ENVIRONNEMENTS POUR LE TEMPS REEL EMBARQUE REPARTI

SPECIALITE SAR, UNIVERSITE PARIS VI

Barème indicatif, **sans document**
Les téléphones portables doivent être éteints et rangés dans vos sacs.

Les **5** parties de l'examen sont indépendantes.

Il sera tenu compte de la présentation et de la clarté dans la rédaction.
Seules les réponses précises et justifiées seront considérées.

I. Chaîne croisée (3pts ~ 15 min)

Question 1 (2 points)

Détaillez la séquence de démarrage usuelle d'une carte embarquée en précisant le rôle du moniteur et du bootloader.

Question 2 (1 points)

Un étudiant développe, sur une machine en salle de tp (linux-x86), un code destiné à être déployé sur un processeur ARM réalisant une transformée de Fourrier (un calcul complexe).

- Le code utilise la bibliothèque mathématique (libm.a -- version « statique »).
- Le code contient la directive #include <libm/math.h> indiquant que les entêtes de la bibliothèque seront recherchées soit dans le répertoire par défaut (/usr/local/include), soit dans la liste de répertoires spécifiée lors de l'édition de lien.
- Le makefile réalise l'édition de lien du programme grâce à la commande suivante : `gcc programme.o -static -lm -o prog`, il en découle que la bibliothèque sera recherchée dans /usr/local/lib/

Affirmation : L'étudiant n'arrivera pas à produire d'exécutable fonctionnel pour ARM dans ce contexte (en supposant qu'il y ait bien un fichier libm/math.h dans /usr/local/include, et un fichier libm.a dans /usr/local/lib).

Fournissez au moins une justification plausible étayant cette affirmation.

II. Problème Système Embarqué Temps Réel (12 pts ~ 1h)

A. Description du système

Le système étudié contrôle une unité de tri de courrier (système factice simplifié). Une phase préliminaire sépare les lettres et les « installe sur un guide ». Ce processus est totalement automatisé et sera considéré comme fiable. Il en découle qu'à l'entrée du système de tri, les lettres passent une par une à travers les différents blocs de traitement constituant la chaine de tri : un bloc de lecture/décodage de l'adresse, et un bloc de routage. Nous allons étudier le bloc de lecture/décodage.

Le bloc de lecture est en charge de décoder le code postal et le type de timbre présent sur chaque lettre. Ces deux informations permettent de déterminer si la lettre est à destination : a) de la France Métropolitaine, b) des DOM-TOM, c) de l'étranger. Ces informations (type de timbre, code postal) sont transmises périodiquement au bloc de routage par « paquet de 5 ». Le routage est en charge de contrôler un guide orientable (possédant trois positions) transférant les lettres dans trois baquets distincts. Les lectures du type de timbre et de l'adresse se font par des capteurs distincts contrôlés de manière indépendante.

Architecture Organique du bloc lecture/décodage :

• Un **unique processus P** regroupe les données partagées par les tâches constituant l'application de lecture/décodage.

- Les **variables** manipulées sont :
  - 3 entiers servant à compter les lettres lues, les adresses décodées, et les paquets transmis au bloc de routage : respectivement **rl**, **da**, et **ps**. (Valeurs initiales **rl=0**, **ra=0**, **ps=0)**.
  - 3 tableaux, chacun pouvant contenir 10 éléments entiers **stamp_type[], zip_code[]**, et **packet[]**.
    - Les deux premiers tableaux sont utilisés comme des « tampons circulaires » pour stocker les dix dernières valeurs lues pour le type de timbre et le code postal d'une lettre.
    - Le tableau **packet[]** permet de préparer le message contenant les types de timbres et les codes postaux de 5 lettres. De façon à tirer au maximum partie des tampons circulaires stamp_type et zip_code, ce sont les 5 valeurs les plus anciennes de stamp_type[] et zip_code[] qui sont recopiées dans packet[] (cf description tâche 4).
  - **Tous les éléments de chaque tableau sont initialisés à 0.**

- Le lot de tâches présent sur le bloc lecture/décodage est ordonnancé selon la politique **Deadline Monotonic**. Toutes les tâches sont **activées au même instant** (t0=0).

  - une tâche périodique (période 100 ms ; échéance relative 50 ms ; wcet 30 ms) nommée T1, qui récupère les données fournies par les capteurs:
    - Exécute int read_stamp_type() *// retourne 1 si l'envoi est pour la France, 2 sinon ; puis place la valeur de retour dans la variable stamp_type[rl modulo 10]*
    - Exécute int read_zip_code(int* pict) *// écrit dans le tampon pict les intensités mesurée sur une matrice de capteur de lumière indépendant (types CMOS) de 300 sur 400 – chaque « pixel » est représenté par un nombre entier.*
      // Note : Cette fonction possède un mécanisme de signalement de ses défaillances à travers son code de retour : 1 fonctionnement normal, 0 le capteur ne répond pas, -1 l'image est tronquée. Dans le cas où l'image est tronquée, les pixels manquants sont mis à 0.
    - Incrémente le compteur de lettres lues (entier rl).

