ARQUITETURA E ORGANIZAÇÃO DE COMPUTADORES

ENGENHARIA DA COMPUTAÇÃO – UFC/SOBRAL

Prof. Wendley S. Silva

Slides adaptados dos originais do prof. Joniel Bastos

ORGANIZAÇÕES DE SISTEMAS DE COMPUTADORES

UNIVERSIDADE FEDERAL DO CEARÁ

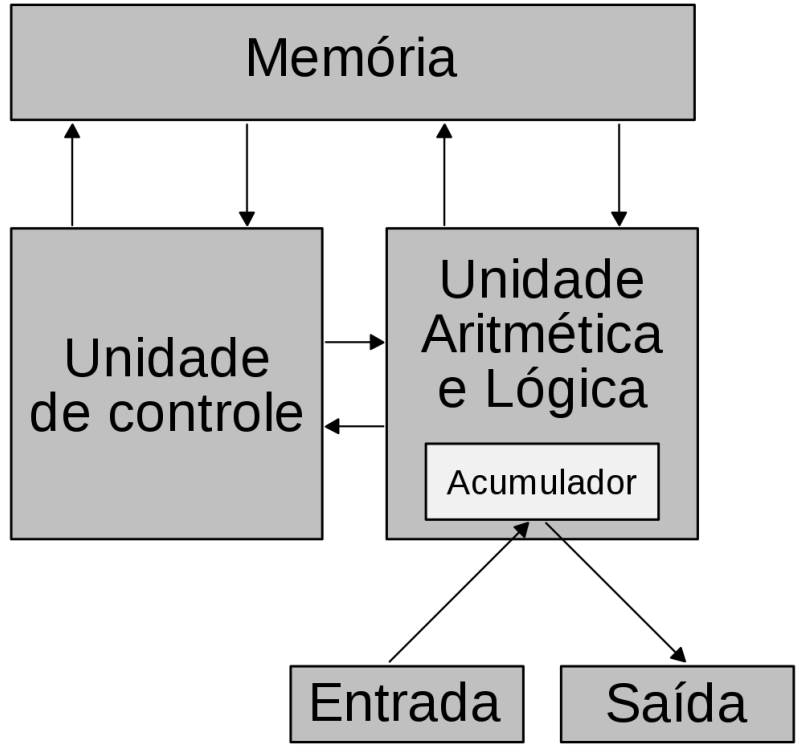
- Processadores
- Memória
  - Memória primária
  - Memória secundária
- Entrada e saída

PROCESSADORES

A UNIDADE CENTRAL DE PROCESSAMENTO E SUA COMPOSIÇÃO

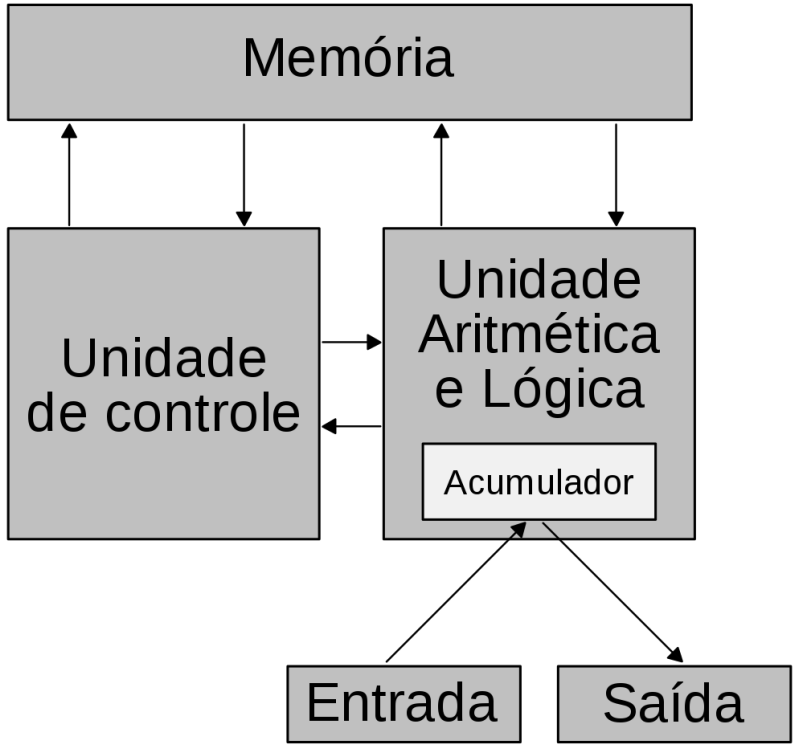
ARQUITETURA DE VON NEUMANN

- É composta por três grandes pilares:
  - Unidade de Processamento Central (CPU)
  - Sistema de memória
  - Sistema de entrada e saída



