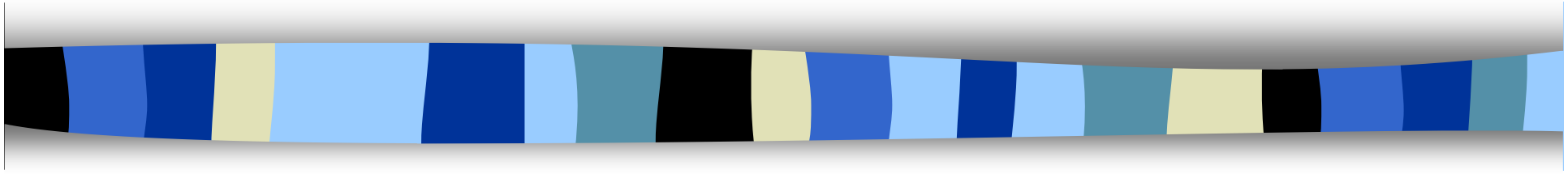
Universidade Federal do Amazonas
Departamento de Eletrônica e Computação

Sistemas de Tempo Real



Lucas Cordeiro
lucascordeiro@ufam.edu.br

Notas de Aula

- Estes slides são baseados nos livros:
  - Prof. Alan Burns da Universidade de York : http://www-users.cs.york.ac.uk/~burns/
  - Prof. Alan Shaw da Universidade de Washington: http://www.cs.washington.edu/people/faculty/shaw/
  - Prof. Joost-Pieter Katoen da Universidade de Aachen: http://www-i2.informatik.rwth-aachen.de/~katoen/

- Os slides estão disponíveis em: **http://home.ufam.edu.br/lucascordeiro/ptr/**

Objetivo do Curso

- Fornecer aos alunos, **conhecimento teórico** para a elaboração de **projetos** e **verificação** de sistemas de tempo real

- Dominar as peculiaridades destes sistemas, sendo capazes de analisar e projetar, de forma confiável, **sistemas de hardware e software** que possuam tais características

Ementa do Curso (1)

- Introdução: o mundo dos sistemas de tempo real
- Arquiteturas de software para sistemas de tempo real
- Especificações de requisitos e de projeto de tempo real
- Sistemas de máquinas de estados
- Especificações declarativas
- Predição de tempo de execução do pior caso (WCET)
- Escalonamento de tarefas em sistemas em tempo real
- Sincronização de processos concorrentes
- Acesso a periféricos
- Gerenciamento de entrada e saída (E/S)
- Linguagens de programação que atendem às especificidades de tempo-real

Ementa do Curso (2)

- Introdução a Verificação de modelos
- Verificação explícita e simbólica
- Lógica temporal linear e de árvore de computação
- Grafos de fluxo de controle
- Semântica de programas
- Satisfação booleana
- Teorias do módulo da satisfação
- Exemplo de aplicações

Conteúdo da Avaliação

- **Lista de exercícios:** Ao final de cada capítulo
- **Prova parcial:** Introdução aos sistemas de tempo real; projetando sistemas de tempo real; desenvolvimento de pequenos sistemas de tempo real; desenvolvimento de grandes sistemas de tempo real; escalonamento; algoritmos clássicos; tarefas periódicas; exclusão mútua e programação concorrente
- **Seminários:** Apresentação de seminários referente a um artigo recente relacionado ao tópico de sistemas de tempo real
- **Projetos:** Desenvolvimento de uma aplicação realística de tempo real

Conteúdo da Avaliação

- **Prova Final:** Todo o conteúdo da disciplina incluindo os seminários.

$$\text{Média Parcial (MP)} = \frac{2 \times \text{NPP} + \text{NS} + \text{NP}}{4}$$

$$\text{Média Final (MF)} = \frac{2 \times MP + PF}{3}$$

$$\text{NPP} = \text{Nota da Prova Parcial}$$

$$\text{NS} = \text{Nota dos Seminários}$$

$$\text{NP} = \text{Nota dos Projetos}$$

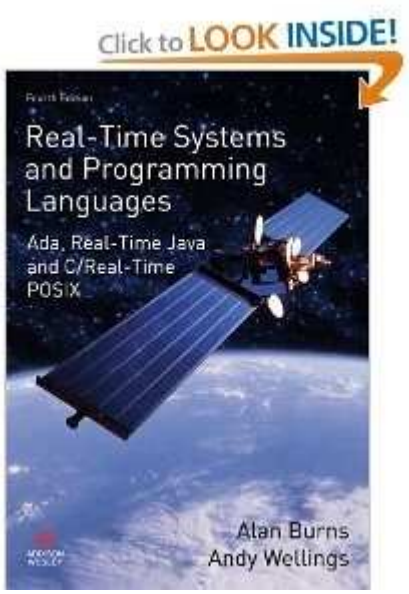
Referências Bibliográficas (1)

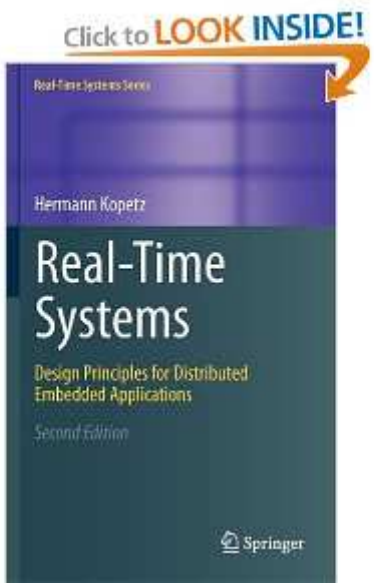
- Burns, Alan e Wellings, Andrew J., *Real-Time Systems And Programming Languages*, Addison Wesley, 2009
- Shaw, Alan C., *Sistemas e Software De Tempo Real*, Bookman Companhia Ed, 2003
- Kopetz, Hermann, *Real-Time Systems : Design Principles for Distributed Embedded Applications*, Kluwer Academic, 1997
- Cooling, J.E., *Software Engineering For Real-Time Systems*, Addison Wesley, 2002
- Jean J. Labrosse, *MicroC/OS II: The Real Time Kernel*, CMP Books, 2002
- John Barnes, *Programming in Ada95*, Second Edition, Addison Wesley, 1998

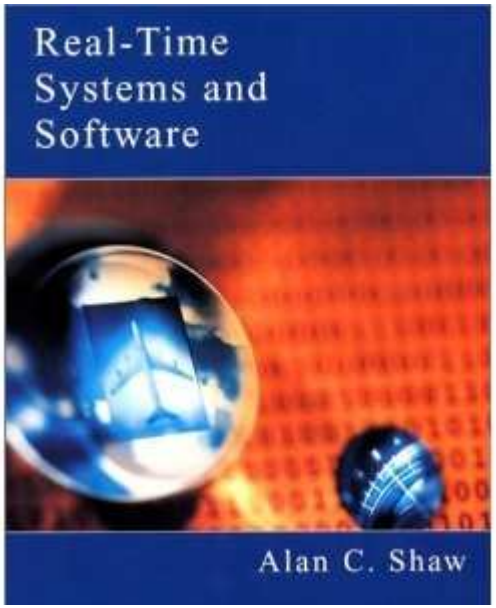
Referências Bibliográficas (2)

- Baier, C. and Katoen, J.-P. *Principles of Model Checking*. The MIT Press, 2008
- Berard, B.; Bidoit, M.; Finkel, A. and F. *Laroussinie Systems and Software Verification: Model-Checking Techniques and Tools*. Springer, 2001
- Clarke. E.; Grumberg, O. and Peled, A. *Model Checking*. The MIT Press, 2000

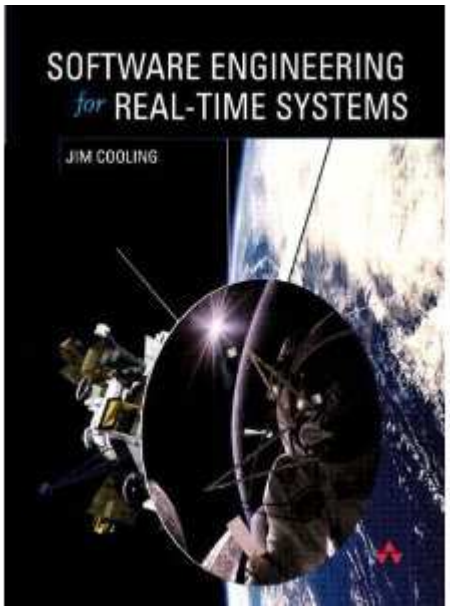
Referências Bibliográficas (3)

