Arquitetura de Computadores

Paralelismo:
Computação de Alto Desempenho

Prof. Dr. Daniel Caetano

2022 - 1

Compreendendo o problema

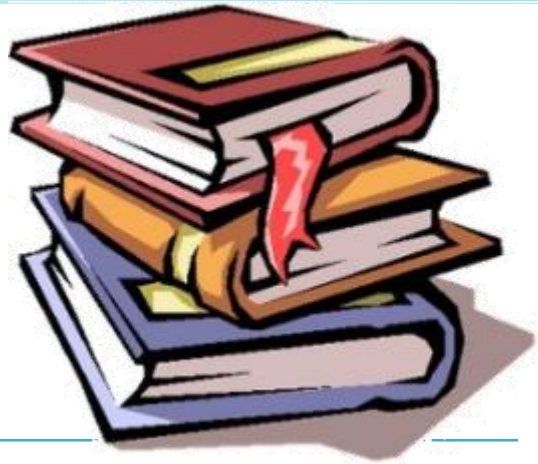
- **Situação:** Hoje temos computadores com vários processadores e vários núcleos de processamento.

Quais são as tecnologias?

Objetivos

- Compreender o funcionamento do PipeLine e da Superpipeline
- Compreender o funcionamento da Arquitetura Superescalar
- Compreender a Arquitetura SMP
- Apresentar o Conceito de Processamento Vetorial

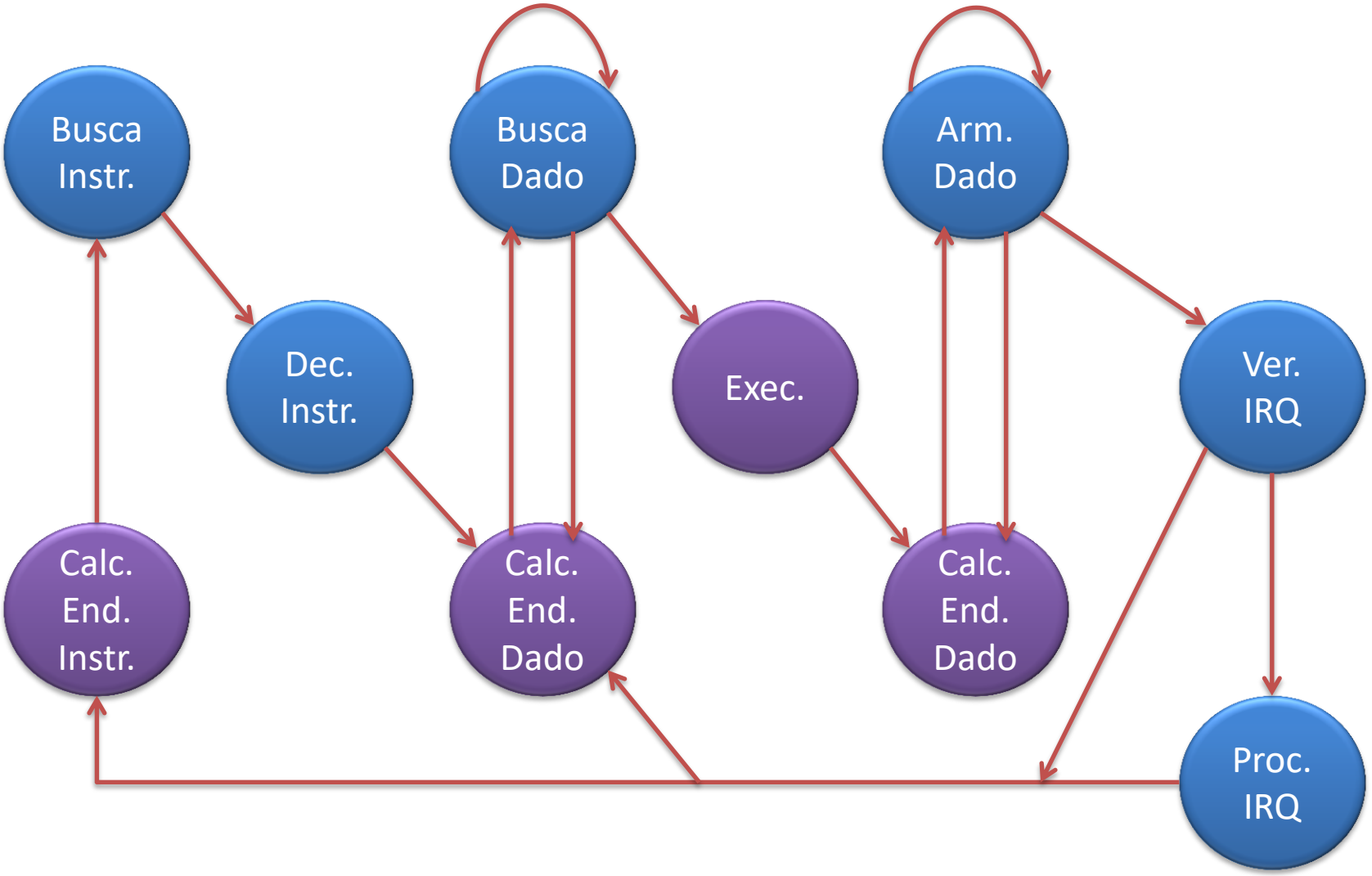
Material de Estudo



Material	Acesso ao Material
Notas de Aula e Apresentação	https://www.caetano.eng.br/aulas/2022a/ara0039.php (Arquitetura de Computadores – Aula 10)
Biblioteca Virtual	• Arquitetura e Organização de Computadores (Stallings)
Material Adicional	1) Paralelismo em nível de instruções: https://eaulas.usp.br/portal/video.action?idItem=14545 2) Sistemas computacionais – Paralelismo e Memória: https://www.youtube.com/watch?v=zZBv2ThqxI8

Retomando:
Ciclo de Execução de Instrução

Diagrama do Ciclo de Instrução



Execução com Pipeline

Pipeline

- Conceito de Linha de Produção
  - Quebrar tarefa complexa em tarefas menores
  - Ex.: Fazer carro
    - Fazer roda
    - Fazer motor
    - Fazer lataria
    - ...
- Por que aplicar isso para CPU?
  - Quando a memória é acessada, a ULA fica ociosa
  - Duas etapas: **busca (UC)** e **execução (ULA)**

Partes da CPU em Funcionamento

- Simplificadamente... sem pipeline

Tempo	Busca(UC)	Execução(ULA)
0	I1	-
1	-	I1
2	I2	-
3	-	I2
4	I3	-
5	-	I3

- E assim por diante...

Partes da CPU em Funcionamento

- Sim **2 ciclos por instrução!**

Tempo	Busca(UC)	Execução(ULA)
0	I1	-
1	-	I1
2	I2	-
3	-	I2
4	I3	-
5	-	I3

- E assim por diante...

Partes da CPU em Funcionamento

- Simplificadamente... **COM** pipeline

Tempo	Busca(UC)	Execução(ULA)
0	I1	-
1	I2	I1
2	I3	I2
3	I4	I3
4	I5	I4
5	I6	I5

- E assim por diante...

Partes da CPU em Funcionamento

- Sim **1 ciclo por instrução!**

Tempo	Busca(UC)	Execução(ULA)
0	I1	-
1	I2	I1
2	I3	I2
3	I4	I3
4	I5	I4
5	I6	I5

- E assim por diante...

Partes da CPU em Funcionamento

- Comparando lado a lado...

