :
   — $T_1 : C = 2, D = 3, P = 3$
   — $T_2 : C = 2, D = 5, P = 8$
   — $T_3 : C = 1, D = 14, P = 24$

Exercice 2

Soit la configuration de tâches périodiques indépendantes suivante :
— $T_1 : C = 4, D = 10, P = 10$
— $T_2 : C = 8, D = 20, P = 20$
On y ajoute les deux requêtes apériodiques suivantes :
— $T_3 : r = 1, C = 2$
— $T_4 : r = 11, C = 1$
Quels sont les temps de réponse de ces tâches apériodiques

1. lorsqu'on les exécute en arrière-plan,
2. lorsqu'on utilise un serveur de scrutation de période 5 et de budget 1,
3. lorsqu'on utilise un serveur de ajournable de période 5 et de budget 1,
4. lorsqu'on utilise un serveur de sporadique De période 5 et de budget 1.

Exercice 3

Soit la configuration de tâches périodiques suivante, qui partagent les ressources R_1, R_2 et R_3 :

	r_0	WCET	D	P
T_1	4	3 : $\square$ R_1 $\square$	8	15
T_2	2	5 : $\square$ R_2 R_2 R_2R_3 $\square$	12	15
T_3	0	6 : $\square$ R_1 R_1 R_1R_3 $R_1R_2R_3$ $\square$	15	15

Cette configuration est-elle ordonnançable ?

Exercice 4

Soit la configuration de tâches périodiques indépendantes suivante :

	WCET	D	P
T_1	11	20	20
T_2	11	20	20
T_3	11	20	20

Cette configuration de tâches est-elle ordonnançable sur deux processeurs ? Justifier.

Systèmes opératoires Temps Réel

Exercice 5

1. Généralement, les systèmes d'exploitation classiques utilisent un ordonnancement de type tourniquet avec priorité. Pourquoi ne sont-ils pas adaptés pour gérer des tâches temps réel ?
2. Dans le système temps réel OSEK, tous les objets sont définis statiquement dans un fichier OIL.
   (a) L'ordonnanceur permet-il un comportement de type RM ? de type EDF ?
   (b) L'ordonnanceur permet-il la préemption de tâche ?
3. Dans la norme ARINC 653 sont définis deux types de ségrégation :
   — la ségrégation spatiale
   — la ségrégation temporelle
   (a) A quoi correspond la ségrégation spatiale ?
   (b) Comment est réalisée la ségrégation temporelle ?
   (c) Que garantissent ces 2 ségrégations ?
   (d) Pensez-vous que Linux (ou ses évolutions) propose ces ségrégations ? Si oui, quel(s) mécanisme(s) de Linux permet(tent) de les réaliser ?

Exercice 6

Nous souhaitons configurer OSEK pour exécuter les tâches définies dans l'exercice 3. Nous supposons qu'un « timer » génère une interruption de catégorie 2 toutes les millisecondes. La gestion de cette interruption est faite par une routine « hook » qui contrôle l'incrémentation du compteur de « ticks ». Le compteur de « ticks » est configuré de manière à s'incrémenter toutes les millisecondes.

1. Ecrire, en langage OIL, la configuration d'une alarme périodique, A1, associée au compteur *SysTimerCnt* et qui réveille la tâche T1. Les informations concernant la date de démarrage et la période de la tâche T1 sont indiquées à l'exercice 3.
2. Ecrire, en langage OIL, la configuration des tâches T1 en considérant qu'elle possède la priorités la plus faible du système (ce qui n'est peut-être pas le cas du calcul de priorité que vous avez appliqué à l'exercice 3).
3. Quelle(s) fonction(s) doit-on exécuter dans le code de T1 pour accéder à la ressource R1 ?
4. Le problème d'inversion de priorité peut-il survenir en OSEK ?