Cours SET (Master 1 ST)

Chapitre 2 : Systèmes embarqués et temps réel

2.1 Introduction

Un système est dit temps réel si l'exactitude des applications ne dépend pas seulement du résultat mais aussi du temps auquel ce résultat est produit. Une réponse hors échéance est invalide Même si son contenu semble correct.

2.2 Gestion de la mémoire :

La gestion de la mémoire est le processus de contrôle et de coordination de la mémoire d'un système, elle réside dans le matériel, dans le système d'exploitation (OS) et dans les programmes et applications.

- Dans le matériel, la gestion de la mémoire implique des composants qui stockent physiquement des données, telles que des puces RAM (mémoire vive), des mémoires cache et des disques à mémoire flash (disques SSD).
- Dans le système d'exploitation, la gestion de la mémoire implique l'allocation (et la réallocation constante) de blocs de mémoire spécifiques à des programmes individuels lorsque les demandes des utilisateurs changent.
- Au niveau de l'application, la gestion de la mémoire garantit la disponibilité de la mémoire adéquate pour les objets et les structures de données de chaque programme en cours d'exécution à tout moment.

Lorsque le programme demande un bloc de mémoire, une partie du gestionnaire de mémoire appelée l'allocateur attribue ce bloc au programme.

Lorsqu'un programme n'a plus besoin des données dans des blocs de mémoire précédemment alloués, ces blocs deviennent disponibles pour la réaffectation. Cette tâche peut être effectuée manuellement (par le programmeur) ou automatiquement (par le gestionnaire de mémoire).

2.3 Gestion de la concurrence :

Très souvent, on doit exécuter plusieurs instructions simultanément, ceci pour plusieurs raisons, parmi ces raisons : la recherche de la diminution des temps de latence. Par exemple, si un programme contrôle des claviers ou des périphériques lents, il ne faut pas bloquer tout un système dans l'attente d'événements très peu fréquents. Une autre raison est qu'il existe des machines avec plusieurs processeurs de calcul, et il faut bien les programmer. De plus un syst embarqué doit pouvoir traiter l'exécution simultanée de plusieurs tâches.

Afin de gérer la concurrence, des notions de bases doivent être définit : processus, sémaphores, conditions.

- Processus : Un processus correspond à l'action de l'exécuter, il contient un texte décrivant une suite d'instructions. Quand il est question de plusieurs processus, la distinction devient importante et on remarque, par exemple, que plusieurs processus peuvent exécuter un même

programme. la fonction *interrupt* peut interrompre l'exécution d'un processus, La fonction *sleep* arrête l'exécution d'un processus.

- Conditions : représente la manière d'utiliser des techniques propres à chaque problème, par exemple utiliser les files d'attentes concurrente ; La fonction *ajouter* attend que la file se vide, la fonction *supprimer* attend que la file se remplisse.
- Sémaphores : Un sémaphore représente un ensemble de ressources et comprend, deux opérations :

  -prendre le sémaphore: P(s) : demande une ressource, et bloque l'appelant si aucune n'est disponible
  -libérer le sémaphore: V(s) : libére une ressource, jamais bloquant.

2.4 Linux pour l'embarqué :

Linux embarqué désigne un système d'exploitation basé sur Linux et adapté à un système embarqué. Ses fonctionnalités sont adaptées aux capacités du système embarqué : il requiert moins de mémoire, boot depuis une mémoire ROM (flash), il ne necessite pas de clavier ou de souris. - Une version de Linux embarqué peut être configurée « à la carte » pour fonctionner sur une plateforme donnée.

Linux possède d'autres atouts très importants pour les systèmes embarqués :

- Portage sur processeurs autres que x86 : PowerPC, ARM, MIPS, 68K, ColdFire, …
- Taille du noyau modeste compatible avec les tailles de mémoires utilisées dans un système embarqué (<500 Ko).
- Différentes distributions proposées suivant le domaine : routeur IP, PDA, téléphone…
- Support du chargement dynamique de modules qui permet d'optimiser la taille du noyau.

On utilise pour le développement sous Linux embarqué les outils traditionnels GNU :

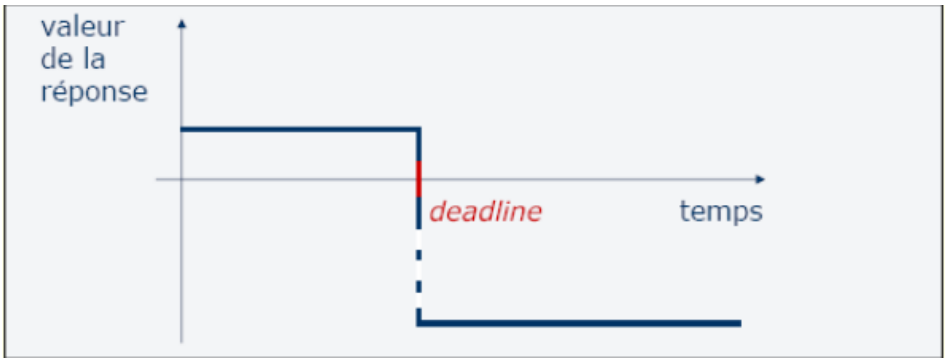
- (cross) compilateurs C/C++. C est préférable pour limiter la taille des exécutables.
- IDE (integrated development environment) (application logicielle qui permet de programmer)
- Debugger (GDB ou GNU). permet de déboguer un programme en cours d'exécution
- Simulateur.

