# Sistemas Embarcados - Sistemas de Tempo Real

Instituto Federal de Pernambuco
Coordenação de Informática/IFPE

Anderson L. S. Moreira
anderson.moreira@recife.ifpe.edu.br

creative commons
COMMONS DEED

**Attribution – ShareAlike 3.0**
**Você é livre para**

Copiar, distribuir, mostrar, e adaptar o trabalho
Para fazer trabalhos derivados
Para fazer uso comercial do trabalho

**Seguindo certas condições**

**Atribuição. Você deve dar os devidos créditos ao autor original.**
**Compartilhar. Se você altera, transformar ou construir em cima deste trabalho, você deverá distribuir o trabalho resultante somente sobre uma licença idêntica a está.**

Para qualquer reuso ou distribuição, você deve deixar claro aos outros os termos de licença deste trabalho.
Qualquer destas condições podem ser modificadas se você tiver permissão do autor original.

**Se uso e outros direitos não são afetados pelas regras acima.**
License text: http://creativecommons.org/licenses/by-sa/3.0/legalcode

# Agenda

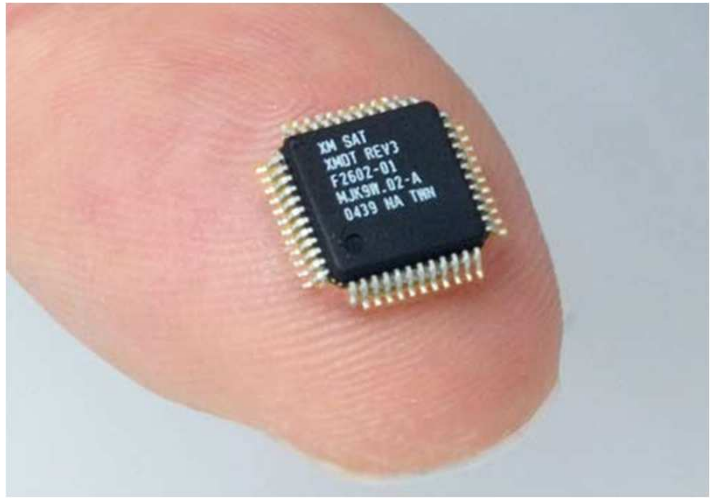
- Introdução
- Sistemas
- Sistemas de Tempo Real
- Eventos, determinismo e utilização de CPU
- Questões de projeto em STR
- Equívocos Comuns
- Histórico

- Sistemas de Tempo Real (STR)
  - Presente em elementos de diferentes domínios
    - Militar, espacial, doméstico, bancário, etc.

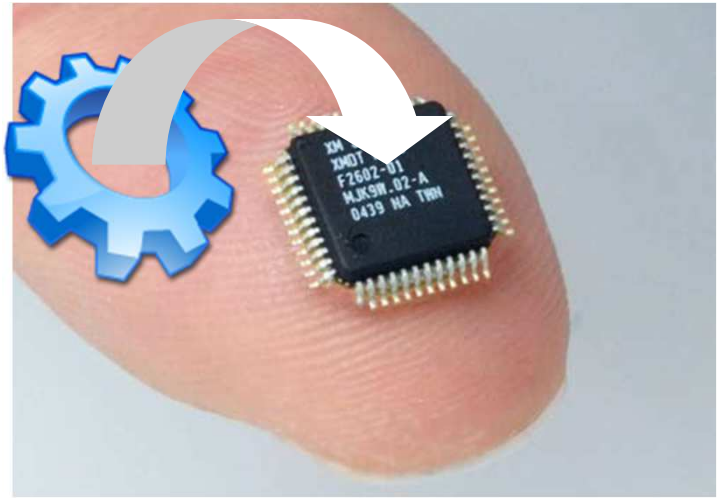
"Um **sistema embarcado** é similar a qualquer sistema computacional que não seja um *desktop*"

(VAHID-UCR)

- Um **sistema embarcado** pode ser:
  - Hardware



- Um **sistema embarcado** pode ser:
  - Hardware
  - Software + Hardware



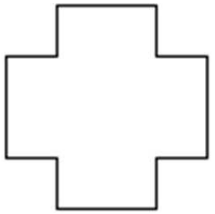
- Algumas situações:
  - Determinar a correta posição de uma aeronave em um dado instante.
  - Reservas de um vôo de Recife para Brasília.
  - Falha de temperatura em uma usina nuclear.
  - Ligar a TV.

