INSTN EXAMEN-SETI 30 /11/2021

Sans document, sans téléphone et sans calculatrice

V. David, S. Louise, F. Thomas

Cet examen (2h45) est constitué de trois parties indépendantes.

PARTIE A

Systèmes asynchrones

Question de cours (3 points)

- **Q1 :** Définir l'atomicité matérielle dans le cas monoprocesseur et dans le cas multiprocesseur à bus mémoire partagé.
- **Q2 :** Pour une machine multiprocesseur à bus mémoire partagé, présenter 3 cas d'instructions pouvant le cas échéant soulever un problème d'atomicité matérielle. Proposez une(des) solution(s) envisageable(s) pour résoudre ce type de problème d'atomicité matérielle pour une machine multiprocesseur à bus partagé.
- **Q3 :** Définir l'atomicité logicielle. Précisez l'impact possible de la durée d'une section atomique sur le fonctionnement d'un calculateur multiprocesseur à bus mémoire partagé.
- **Q4 :** Définir l'interblocage et la famine. Donnez des exemples et présentez une méthode pour garantir l'absence d'interblocage.

Sans document, sans téléphone et sans calculatrice

Problèmes : synchronisation avec les sémaphores (3 points)

Problème 1 :

Deux tâches T1 et T2 pilotent deux appareils qui utilisent la même aire de travail.

Q1 : Ecrire une synchronisation entre T1 et T2 qui garantisse que l'aire de travail n'est utilisée que par un seul appareil à la fois.

Problème 2 :

Trois tâches P, C1 et C2 pilotent trois appareils. L'appareil piloté par P dépose des pièces sur un plateau central fixe à 6 places. Les appareils pilotés par C1 et C2 prennent une pièce sur le plateau et la déposent sur un poste d'usinage sans limite de capacité.

Q2 : Ecrire les synchronisations entre P, C1 et C2 qui garantissent que quand P lance son appareil, il est certain qu'il y a une place disponible sur le plateau et que quand C1 ou C2 lancent leur appareil, il est certain qu'il y a au moins une pièce sur le plateau central. P, C1 et C2 doivent travailler en parallèle chaque fois que cela est possible.

Q3 : Le poste d'usinage a maintenant une limite de stockage de capacité égale à 10, les pièces étant enlevées par un quatrième appareil U. Modifier les programmes en conséquences.

Sans document, sans téléphone et sans calculatrice

PARTIE B

Modélisation et analyses fondées sur les réseaux de Petri (5 points)

Considérons le schéma d'une application d'aide à la conduite pour véhicule automobile suivante :

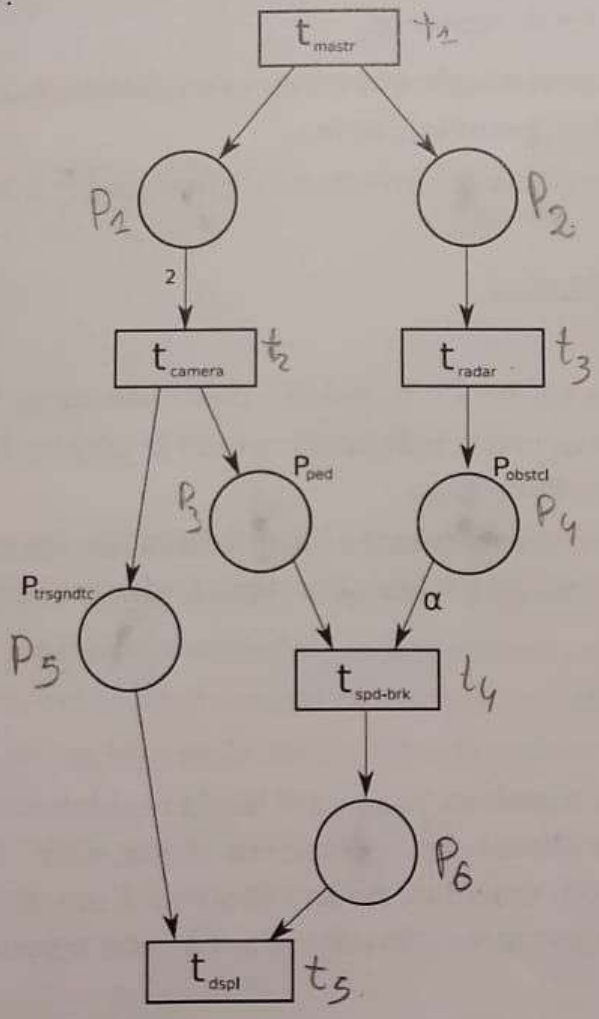

Illustration 1 : application ADAS

Sans document, sans téléphone et sans calculatrice

L'application nécessite un équilibrage des communications entre les modules. Pour cela il faut calculer le paramètre α.

Algèbre linéaire

Écrire la matrice d'incidence.

Calculer les invariants de marquage et en déduire la valeur du paramètre α pour équilibrer les communications.

Comment interpréter les invariants de marquages que l'on en déduit.

Graphe de marquage

Considérant la valeur de α déduite précédemment, faire deux tirage de la transition t_{mastr} et de cette configuration faire le graphe de marquage du graphe sans retirer la transition t_{mastr}.

Si la transition correspondant à l'acquisition caméra t_{camera} est faite à 60Hz, à quelle fréquence est fait l'échantillonnage Radar associé à la transition t_{radar} ?

Programmation

Considérons la transition $t_{spd\text{-}brk}$ qui fait la régulation de vitesse et le freinage d'urgence. En utilisant les primitives *begin_atomic()* et *end_atomic()* qui permettent respectivement de rentrer et sortir d'une section atomique, donner le squelette de programme correspondant à cette transition.

Sans document, sans téléphone et sans calculatrice

```
int main() {
        int i;
        pthread_t tid[3];
        for(i =0; i < 3; i++)
                pthread_create(&tid[i], NULL, myfunc, &i);
}
```

4. La fonction suivante est-elle réentrante ? Est-elle threadsafe ? Pourquoi ?

```
void swap(int *x, int *y) {
        static int t;
        t = *x;
        printf(« %i\n »,t);
        *x = *y;
        *y = t;
}
```

5. En utilisant la fonction POSIX atomic_compare_exchange_weak(volatile int* object, int *expected, int desired), écrivez les fonctions takeMyLock () et releaseMyLock() en langage C permettant respectivement de prendre et vendre un verrou nommé myLock. Vous écrivez la déclaration de ce lock et le contenu de ces deux fonctions.