Module STR
Examen session 2, juillet 2014
Systèmes temps réel
Durée : 1h20
Tous documents autorisés

Singhoff Frank
singhoff@univ-brest.fr
C-203

NB : ce document contient 2 feuilles recto/verso.

Exercice n^o 1 : Questions de cours (3 points)

– Question n^o 1.1 : 1,5 points

– Pourquoi modélise-t-on les fonctions d'un systèmes temps réel critique/dur par des tâches périodiques ?

– Question n^o 1.2 : 1,5 points

– On vous demande d'implanter une application temps réel dur/critique au sein d'un avion. Vous avez le choix entre un premier système d'exploitation fournissant un algorithme d'ordonnancement à priorité fixe et un second fournissant EDF. Lequel choisissez vous et pourquoi ?

Exercice n^o 2 : Ordonnancement temps réel (6 points)

Soit trois tâches périodiques définies par les paramètres suivants : C1=1, C2=1, C3=3, P1=4, P2=4, P3=7. Toutes les tâches arrivent à l'instant zéro. Les échéances sont égales aux périodes.

On suppose que l'on utilise un ordonnancement Rate Monotonic préemptif é à priorité fixe.

– Calculer le taux d'occupation du processeur et le nombre d'unités de temps libre sur la période d'étude. Que peux-t-on conclure sur l'ordonnançabilité du jeu de tâches ?
– Confirmer ou infirmer les résultats de la question précédente en dessinant sur la période d'étude, l'ordonnancement généré.
– Calculer le pire temps de réponse de chaque tâche.

Exercice n^o 3 : Langage Ada (5 points)

Soit le programme suivant :

```
with Ada.Text_IO;                use Ada.Text_IO;
with Ada.Numerics.Float_Random;  use Ada.Numerics.Float_Random;

procedure Exercise is

   Generator : Ada.Numerics.Float_Random.Generator;

   task type Do_Nothing (Letter : Character);
   task body Do_Nothing is
      Period : Duration;
   begin
      -- Tirage aleatoire d'une valeur entiere
      -- (duree d'attente)
      Period := Duration (Random (Generator)) / 10;
      if Letter = 'W' or Letter = 'X' then
         Period := 2 * Period;
      end if;
      for Count in 1 .. 10 loop
```

```
         Put (Letter);
         delay Period;
      end loop;
   end Do_Nothing;

   type A_Ptr is access Do_Nothing;

   procedure Do_Nothing_More is
      type B_Ptr is access Do_Nothing;
      X : A_Ptr;
      Y : B_Ptr;
   begin
      X := new Do_Nothing (Letter => 'X');
      declare
         Z : Do_Nothing (Letter => 'Z');
      begin
         for Count in 1 .. 10 loop
            Put ('-');
            delay 0.008;
         end loop;
      end;
      Y := new Do_Nothing (Letter => 'Y');
   end Do_Nothing_More;

   W : Do_Nothing (Letter  => 'W');

begin
   Put_Line ("Calling Do_Nothing_More");
   Do_Nothing_More;
   Put_Line ("Returned from Do_Nothing_More");
end Exercise;
```

Questions :

1. Combien a-t-on de tâches dans ce programme ?
2. Pour chaque tâche x du programme, indiquer quelle est la tâche mère qui créée x.
3. Pour chaque tâche, indiquer si elle est allouée statiquement ou dynamiquement.
4. Est on certain que rien ne sera affiché à l'écran avant l'affichage de la phrase *"Calling Do_ Nothing_ More"* ? Justifier votre réponse.
5. Est on certain que la dernière chose à être affiché à l'écran sera *"Returned from Do_ Nothing_ More"* ? Justifier votre réponse.
6. Est on certain que 'Y' sera affiché avant 'Z'. Justifier votre réponse.
7. Est on certain que 'Y' sera affiché avant 'X'. Justifier votre réponse.