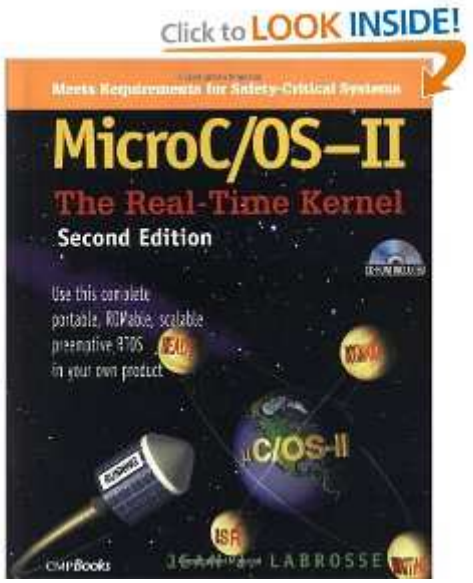


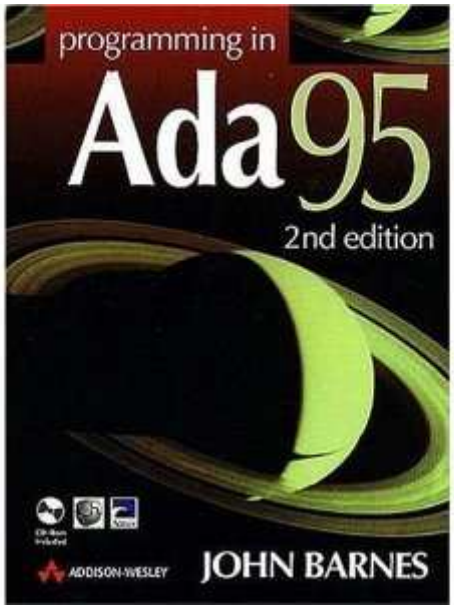




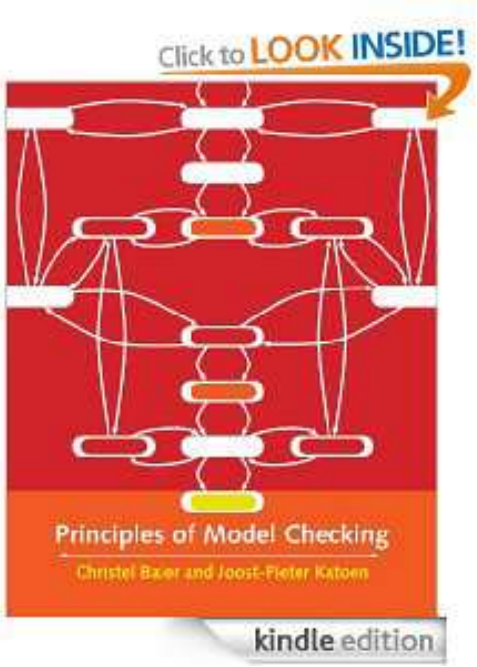
Referências Bibliográficas (4)

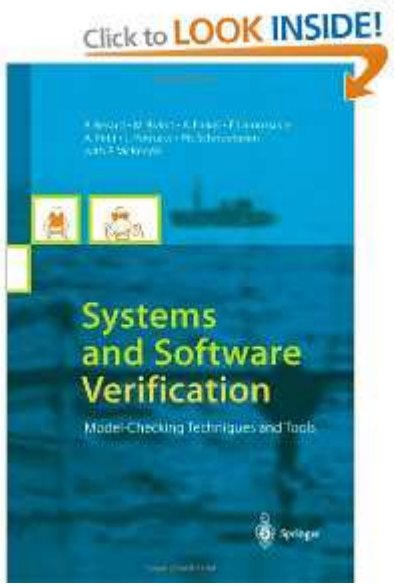


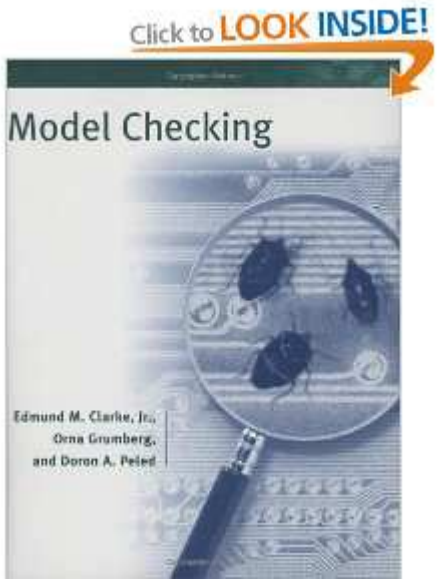




Referências Bibliográficas (5)







O que é um Sistema de Tempo Real?

- Um sistema de tempo real é qualquer sistema de processamento de informação que deve:
  - responder a um ***estímulo*** de entrada ***gerado externamente*** dentro de um período de **tempo finito** e **específico**
    - A corretude depende não somente dos **resultados lógicos**, mas como também do **tempo** que o resultado foi entregue
    - **Falha para responder** é tão ruim quanto uma resposta errada!
- O computador é um componente dentro um sistema maior de engenharia => EMBEDDED COMPUTER SYSTEM
- 99% de todos os processadores são destinados para o mercado de sistemas embarcados

Definição (1)

- Young (1982) define um sistema de tempo real como:

  *"any **information processing** activity or system which has to respond to **externally input stimuli** within a **finite** and **specified period**"*

- Uma outra definição é (Randell et al., 1995):

  *"A real-time system is a system that is required to **react to stimuli from the environment** (including the passage of physical time) **within time intervals** dictated by the environment"*

Definição (2)

by Alan Shaw

- São sistemas que **monitoram**, **respondem** ou **controlam** um **ambiente externo**
- Ambiente conectado ao sistema de computação (SC) através de **sensores**, **atuadores** e outras **interfaces de E/S**.
- O SC deve satisfazer a várias **restrições**, principalmente as impostas a ele pelo comportamento de tempo-real do mundo externo
- Pode ser chamado de sistema **reativo** (se reagir a eventos externos) ou sistema **embarcado** (se estiver dentro de um sistema maior)

Terminologia

- Hard real-time — sistemas onde é absolutamente imperativo que as respostas ocorram dentro de prazo de entrega solicitado (sistema de controle de vôo)

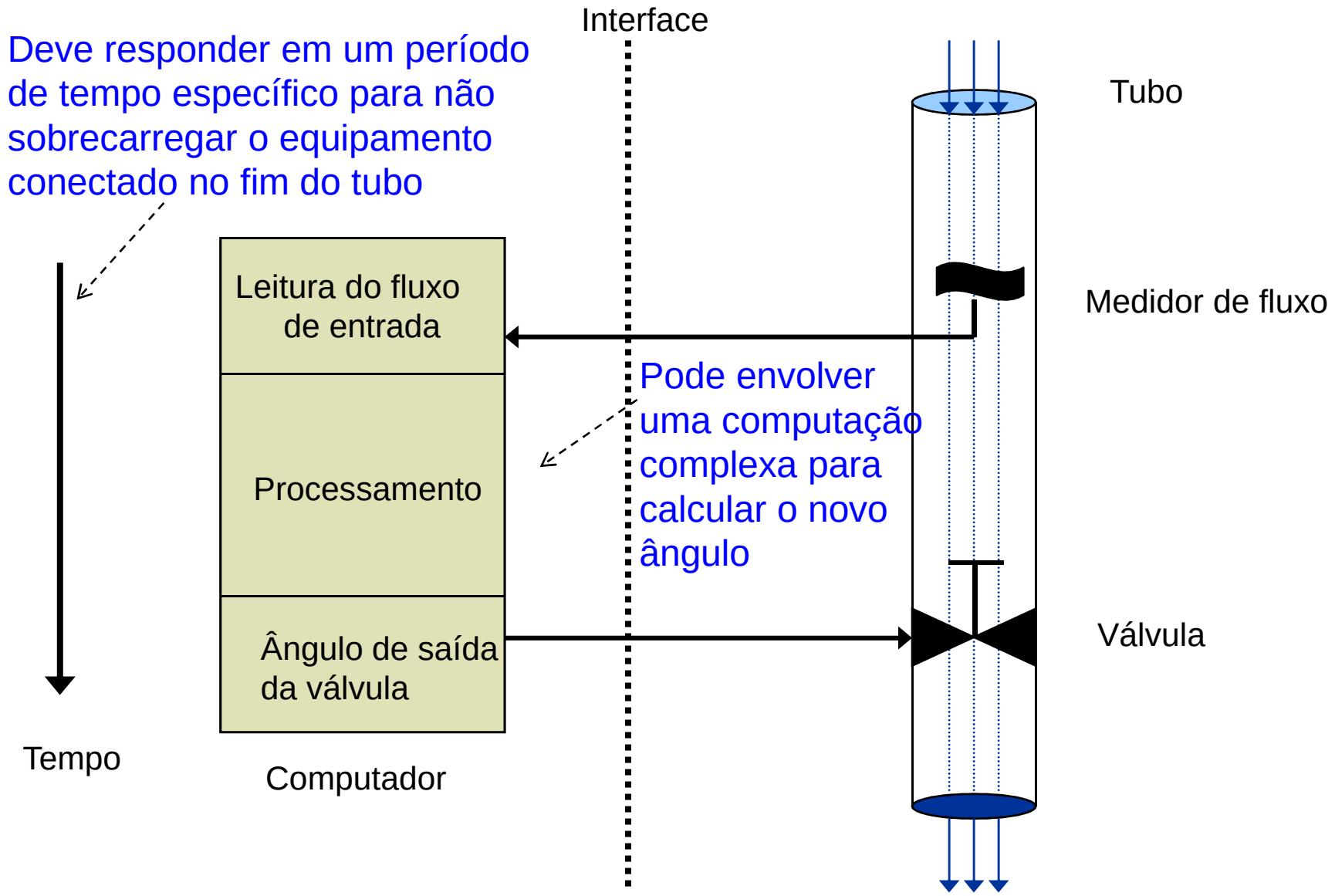
- Soft real-time — sistemas onde os prazos de entrega são importantes, mas continuarão funcionando “corretamente” se os prazos não forem atendidos ocasionalmente (sistema de aquisição de dados)

