Cours A7 : Temps Réel

Pierre.Paradinas / @ / cnam.fr
Cnam/Cedric
Systèmes Enfouis et Embarqués (SEE)

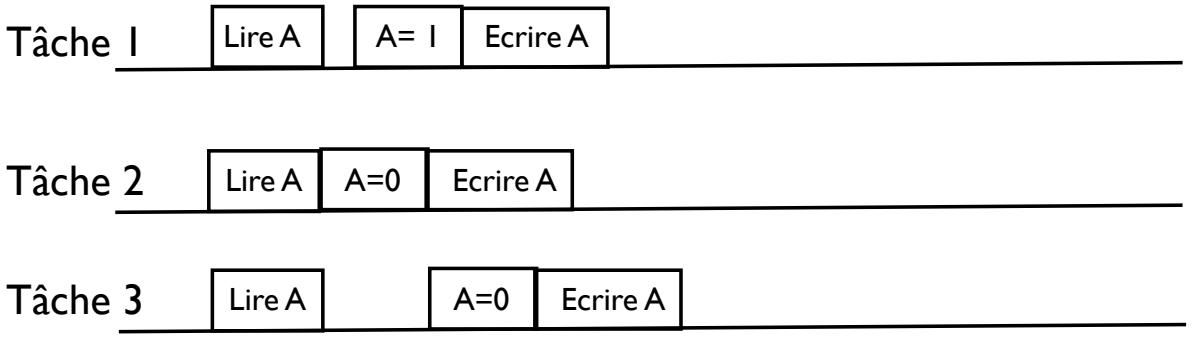
Motivations

- Du jour :
  - Partage des ressources,
  - L'exclusion mutuelle,
  - Synchronisation.
- Comprendre les mécanismes systèmes mis en oeuvre dans un système temps réel

Notre système

- Gestion de plusieurs tâches.
- Pour chaque tâche :
  - Contexte: son nom, les registres d'états, etc.
  - CO (compteur ordinal) et une zone mémoire de programme.
  - PP (pointeur de pile) et données temporaires.

Conflit d'accès



Que se passe-t-il pour A ?
Le résultat est-il ?
0 1 0
1 0 0
0 0 1

Définitions

- Section critique : on appelle section critique une séquence d'opérations qui doivent être exécutées sans être interrompues. On parlera d'actions atomiques.
- On dira que des tâches sont en exclusion mutuelle quand se partageant une ressource une seule tâche à la fois y accéder.

Les solutions

- Basées sur le matériel
- Basées sur le logiciel

Masquage matériel

- Par matériel les sections critiques sont “protégées”
  - Les interruptions sont interdites.
  - La préemption de code ou l’interruption de tâche en cours d’exécution sont interdits.
- Les sections critiques de code doivent être courte, c’est aussi le seul moyen de protéger une ressource et/ou un traitement utilisé par une interruption.

Masquage par verrou

- Si deux tâches veulent utiliser une même ressource elle peuvent convenir d'utiliser une variable (var_acces) pour signaler que la ressource est "occupée".

Masquage par verrou (l'algorithme)

```
Si (var_acces =0) alors
  var_acces = 1;
  acceder_ressource;
  var_acces = 0;
fsi
```

- Si interruption pendant cette séquence : que peut il se passer ?

Masquage par verrou (l'algorithme)

```
interdire_interruption;
Si (var_acces =0) alors
   var_acces = 1;
   autoriser_interruption
   acceder_ressource;
   interdire_interruption;
   var_acces = 0;
   autoriser_interruption;
fsi
autoriser_interruption;
```

Masquage logiciel

- Les SEs disposent de fonctions :
  - Début_Section : suspend toute préemption de tâche
  - Fin_Section : réautorise la préemption
- Néanmoins les interruptions continuent à être gérées normalement
- Attention au code contenu dans la section....
  - exemple

Masquage logiciel

```
Début_Section
  Print (”Hello world”);
  Attendre (clavier);
  Print (”clavier”);
Fin_Section
```

- Quand finit cette interruption ?
- Pas code bloquant dans une SC.

Masquage matériel/logiciel

- La gestion de sections critiques
  - Par matériel : tâches et interruptions sont traitées atomiquement
  - Par logiciel uniquement les tâches sont atomiquement.