ARQUITETURA DE VON NEUMANN

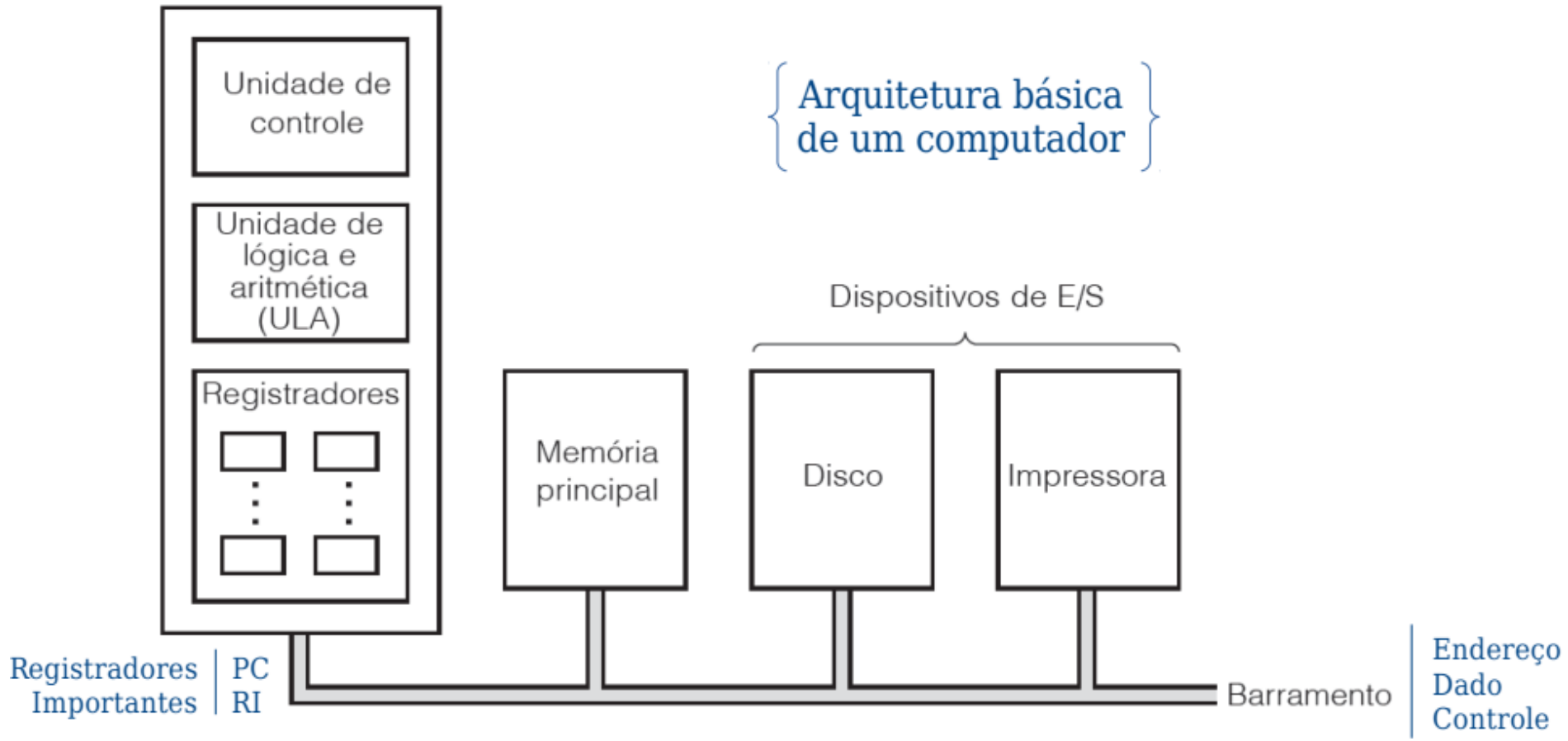
- Gargalo de Von Neumann
  - Limitação na taxa de transferência entre a CPU e a memória em comparação com a quantidade de memória
  - Transferência é menor do que a taxa que o processador consegue trabalhar e menor do que a quantidade de memória em geral disponível
  - Gera desperdício de tempo (CPU em espera)