- Real real-time — sistemas que são *hard real-time* e que os tempos de respostas são curtos (sistema de guia de míssil)

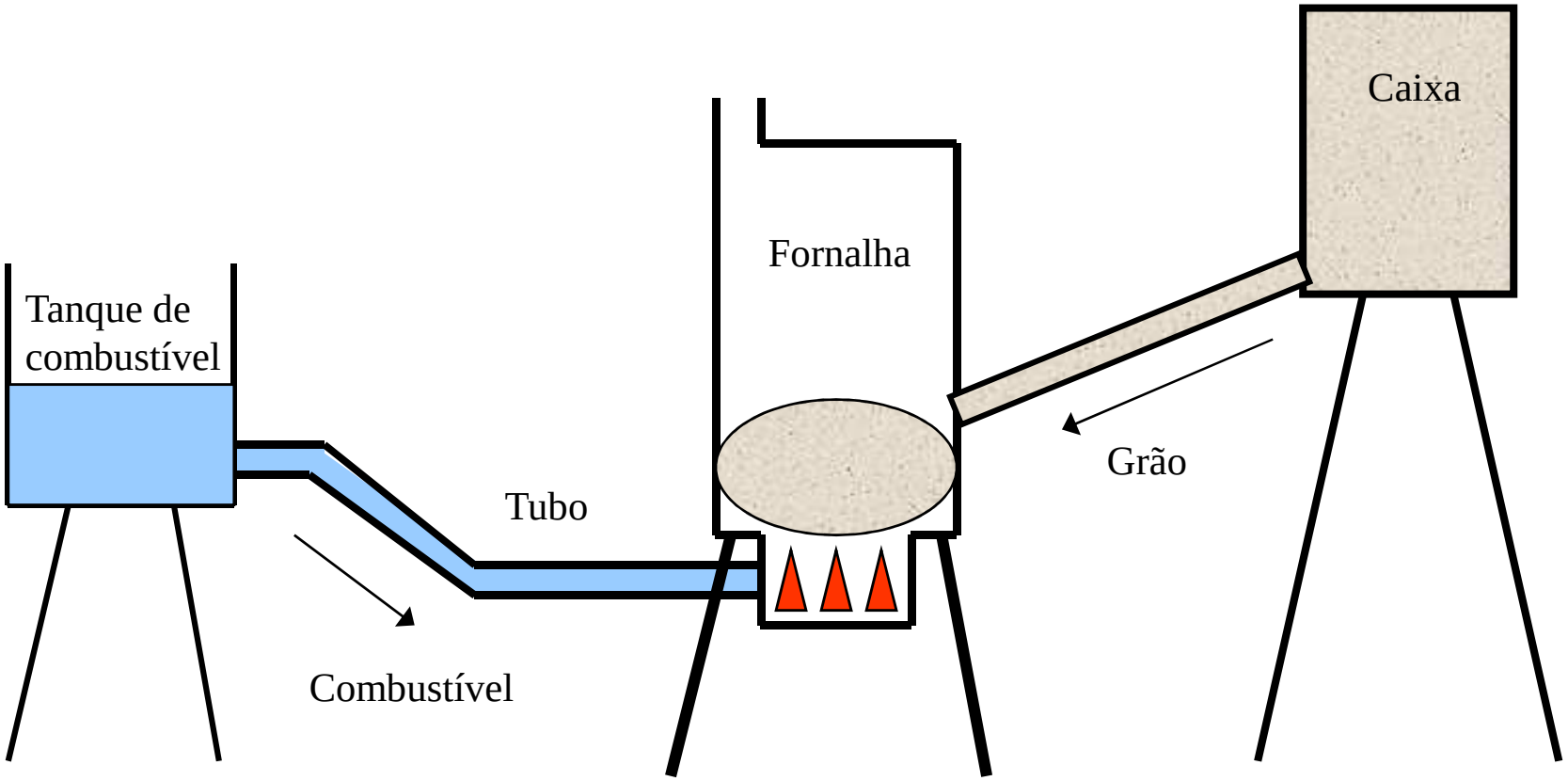
- Firm real-time — sistemas que são *soft real-time* mas que não existe benefício de entregas de serviço com atraso.

Um único sistema pode ter sub-sistemas *hard*, *soft* e *real real-time* (função custo associada com cada prazo de entrega)