  - une tâche périodique (période 100 ms ; échéance relative 80 ms ; wcet 20 ms) nommée T2, qui décode l'image capturée :
    - Exécute la fonction int decode_zip (int*pict) *// retourne le nombre correspondant au code postal décodé dans l'image pict. Suppose un contenu intègre de « pict »* ***(si l'image est tronquée la fonction peut ne pas terminer),***
    - Place la valeur retournée dans zip_code[da modulo 10]
    - Incrémente le compteur d'adresses décodées (entier da)

- une tâche périodique (période 100 ms ; échéance relative 10 ms ; wcet 5ms) nommée T3, qui contrôle le fonctionnement de la tâche T2 :
  - si rl>da,
    - arrêt de l'instance précédente de T2 (appel à stop_task(''T2''))
    - mettre -1 dans zip_code[da modulo 10],
    - affecter da à rl : da = rl ;
    - réactiver T2 pour qu'elle s'exécute dans la période en cours.

- une tâche périodique (période 500 ms ; échéance relative 500 ms ; wcet 20ms) nommée T4 qui envoie les informations sur le réseau à destination de la carte de contrôle de l'actuateur:
  - si ps==0, alors ps=ps+1 ; (exécution « triviale » lors de la première période)
  - sinon
    - recopie les valeurs stamp_type[((ps-1)*5+i) modulo 10] pour i variant de 0 à 4 dans packet[i],
    - recopie les valeurs de zip_code [((ps-1)*5+j) modulo 10] pour j de 0 à 4 dans packet[j+5]
    - exécute la fonction send_packet(packet) qui transmet via le réseau temps réel le contenu du tableau packet[].
    - Incrémenter ps : ps =ps+1 ;

- Le lot de tâches a été prouvé ordonnançable sous la condition du respect des WCETs.

B. Ordonnancement (1,5 points)

Question II.1 (1,5 points)
Calculez l'hyper période du processus P qui contient les tâches T1 T2 T3 et T4. Représenter sous la forme d'un chronogramme l'exécution de ces tâches, en supposant qu'elles s'exécutent toutes en consommant exactement leurs WCET respectifs (échelle recommandée 1 carreau 10 ms).

C. Modélisation AADL (4,5 pts)

En utilisant le langage AADL, nous avons modélisé partiellement le système de tri présenté en introduction du problème. Ce modèle est fourni à la fin du problème. Dans ce modèle, il manque la spécification des interfaces de la tâche T2 et la spécification du capteur de lumière. Dans les questions qui suivent, nous vous demandons de compléter cette modélisation pour les composants T2 et P.impl (recopier et compléter T2, mais ne recopier pas P.impl, définir sa section connections sera suffisant). Les aspects « séquences d'appels » ne seront pas modélisés.

Question II.2 (3 points)

a) Compléter la spécification de la tâche T2 (recopier et compléter le modèle du thread en précisant ses interfaces et ses propriétés)

b) Compléter la spécification du processus P (e.g. définir la section connexion)

c) **Quelle catégorie de composant AADL doit-on utiliser pour modéliser la « matrice de capteur de lumière »** évoquée dans la description de la tâche T1 **?**

Question II.3 (1,5 points)
En AADL, un sous-programme (subprogram) peut avoir

- un ou plusieurs paramètres d'entrée (in parameter) ;
- un ou plusieurs paramètres de sortie (out parameter) ;
- un ou plusieurs paramètres d'entrée-sortie (in out parameter) ;

Supposons que le prototype (signature) de la fonction read_zip_code() ait été proposé afin de représenter fidèlement la structure d'un subprogram AADL ayant un unique out parameter.

a) Expliquer en quoi ce choix d'utiliser la valeur de retour de la fonction C pour modéliser un *out parameter* est difficilement extensible pour implémenter un composant *subprogram* ayant plusieurs *out parameter* ou *in out parameter*.

b) Proposez un prototype de fonction C (signature), pour représenter le sous-programme suivant

```
subprogram operation
features
  a: in parameter int;
  b: out parameter int;
  c: in out parameter int;
  d: out parameter int;

end operation;
```

Justifiez en quoi la signature proposée permet de respecter la sémantique des « parameter » (on suppose le type int est défini dans le modèle AADL et qu'il est équivalent au type int en C).

