

A interface hardware / software

Arquitetura de Computadores III

Apresentação da disciplina

Professor Henrique Cota de Freitas

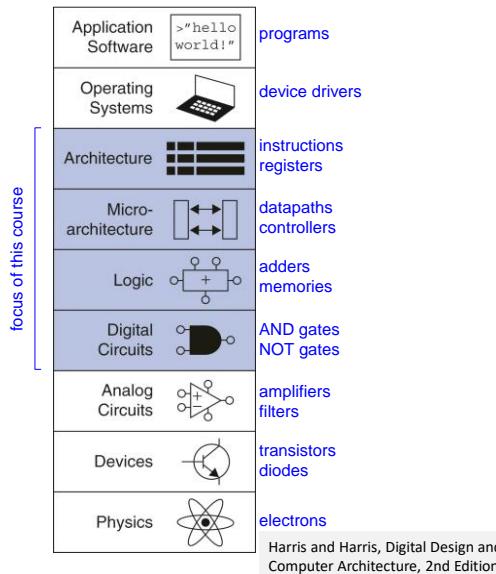
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

1

1

O que é Arquitetura de Computadores?



2024

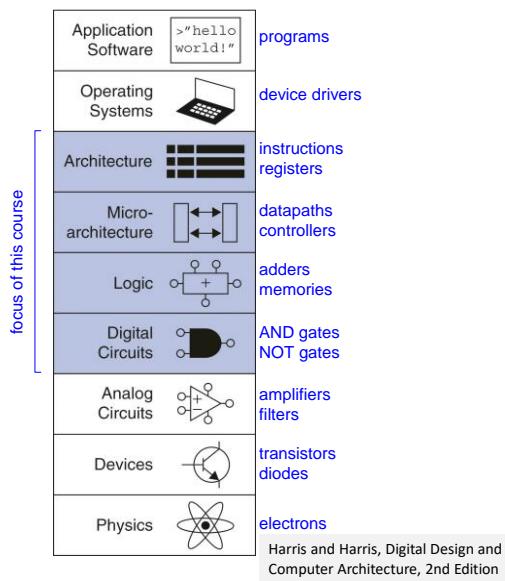
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

2

2

O que é Arquitetura de Computadores?

- Interface:
 - Software / Hardware



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

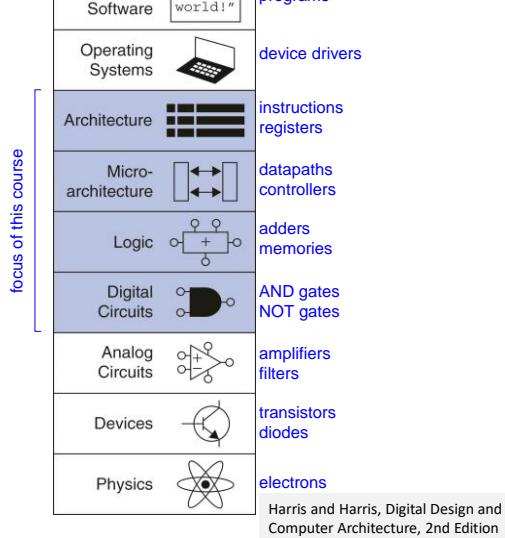
3

3

O que é Arquitetura de Computadores?

- Interface:
 - Software / Hardware

Software



2024

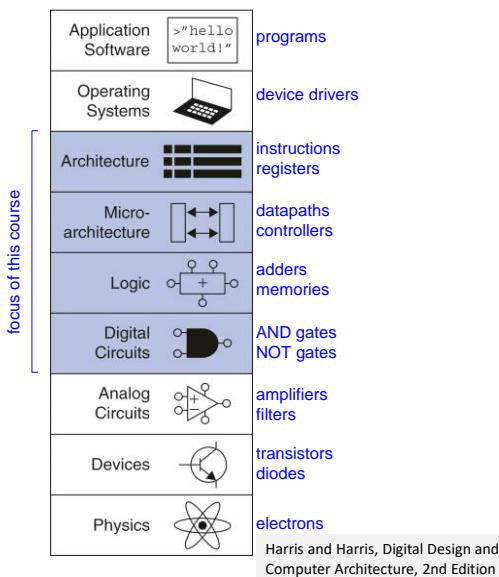
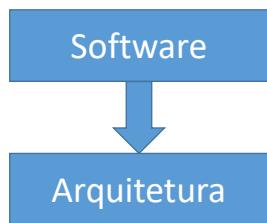
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

4

4

O que é Arquitetura de Computadores?

- Interface:
 - Software / Hardware



2024

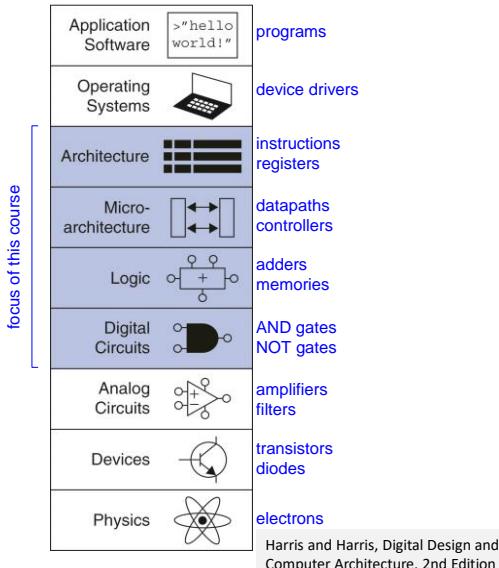
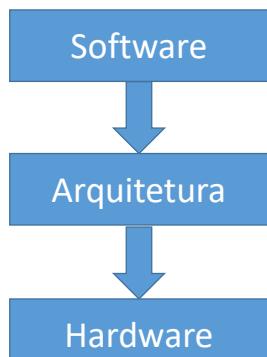
Arquitetura de Computadores III - Engenharia e Ciéncia da Computaçao - PUC Minas

5

5

O que é Arquitetura de Computadores?

- Interface:
 - Software / Hardware



2024

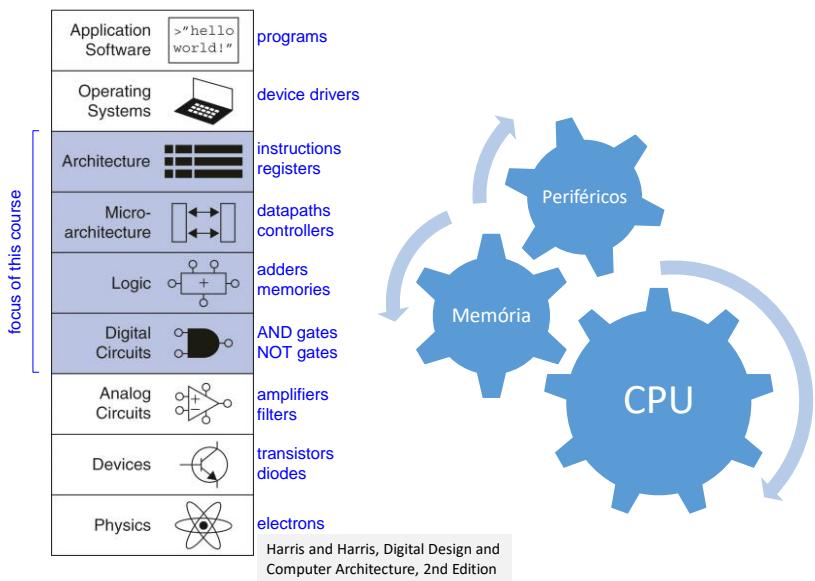
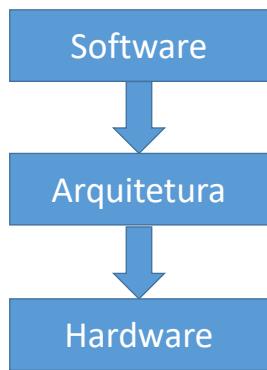
Arquitetura de Computadores III - Engenharia e Ciéncia da Computaçao - PUC Minas

6

6

O que é Arquitetura de Computadores?

- Interface:
 - Software / Hardware



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

7

7

Arquitetura de Computadores I, II e III

Ciênci/Engenharia
da Computação
PUC Minas

2º período

Arquitetura de Computadores I

3º período

Arquitetura de Computadores II

5º período

Arquitetura de Computadores III

focus of this course

Application Software	>"hello world!"	programs
Operating Systems		device drivers
Architecture		instructions registers
Micro-architecture		datapaths controllers
Logic		adders memories
Digital Circuits		AND gates NOT gates
Analog Circuits		amplifiers filters
Devices		transistors diodes
Physics		electrons

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

8

8

Arquitetura de Computadores III

- Unidades de aula:
 - Hierarquia de memória
 - Pipeline escalar de instruções (com revisão de monociclo e multiciclo)
 - Pipeline superescalar de instruções (despacho e terminação - Tomasulo)
 - Suporte multithreading (hardware multithreading)
 - Arquiteturas multicore (origem, história e exemplos)
 - Arquiteturas de processadores multi/many-core e redes-em-chip
 - Arquiteturas paralelas de memória compartilhada e distribuída, clusters e supercomputadores
 - Arquiteturas Big Data, Machine Learning, Neuromórficas e Quânticas

Bibliografia básica e complementar

- PATTERSON, David A. Organização e projeto de computadores a interface hardware/software
- HENNESSY, John L. Arquitetura de computadores uma abordagem quantitativa
- STALLINGS, William. Arquitetura e organização de computadores
- FLICH, Jose ; BERTOZZI, Davide (Ed.). Designing network on-chip architectures in the nanoscale era
- HARRIS, David Money; HARRIS, Sarah L. Digital design and computer architecture
- PACHECO, Peter S. An introduction to parallel programming
- TANENBAUM, Andrew S.; AUSTIN, Todd. Organização estruturada de computadores
- HAGER, Georg; WELLEIN, Gerhard. Introduction to high performance computing for scientists and engineers

Distribuição de pontos (aguardando confirmação da coordenação)

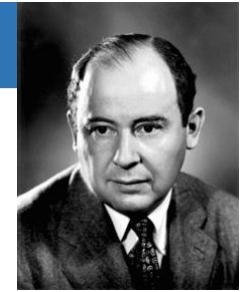
- Provas individuais: total de 55 pontos
 - 1^a prova: 25 pontos
 - 2^a prova: 30 pontos
- Avaliação de Desempenho Acadêmico (ADA): total de 5 pontos
- Trabalhos: total de 40 pontos
 - Trabalhos em grupo (dois trabalhos: 20 pts + 20 pts)
- Reavaliação: 55 pontos (todo conteúdo da disciplina)
 - A prova de reavaliação é aplicada quando o(a) aluno(a) não atinge os 60 pontos mínimos para aprovação e possui 5 ou mais pontos em trabalhos e ADA.
 - A nota da reavaliação substitui as notas das duas provas para composição da nota final com as demais notas.

A interface hardware / software

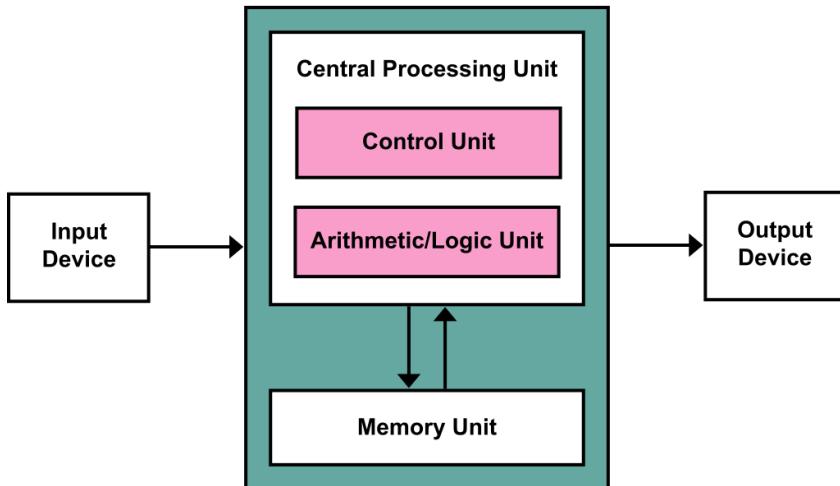
Arquitetura de Computadores III

Hierarquia de Memória

Arquitetura de von Neumann

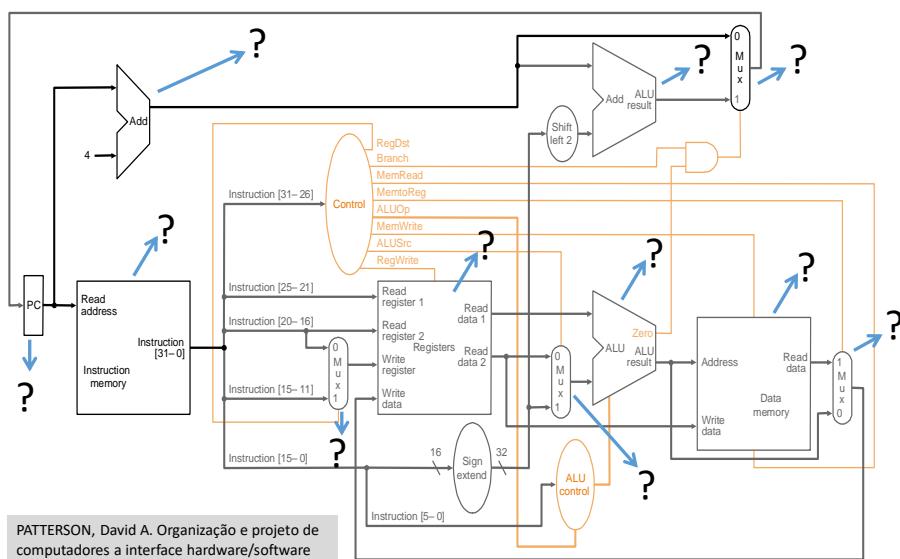


John von Neumann

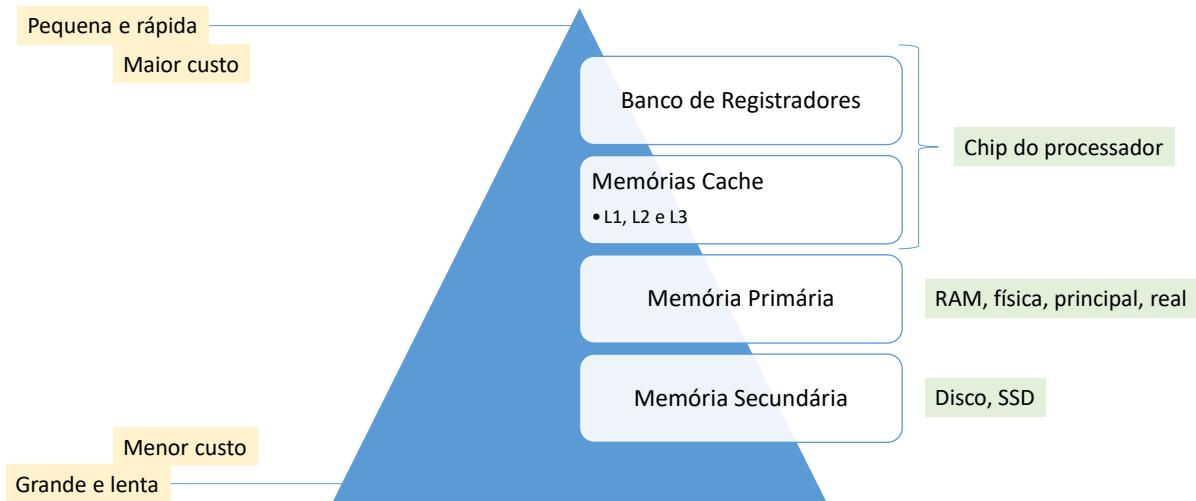


https://en.wikipedia.org/wiki/Von_Neumann_architecture

Uma versão da arquitetura do Processador MIPS



Hierarquia de Memória



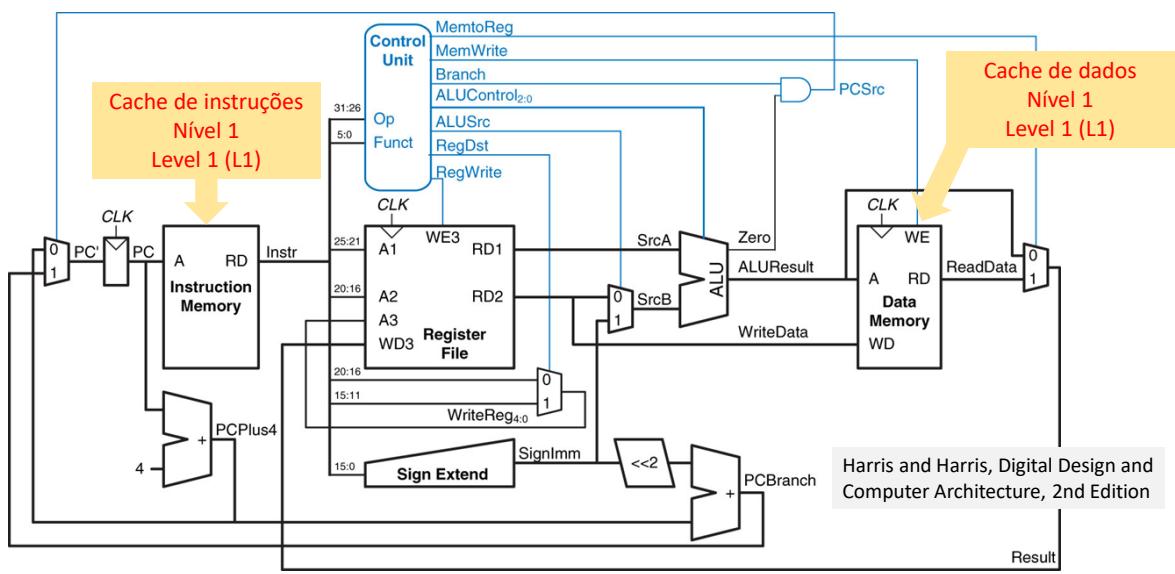
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

15

15

Onde aplicar os conceitos? (processador MIPS)



2024

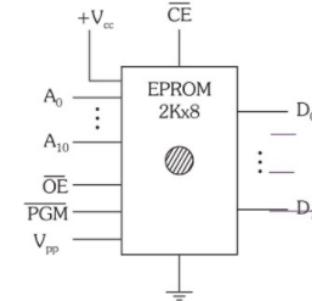
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

16

16

Memória

- RAM: Random-Access Memory
- ROM: Read-Only Memory
- PROM: Programmable ROM
 - EPROM: Apagável com radiação ultravioleta
 - EEPROM: Apagável por sinais elétricos.
- Registradores?



Editora Érica - Sistemas Digitais: Circuitos Combinacionais e Sequenciais - Francisco Gabriel Capuano - 1a Edição

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

17

17

Memória

- Célula RAM estática: A memória estática é capaz de manter os bits de dados armazenados apenas enquanto a fonte de alimentação estiver conectada ao circuito. Uma célula SRAM é equivalente ao *flip-flop*.
- Célula RAM dinâmica: A DRAM é similar a SRAM. A diferença é o projeto das células. As células dinâmicas são **mais simples** e necessitam de **menos área no chip**. Isto permite que a DRAM seja construída com densidades de armazenamento maiores, reduzindo o custo do bit. A DRAM é muito utilizada para **memórias principais** dos computadores. A desvantagem da DRAM é que as células **são mais lentas**. Os tempos de leitura e escrita são maiores. Uma célula DRAM é construída a partir de **capacitores**, demandando **refresh** de memória para manter os dados armazenados.

O que é uma palavra de dados?

- Conjunto de bits?
- Quantos bits?
- O que significa processador de 32 bits? E de 64 bits?
- Como representar a palavra de dados?
- Como armazenar uma palavra de dados?

2024

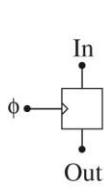
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

19

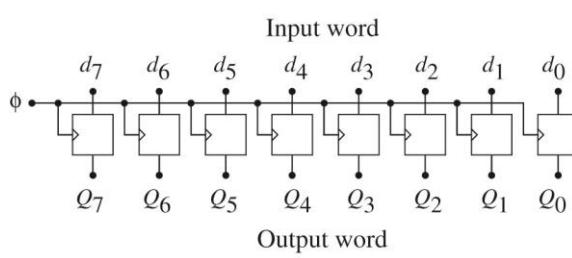
19

Registradores

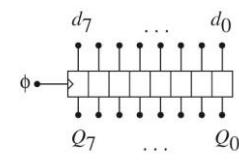
- Um registrador é um elemento lógico utilizado para armazenar uma palavra binária de n-bits.



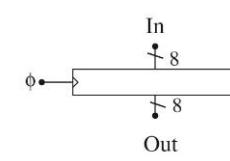
(a) Single cell



(b) 8-bit register



(a) Individual cells



(b) Single register

Uyemura, Sistemas Digitais, Uma abordagem Integrada, Thomson, 2002

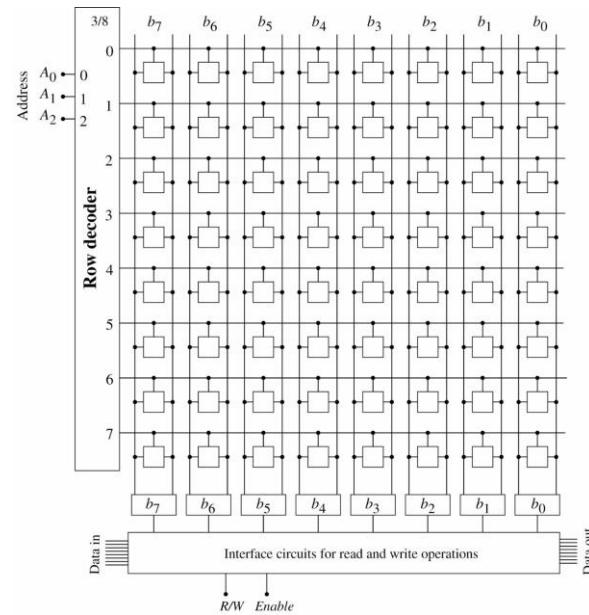
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

20

20

Memória (arranjo de SRAM – matriz 8x8)



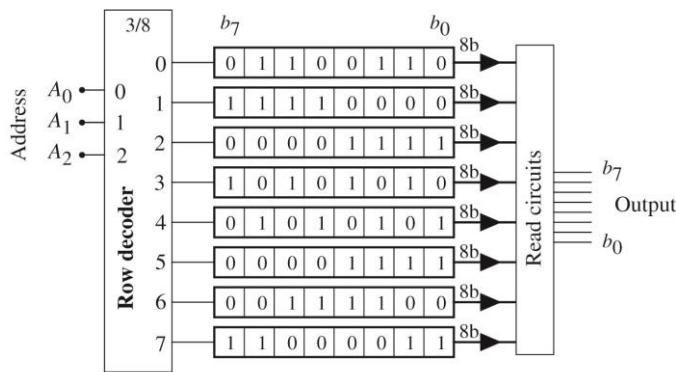
Uyemura, Sistemas Digitais, Uma abordagem Integrada, Thomson, 2002

2024

21

21

Memória (leitura em uma matriz SRAM)



(a) Simplified network

Address $A_2\ A_1\ A_0$	Row	Output $b_7 \dots b_0$
0 0 0	0	0110 0110
0 0 1	1	1111 0000
0 1 0	2	0000 1111
0 1 1	3	1010 1010
1 0 0	4	0101 0101
1 0 1	5	0000 1111
1 1 0	6	0011 1100
1 1 1	7	1100 0011

(b) Function table

Uyemura, Sistemas Digitais, Uma abordagem Integrada, Thomson, 2002

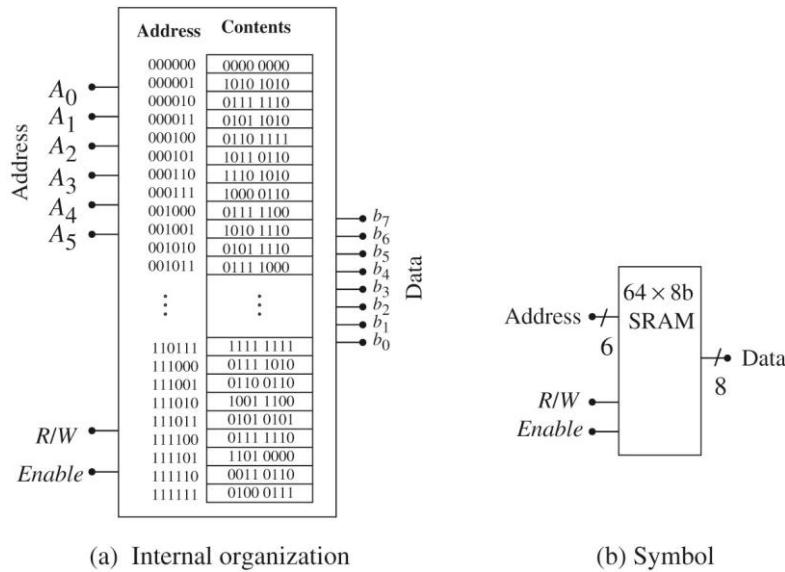
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

22

22

Memória (SRAM 64x8)



Uyemura, Sistemas Digitais, Uma abordagem Integrada, Thomson, 2002

2024

(a) Internal organization

(b) Symbol

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

23

23

Prefixos de unidades

Exp.	Explicit	Prefix	Exp.	Explicit	Prefix
10^{-3}	0.001	milli	10^3	1,000	Kilo
10^{-6}	0.000001	micro	10^6	1,000,000	Mega
10^{-9}	0.000000001	nano	10^9	1,000,000,000	Giga
10^{-12}	0.000000000001	pico	10^{12}	1,000,000,000,000	Tera
10^{-15}	0.00000000000001	femto	10^{15}	1,000,000,000,000,000	Peta
10^{-18}	0.0000000000000001	atto	10^{18}	1,000,000,000,000,000,000	Exa
10^{-21}	0.000000000000000001	zepto	10^{21}	1,000,000,000,000,000,000,000	Zetta
10^{-24}	0.00000000000000000001	yocto	10^{24}	1,000,000,000,000,000,000,000,000	Yotta

Como seria para a base binária?

Kilo =

Mega =

Giga =

Tera =

1ms, 1μs, 1ns

1000ps = 1ns

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

24

24

Prefixos de unidades

Exp.	Explicit	Prefix	Exp.	Explicit	Prefix
10^{-3}	0.001	milli	10^3	1,000	Kilo
10^{-6}	0.000001	micro	10^6	1,000,000	Mega
10^{-9}	0.000000001	nano	10^9	1,000,000,000	Giga
10^{-12}	0.000000000001	pico	10^{12}	1,000,000,000,000	Tera
10^{-15}	0.000000000000001	femto	10^{15}	1,000,000,000,000,000	Peta
10^{-18}	0.000000000000000001	atto	10^{18}	1,000,000,000,000,000,000	Exa
10^{-21}	0.000000000000000000001	zepto	10^{21}	1,000,000,000,000,000,000,000	Zetta
10^{-24}	0.0000000000000000000000000000001	yocto	10^{24}	1,000,000,000,000,000,000,000,000	Yotta

Como seria para a base binária?

$$\text{Kilo} = 2^{10}$$

$$\text{Mega} = 2^{20}$$

$$\text{Giga} = 2^{30}$$

$$\text{Tera} = 2^{40}$$

$$1\text{ms}, 1\mu\text{s}, 1\text{ns}$$

$$1000\text{ps} = 1\text{ns}$$

Prefixos de unidades

Exp.	Explicit	Prefix	Exp.	Explicit	Prefix
10^{-3}	0.001	milli	10^3	1,000	Kilo
10^{-6}	0.000001	micro	10^6	1,000,000	Mega
10^{-9}	0.000000001	nano	10^9	1,000,000,000	Giga
10^{-12}	0.000000000001	pico	10^{12}	1,000,000,000,000	Tera
10^{-15}	0.000000000000001	femto	10^{15}	1,000,000,000,000,000	Peta
10^{-18}	0.000000000000000001	atto	10^{18}	1,000,000,000,000,000,000	Exa
10^{-21}	0.000000000000000000001	zepto	10^{21}	1,000,000,000,000,000,000,000	Zetta
10^{-24}	0.0000000000000000000000000000001	yocto	10^{24}	1,000,000,000,000,000,000,000,000	Yotta

Como seria para a base binária?

$$\text{Kilo} = 2^{10}$$

$$\text{Mega} = 2^{20}$$

$$\text{Giga} = 2^{30}$$

$$\text{Tera} = 2^{40}$$

$$1\text{ms}, 1\mu\text{s}, 1\text{ns}$$

$$1000\text{ps} = 1\text{ns}$$

$$1\text{kb}, 1\text{Mb}, 1\text{Gb}$$

$$1\text{kbps}, 1\text{kb/s}$$

$$128\text{Mb/s}$$

Prefixos de unidades

Exp.	Explicit	Prefix	Exp.	Explicit	Prefix
10^{-3}	0.001	milli	10^3	1,000	Kilo
10^{-6}	0.000001	micro	10^6	1,000,000	Mega
10^{-9}	0.000000001	nano	10^9	1,000,000,000	Giga
10^{-12}	0.000000000001	pico	10^{12}	1,000,000,000,000	Tera
10^{-15}	0.000000000000001	femto	10^{15}	1,000,000,000,000,000	Peta
10^{-18}	0.0000000000000001	atto	10^{18}	1,000,000,000,000,000,000	Exa
10^{-21}	0.0000000000000001	zepto	10^{21}	1,000,000,000,000,000,000,000	Zetta
10^{-24}	0.0000000000000001	yocto	10^{24}	1,000,000,000,000,000,000,000,000	Yotta

Como seria para a base binária?

$$\text{Kilo} = 2^{10}$$

$$\text{Mega} = 2^{20}$$

$$\text{Giga} = 2^{30}$$

$$\text{Tera} = 2^{40}$$

$$1 \text{ kb} / 1 \text{ kbps} = ?$$

$$1 \times 1024 / 1 \times 1000 = 1,024 \text{ s}$$

Prefixos de unidades

Múltiplos de bits			[Esconder]		
Prefixo do SI			Prefixo binário		
Nome	Símbolo	Múltiplo	Nome	Símbolo	Múltiplo
bit	b	10^0	bit	b	2^0
quilobit	kb	10^3	kibibit	Kib	2^{10}
megabit	Mb	10^6	mebibit	Mib	2^{20}
gigabit	Gb	10^9	gibibit	Gib	2^{30}
terabit	Tb	10^{12}	tebibit	Tib	2^{40}
petabit	Pb	10^{15}	pebibit	Pib	2^{50}
exabit	Eb	10^{18}	exbibit	Eib	2^{60}
zettabit	Zb	10^{21}	zebibit	Zib	2^{70}
yottabit	Yb	10^{24}	yobibit	Yib	2^{80}

Forma mais comum de contagem de múltiplos de bits ainda é pelos prefixos da SI, pois os dados são tratados pontualmente, ou seja, ou é 0 ou é 1.