Sequência no Tempo	SEM pipeline		COM pipeline	
	Busca	Execução	Busca	Execução
0	I1	-	I1	-
1	-	I1	I2	I1
2	I2	-	I3	I2
3	-	I2	I4	I3
4	I3	-	I5	I4

- Maior eficiência: + instruções, - tempo

Pipeline de Múltiplos Níveis

- Quebrar processamento em 6 etapas
  - **BI**: Busca de Instruções
  - **DI**: Decodificação de Instruções
  - **CO**: Cálculo de Operandos
  - **BO**: Busca de Operandos
  - **EI**: Execução da Instrução
  - **EO**: Escrita de Operando
- Cada etapa dura 0,33 comparado com as anteriores
- Como é o processamento com tudo isso?

Pipeline de Múltiplos Níveis

- Sem pipeline

Tempo	BI	DI	CO	BO	EI	EO
0,00	I1	-	-	-	-	-
0,33	-	I1	-	-	-	-
0,66	-	-	I1	-	-	-
1,00	-	-	-	I1	-	-
1,33	-	-	-	-	I1	-
1,66	-	-	-	-	-	I1
2,00	I2	-	-	-	-	-

Pipeline de Múltiplos Níveis

- Sen

2 ciclos por instrução!

Tempo	BI	DI	CO	BO	EI	EO
0,00	I1	-	-	-	-	-
0,33	-	I1	-	-	-	-
0,66	-	-	I1	-	-	-
1,00	-	-	-	I1	-	-
1,33	-	-	-	-	I1	-
1,66	-	-	-	-	-	I1
2,00	I2	-	-	-	-	-

Pipeline de Múltiplos Níveis

- **Com** pipeline

Tempo	BI	DI	CO	BO	EI	EO
0,00	I1	-	-	-	-	-
0,33	I2	I1	-	-	-	-
0,66	I3	I2	I1	-	-	-
1,00	I4	I3	I2	I1	-	-
1,33	I5	I4	I3	I2	I1	-
1,66	I6	I5	I4	I3	I2	I1
2,00	I7	I6	I5	I4	I3	I2

Pipeline de Múltiplos Níveis

- **Co**

0,33 ciclos por instrução!

3 instruções por ciclo!

Temp						
0,00						
0,33	I2	I1	-	-	-	-
0,66	I3	I2	I1	-	-	-
1,00	I4	I3	I2	I1	-	-
1,33	I5	I4	I3	I2	I1	-
1,66	I6	I5	I4	I3	I2	I1
2,00	I7	I6	I5	I4	I3	I2

Pipeline de Múltiplos Níveis

- Isso nem sempre é perfeito...
  - E se o resultado de I2 depende de I1?

- I1 - LD A, 10 ; A = 10
- I2 - ADD A, 20 ; A = A + 20
- I3 - LD B, A ; B = A

- Qual o valor de B no final?
- Qual o valor de B se I1 e I2 forem invertidos?

Pipeline de Múltiplos Níveis

- Se I2 depende de I1, I2 tem que esperar I1

Tempo	BI	DI	CO	BO	EI	EO
0,00	I1	-	-	-	-	-
0,33	I2	I1	-	-	-	-
0,66	I3	I2	I1	-	-	-
1,00	I4	I3	I2	I1	-	-
1,33	I5	I4	I3	I2	I1	-
1,66	I5	I4	I3	I2	-	I1
2,00	I6	I5	I4	I3	I2	-

Pipeline de Múltiplos Níveis

- Se I1

Quanto mais níveis de pipeline, maior a chance de ocorrer esse tipo de dependência e espera!

Temp						
0,00						
0,33						
0,66						
1,00						
1,33	I5	I4	I3	I2	I1	-
1,66	I5	I4	I3	I2	-	I1
2,00	I6	I5	I4	I3	I2	-

Pipeline de Múltiplos Níveis

- Intel chegou a usar cerca de 20 níveis de pipeline
- Problemas
  - O processador passa a **aquecer demais**!
  - Pipeline depende da previsão de sequencia de instruções (um **if** pode estragar tudo!)
  - A partir de um certo nível número de níveis, a probabilidade de "**um if estragar tudo**" é alta
- Pentium 4 abandonado...
- voltaram ao Pentium M (P3)... PD, c2 e i3/5/7

Arquitetura Superpipeline

Pipeline de Múltiplos Estágios

- Todos os estágios demoram o mesmo tempo?

Tempo	BI	DI	CO	BO	EI	EO
0,00	I1	-	-	-	-	-
0,33	I2	I1	-	-	-	-
0,66	I3	I2	I1	-	-	-
1,00	I4	I3	I2	I1	-	-
1,33	I5	I4	I3	I2	I1	-
1,66	I6	I5	I4	I3	I2	I1
2,00	I7	I6	I5	I4	I3	I2

Superpipeline

- No pipeline tradicional
  - **Tempo da instrução:** Limitado pelo estágio mais lento
- Exemplo:

ϕ	Tempo	E1	E2	E3	E4
1	0,00	I1	-	-	-
2	0,50	I2	I1	-	-
3	1,00	I3	I2	I1	-
4	1,50	I4	I3	I2	I1
5	2,00	I5	I4	I3	I2
6	2,50	I6	I5	I4	I3
7	3,00	I7	I6	I5	I4

Tempos Iguais
En = 1 ciclo

1 Instrução por Ciclo

Superpipeline

- No pipeline tradicional
  - **Tempo da instrução:** Limitado pelo estágio mais lento
- Exemplo:

ϕ	Tempo	E1	**E2**	E3	E4
1	0,00	I1	-	-	-
2	0,50	I2	I1	-	-
3	1,00	I2	I1	-	-
4	1,50	I3	I2	I1	-
5	2,00	I3	I2	-	I1
6	2,50	I4	I3	I2	-
7	3,00	I4	I3	-	I2

E1/E3/E4 = 1 ciclo

E2 = 2 ciclos

0,5 Instrução por Ciclo

Superpipeline

- Tempo de cada estágio: controlado pelo clock
  - Se estágio lento: 2 ciclos de clock...
  - Uma instrução a cada 2 ciclos
- Acelerar o clock geral... resolve?
  - Acelera todos os estágios, mas...
  - Se estágio lento: 2 ciclos de clock...
  - Continua uma instrução a cada 2 ciclos!
  - **Periféricos e memória podem deixar de funcionar!**
- O que fazer, então?

Superpipeline

- Exemplo Superpipeline
- Exemplo:

E1/E3/E4 = 1 ciclo
E2 = 2 ciclos internos

ϕe	ϕi	t	E1	**E2**	E3	E4
1	1	0,00	I1	-	-	-
1	2	0,25	I2	I1	-	-
2	3	0,50	I2	I1	-	-
2	4	0,75	I3	I2	I1	-
3	5	2,00	I3	I2	-	I1
3	6	2,50	I4	I3	I2	-
4	7	3,00	I4	I3	-	I2

1 Instrução por Ciclo **externo**

Superpipeline

- Permitir um **clock interno** diferente (+ rápido)
  - Execução mais rápida dos estágios mais lentos
  - Exemplo: clock interno = 2x clock externo...
    - Agora estágio lento é cumprido em 2 ciclos **internos**
    - Mas cada ciclo externo equivale a dois internos...

