Année 2012 - 2013

INF342 - COU
Contrôle de Connaissances
25 juin 2013
3h00 - Sans document – Barème indicatif

Veuillez répondre à cette partie sur une feuille séparée

I. Ordonnancement (6 points)

On rappelle que $3*(2^{1/3} - 1) = 0,779$.

task Task1 **is** **entry** Start (T0 : Time); **end** Task1;	**task** Task2 **is** **entry** Start (T0 : Time); **end** Task2;	**task** Task3 **is** **entry** Start (T0 : Time); **end** Task3;
task body Task1 **is** T : Time; **begin** **accept** (T0 : Time) **do** T := T0; **end;** **loop** T := T + 8.0; Compute (2.0); **delay** T - Clock; **end loop;** **end** Task1;	**task body** Task2 **is** T : Time; **begin** **accept** (T0 : Time) **do** T := T0; **end;** **loop** T := T + 12.0; Compute (4.0); **delay** T - Clock; **end loop;** **end** Task2;	**task body** Task3 **is** T : Time; **begin** **accept** (T0 : Time) **do** T := T0; **end;** **loop** T := T + 16.0; Compute (4.0); **delay** T - Clock; **end loop;** **end** Task3;

On considère une configuration de trois tâches périodiques et indépendantes comme décrite ci-dessus. La procédure Compute effectue un calcul dont la durée en seconde est indiquée comme paramètre d'entrée (on reprend ici la fonction utilisée en travaux pratiques).

Ordonnancement temps réel (2.5pts)

On souhaite appliquer l'ordonnancement Rate Monotonic Scheduling à ce groupe de tâches.

- Donner les périodes et temps de calcul des tâches. Donner le facteur d'utilisation U.
- Peut-on conclure que ce groupe de tâches est ordonnançable par Rate Monotonic Scheduling ? On appliquera le test d'ordonnançabilité et/ou le théorème de la zone critique. On détaillera le raisonnement dans les deux cas.

Ada pour le temps réel (1.5pts)

On souhaite que les tâches s'exécutent au niveau de priorité statique prévu par RMS. Les priorités s'étaleraient de Default_Priority + 1 .. Default_Priority + 3.

- Modifier le code ci-dessus pour que les tâches s'exécutent au niveau de priorité prévu.

Cependant le code ci-dessus pose problème en cas d'exécution dans un contexte temps réel.

- Identifier la construction inappropriée en justifiant et proposer une construction alternative.

Serveur de tâches apériodiques (2pts)

Rappeler le fonctionnement d'un serveur à scrutation et celui d'un serveur différé. Expliciter la différence entre ces deux serveurs.

Veuillez répondre à cette partie sur une feuille séparée

IV. Bus et Réseaux Temps Réel (2 points)

Question IV-1

Dans le cas de l'ordonnancement EDF, rappeler la formule qui permet de modifier la date de réveil r_i d'une tâche Ti caractérisée par les coefficients (r_i, C_i, D_i) de façon à prendre en compte les dépendances dues aux communications par message.
On appelle Δ^j_i le délai de transmission entre deux tâches i et j.
On appellera cette nouvelle date r^*_i.
Hypothèse : une tâche ne démarre que si elle a reçu tous les messages qu'elle attend et ne transmet des messages qu'à la fin de son exécution.

Question IV-2

Soit 4 tâches, ordonnancées EDF, caractérisées ainsi :

Tâche T_i	r_i	C_i	D_i
T1	0	1	2
T2	3	2	6
T3	0	1	4
T4	0	2	8

Une tâche ne démarre que si elle a reçu tous les messages qu'elle attend et ne transmet des messages qu'à la fin de son exécution. Ces tâches sont réparties sur 3 sites A, B et C,
comme indiqué par le tableau suivant :

Site A	Site B	Site C
Tâche T_1	Tâche T_2	Tâche T_3
		Tâche T_4

Sens d'émission des messages :
De T1 vers T3 et T4, de T2 vers T4.

- **a-** Calculer les dates de réveil r^*_i pour prendre en compte les dépendances entre ces tâches.
- **b-** Quelles doivent être les valeurs minimales (valeurs données en fonction des Δ^i_j) des échéances D_i pour que chacune des tâches T_i respecte la sienne ?

V. Java Temps Réel (2 points)

Question V-1

Soit une application temps réel multithread dans laquelle il y a un seul thread (TA) du type NoHeapRealTimeThread. Il consomme des données produites par un thread (TB) du type RealTimeThread. Quel dysfonctionnement peut se produire ? Pourquoi, donner un exemple.

Question V-2

Expliquez le fonctionnement de la méthode waitForNextPeriod, dans quel cas l'utilise-t-on ? Donner un exemple (les fautes de syntaxe ne sont pas importantes).