<https://pt.wikipedia.org/wiki/Kibibit>

- Há muita discussão sobre o uso dos prefixos. O que deveria ser um padrão, ou seja, a base do prefixo é determinada pela unidade, não é levado em consideração por muitos. Por isso, há quem diga:

- k, M, G, etc. são sempre para bases decimais 😐
- Ki, Mi, Gi, etc. são sempre para bases binárias 😐

- k, M, G, etc. possuem bases dependentes das unidades, afinal, são prefixos de unidades



Prefixos de unidades

- A discussão se k é minúsculo ou maiúsculo, se é base decimal ou binária, seria facilmente resolvida se as **atenções estivessem apenas nas unidades**, mantendo por padrão o **k minúsculo**. Exemplo:
 - 1km: unidade metro, base decimal, $k = 10^3$
 - 1kb: unidade bit, base binária, $k = 2^{10}$
- K maiúsculo na tentativa de diferenciar é desnecessário, porque a base é determinada pela unidade e não pelo K ou k. K maiúsculo é a unidade Kelvin (temperatura).
- A proposta do Ki, Mi, etc., apenas acrescenta outra forma de representar o que já possui representação. Por isso, há software, SO, artigos e livros que usam as três formas (kb, Kb e Kib), o que aumenta a atenção que deve ser dada.
 - O que deveria ser um padrão, k minúsculo e base determinada pela unidade, foi “ignorada” ao longo dos anos aumentando a confusão sobre o tema.

Palavra de dados

- 1 byte (1B) é o conjunto de 8 bits (8b).
- Uma palavra de 8 bits = 1 byte.

Tamanho da palavra	Número de valores	Abreviação para valor
8b	$2^8 = 256$	
10b	$2^{10} = 1024$	1kb
16b	$2^{16} = 65\,536$	64kb
20b	$2^{20} = 1\,048\,576$	1Mb
28b	$2^{28} = 268\,435\,456$	256Mb
30b	$2^{30} = 1\,073\,741\,820$	1Gb

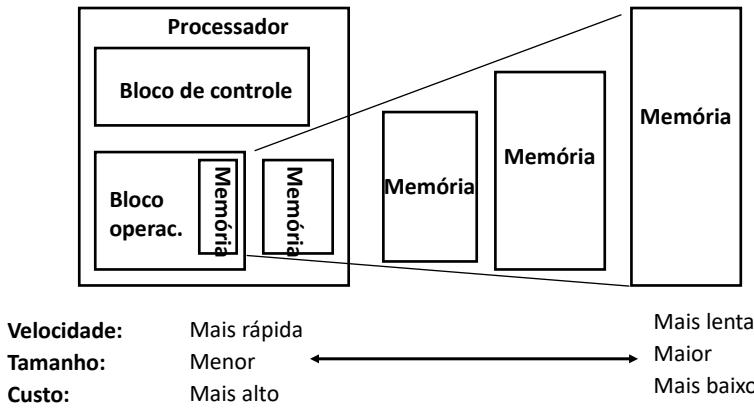
Palavra de dados

- Processador com uma palavra de 16b enxerga 64k endereços.
 - Se cada endereço armazena 16 bits (2 bytes) de dados, a memória é de 128kB.
- Processador com uma palavra de 32b enxerga quantos endereços?
 - $2^{32} = 2^2 \times 2^{30} = 4 \times G = 4\text{Giga}$ endereços.
 - Se cada endereço armazena 1 byte, a memória é de 4GB.
- Processador com uma palavra de 64b enxerga quantos endereços?
 - $2^{64} = 2^4 \times 2^{60} = 16 \times E = 16\text{Exa}$ endereços.
 - Se cada endereço armazena 1 byte, a memória é de 16EB.

Hierarquia de Memória

- Objetivo: oferecer ilusão de máximo tamanho de memória, com mínimo custo.
- Máxima velocidade.
- Cada nível contém cópia de parte da informação armazenada no nível superior seguinte.

Hierarquia de Memória



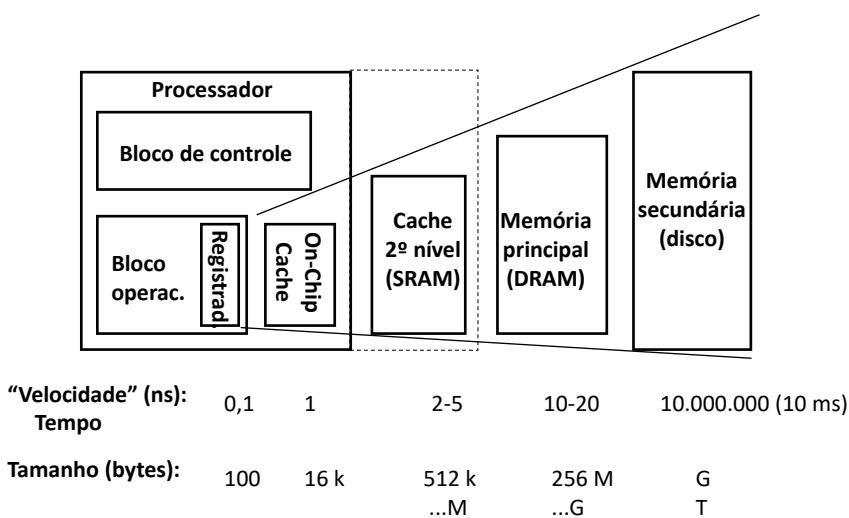
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

33

33

Hierarquia de Memória



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

34

34

Hierarquia de Memória

- Como a hierarquia é gerenciada?
- Registradores <-> memória
 - pelo compilador
- cache <-> memória principal
 - pelo hardware
- memória principal <-> disco
 - pelo hardware e pelo sistema operacional (memória virtual)
 - pelo programador (arquivos)

Hierarquia de Memória

Princípio da Localidade

- Espacial: se um dado é referenciado, seus vizinhos tendem a ser referenciados logo.
- Temporal: um dado referenciado, tende a ser referenciado novamente.

Como explorar o princípio de localidade numa hierarquia de memória?

- Localidade Temporal
 - => Mantenha itens de dados mais recentemente acessados nos níveis da hierarquia mais próximos do processador
- Localidade Espacial
 - => Mova blocos de palavras contíguas para os níveis da hierarquia mais próximos do processador

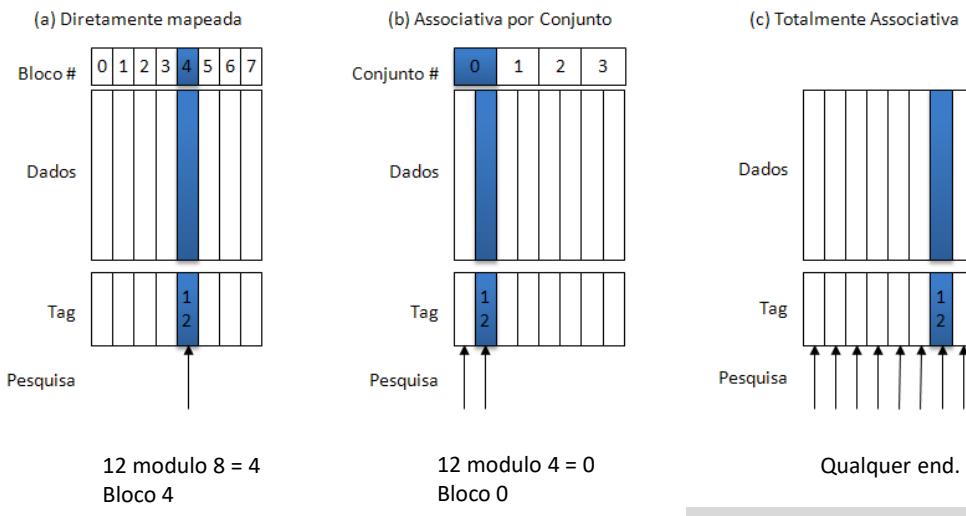
Organizações de Memória Cache

- Processador gera endereço de memória e o envia à cache
- Cache deve
 - verificar se tem cópia da posição de memória correspondente
 - se tem, encontrar a posição da cache onde está esta cópia
 - se não tem, trazer o conteúdo da memória principal e escolher posição da cache onde a cópia será armazenada
- Mapeamento entre endereços de memória principal e endereços de cache resolve estas 3 questões
 - deve ser executado em hardware
- Estratégias de organização (mapeamento) da cache
 - mapeamento completamente associativo
 - mapeamento direto
 - mapeamento set-associativo

Memórias Cache

- Características
 - Pouco espaço de armazenamento
 - Alto custo financeiro
 - Baixo tempo de acesso
- Conceitos
 - Palavra: conjunto de um ou mais bytes.
 - Bloco: conjunto de uma ou mais palavras (unidade da cache)
 - Bit de Válido: indica se o dado ou bloco está válido
 - Tag ou rótulo: parte do endereço de uma palavra na memória principal
 - Slot: cada linha de uma cache, que pode armazenar um ou mais blocos dependendo da organização da cache.
 - Comparador: compara a tag de um endereço de uma palavra, com as tags dos endereços armazenados na cache

Modos de Mapeamento



2024

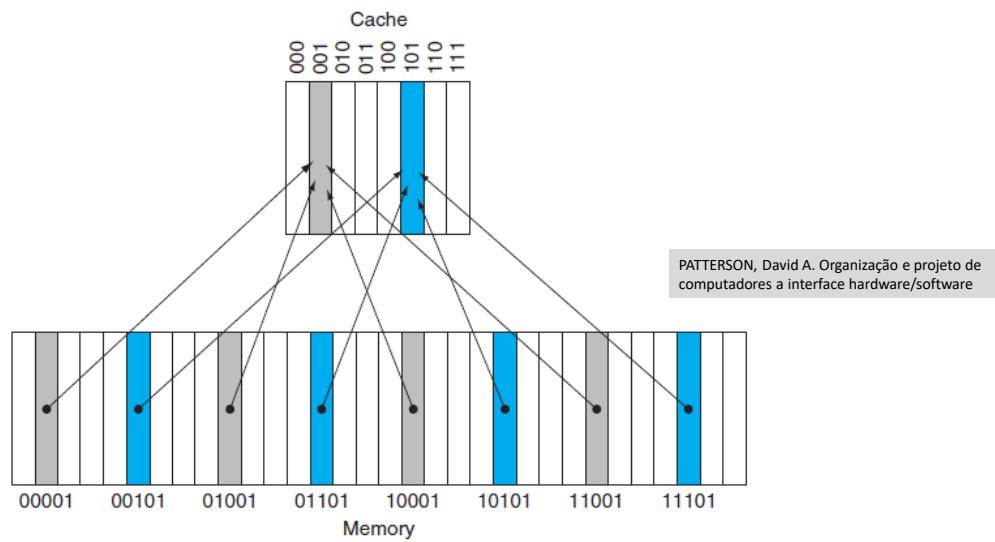
Arquitetura de Computadores III - Engenharia e Ciência da Computação

PATTERSON, David A. Organização e projeto de computadores a interface hardware/software

39

39

Mapeamento Direto



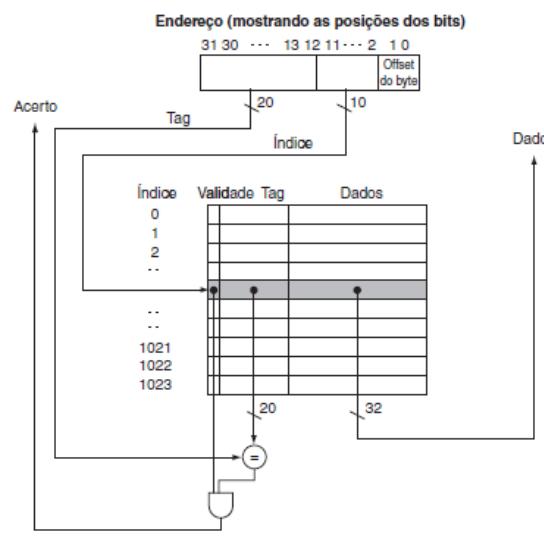
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

40

40

Tamanho da linha



PATTERSON, David A. Organização e projeto de computadores a interface hardware/software

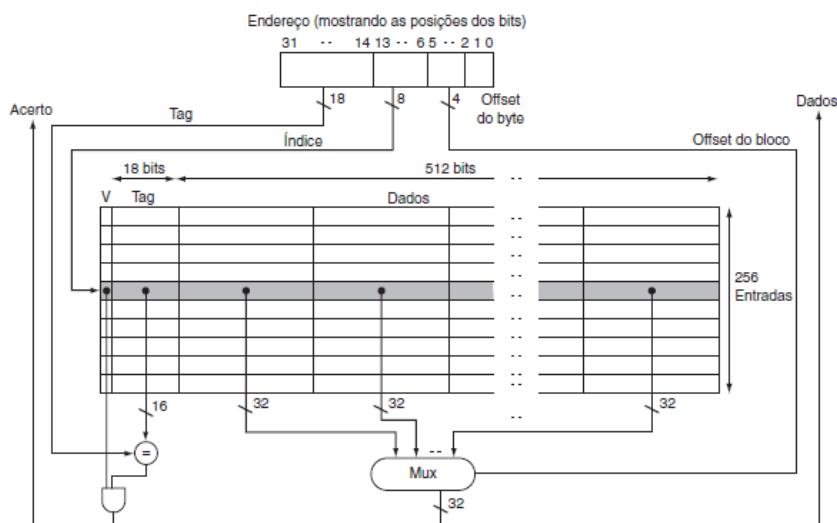
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

41

41

Tamanho da linha tirando vantagem da localidade espacial



PATTERSON, David A. Organização e projeto de computadores a interface hardware/software

2024

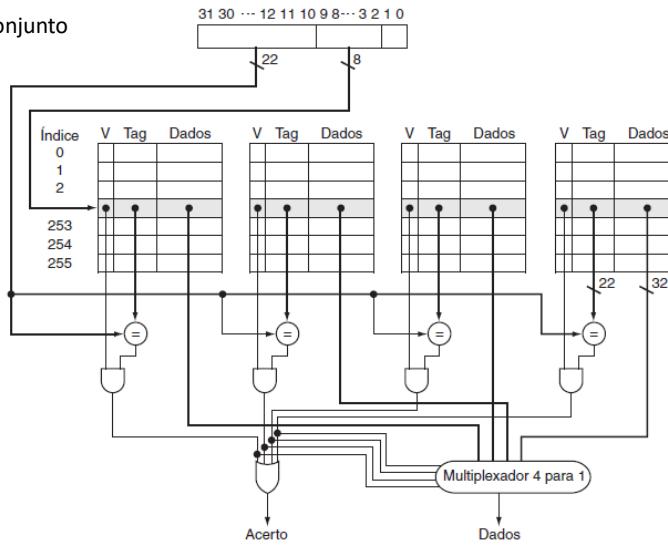
Arquitetura de Computadores III - Engenharia e Ciência da Computação

42

42

Organizações da Cache

Associativa por conjunto
4 vias



256 conjuntos
1 possui 4 vias
 $1 \text{ via} = 1 + 22 + 32 = 55 \text{ bits}$

256 conjuntos x 4 = 1024 linhas

$$55 \times 1024 = 55 \text{ kb}$$

PATTERSON, David A. Organização e projeto de computadores a interface hardware/software

2024

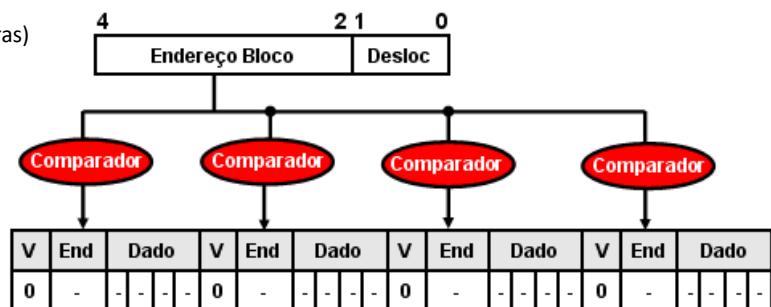
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

43

43

Organizações da Cache

- **Completamente Associativa**
 - Possui um slot com N blocos
 - Necessita de N comparações
 - Endereça o bloco diretamente
 - Exemplo:
 - Endereços de 5 bits (32 palavras)
 - 1 slot (com 4 blocos)
 - Blocos de 4 palavras



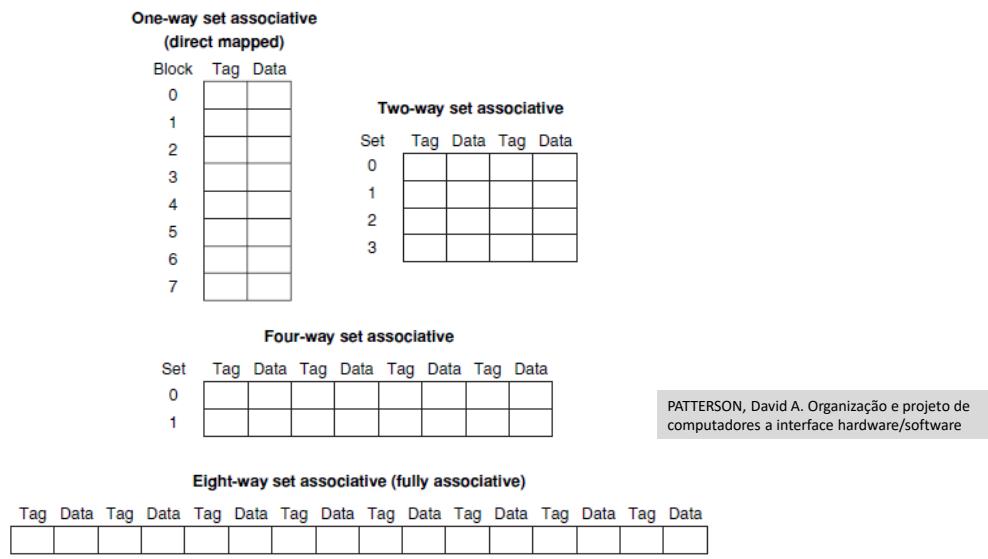
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

44

44

Caches com o mesmo tamanho de dados



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

45

45

Acesso à Cache

- Quando a cache estiver sem espaço, qual bloco será substituído?
 - Mapeamento Direto: o bloco que estiver no slot
 - Associativa por conjunto e Completamente Associativa: usar uma política de substituição
 - LRU: substituir o bloco menos recentemente utilizado
 - Item (endereço de linha) vai para a frente da lista
 - LFU: substituir o bloco menos frequentemente utilizado
 - Contador incrementado quando bloco é acessado
 - FIFO: substituir o primeiro bloco que entrou na cache
 - Aleatório: escolher um bloco qualquer

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

46

46

Mapeamento Direto

0000 0 miss	0001 1 miss	0010 2 miss	0011 3 miss
00 Mem(0)	00 Mem(0)	00 Mem(0)	00 Mem(0)
	00 Mem(1)	00 Mem(1)	00 Mem(1)
		00 Mem(2)	00 Mem(2)
			00 Mem(3)
0100 4 miss	0011 3 hit	0100 4 hit	1111 15 miss
01 00 Mem(0) 4	01 Mem(4)	01 Mem(4)	01 Mem(4)
00 Mem(1)	00 Mem(1)	00 Mem(1)	00 Mem(1)
00 Mem(2)	00 Mem(2)	00 Mem(2)	00 Mem(2)
00 Mem(3)	00 Mem(3)	00 Mem(3)	11 00 Mem(3) 15

8 requests, 2 hits

Completamente Associativo

000 0	010 2	001 1	011 3
000 Mem(0)	000 Mem(0)	000 Mem(0)	000 Mem(0)
	010 Mem(2)	010 Mem(2)	010 Mem(2)
		001 Mem(1)	001 Mem(1)
			011 Mem(3)
100 4 4	011 3	100 4	111 15
100 000 Mem(0) 4	100 Mem(4)	100 Mem(4)	111 100 Mem(4) 15
010 Mem(2)	010 Mem(2)	010 Mem(2)	010 Mem(2)
001 Mem(1)	001 Mem(1)	001 Mem(1)	001 Mem(1)
011 Mem(3)	011 Mem(3)	011 Mem(3)	011 Mem(3)

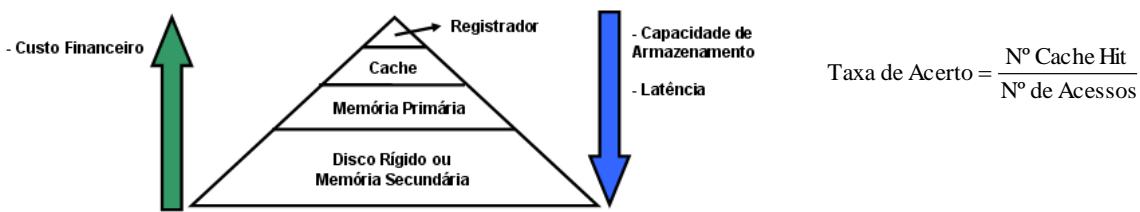
8 requests, 2 hits

Acesso à Cache

- Quando ocorrer uma escrita, como manter a coerência com a memória principal?
 - Write-through: a palavra é escrita tanto no bloco da cache, quanto no bloco da memória principal.
 - Write-back: a palavra é escrita somente no bloco da cache. Quando este bloco for substituído, então a palavra será escrita na memória principal.

Cache hit e Cache miss

- Cache Hit: acerto na cache, ou seja, o dado procurado já se encontra carregado na cache
 - **Hit Time:** tempo de acesso ao nível **superior**, que consiste de tempo de acesso + tempo para determinar hit/miss
- Cache Miss: falta na cache, ou seja, o dado procurado ainda tem que ser buscado na memória principal.
 - **Miss Penalty:** tempo gasto para substituir um bloco no nível **superior** + tempo para fornecer o bloco ao processador
- Métrica
 - Taxa de acerto: dado um número de acessos a cache, qual a porcentagem de cache hit.



Tipos de cache miss

- **compulsórios** (cold start ou chaveamento de processos, primeira referência): primeiro acesso a uma linha
 - é um “fato da vida”: não se pode fazer muito a respeito
 - se o programa vai executar “bilhões” de instruções, misses compulsórios são insignificantes
- de **conflito** (ou colisão)
 - múltiplas linhas de memória acessando o mesmo conjunto da cache conjunto-associativa ou mesma linha da cache com mapeamento direto
 - solução 1: aumentar tamanho da cache
 - solução 2: aumentar associatividade
- de **capacidade**
 - cache não pode conter todas as linhas accessadas pelo programa
 - solução: aumentar tamanho da cache
- **invalidação**: outro processo (p.ex. I/O) atualiza memória

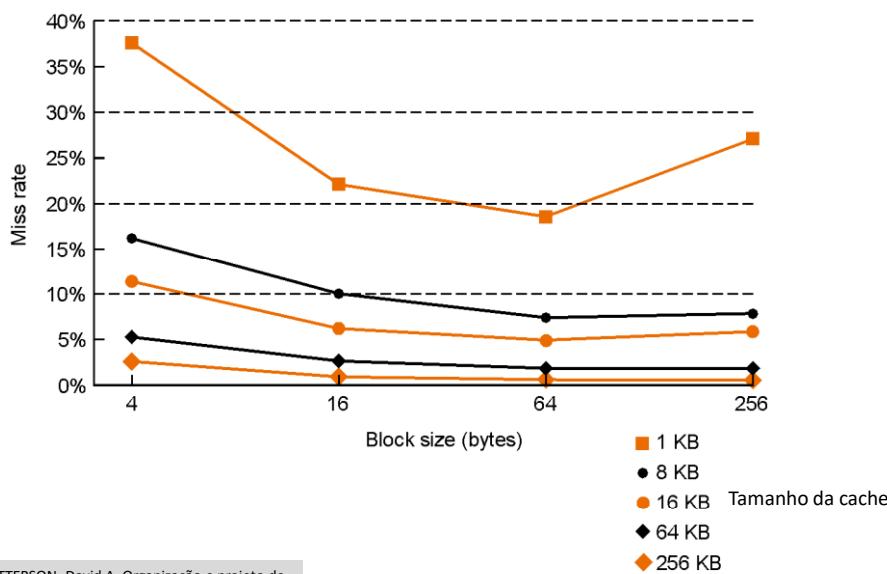
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

51

51

Tamanho da linha vs. miss ratio



2024

PATTERSON, David A. Organização e projeto de computadores a interface hardware/software

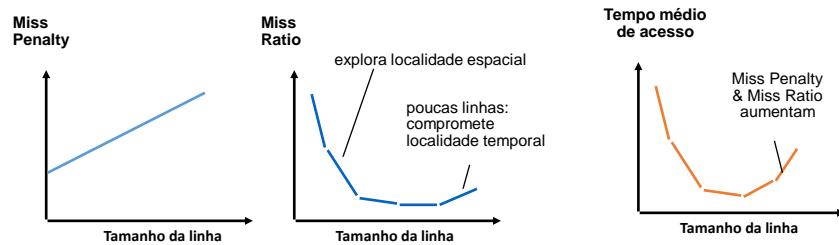
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

52

52

Tamanho da linha

- em geral, uma linha maior aproveita melhor a localidade espacial **MAS**
 - linha maior significa maior *miss penalty*
 - demora mais tempo para preencher a linha
 - se tamanho da linha é grande demais em relação ao tamanho da cache, *miss ratio* vai aumentar
 - muito poucas linhas
- em geral, tempo médio de acesso = Hit Time x (1 - Miss Ratio) + Miss Penalty x Miss Ratio



Quantos bits tem a cache no total?

- supondo cache com mapeamento direto, com 64 kB de dados, linha com uma palavra de 32 bits (4 bytes), e endereços de 32 bits
- $64\text{ kB} \rightarrow 16\text{ kpalavras}$, 2^{14} palavras, neste caso 2^{14} linhas
- cada linha tem **32** bits de dados mais um tag (**32-14-2** bits) mais um bit de validade:

$$2^{14} \times (32 + 32 - 14 - 2 + 1) = 2^{14} \times 49 = 784 \times 2^{10} = 784\text{ kbits}$$
- 98 kB para 64 kB de dados, ou 50% a mais

Índice e offset

Quantos bits tem a cache no total?