ARQUITETURA DE VON NEUMANN

UNIVERSIDADE FEDERAL DO CEARÁ

- Organização de um computador simples com uma CPU e dois dispositivos de E/S



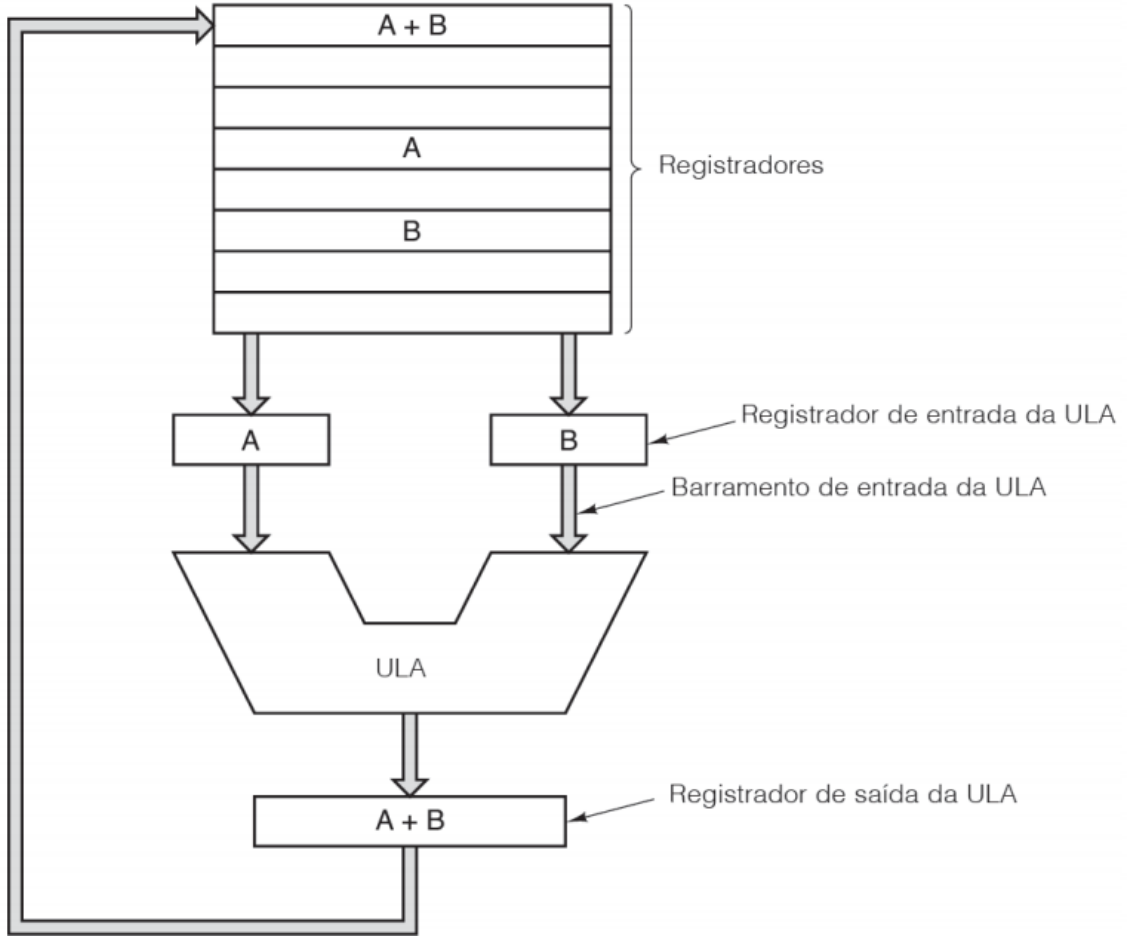
ARQUITETURA DE VON NEUMANN

UNIVERSIDADE FEDERAL DO CEARÁ

- ULA: Um circuito digital que realiza operações lógicas e aritméticas. A ULA é uma peça fundamental da unidade central de processamento (CPU)
- Banco de registradores: é uma pequena área de armazenamento para os dados que o processador está usando, são acessados mais rapidamente do que os dados armazenados no sistema de memória, geralmente suportam vários acessos simultâneos
- Unidade de controle: Responsável pelo controle do resto do processador, determinando quando as instruções podem ser executadas e quais operações são necessárias para executar cada instrução
- Contador de Programa (PC)
- Registrador de Instrução (IR)

ORGANIZAÇÃO DA CPU

- Caminho de dados
  - Composta por registrados (1 a 32), ULA e barramentos que conectam as partes
- Tipos de instruções:
  - Registrador-memória
  - Registrador-registrador



EXECUÇÃO DE INSTRUÇÕES

UNIVERSIDADE FEDERAL DO CEARÁ

- A CPU executa cada instrução em uma série de pequenas etapas. Em termos simples, as etapas são as seguintes:
  1. Trazer a próxima instrução da memória até o registrador de instrução
  2. Alterar o contador de programa para que aponte para a próxima instrução
  3. Determinar o tipo de instrução trazida
  4. Se a instrução usar uma palavra na memória, determinar onde essa palavra está
  5. Trazer a palavra para dentro de um registrador da CPU, se necessário
  6. Executar a instrução
  7. Voltar à etapa 1 para iniciar a execução da instrução seguinte
- Ciclo **buscar-decodificar-executar**

EQUIVALÊNCIA NO PROJETO DO COMPUTADOR

UNIVERSIDADE FEDERAL DO CEARÁ

- A equivalência entre processadores de hardware e interpretadores é de grande importância para a organização de computadores
- Após especificar uma linguagem de máquina L para o computador, os projetistas devem decidir se irão desenvolver um processador de hardware ou um interpretador
  - Se a equipe decide por utilizar um interpretador, ela deve providenciar a máquina que irá executar o interpretador
  - É possível ainda uma construção híbrida

EQUIVALÊNCIA NO PROJETO DO COMPUTADOR

UNIVERSIDADE FEDERAL DO CEARÁ

- Quando usar um interpretador?
  - Quando o conjunto de instruções que a linguagem possui é muito complexa e com muitas opções, o que significa alto custo para o desenvolvimento do hardware
  - Um interpretador subdivide as instruções da máquina em diversas etapas
- Consequência: a máquina na qual o interpretador roda deve ser muito mais simples e menos cara do que seria um processador de hardware

EQUIVALÊNCIA NO PROJETO DO COMPUTADOR

UNIVERSIDADE FEDERAL DO CEARÁ

- Máquinas interpretadas custariam menos para serem produzidas, pois o custo de desenvolvimento de hardware, é maior do que de desenvolvimento de software
- No entanto, máquinas que executam as instruções diretamente são mais rápidas
- Essa equivalência permitiu criar computadores de diferentes custos e velocidade que fazem basicamente as mesmas coisas
  - Implementação em hardware era usada em modelos mais caros (mais rápidos), enquanto que a interpretação era usada em modelos mais baratos (mais lentos)

