UE STR
Examen session 2, 3 juillet 2017
Systèmes temps réel
Durée : 1h20 - 13 points
Tous documents autorisés

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

NB : ce document contient 2 feuilles recto/verso.

Exercice n^o 1 : Question de cours (2 points)

Expliquer ce qu'on entend par "inversion de priorité". Donner un exemple.

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

On se propose de modéliser un système informatique embarqué dans un véhicule automobile. On s'intéresse à une application offrant les fonctionnalités suivantes :

- Analyse de l'état du véhicule grâce à des capteurs afin d'assurer la sécurité des passagers.
- Pilotage du moteur. En particulier, en cas de danger, le processeur diminue la vitesse, voire stoppe automatiquement le véhicule.
- Rafraichissement du tableau de bord qui donne au conducteur l'accès aux informations produites par les deux fontionnalités ci-dessus.

On regarde les contraintes temporelles des trois traitements ci-dessus. Il s'agit en fait de trois tâches implantées sur un exécutif temps réel POSIX 1003 dont les niveaux de priorité vont de 0 à 31 (0 est le niveau le plus élevé). Sur cette machine seules les politiques SCHED_RR et SCHED_FIFO sont disponibles.

Les tâches sont définies par les paramètres suivants :

- La tâche $T1$ (analyse de l'état du véhicule) est une tâche périodique dont la période est de 10 milli-secondes. Son temps d'exécution est de moins d'une milli-seconde.
- La tâche $T2$ (contrôle du moteur) est une tâche périodique dont la période est de 15 milli-secondes et son temps d'exécution est de 5 milli-secondes exactement.
- La tâche $T3$ (rafraichissement du tableau de bord) est une tâche périodique dont la période est de 1000 milli-secondes et son temps d'exécution varie entre 50 et 150 milli-secondes.

Toutes les tâches arrivent à l'instant zéro et les échéances sont égales aux périodes.

Questions :

1. Quelles peuvent être les priorités de ces tâches sur cette machine POSIX si l'on souhaite affecter les priorités selon l'algorithme **Rate Monotonic**.
2. Sans ordonnancer le jeu de tâches, est il possible de savoir si les tâches respecteront leur échéance ? Si oui comment.
3. Ordonnancer ces tâches en mode préemptif sur les 50 premières milli-secondes.
4. En général, on suppose que les tâches arrivent au même instant (zéro). A votre avis pourquoi ?

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

On souhaite écrire un programme qui permet à un utilisateur de saisir une température. La température, une fois saisie, est transmise à deux tâches qui respectivement affichent la valeur courante de la température ainsi que les valeurs maximales et minimales. Les trois tâches réalisent

des accès exclusifs à l'écran (l'écran est successivement vérouillé et dévérouillé par les tâches lorsqu'elles souhaitent afficher une information).

Enfin, chaque affichage est numéroté afin de pouvoir reconstruire l'historique des afffichages après exécution de l'application. L'affichage numéroté est réalisé grâce à l'invocation de la procédure *Put* du paquetage *numeroted_io.ads*. Cette procédure affiche une String préfixée par un numéro d'ordre. Le programme en question est donné en annexe.

Ce programme contient 5 **types** d'erreur **différents** qui peuvent être
— soit des erreurs de compilation.
— soit des exceptions levées pendant l'exécution.
— soit des erreurs de synchronisation entre les tâches.
Décrivez ces erreurs et proposez une correction.

Annexe : le programme Ada à corriger

```
------------------------------ Fichier numeroted_io.ads  --------------
package Numeroted_Io is

   -- Procédure effectuant un affichage précédé d'un compteur
   -- afin de pouvoir ordonner les affichages réalisés en parallèle
   --
   procedure Put(
         S : in     String );

end Numeroted_Io;
------------------------------ Fichier temperature.adb  --------------
with Text_Io;
use Text_Io;
use Text_Io.Integer_Io;
with Numeroted_Io;
use Numeroted_Io;

procedure Temperature is

   -- Le type temperature
   --
   type Type_Temperature is new Integer range 0 .. 300;

   -- Interface des taches
   --
   task Verrou_Ecran is
      entry Verouille;
      entry Deverouille;
   end Verrou_Ecran;

   task Actuelle is
      entry Valeur (
            V : in     Type_Temperature );
   end Actuelle;

   task Min_Max is
      entry Valeur (
            V : in     Type_Temperature );
   end Min_Max;


   -- Corps des taches
   --
   task body Verrou_Ecran is
   begin
      loop
         accept Verouille;
         accept Deverouille;
      end loop;
   end Verrou_Ecran;
```

```
   task  body Actuelle is
   begin
      loop
         accept Valeur (
               V : in     Type_Temperature ) do
            Verrou_Ecran.Verouille;
            Put(" valeur actuelle = ");
            Put(V);
            New_Line;
            Verrou_Ecran.Deverouille;
         end Valeur;
      end loop;
   end Actuelle;

   task  body Min_Max is
      Ma : Type_Temperature := -1;
      Mi : Type_Temperature := 300;
   begin
      loop
         accept Valeur (
               V : in     Type_Temperature ) do
            if V > Ma then
               Ma:=V;
            end if;
            if V < Mi then
               Mi:=V;
            end if;
        end Valeur;

        Verrou_Ecran.Verouille;
        Put(" valeur max = ");
        Put(Ma);
        New_Line;
        Put(" valeur min = ");
        Put(Mi);
        New_Line;
        Verrou_Ecran.Deverouille;

      end loop;
   end Min_Max;


   Une_Temperature : Integer;

begin
   loop
      Verrou_Ecran.Verouille;
      Put("Saisir une temperature : ");
      Get(Une_Temperature);
      Actuelle.Valeur(Une_Temperature);
      Min_Max.Valeur(Une_Temperature);
      New_Line;
      New_Line;
      Verrou_Ecran.Deverouille;
   end loop;
end Temperature;
```