Real-time systems

Introduction, problématiques

Laure Gonnord

University of Lyon/ LIP

MIF18 - M1 optional course. Feb-June 2019

Plan

Introduction
Généralités

Langages pour le temps-réel

Systèmes pour le temps-réel

Lyon 1

Contexte - Système embarqué

Définition

An embedded system is a special-purpose computer system designed to perform one or a few dedicated functions. It is usually embedded as part of a complete device including hardware and mechanical parts. (Wikipedia)

- Industrie légère : téléphones portables, appareils ménagers
- Industrie lourde : aéronautique, aérospatiale, ...

Contraintes

- Gestion d'un système physique dans son environnement.
- Contraintes d'échéances : molles/dures.
- Importance de l'optimisation (mémoire, énergie).

Lyon 1

Contenu du cours

9H de cours :

- Contraintes d'un système temps réel.
- Système temps réel : ordonnancement, gestion mémoire, synchronisation.
- Méthode synchrone pour la conception des systèmes temps réels.

21h de TP : **Arduino, ordonnancement, plus ou moins haut niveau . . .**

Lyon 1

Plan

Introduction
Généralités

Langages pour le temps-réel

Systèmes pour le temps-réel

Lyon 1

Credits

Intro honteusement pompée sur les transparents
`http://beru.univ-brest.fr/~singhoff/supports.html`

Définition

"En informatique temps réel, le comportement correct d'un système dépend, non seulement des résultats logiques des traitements, mais aussi du temps auquel les résultats sont produits. "[sta88]

Différentes notions de déterminisme

Le **déterminisme** est une notion-clef :

- Déterminisme logique : les mêmes entrées appliquées au système produisent les mêmes résultats.
- Déterminisme temporel : respect des contraintes temporelles (échéances, rythme, . . .).

Lyon 1

Les contraintes de temps

Important Un système temps réel n'est pas un système "qui va vite" mais un système qui satisfait à des contraintes temporelles.

Quelques ordres de grandeur :

- La milliseconde pour les systèmes radar.
- La seconde pour les systèmes de visualisation humain.
- qq heures : production chimique
- ...

Des contraintes plus ou moins dures

Le besoin en **garantie de service** (niveau de respect des contraintes) peut être différent :

- Systèmes temps réel dur ou critique.
- Systèmes temps réel souple.

Exemple 1 : domaine de l'avionique

Système temps réel critique :

- Contraintes temporelles : temps de réponse, échéance, date d'exécution au plus tôt, cadence, etc.
- Dimensionnement au pire cas et réservation des ressources.
- Utilisation de redondance matérielle et logicielle.
- Matériel et logiciel dédiés. Système fermé, validé a priori.
- Système réparti synchrone : commandes de vol, radars, moteurs, etc.

Exemple 2 : multimédia sur le Web

Système temps réel souple :

- Contraintes temporelles : gigue, délais de bout en bout, temps de réponse. Synchronisations intra et inter-flux.
- Plate-forme généraliste. Non déterminisme temporel à cause du matériel et du logiciel (ex : PC + windows).
- Application interactive.
- Nombre de flots inconnu.
- Débits variables et difficiles à estimer hors ligne.

Lyon 1

Autres exemples / domaines d'activité

- Transports (métro, aérospatiale, SIG : systèmes d'info géographique et systèmes de régulation automobile).
- Médias (décodeurs numériques openTV).
- Services téléphoniques (téléphone mobile, auto-commutateur).
- Supervision médicale, écologique.
- Système de production industriel : centrale nucléaire, chaîne de montage, usine chimique.
- Robotique (ex : PathFinder)

Lyon 1

Plan

Introduction
Généralités

Langages pour le temps-réel

Systèmes pour le temps-réel

Lyon 1

Langages de bas niveau

C/C++, Assembleur :

- Largement diffusés et utilisés à ce jour.
- Accès direct aux ressources de bas niveau. Idéal pour les E/S.
- Doit être couplé avec les services du système (pour synchronisation, ordonnancement) ▶bibliothèques.
- Langage généralement restreint (sous-partie ayant un comportement temporel déterministe facile à évaluer).
- Peu adapté aux logiciels complexes et/ou volumineux.
- Pas de normalisation, donc peu de portabilité : logiciel “ad hoc”

▶On pratiquera un peu.

Lyon 1

Un exemple de langage haut niveau

Ex : Ada 2012 [zaf99] Langage conçu, entre autres, pour le support des applications temps réel :

- Abstractions temps réel : tâche, interruption, ordonnancement (priorité fixe et dynamique), synchronisation (sémaphore), timer et gestion du temps, outils de communication (rendez-vous).
- Interface/syntaxe et COMPORTEMENT du langage normalisés par l'ISO ▶forte portabilité.
- Compilation séparée. Typage fort. ▶Fiable. Adapté à la production de logiciels volumineux.

▶Langage complexe, que nous n'utiliserons pas (dommage).

Lyon 1

Une autre solution : l'approche synchrone

Ex : Lustre [halb91]

- Séparation des problématiques : fonctionnel/temps.
- Paradigme de programmation plus simple : ordonnancement calculé à la compilation, communication mémoire partagée.
- La gestion du temps est faite à postériori (horloges physiques).
- Langage haut niveau avec peu de primitives de construction. Compilation séparée, typage.
- Générateur de code C/C++ ▶portabilité intéressante
- Un écosystème avec des outils de vérification formelle.

▶On pratiquera aussi.

Lyon 1

Plan

Introduction
Généralités

Langages pour le temps-réel

Systèmes pour le temps-réel

Lyon 1

Problématique

Objet : montrer que réaliser un système temps-réel (matériel+logiciel) est un **cauchemard**.