EXECUÇÃO DE INSTRUÇÕES

- Computadores simples com instruções interpretadas tinham benefícios, entre os quais os mais importantes eram:
  1. A capacidade de corrigir em campo instruções executadas incorretamente ou até compensar deficiências de projeto no hardware básico
  2. A oportunidade de acrescentar novas instruções a um custo mínimo, mesmo após a entrega da máquina
  3. Projeto estruturado que permitia desenvolvimento, teste e documentação eficientes de instruções complexas
- Tais benefícios permitiu a criação de computadores mais baratos e com conjuntos de instruções cada vez mais complexos
  - O VAX da Digital Equipment Corporation (DEC) tinha várias centenas de instruções e mais de 200 modos diferentes de especificar os operandos a serem usados em cada instrução
  - Entretanto, a falta de desempenho foi fatal para o VAX e para a DEC

LINHAS DE ARQUITETURA

RISC *VERSUS* CISC

LINHAS DE ARQUITETURA

UNIVERSIDADE FEDERAL DO CEARÁ

- Os projetistas tentavam fechar a "lacuna semântica" entre o que as máquinas podiam fazer e o que as linguagens de programação de alto nível demandavam.
- Arquitetura CISC – Complex Instruction Set Computer
  - Arquitetura cujo conjunto de instruções era bastante complexo
  - Capaz de executar centenas de instruções complexas diferentes sendo extremamente versátil
  - Exemplos de processadores CISC são os 386 e os 486 da Intel

LINHAS DE ARQUITETURA

UNIVERSIDADE FEDERAL DO CEARÁ

- Arquitetura RISC – Reduced Instruction Set Computer
  - Arquitetura cujo conjunto de instruções era simplificado (reduzido)
  - Favorece um conjunto simples e pequeno de instruções que levam aproximadamente a mesma quantidade de tempo para serem executadas
  - Exemplos de processadores RISC são os Alpha, SPARC, MIPS, e PowerPC
- Computadores atuais utilizam uma implementação híbrida, misturando as duas linhas

PROJETO PARA COMPUTADORES MODERNOS

UNIVERSIDADE FEDERAL DO CEARÁ

- Há um conjunto de princípios de projeto, às vezes denominados princípios de projeto RISC, que os arquitetos de CPUs de uso geral se esforçam por seguir:

1. Todas as instruções são executadas diretamente por hardware
   - Eliminar um nível de interpretação dá alta velocidade para a maioria das instruções
   - Em computadores CISC, instruções complexas podem ser subdivididas em partes separadas que então podem ser executadas como uma sequência de microinstruções
     - Torna a máquina mais lenta, mas pode ser aceitável para instruções que ocorrem com menos frequência

PROJETO PARA COMPUTADORES MODERNOS

UNIVERSIDADE FEDERAL DO CEARÁ

2. É preciso maximizar a taxa de execução das instruções – MIPS
   - MIPS (Microprocessor without Interlocked Pipeline Stages) – microprocessador sem estágios paralelos de interbloqueio
3. Instruções devem ser fáceis de decodificar
   - Instruções regulares, de comprimento fixo, pequeno número de campos
4. Somente LOAD e STORE devem referenciar a memória
5. É preciso providenciar muitos registradores

PARALELISMO

COMO É USADO PARA MELHORAR O DESEMPENHO DAS MÁQUINAS

PARALELISMO

UNIVERSIDADE FEDERAL DO CEARÁ

- Duas formas gerais:
  - No nível de instrução – O paralelismo é explorado dentro de instruções individuais para obter da máquina mais instruções por segundo
  - No nível de processador – Várias CPUs trabalham juntas no mesmo problema
- Cada abordagem tem seus próprios méritos

PIPELINING (PARALELISMO, CANALIZAÇÃO)

UNIVERSIDADE FEDERAL DO CEARÁ

Fábrica de Bolos

1. Aquisição da matéria-prima
2. Processo de produção do bolo
3. Processo de embalagem para expedição

PIPELINING (PARALELISMO, CANALIZAÇÃO)

UNIVERSIDADE FEDERAL DO CEARÁ

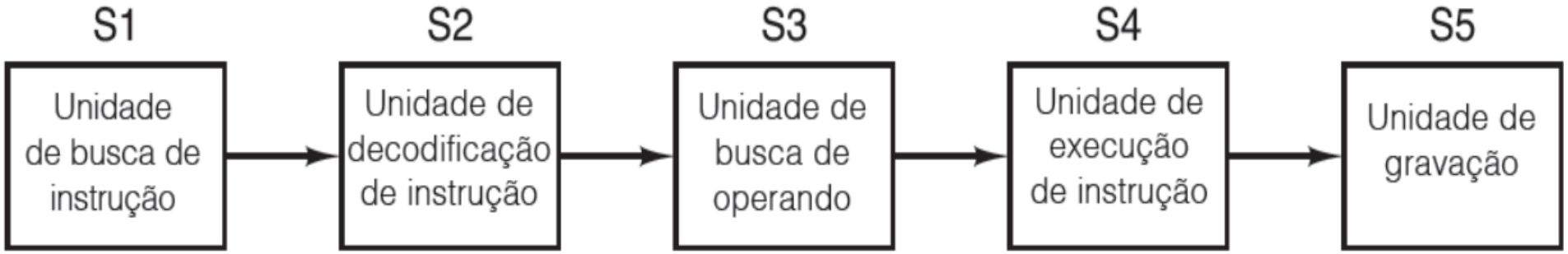
Processo de Embalagem

1. A cada 10 segundos, o funcionário 1 coloca na esteira uma embalagem de bolo vazia
2. A embalagem é transportada até o funcionário 2, que coloca o bolo dentro dela
3. Na estação seguinte, o funcionário 3 fecha e sela a embalagem
4. A embalagem já selada com o bolo dentro parte para o funcionário 4, que acrescenta uma etiqueta com descrições sobre o bolo contido
5. A embalagem já selada com o bolo dentro parte para o funcionário 4, que acrescenta uma etiqueta com descrições sobre o bolo contido