- supondo cache com mapeamento direto, com 16 kB de dados, blocos de 4 palavras, sendo cada palavra de 32 bits e endereços de 32 bits
- 16 kB -> 4 kpalavras, 2^{12} palavras
- Bloco de 4 palavras (2^2), 2^{10} blocos (linhas)
- cada bloco tem 32 bits x 4 = 128 bits de dados mais um tag (32-10-2-2 bits) mais um bit de validade:

$$2^{10} \times (128 + 32 - 10 - 2 - 2 + 1) = 2^{10} \times 147 = 147 \text{ kbits}$$
- 18.4 kB para 16 kB de dados, ou 15% a mais

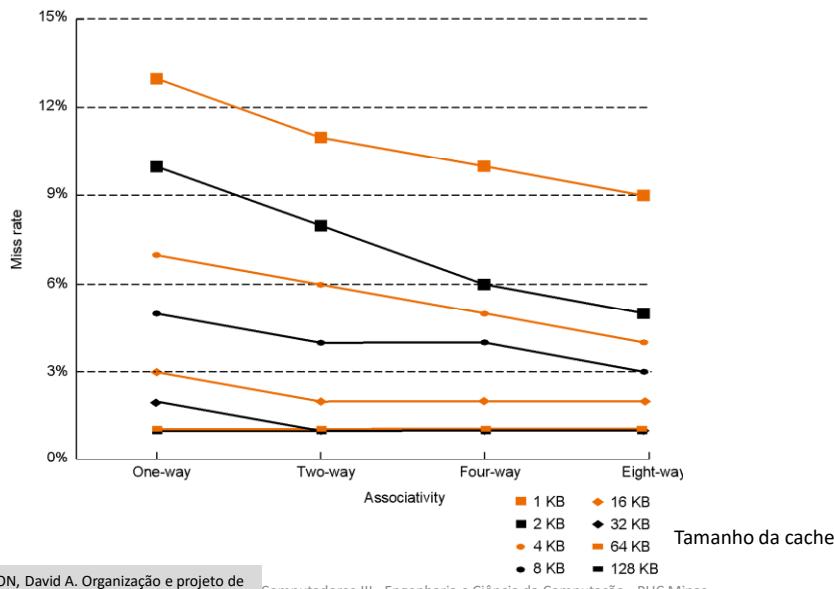
Exercício 5 da lista

- 1024kB
- 64 bits = 8 Bytes
- Qtd palavras = $1024 \text{ k} / 8\text{byte} = 128\text{k}$ palavras
- Qtd linhas/blocos = $128 \text{ k} / 8\text{palavras} = 16\text{k}$ linhas
- 2 vias
- Qtd conjuntos = $16\text{k} / 2\text{vias} = 8\text{k}$ conjuntos = $2^3 \times 2^{10} = 2^{13}$
- Tamanho do índice em bits = 13
- Endereço = 64 bits; [TAG | Índice | Off Bloco | Off byte]
- 8 palavras dentro do bloco = 3 bits de offset bloco
- 8 bytes é o tamanho da palavra = 3 bits de offset byte
- Endereço 64bits; $64-13-3-3 = 45$ bits de TAG
- Tamanho da cache = $2^{13} \times (1+45+8\times64)\times2 = \dots 558\times16\times 2^{10} = 8928\text{kbits}$

V TAG linha/bloco	V TAG linha/bloco
1 45 8x64	1 45 8x64

 2^{13}

Impacto da associatividade



2024 PATTERSON, David A. Organização e projeto de computadores a interface hardware/software

Computadores III - Engenharia e Ciência da Computação - PUC Minas

57

57

Impacto no desempenho

- Medindo o impacto do hit ratio no tempo efetivo de acesso

Tc = tempo de acesso à memória cache

Tm = tempo de acesso à memória principal

Tce = tempo efetivo de acesso à memória cache, considerando efeito dos misses

$$Tce = h Tc + (1 - h) Tm$$

$$\text{se } Tc = 1 \text{ ns}, Tm = 20 \text{ ns}$$

$$h = 0.85 \quad 0.95 \quad 0.99 \quad 1.0$$



$$\text{então } Tce = 3.85 \text{ ns} \quad 1.95 \text{ ns} \quad 1.19 \text{ ns} \quad 1 \text{ ns}$$

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

58

58

Impacto no desempenho

- Supondo um processador que executa um programa com:
 - CPI = 1.1
 - 50% aritm/lógica, 30% load/store, 20% desvios
- Supondo que 10% das operações de acesso a dados na memória sejam misses. Cada miss resulta numa penalidade de 50 ciclos.

$$\begin{aligned} \text{CPI} &= \text{CPI ideal} + \text{nº médio de stalls por instrução} \\ &= 1.1 \text{ ciclos} + 0.30 \times 0.10 \times 50 \\ &= 1.1 \text{ ciclos} + 1.5 \text{ ciclos} = 2.6 \end{aligned}$$

CPI ideal	1.1
Data misses	1.5
Instr.misses	0.5

- 58 % do tempo o processador está parado esperando pela memória!

- um miss ratio de 1% no fetch de instruções resultaria na adição de 0.5 ciclos ao CPI médio

2024 Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

59

59

Hierarquia de caches

- caches integradas dentro de um processador têm limitação de tamanho
- miss penalty na cache é muito grande, pela diferença entre os tempos de acesso da cache e da memória principal
- solução: caches em dois ou três níveis
 - cache integrada (L1, de primeiro nível) é de tamanho pequeno, p.ex. 8 kbytes, e tempo de acesso menor
 - cache secundária (L2) tem tamanho maior, p.ex. 256 kbytes, e tempo de acesso maior
- cache de terceiro nível (L3)
 - cache L3 (fora) dentro do chip do processador, cache L2 dentro
- misses podem ocorrer em referências a qualquer nível de cache
- transferências entre níveis de cache apresentam mesmos problemas e possíveis soluções já discutidos

2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

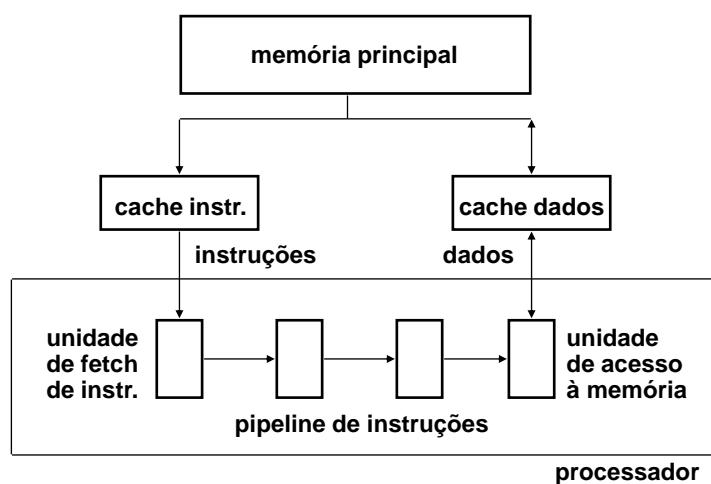
60

60

Caches de dados e instruções

- dados e instruções: cache unificada x caches separadas
- vantagens das caches separadas
 - política de escrita só precisa ser aplicada à cache de dados
 - caminhos separados entre memória principal e cada cache, permitindo transferências simultâneas (p.ex. num pipeline)
 - estratégias diferentes para cada cache: tamanho total, tamanho de linha, organização
- caches separadas são usadas na maioria dos processadores, no nível L1
- caches unificadas nos níveis L2 e L3

Caches de dados e instruções



Desempenho em caches multinível

- Suponha que o processador tenha um CPI de 1,0 e que todas as referencias acertem na cache primária a uma velocidade de clock de 5GHz (0,2ns). O tempo de acesso à memória principal é de 100ns com todos os tratamentos de faltas. Taxa de falhas por instrução na cache primária é de 2%.
- O quanto mais rápido será o processador se acrescentarmos uma cache secundária com tempo de acesso de 5ns para um acerto ou uma falha e que seja grande o suficiente para que a taxa de falhas na L2 seja de 0,5%?

Desempenho em caches multinível

- Penalidade de falha para memória principal:
 - $100\text{ns} = 500 \text{ ciclos de clock}$.
 - $0,2\text{ns/ciclo de clock}$
- Para processador com apenas L1:
 - $\text{CPI total} = 1,0 + \text{ciclos de stall de memória por instrução} = 1,0 + 2\% \times 500 = 11,0$
- Em relação a L1, penalidade de falha para L2:
 - $5\text{ns} / 0,2\text{ns} = 25 \text{ ciclos de clock}$
- Para cache de dois níveis:
 - $\text{CPI total} = 1 + \text{stall L1} + \text{stall L2} = 1 + 2\% \times 25 + 0,5\% \times 500 = 1 + 0,5 + 2,5 = 4,0$
- Portanto, com cache L2:
 - $11,0 / 4,0 = 2,8 \text{ vezes mais rápido}$

Hierarquia de memória

Memória Virtual

Uma técnica

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

65

65

Introdução

- memória principal semicondutora
 - capacidade limitada
 - tempo de acesso entre 10 e 20 ns
- memória secundária em disco
 - Capacidade limitada (obviamente) mas muito maior
 - tempo de latência entre 10 e 30 ms

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

66

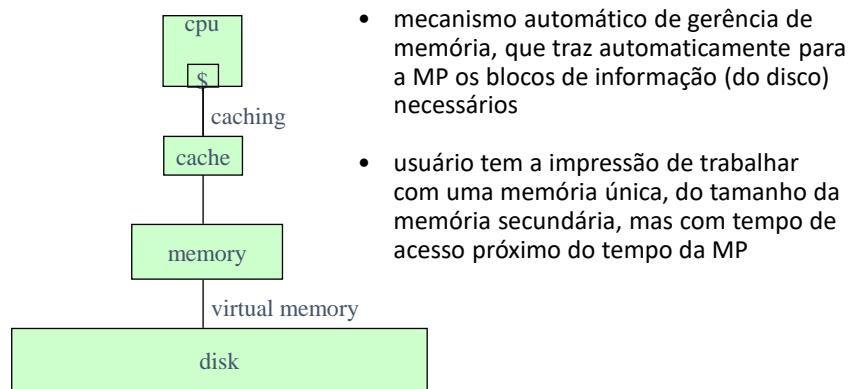
66

O problema

- Um computador tem 32 kbytes de memória principal
- Como podemos:
 - rodar programas que usam mais do que 32 kbytes?
 - permitir que vários usuários usem o computador?
 - executar vários programas ao mesmo tempo?
- O problema é independente do tamanho da memória. Pode trocar para 32 MB ou 32 GB, que o problema permanece. A memória é dimensionada para atender a carga de trabalho típica do sistema.

Memória virtual: a solução

- *Memória Virtual* : técnica que nos permite ver a memória principal como uma cache de grande capacidade de armazenamento
- É apenas mais um nível na hierarquia de memórias



Tempo de acesso

Tempo médio de acesso T_{ma} é dado por

$$T_{ma} = T_m + (1 - h) T_s$$

onde T_m = tempo de acesso à MP
 T_s = tempo de acesso ao disco
 h = hit ratio

p.ex. se $T_m = 20 \text{ ns}$, $T_s = 20 \text{ ms}$, $h = 0.9999$
então $T_{ma} = 2,02 \mu\text{s}$ (100 x maior do que T_m)

Por que MV é diferente das caches?

- Miss penalty é MUITO maior (milhões de ciclos)! Se informação não está na memória, está no disco!
- Logo:
 - miss ratio precisa ser bem menor do que em cache
 - alta penalidade do miss => necessário buscar blocos maiores em disco
 - princípio de localidade opera sobre blocos maiores de dados ou instruções e leva a hit ratios bem mais elevados
 - Mapeamento totalmente associativo das páginas
 - misses são tratados por software (há tempo disponível)
 - técnica de escrita write-through não é uma opção. Usa-se write-back.

Terminologia

- mesma idéia da cache, mas com terminologia diferente

cache	MV
bloco	página
cache miss	<i>page fault</i>
endereço	endereço virtual (ou lógico)
índice	endereço real (ou físico)

- endereço *virtual (lógico)*: gerado pelo programa
 - deve endereçar todo espaço em disco
 - maior número de bits
- endereço *real (físico)*: endereço na memória principal
 - menor número de bits

Unidade de Gerenciamento de Memória

- MMU (Memory Management Unit)
 - gerência da hierarquia de memória
 - proteção de memória
 - usualmente integrada dentro do microprocessador
- MMU deve fazer mapeamento do endereço virtual para endereço real
- SO usa a MMU

Paginação (gerenciamento de memória)

- Por que paginação? Resposta: mecanismo simples para tradução de endereços virtuais em reais e para gerenciamento do espaço de memória
- espaços de memória real e virtual divididos em blocos chamados de páginas
 - páginas tem tipicamente de 4 kbytes a 16 kbytes
 - Páginas para sistemas embarcados são de 1 kbytes
- endereços virtuais e reais divididos em 2 campos
 - endereço da página
 - endereço da linha (ou palavra), dentro da página

2024

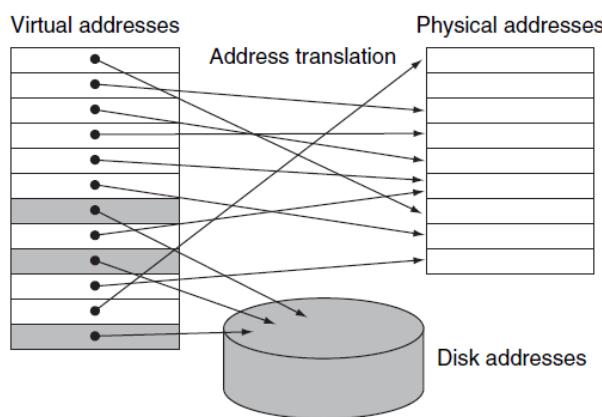
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

73

73

Paginação

- É apenas uma função de mapeamento dos endereços virtuais (do disco) para endereços reais (físicos) na memória principal



PATTERSON, David A. Organização e projeto de computadores a interface hardware/software

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

74

74

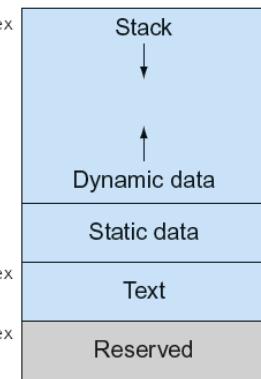
Layout de memória (um processo)

- Text: program code
- Static data: global variables
 - e.g., static variables in C, constant arrays and strings
- Dynamic data: heap
 - E.g., malloc in C, new in Java
- Stack: automatic storage

SP → 0000 007f ffff fffc_{hex}

0000 0000 1000 0000_{hex}

PC → 0000 0000 0040 0000_{hex}



PATTERSON, D. and HENNESSY, J.,
Computer Organization and Design: The
Interface Hardware/Software. ARM edition.

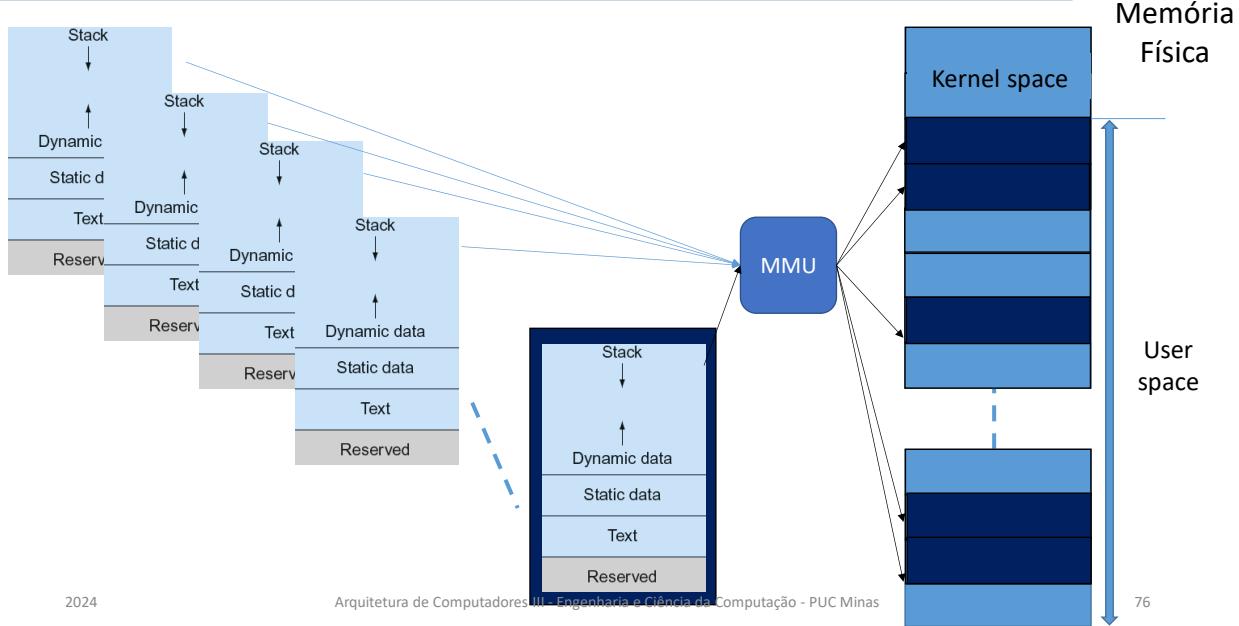
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

75

75

Layout de memória (vários processos)



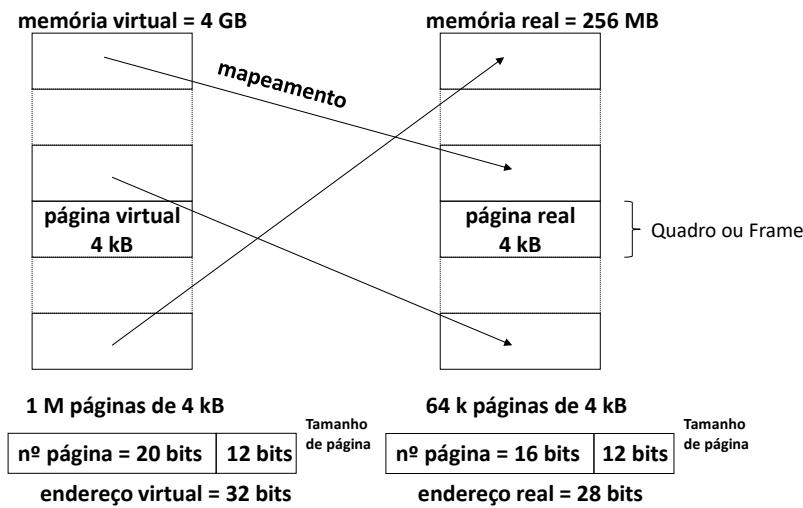
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

76

76

Paginação

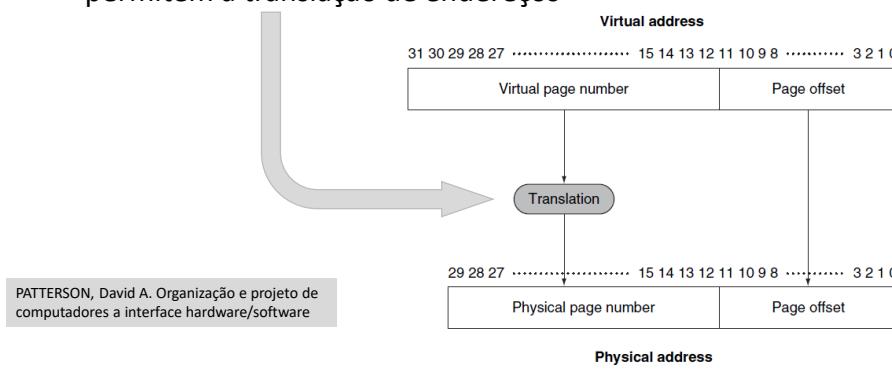


Paginação

- Page fault ocorre quando a página virtual não está na memória principal
- Mapeamento completamente associativo, mais eficiente, ajuda a diminuir alta penalidade dos page faults
- Como transformar endereçamento original do programa no endereçamento real?
- page tables
 - guardam a correspondência entre páginas virtuais e páginas reais
 - permitem a translação de endereços

Paginação

- Como transformar endereçamento original do programa no endereçamento real?
- Page tables
 - guardam a correspondência entre páginas virtuais e páginas reais
 - permitem a translação de endereços



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

79

79

Gerência de processos

- cada processo tem sua própria tabela de páginas
 - processos são compilados para espaços de endereçamento virtuais
 - tabela de páginas define toda a utilização do espaço de endereçamento pelo processo
- sistema operacional é responsável pela alocação de espaço físico para o espaço virtual de cada processo
 - SO carrega tabela de páginas de cada processo
- hardware possui registrador que aponta para início da tabela de páginas do processo atual
- quando novo processo passa a ser ativo, sistema operacional só precisa atualizar valor deste registrador

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

80

80

Paginação

- main memory translation table (MMTT)
 - implementada em hardware (*em tese*)
 - tamanho = nº de páginas na memória principal
- disk memory translation table (DMTT)
 - implementada em software, armazenada na memória principal
 - tamanho = nº de páginas em disco
- **algoritmo de substituição**, em software, para selecionar página da memória principal a ser substituída em caso de page fault
- bom desempenho é garantido pelo princípio de localidade

2024

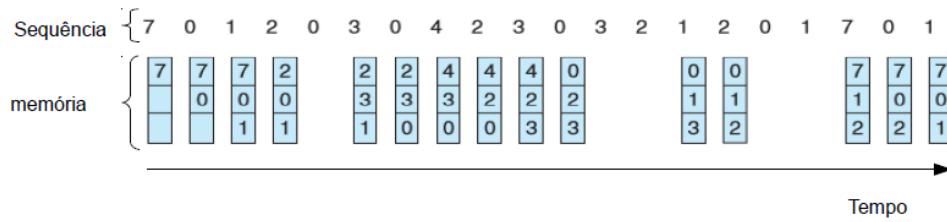
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

81

81

Substituição de páginas

- FIFO
- Memória com 3 frames (quadros)
 - Frame tem tamanho igual a uma página virtual



- Total de requisições = 20
- Total de falhas = 15
- Taxa de falha = $15/20 = 0,75$

https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/9_VirtualMemory.html

2024

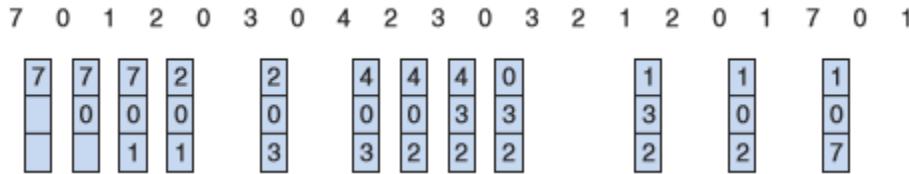
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

82

82

Substituição de páginas

- LRU
- Memória com 3 frames (quadros)
 - Frame tem tamanho igual a uma página virtual



- Total de requisições = 20
- Total de falhas = 12
- Taxa de falha = $12/20 = 0,6$

https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/9_VirtualMemory.html

2024

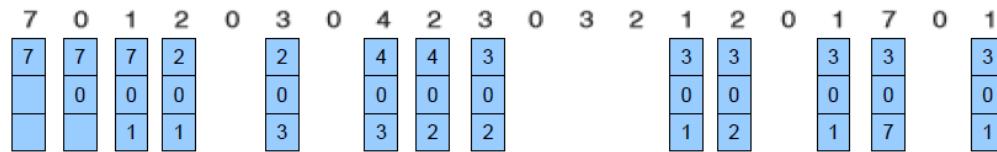
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

83

83

Substituição de páginas

- LFU
- FIFO como critério de desempate
- Memória com 3 frames (quadros)
 - Frame tem tamanho igual a uma página virtual



- Total de requisições = 20
- Total de falhas = 13
- Taxa de falha = $13/20 = 0,65$

https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/9_VirtualMemory.html

2024

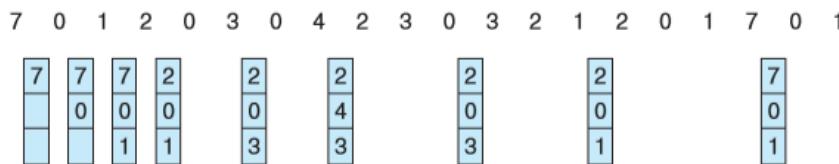
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

84

84

Substituição de páginas

- OPT (solução ótima)
 - Substitui a página que não será usada por um tempo mais longo (prever futuro)
- Memória com 3 frames (quadros)
 - Frame tem tamanho igual a uma página virtual



- Total de requisições = 20
- Total de falhas = 9
- Taxa de falha = $9/20 = 0,45$

https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/9_VirtualMemory.html

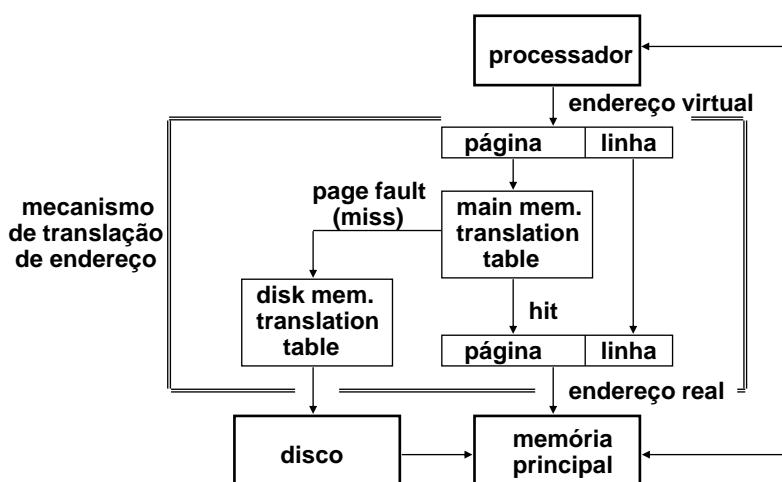
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

85

85

Paginação (translação do endereço)



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

86

86

Translation Look-Aside Buffer

- nº de páginas na memória secundária é muito grande
 - espaço virtual de 2^{32} bytes, páginas de 4k bytes, 4 bytes por entrada na tabela
 - 4 MBytes apenas para a tabela de páginas!!!
 - tamanho excessivo da main memory translation table
- se tabela ficar na memória principal => dois acessos à memória a cada cache miss
- working set = conjunto de páginas mais prováveis de serem acessadas num dado momento, devido ao princípio de localidade
- Translation Look-Aside Buffer (TLB)
 - implementado em hardware
 - traduz endereços virtuais para endereços reais
 - só inclui páginas do working set
 - pode ser considerado como uma “cache” da MMTT
- Main Memory Translation Table (MMTT)
 - implementada em **software**

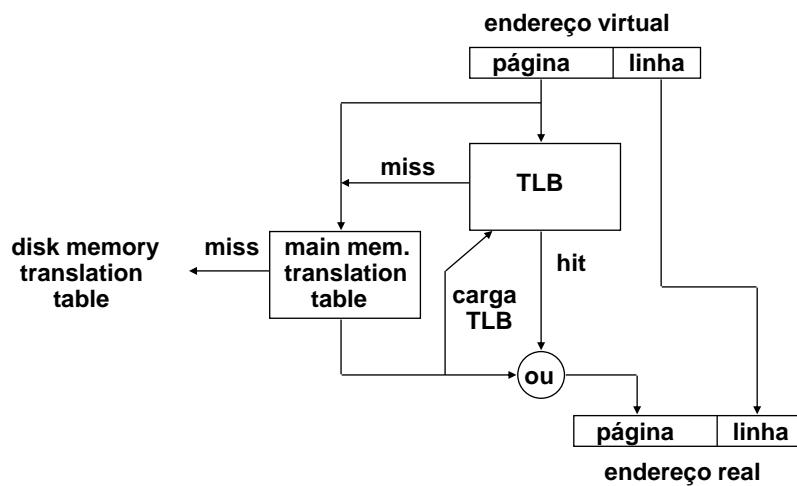
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

87

87

Translation Look-Aside Buffer



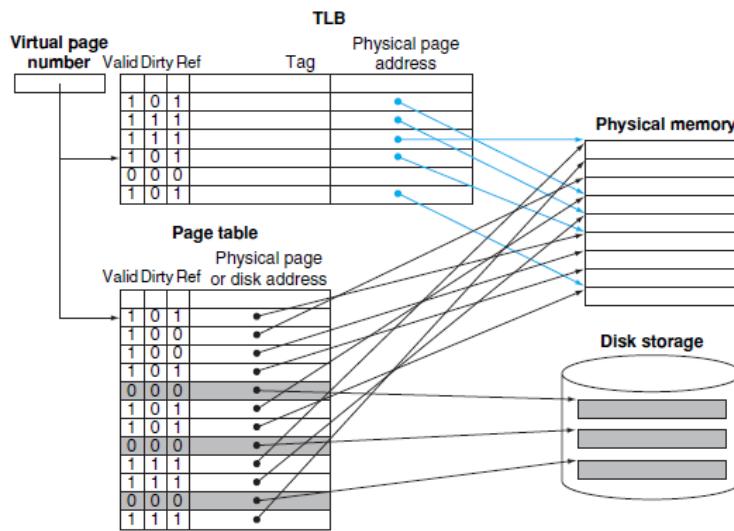
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

88

88

Translation Look-Aside Buffer



2024

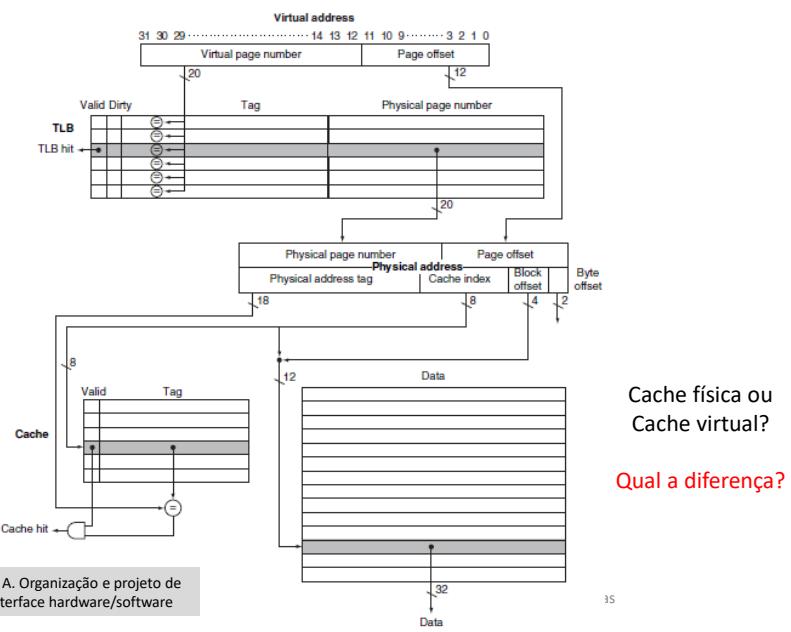
Arquitetura de Computadores III - Engenharia e Ciência da Computação

PATTERSON, David A. Organização e projeto de computadores a interface hardware/software

89

89

Translation Look-Aside Buffer



2024

PATTERSON, David A. Organização e projeto de computadores a interface hardware/software

35

90

90

Translation Look-Aside Buffer

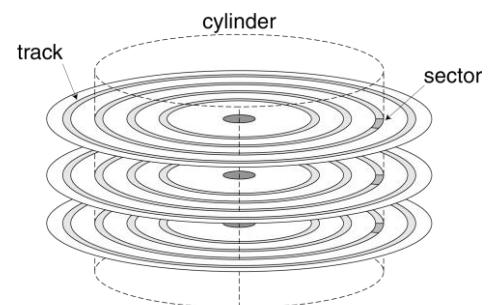
TLB	Page table	Cache	Possible? If so, under what circumstance?
Hit	Hit	Miss	Possible, although the page table is never really checked if TLB hits.
Miss	Hit	Hit	TLB misses, but entry found in page table; after retry, data is found in cache.
Miss	Hit	Miss	TLB misses, but entry found in page table; after retry, data misses in cache.
Miss	Miss	Miss	TLB misses and is followed by a page fault; after retry, data must miss in cache.
Hit	Miss	Miss	Impossible: cannot have a translation in TLB if page is not present in memory.
Hit	Miss	Hit	Impossible: cannot have a translation in TLB if page is not present in memory.
Miss	Miss	Hit	Impossible: data cannot be allowed in cache if the page is not in memory.