Veuillez répondre à cette partie sur une feuille séparée

VII. Tolérance aux Fautes (3 pts)

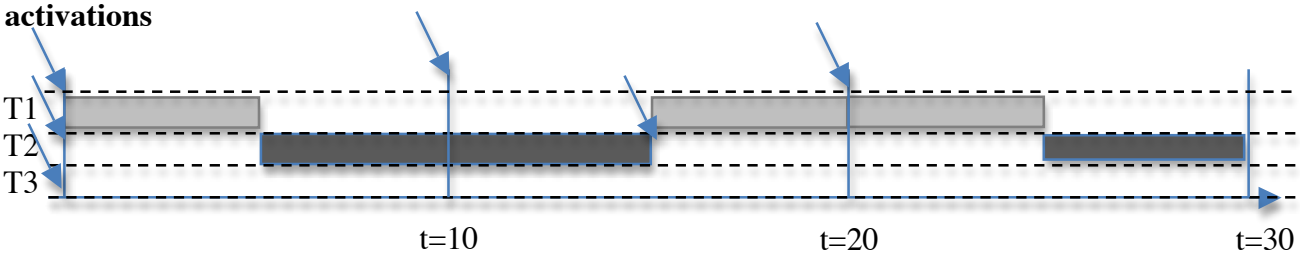
Nous considérons le cas d'un système d'exploitation implémentant un ordonnanceur à priorité fixe pour un modèle de tâches périodiques préemptibles (comportement similaire à OSEK) avec une tâche au plus par niveau de priorité. Le système est déployé sur un mono-cœur.
L'états des tâches vis à vis de l'ordonnanceur peut prendre les valeurs suivantes : DORMANT, READY, RUNNING. L'état DORMANT correspond à une tâche déjà terminée ou une tâche qui n'a pas encore été activée. L'état READY correspond à une tâche activée mais non exécutée en ce moment, l'état RUNNING correspond à une tâche en cours d'exécution.
Le lot de tâche suivant est déployé sur ce système :

Tâche	Période	Priorité	WCET
T1	10	3	5
T2	15	2	5
T3	30	1	4

Elément de spécification du comportement attendu :
Nous supposerons que la définition des périodes est une spécification du comportement attendu des tâches : chaque tâche est activée et complétée exactement une fois au cours d'une période.

Nous considérons que l'ordonnanceur est un composant dont la spécification exige qu'il implémente un ordonnancement à priorité fixe.

Question VII-1 Préciser en quoi le comportement décrit sur le chronogramme présenté ci-dessous permet de dire qu'une défaillance a eu lieu. Identifiez l'élément affecté (tâche, ordonnanceur) pour chaque observation permettant de conclure à une défaillance. (1,5 pt)



Question VII-2 Expliquer pourquoi l'état suivant correspondrait à une erreur de l'ordonnanceur s'il était observé à l'instant 9 ; en supposant que toute les tâches sont activées à t=0. (1 pt)

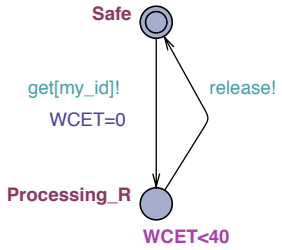
Tâche	T1	T2	T3
Etat	RUNNING	READY	READY

On supposera que l'ordonnanceur ne peut plus être sujet aux inversions de priorités.

Question VII-3 Sous quelles conditions T1 peut elle être encore en train de s'exécuter à t=9 ? Expliquez en quoi c'est un plus gênant que la situation décrite dans le chronogramme décrit en VII-1 (0,5 pt)

VIII. Méthodes formelles (2 pts)

Nous considérons l'automate temporisé suivant :



Question VIII-1 En supposant que WCET <40 est un invariant, donner un exemple de séquence d'événements temporisés qui ne peut pas être acceptée par cet automate. On supposera que my_id vaut 0.

Question VIII-2 Indiquer comment spécifier qu'il faut un temps minimal (par exemple 20) entre deux exécutions du cycle get[0] puis release.

Veuillez répondre à cette partie sur une feuille séparée

IX. Langage AADL (5 points) :

Les questions suivantes s'appuient sur le modèle AADL proposé sur la page suivante.

Question IX-1 (1,5 point)
Quelle(s) critique(s) pouvez vous formuler concernant l'expression du temps d'exécution deu *thread* ***t1*** ?

Question IX-2 (1,5 points)
Le process ***proc.impl*** contient un sous-composant ***d*** de type *data*. Il s'agit d'une variable globale à laquelle les *threads* ***th1*** et ***th2*** ont accès. La variable ***d*** correspondant au modèle doit-elle être protgée ? Est-ce le cas d'après les informations contenues dans le modèle ?

Question IX-3 (1 point)
Représenter l'arbre d'instances obtenu après instanciation du système ***s.impl***. Chaque nœud de l'arbre représente une instance, chaque arc représente le lien de contenant/contenu entre instances. Pour chaque instance, donner son type de référence (*type* ou *implémentation* en AADL). L'instance de ***s.impl*** sera nommée **root**.
Les connections entre threads sont-elles cohérentes ?

Question IX-4 (1 point)
Sur le *thread* ***t1***, nous avons un port d'événement entrant sur un *thread* périodique. Cette conception vous paraît-elle acceptable d'un point de vue sémantique, et pourquoi ?

```
data int
end int ;

thread t1
features
  s: in event port;
  da : requires data access int;
properties
 dispatch_protocol => periodic;
  period => 5 ms;
  priority => 2;
  Execution_Time =>1,5ms .. 2ms;
end t1;

thread t2
features
  s: in event port;
  da : requires data access int;
properties
  dispatch_protocol => aperiodic;
  period => 10 ms;
  priority => 5;
end t2;

thread t3
features
  s: out event port;
properties
  dispatch_protocol => periodic;
  period => 15 ms;
  priority => 4;
end t3;
```

```
process proc
end proc;

process implementation proc.impl
subcomponents
  th1: thread t1;
  th2: thread t2;
  d: data int;
connections
  cnx: port th1.s -> th2.s;
  cnx1 : access th1.da -> d;
  cnx2 : access th2.da -> d;
end proc.impl;

processor cpu
properties
  scheduling_protocol =>
Rate_Monotonic_Scheduling;
end cpu;

system s
end s;

system implementation s.impl
subcomponents
  c: processor cpu;
  p: process proc;
properties
  actual_processor_binding =>
reference(c) applies to p;
end s.impl;
```