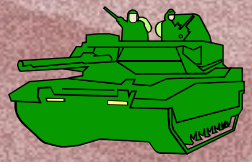
Sistemas de Tempo Real

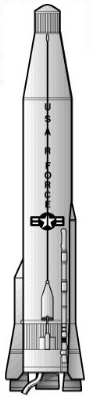
Docente

Paulo Pedreiras
pbrp@ua.pt
http://www.ieeta.pt/~pedreiras

Adaptado dos slides desenvolvidos pelo Prof. Doutor Luís Almeida
para a disciplina "Sistemas de Tempo-Real"

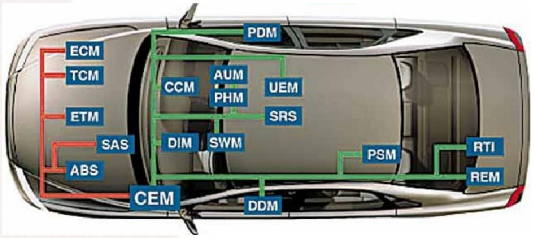


Noções preliminares



Afinal o que são “Sistemas de Tempo Real” ??!

- Sistemas **computacionais**
- Estão sujeitos a um **tempo real**
  tempo que progride continuamente e durante o qual *o mundo* prossegue ao seu ritmo próprio
- Aqueles em que não se pode dizer
  Ó tempo, volta p’ra trás...
- Ou por outras palavras, aqueles em que
  O que está feito, feito está! E há consequências...
- Por isso, a única forma de funcionarem corretamente
  É fazendo certo no instante certo !

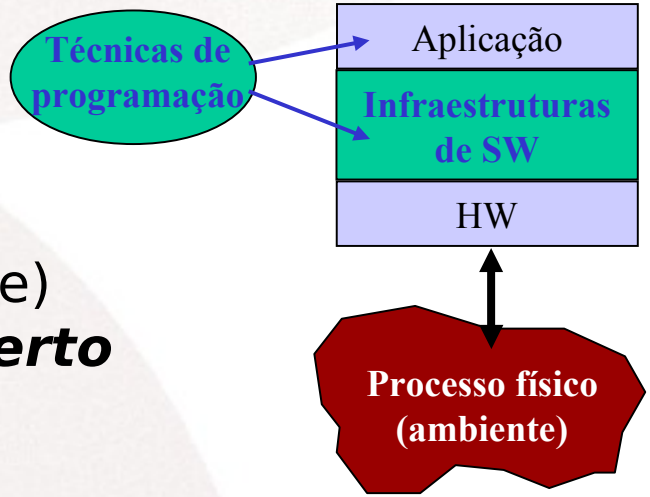


Objectivo da disciplina

Tema principal:

- Infraestruturas de software e técnicas de programação para sistemas que interagem com (ou simulam) um processo físico (ambiente)

para que façam certo no instante certo



Pretende-se abordar:

- a origem e caracterização das restrições impostas pelo ambiente ao comportamento temporal do sistema computacional;
- a forma como o sistema computacional mantém o conhecimento do estado do ambiente que o rodeia;
- a teoria de escalonamento de atividades concorrentes associadas a processos de tempo-real;
- e a constituição e construção de sistemas operativos / executivos de tempo-real.

FAQ

Não bastaria usar um processador rápido? ;

- Se for para executar um programa com estrutura trivial (tipo um único ciclo infinito), é provável que sim. Se o computador tiver de executar várias tarefas em simultâneo, a rapidez de processamento já não basta. Umas tarefas podem bloquear outras e causar atrasos demasiado grandes ou mesmo imprevisíveis.

Então o que é necessário?

- Escalonamento! que é como quem diz, ordenação correta das tarefas a executar. Existem critérios de ordenação que nos permitem restringir e determinar os atrasos máximos que as tarefas poderão sofrer.

FAQ

Mas isso então só se aplica quando é necessário *multi-tasking*... ?

- Conforme foi dito atrás, estamos a considerar situações em que um computador tem de executar várias tarefas simultaneamente. Será normal que nessas situações se utilize um sistema operativo (ou apenas um *kernel*) *multi-tasking.* Mas, muitas vezes, mesmo quando o corpo principal do programa é um simples ciclo infinito, existem várias tarefas encapotadas dentro de rotinas de interrupção assíncronas, o que leva à mesma situação. O disparo de rotinas de interrupção também pode ser atrasado, ou até descartado. É necessário usar técnicas adequadas para restringir e determinar esses atrasos.

FAQ

E esses atrasos são assim relevantes?