PATTERSON, David A. Organização e projeto de computadores a interface hardware/software

Disco

- Para acessar os dados:

- Tempo de busca: posiciona a cabeça sobre a trilha correta (média de 3 a 14 ms)
- Latência rotacional: espera pelo setor desejado (0,5 rpm)
- Tempo de transferência: recupera os dados (um ou mais setores; ex: 30 a 80 MB/seg)



PATTERSON, David A. Organização e projeto de computadores a interface hardware/software

Desempenho do Disco

- Tempo de Disco = Tempo de busca + Latência rotacional + Tempo de transferência
- A latência rotacional média para a informação desejada está a meio caminho ao redor do disco.
- TB = 1ms
- $$\text{LRM} = \frac{0,5 \text{ rotação}}{5400 \text{ RPM}} = \frac{0,5 \text{ rotação}}{5400\text{RPM}/60(\text{seg})} = 0,0056\text{s} = 5,6\text{ms}$$
- $$\text{TT} = 1\text{kB} / 50\text{MB/s} = 1 * 2^{10} / 50 * 10^6 = 20,48\mu\text{s}$$
- $$\text{TD} = 1\text{ms} + 5,6\text{ms} + 0,02048\text{ms} = 6,62048\text{ms}$$

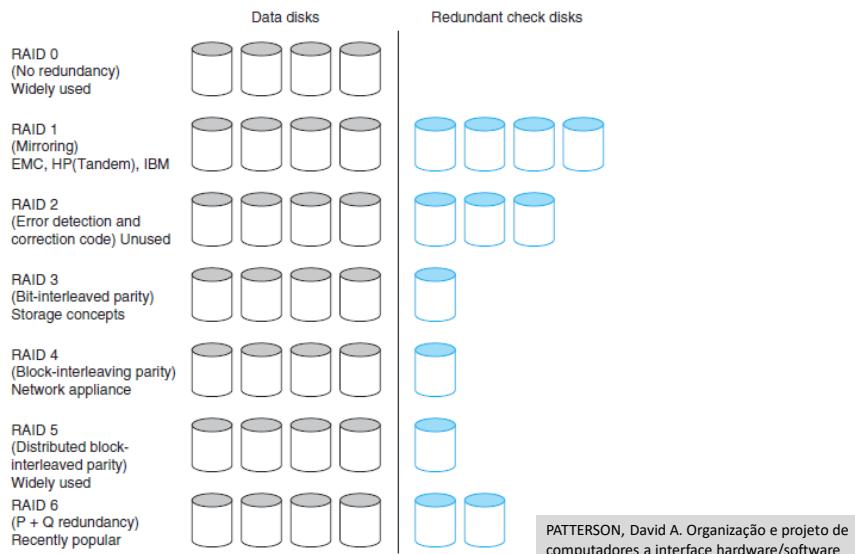
2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

93

93

Redundant Arrays of Inexpensive Disks (RAID)



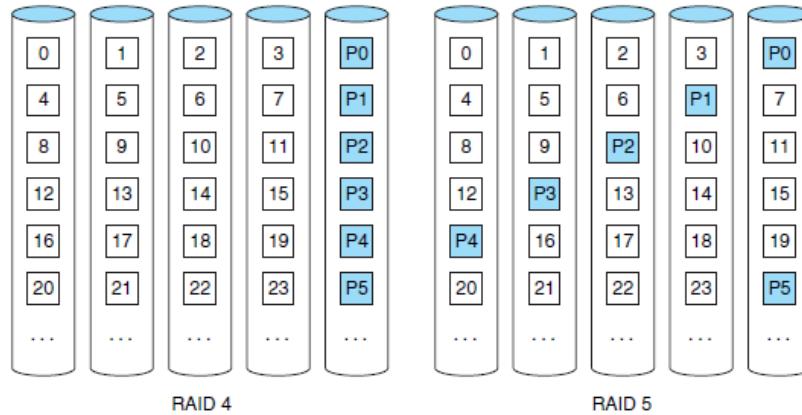
2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

94

94

Redundant Arrays of Inexpensive Disks (RAID)



PATTERSON, David A. Organização e projeto de computadores a interface hardware/software

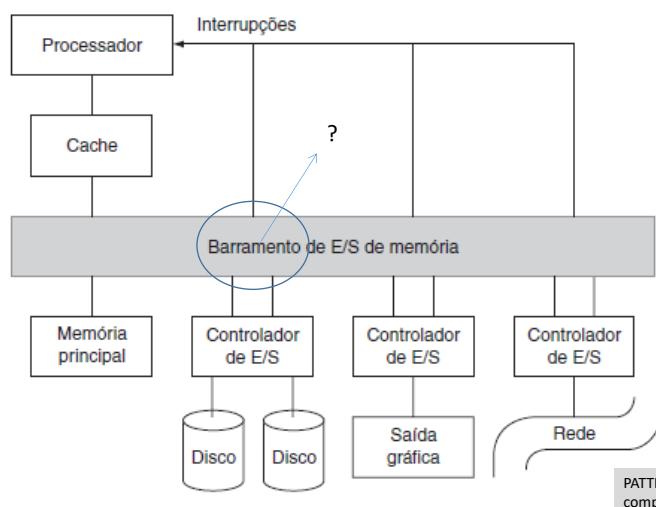
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

95

95

Visão geral



PATTERSON, David A. Organização e projeto de computadores a interface hardware/software

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

96

96

A interface hardware / software

Arquitetura de Computadores III

Pipeline Escalar de Instruções

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

97

97

A interface hardware / software

O Processador: Caminho de Dados e Controle

2024

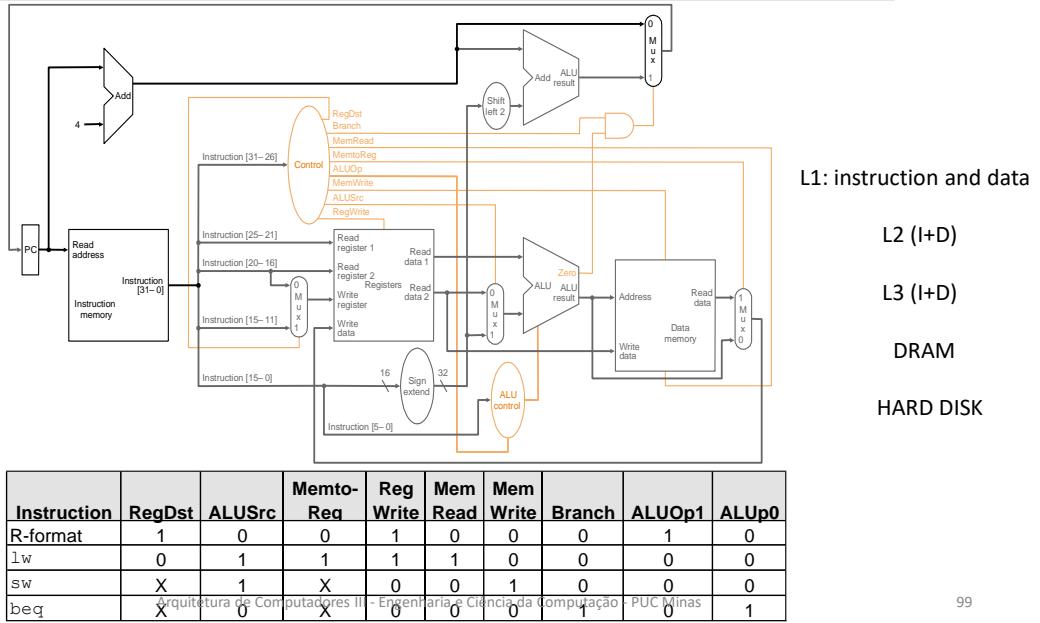
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

98

98

49

Controle



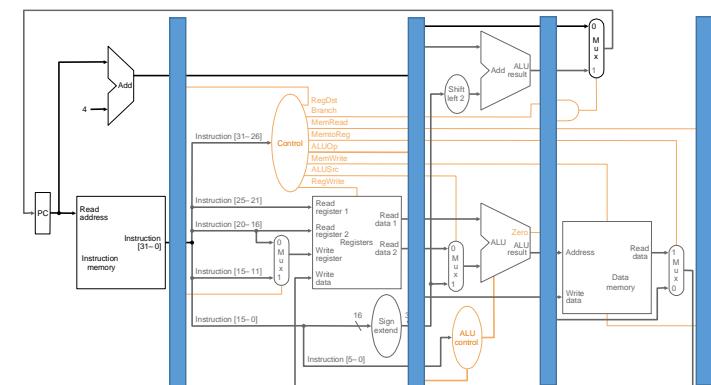
PATTERSON, David A.
Organização e projeto
de computadores a
interface
hardware/software

2024

99

99

Controle



Instruction	RegDst	ALUSrc	Memto-Req	Reg Write	Mem Read	Mem Write	Branch	ALUOp1	ALUOp0
R-format	1	0	0	1	0	0	0	1	0
lw	0	1	1	1	1	0	0	0	0
sw	X	1	X	0	0	1	0	0	0
beq	X	0	X	0	0	0	1	0	1

PATTERSON, David A.
Organização e projeto
de computadores a
interface
hardware/software

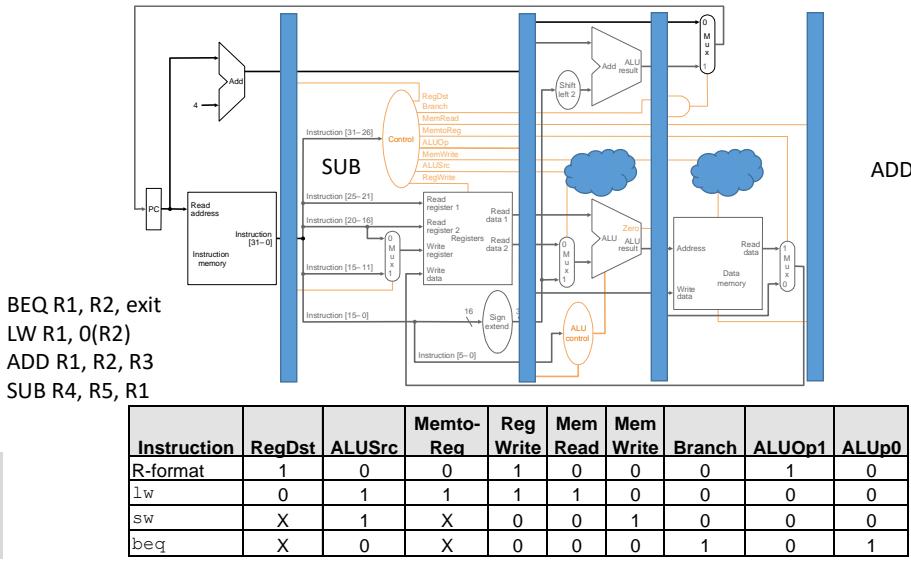
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

100

100

Controle



PATTERSON, David A.
Organização e projeto
de computadores a
interface
hardware/software

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

101

101

Implementação de um único ciclo

Classe	Memória de Instruções	Leitura Registrador	Operação UAL	Memória de Dados	Escrita no registrador	Total
Formato R	2	1	2 (16)	0	1	6 ns
Load Word	2	1	2	2	1	8 ns
Store Word	2	1	2	2		7 ns
Branch	2	1	2			5 ns
Jump	2					2 ns

Formato R sem e com ponto flutuante

PATTERSON, David A.
Organização e projeto
de computadores a
interface
hardware/software

- O ciclo de clock é definido em função da duração da instrução mais longa = 8 ns.
- Uma máquina com ciclo de clock variável: 2ns a 8ns.
 - Ciclo de clock: $8 \times 24\% + 7 \times 12\% + 6 \times 44\% + 5 \times 18\% + 2 \times 2\% = 6,3$ ns
- Ganho de desempenho: $\text{Tempo execução clock fixo} / \text{Tempo de execução clock variável} = 8 / 6,3 = 1,27$
- No caso de operações em ponto flutuante podemos ter para a multiplicação: $2 + 1 + 16 + 1 = 20$ ns
- A mesma relação pode ser feita para operações em ponto flutuante e o ganho favorável ao clock variável pode ser ainda maior.

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

102

102

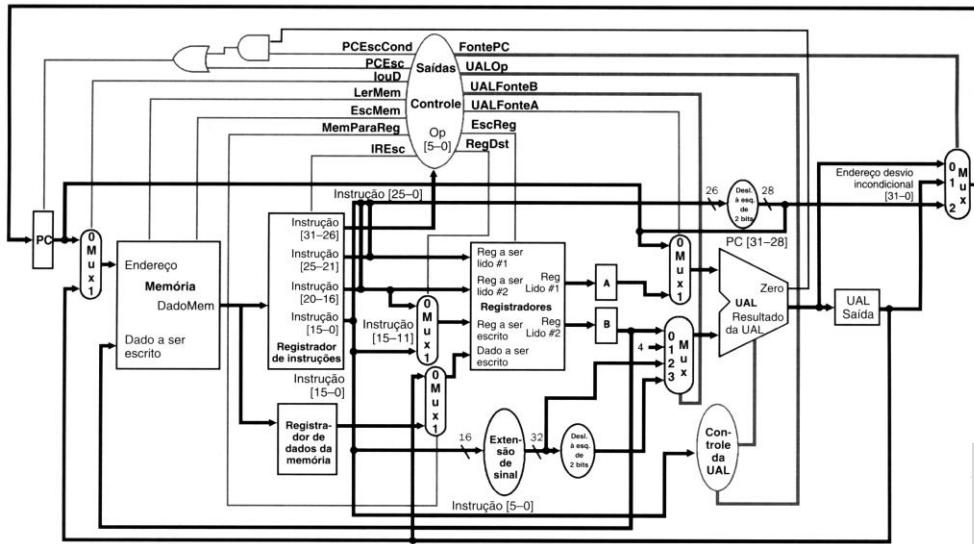
Cinco passos para execução multiciclo

- Busca de instrução
 - Pode ser descrito de forma sucinta usando a “Linguagem Transferência-Registrador” - RTL “Register-Transfer Language”
 - IR = Memory[PC];
 - PC = PC + 4;
- Decodifica instrução e Busca Registrador
 - Leia os registradores rs e rt para o caso de precisarmos deles
 - Compute o endereço de desvio no caso da instrução ser um desvio
 - RTL:
 - A = Reg[IR[25-21]];
 - B = Reg[IR[20-16]];
 - ALUOut = PC + (sign-extend(IR[15-0]) << 2);
 - Nós não ativamos linhas de controle baseados em tipo de instrução.
- Execução, Cálculo de Endereço de Memória, ou Conclusão de Desvio
 - A ULA está desempenhando uma das 3 funções, baseada no tipo de instrução
 - Referência à memória: ALUOut = A + sign-extend(IR[15-0]);
 - Tipo-R: ALUOut = A op B;
 - Desvio: if (A==B) PC = ALUOut;
- Acesso à Memória ou Conclusão de instruções tipo-R
 - Carrega ou armazena na memória: MDR = Memory[ALUOut]; ou Memory[ALUOut] = B;
 - Finaliza instruções Tipo-R: Reg[IR[15-11]] = ALUOut;
- Passo de “Write-back”
 - Reg[IR[20-16]] = MDR;
-

PATTERSON, David A.
Organização e projeto
de computadores a
interface
hardware/software

Instruções levam de 3 a 5 ciclos.

Abordagem multiciclo



PATTERSON, David A.
Organização e projeto
de computadores a
interface
hardware/software

Exemplo de sinais de controle de uma instrução para abordagem multiciclo

- Busca da instrução:
 - LerMem =1, IREsc = 1, IouD = 0, UALFonteA = 0, UALFonteB = 01, UALOp = 00, PCEsc = 1
- Decodificação e busca do registrador:
 - UALFonteA = 0, UALFonteB = 11, UALOp = 00
- Execução, cálculo do endereço de memória ou efetivação do desvio condicional:
 - Instruções de referência à memória: UALFonteA = 1, UALFonteB = 10, UALOp = 00
 - Instruções do tipo R: UALFonteA = 1, UALFonteB = 00, UALOp = 10
 - Desvio condicional: UALFonteA = 1, UALFonteB = 00, UALOp = 01, PCEscCond = 1, FontePC = 01
 - Desvio incondicional: FontePC = 11, PCEsc = 1
- Escrita em memória ou registrador:
 - Instruções de referência à memória: LerMem = 1 ou EscMem = 1, se lw, IouD = 1
 - Instruções do tipo R: RegDst = 1, EscReg = 1, MemParaReg = 0
- Leitura de memória e escrita em registrador:
 - MemParaReg = 1, EscReg = 1, RegDst = 0

PATTERSON, David A.
Organização e projeto
de computadores a
interface
hardware/software

Questão simples

- Quantos ciclos leva para executar esse código?

```

lw $t2, 0($t3)
lw $t3, 4($t3)
beq $t2, $t3, Label #assuma não
add $t5, $t2, $t3
sw $t5, 8($t3)

```

Label: ...

- O que está acontecendo durante o oitavo ciclo de execução?
- Em que ciclo acontece realmente a adição de \$t2 e \$t3?

A interface hardware / software

Pipeline

MIPS

2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computaçäo - PUC Minas

107

107

Pipeline -primeira parte

- 1. Introdução
- 2. Pipelines aritméticos
- 3. Pipelines de instruções
- 4. Desempenho
- 5. Conflitos de memória
- 6. Dependências em desvios

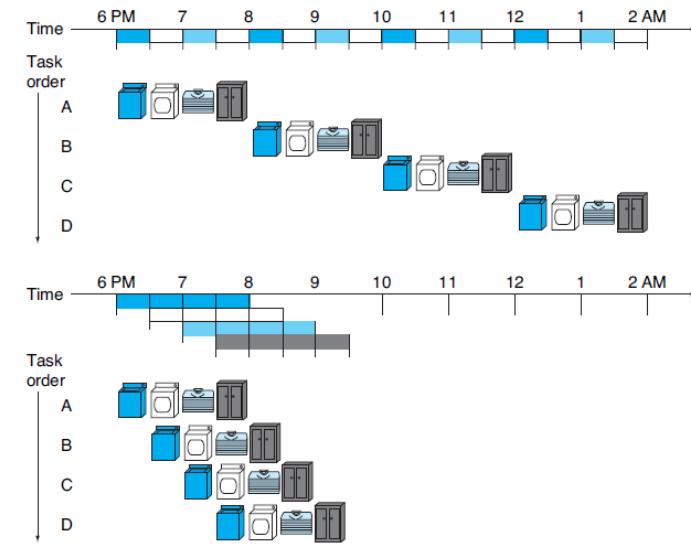
2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computaçäo - PUC Minas

108

108

Introdução



PATTERSON, David A.
Organização e projeto
de computadores a
interface
hardware/software

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

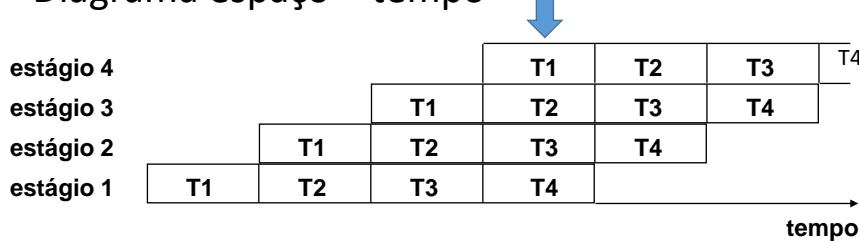
109

109

Introdução

- Objetivo: aumento de desempenho
 - divisão de uma tarefa em N estágios
 - N tarefas executadas em paralelo, uma em cada estágio

- Diagrama espaço – tempo



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

110

110

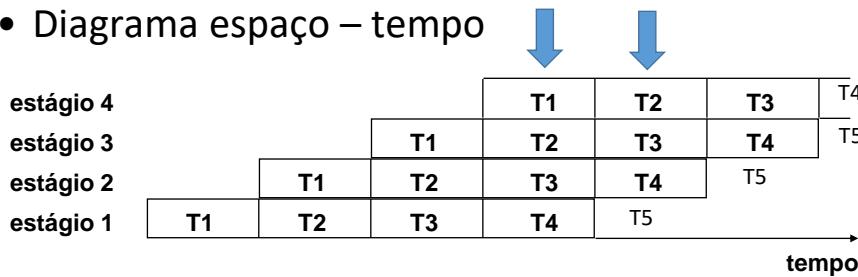
Introdução

ADD R1, R2, R3
SUB R4, R5, R1

- **Objetivo: aumento de desempenho**

- divisão de uma tarefa em N estágios
- N tarefas executadas em paralelo, uma em cada estágio

- **Diagrama espaço – tempo**

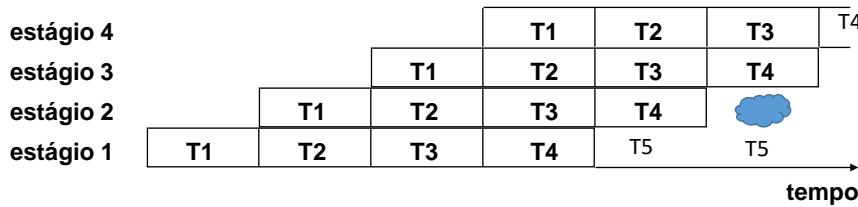


Introdução

- **Objetivo: aumento de desempenho**

- divisão de uma tarefa em N estágios
- N tarefas executadas em paralelo, uma em cada estágio

- **Diagrama espaço – tempo**

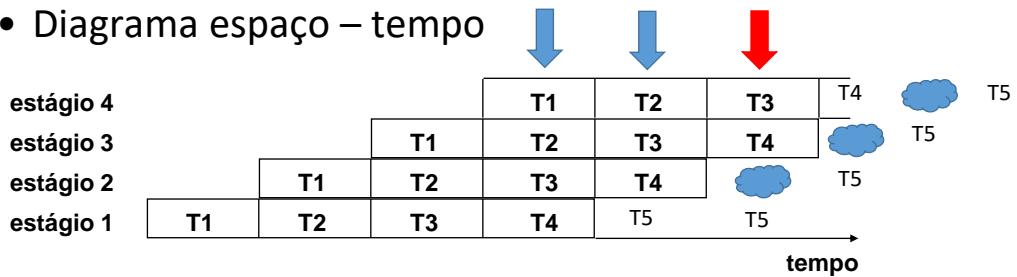


Introdução

- **Objetivo: aumento de desempenho**

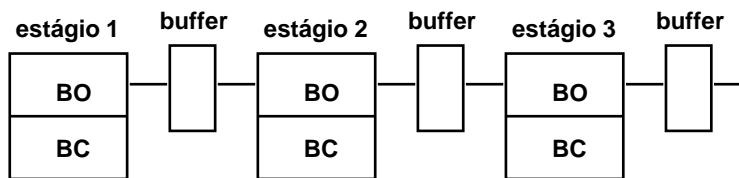
- divisão de uma tarefa em N estágios
- N tarefas executadas em paralelo, uma em cada estágio

- **Diagrama espaço – tempo**



Introdução

- bloco operacional e bloco de controle independentes para cada estágio
- necessidade de buffers entre os estágios



- pipelines de instruções
- pipelines aritméticos

Pipelines Aritméticos

- exemplo: soma em ponto flutuante executada em 4 estágios
 - 1. comparar expoentes
 - 2. acertar expoentes dos operandos
 - 3. somar
 - 4. normalizar resultado

- exemplo

0.157	$\times 10^6$	-	0.842	$\times 10^5$
0.157	$\times 10^6$	-	0.0842	$\times 10^6$
0.0628	$\times 10^6$			
0.628	$\times 10^5$			

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

115

115

Pipelines de Instruções

- 2 estágios
 - fetch / decodificação, execução
- 3 estágios
 - fetch, decodificação / busca de operandos, execução
- 4 estágios
 - fetch, decodificação / busca de operandos, execução, store
- 5 estágios
 - fetch, decodificação / cálculo de endereço de operandos, busca de operandos, execução, store
- 6 estágios
 - fetch, decodificação, cálculo de endereço de operandos, busca de operandos, execução, store
 - estágio só para decodificação é bom em processadores CISC

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

116

116

Desempenho

- existe um tempo inicial até que o pipeline “encha”
- cada tarefa leva o mesmo tempo, com ou sem pipeline
- média de tempo por tarefa é no entanto dividida por N

$\frac{s \text{ tarefas}}{N \text{ estágios}}$

primeira tarefa: N ciclos de relógio

s – 1 tarefas seguintes: s – 1 ciclos de relógio

Tempo total com pipeline = N + (s – 1)

Tempo total sem pipeline = s N

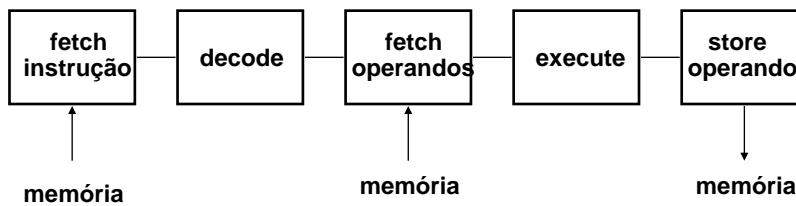
$$\text{Speed-up teórico} = \frac{s N}{N + (s - 1)}$$

Problemas no desempenho

- como dividir todas as instruções num mesmo conjunto de estágios?
- como obter estágios com tempos de execução similares?
- conflitos de memória
 - acessos simultâneos à memória por 2 ou mais estágios
- dependências de dados
 - instruções dependem de resultados de instruções anteriores, ainda não completadas
- instruções de desvio
 - instrução seguinte não está no endereço seguinte ao da instrução anterior

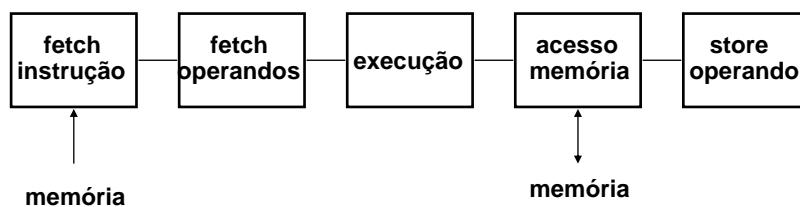
Conflitos de memória

- problema: acessos simultâneos à memória por 2 ou mais estágios



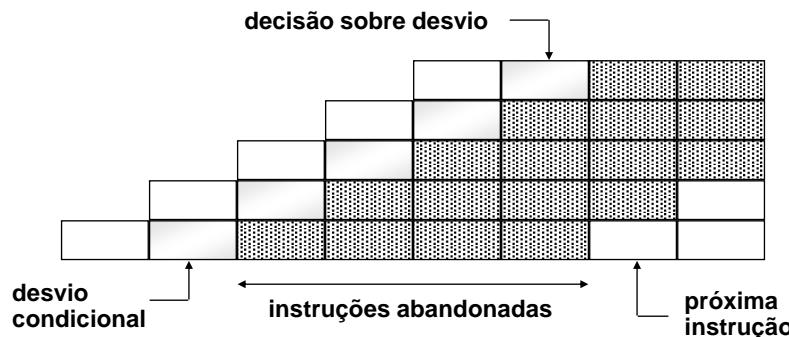
Processadores RISC

- memória é acessada apenas por instruções LOAD e STORE
- apenas um estágio do pipeline faz acesso a operandos de memória
 - apenas 1 instrução pode estar executando acesso a dados a cada instante
- se caches de dados e instruções são separadas, não há nenhum conflito de acesso à memória



Dependências em desvios

- efeito de desvios condicionais
 - se o desvio ocorre, pipeline precisa ser esvaziado
 - não se sabe se desvio ocorrerá ou não até o momento de sua execução



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

121

121

Dependências de desvios

- instruções abandonadas não podem ter afetado conteúdo de registradores e memórias
 - isto é usualmente automático, porque escrita de valores é sempre feita no último estágio do pipeline
- deve-se procurar antecipar a decisão sobre o desvio para o estágio mais cedo possível
- desvios incondicionais
 - sabe-se que é um desvio desde a decodificação da instrução (segundo estágio do pipeline)
 - é possível evitar abandono de número maior de instruções
 - problema: em que estágio é feito o cálculo do endereço efetivo do desvio?