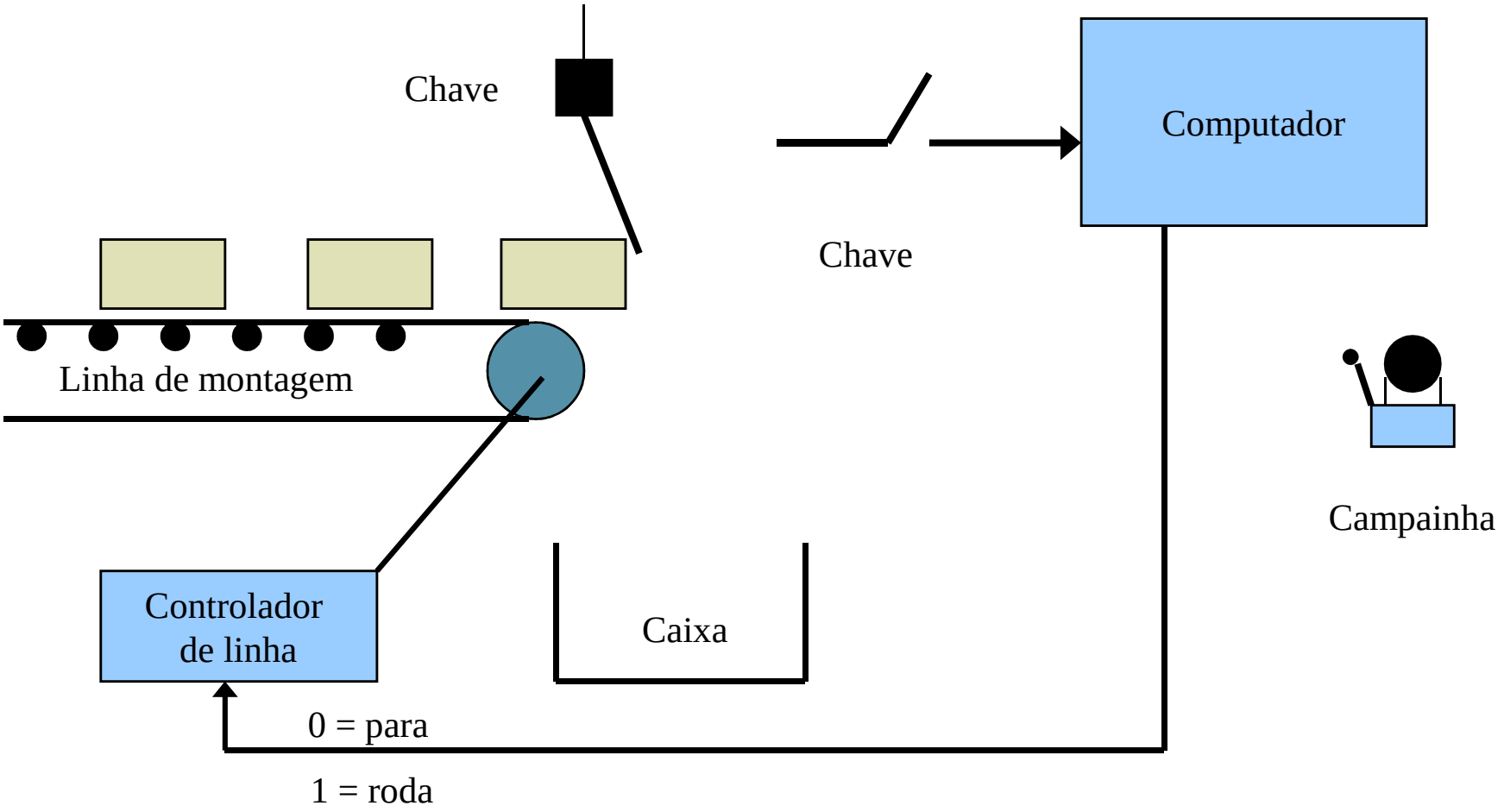
Sistema de Controle de Fluído



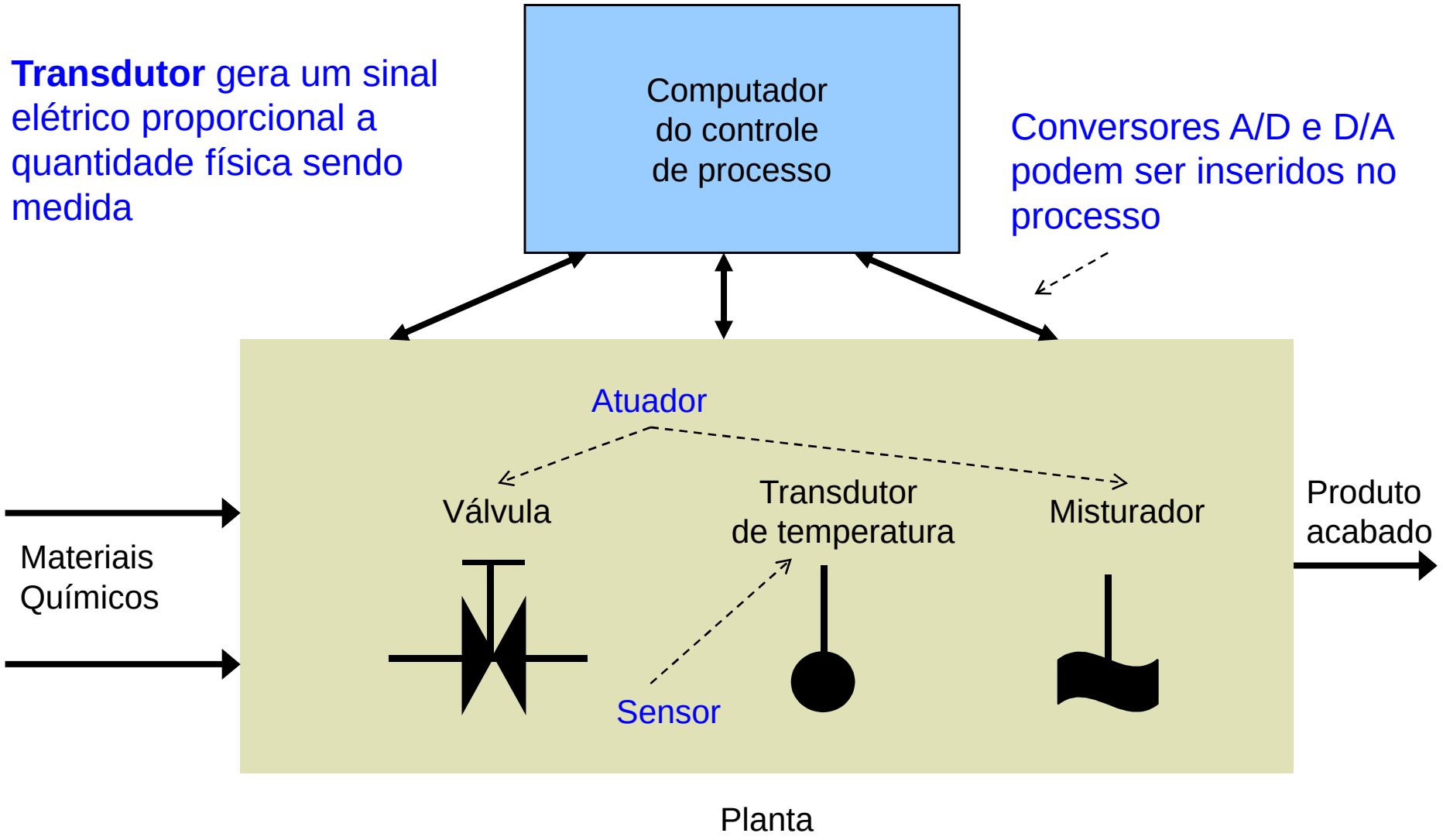
Planta de Torrefação de Grão



Estação de Empacotamento

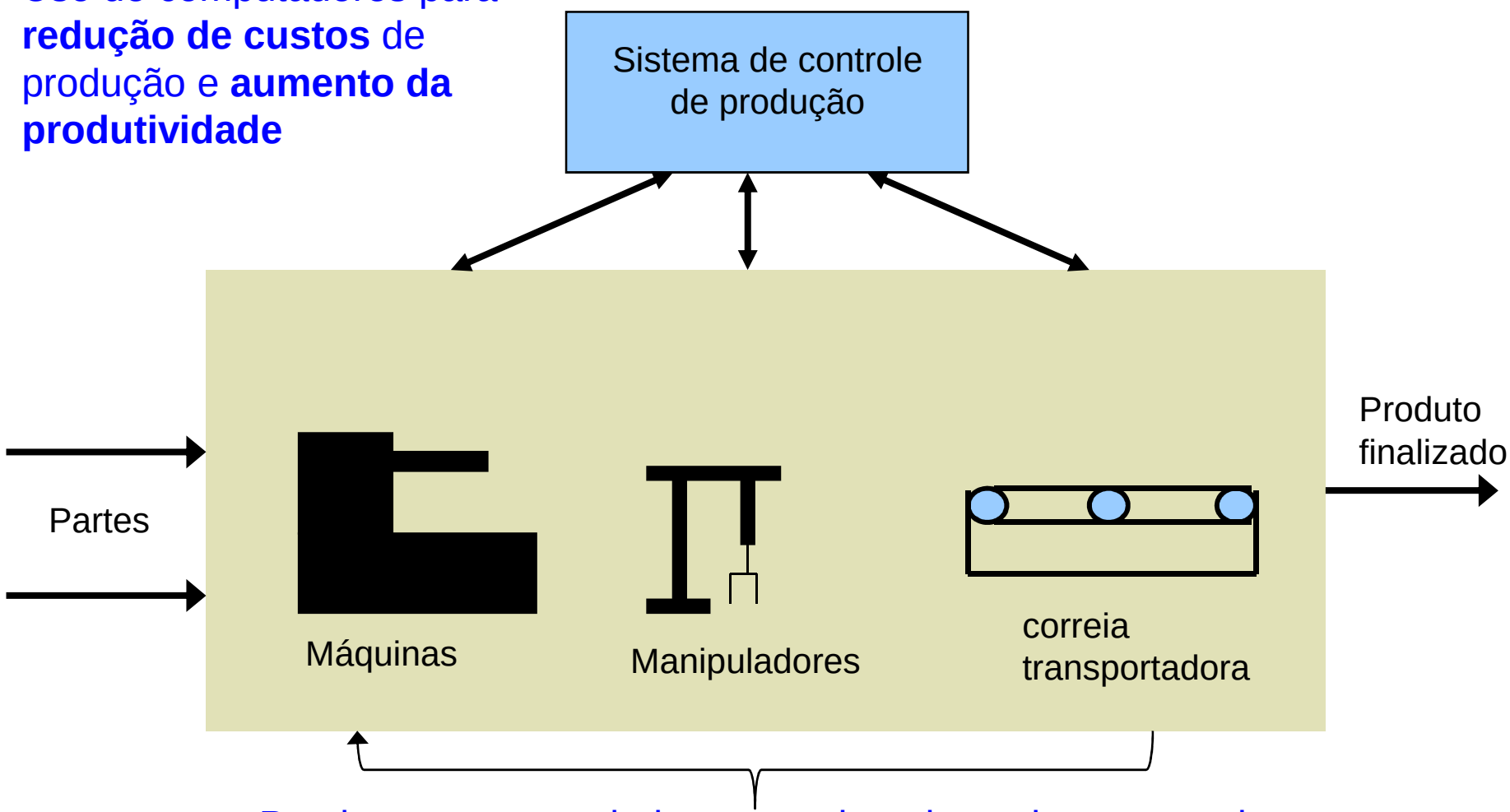


Sistema de Controle de Processo



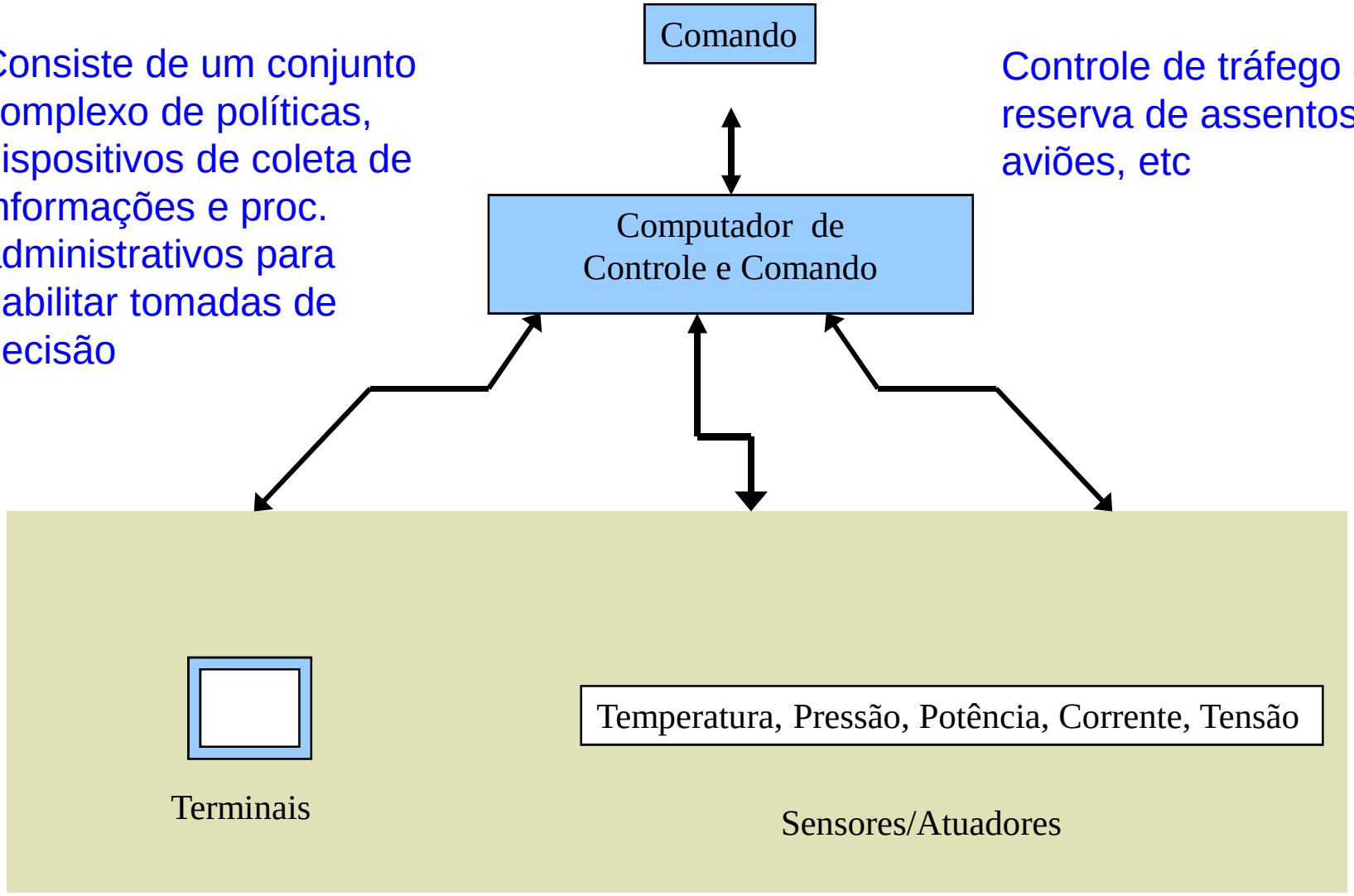
Sistema de Controle de Produção