2.4 Présentation des systèmes temps réel embarqués :

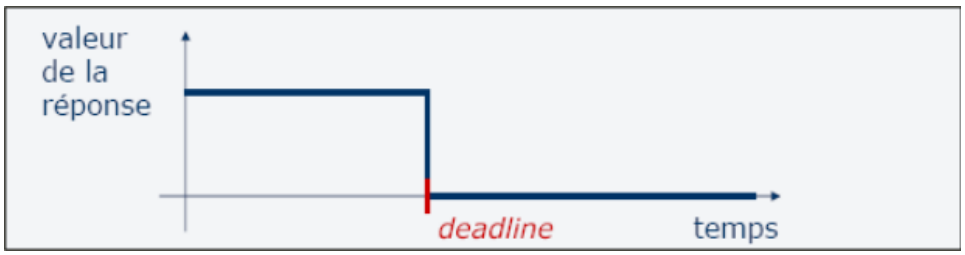
Un système temps réel n'est pas un système « qui va vite / rapide» mais un système qui satisfait des contraintes temporelles (les contraintes de temps dépendent de l'application et de l'environnement alors que la rapidité dépend de la technologie utilisée, celle du processeur par exemple).

On peut classer un système temps réel en trois catégories :

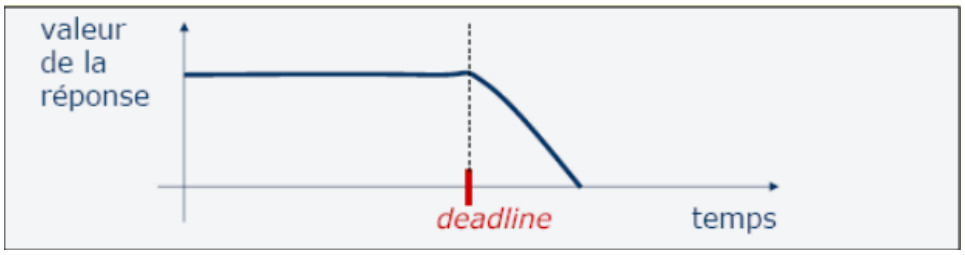
- **Temps-réel dur :** un retard dans l'obtention du résultat le rend inutile (Ex.: contrôle aérien, système de conduite de missile, etc.)



- **Temps-réel ferme :** un retard, ne sert plus à rien une fois le deadline passe.
  Exemple : transaction en bourse



- **Temps-réel mou :** un retard dans l'obtention du résultat n'est pas dramatique mais réduit progressivement son interet
  Exemple : logiciel embarqué du téléphone portable, iPod, ...etc



Classification des SETR (Système Embarqué Temps Réel)

- dans un système donné, des tâches temps réel dures et lâches peuvent cohabiter, éventuellement avec des tâches sans contraintes temporelles.
- un même système peut avoir des sous-systemes TR dur, TR-ferme ou TR-mou.
- critère de respect des contraintes temporelles :
  - booléen pour le temps réel dur : vrai/faux
  - complexe, doit être défini pour chaque tâche dans le cas d'une application temps réel lâche.

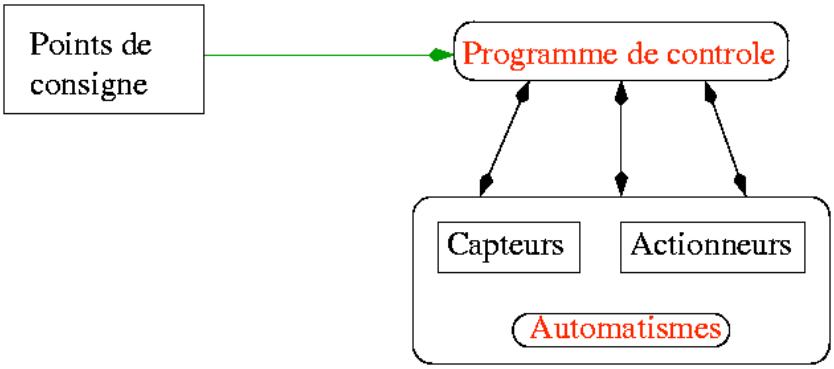
Un SETR doit être prévisible c'est-à-dire qu'il doit respecter ses contraintes temporelles, déterministe : c'est-à-dire le fonctionnement correcte dans l'exécution de l'ensemble du système et enfin fiable : c'est-à-dire qu'il doit être capable à s'exécuter aussi bien dans un environnement normal, hostile ou inattendu.

2.5 Structure et fonctionnement des systèmes temps réel embarqués :

Structure générale d'un SETR :

Boucle ouverte :

Boucle fermée :



Un système temps réel est une association logicielmatériel où le logiciel permet, entre autre, une gestion adéquate des ressources matérielles en vue de remplir certaines tâches ou fonctions dans des limites temporelles bien précises,

¬ La partie du logiciel qui réalise cette gestion est le système d'exploitation ou noyau temps réel,

L'architecture d'un système informatique temps réel est la suivante :