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

122

122

Técnicas de tratamento dos desvios condicionais

- 1. Executar os dois caminhos do desvio
- buffers paralelos de instruções

- 2. Prever o sentido do desvio
- predição estática
- predição dinâmica

- 3. Eliminar o problema
- “delayed branch”

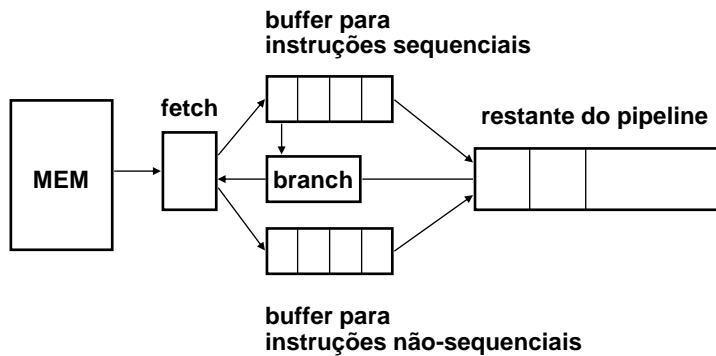
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

123

123

Buffers paralelos de instruções



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

124

124

Predição estática

- supor sempre mesma direção para o desvio
 - desvio sempre ocorre
 - desvio nunca ocorre
- compilador define direção mais provável
 - instrução de desvio contém bit de predição, ligado / desligado pelo compilador
 - início de laço (ou desvio para frente): desvio improvável
 - final de laço (ou desvio para trás): desvio provável
- até 85 % de acerto é possível

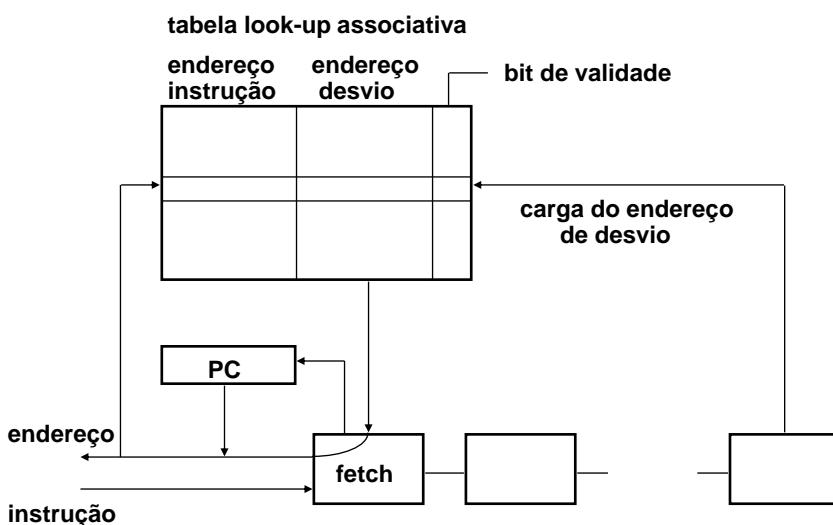
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

125

125

Predição dinâmica



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

126

126

Predição dinâmica

- tabela look-up associativa armazena triplas
 - endereços das instruções de desvio condicional mais recentemente executadas
 - endereços de destino destes desvios
 - bit de validade, indicando se desvio foi tomado na última execução
- quando instrução de desvio condicional é buscada na memória
 - é feita comparação associativa na tabela, à procura do endereço desta instrução
 - se endereço é encontrado e bit de validade está ligado, o endereço de desvio armazenado na tabela é usado
 - ao final da execução da instrução, endereço efetivo de destino do desvio e bit de validade são atualizados na tabela
- tabela pode utilizar diversos mapeamentos e algoritmos de substituição

Predição dinâmica

- variação: branch history table
 - contador associado a cada posição da tabela
 - a cada vez que uma instrução de desvio contida na tabela é executada ...
 - contador é incrementado se desvio ocorre
 - contador é decrementado se desvio não ocorre
 - valor do contador é utilizado para a predição

Delayed Branch

- desvio não ocorre imediatamente, e sim apenas após uma ou mais instruções seguintes
- caso mais simples: pipeline com 2 estágios – fetch + execute
 - desvio é feito depois da instrução seguinte
 - instrução seguinte não pode ser necessária para decisão sobre ocorrência do desvio
 - compilador reorganiza código
 - tipicamente, em 70% dos casos encontra-se instrução para colocar após o desvio
- pipeline com N estágios
 - desvio é feito depois de $N - 1$ instruções

Pipeline - segunda parte

- 1. Introdução
- 2. Dependências verdadeiras
- 3. Dependências falsas
- 4. Pipeline interlock
- 5. Forwarding (adiantamento de dados)

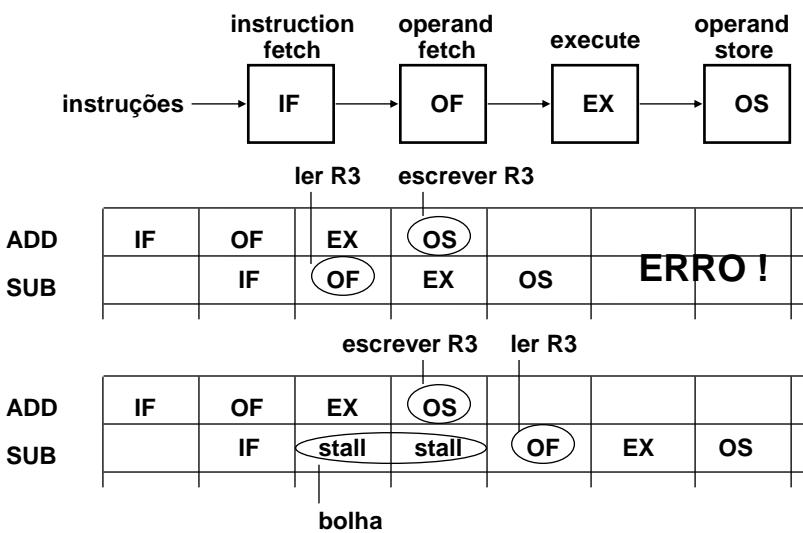
Introdução

- problema: instruções consecutivas podem fazer acesso aos mesmos operandos
 - execução da instrução seguinte pode depender de operando calculado pela instrução anterior
- caso particular: instrução precisa de resultado anterior (p.ex. registrador) para cálculo de endereço efetivo de operando
- tipos de dependências de dados
 - dependência verdadeira
 - antidependência
 - dependência de saída

Dependências verdadeiras

- exemplo
 - 1. ADD R3, R2, R1 ; $R3 = R2 + R1$
 - 2. SUB R4, R3, 1 ; $R4 = R3 - 1$
- instrução 2 depende de valor de R3 calculado pela instrução 1
- instrução 1 precisa atualizar valor de R3 antes que instrução 2 busque os seus operandos
- read-after-write hazard
- pipeline precisa ser parado durante certo número de ciclos

Dependências verdadeiras



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

133

133

Dependências falsas

- Antidependência
- exemplo
 - 1. ADD R3, R2, R1 ; R3 = R2 + R1
 - 2. SUB R2, R4, 1 ; R2 = R4 – 1
- instrução 1 utiliza operando em R2 que é escrito pela instrução 2
- instrução 2 não pode salvar resultado em R2 antes que instrução 1 tenha lido seus operandos
- write-after-read hazard
- não é um problema em pipelines onde a ordem de execução das instruções é mantida
 - escrita do resultado é sempre o último estágio
- problema em processadores superescalares

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

134

134

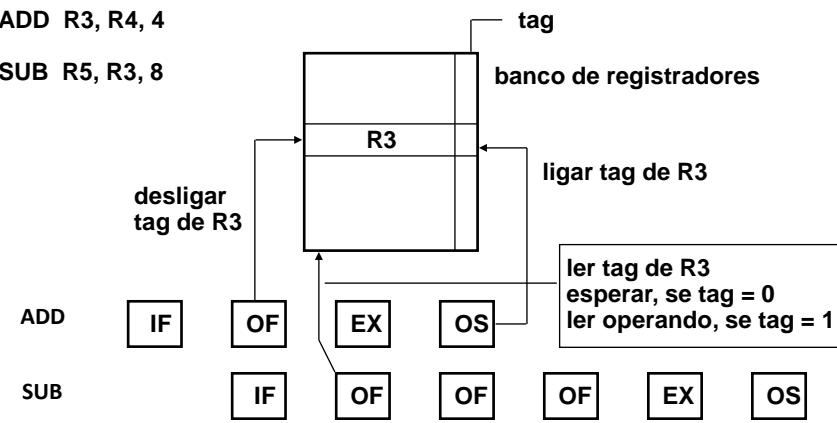
Dependências falsas

- Dependência de saída
- exemplo
 - 1. ADD R3, R2, R1 ; $R3 = R2 + R1$
 - 2. SUB R2, R3, 1 ; $R2 = R3 - 1$
 - 3. ADD R3, R2, R5 ; $R3 = R2 + R5$
- instruções 1 e 3 escrevem no mesmo operando em R3
- instrução 1 tem que escrever seu resultado em R3 antes do que a instrução 3, senão valor final de R3 ficará errado
- write-after-write hazard
- também só é problema em processadores superescalares

Pipeline interlock

- método para manter seqüênciia correta de leituras e escritas em registradores
- tag de 1 bit é associado a cada registrador
 - tag = 0 indica valor não válido, = 1 indica valor válido
- se instrução que é buscada e decodificada escreve num registrador, o tag do mesmo é zerado
- tag é ligado quando instrução escreve o valor no registrador
- outras instruções que sejam executadas enquanto tag está zerado devem esperar tag = 1 para ler valor deste registrador

Pipeline interlock



2024

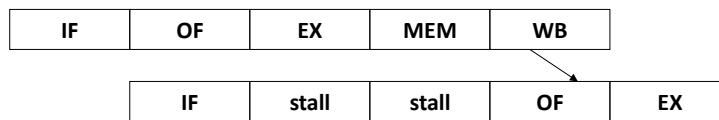
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

137

137

Forwarding

- exemplo
 - ADD R3, R2, R0
 - SUB R4, R3, 8
- instrução SUB precisa do valor de R3, calculado pela instrução ADD
 - valor é escrito em R3 por ADD no último estágio WB (write-back)
 - valor é necessário em SUB no terceiro estágio
 - instrução SUB ficará presa por 2 ciclos no 2º estágio do pipeline



supõe-se escrita no banco de registradores na primeira metade do ciclo e leitura na segunda metade

2024

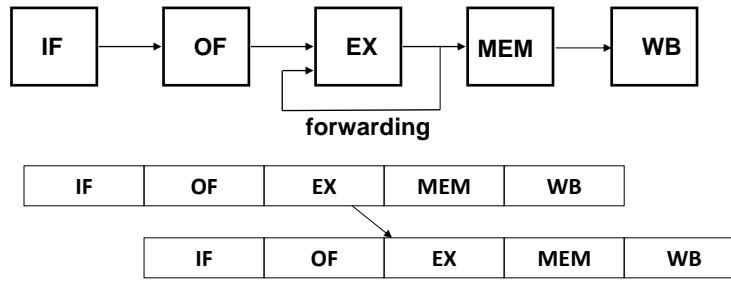
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

138

138

Forwarding

- caminho interno dentro do pipeline entre a saída da ALU e a entrada da ALU
 - evita stall do pipeline



2024

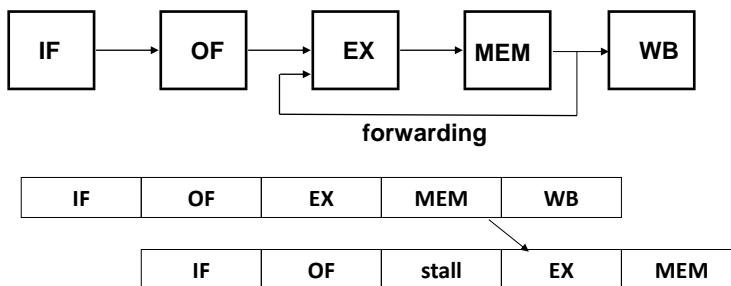
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

139

139

Forwarding

- exemplo 2
 - LOAD R3, 100 (R0)
 - ADD R1, R2, R3
- forwarding: caminho interno dentro do pipeline entre a saída da memória de dados e a entrada da ALU



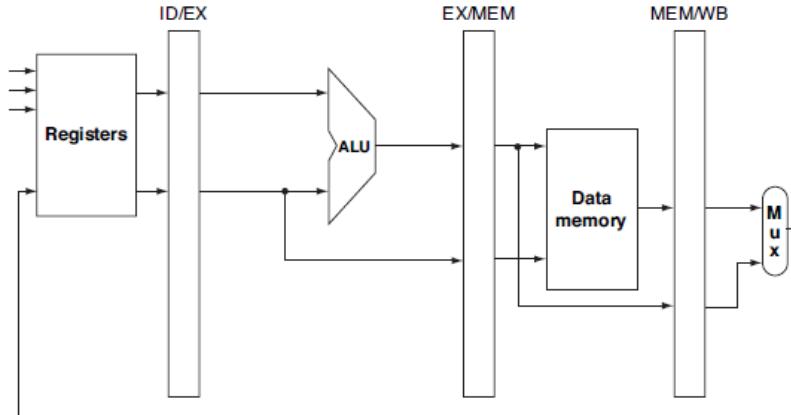
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

140

140

Como fazer o adiantamento de dados?



PATTERSON, David A.
Organização e projeto
de computadores a
interface
hardware/software

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

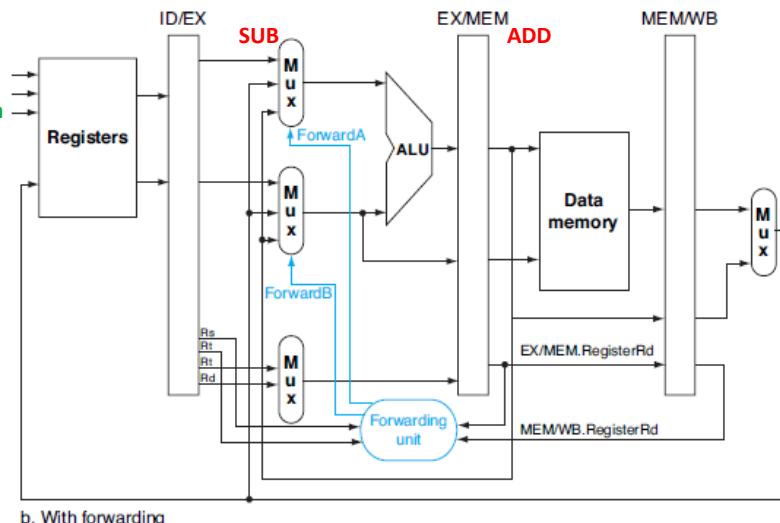
141

141

Como fazer o adiantamento de dados?

End escrita = End leitura

ADD R1, R2, R3
SUB R4, R1, R5



PATTERSON, David A.
Organização e projeto
de computadores a
interface
hardware/software

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

142

142

Pipeline -terceira parte

- 1. Pipeline do MIPS
- 2. Dependências no Pipeline
- 3. Forwarding
- 4. Exemplos
- 5. Instruções de Desvio no Pipeline

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

143

143

Pipeline do MIPS

- IF – Instruction Fetch
- ID – Instruction Decode
- EX – Execution
- MEM – Memory Access
- WB – Write Back

Instrução	Clock								
	1	2	3	4	5	6	7	8	9
i	IF	ID	EX	MEM	WB				
i + 1		IF	ID	EX	MEM	WB			
i + 2			IF	ID	EX	MEM	WB		
i + 3				IF	ID	EX	MEM	WB	
i + 4					IF	ID	EX	MEM	WB

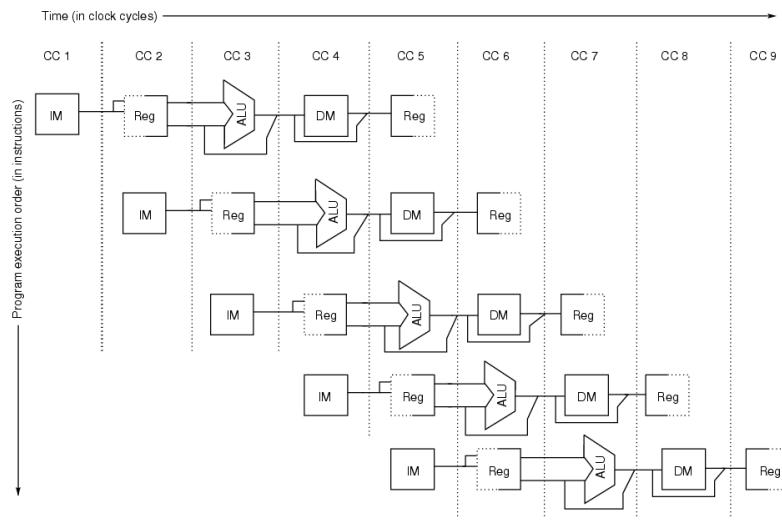
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

144

144

Pipeline do MIPS



PATTERSON, David A.
Organização e projeto
de computadores a
interface
hardware/software

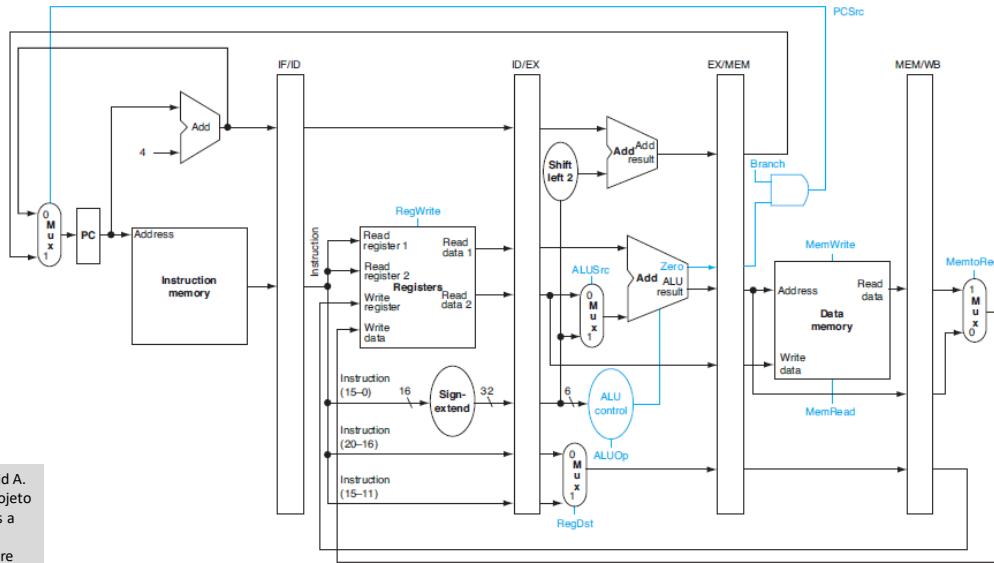
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

145

145

Pipeline do MIPS



PATTERSON, David A.
Organização e projeto
de computadores a
interface
hardware/software

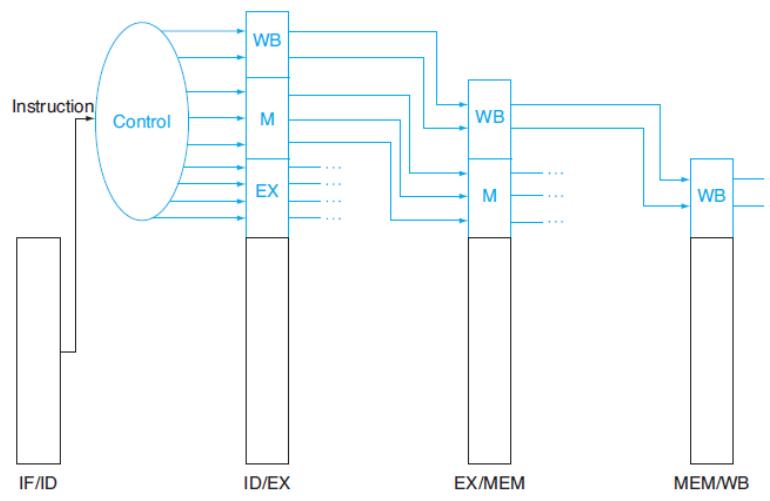
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

146

146

Pipeline do MIPS



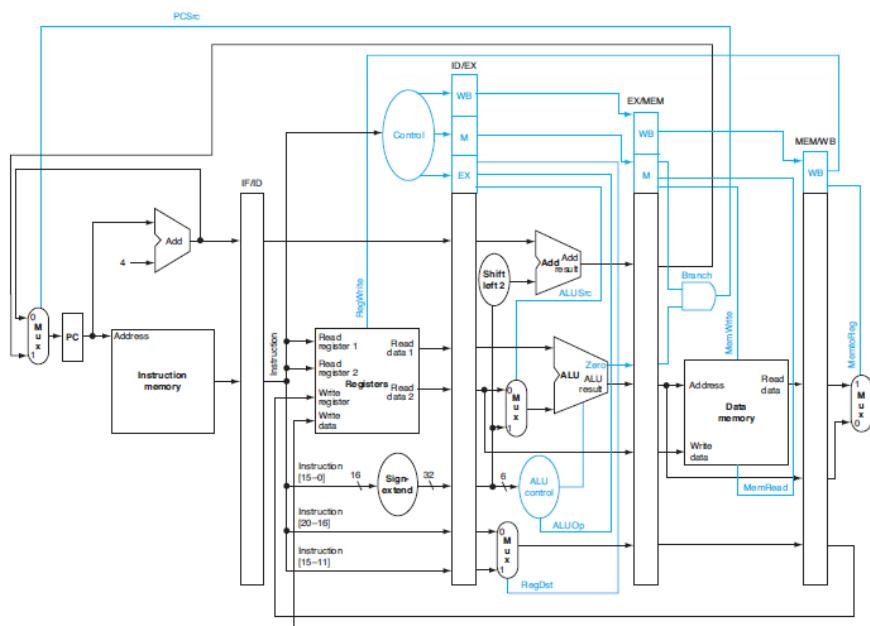
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

147

147

Pipeline do MIPS

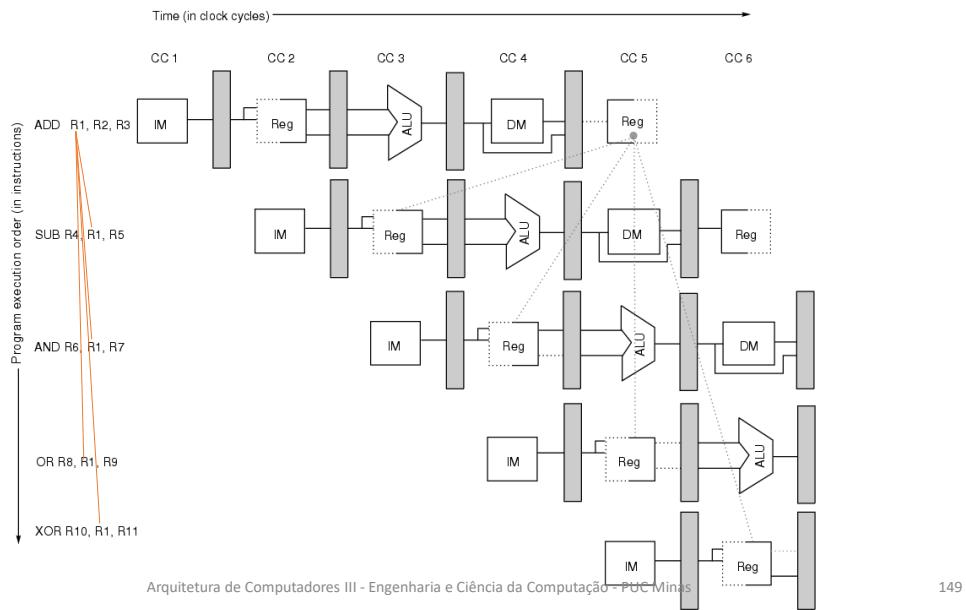


2024

148

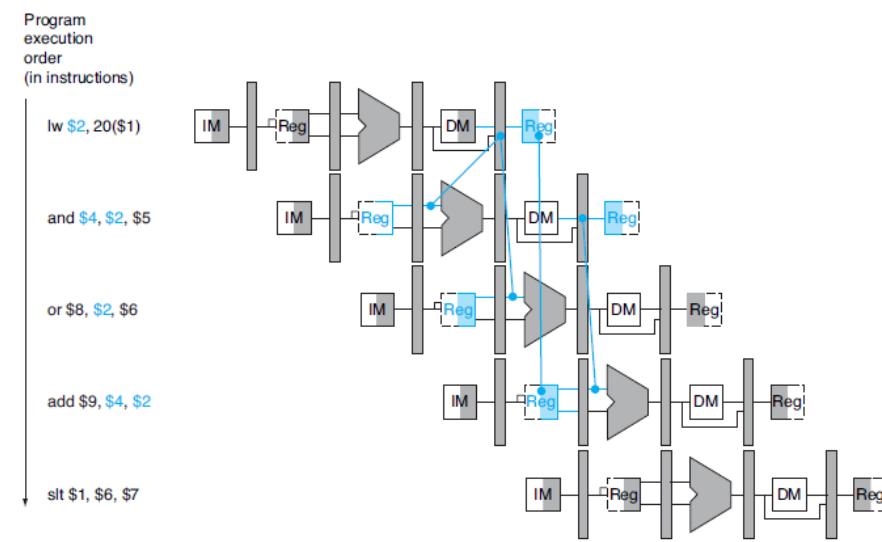
148

Dependências no Pipeline



149

Forwarding nem sempre elimina todas as bolhas



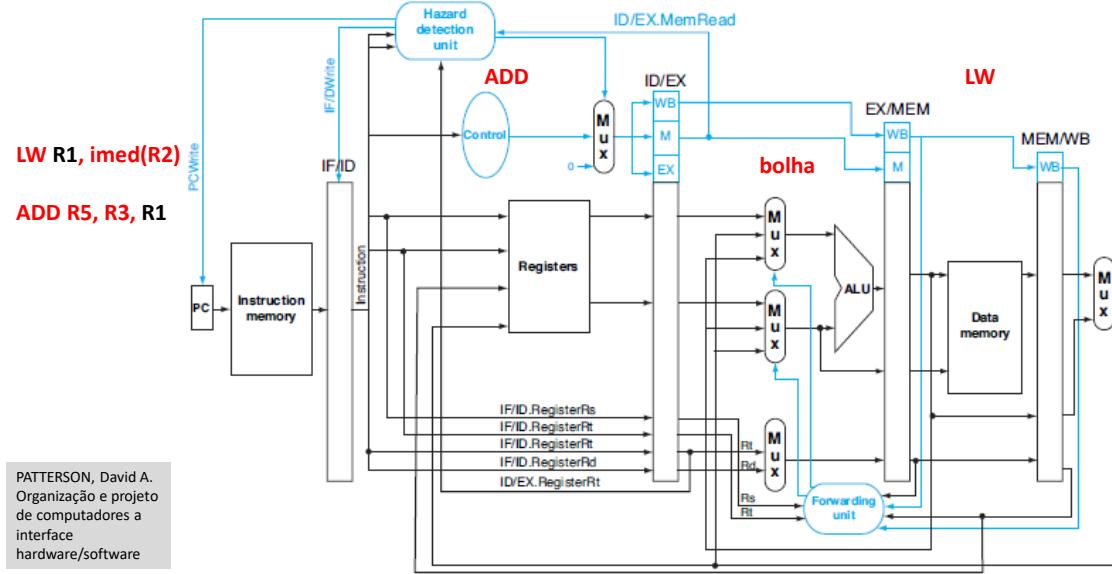
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

150

150

Como gerar bolha?