Les systèmes généralistes à temps partagé 1/2

Objectifs de tels systèmes :

- Faciliter l'accès aux ressources.
- Masquer les ressources (process, mémoire, disques).
- Recherche de l'équité, maximisation du débit global.

Les systèmes généralistes à temps partagé 2/2

Deux styles de programmation :

- Mono-programmation : interaction synchrone entre le couple processeur/mémoire et les périphériques.
- Multi-programmation : interaction asynchrone. Le processeur profite du temps ainsi libéré pour effectuer d'autres traitements.

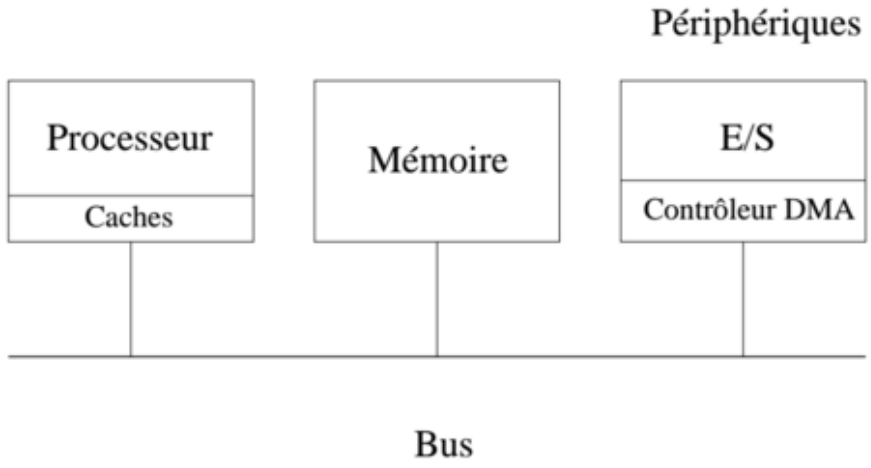
Lyon 1

Ordonnancement sous Linux

Les soucis des ordonnanceurs généralistes :

- Pas de prise en compte de l'urgence ou de contrainte temporelle.
- Politique souvent opaque.
- Temps de réponse inconnue.

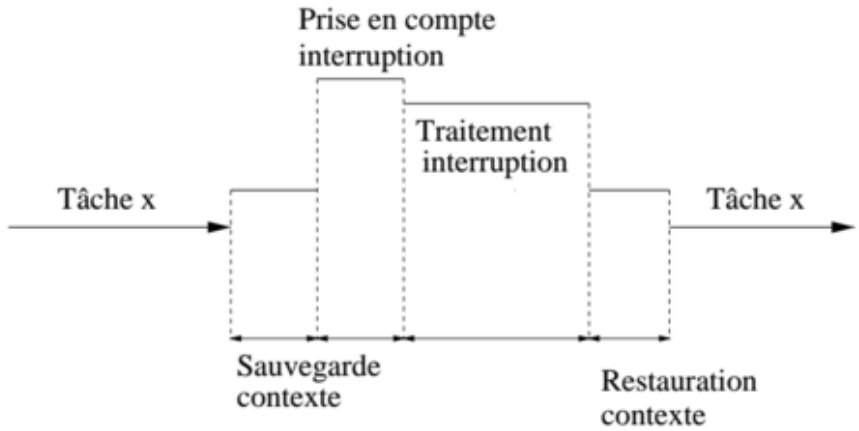
Les entrées/sorties 1/2



Modes d'échanges :

- Scrutation active de l'unité : polling. **Monopolise le processeur**. Comportement temporel déterministe.
- Interruptions.
- Accès direct à la mémoire (DMA ; Direct Memory Access). Contention sur le bus. Comportement temporel **indéterministe**.

Les entrées/sorties 2/2



Interruptions et préemptivité :

- Priorité des interruptions non modifiable par le concepteur.
- Nombre d'occurrences inconnu. Acquisition et traitement successifs ▶long, non déterministe.
- UNIX = verrouillage important, peu préemptif.

▶Il faut découpler acquisition et traitement des interruptions, **source d'erreur**. Attention au choix du matériel.

Problématiques d'accès aux ressources

Synchronisation, communication et accès aux ressources effectués grâce à des sémaphores :

- Sémaphore = compteur entier/booléen + file d'attente.
- Utilisation : exclusion mutuelle, paradigme classique de coopération (producteur/consommateur, lecteur/rédacteur).

Mais :

- Certaines tâches sont bloquées ▶Impact sur l'ordo.
- **inversion de priorité** : tâche de + faible prio bloquant une tâche de plus forte prio pendant une durée inconnue.

Lyon 1

Conclusion

Les systèmes généralistes ne sont pas **temps-réel** :

- Ils ne sont pas déterministes : matériel / logiciel.
- L'ordonnanceur temps partagé n'offre aucune garantie.
- (linux) le noyau est non préemptif (donc si un autre processus + prio a besoin du processeur FAIL)

▶POSIX 1003.1b sous Linux pour systèmes TR souples

▶Systèmes dédiés pour systèmes TR durs

▶Entre les deux : utiliser un **micronoyau temps réel** cohabitant avec le noyau linux (ex RTLINUX).

Lyon 1

Bibliography I


Nicolas Halbwachs, Paul Caspi, Pascal Raymond, and Daniel Pilaud.
The synchronous data flow programming language lustre.
Proceedings of the IEEE, 79, 1991.


John Stankovitch.
Misconceptions about real-time computing.
Proceedings of the IEEE, 21, 1988.


Luigi Zaffalon and Pierre Breguet.
Programmation concurrente et temps réel avec ADA 95.
Presses polytechniques et universitaires romandes.