Uso de computadores para **redução de custos** de produção e **aumento da produtividade**



Sistema de Controle e Comando

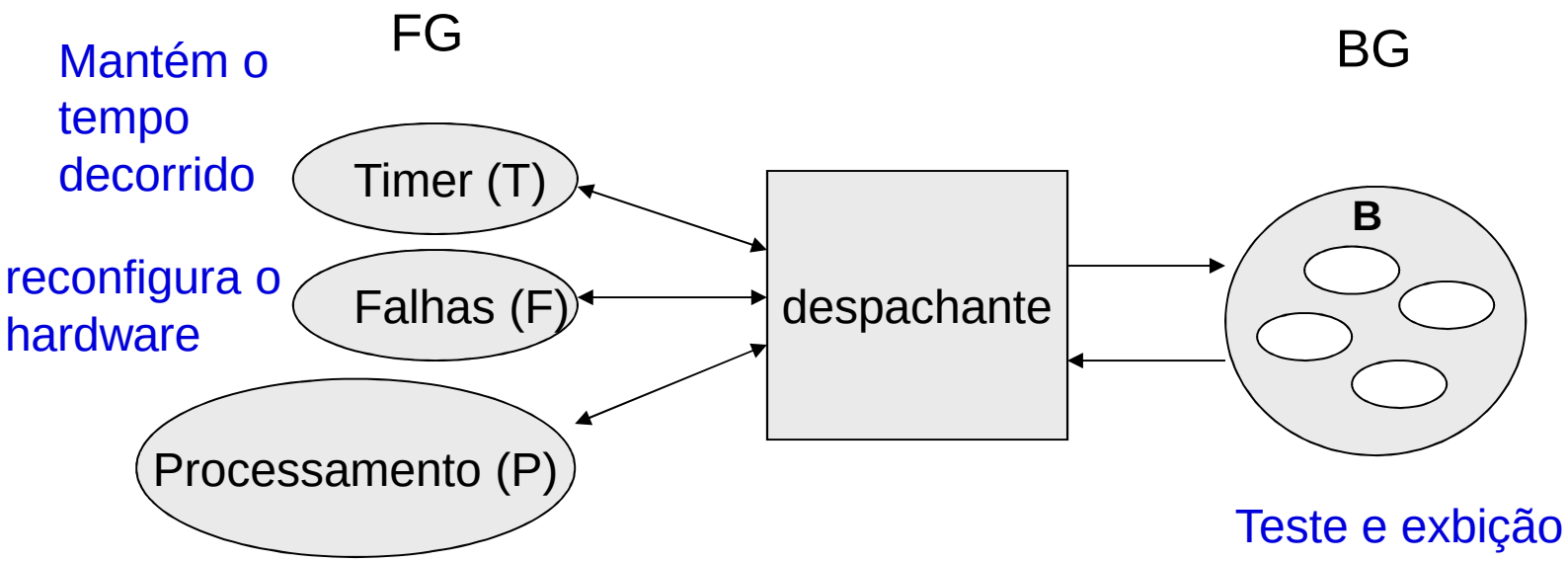
Consiste de um conjunto complexo de políticas, dispositivos de coleta de informações e proc. administrativos para habilitar tomadas de decisão