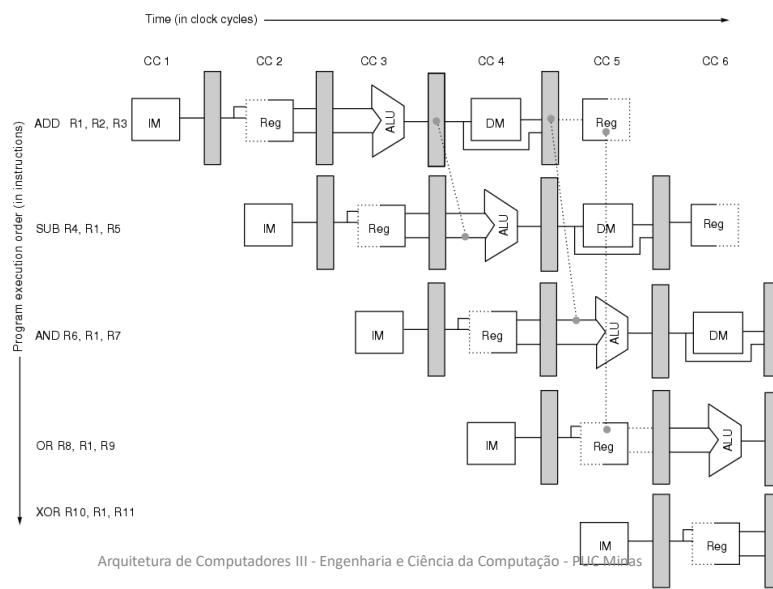
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

151

151

Forwarding



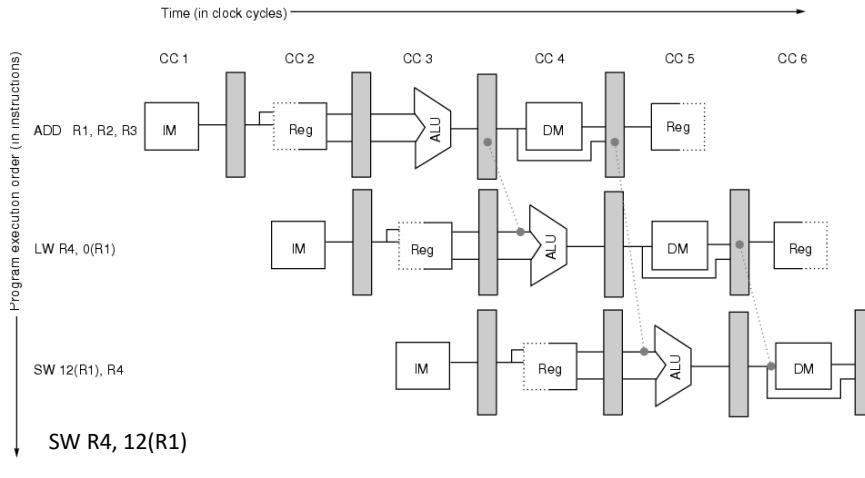
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

152

152

Forwarding



PATTERSON, David A.
Organização e projeto
de computadores a
interface
hardware/software

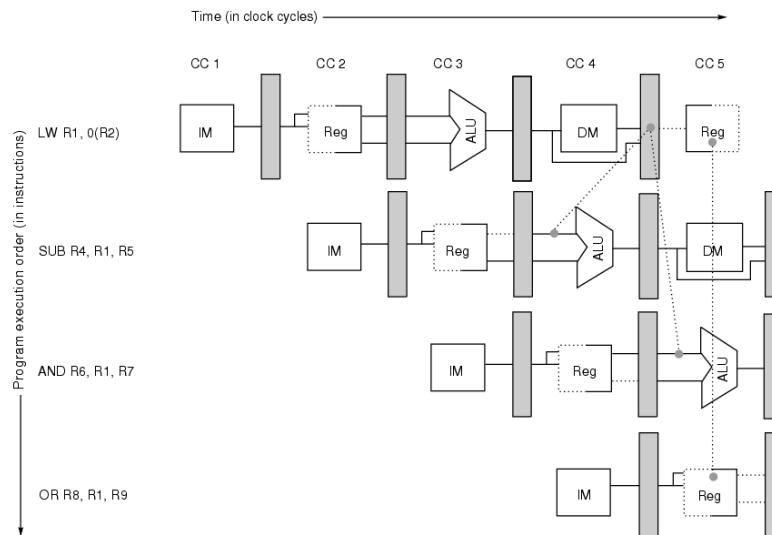
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

153

153

Forwarding



PATTERSON, David A.
Organização e projeto
de computadores a
interface
hardware/software

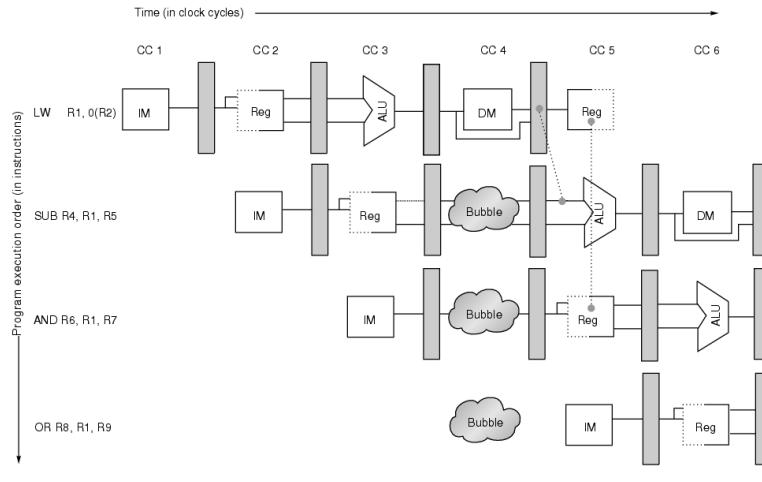
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

154

154

Forwarding



PATTERSON, David A.
Organização e projeto
de computadores a
interface
hardware/software

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

155

155

Instruções de Desvio no Pipeline

Desvio	IF	ID	EX	MEM	WB
IS	IF	stall	stall	IF	ID
IS + 1				IF	ID
IS + 2				IF	ID
IS + 3				IF	ID
IS + 4				IF	ID
IS + 5					IF

IS – Instrução Sucessora.

- Uma instrução de desvio causa um atraso de três ciclos no pipeline:
 - Um ciclo por repetir o estágio de busca (IF);
 - Dois ciclos ociosos (stall).

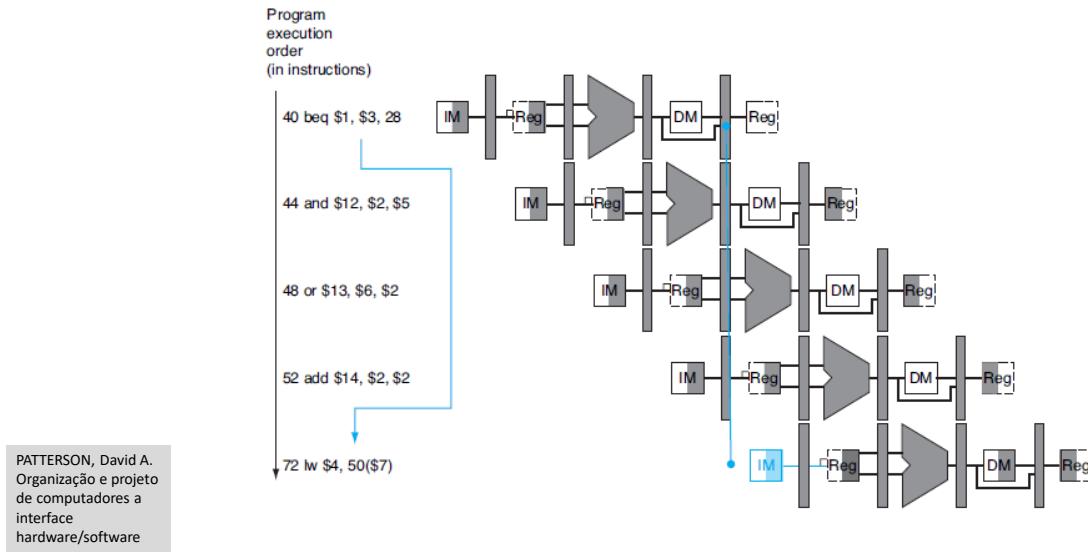
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

156

156

Desvio e descarte de instruções



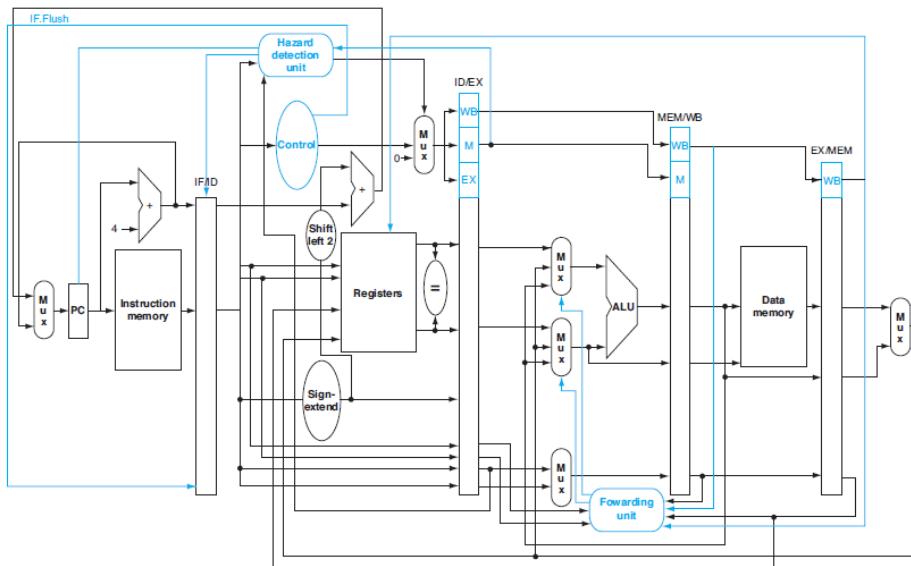
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

157

157

Descartando instruções



PATTERSON, David A.
Organização e projeto
de computadores a
interface
hardware/software

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

158

158

Exercício 1

- Explicar o que acontece na execução das instruções abaixo em ordem de entrada no pipeline do processador. Calcule quantidade de ciclos e CPI.

BNE = tomada de decisão no 3º estágio. Há adiantamento de dados. Escrita e leitura no mesmo ciclo.

1. Loop:	add \$t1, \$s3, \$s3	Ciclos	Ciclos bolha
2.	add \$t1, \$t1, \$t1	5	0
3.	add \$t1, \$t1, \$s6	1	0
4.	lw \$t0, 0(\$t1)	1	0
5.	bne \$t0, \$s5, Exit # considere existência do desvio	1	1
6.	add \$s3, \$s3, \$s4	0	1
7.	j Loop # jump - instrução de desvio incondicional	0	1
8. Exit:	sub \$s1, \$s5, \$s4	1	0

Exercício 1

- Explicar o que acontece na execução das instruções abaixo em ordem de entrada no pipeline do processador. Calcule quantidade de ciclos e CPI.

BNE = tomada de decisão no 3º estágio. Há adiantamento de dados. Escrita e leitura no mesmo ciclo.

1. Loop:	add \$t1, \$s3, \$s3	Ciclos	Ciclos bolha
2.	add \$t1, \$t1, \$t1	5	0
3.	add \$t1, \$t1, \$s6	1	0
4.	lw \$t0, 0(\$t1)	1	0
5.	bne \$t0, \$s5, Exit # considere existência do desvio	1	1
6.	add \$s3, \$s3, \$s4	0	1
7.	j Loop # jump - instrução de desvio incondicional	0	1
8. Exit:	sub \$s1, \$s5, \$s4	1	0

Ciclos: 13 , Qtd-Inst.: 6, CPI: 13/6 = 2,17

Exercício 2

- Explicar o que acontece na execução das instruções abaixo em ordem de entrada no pipeline do processador. Calcule quantidade de ciclos e CPI. :

BEQ= tomada de decisão no 2º estágio. Não há adiantamento de dados. Escrita e leitura no mesmo ciclo.

		Ciclos	Ciclos bolha
1.	Loop: sub \$s2, \$s3, \$s4		
2.	lw \$s0, 4(\$s2)		
3.	lw \$s1, 0(\$s3)		
4.	slt \$t0, \$s0, \$s1		
5.	beq \$t0, \$zero, Loop # considere não há desvio		
6.	sub \$t0, \$t0, \$t0		
7.	beq \$t0, \$zero, Loop # considere que há desvio		

Exercício 2

- Explicar o que acontece na execução das instruções abaixo em ordem de entrada no pipeline do processador. Calcule quantidade de ciclos e CPI. :

BEQ= tomada de decisão no 2º estágio. Não há adiantamento de dados. Escrita e leitura no mesmo ciclo.

		Ciclos	Ciclos bolha
1.	Loop: sub \$s2, \$s3, \$s4	5	0
2.	lw \$s0, 4(\$s2)	1	2
3.	lw \$s1, 0(\$s3)	1	0
4.	slt \$t0, \$s0, \$s1	1	2
5.	beq \$t0, \$zero, Loop # considere não há desvio	1	0
6.	sub \$t0, \$t0, \$t0	1	2
7.	beq \$t0, \$zero, Loop # considere que há desvio		

Ciclos: 19, Qtd-Inst: 7 , CPI: 19/7= 2,71

Exercício da lista (fluxo 1)

	Ciclos	Ciclos bolha
• 1. lw \$t0, 0(\$t0)	5	0 / 0
• 2. lw \$t1, 0(\$t0)	1	1 / 2
• 3. beq \$t0, \$t1, Exit # desvio	1	2 / 2
• 4. sw \$t1, 0(\$t0)	0	1 / 1
• 5. lw \$t3, 0(\$t1)	0	0 / 0
• 6. Exit: sub \$t2, \$t3, \$s4	1	0 / 0
• 7. slt \$s5, \$t2, \$t3	1	0 / 2
• 8. sw \$t5, 0(\$t0)	1	0 / 0
• 9. lw \$t2, 0(\$s2)	1	0 / 0
• 10. add \$s4, \$s2, \$s1	1	0 / 0

• Ciclos: 16, Inst: 8, CPI = 16 / 8 = 2 19/8 = 2,37

Exercício da lista: fluxo 2 (decisão de desvio no 2º estágio)

CA/AS: Com adiantamento / Sem adiantamento		
• 1. lw \$t0, 0(\$t0)	• 5	0/0
• 2. lw \$t1, 0(\$t0)	• 1	1/2
• 3. sw \$t1, 0(\$t0)	• 1	0/2
• 4. lw \$t3, 0(\$t1)	• 1	0/0
• 5. slt \$s5, \$t0, \$t3	• 1	1/2
• 6. beq \$t0, \$s5, Exit # desvio	• 1	2/2
• 7. sub \$s2, \$s3, \$s1	• 0	1/1
• 8. lw \$t2, 0(\$s2)	• 0	0/0
• 9. Exit: sub \$t2, \$t3, \$s4	• 1	0/0

• Ciclos 16 e 20, Inst. 7; CPI= 16/7 = 2,28
CPI= 20/7 = 2,85

Arquitetura de Computadores III

Pipeline Superescalar de Instruções

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

165

165

Superescalaridade

- 1. Introdução
- 2. Despacho em ordem, terminação em ordem
- 3. Despacho em ordem, terminação fora-de-ordem
- 4. Despacho fora-de-ordem, terminação fora-de-ordem
- 5. Janela de instruções centralizada
- 6. Janela de instruções distribuída
- 7. Exemplo
- 8. Renomeação de registradores

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

166

166

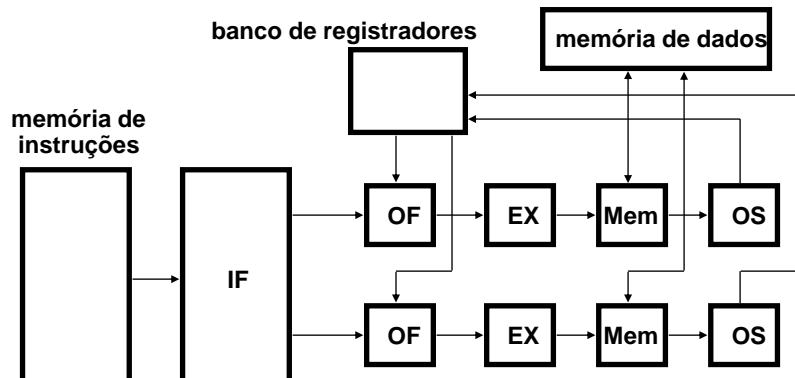
Introdução

- princípios da super-escalaridade
 - várias unidades de execução
 - várias instruções completadas simultaneamente em cada ciclo de relógio
- hardware é responsável pela extração de paralelismo
- na prática, obtém-se IPC pouco maior do que 2
 - limitação do paralelismo intrínseco dos programas
- problemas com a execução simultânea de instruções
 - conflitos de acesso a recursos comuns
 - memória
 - dependências de dados
 - verdadeiras
 - falsas - anti-dependências, dependências de saída
 - dependências de controle (desvios)

Introdução

- pipelines ou unidades funcionais podem operar com velocidades variáveis – latências
- término das instruções pode não seguir a seqüência estabelecida no programa
- processador com capacidade de “look-ahead”
 - se há conflito que impede execução da instrução atual, processador
 - examina instruções além do ponto atual do programa
 - procura instruções que sejam independentes
 - executa estas instruções
- possibilidade de execução fora de ordem
 - cuidado para manter a correção dos resultados do programa

Processador com 2 pipelines



exemplo: Pentium I

cache de instruções precisa fornecer dobro de instruções por ciclo

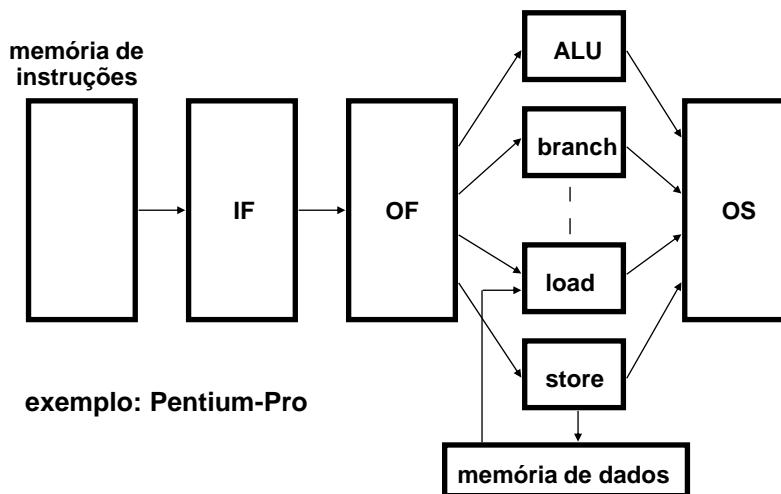
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

169

169

Unidades de execução especializadas



exemplo: Pentium-Pro

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

170

170

Despacho e terminação de instruções

- despacho de instruções
 - refere-se ao fornecimento de instruções para as unidades funcionais
- terminação de instruções
 - refere-se à escrita de resultados (em registradores, no caso de processadores RISC)
- alternativas
 - despacho em ordem, terminação em ordem
 - despacho em ordem, terminação fora de ordem
 - despacho fora de ordem, terminação fora de ordem

Despacho em ordem, terminação em ordem

- despacho de novas instruções só é feito quando instruções anteriormente despachadas já foram executadas
- despacho é congelado ...
 - quando existe conflito por unidade funcional
 - quando unidade funcional exige mais de um ciclo para gerar resultado
- exemplo, supondo processador que pode a cada ciclo ...
 - decodificar 2 instruções
 - executar até 3 instruções em 3 unidades funcionais distintas
 - escrever resultados de 2 instruções

Despacho em ordem, terminação em ordem

decodificação		execução		write-back		ciclo
I1	I2					1
I3	I4	I1				2
I3	I4	I1				3
	I4		I3			4
I5	I6			I1	I2	5
	I6		I4	I3		6
		I5			I4	7
		I6			I5	8
					I6	

restrições:

- fase de execução de I1 exige 2 ciclos
- I3 e I4 precisam da mesma unidade funcional
- I5 e I6 precisam da mesma unidade funcional
- I5 depende do valor produzido por I4

6 instruções em
6 ciclos
IPC = 1.0

Despacho em ordem, terminação fora de ordem

- despacho não espera que instruções anteriores já tenham sido executadas
 - ou seja: despacho não é congelado quando unidades funcionais levam mais de um ciclo para executar instrução
- consequência: uma unidade funcional pode completar uma instrução após instruções subsequentes já terem sido completadas
- despacho ainda precisa ser congelado quando ...
 - há conflito por uma unidade funcional
 - há uma dependência de dados verdadeira

Despacho em ordem, terminação fora de ordem

decodificação		execução			write-back		ciclo
I1	I2						1
I3	I4	I1	I2	I3			2
	I4			I4			3
I5	I6			I5			4
	I6			I6			5
							6
							7

6 instruções em
5 ciclos
IPC = 1.2

notar:

- I1 termina fora de ordem em relação a I2
- I3 é executada concorrentemente com último ciclo de execução de I1
- tempo total reduzido para 7 ciclos

Despacho em ordem, terminação fora de ordem

- supondo a seguinte situação
 - R3 := R3 op R5
 - R4 := R3 + 1
 - R3 := R5 + 1
- dependência de saída
 - 1^a e 3^a instrução escrevem em R3
 - valor final de R3 deve ser o escrito pela 3^a instrução
 - atribuição da 1^a instrução não pode ser feita após atribuição da 3^a instrução
 - despacho da 3^a instrução precisa ser congelado
- terminação fora de ordem ...
 - exige controle mais complexo para testar dependências de dados
 - torna mais difícil o tratamento de interrupções

Despacho fora de ordem, terminação fora de ordem

- problemas do despacho em ordem
 - decodificação de instruções é congelada quando instrução cria ...
 - conflito de recurso
 - dependência verdadeira ou dependência de saída
 - consequência: processador não tem capacidade de look-ahead além da instrução que causou o problema, mesmo que haja instruções posteriores independentes
- solução
 - isolar estágio de decodificação do estágio de execução
 - continuar buscando e decodificando instruções, mesmo que elas não possam ser executadas imediatamente
 - inclusão de um buffer entre os estágios de decodificação e execução: janela de instruções
 - instruções são buscadas de janela independentemente de sua ordem de chegada: despacho fora de ordem

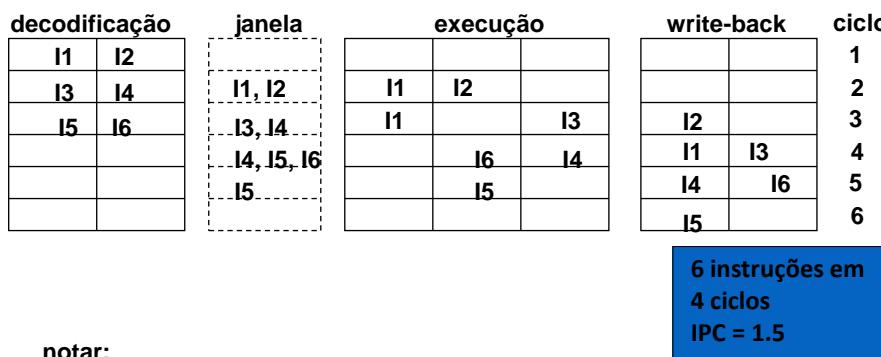
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

177

177

Despacho fora de ordem, terminação fora de ordem

**notar:**

- estágio de decodificação opera a velocidade máxima, pois independe do estágio de execução
- I6 é independente e pode ser executada fora de ordem, concorrentemente com I4
- tempo total reduzido para 6 ciclos

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

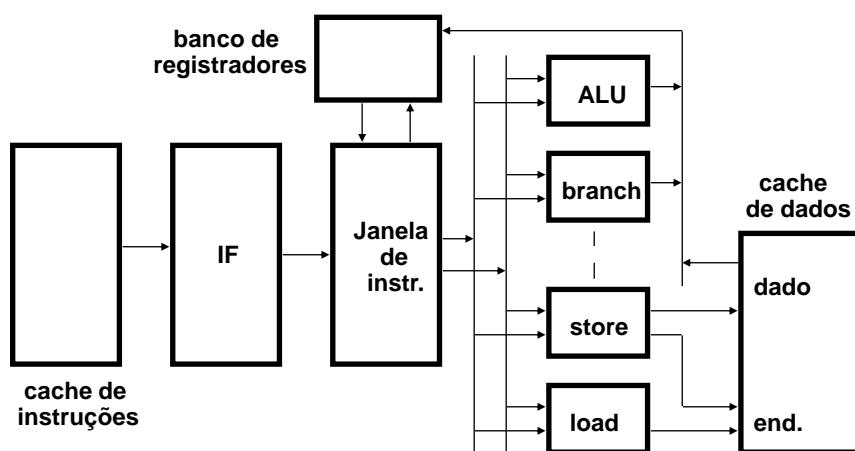
178

178

Despacho fora de ordem, terminação fora de ordem

- supondo a seguinte situação
 - $R4 := R3 + 1$
 - $R3 := R5 + 1$
- anti-dependência
 - 2^a instrução escreve em R3
 - 1^a instrução precisa ler valor de R3 antes que 2^a instrução escreva novo valor
 - despacho da 2^a instrução precisa ser congelado até que 1^a instrução tenha lido valor de R3

Janela de instruções centralizada



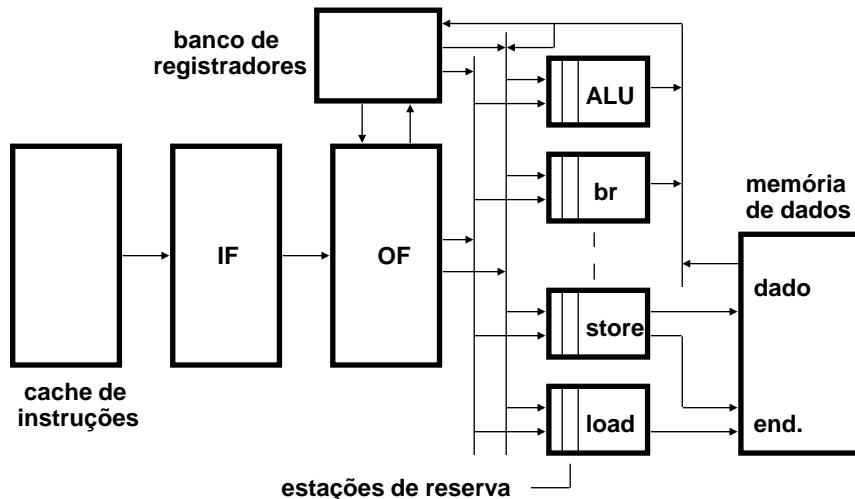
Janela de instruções centralizada

- “janela de instruções” é um buffer que armazena todas as instruções pendentes para execução
- instrução enviada para unidade de execução correspondente quando operandos estão disponíveis
 - operandos buscados no banco de registradores
- se operando não está disponível, identificador de registrador é colocado na instrução
 - quando instrução atualiza este registrador, janela de instruções é pesquisada associativamente e identificador do registrador é substituído pelo valor do operando

Janela de instruções centralizada

instr.	código	registr. destino	oper. 1	reg.1	oper. 2	reg.2
1	operação	ID	valor			ID
2	operação	ID	valor			ID
3	operação	ID		ID	valor	
4	operação	ID	valor		valor	
5	operação	ID		ID		ID

Janela de instruções distribuída



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

183

183

Janela de instruções distribuída

- cada unidade de execução tem uma “estação de reserva”
 - estação tem capacidade para armazenar 2 a 6 instruções
- instruções são decodificadas e enviadas para a estação de reserva apropriada
- instruções são enviadas para unidade de execução quando operandos estão disponíveis
- mesmo mecanismo de identificação de registradores nas instruções
- quando registradores são atualizados, valores são passados diretamente para as estações de reserva
 - busca associativa para substituição de identificadores por valores
- **Algoritmo de Tomasulo:** combinação de estações de reserva distribuídas com renomeação de registradores

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

184

184

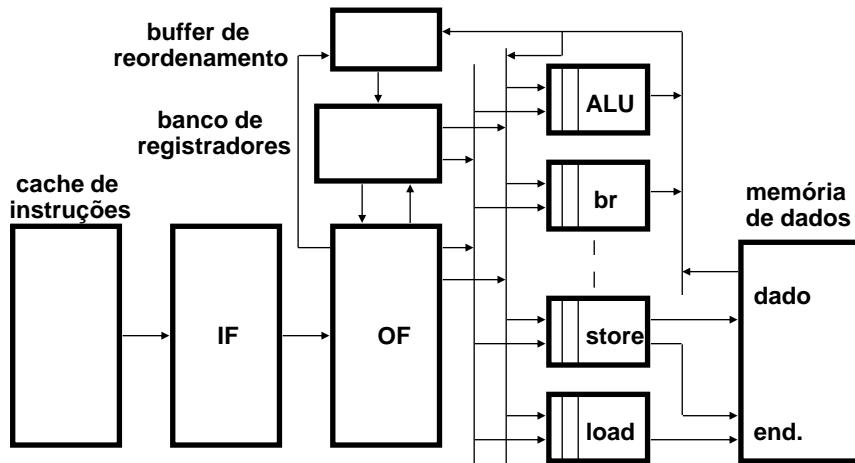
Renomeação de registradores

- antidependências e dependências de saída são causadas pela reutilização de registradores
- efeito destas dependências pode ser reduzido pelo aumento do número de registradores ou pela utilização de outros registradores disponíveis
- exemplo
 - ADD R1, R2, R3 ; R1 = R2 + R3
 - ADD R2, R8, 1 ; R2 = R8 + 1 antidependência em R2
 - ADD R1, R4, R5 ; R1 = R4 + R5 dependência de saída em R1
- utilizando 2 outros registradores R6 e R7 pode-se eliminar as dependências falsas
 - ADD R1, R2, R3 ; R1 = R2 + R3
 - ADD R6, R8, 1 ; R6 = R8 + 1
 - ADD R7, R4, R5 ; R7 = R4 + R5

Renomeação de registradores

- não é possível criar número ilimitado de registradores
- arquitetura deve manter compatibilidade quanto aos registradores visíveis para o programador
- solução
 - utilizar banco de registradores interno, bem maior do que o banco visível
 - renomear registradores temporariamente
 - cada registrador visível que é escrito numa instrução é renomeado para um registrador interno escolhido dinamicamente
- no exemplo anterior, supondo registradores internos Ra, Rb, Rc, Rd, Re, Rf, Rg, Rh
 - ADD Ra, Rb, Rc
 - ADD Rd, Rh, 1
 - ADD Re, Rf, Rg
- antidependência e dependência de saída foram eliminadas

Buffer de reordenamento



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

187

187

Buffer de reordenamento

- buffer é organizado como FIFO
- quando decodifica-se instrução que escreve em registrador, posição do buffer é alocada para o resultado
- cada posição do buffer contém
 - número do registrador original
 - campo para armazenamento do resultado
 - tag de renomeação
- quando resultado está disponível, valor é escrito no buffer
 - valor é simultaneamente enviado para estações de reserva e substitui tag de renomeação correspondente, se encontrado

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

188

188

Exemplo

- supondo um processador superescalar com a seguinte configuração:
 - 4 unidades funcionais - 2 somadores, 1 multiplicador, 1 load/store
 - pode executar 4 instruções por ciclo em cada estágio do pipeline
 - latências
 - somador - 1 ciclo
 - multiplicador - 2 ciclos
 - load/store - 2 ciclos
- deve ser executado o seguinte programa:
 - ADD R1, R2, R3
 - LW R10, 100 (R5)
 - ADD R5, R1, R6
 - MUL R7, R4, R8
 - ADD R2, R7, R3
 - ADD R9, R4, R10
 - ADD R11, R4, R6