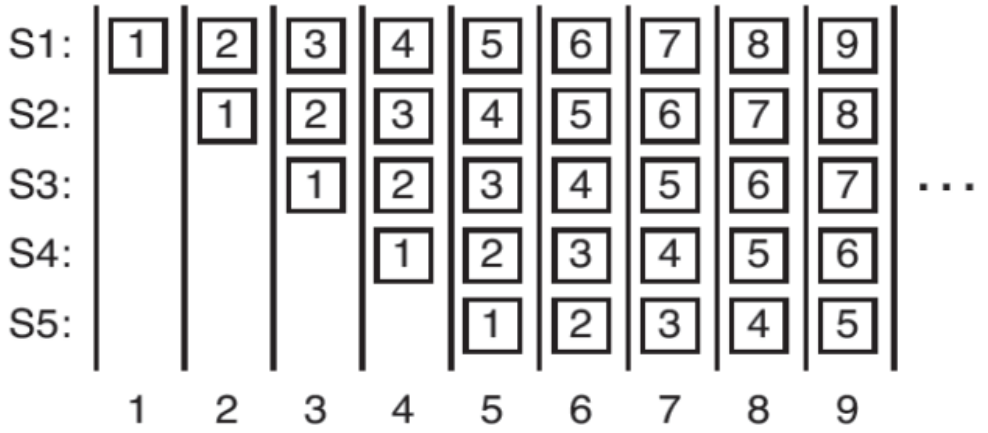
PIPELINING (PARALELISMO, CANALIZAÇÃO)

UNIVERSIDADE FEDERAL DO CEARÁ

- Pipeline de cinco estágios



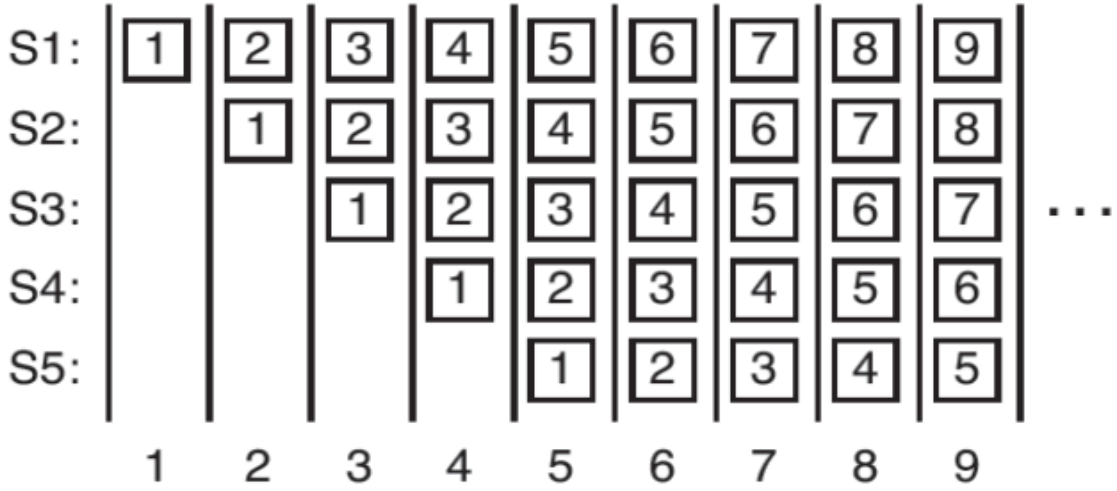
- Estado de cada estágio como uma função do tempo
- **Latência**
  - o tempo que demora para executar uma instrução
- **Largura de Banda** (quantos MiPS a CPU tem)
- Ciclo de T ns e n estágios no pipeline, a latência é nT.



PIPELINING (PARALELISMO, CANALIZAÇÃO)

- Pipeline de cinco estágios

- Suponhamos tempo de ciclo de 2 ns:
  - Uma instrução = 10 ns
  - Parece 100 MIPS
  - Na realidade opera a 500 MIPS