Controle de tráfego aéreo, reserva de assentos de aviões, etc

Monitor de Usina Nuclear

Sistema opera com redundância e eleição sobre os resultados



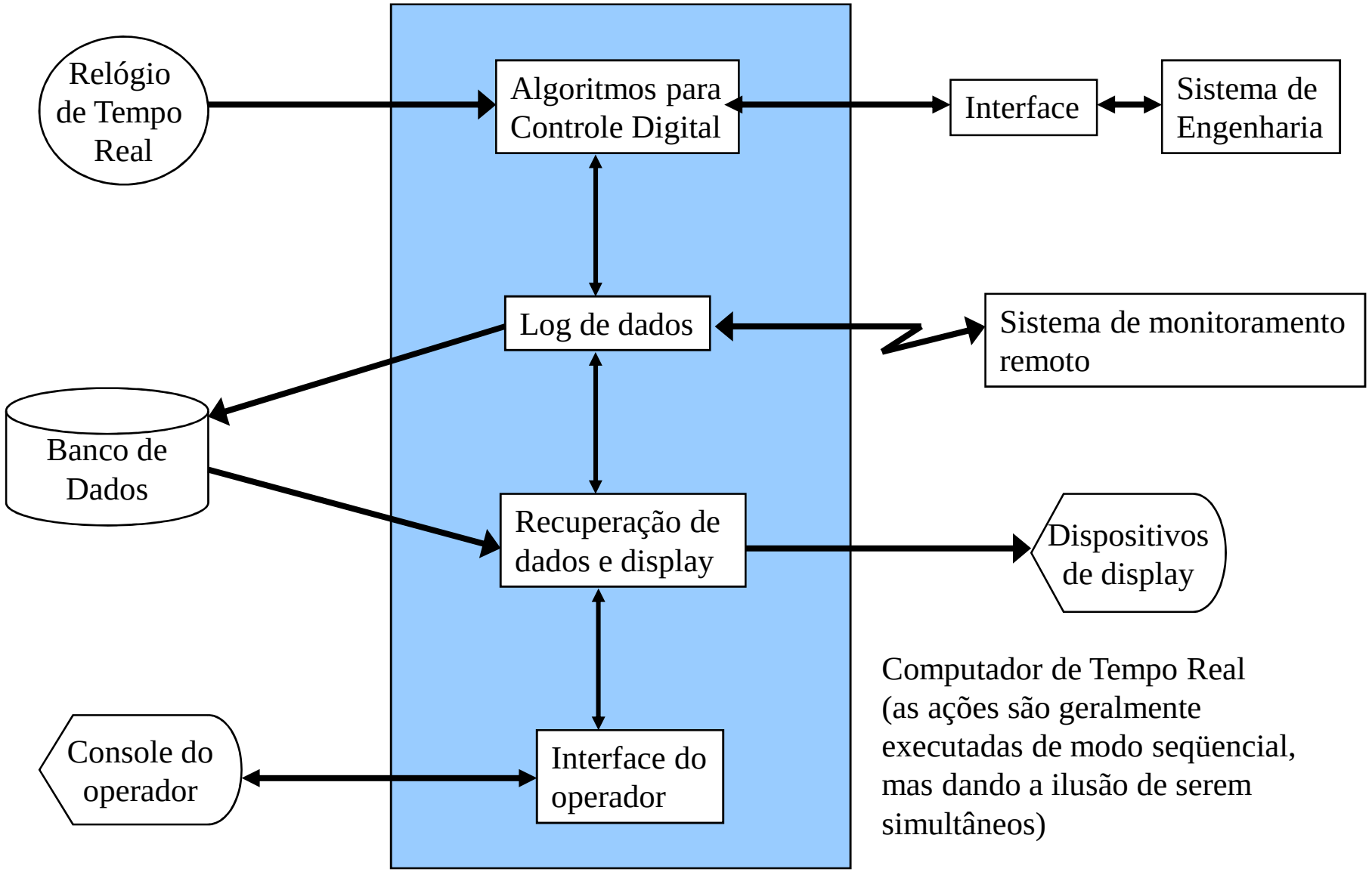
Outros Exemplos...

- **Sistemas de controle** de veículos para automóveis, metrôs, aeronaves, ferrovias e navios
- **Controle de tráfego** para auto-estradas, espaço aéreo, trilhos de ferrovias e corredores de navegação marítima
- **Controle de processo** para usinas de energia, indústrias químicas e para produtos de consumo, como refrigerantes e cerveja
- **Sistemas médicos** para radioterapia, monitoramento de pacientes e desfibrilamento
- **Uso militares** como controle de tiro, rastreamento e sistemas de comando e controle
- Sistema de **manufatura** com robôs

Outros Exemplos...

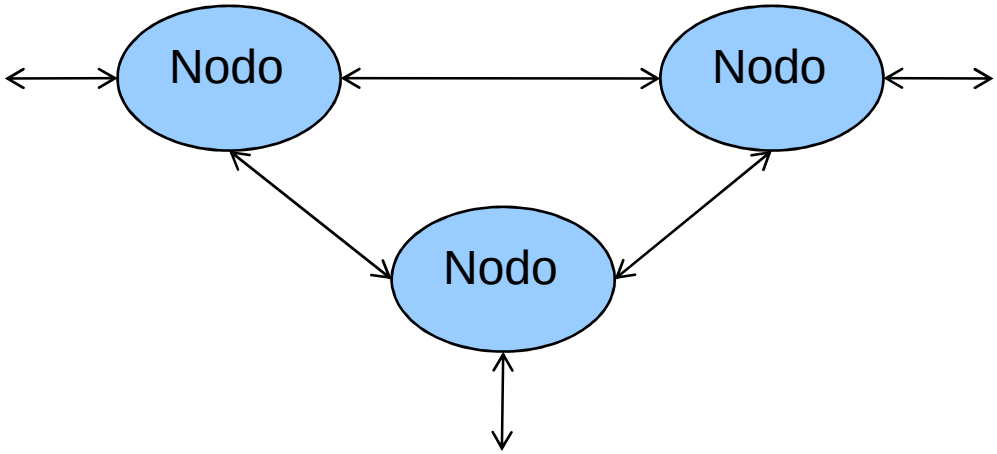
- Telefone, rádio e comunicação por satélite
- Jogos por computador
- **Sistemas de multimídia** que provêm interfaces textuais, gráficas, de áudio e de vídeo
- **Sistemas domésticos** para monitoramento e controle de eletrodomésticos
- **Sistemas de automação predial** que controlam temperatura ambiental, iluminação, portas e elevadores

Um Sistema Embarcado Típico

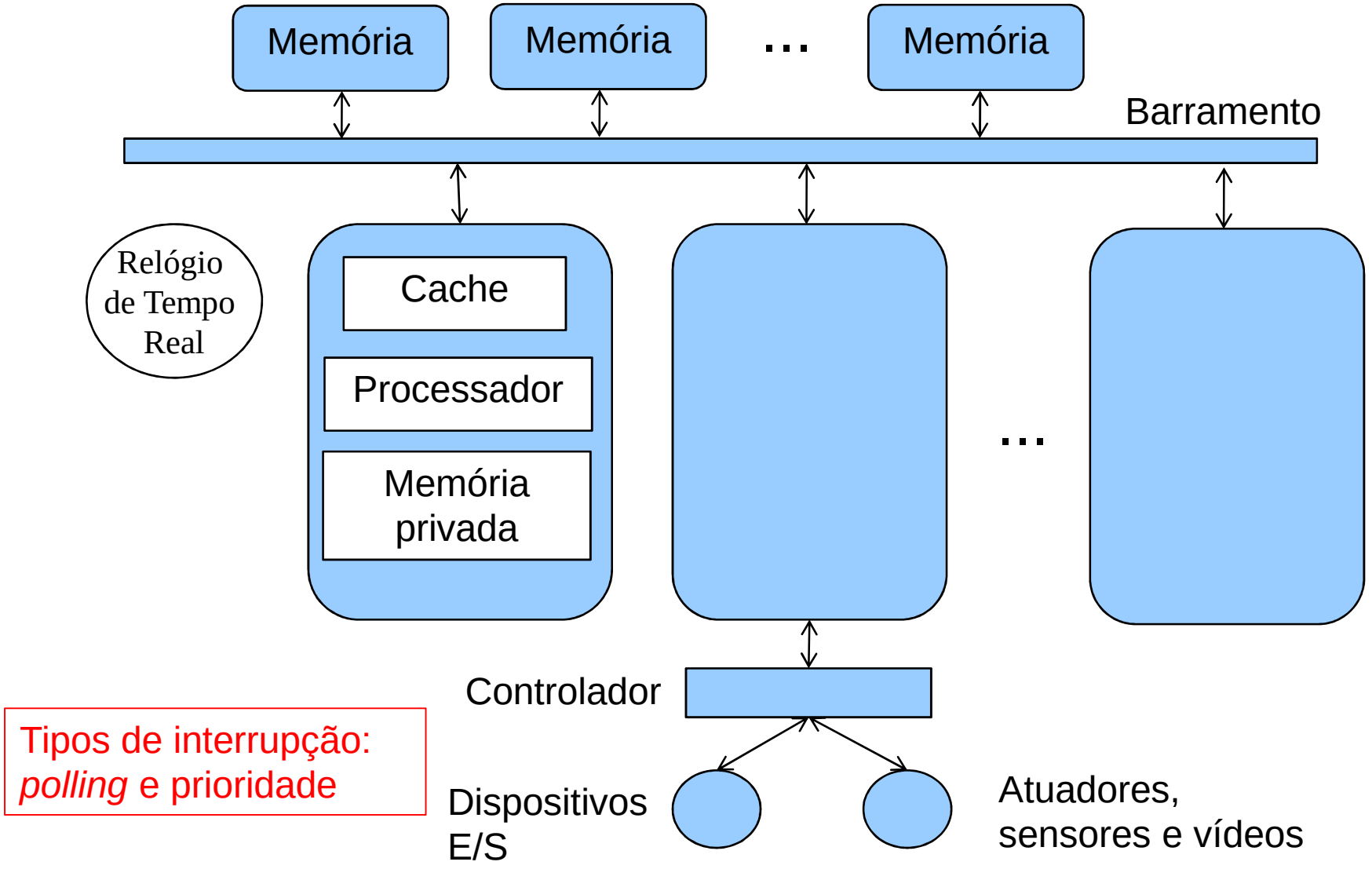


Computador de Tempo Real
(as ações são geralmente executadas de modo seqüencial, mas dando a ilusão de serem simultâneos)