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

189

189

Exemplo

ciclo	somador 1	somador 2	multiplicador	load/store
1	R1 = R2 + R3			
2				
3				

ADD R1, R2, R3	→ dependências verdadeiras
LW R10, 100 (R5)	
ADD R5, R1, R6	→ dependências falsas
MUL R7, R4, R8	
ADD R2, R7, R3	
ADD R9, R4, R10	
ADD R11, R4, R6	

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

190

190

Exemplo

ciclo	somador 1	somador 2	multiplicador	load/store
1	$R1 = R2 + R3$			$R10 = \text{mem } (R5+100)$
2				$R10 = \text{mem } (R5+100)$
3				

ADD R1, R2, R3 dependências verdadeiras
LW R10, 100 (R5)
ADD R5, R1, R6 dependências falsas
MUL R7, R4, R8
ADD R2, R7, R3
ADD R9, R4, R10
ADD R11, R4, R6

Exemplo

ciclo	somador 1	somador 2	multiplicador	load/store
1	$R1 = R2 + R3$			$R10 = \text{mem } (R5+100)$
2	$R5 = R1 + R6$			$R10 = \text{mem } (R5+100)$
3				

ADD R1, R2, R3 dependências verdadeiras
LW R10, 100 (R5)
ADD R5, R1, R6 dependências falsas
MUL R7, R4, R8
ADD R2, R7, R3
ADD R9, R4, R10
ADD R11, R4, R6

Exemplo

ciclo	somador 1	somador 2	multiplicador	load/store
1	$R1 = R2 + R3$		$R7 = R4 * R8$	$R10 = \text{mem } (R5+100)$
2	$R5 = R1 + R6$		$R7 = R4 * R8$	$R10 = \text{mem } (R5+100)$
3				

ADD R1, R2, R3 dependências verdadeiras
 LW R10, 100 (R5)
 ADD R5, R1, R6 dependências falsas
 MUL R7, R4, R8
 ADD R2, R7, R3
 ADD R9, R4, R10
 ADD R11, R4, R6

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

193

193

Exemplo

ciclo	somador 1	somador 2	multiplicador	load/store
1	$R1 = R2 + R3$		$R7 = R4 * R8$	$R10 = \text{mem } (R5+100)$
2	$R5 = R1 + R6$		$R7 = R4 * R8$	$R10 = \text{mem } (R5+100)$
3		$R2 = R7 + R3$		

ADD R1, R2, R3 dependências verdadeiras
 LW R10, 100 (R5)
 ADD R5, R1, R6 dependências falsas
 MUL R7, R4, R8
 ADD R2, R7, R3
 ADD R9, R4, R10
 ADD R11, R4, R6

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

194

194

Exemplo

ciclo	somador 1	somador 2	multiplicador	load/store
1	$R1 = R2 + R3$		$R7 = R4 * R8$	$R10 = \text{mem } (R5+100)$
2	$R5 = R1 + R6$		$R7 = R4 * R8$	$R10 = \text{mem } (R5+100)$
3	$R9 = R4 + R10$	$R2 = R7 + R3$		

ADD R1, R2, R3 dependências verdadeiras
 LW R10, 100 (R5) dependências falsas
 ADD R5, R1, R6
 MUL R7, R4, R8
 ADD R2, R7, R3
 ADD R9, R4, R10
 ADD R11, R4, R6

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

195

195

Exemplo

ciclo	somador 1	somador 2	multiplicador	load/store
1	$R1 = R2 + R3$	$R11 = R4 + R6$	$R7 = R4 * R8$	$R10 = \text{mem } (R5+100)$
2	$R5 = R1 + R6$		$R7 = R4 * R8$	$R10 = \text{mem } (R5+100)$
3	$R9 = R4 + R10$	$R2 = R7 + R3$		

ADD R1, R2, R3 dependências verdadeiras
 LW R10, 100 (R5) dependências falsas
 ADD R5, R1, R6
 MUL R7, R4, R8
 ADD R2, R7, R3
 ADD R9, R4, R10
 ADD R11, R4, R6

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

196

196

Exemplo

ciclo	somador 1	somador 2	multiplicador	load/store
1	$R1 = R2 + R3$	$R11 = R4 + R6$	$R7 = R4 * R8$	$R10 = \text{mem } (R5+100)$
2	$RA = R1 + R6$		$R7 = R4 * R8$	$R10 = \text{mem } (R5+100)$
3	$R9 = R4 + R10$	$RB = R7 + R3$		

ADD R1, R2, R3
LW R10, 100 (R5)
ADD RA, R1, R6
MUL R7, R4, R8
ADD RB, R7, R3
SUB R22, R23, R2
ADD R9, R4, R10
ADD R11, R4, R6

dependências verdadeiras
 dependências falsas

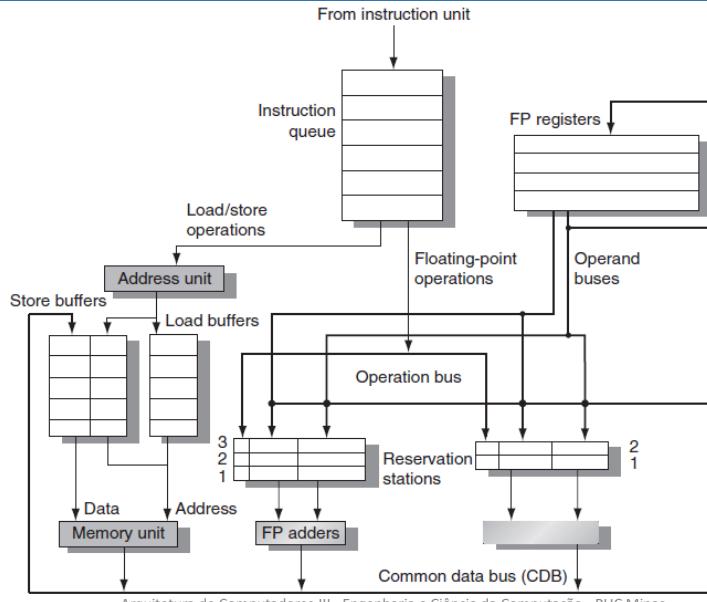
Exemplo

ciclo	somador 1	somador 2	multiplicador	load/store
1	$R1 = R2 + R3$	$R11 = R4 + R6$	$R7 = R4 * R8$	$R10 = \text{mem } (R5+100)$
2	$RA = R1 + R6$		$R7 = R4 * R8$	$R10 = \text{mem } (R5+100)$
3	$R9 = R4 + R10$	$RB = R7 + R3$		

ADD R1, R2, R3
LW R10, 100 (R5)
ADD RA, R1, R6
MUL R7, R4, R8
ADD RB, R7, R3
SUB R22, R23, RB
ADD R9, R4, R10
ADD R11, R4, R6

dependências verdadeiras
 dependências falsas

Algoritmo de Tomasulo – Ponto flutuante no MIPS



HENNESSY, John L.
Arquitetura de
computadores uma
abordagem
quantitativa

2024

199

199

Instruction status							
Instruction	Issue		Execute		Write result		
L.D F6,32(R2)			✓		✓		
L.D F2,44(R3)			✓		✓		
MUL.D F0,F2,F4			✓				
SUB.D F8,F2,F6			✓				
DIV.D F10,F0,F6			✓				
ADD.D F6,F8,F2			✓				

Reservation stations							
Name	Busy	Op	V _j	V _k	Q _j	Q _k	A
Load1	No						
Load2	Yes	Load					44 + Regs[R3]
Add1	Yes	SUB		Mem[32 + Regs[R2]]		Load2	
Add2	Yes	ADD				Add1	Load2
Add3	No						
Mult1	Yes	MUL		Regs[F4]		Load2	
Mult2	Yes	DIV		Mem[32 + Regs[R2]]		Mult1	

Register status								
Field	F0	F2	F4	F6	F8	F10	F12	... F30
Qi	Mult1	Load2	Computadores III - Engenharia e Ciéncia da Computação	Add1	Add2	Mult1	Mult2	

HENNESSY, John L.
Arquitetura de
computadores uma
abordagem
quantitativa

2024

200

200

Instruction status							
Instruction		Issue	Execute	Write result			
L.D	F6,32(R2)	✓	✓	✓			
L.D	F2,44(R3)	✓	✓				
MUL.D	F0,F2,F4	✓					
SUB.D	F8,F2,F6	✓					
DIV.D		✓					
ADD.D		✓					

Reservation stations							
Name	Busy	Op	Vj	Vk	Qj	Qk	A
Load1	No						
Load2	Yes	Load					44 + Regs[R3]
Add1	Yes	SUB		Mem[32 + Regs[R2]]	Load2		
Add2	Yes	ADD			Add1	Load2	
Add3	No						
Mult1	Yes	MUL		Regs[F4]		Load2	
Mult2	Yes	DIV		Mem[32 + Regs[R2]]		Mult1	

Register status								
Field	F0	F2	F4	F6	F8	F10	F12	... F30
Qi	Mult1	Load2	Add2	Add1	Add1	Load2	Mult1	PUC Minas

HENNESSY, John L.
Arquitetura de
computadores uma
abordagem
quantitativa

2024

201

201

Instruction status							
Instruction		Issue	Execute	Write result			
L.D	F6,32(R2)	✓	✓	✓			
L.D	F2,44(R3)	✓	✓				
MUL.D	F0,F2,F4	✓					
SUB.D	F8,F2,F6	✓					
DIV.D		✓					
ADD.D		✓					

Reservation stations							
Name	Busy	Op	Vj	Vk	Qj	Qk	A
Load1	No						
Load2	Yes	Load					44 + Regs[R3]
Add1	Yes	SUB		Mem[32 + Regs[R2]]	Load2		
Add2	Yes	ADD			Add1	Load2	
Add3	No						
Mult1	Yes	MUL		Regs[F4]		Load2	
Mult2	Yes	DIV		Mem[32 + Regs[R2]]		Mult1	

Register status								
Field	F0	F2	F4	F6	F8	F10	F12	... F30
Qi	Mult1	Load2	Add2	Add1	Add1	Load2	Mult1	PUC Minas

HENNESSY, John L.
Arquitetura de
computadores uma
abordagem
quantitativa

2024

202

202

Instruction status							
Instruction		Issue	Execute	Write result			
L.D	F6,32(R2)	✓	✓	✓			
L.D	F2,44(R3)	✓	✓				
MUL.D	F0,F2,F4	✓					
SUB.D	F8,F2,F6	✓					
DIV.D	F10,F0,F6	✓					
ADD.D	F6,F8,F2	✓					

Reservation stations							
Name	Busy	Op	Vj	Vk	Qj	Qk	A
Load1	No						
Load2	Yes	Load					44 + Regs[R3]
Add1	Yes	SUB		Mem[32 + Regs[R2]]	Load2		
Add2	Yes	ADD			Add1	Load2	
Add3	No						
Mult1	Yes	MUL		Regs[F4]	Load2		
Mult2	Yes	DIV		Mem[32 + Regs[R2]]	Mult1		

Register status							
Field	F0	F2	F4	F6	F8	F10	F12
Qi	Mult1	Load2	Add2	Add1	Add1	Mult1	Mult2

HENNESSY, John L.
Arquitetura de
computadores uma
abordagem
quantitativa

2024

203

Estações de reserva que
produzirão o resultado!

Instruction status							
Instruction		Issue	Execute	Write result			
L.D	F6,32(R2)	✓	✓	✓			
L.D	F2,44(R3)	✓	✓				
MUL.D	F0,F2,F4	✓					
SUB.D	F8,F2,F6	✓					
DIV.D	F10,F0,F6	✓					
ADD.D	F6,F8,F2	✓					

Reservation stations							
Name	Busy	Op	Vj	Vk	Qj	Qk	A
Load1	No						
Load2	Yes	Load					44 + Regs[R3]
Add1	Yes	SUB		Mem[32 + Regs[R2]]	Load2		
Add2	Yes	ADD			Add1	Load2	
Add3	No						
Mult1	Yes	MUL		Regs[F4]	Load2		
Mult2	Yes	DIV		Mem[32 + Regs[R2]]	Mult1		

Register status							
Field	F0	F2	F4	F6	F8	F10	F12
Qi	Mult1	Load2	Add2	Add1	Add1	Mult1	Mult2

HENNESSY, John L.
Arquitetura de
computadores uma
abordagem
quantitativa

2024

204

Valor dos operandos fonte!

Reservation stations							
Name	Busy	Op	Vj	Vk	Qj	Qk	A
Load1	No						
Load2	Yes	Load					44 + Regs[R3]
Add1	Yes	SUB		Mem[32 + Regs[R2]]	Load2		
Add2	Yes	ADD			Add1	Load2	
Add3	No						
Mult1	Yes	MUL		Regs[F4]	Load2		
Mult2	Yes	DIV		Mem[32 + Regs[R2]]	Mult1		

Register status							
Field	F0	F2	F4	F6	F8	F10	F12
Qi	Mult1	Load2	Add2	Add1	Add1	Mult1	Mult2

204

Instruction status							
Instruction		Issue	Execute	Write result			
L.D	F6,32(R2)	✓	✓	✓			
L.D	F2,44(R3)	✓	✓				
MUL.D	F0,F2,F4	✓					
SUB.D	F8,F2,F6	✓					
DIV.D	F10,F0,F6	✓					
ADD.D	F6,F8,F2	✓					

Reservation stations							
Name	Busy	Op	Vj	Vk	Qj	Qk	A
Load1	No						
Load2	Yes	Load					44 + Regs[R3]
Add1	Yes	SUB		Mem[32 + Regs[R2]]	Load2		
Add2	Yes	ADD			Add1	Load2	
Add3	No						
Mult1	Yes	MUL		Regs[F4]		Load2	
Mult2	Yes	DIV		Mem[32 + Regs[R2]]	Mult1		

Register status								
Field	F0	F2	F4	F6	F8	F10	F12	... F30
Qi	Mult1	Load2	Add2	Add1	Add1	Load2	Mult1	PUC Minas

HENNESSY, John L.
Arquitetura de
computadores uma
abordagem
quantitativa

2024

205

Instruction status							
Instruction		Issue	Execute	Write result			
L.D	F6,32(R2)	✓	✓	✓			
L.D	F2,44(R3)	✓	✓				
MUL.D	F0,F2,F4	✓					
SUB.D	F8,F2,F6	✓					
DIV.D	F10,F0,F6	✓					
ADD.D	F6,F8,F2	✓					

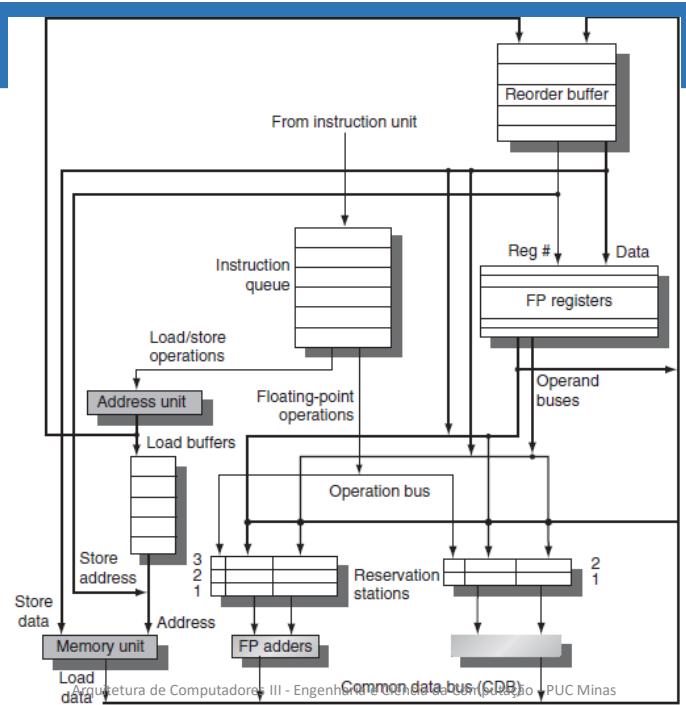
Reservation stations							
Name	Busy	Op	Vj	Vk	Qj	Qk	A
Load1	No						
Load2	Yes	Load					44 + Regs[R3]
Add1	Yes	SUB		Mem[32 + Regs[R2]]	Load2		
Add2	Yes	ADD			Add1	Load2	
Add3	No						
Mult1	Yes	MUL		Regs[F4]		Load2	
Mult2	Yes	DIV		Mem[32 + Regs[R2]]	Mult1		

Register status								
Field	F0	F2	F4	F6	F8	F10	F12	... F30
Qi	Mult1	Load2	Add2	Add1	Add1	Load2	Mult1	PUC Minas

HENNESSY, John L.
Arquitetura de
computadores uma
abordagem
quantitativa

2024

206



HENNESSY, John L.
Arquitetura de
computadores uma
abordagem
quantitativa

2024

207

207

Reorder buffer						
Entry	Busy	Instruction	State	Destination	Value	
1	No	L.D	F6,32(R2)	Commit	F6	Mem[32 + Regs[R2]]
2	No	L.D	F2,44(R3)	Commit	F2	Mem[44 + Regs[R3]]
3	Yes	MUL.D	F0,F2,F4	Write result	F0	#2 × Regs[F4]
4	Yes	SUB.D	F8,F2,F6	Write result	F8	#2 – #1
5	Yes	DIV.D	F10,F0,F6	Execute	F10	
6	Yes	ADD.D	F6,F8,F2	Write result	F6	#4 + #2

Reservation stations							
Name	Busy	Op	Vj	Vk	Qj	Qk	Dest A
Load1	No						
Load2	No						
Add1	No						
Add2	No						
Add3	No						
Mult1	No	MUL.D	Mem[44 + Regs[R3]]	Regs[F4]			#3
Mult2	Yes	DIV.D		Mem[32 + Regs[R2]]	#3		#5

FP register status										
Field	F0	F1	F2	F3	F4	F5	F6	F7	F8	F10
Reorder #	3						6		4	5
Busy	Yes	No	No	No	No	No	Yes		Yes	Yes

HENNESSY, John L.
Arquitetura de
computadores uma
abordagem
quantitativa

2024

208

208

A interface hardware / software

Arquitetura de Computadores III

Arquitetura de Suporte Multithreading

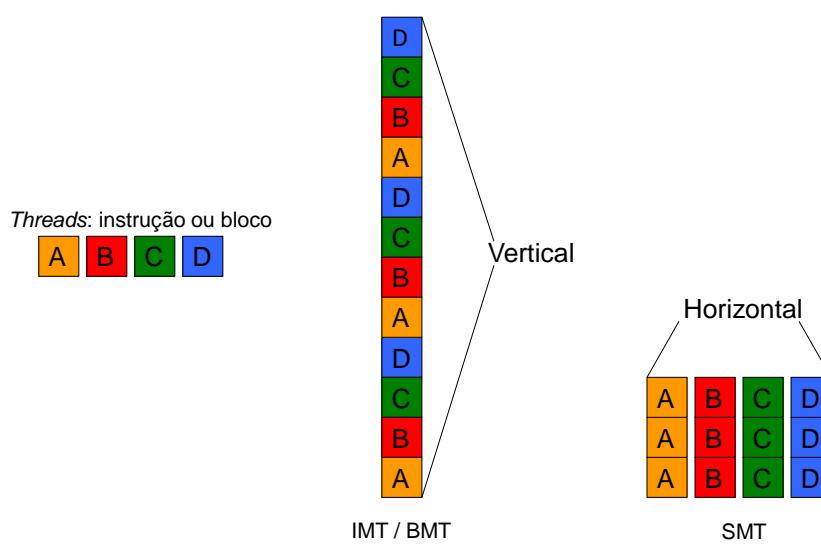
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

209

209

Suporte a múltiplas threads



2024

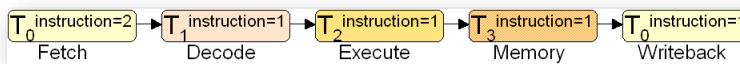
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

210

210

Suporte a múltiplas threads

Quantas instruções são executadas simultaneamente?



Qual o impacto na arquitetura?

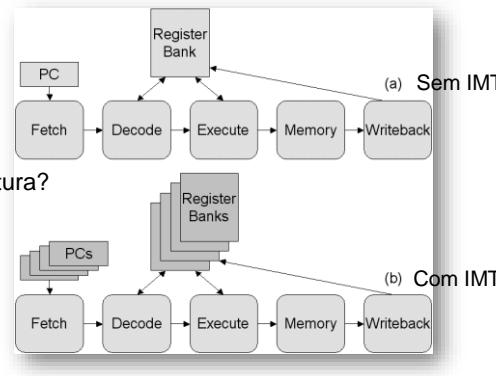
FREITAS, H. C.; MADRUGA, F. L.; ALVES, M. A. Z.; NAVAUX, P. O. A. Design of Interleaved Multithreading for Network Processors on Chip, IEEE International Symposium on Circuits and Systems, ISCAS, Taipei, p. 2213-2216, 2009

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

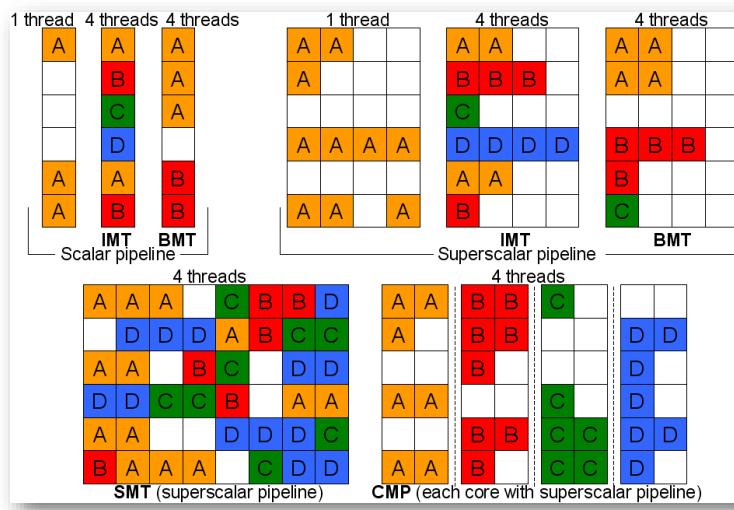
211

211



Suporte a múltiplas threads

CMT: Chip Multithreading



FREITAS, H. C.; MADRUGA, F. L.; ALVES, M. A. Z.; NAVAUX, P. O. A. Design of Interleaved Multithreading for Network Processors on Chip, IEEE International Symposium on Circuits and Systems, ISCAS, Taipei, p. 2213-2216, 2009

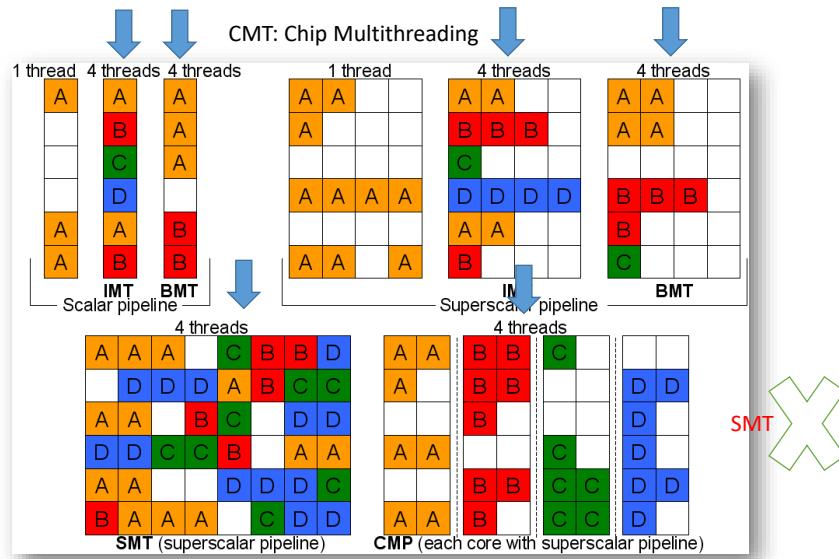
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

212

212

Suporte a múltiplas threads



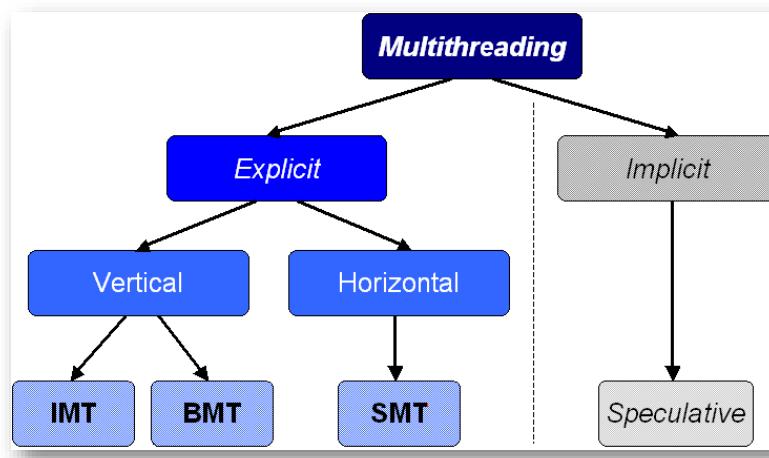
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

213

213

Suporte a múltiplas threads



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

214

214

Suporte a múltiplas threads (foco no SMT)

- Benefícios:
 - CPI => IPC (Superescalar).
 - Vazão de instruções (Superescalar) => Vazão de threads (SMT).
 - Ilusão de mais de um núcleo de processamento.
 - Não existe o esvaziamento de pipeline comum no BMT.
 - Não há atraso na execução de threads, comum no IMT/BMT.
- Desafios / Problemas:
 - Tamanho da arquitetura.
 - Banco de registradores muito grande para guardar vários contextos.
 - Divisão de recursos e equilíbrio de desempenho.
 - Conflitos de cache sem degradação de desempenho.

2024

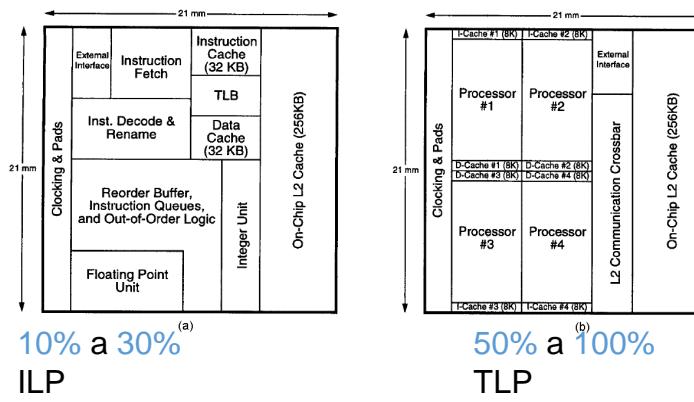
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

215

215

Processadores multicore

- (a) Superescalaridade de seis vias de execução.
 (b) Chip multicore. Cada core superescalar com duas vias de execução.



OLUKOTUN, K. et al., The Case for a Single-Chip Multiprocessor, 7th International Conference on Architectural Support for Programming Languages and Operating Systems, p. 2-11, 1996.

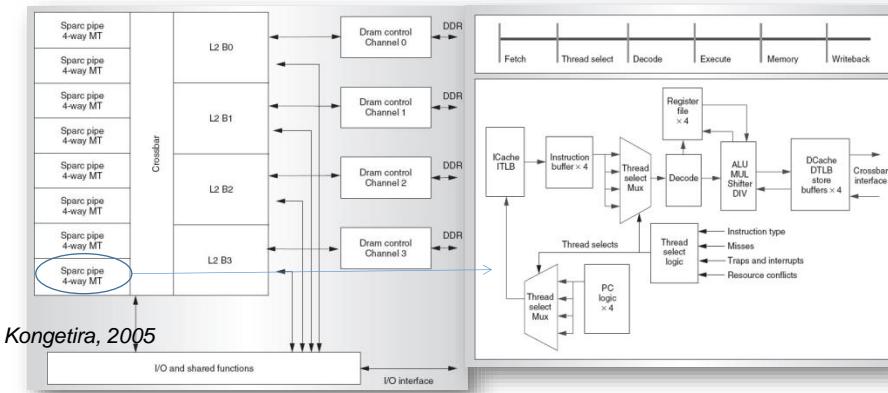
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

216

216

Processadores multicore

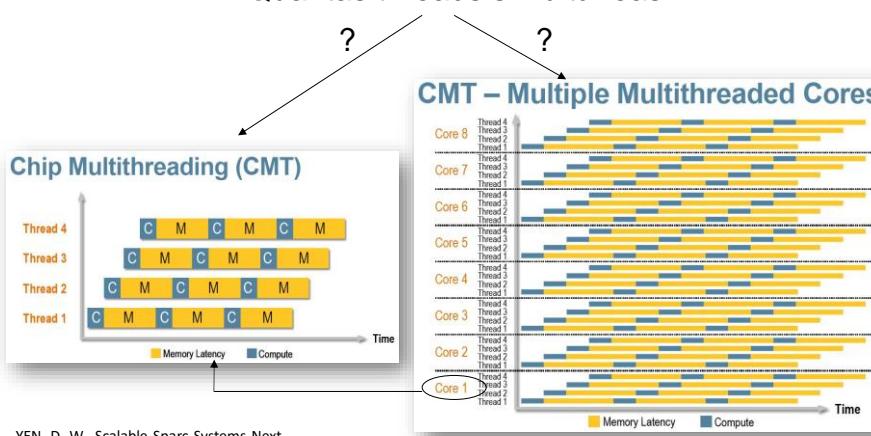


- Suporte a 4 threads IMT por núcleo (32 threads ativas, 8 threads simultâneas).
- Crossbar switch de 134,4 GB/s.
- 4 canais DDR 23GB/s.
- Potência < 80W.