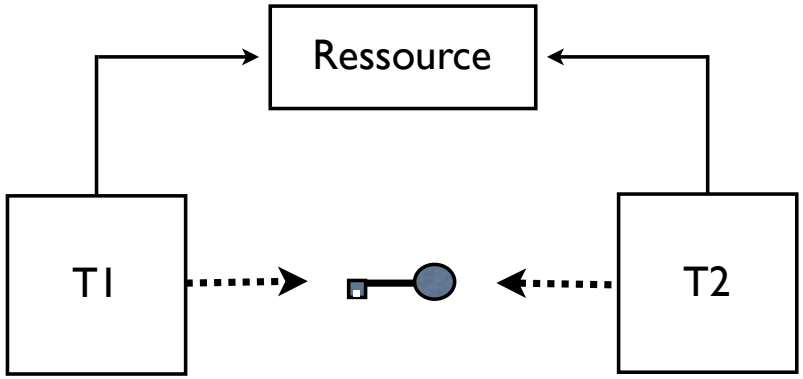
Les sémaphores

- Introduit en 1960 par Edgser Dijkstra.
- Protocole très utilisé dans les SEs multitâches.
  - contrôler l'accès à une ressource partagée.
    - exclusion mutuelle (mutex).
  - signaler des événements.
  - synchroniser des tâches.

Sémaphore binaire (idée)

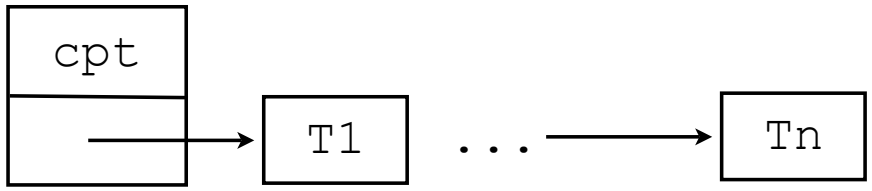
- Idée :
  - Le sémaphore est la clé pour accéder à la ressource,
  - Si je veux accéder à la ressource “je demande la clé”,
  - J’accède à la ressource, je rends la clé.
  - Si la clé est prise, j’attends la ressource pour y accéder.

Sémaphore binaire (schéma)



Sémaphore binaire (définition)

- Géré par le SE



- Fonction P et V de gestion du sémaphore
  - P est activé à l'entrée de la SC
  - V est activé à la sortie de la SC

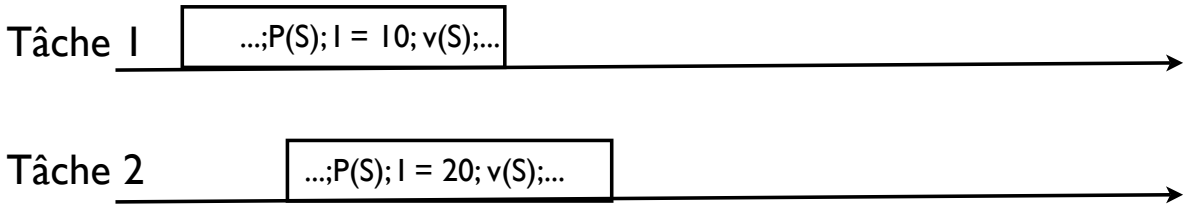
Sémaphore binaire (algorithme)

```
P
   cpt = cpt -1
   si cpt < 0 alors
      interrompre tâche;
      file <- ;
   sinon donner acces;
V
   cpt++;
   si cpt >= 0 alors sortir_tache_file;

Initialiser
   cpt = 1;
```

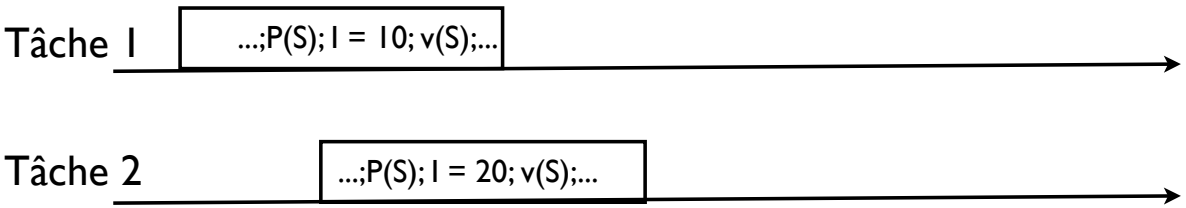
Exemple

Toutes les tâches ont P en entrée et V en sortie et partage un sémaphore S pour la ressource I.