- Sistemas Computacionais
  - Propósito Geral
    - Processadores programáveis
  - Propósito Específico (Hardware)
    - Executam um único programa
  - Domínio Específico
    - Otimizado para uma classe de aplicações.
      - Exemplo: DSP (Digital Signal Processors)

Funcionalidade Desejada

Processador de uso geral

Processador de aplicação específica



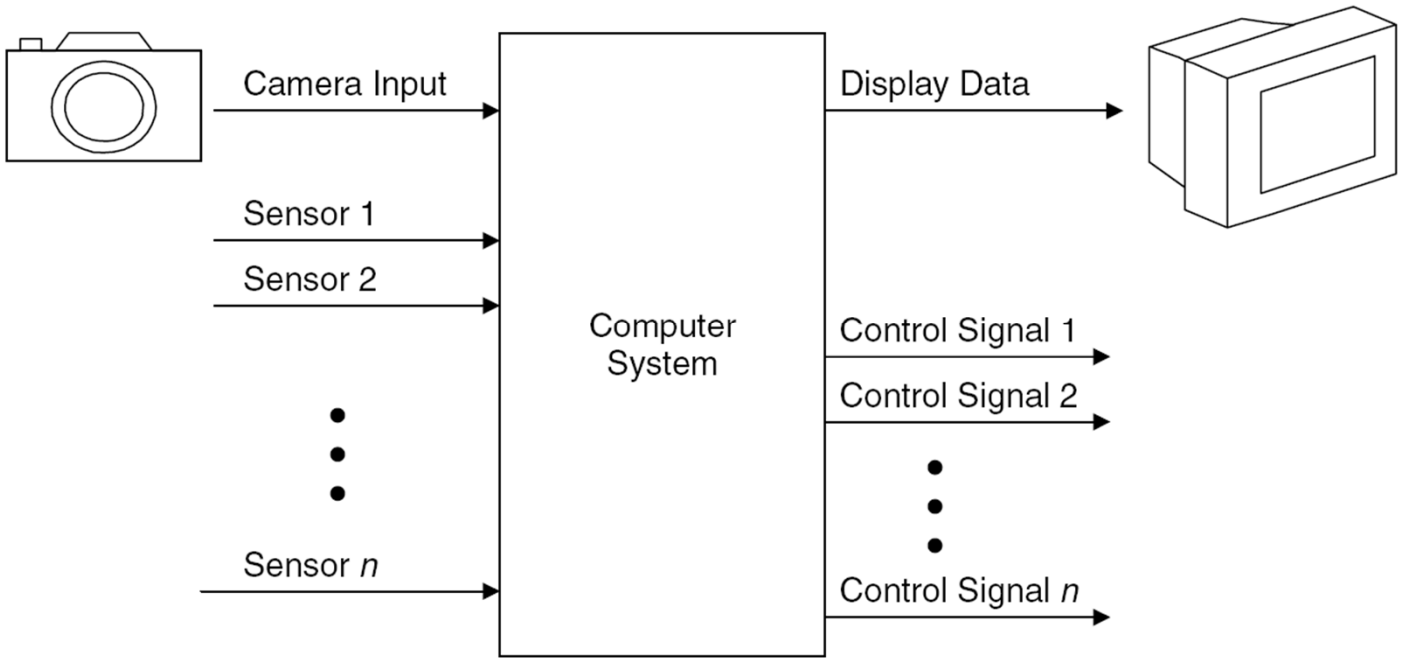
Processador de propósito único

- O *Hardware* de computadores de propósito geral resolve problemas pela execução repetitiva de macroinstruções, conhecidas, coletivamente como software.
- Software
  - Programas de Sistema
    - Softwares que faz interface com o hardware, tais como escalonadores, *drivers* e programas que atuam como ferramenta para o desenvolvimento de aplicações (compiladores, assemblers, *linkers*).