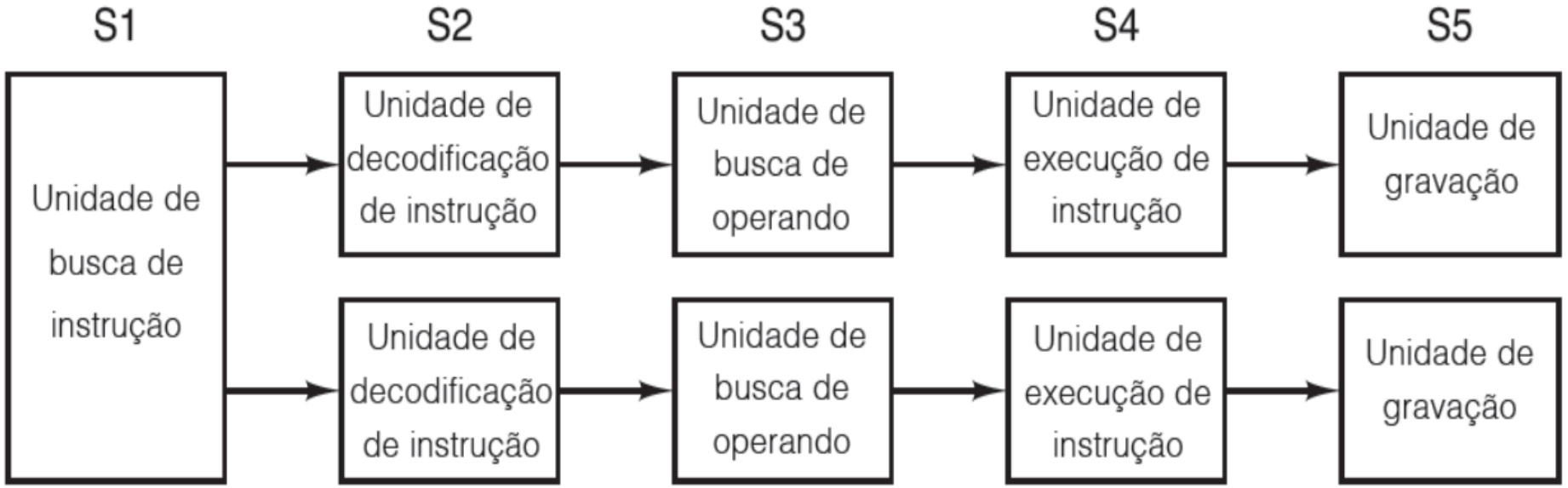
PIPELINING (PARALELISMO, CANALIZAÇÃO)

UNIVERSIDADE FEDERAL DO CEARÁ

- **Largura de Banda de processador** (quantos MiPS a CPU tem)
  - Ciclo de **T** *ns* e **n** estágios no pipeline, a latência é nT ns.
  - Uma instrução é concluída a cada ciclo de clock e que há 10^9/T ciclos de clock por segundo, o número de instruções executadas por segundo é 10^9 /T.

PIPELINING (PARALELISMO, CANALIZAÇÃO)

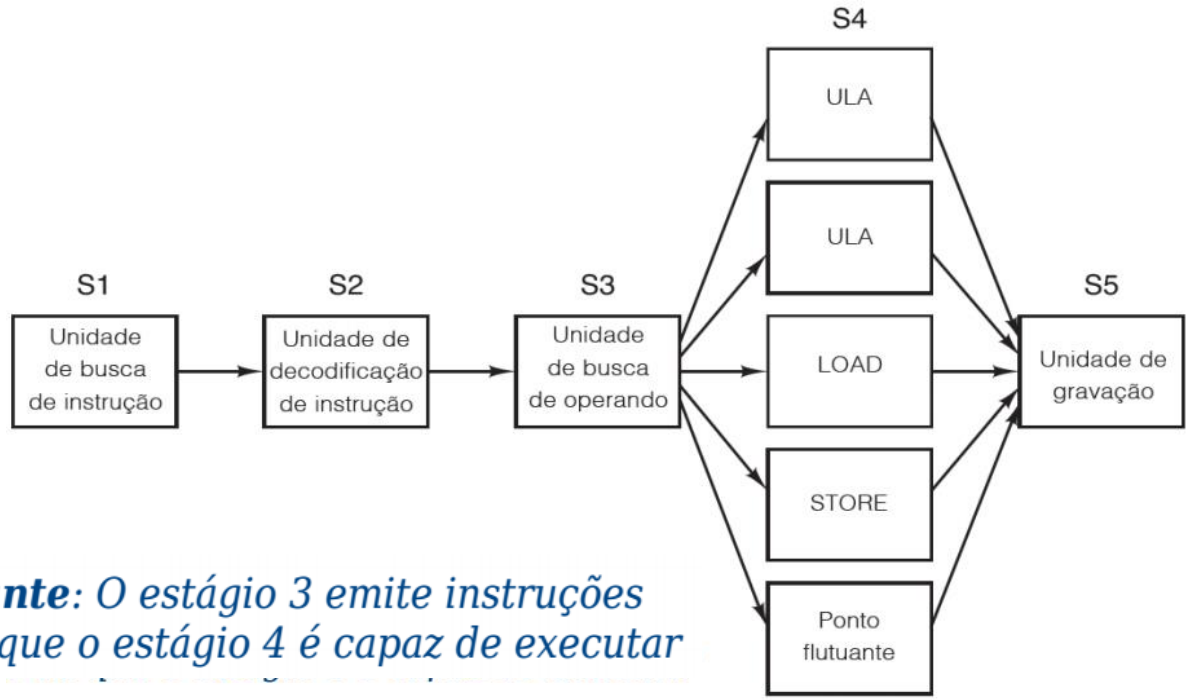
- Pipelines duplos de cinco estágios com uma unidade de busca de instrução em comum



- O compilador deve garantir que não haja dependência entre os pipelines ou conflitos devem ser detectados por hardware extra

PIPELINING (PARALELISMO, CANALIZAÇÃO)

- Duplicar o número de pipelines exigia aumentar a complexidade do hardware
- A ideia básica é ter apenas um único pipeline, mas lhe dar várias unidades funcionais:
- Superescalar