Um Sistema Embarcado Distribuído (1)



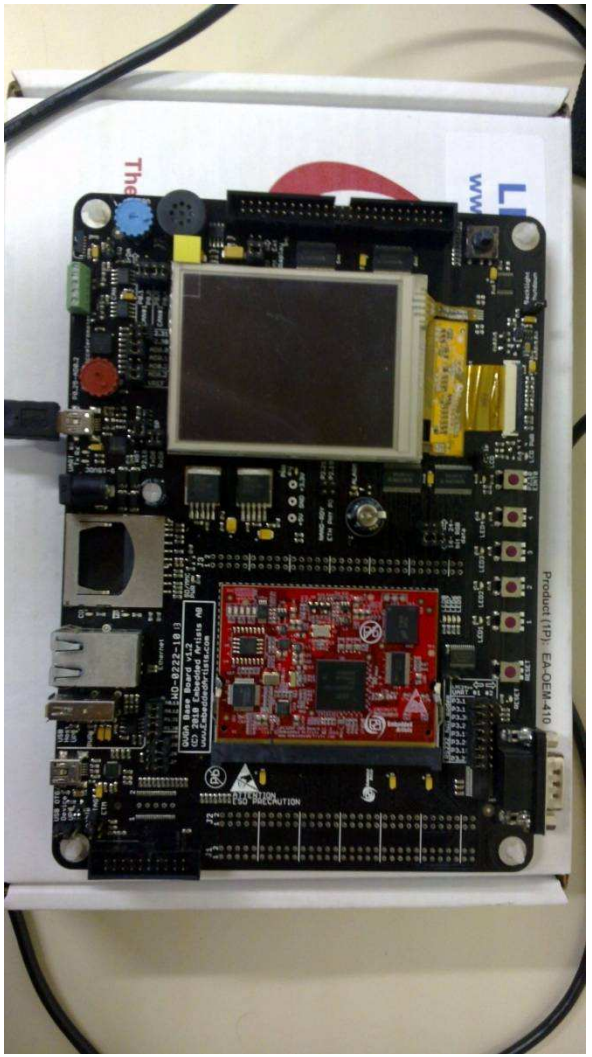
Um Sistema Embarcado Distribuído (2)



Tipos de interrupção: *polling* e prioridade

LPC3250

- **Principais características:**
  - ARM9 208MHz
  - 64MB DRAM
  - 64MB SRAM
  - Ethernet 10/100
  - USB OTG
  - LCD
  - touchscreen 3.2"



Características de um STR (1)

- **Grandes e complexos** — variam de algumas centenas de linhas em assembly ou C para milhões de linhas de ADA estimado para o *Space Station Freedom*
  - *Tamanho do sistema está relacionado a variedade (número de instruções, esforço de desenvolvimento, responder a eventos externos)*
- **Controle concorrente dos componentes do sistema** — dispositivos que operam em paralelo no mundo real (melhor modelar este paralelismo através de entidades concorrente no programa)
- **Facilidade de interagir com o hardware de propósito especial** — precisa ser capaz de programar os dispositivos em uma maneira abstrata e confiável

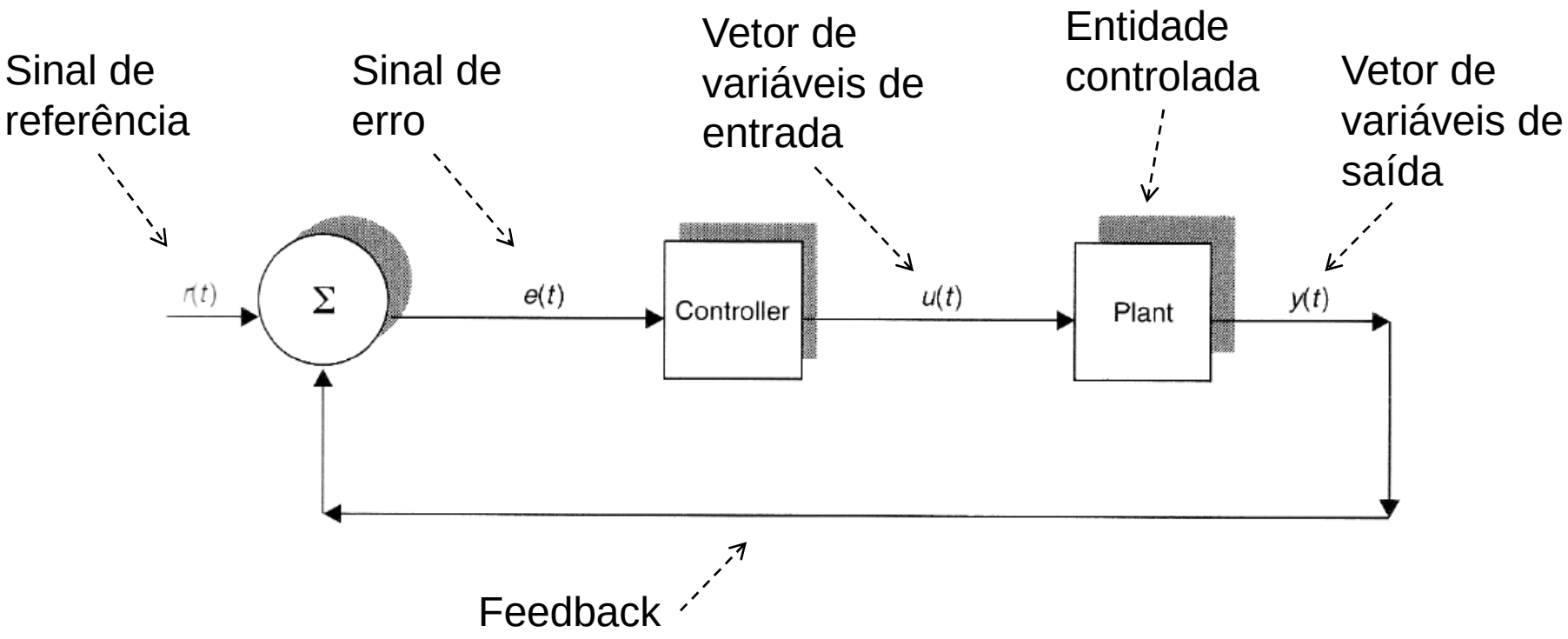
Características de um STR (2)

- **Extrema confiabilidade e segurança** — sistemas embarcados tipicamente controlam o ambiente no qual eles operam; falha para controlar pode resultar em perda de vidas, danos ao meio ambiente e perda econômica
- **Garantia nos tempos de resposta** — nós precisamos ser capazes de prever com confidência o tempo de resposta no pior caso para os sistemas; *eficiência* é importante mais *previsão* é essencial

Nem todo STR exibe todas estas características, porém as linguagens e SOs usados para desenvolver STR devem fornecer facilidades que suportem estas características

Manipulação de Números Reais (1)