Modèle AADL du cas d'étude « Système de tri postal »

```
data pic_type
subcomponents
  matrix: data int[300][400];
end pic_type;

data integer_array
subcomponents
  array: data int[10];
end integer_array;

thread T1
features
  stamp_type: requires data access
integer_array;
  pict: requires data access pic_type;
end T1;

thread T2
-- A COMPLETER: ajouter ici les
interfaces de T2.
properties
  Dispatch_Protocol => -- a completer
  Period => -- a completer
  Deadline => 80 ms;
  Execution_Time => -- a completer
end T2;

thread T3
features
  stop: out event port;
end T3;

thread T4
features
  stamp_type: requires data access
integer_array;
  zip_code: requires data access
integer_array;
  packet: requires data access integer_array;
end T4;

process P
end P;

process implementation P.impl
subcomponents
  T1_s: thread T1;
  T2_s: thread T2;
  T3_s: thread T3;
  T4_s: thread T4;
  pict: data pic_type;
  stamp_type: data integer_array;
  zip_code: data integer_array;
  packet:data integer_array;
connections

  -- A COMPLETER: ajouter ici les connections
  -- reliant les interfaces de T2 éléments du
  -- système .

end P.impl;
```

Seconde partie du problème sur la page suivante

D. Etude de sûreté de fonctionnement (6 pts)

Rappel :

L'interface d'une tâche est définie de la manière suivante (interface = état partagé avec l'environnement)

- les variables partagées stamp_type[], zip_code[], da, rl, …
- l'état du thread lui même (running, suspended, stopped, waiting_activation)

Une défaillance temporelle du thread se produit si le thread est dans l'état running à une date correspondant à l'une de ses échéances.

Cela correspond à dire que :

* Soit son WCET théorique n'est pas respecté
* Soit qu'une tâche plus prioritaire ne respecte pas son WCET

L'interface d'une fonction C est constituée de ses paramètres d'entrée et de sa valeur de retour (lorsque la fonction dispose d'un type de retour différent de void)

Un composant auto-testable est un composant qui contient une procédure documentée de signalement de ses défaillances

Question II.4 (1,5 points)

Identifier un composant auto-testable (parmi les fonctions C) de la description du système p2, détailler sa politique de signalement d'erreur ?

Question II.5 (2,5 points)

Pour ces questions il est recommandé d'utiliser (sans que cela soit nécessaire) le chronogramme de l'exécution des tâches du système (voir question II.1). Nous supposerons le temps d'exécution des tâches égal à leur WCET en l'absence de défaillances temporelles.

a) Expliquer en quoi la tâche T3 permet de détecter une défaillance temporelle de T2, lorsque la fonction decode_code (int*pict) boucle indéfiniment ?

b) Donner un exemple, où T3 ne « détecte » rien même si T2 dépasse son échéance

c) La tâche T3 met en place un mécanisme de recouvrement qui affecte l'état de T2. Ce type de mécanisme est qualifié soit de forward recovery, soit de backward recovery (recouvrement arrière ou avant).
Déterminer à quelle catégorie ce mécanisme appartient. Justifier la réponse.

Question II.6 (2 points)

La fonction int read_zip_code(int*pict) possède deux modes de défaillance distincts. Dans le cas d'une défaillance correspondant à une image tronquée (retourne -1), peut-on dire que la tâche T1 est une zone de confinement pour cette erreur ?

Justifier votre réponse par un contre exemple en cas de propagation à une autre partie de P, ou en indiquant quel élément de T1 assure le confinement.

I. Bus et réseaux temps réel (5 pts)

Question 1 (2 points)
Soit 3 tâches T1, T2 et T3 situées sur 3 sites S1, S2 et S3. Ces sites sont reliés par un bus unique, du type CAN (protocole : *bitwise AND)*.
Les priorités de T1, T2 et T3 sont 18, 23 et 19. Elles sont toutes activées au temps 1, dès leur activation, ces tâches essaient de prendre le bus pour émettre.
Donner un chronogramme sur lequel sont représentés : l'état du bus, les opérations effectuées sur le bus par chaque tâche ainsi que l'état de ces tâches par rapport au bus.

Question 2 (0,5 point)
Dans le cas de l'ordonnancement EDF distribué, 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. Expliquer la signification de chacun de ces paramètres. On appellera cette nouvelle date r^*_i.

Soit 5 tâches, ordonnancées par EDF et caractérisées ainsi :

Tâche T_i	r_i	C_i
T1	1	1
T2	3	2
T3	0	1
T4	0	2
T5	0	2

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 4 sites A, B, C et D.

La répartition des tâches sur les sites est donnée par le tableau suivant :

Site A	**Site B**	**Site C**	**Site D**
Tâche T_1	Tâche T_2	Tâche T_3	Tâche T_5
		Tâche T_4	

Sens d'émission des messages : de T1 vers T3, de T2 vers T4, de T3 vers T5, de T4 vers T5.

Question 3 (1,5 points)

Calculer les dates de réveil r^*_i pour prendre en compte les dépendances entre les tâches précédentes.

Question 4 (1 point)

1. Quelles doivent être les valeurs minimales (valeurs données en fonction des délais de transmission entre sites) des échéances D_i pour que chacune des tâches T_i respecte la sienne ?
2. Si les délais de transmissions entre (A, C) et (B, C) valent 1, quelle contrainte doit satisfaire le délai de transmission entre les sites C et D pour que T5 respecte son échéance ?