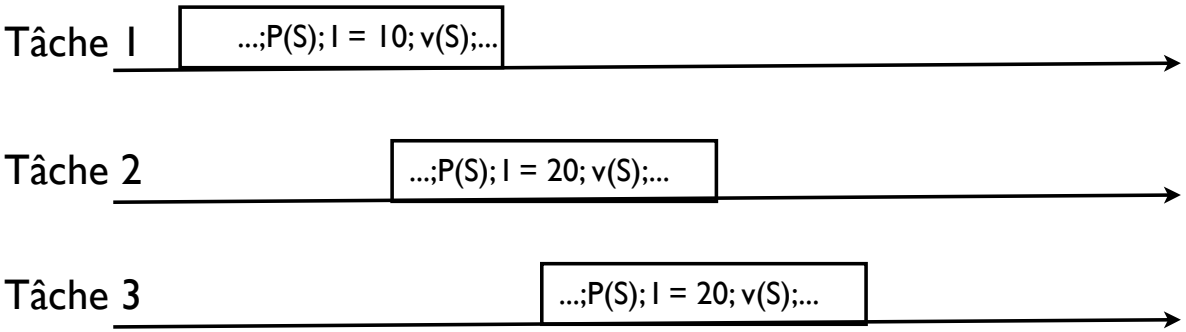
Donner le comportement du sémaphore S et de la file F

Exemple (suite)



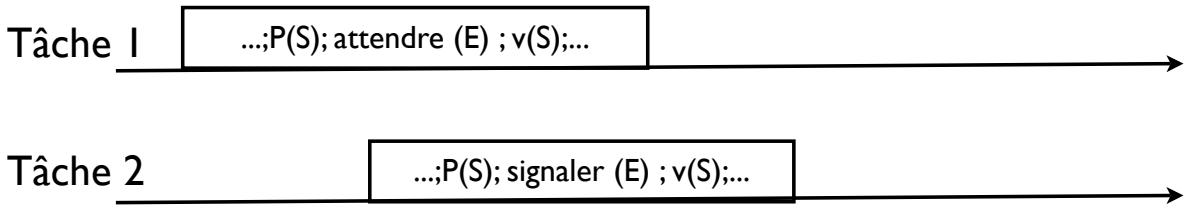
Init	S = 1	F = ∅
T1 (P)	0	∅
T2 (P)	-1	T2
T1 (V)	0	∅
T2 (V)	1	∅

Exemple (suite)



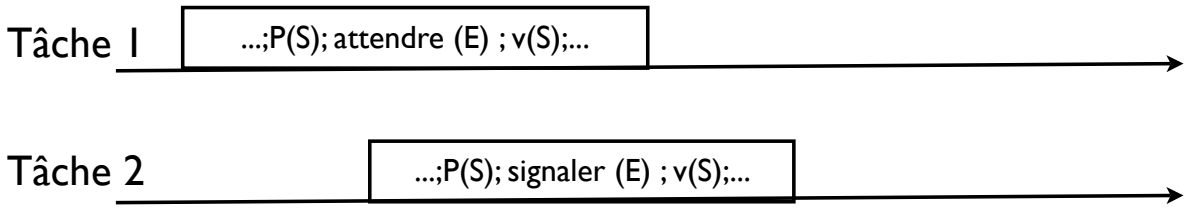
Init	S = 1	F = ∅
T1 (P)	0	∅
T2 (P)	-1	T2
T3 (P)	-2	T2, T3
T1 (V)	-1	T3
T2 (V)	0	∅
T3 (V)	1	∅

Etreinte fatale



- Que se passe -t-il dans ce cas ?

Etreinte fatale (suite)



- T2 est mise en attente
- T1 est en attente de T2
- Solution :
  - délai d'attente

Sémaphore à n positions

- On utilise un sémaphore à n positions (ou compteur) quand plusieurs tâches peuvent utiliser une même ressource
  - exemple : tampon d'impression
- Initialisation de S à la valeur n (nombre de positions)

Sémaphore à n positions (algorithme)

```
P
   cpt = cpt -1
   si cpt < 0 alors
      interrompre tâche;
      file <- tache
   sinon donner acces
V
   cpt++;
   si cpt >= 0 alors sortir_tache_file;
Initialiser
   cpt = n
```