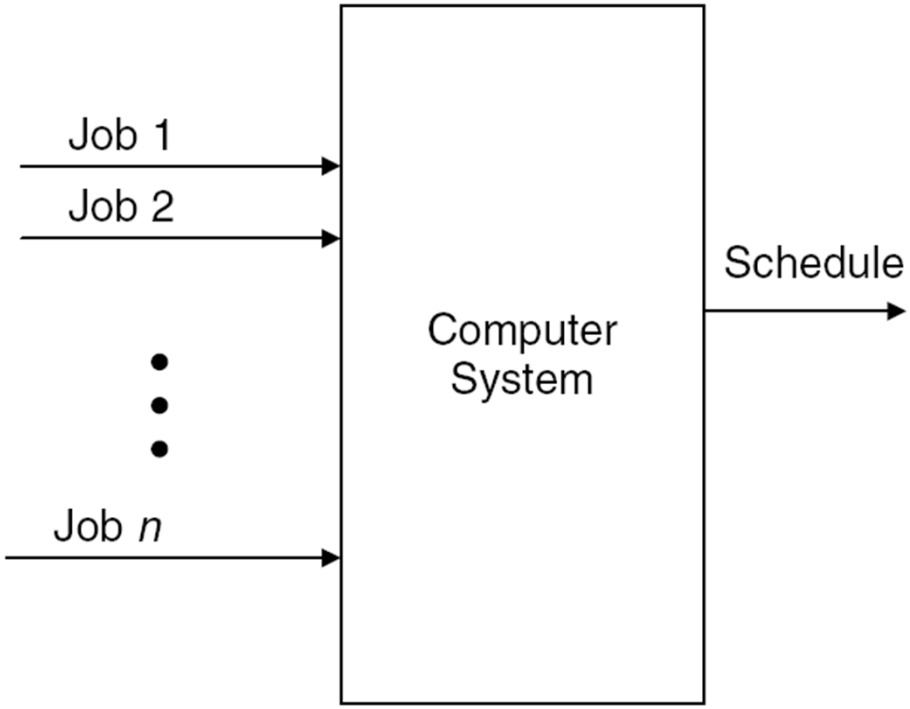
- Sistema Operacional
  - Uma coleção especializada de programas de sistema que gerenciam os recursos físicos do computador.

- Aplicativos
  - Programas escritos para resolver um problema específico
    - Exemplo: folha de pagamento, estoque e navegação.