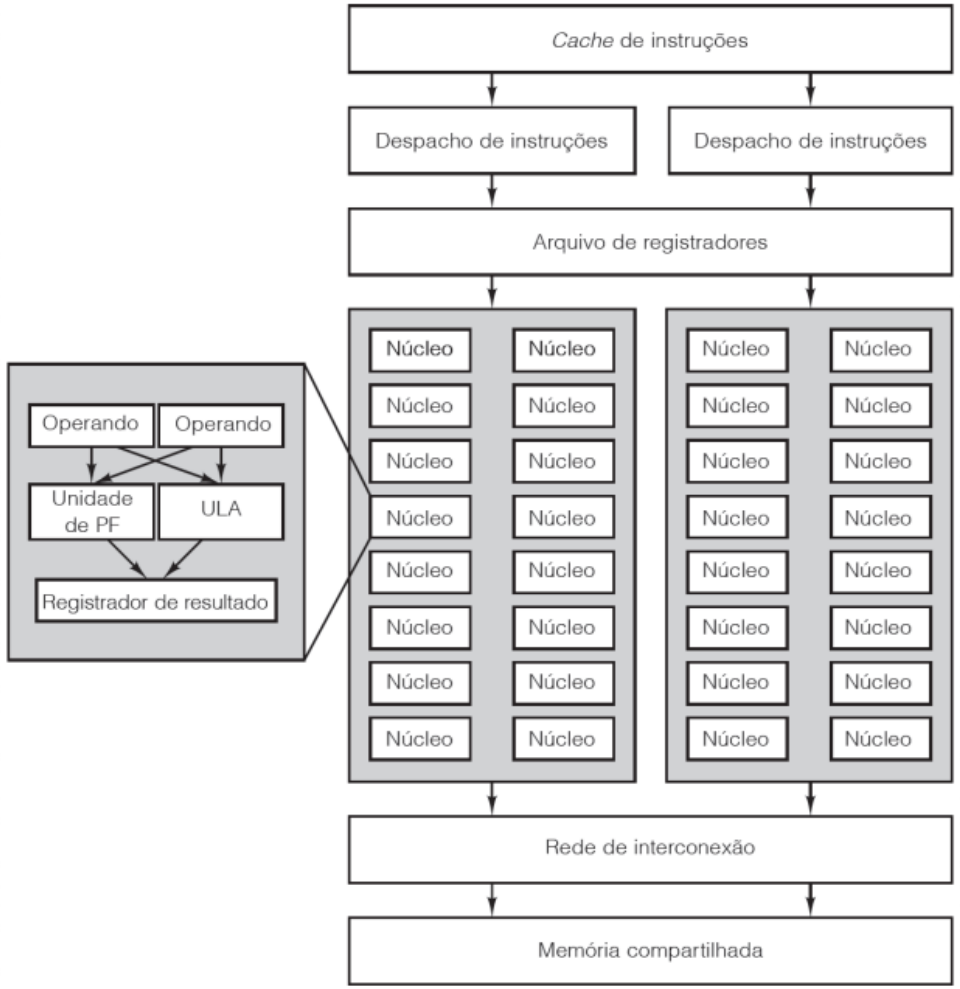
***Observação importante**: O estágio 3 emite instruções muito mais rápido do que o estágio 4 é capaz de executar*

PARALELISMO NO NÍVEL DO PROCESSADOR

UNIVERSIDADE FEDERAL DO CEARÁ

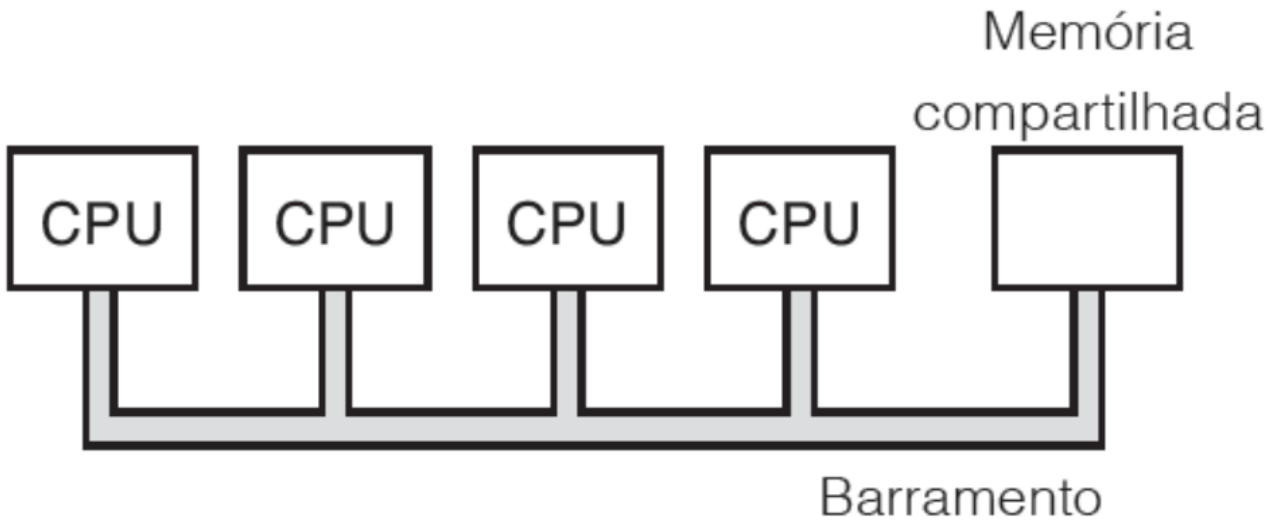
- Um processador SIMD (Single Instruction-stream Multiple data-stream) consiste em um grande número de processadores idênticos que efetuam a mesma sequência de instruções sobre diferentes conjuntos de dados
  - Requer regularidade entre as instruções, como operações em matrizes
  - Ideal para processamento gráfico
- As modernas unidades de processamento de gráficos (GPUs) contam bastante com o processamento SIMD para fornecer poder computacional maciço com poucos transistores
  - Uma menor quantidade de transistores proporciona menos geração de calor