- Superpipeline: pipeline com clock interno da CPU maior que o clock externo

ARQUITETURA SUPERESCALAR

Paralelismo em Nível de Instruções

- Arquitetura pipeline: domínio inicial da Intel
- AMD vinha “colada”... Intel tinha que inovar
  - *“Executar várias instruções ao mesmo tempo!”*
  - Múltiplos pipelines: instruções distribuídas
  - Ao invés de acelerar uma fábrica...
    - Vamos ter várias fábricas!
- Sempre posso executar instruções simultaneamente?
- **Instruções Independentes**
- O que são?

Instruções Independentes

- Exemplo...
- LD A, 17 ; Carrega A com 17
- ADD A,20 ; Soma 20 em A
- Quanto vale A?

- Se inverter as instruções o resultado é o mesmo?

- ADD A, 20 ; Soma 20 em A
- LD A,17 ; Carrega A com 17
- Quanto vale A?

- O resultados dependem da ordem das instruções

Instruções Independentes

- Vejamos um outro programa
- LD A, 17 ; Carrega A com 17
- ADD A,20 ; Soma 20 em A
- LD C, A ; Guarda resultado (37) em C
- LD A, 30 ; Carrega A com 30
- ADD A, 10 ; Soma 10 em A
- LD D, A ; Guarda resultado (40) em D
- Essas linhas são dependentes?

Instruções Independentes

- Vejamos um outro programa
- LD A, 17 ; Carrega A com 17
- ADD A,20 ; Soma 20 em A
- LD C, A ; Guarda resultado (37) em C
- LD A, 30 ; Carrega A com 30
- ADD A, 10 ; Soma 10 em A
- LD D, A ; Guarda resultado (40) em D
- Essas linhas são dependentes?
- Vamos trocar os registradores...
  - Trocar A por B na parte “azul”

Instruções Independentes

- Vejamos um outro programa
- LD A, 17 ; Carrega A com 17
- ADD A,20 ; Soma 20 em A
- LD C, A ; Guarda resultado (37) em C
- LD **B**, 30 ; Carrega B com 30
- ADD **B**, 10 ; Soma 10 em B
- LD D, **B** ; Guarda resultado (40) em D
- Essas linhas são dependentes?
- E vamos reorganizar as linhas...

Instruções Independentes

- Vejamos um outro programa

```
LD      A, 17       ; Carrega A com 17
LD      B, 30       ; Carrega B com 30
ADD     A,20        ; Soma 20 em A
ADD     B, 10       ; Soma 10 em B
LD      C, A        ; Guarda resultado (37) em C
LD      D, B        ; Guarda resultado (40) em D
```

- Ainda são dependentes?

Instruções Independentes

- Vejamos um outro programa

```
LD    A, 17
LD    B, 30
ADD   A,20
ADD   B, 10
LD    C, A
LD    D, B
```

Otimizador de código para processadores superescalares

- Ainda são dependentes?