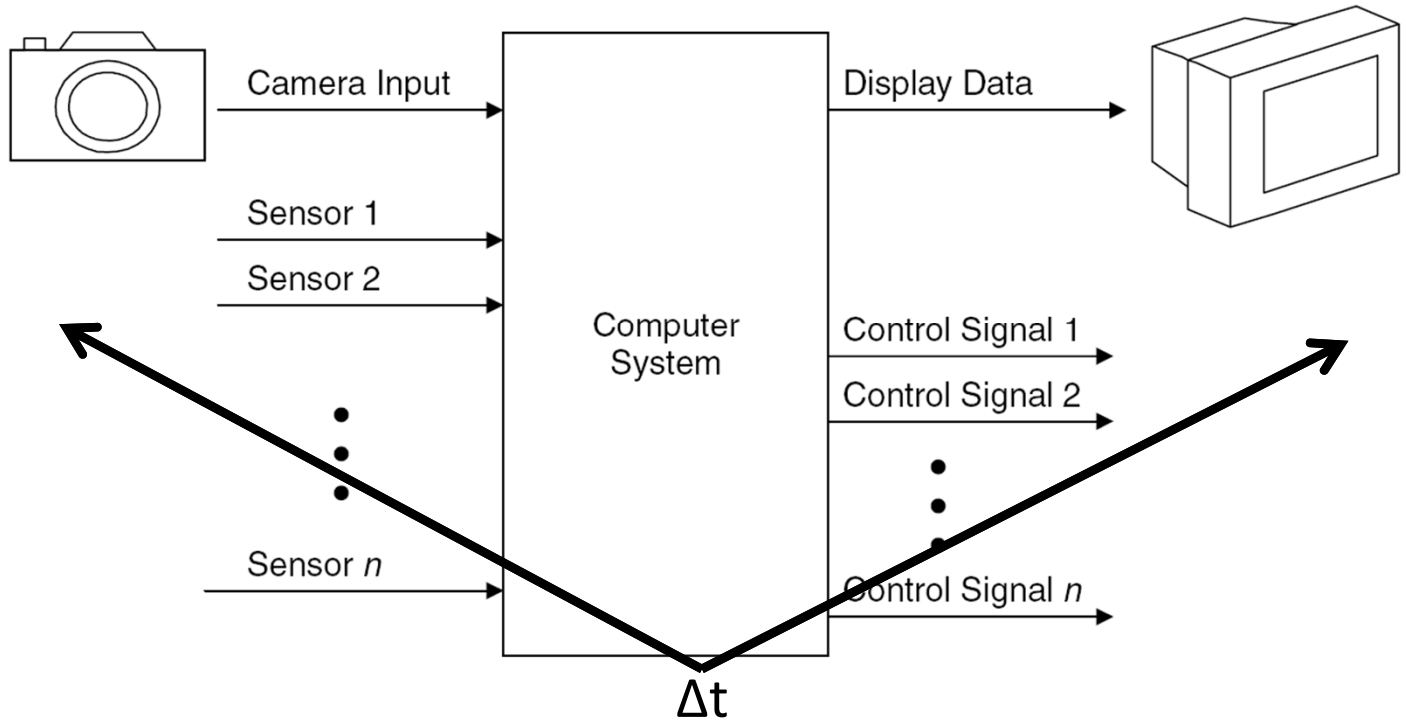
- Sistema
  - Um sistema é um mapeamento de um conjunto de entradas para um conjunto de saídas.



- Jobs (tarefas)
- Schedule (escalonar)