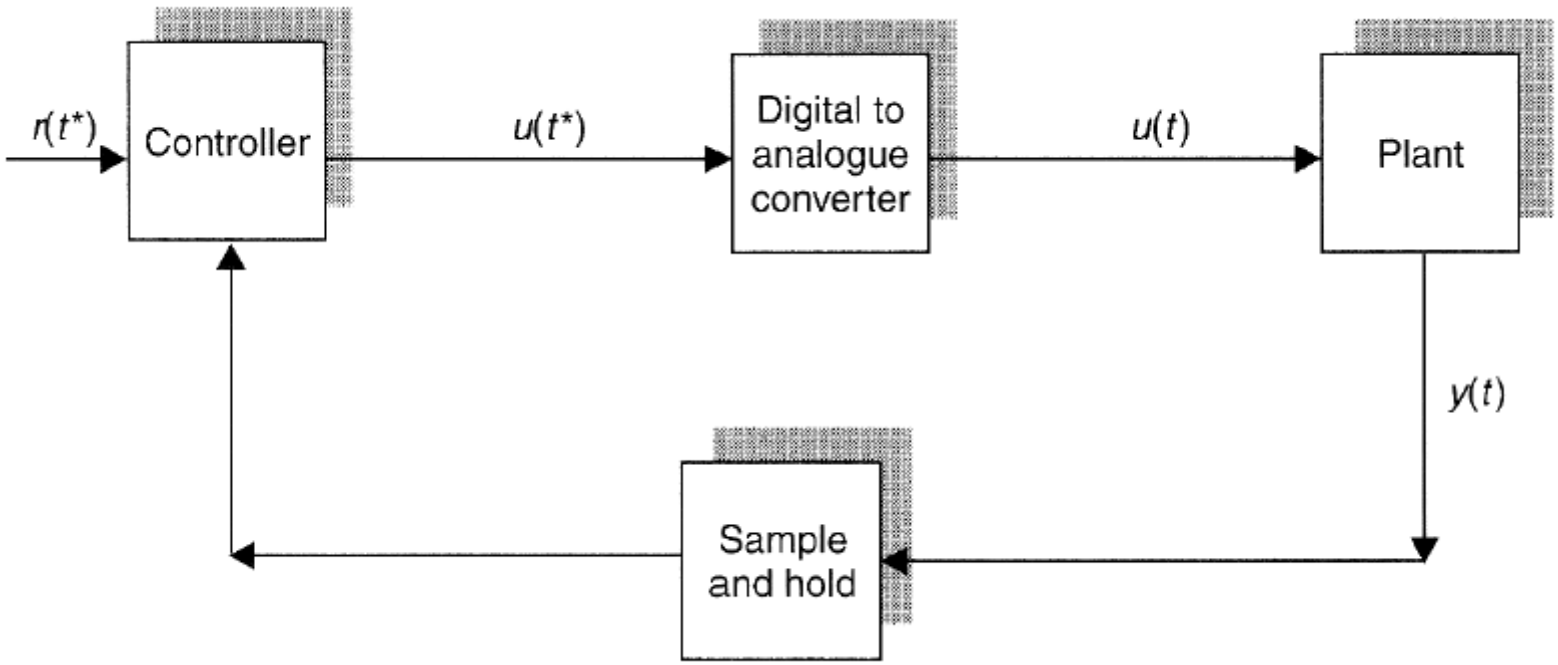
Um Simples Controlador Analógico



É necessário ter um modelo matemático da planta usando equações diferenciais de primeira ordem (teoria de controle)

Manipulação de Números Reais (2)

Um Simples Controlador Computadorizado



Dentro do computador, as equações diferencias podem ser resolvidas através de técnicas numéricas

São Sistemas de Tempo Real?

- Um sistema de folha de pagamento que produz contra-cheques de empregados a cada duas semanas?
- Um sistema de cadastro esportivo que registra, mantém e exibe resultados durante eventos esportivos, tais como jogos de beisebol ou provas de atletismo?
- Um controlador de cancela em uma interseção de trilhos de ferrovia com uma rodovia, o qual controla a abertura e o fechamento do acesso ao cruzamento, para assegurar que a rodovia fique bloqueada sempre que um trem esteja na área de interseção?
- Um sistema de registro médico que mantém os históricos médicos de pacientes numa clínica?

Por quê Garantir Confiabilidade do Código? (1)

- Funcionalidade demanada aumentou de forma significativa
  - teste e revisão em pares
- Processadores *multi-core* com memória compartilhada escalável

```
void *threadA(void *arg) {
  lock(&mutex);
  x++;
  if (x == 1) lock(&lock);
  unlock(&mutex);
  lock(&mutex);
  x--;
  if (x == 0) unlock(&lock);
  unlock(&mutex);
}
```

(CS1)

(CS3)

Deadlock

```
void *threadB(void *arg) {
  lock(&mutex);
  y++;
  if (y == 1) lock(&lock);
  unlock(&mutex);
  lock(&mutex);
  y--;
  if (y == 0) unlock(&lock);
  unlock(&mutex);
}
```

(CS2)

Por quê Garantir Confiabilidade do Código? (2)

Buffer Circular usando FIFO:

```
static char buffer[BUFFER_MAX];
void initLog(int max) {
  buffer_size = max;
  first = next = 0;
}

int removeLogElem(void) {
  first++;
  return buffer[first-1];
}

void insertLogElem(int b) {
  if (next < buffer_size) {
    buffer[next] = b;
    next = (next+1)%buffer_size;
  }
}
```

Caso de Teste:
Checar se as mensagens são adicionadas e removidas do buffer circular

```
static void testCircularBuffer(void) {
  int senData[] = {1, -128, 98, 88, 59,
                   1, -128, 90, 0, -37};
  int i;
  initLog(5);
  for(i=0; i<10; i++)
    insertLogElem(senData[i]);
  for(i=5; i<10; i++)
    ASSERT_EQUAL_INT(senData[i],
                     removeLogElem());
}
```

Por quê Garantir Confiabilidade do Código? (3)

Buffer Circular usando FIFO:

```
static char buffer[BUFFER_MAX];
void initLog(int max) {
  buffer_size = max;
  first = next = 0;
}

int removeLogElem(void) {
  first++;
  return buffer[first-1];
}

void insertLogElem(int b) {
  if (next < buffer_size) {
    buffer[next] = b;
    next = (next+1)%buffer_size;
  }
}
```

Mas: implementação é falha!

O array buffer é do tipo char e tamanho BUFFER_MAX

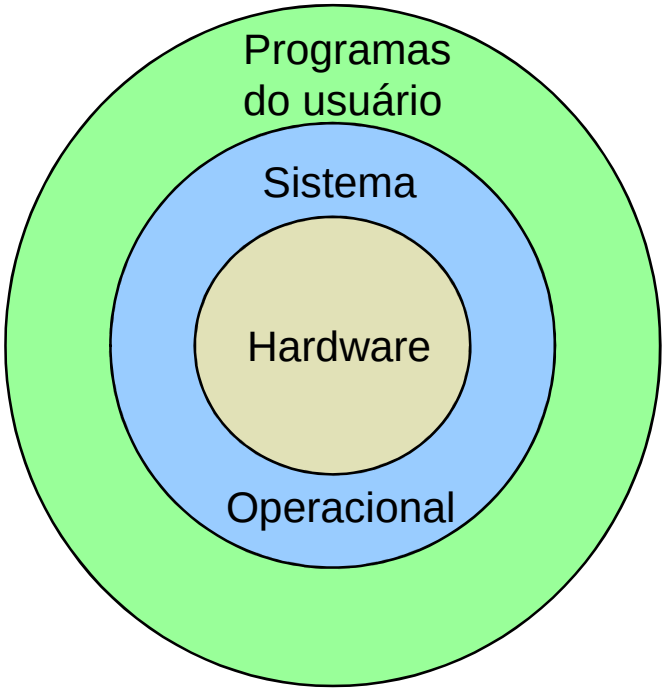
Incrementa *first* sem checar o limite do array: *buffer overflow*

Atribui uma variável inteira a um char: *typecast overflow*

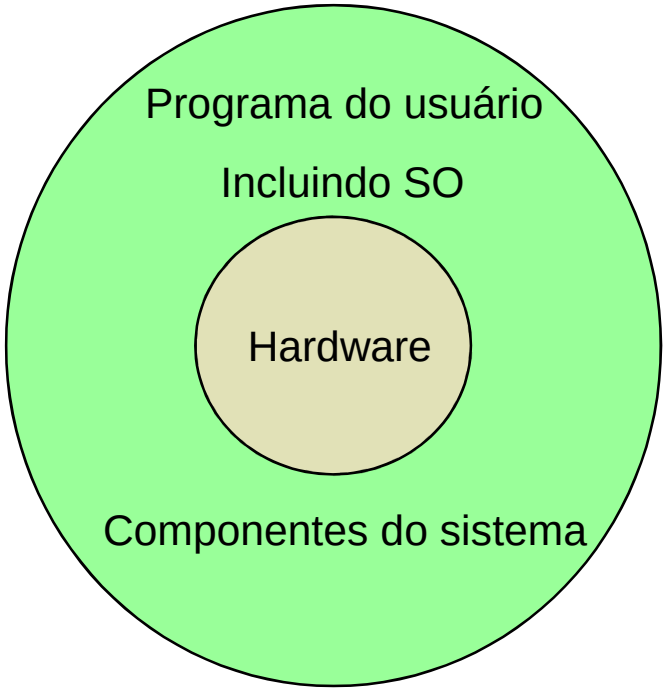
Linguagens de Programação de Tempo Real

- Linguagem Assembly
- Linguagens de sistemas sequênciais — p.e. RTL/2, Coral 66, Jovial, C
  - Normalmente exigem suporte de SO
- Linguagens concorrentes de alto nível. Ação da crise do software. p.e. Ada, Chill, Modula-2, Mesa, Java.
  - Não exigem suporte do SO!
- Nós consideraremos:
  - Java/Real-Time Java
  - C e Real-Time POSIX
  - Ada 95

Linguagens de Tempo Real e SO



Configuração de um SO Típico



Configuração Típica Embarcada

Resumo

- Duas principais classes podem ser identificadas:
  - sistemas de tempo real crítico
  - sistemas de tempo real brando
- As características básicas de um sistema embarcado ou de tempo real são:
  - tamanho e complexidade
  - manipulação de números reais
  - extrema confiabilidade e segurança
  - controle concorrente de componentes isolados do sistema
  - controle de tempo real
  - interação com as interfaces do hardware
  - implementação eficiente