- Bom, se estivermos a falar de sistemas de controlo, e se os atrasos forem tais que levam à perda de amostras, é provável que se perca o controlo! Se isso acontecer num avião... ou num carro com atuação eletrónica (X-by-wire)... ou num robô que se movimenta perto de outros equipamentos e pessoas... ou num foguetão... haverá danos graves! Por outro lado, se estivermos a falar de sistemas multimédia, desde consolas de jogos a DVDs, ou de *routers* em redes de computadores, atrasos nas tarefas não provocarão a morte a ninguém mas haverá uma perda de Qualidade-de-Serviço.

Bibliografia

Preferencial

- G. Buttazzo (1997). ***Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications***. Kluwer Academic Publishers.
- H. Kopetz (1997). ***Design Principles for Distributed Embedded Applications***, Kluwer Academic Publishers.

Complementar (disponível na biblioteca da UA)

- P. Veríssimo and L. Rodrigues (2001). ***Distributed System for Systems Architects***. Kluwer Academic Publishers.
- Jane W.S. Liu (2000). ***Real-Time Systems***. Prentice Hall.
- Briand, L. and Roy, D.M. (1999). ***Meeting Deadlines in Hard Real-Time Systems: the Rate-Monotonic Approach***. IEEE Computer Society Press, Los Alamitos (CA), USA. (cont)

Bibliografia

Complementar (cont.)

- Stankovic, J. *et al.* (1998). ***Deadline Scheduling for Real-Time Systems: EDF and Related Algorithms***. Kluwer Academic Publishers.
- Krishna, C.M. and K. Shin (1997). ***Real-Time Systems***. McGraw-Hill.
- N. Nissanke (1997), ***Real-Time Systems***, Prentice-Hall.
- Laplante, P.A., ***Real-Time Systems Design and Analysis - An Engineer's Handbook*** *(2nd ed.).* IEEE Press, 1997.
- Welling, A. and A. Burns (1996). ***Real-Time Systems and Their Programming Languages*** *(2nd ed.).* Int. Computer Science Series, Addison-Wesley.
- Klein, M. *et al.* (1993), ***A Practitioner's Handbook for Real-Time Analysis: Guide to Rate-Monotonic Analysis for Real-Time Systems***. Kluwer Academic Publishers.

Organização das aulas

- ***Componente teórica*** **-** apresentação e discussão dos conceitos e técnicas
  - É recomendável a leitura de partes específicas dos livros aconselhados
  - Os slides serão disponibilizados no site da disciplina
  - Apresentações e discussão de trabalhos de pesquisa

- ***Componente práticas*** **-** aplicação das técnicas abordadas em casos concretos
  - Trabalho individual ou em grupos de dois elementos
  - Aulas tutoriais para estabelecer uma base experimental comum (Linux (GPOS), SHaRK e/ou RTAI)
  - Um mini-projeto por grupo:
    - Propostas a divulgar em breve; sugestões são bem-vindas

Regras de Avaliação

- **Classificação** final da disciplina:

Época normal

- Teórica - 50% : 40% teste único e 10% participação nas sessões de discussão (5% pelo docente e 5% pelos pares)
- Prática – 50% : {25% mini-projeto, 5% livro de registo, 10% apresentação} + 10% para trabalho complementar aos tutoriais

Época de recurso:

- Teórica - 50%
  - teste único, conta o melhor em relação à época normal
- Prática – 50%
  - nota correspondente da época normal ou teste prático

Obs: NOTA MÍNIMA de qualquer componente de 7.0 valores

Programa e planificação das aulas

Sistemas Tempo-Real

Ano letivo 2011/2012

Planificação das aulas

15 de setembro

Aula 0+1: apresentação da disciplina + conceitos básicos e requisitos de sistemas tempo-real

22 de setembro

Aula 2: modelos computacionais + tutorial GPOS

29 de setembro

Aula 3: kernels + tutorial GPOS (conclusão)

6 outubro

Aula 4: escalonamento (conceitos básicos) + tutorial SHaRK

13 outubro

Aula 5: escalonamento periódico FP + tutorial SHaRK

Programa e planificação das aulas

Planificação das aulas (continuação)

20 outubro
Aula 6: escalonamento periódico EDF + tutorial SHaRK (conclusão)
27 outubro
Aula 7: recursos partilhados + tutorial RTAI
3 novembro
Aula 8: escalonamento de tarefas aperiódicas + tutorial RTAI (conclusão)
10 novembro
Aula 9: outros aspetos de escalonamento TR + mini-projeto
17 novembro
Aula 10: otimização + mini-projeto
24 novembro
Aula 11: Mini-projeto
15 dezembro
Aula 12: Apresentação dos mini-projetos + considerações finais

E agora ....

É tempo de iniciar o trabalho!