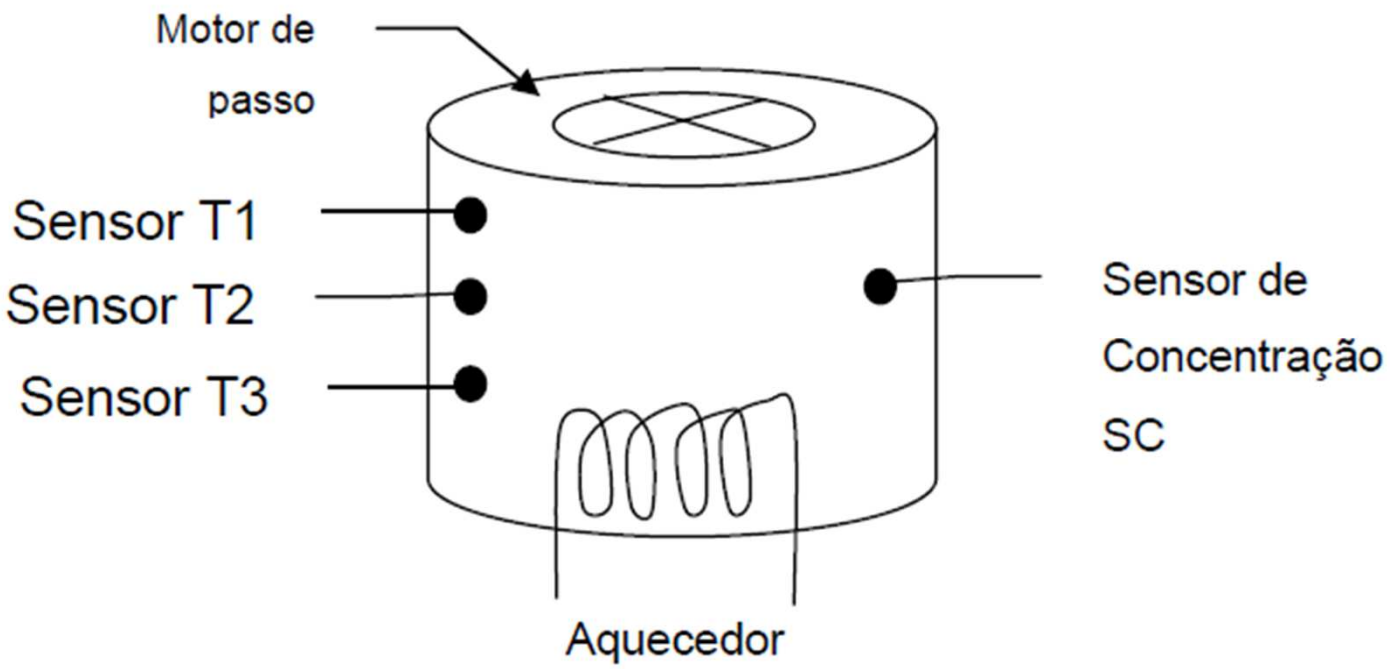


- Tempo de resposta
  - Tempo entre o estímulo e a resposta do sistema computacional.



- Sistemas de tempo real
  - Um sistema que deve satisfazer restrições explícitas de tempo de resposta. O não cumprimento das mesmas envolve severas conseqüências, incluindo falha.
  - Um sistema cujo comportamento correto é baseado tanto no fornecimento de resultados corretos, bem como no tempo que estes resultados são fornecidos.
- Sistema Falho
  - Um sistema que não pode satisfazer um ou mais dos requisitos estipulados em sua especificação.

- Em geral são:
  - Reativos
    - São sistemas cujo escalonamento é dirigido pela interação com seu ambiente.
      - Exemplo: Sistema para controle de incêndios que reage ao pressionar de um botão.
  - Embarcados
    - Fazem partes de sistemas maiores não computacionais.
      - Exemplo: Controle de injeção de combustível, airbag, freios, etc.




Tanque de armazenamento de combustível controlado por sensor.

- Classificação:
  - Sistemas de tempo real são classificados de acordo com o impacto gerado por uma falha ao atender seus requisitos de tempo.
  - Ao todo existem três tipos:
    - **Soft, Firm e Hard**.

- Soft real time
  - Sistema em que o desempenho é degradado mas não resulta em falhas, no caso de não atendimento de suas restrições de tempo.
    - Exemplos: Um editor de texto, sistema de folha de pagamento, sistema de processamento de matrículas.

- Hard real time
  - Sistema em que uma falha relacionada a um único deadline pode provocar falhas completas do sistema ou até mesmo catástrofes.
    - Sistema para controle de freios, Airbag, etc.

- Firm Real Time
  - Sistema em que a perda de poucos deadlines não provocam falha total, no entanto, a perda de uma quantidade muito grande podem provocar falhas completa do sistema ou até mesmo catástrofes.
    - Sistema de controle de navegação

| Sistema | Tipo de Sistema | Cenário |
|---|---|---|
| Máquinas de auto-atendimento | Soft | Perda de muitos deadlines não provocarão falhas catastróficas, somente o desempenho é degradado |
| Sistema de navegação embutida para controle de robôs autônomos agrícolas. | Firm | Excessiva perda de deadlines podem fazer que o robô danifique toda uma plantação |
| Sistema de controle de armas em caças. | Hard | Perda de um único deadline pode fazer com que o alvo seja perdido. |

- Antes de aprofundarmos na teoria de tempo real devemos revisar/citar alguns conceitos, são eles:
  - Eventos
  - Determinismo
  - Utilização de CPU

- Qualquer acontecimento que provoque uma mudança no fluxo de controle.
- Exemplos:
  - Instanciação de um objeto, if-then, goto, case, chamada de métodos, interrupção, etc.