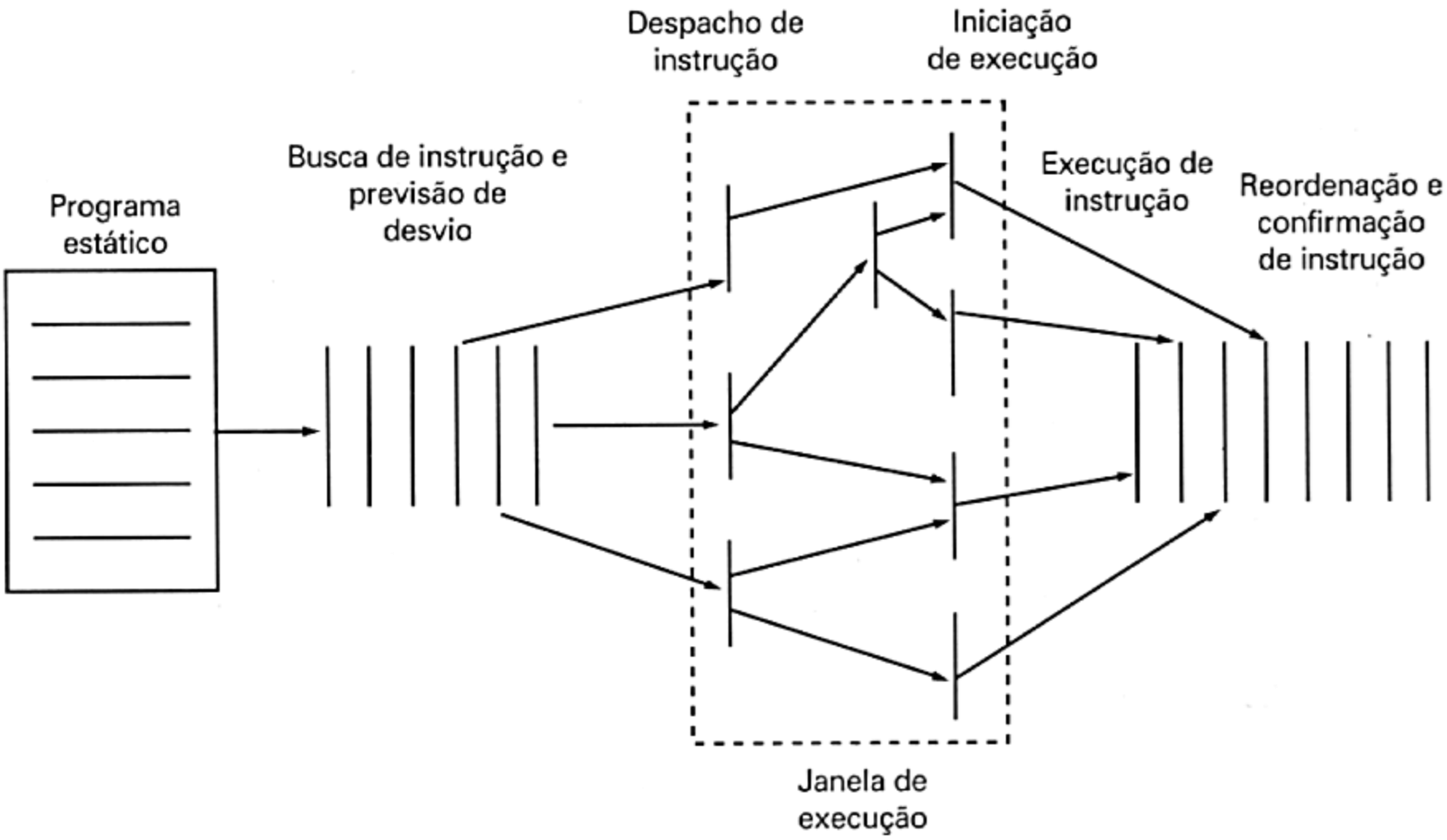
Instruções Independentes

- Vejamos um outro programa
-
-
-
-
-
- LD D, B ; Guarda resultado (40) em D

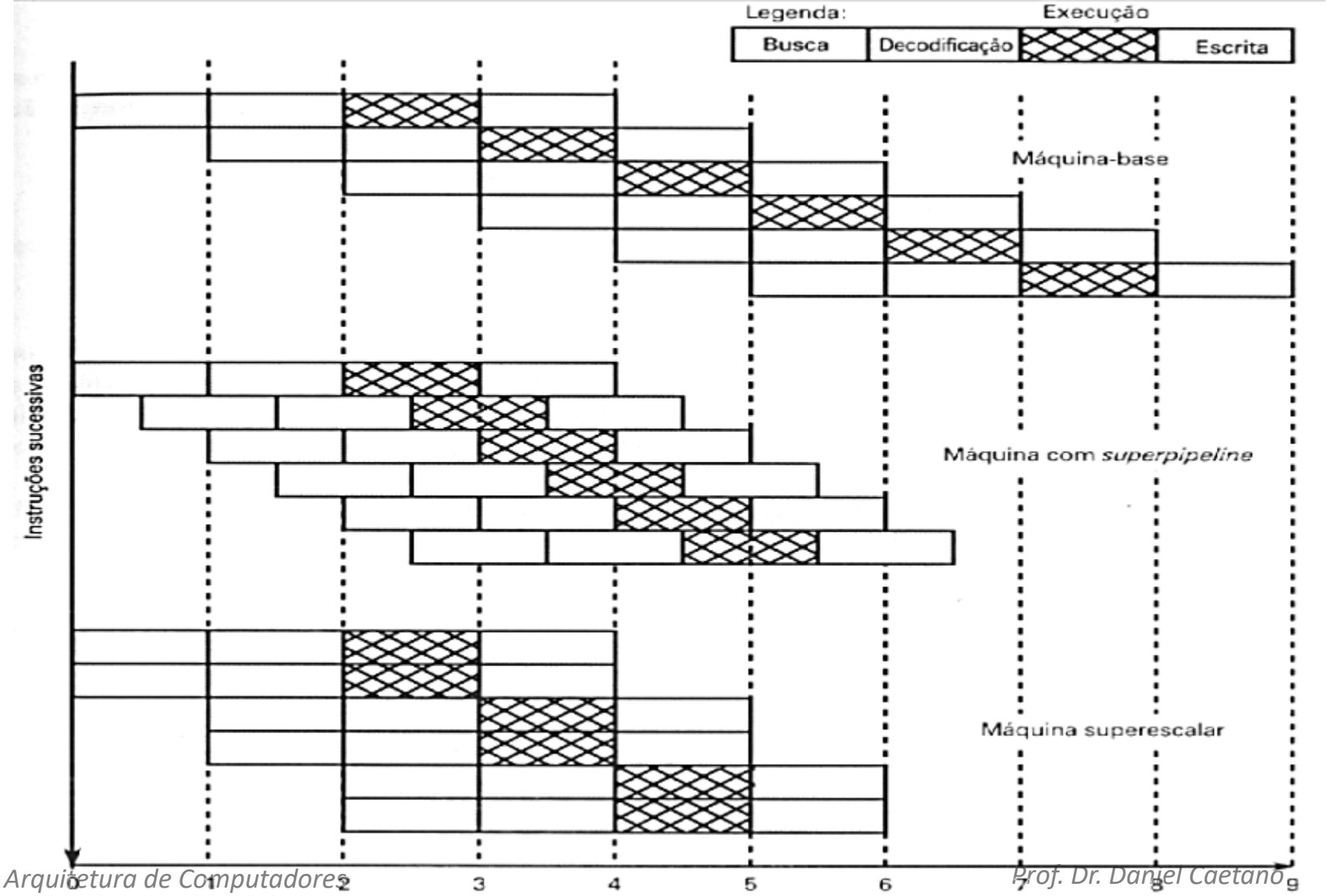
Superescalar:
Excelente para
processamento de imagens:
Pixels são Independentes!

- Ainda são dependentes?

Lógica Superescalar



(Super)Pipeline x Superescalar



Tipos de Multiprocessamento

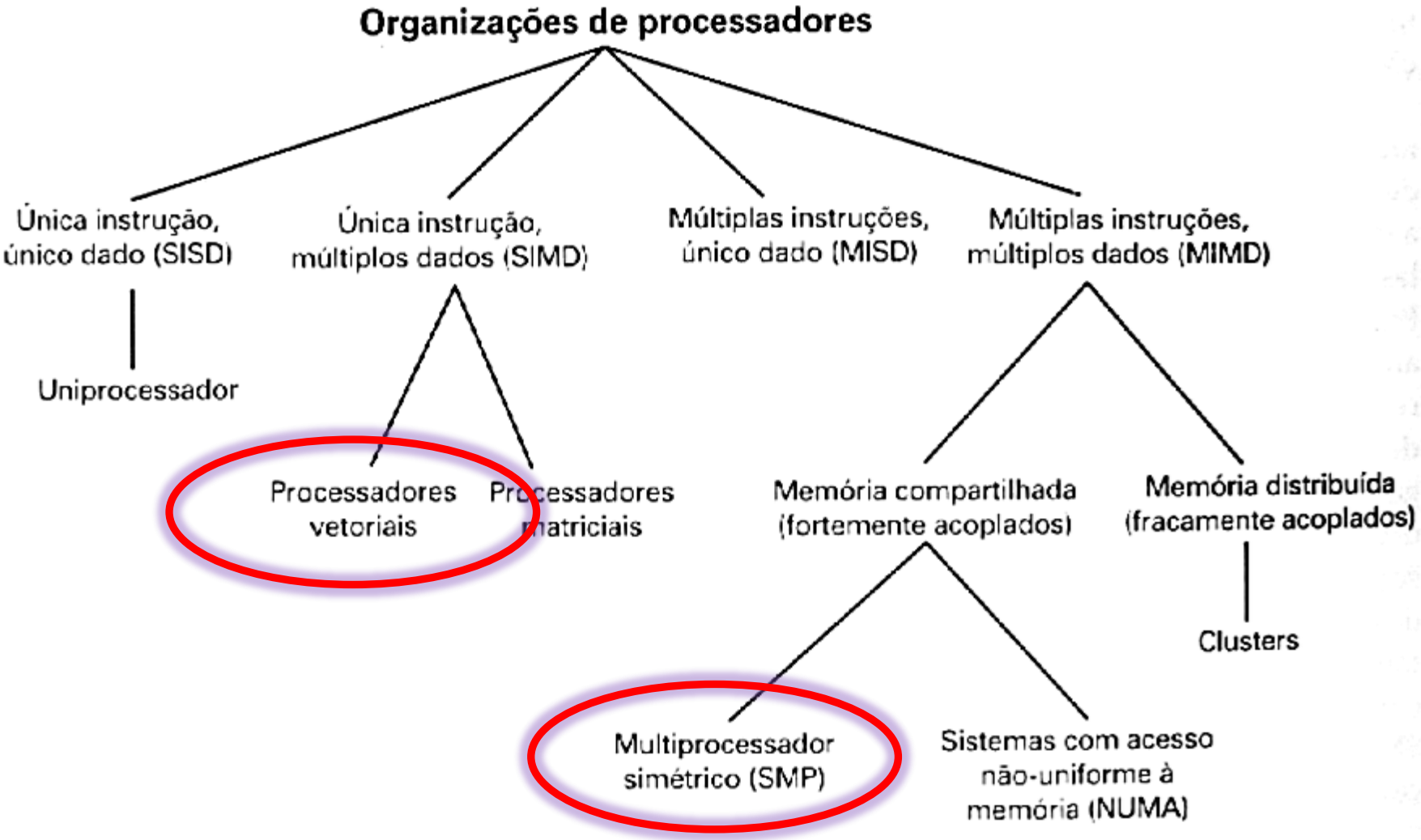
Tipos de Multiprocessamento

- Superescalar
  - Paralelismo em nível de instruções
- Múltiplos programas...
  - Paralelismo em nível de processos?
- Os tipos mais conhecidos são
  - **SMP**: Symetric MultiProcessing
  - **Clusters**
- Conceito: Computador Completo

Tipos de Multiprocessamento

- Classificação do Processamento:
  - **SISD**: Single Instruction, Single Data
    - Pipelines e Processadores Escalares
  - **MIMD**: Multiple Instruction, Multiple Data
    - SMP e Clusters
  - **SIMD**: Single Instruction, Multiple Data
    - Processamento Vetorial e Matricial
  - **MISD**: Multiple Instruction, Single Data
    - Teórico!