PARALELISMO NO NÍVEL DO PROCESSADOR



PARALELISMO NO NÍVEL DO PROCESSADOR

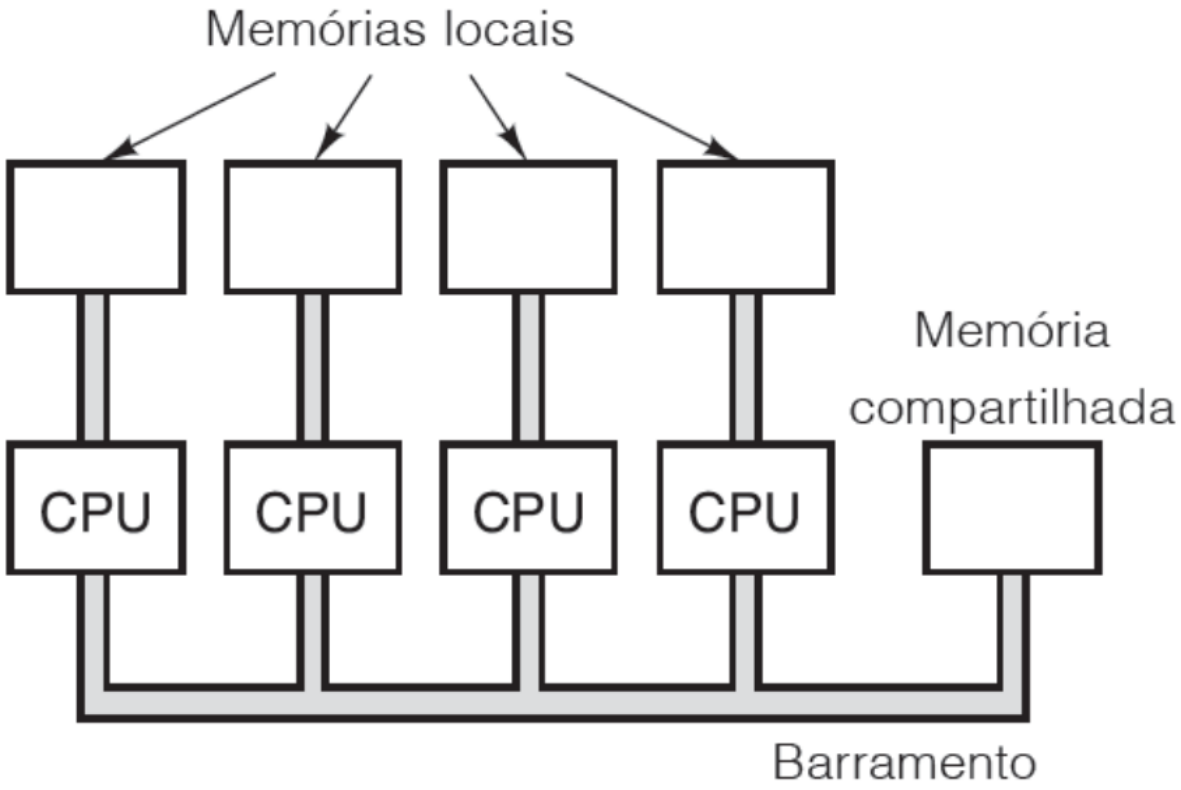
- Multiprocessador de barramento único



- Cada CPU é independente – cada uma possui sua própria unidade de controle, enquanto que processadores SIMD possuem uma única UC compartilhada entre os vários processadores

PARALELISMO NO NÍVEL DO PROCESSADOR

- Multicomputador com memórias locais



PARALELISMO NO NÍVEL DO PROCESSADOR

UNIVERSIDADE FEDERAL DO CEARÁ

- Multicomputador
  - Dificuldade está em conectar todos os processadores à memória
  - Grande número de computadores conectados com sua própria memória
  - Enviam mensagem entre eles

PARALELISMO NO NÍVEL DO PROCESSADOR

UNIVERSIDADE FEDERAL DO CEARÁ

- Costuma-se dizer que as CPUs de um multicomputador são fracamente acopladas, em contraste com as CPUs fortemente acopladas de um multiprocessador
- As CPUs de um multicomputador se comunicam enviando mensagens umas às outras
- Multiprocessadores são mais fáceis de programar
- Multicomputadores são mais fáceis de construir

REFERÊNCIAS

- TANENBAUM, A. S. Organização Estruturada de Computadores. Editora LTC, 5 ed, Rio de Janeiro, 2007.
- STALLINGS, W. Arquitetura e Organização de Computadores. Editora Prentice Hall, 5 edição, 2002.