- Eventos Síncronos
  - Eventos que podem ser previstos no fluxo de controle.
    - Traps, desvios condicionais.
- Eventos Assíncronos
  - Acontecem em pontos imprevisíveis do fluxo de controle e são geralmente causados por fontes externas.
    - Interrupção gerada pelo clock, etc.

- Eventos Periódicos
  - Eventos que ocorrem em intervalos regulares.
    - Um clock que pulsa regularmente a cada 5 segundos.
- Eventos Aperiódicos
  - Eventos que não ocorrem em intervalos regulares.
- Eventos Esporádicos
  - Eventos aperiódicos que ocorrem raramente.

# Eventos - Tipos

| | Periódico | Aperiódico | Esporádico |
|---|---|---|---|
| Síncrono | Código Cíclico | Instruções condicionais ou incondicionais | Exceções internas |
| | Processo escalonado por um clock interno. | Coletor de Lixo | Traps |
| Assíncrono | Interrupção gerada por clock | Interrupção regular, porém sem período fixado | Exceção gerada externamente. “Eventos randômicos” |

- Em todo sistema, manter o controle é extremamente importante
- O software deve estar apto a antecipar e evitar os estados “inconsistentes”.
- Sistemas controlados por software devem trazer e executar instruções da área de memória destinada aos programas.
- O sistema deve antecipar como o sistema se comportará em todas as circunstâncias possíveis.

- Sistema Determinístico
  - Um sistema é determinístico se, para cada possível estado e cada conjunto de entradas, um único conjunto de saídas e o próximo estado do sistema podem ser determinados.
    - Se um sistema é determinístico, todos os eventos que o compõem são determinísticos.
    - Se um sistema determinístico, possui tempo de resposta conhecido para cada conjunto de respostas conhecidos, o sistema também possui determinismo temporal.

## Utilização de CPU (U)

- Medida do percentual de processamento *não ocioso.*
  - *Ver aula 4 de Evolução do PC*

# Utilização de CPU (U)

| Utilization (%) | Zone Type | Typical Application |
|---|---|---|
| 0–25 | Significant excess processing power – CPU may be more powerful than necessary | Various |
| 26–50 | Very safe | Various |
| 51–68 | Safe | Various |
| 69 | Theoretical limit | Embedded systems |
| 70–82 | Questionable | Embedded systems |
| 83–99 | Dangerous | Embedded systems |
| 100+ | Overload | Stressed systems |

