ENSEEIHT - 3TR - SRE
Contrôle de systèmes temps réel - 1 avril 2019
Durée : 1 heure - Tous documents autorisés

Exercice 1

Soit la configuration de tâches périodiques indépendantes suivante :

	WCET	D	P
T_1	2	5	5
T_2	2	3	10
T_3	3	8	20
T_4	5	18	20

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 ?

Exercice 2

Soit la configuration de tâches indépendantes suivante :

	Réveil	WCET	D	P
T_1	0	5	10	10
T_2	0	5	20	20
T_3	1	1		
T_4	9	2		
T_5	15	1		

Les tâches périodiques (T_1 et T_2) sont ordonnancées par Rate Monotonic.

1. Quels sont les temps de réponse des tâches apériodiques (T_3, T_4 et T_5) si elles sont ordonnancées par une méthode d'arrière-plan ?
2. Même question avec un serveur de scrutation de budget 1 et de période 4.
3. Même question avec un serveur ajournable de budget 1 et de période 4.
4. Même question avec un serveur sporadique de budget 1 et de période 4.

Exercice 3

Soit la configuration de tâches périodiques suivante, qui partagent les ressources R_1, R_2, R_3 et R_4 :

	r_0	WCET	D	P
T_1	5	3 : R_1	5	25
T_2	4	3 : R_3	8	25
T_3	3	6 : R_4 R_4 R_4 R_2R_4 R_2R_4	15	25
T_4	2	6 : R_3 R_3 R_3 R_3R_4 R_3R_4	22	25
T_5	0	6 : R_1 R_1R_2 R_1R_2 $R_1R_2R_3$	25	25

Cette configuration est-elle ordonnaçable ?

Exercice 4

Soit la configuration de tâches périodiques indépendantes suivante :

	WCET	D	P
T_1	15	25	25
T_2	40	50	50
T_3	50	100	100
T_4	20	200	200

Cette configurtion de tâches est-elle ordonnançable sur deux processeurs ?

Systèmes opératoires Temps Réel

Question 1 (Question de synthèse) :

1. 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 ?
2. Dans la norme ARINC 653, la notion de partition est centrale.
   (a) Comment l'exécution de ces partitions s'effectue-t-elle sur le calculateur avionique ?
   (b) Que se passe-t-il si deux processus d'une même partition souhaitent utiliser une même variable v ?
   (c) Même question en considérant deux processus de deux partitions différentes (un processus de chaque partition manipule les données d'une même variable v) ?

Question 2 (OSEK) :

La figure 1 décrit le fonctionnement, en langage C pour OSEK, de 2 tâches temps réel :
— La tâche TASK1 contrôle le mouvement d'un robot (avance, recule, tourne).

```
TASK(Task1)
{
  switch (state) {
  case FWD:
    if (obstacle==0) {
      /* En avant */
    }  else {
      top= 0;
      state=BWD;
    }
    break;
  case BWD:
    top++;
    /* En arriere */
    if (top == 100) {
        state=TURN;
        top=0;
    }
    break;
  case TURN:
    top++;
    /* Tourne */
    if (top == 50) {
      top=0;
      state=FWD;
    }
    break;
  TerminateTask();
}
```

```
TASK(Task2)
{
  if (distance_obstacle() < 30) {
    obstacle=1;
  }
  else obstacle=0;
  TerminateTask();
}
```

FIGURE 1 – Code C de 2 tâches

— La tâche TASK2 teste périodiquement la présence d'un obstacle.

La variable `obstacle` est partagée entre les 2 tâches.

1. Quel est le rôle de l'instruction `TerminateTask()` ?
2. Quel problème peut survenir lors de l'utilisation de la variable partagée `obstacle` dans le code ? Que faut-il faire, en OSEK, pour y remédier ?
3. Le problème d'inversion de priorité peut-il survenir en OSEK ?

OSEK possède un « timer » qui 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 ».

3. A quoi servent les routines « hook » ?
4. Ecrire, en langage OIL, la configuration d'un compteur nommé *SysTimerCnt* qui permet l'incrémentation d'un « tick » toutes les 3 ms, la période minimale d'une tâche étant d'un « tick ».
5. Ecrire, en langage OIL, la configuration d'une alarme périodique, *Alarm1*, associée au compteur *SysTimerCnt* et qui réveille la tâche `TASK1`. La période de cette alarme est 5 ticks et sa 1ère occurrence est produite à 1 tick.
6. Quelle est la période, en milliseconde, de l'alarme *Alarm1* ?

La période de la tâche `TASK1` est donnée à l'étape précédente. La tâche `TASK2` possède une période de 60ms.

7. Quelle est la tâche prioritaire ? Justifiez.
8. Ecrire, en langage OIL, la configuration des tâches `TASK1` et `TASK2` en considérant qu'elles possèdent les priorités les plus faibles du système.