Tipos de Multiprocessamento



ARQUITETURA SMP

Arquitetura SMP

- SMP: Symmetric MultiProcessing
  - Queda de custos, crescente demanda...
  - Mais comum nos PCs modernos

Arquitetura SMP

- 5 Características Básicas
  1. 2 ou + processadores (capacidades comparáveis)
  2. CPUs compartilham memória/barramento
     - Tempo de acesso praticamente igual entre elas
  3. Dispositivos de E/S compartilhados
  4. CPUs executam mesmas funções (simetria)
  5. SO permite integração (processos/arquivos/dados)
- SO quem divide as tarefas!

Arquitetura SMP

- 4 Potenciais Vantagens
  1. **Desempenho** (SMP x UNI)
  2. **Disponibilidade** (falência de CPUs)
  3. **Crescimento Incremental** (Adic. Processadores)
  4. **Escalabilidade** (custos/desempenho variados)

- Atualmente: economia de energia
  - Desligar CPUs inativas
  - Reduzir a velocidade de processamento (clock)

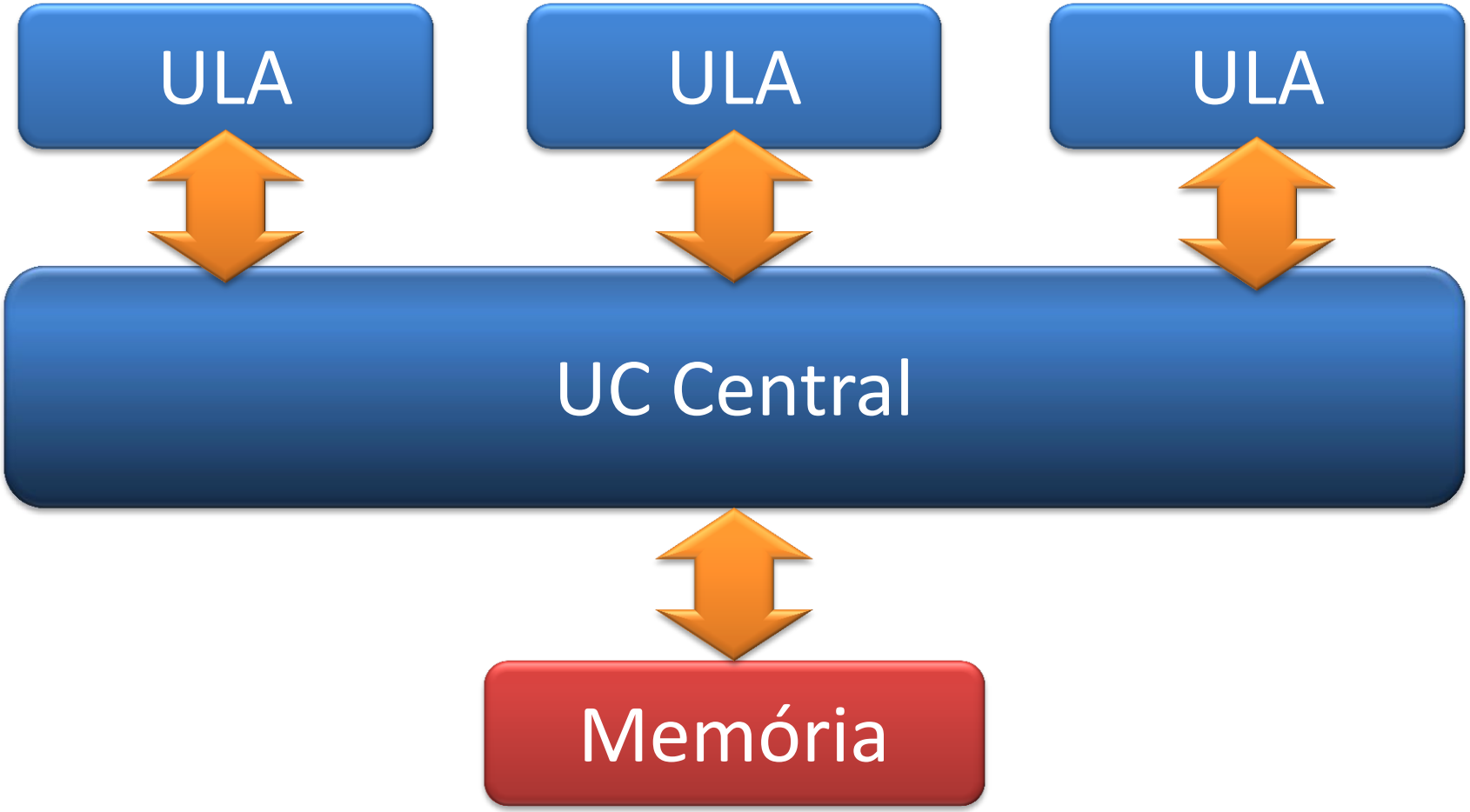
Organização SMP

- Principal desafio
  - Resolver conflitos: CPUs x Memória
  - Coordenar os acessos

- Várias estratégias de implementação
  - Unidade de Controle Central
  - Tempo Compartilhado
  - Memórias com Múltiplas Portas