## Utilização de CPU (U)

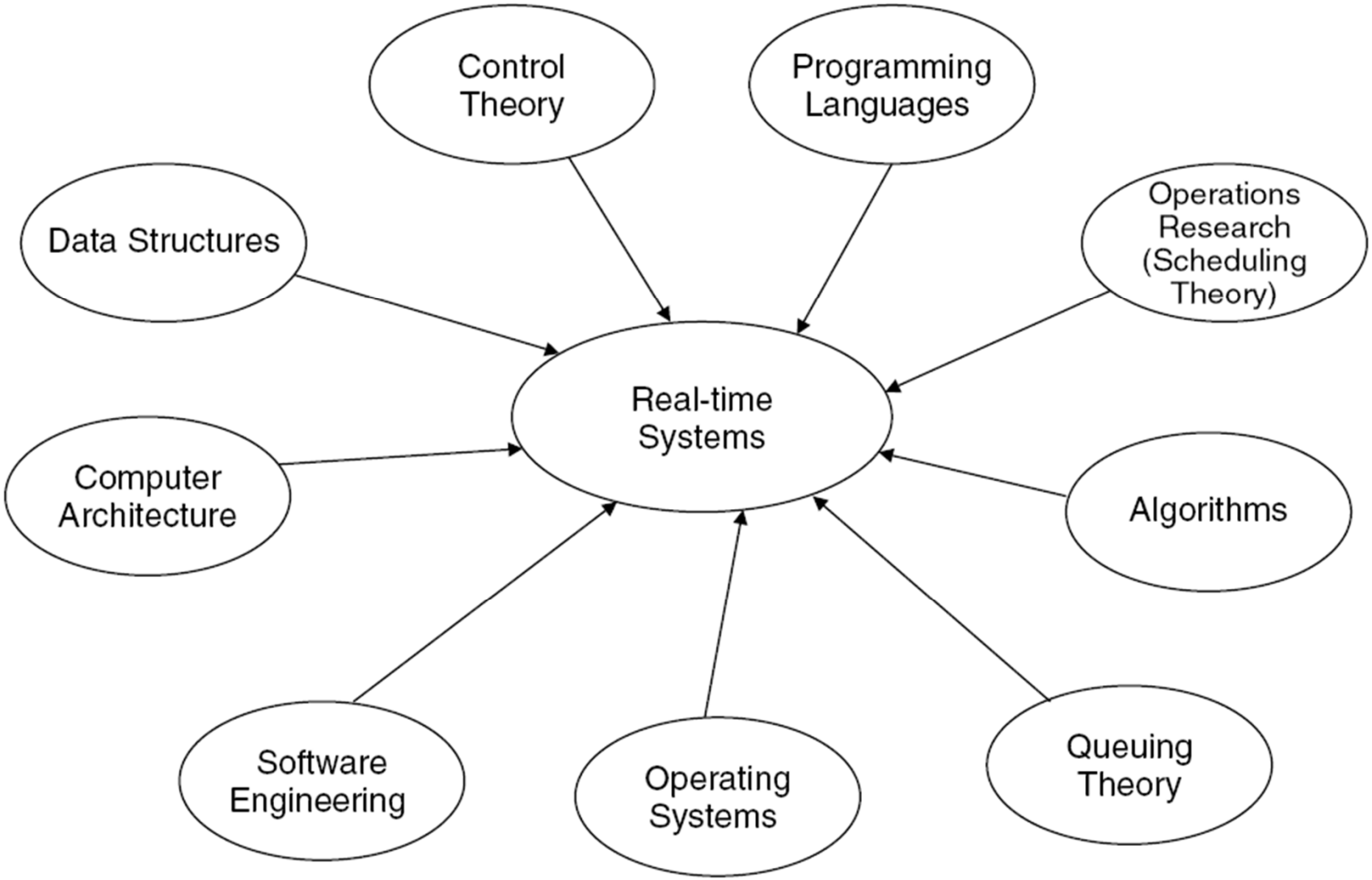
- Fórmula para Cálculo

$$U = \sum_{i}^{n} u_i = \sum_{i}^{n} \frac{e_i}{p_i},$$

$onde\ u_i\ é\ o\ fator\ de\ utilização,$

$e_i\ é\ o\ tempo\ de\ execução\ (pior\ caso)\ e$

$p_i\ é\ o\ período\ de\ execução\ (médio\ para\ tarefas\ aperiódicas)$


Control Theory
Programming Languages
Data Structures
Operations Research (Scheduling Theory)
Real-time Systems
Computer Architecture
Algorithms
Software Engineering
Operating Systems
Queuing Theory

- O projeto e implementação de STR requer atenção para numerosos problemas:
  - Seleção de hw e sw, e avaliação do trade-off necessário para uma solução barata e eficaz.
  - Cuidadosa especificação e projeto de STR, levando em consideração aspectos temporais.
  - Entendimento das características das linguagens de programação.

- Tolerância a falhas e segurança.
- Projeto e administração de teste, seleção de teste e equipamento de desenvolvimento.
- Uso de tecnologias abertas.
- Medir e predizer tempo de resposta.

- Hard Real Time vs Soft Real Time

| Característica | Hard-real time | Soft real time |
|---|---|---|
| Tempo de resposta | Requisito alto | Requisito desejável |
| Desempenho sob alta carga | Previsível | Não previsto |
| Controle | Ambiente | Computacional |
| Segurança | Geralmente Crítica | Não Crítica |
| Tamanho dos dados | Pequeno/médio | Grande |
| Tipo de redundância | Ativa | Baseada em checkpoint |
| Integridade dos dados | Curta | Longa |
| Detecção de erros | Autônoma | Assistida pelo usuário |

- Falha-seguro vs Falha-Operacional
  - Em sistemas do tipo falha-seguro, o STR ao falhar entra em um estado seguro.
    - Sistema de controle de trens.
  - Em sistemas do tipo falha-operacional, o STR ao falhar deve prover um conjunto mínimo de serviços e continuar operacional.
    - Sistema de controle de um avião

- Garantia de Resposta vs Melhor Esforço
  - Em garantia de resposta, o sistema é projetado para sempre cumprir seus deadlines.
  - Em melhor esforço, o sistema faz o melhor possível.

- Sensível a eventos e sensível a tempo
  - Um dos temas mais polêmicos em STR
  - Como o nome sugere, sistemas sensíveis a eventos reagem a eventos lançados pelo sistema (ou ambiente)
  - Sistemas sensíveis a tempo reagem em intervalos de tempo pré-definido, respectivamente.
  - Todo sistema sensível a evento pode ser modelado como um sistema sensível a eventos e vice-versa?

- Sistemas de Tempo Real são sinônimos de Sistemas Rápidos.
- Análise de taxa monotônica tem resolvido todos os problemas de tempo real.
- Há uma teoria universal, largamente aceita metodologia para especificação e projeto de sistemas de tempo real.

- Não existe necessidade por construção de novos sistemas operacionais de tempo real.
- O estudo de sistemas de tempo real se resume a teoria do escalonamento.

| Year | Landmark | Developer | Development | Innovations |
|---|---|---|---|---|
| 1947 | Whirlwind | MIT/US Navy | Flight simulator | Ferrite core memory, "real response times" |
| 1957 | SAGE | IBM | Air defense | Specifically designed for real-time |
| 1958 | Scientific 1103A | Univac | General purpose | Hardware interrupt |
| 1959 | SABRE | IBM | Airline reservation | Hub-go-ahead policy |
| 1962 | Basic Executive | IBM | General purpose | First real-time executive |
| 1963 | Basic Executive II | IBM | General purpose | Diverse real-time scheduling, Disk resident user/systems programs |

| Year | Landmark | Developer | Development | Innovations |
|---|---|---|---|---|
| 1970s | RSX, RTE | DEC, HP | Real-time operating systems | Hosted by minicomputers |
| 1973 | Rate-monotonic system | Liu and Layland | Theory | Stated upper bound on utilization for schedulable systems |
| 1980s | RMX-80, MROS 68K, VRTX, etc. | Various | Real-time operating system | Hosted by microprocessors |

- Primeiros programas
  - Escritos diretamente em microcódigo, linguagem assembly e mais tarde em linguagens de alto nível.

- Primeiras linguagens de alto nível
  - Fortran, CMS e JOVIAL (linguagens preferidas do exército, marinha e força aérea americana, respectivamente).

- Em 1983 surge ADA
  - Encomendada pelo Departamento de Defesa Americano com o intuito de:
    - Utilizar uma única linguagem para desenvolvimento de STR.
    - Possuir construções de alto-nível
    - Prover características que facilitassem a programação de STR.
  - Em 1995, lançam ADA95

- Atualmente, ADA não é uma linguagem muito aplicada em STR.
  - No entanto, algumas empresas automobilísticas (em especial, as alemãs), bem como o departamento de defesa americano e francês continuam fazendo uso dela.
- Grande parte dos sistemas atuais são desenvolvidos em C, C++, assembly, Fortran.
- Java vem ganhando força nos últimos anos. :S

- Para o banco de conhecimentos:
  - Criar um git, setar para
    - /alsmoreira
    - Recomendo GitHub ou Bitbucket
- Pesquisar sobre o seguinte tema:
  - Java e Sistemas de Tempo Real: Vale a pena utilizar?


Atlassian Bitbucket

- Farines – Capítulo 1
- Laplace – Capítulo 1
- Kopetz – Capítulo 1

# Dúvidas?

# Sistemas Embarcados – Sistemas de Tempo Real

Instituto Federal de Pernambuco
Coordenação de Informática/IFPE

Anderson L. S. Moreira
anderson.moreira@recife.ifpe.edu.br