SMP: Unidade de Controle Central

- Estratégia Original: 1 UC, várias ULAs

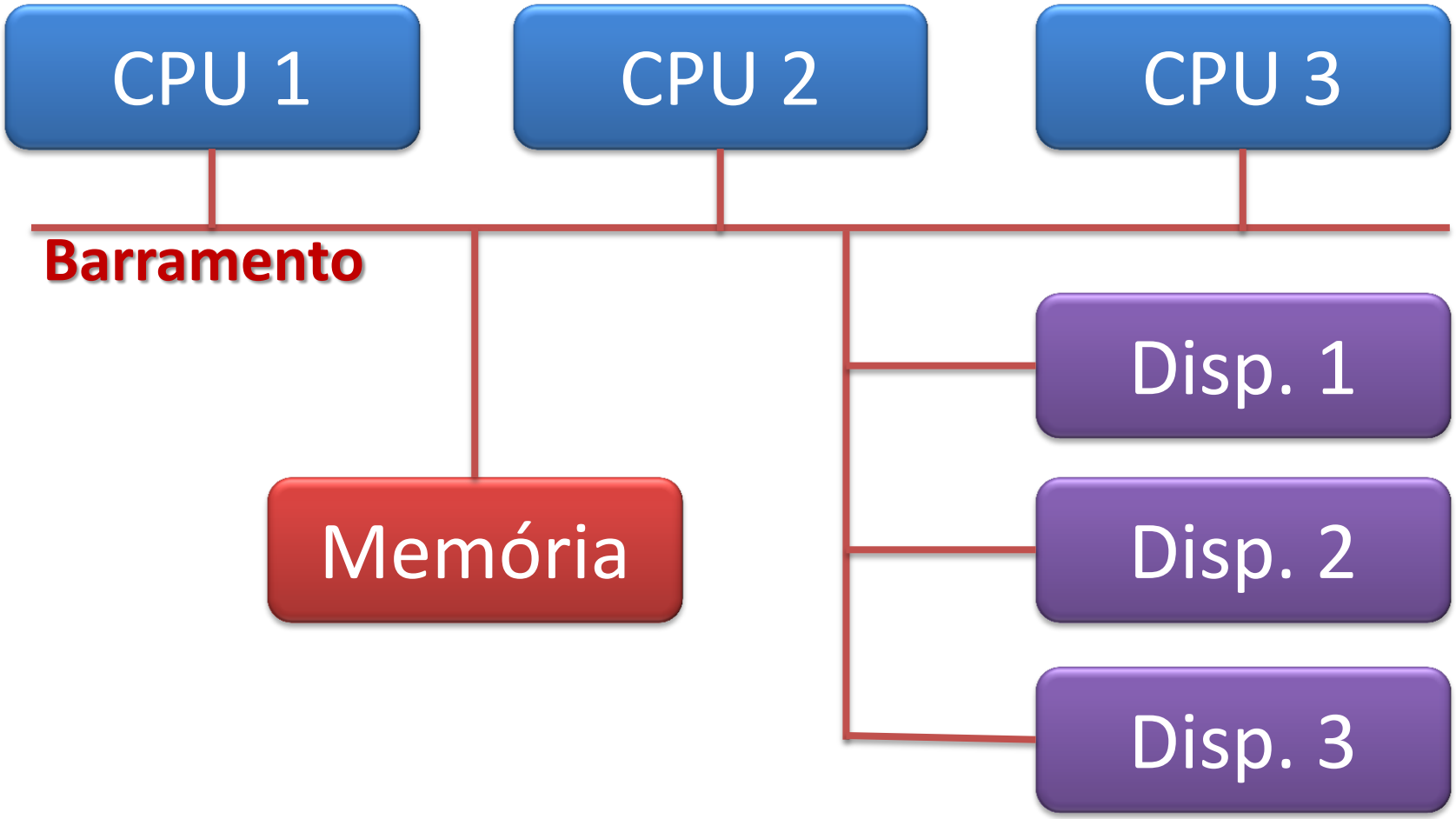


SMP: Unidade de Controle Central

- Desenvolvido pela IBM
- Muito usado entre 1960 e 1970
- Caiu em desuso
  - Alta complexidade
  - Alto custo de desenvolvimento

SMP: Tempo Compartilhado

- Tempo Compartilhado ou Barramento Comum

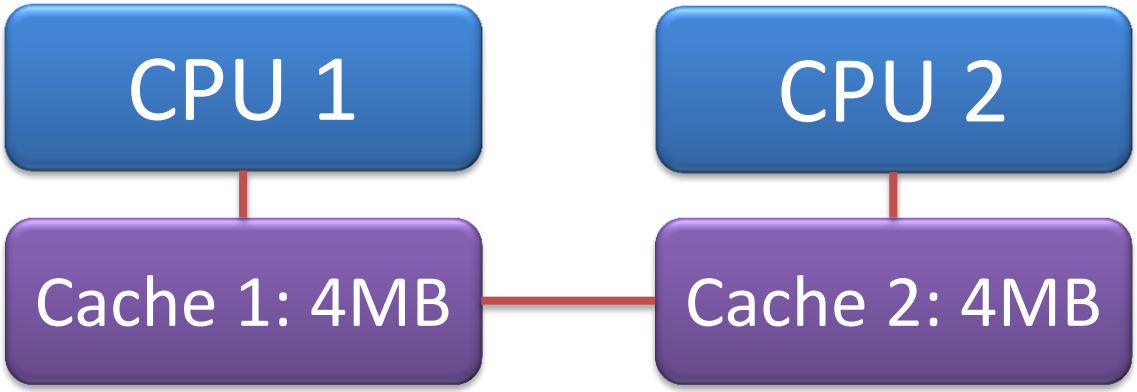


SMP: Tempo Compartilhado

- Vantagens
  - **Simplicidade** (mais simples, parecido com UNI)
  - **Flexibilidade** (adicionar processadores)
  - **Confiabilidade** (barramento passivo x falha CPU)
- Desvantagens
  - Complexidade do gerenciamento do barramento?
  - Gargalo no acesso à memória
    - Cache por CPU x Coerência de Cache

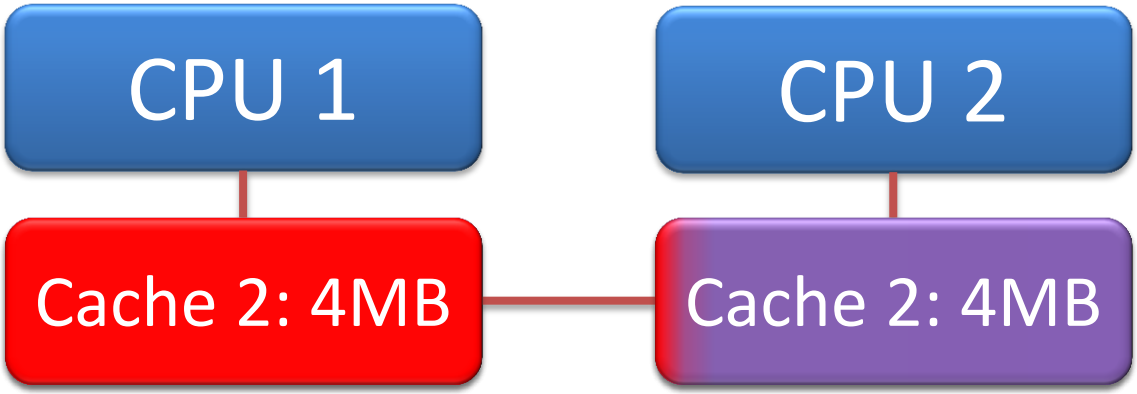
SMP: Tempo Compartilhado

- AMD: Iniciou com Cache Fixo



SMP: Tempo Compartilhado

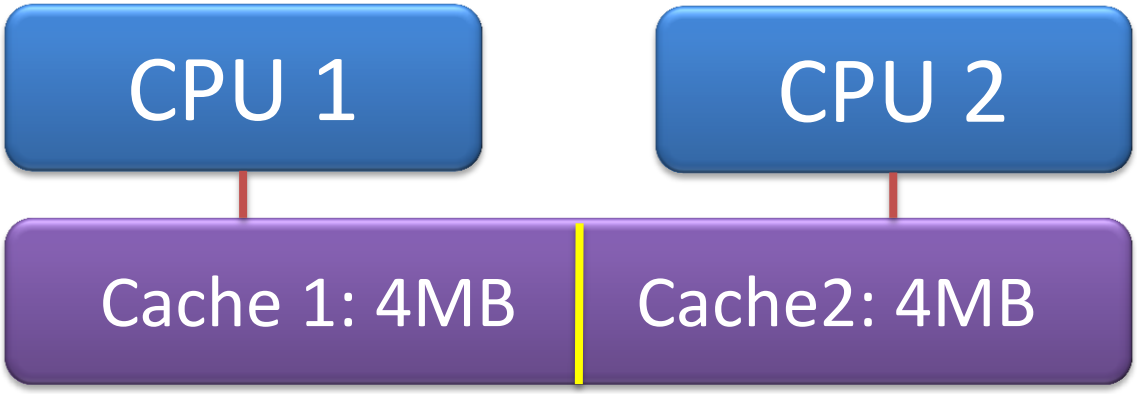
- AMD: Iniciou com Cache Fixo



- Situação:
  - CPU1: processo usando todo o cache (e quer mais!)
  - CPU2: processo não usando quase nenhum cache
- Uso ineficiente do cache → desempenho pior

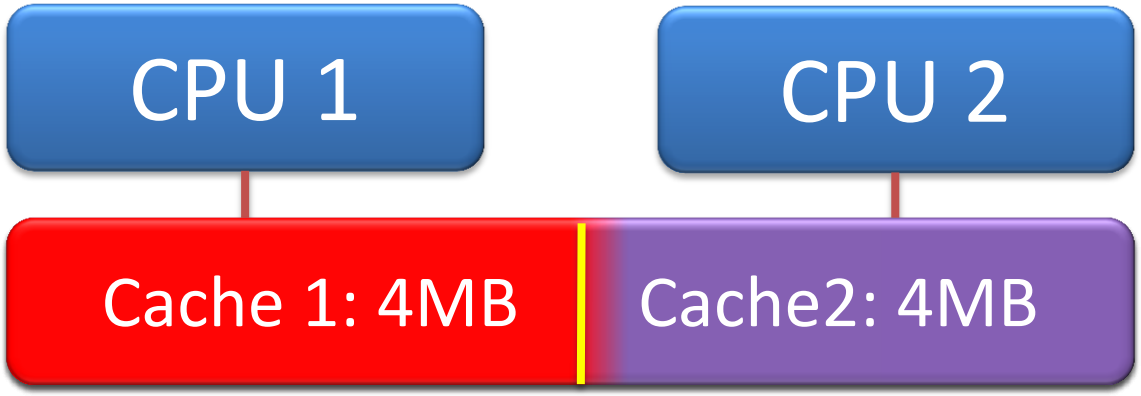
SMP: Tempo Compartilhado

- Intel: Introduziu o Cache Dinâmico



SMP: Tempo Compartilhado

- Intel: Introduziu o Cache Dinâmico



- Situação:
  - CPU1: processo usando todo o cache (e quer mais!)
  - CPU2: processo não usando quase nenhum cache
- O que acontece?

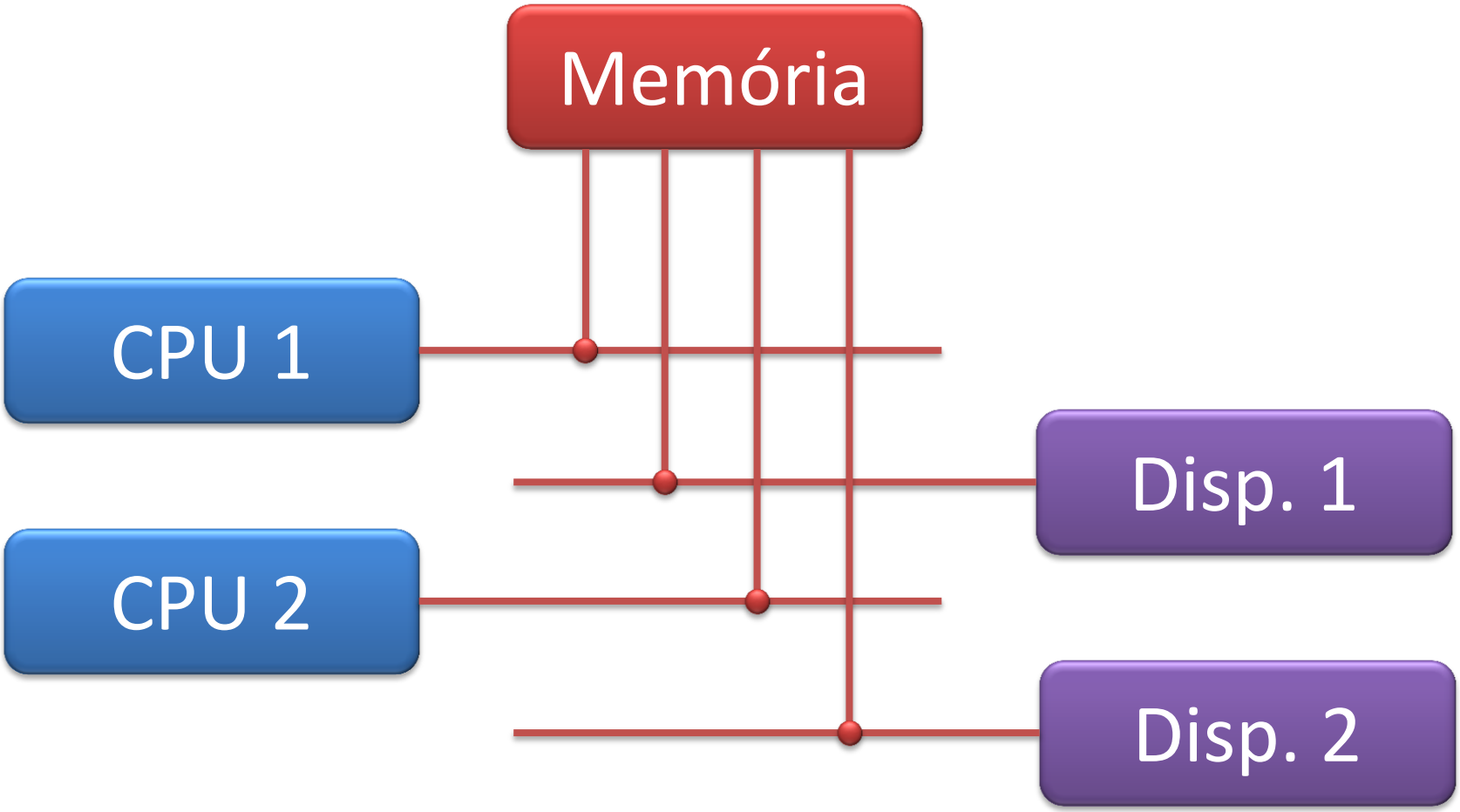
SMP: Tempo Compartilhado

- Intel: Cache Dinâmico

- Situação:
  - CPU1: processo usando todo o cache (e quer mais!)
  - CPU2: processo não usando quase nenhum cache
- Reajuste do Tamanho dos Caches
  - Muito mais eficiente!

SMP: Multiport Memory

- Acesso à Memória Diferenciado

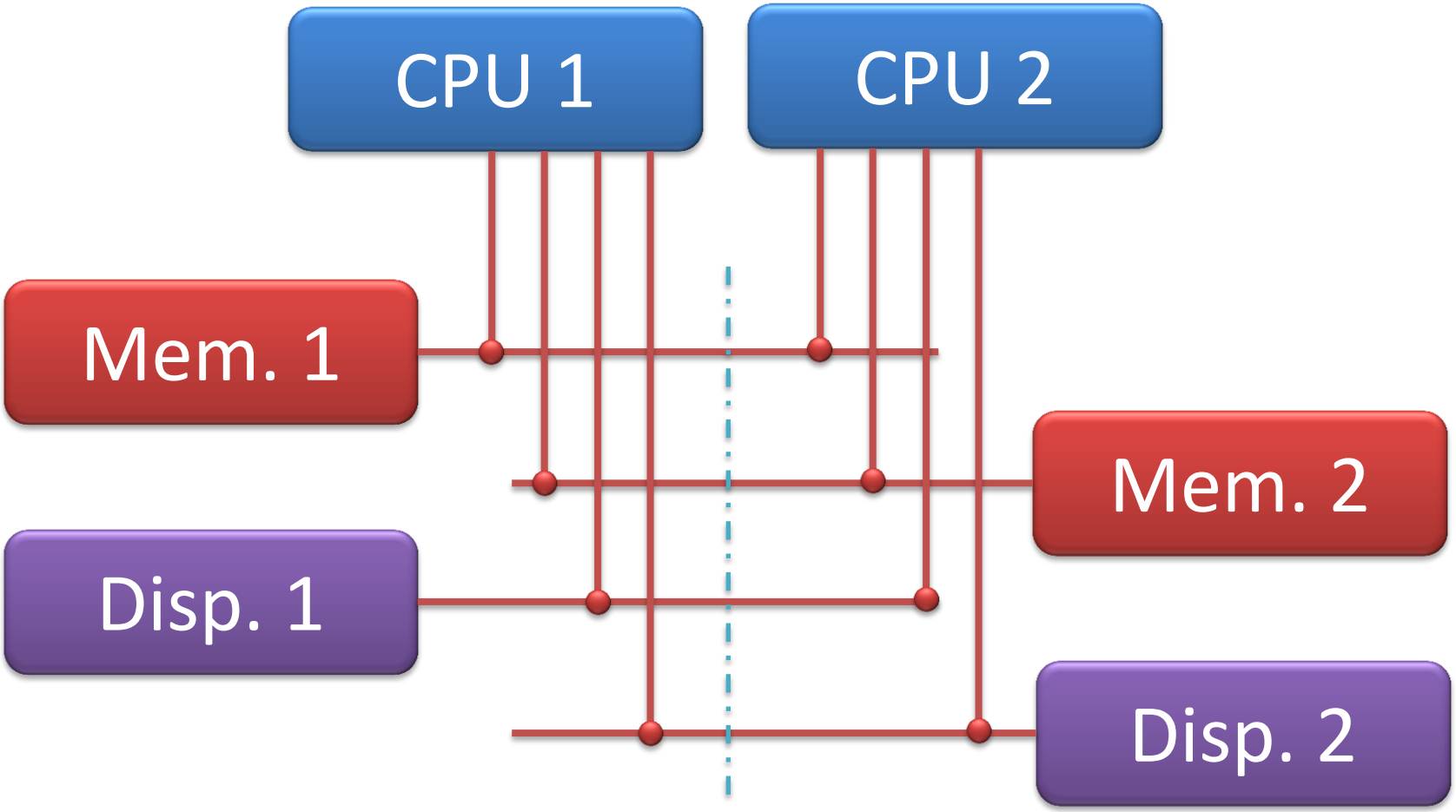


SMP: Multiport Memory

- Vantagens
  - **Redução de Conflitos** (arbitragem)
- Desvantagens
  - Memória Multiport é cara
  - Difícil manutenção de coerência de cache
  - Em alguns casos, memória ainda é gargalo
    - Solução: NUMA
      - Non-Uniform Memory Access
      - Cada CPU tem preferência a uma memória específica

NUMA

- Acesso à Memória Preferencial



EXTRA:

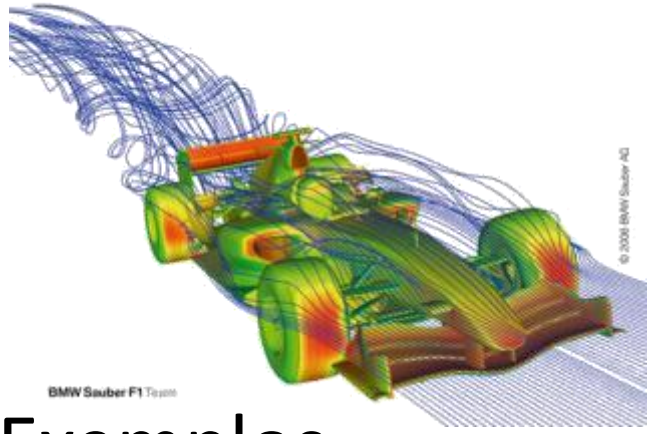
PROCESSAMENTO VETORIAL

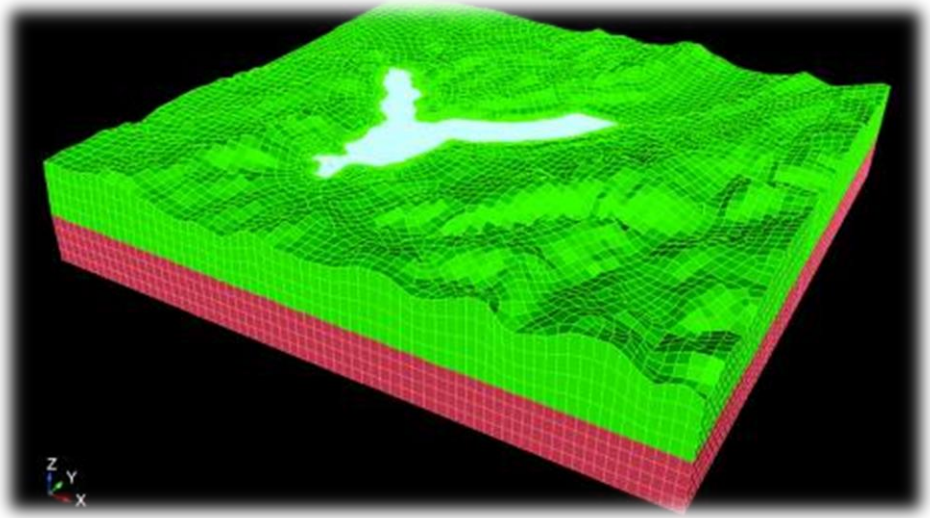
Processamento Vetorial

- Supercomputadores: SIMD
- Problemas: Aerodinâmica, meteorologia, etc.
  - Precisão numérica em ponto flutuante
  - Operações complexas, grandes vetores de números

Processamento Vetorial

- Simulação de Campos Contínuos
  - Situação física é descrita por uma superfície 3D





- Exemplos
  - Evolução climática
  - Velocidades de jato de propulsão
  - Transitórios elétricos em circuitos
  - Etc.

Processamento Vetorial

- Soma Vetorial Simples

$$\begin{bmatrix} 1,5 \\ 7,1 \\ 6,9 \\ 100,5 \\ 0 \\ 59,7 \end{bmatrix} + \begin{bmatrix} 2,0 \\ 39,7 \\ 1000,003 \\ 11 \\ 21,1 \\ 19,7 \end{bmatrix} = \begin{bmatrix} 3,5 \\ 46,8 \\ 1006,903 \\ 111,5 \\ 21,1 \\ 79,4 \end{bmatrix}$$

$$A + B = C$$

Processamento Vetorial

- Existem poucos computadores assim
  - Grandes centros de pesquisa
- Preço: alguns milhões de dólares
- Fica fácil resolver problemas complexos?
  - Um problema climático simples: 10^{15} operações
    - Se cada uma levar 1ns (1GHz), tempo total: 11,6 dias
  - Vários dias para resolver problemas medianos
  - Bastante pesquisa nessa área

ENCERRAMENTO

Resumo e Próximos Passos

- Pipelines e Superpipelines
- Superescalar
- Processamento SMP
- Processamento Vetorial
- **Pós Aula**: Saiba Mais, A Seguir... e Desafio!
  - No mural: https://padlet.com/djcaetano/arquitetura/

- O que é uma CPU RISC?
  - O que são microprogramas?

PERGUNTAS?