KONGETIRA, P. et al., Niagara: a 32-way multithreaded Sparc processor, IEEE MICRO, v. 25, Issue 2, p. 21-29, March-April 2005.

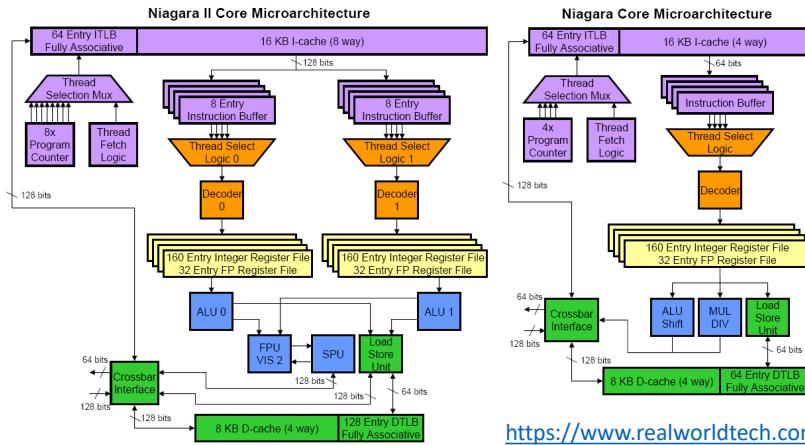
Processadores multicore

Quantas threads simultâneas?



YEN, D. W., Scalable Sparc Systems Next-Generation Computing, Scalable Systems Group, Sun Microsystems, 2005

Processadores multicore



<https://www.realworldtech.com/niagara2/2/>

- Suporta a 8 threads por núcleo (64 threads ativas, 16 threads simultâneas).
- Instruções executadas em ordem, previsão de desvio estático ou pela última decisão tomada.
- Crossbar switch de 268,8 GB/s.

2024

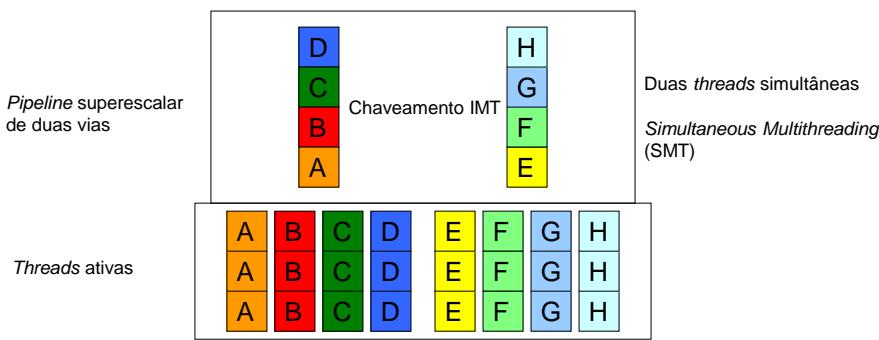
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

219

219

Processadores multicore

- SMT combinado com IMT em *chip multi-core*.
 - Superescalabilidade associada a SMT com entrelaçamento de instruções.
 - Aumenta desempenho para cargas de trabalho de propósitos gerais.
 - Reduz tamanho e complexidade da arquitetura superescalar do processador.



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

220

220

A interface hardware / software

Arquitetura de Computadores III

Arquiteturas Multicore de propósito geral (história)

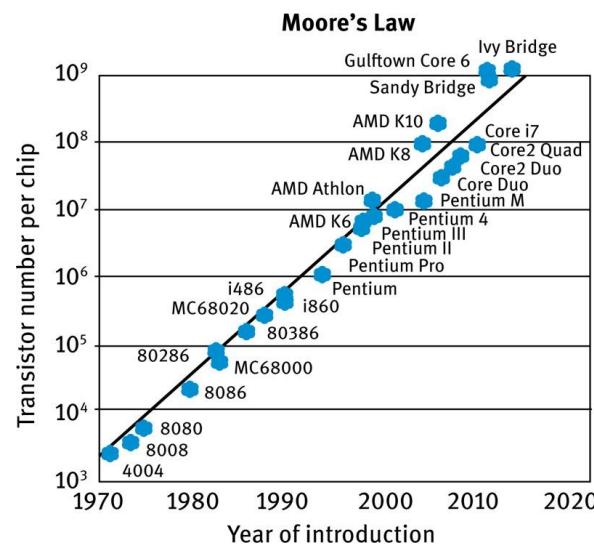
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

221

221

Lei de Moore



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

222

222

Dennard Scaling

- Há redução em área, tensão e atraso do circuito, com impacto em:
 - Aumento da frequência.
 - Redução em todas as distâncias levando a uma redução na capacitância.
 - Consequentemente, há uma redução no consumo de potência.
- Portanto, mesmo com o aumento em 2x na densidade de transistores (Lei de Moore) o consumo de potência permanece o mesmo.
- **No entanto**, Dennard ignorou as correntes de fuga e limites de tensão e, portanto:
 - Com os transistores ficando menores, a potência aumenta porque não escala em função do tamanho.
 - Isso ficou conhecido como **Power Wall** que limita na prática processadores com **frequência de até 4GHz, desde 2006**.

https://en.wikipedia.org/wiki/Dennard_scaling

2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

223

223

Qual a “origem” do processador multicore?

Propósito geral

- Em uma fábrica de processadores tão tão distante...
 - O diretor para o arquiteto: Precisamos de mais desempenho!
 - O arquiteto para o diretor: Não é possível aumentar o paralelismo de instruções!
 - O diretor para o engenheiro: Precisamos de mais desempenho!
 - O engenheiro para o diretor: Não é possível aumentar a frequencia, o chip vai “queimar”!
- Quem poderá nos socorrer!?

Dennard scaling?

Se este miniconto coincidir com fatos, é pura sorte!

2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

224

224

Qual a “origem” do processador multicore?

Propósito geral

- Em uma fábrica de processadores tão tão distante...
 - Alguém diz: E a Lei de Moore!?
 - A Lei de Moore está relacionada à capacidade de integração.
 - Não está relacionada ao aumento de frequencia.
 - Está relacionada a quantidade de transistores em um mesmo espaço.
 - Se diminuirmos o tamanho dos transistores?
 - 180 nm, 130 nm, 90 nm, 65 nm, 45 nm, 32 nm, 22 nm....
 - Vamos aumentar a quantidade de processadores dentro do chip de processador!
 - Vamos chamá-los de núcleos!
 - Portanto, não adianta apenas aumentar paralelismo de instruções, nem frequência de operação!
 - Precisamos aumentar a quantidade de núcleos!

Se este miniconto coincidir com fatos, é pura sorte!

Dennard scaling?

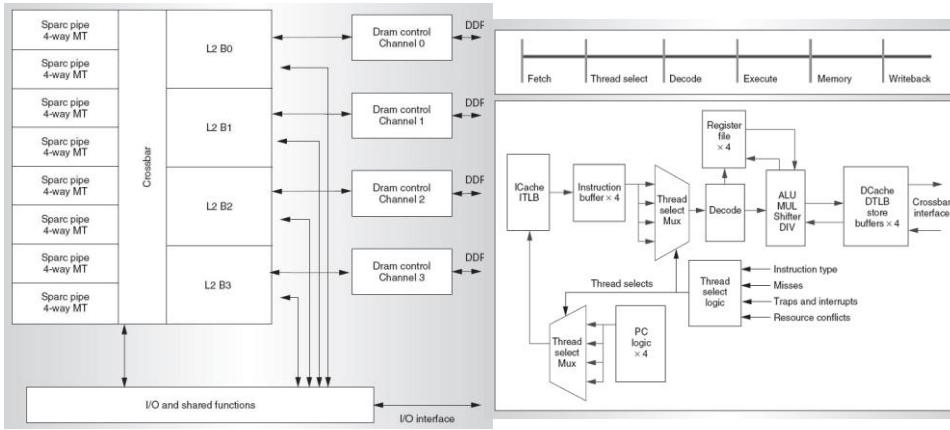
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

225

225

UltraSparc-T1 (Niagara 1)



- Suporte a 4 threads IMT por núcleo (32 threads ativas, 8 threads simultâneas).
- Crossbar switch de 134,4 GB/s.
- 4 canais DDR 23GB/s.
- Potência < 80W.

KONGETIRA, P. et al., Niagara: a 32-way multithreaded Sparc processor, IEEE MICRO, v. 25, Issue 2, p. 21-29, March-April 2005.

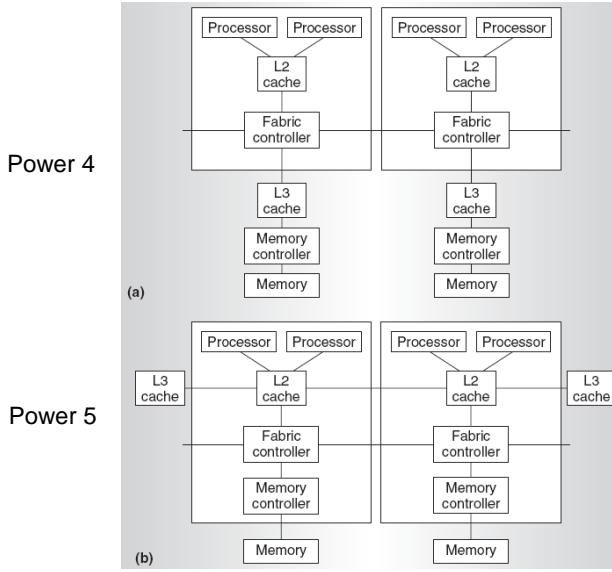
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

226

226

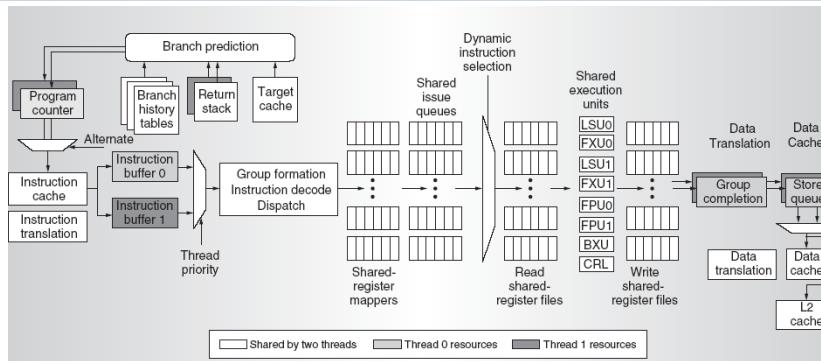
Power 4 e Power 5



- Os processadores Power 4 e Power5 possuem códigos binários e estrutura compatíveis.
- Diferença no acesso a cache L3.
- A grande diferença: o Power 5 suporta a duas threads simultâneas (SMT) por núcleo.
 - O Power5 possui dois núcleos físicos, mas quatro núcleos lógicos de processamento.

KALLA, R., SINHAROY, B., TENDLER, J. M., IBM Power5 Chip: A Dual-Core Multithreaded Processor, IEEE MICRO, v. 24, Issue 2, p. 40-47, 2004

Power 5



- A informação mais interessante no projeto do Power5 é justamente o fato do desempenho não melhorar com o suporte SMT por núcleo. Os principais motivos são:
 - O número limitado de unidades de execução que são compartilhados entre as duas threads.
 - O alto consumo da largura de banda de memória pelas duas threads.
- Esta é a principal razão para que o Power5 também suporte apenas uma thread por núcleo.

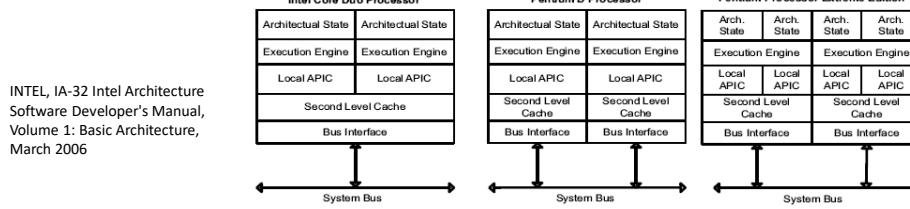
KALLA, R., SINHAROY, B., TENDLER, J. M., IBM Power5 Chip: A Dual-Core Multithreaded Processor, IEEE MICRO, v. 24, Issue 2, p. 40-47, 2004

Intel Dual Core

- Os processadores Pentium D e Core Duo, são compostos por dois núcleos, mas sem suporte a hyperthreading. Fisicamente e logicamente são dois núcleos internos.

- Diferenças básicas:

- Suporte a múltiplas threads
- Cache nível L2



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

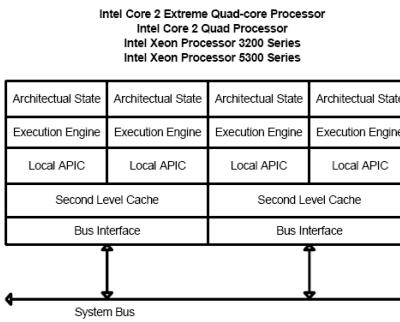
229

229

Intel Quad Core

- O projeto de um processador com 4 núcleos pela Intel é basicamente a união de dois processadores dual core.
- A cache L2 compartilhada ganhou força.
- SMT não é utilizado.

INTEL, IA-32 Intel Architecture
Software Developer's Manual,
Volume 1: Basic Architecture,
March 2006



2024

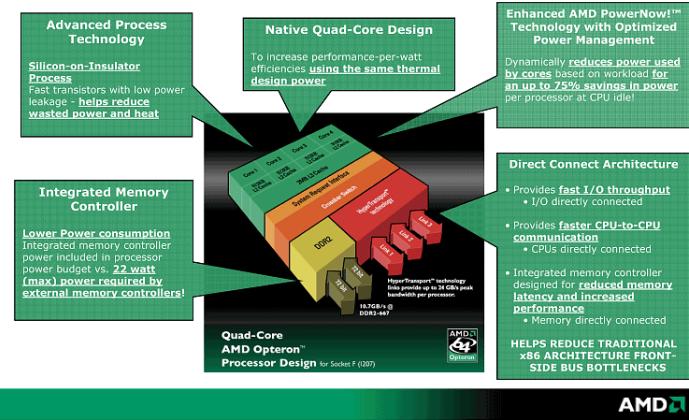
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

230

230

AMD Quad Core

Technologies Enabling Performance-Per-Watt Advantages for Quad-Core



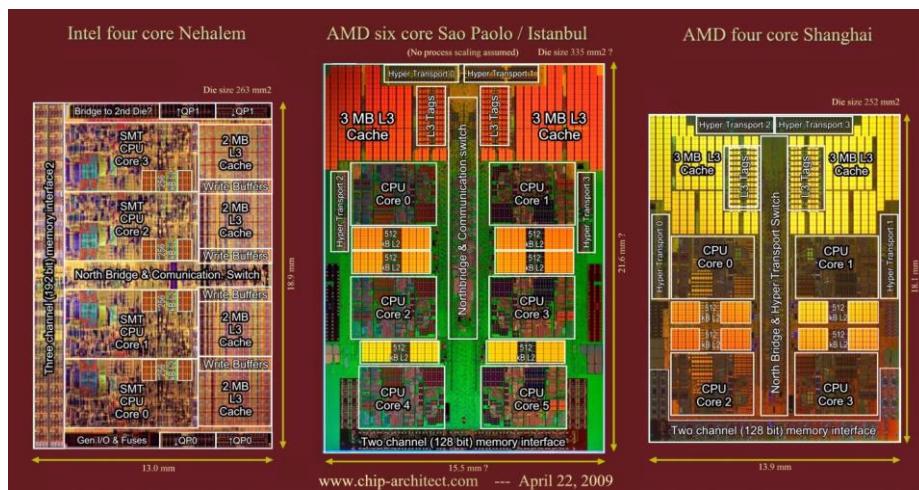
2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computaçao - PUC Minas

231

231

Nehalem, Istanbul, Shanghai



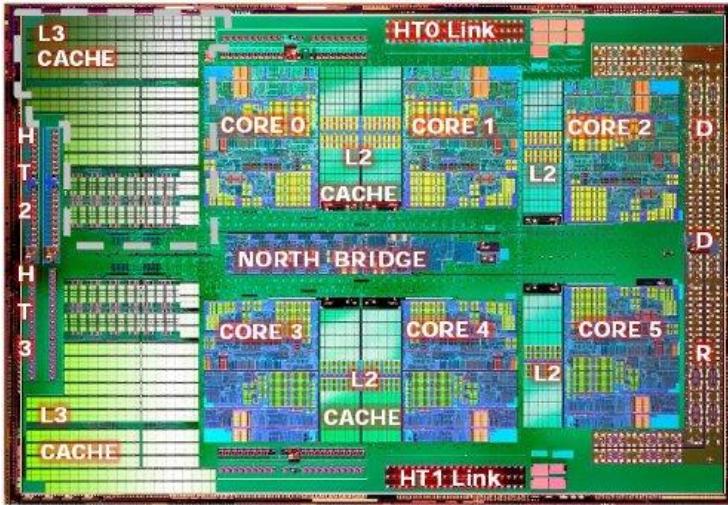
2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computaçao - PUC Minas

232

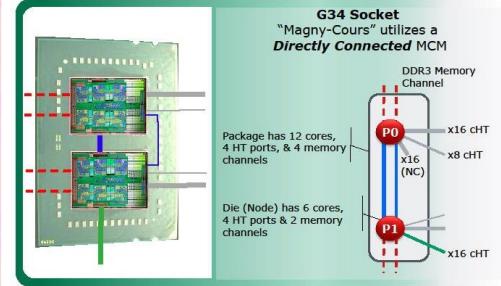
232

AMD Magny Cours (12 núcleos)



Multi-Chip Module (MCM)

MCM 2.0 Logical View



Chiplets são pequenos circuitos integrados modulares que podem ser combinados para criar um sistema em um chip (SoC) mais complexo ou um design de múltiplos núcleos. Diferentemente dos chips monolíticos tradicionais, que integram todas as funcionalidades em um único chip de silício, os chiplets dividem essas funcionalidades em núcleos menores e especializados.

<https://www.techpowerup.com/102445/amd-demos-48-core-magny-cours-system-details-architecture>

2024

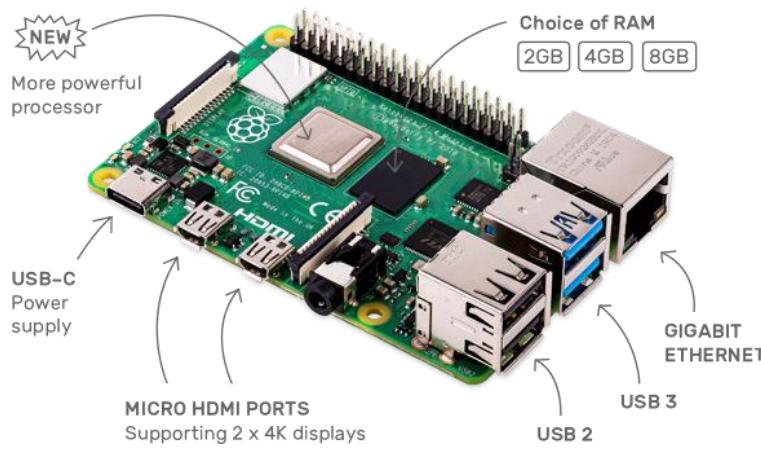
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

233

233

Arquiteturas de Processadores Multicore

Raspberry Pi 4 Model B



- Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
- 2GB, 4GB or 8GB LPDDR4-3200 SDRAM (depending on model)
- 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE
- Raspberry Pi standard 40 pin GPIO header (fully backwards compatible with previous boards)
- Micro-SD card slot for loading operating system and data storage
- Starting at \$35

<https://www.raspberrypi.org/products/raspberry-pi-4-model-b/>

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

234

234

Arquiteturas de Processadores Multicore

Parallelia Board



- **18-core** credit card sized computer
- #1 in energy efficiency @ 5W
- 16-core Epiphany RISC SOC
- Zynq SOC (FPGA + ARM A9)
- Gigabit Ethernet
- 1GB SDRAM
- Micro-SD storage
- Up to 48 GPIO pins
- HDMI, USB (optional)
- Open source design files
- Runs Linux
- Starting at \$99

<https://www.parallelia.org/>

2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computaçao - PUC Minas

235

235

Processor Intel i9



Lançamento: 2017

10 cores, 20 threads
Frequênciá base: 3,3GHz
13.75 MB L3 Cache
140 W

Dependendo do modelo a quantidade de transistors pode variar de 3 a 22 bilhões.

2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computaçao - PUC Minas

236

236

Revolução Multicore - CPU

- Xeon 2007: Quad-core CPUs
- Xeon 2008: 6-core CPUs
- Xeon 2010: 8-core CPUs
- Xeon 2013: 12-core CPUs
- Xeon 2015: 22-core CPUs
- Xeon 2017: 32-core CPUs
- Xeon 2019: 48-core CPUs

Intel Xeon Cascade Lake é um processador com 48 núcleos

Voltado a servidores de alto desempenho, processador Intel Xeon Cascade Lake AP tem até 48 núcleos e será lançado em 2019

Um mês depois de revelar os primeiros processadores Core de nona geração, a Intel anunciou novos chips para servidores: a linha Xeon Cascade Lake Advanced Performance (ou Xeon Cascade Lake AP), como foi batizada, pode contar com até 48 núcleos em uma única unidade.



<https://tecnoblog.net/noticias/2018/11/06/intel-xeon-48-nucleos/>

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

237

237

How AMD Used Its Chiplet Advantage to Design Bergamo CPU

“Bergamo” with “Zen 4c”
8 CCDs, 16 cores per CCD
Core Complex Die

Zen 4 | “Genoa” 4th Gen AMD EPYC™ CPU
Optimized for performance-per-core
12 x 8-core CCDs | Up to 96 cores

Zen 4c | “Bergamo” 4th Gen AMD EPYC™ CPU
Optimized for performance-per-watt
8 x 16-core CCDs | Up to 128 cores

The power-optimized version of the Zen 4 architecture.

Bergamo has 82 billion transistors, and supports 128 cores per socket.

<https://www.hpcwire.com/2023/06/14/how-amd-used-its-chiplet-advantage-to-design-mi300x-gpu-and-bergamo-cpu/>

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

238

238

How AMD Used Its Chiplet Advantage to Design Bergamo CPU

"Bergamo" with "Zen 4c"
8 CCDs, 16 cores per CCD

Core Complex Die

Zen 4	"Bergamo" 4 th Gen AMD EPYC® CPU
Optimized for performance-per-core 12 x 8-core CCDs Up to 96 cores	Optimized for performance-per-watt 8 x 16-core CCDs Up to 128 cores

The top-line Bergamo chip, called Epyc 9754, has 128 cores, runs 256 threads, and draws up to 360W of power. The chip has 256MB of L3 cache and runs at frequencies up to 3.10GHz. The mid-tier 9754S runs only one thread per core. The third chip in the family, the Epyc 9734 has 112 cores, runs two threads per core, and draws 320 watts of power.

<https://www.hpcwire.com/2023/06/14/how-amd-used-its-chiplet-advantage-to-design-mi300x-gpu-and-bergamo-cpu/>

The power-optimized version of the Zen 4 architecture.

Bergamo has 82 billion transistors, and supports 128 cores per socket.

2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

239

239

Neural Processing Unit (NPU)

Processador Intel® Core™ Ultra 9 285K



Intel® Gaussian & Neural Accelerator (GNA) é um bloco de acelerador de ultrabaixa potência, projetado para executar cargas de trabalho de IA centradas em áudio e fala. O Intel® GNA é projetado para executar redes neurais baseadas em áudio com ultrabaixa potência, além de aliviar simultaneamente a CPU dessa carga de trabalho.

Número de núcleos: **24**

Nº de Performance-cores: **8**

Nº de Efficient-cores: **16**

Total de threads: **24**

Frequênci turbo max: **5.7 GHz**

Frequênci base do Performance-core: **3.7 GHz**

Frequênci base do Efficient-core: **3.2 GHz**

Cache **36 MB** Intel® Smart Cache

Cache L2 total: **40 MB**

Potênci básica do processador: **125 W**

O processador Intel® Core™ Ultra 9 285K possui **4 núcleos Xe** dedicados para gráficos integrados. Esses núcleos Xe formam um total de **64 unidades de execução (EUs)**, ou seja, **512 shaders unificados**.

2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

240

240

A interface hardware / software

Arquitetura de Computadores III

Arquiteturas Manycore e Redes-em-Chip

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

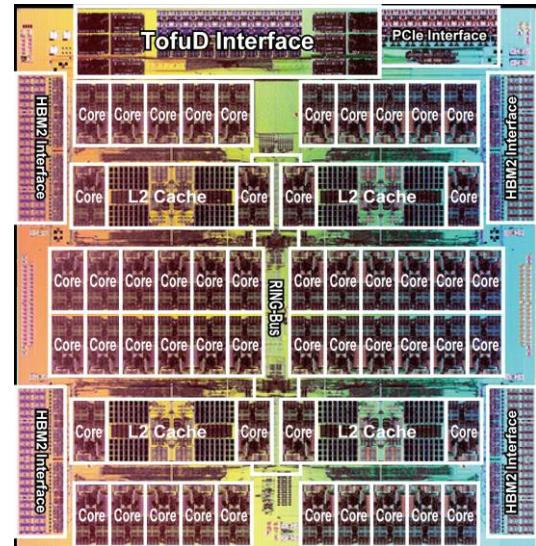
241

241

Processador Fujitsu ARM A64FX

- Total de 7.299.072 núcleos.
- Processador A64FX 48C 2.2GHz
- Arquitetura ARM projetada pela Fujitsu

<https://www.r-ccs.riken.jp/en/fugaku/project>



2024

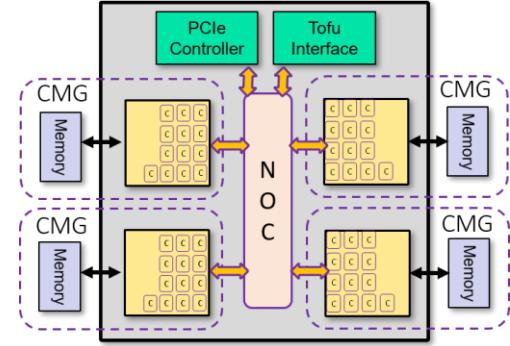
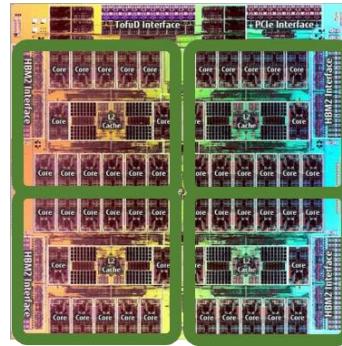
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

242

242

121

Processador Fujitsu ARM A64FX



<https://postk-web.r-ccs.riken.jp/spec.html>

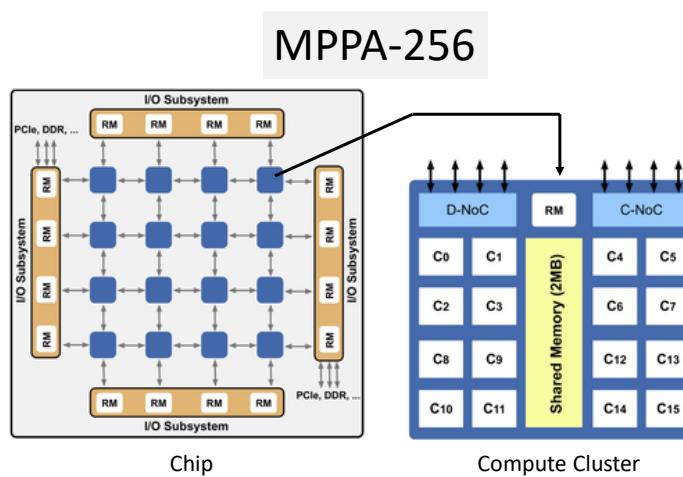
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

243

243

Arquiteturas Kalray MPPA 256



<https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.3892>

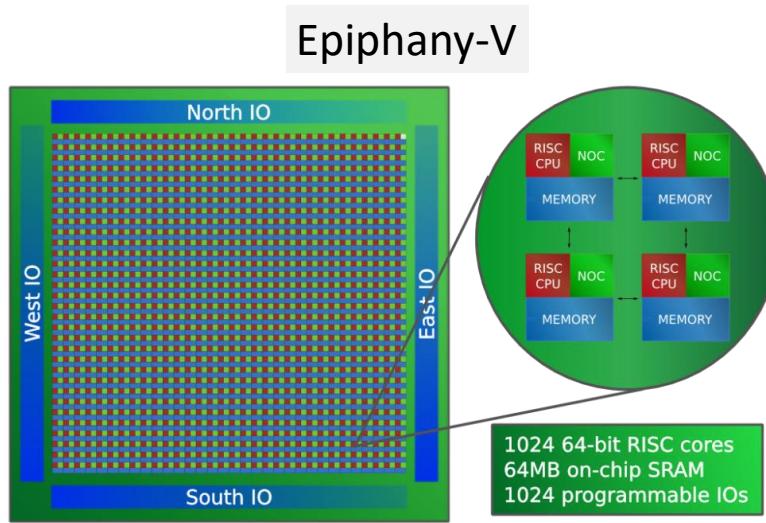
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

244

244

Arquitetura Epiphany-V



<https://www.parallel.org/2016/10/05/epiphany-v-a-1024-core-64-bit-risc-processor/>

2024

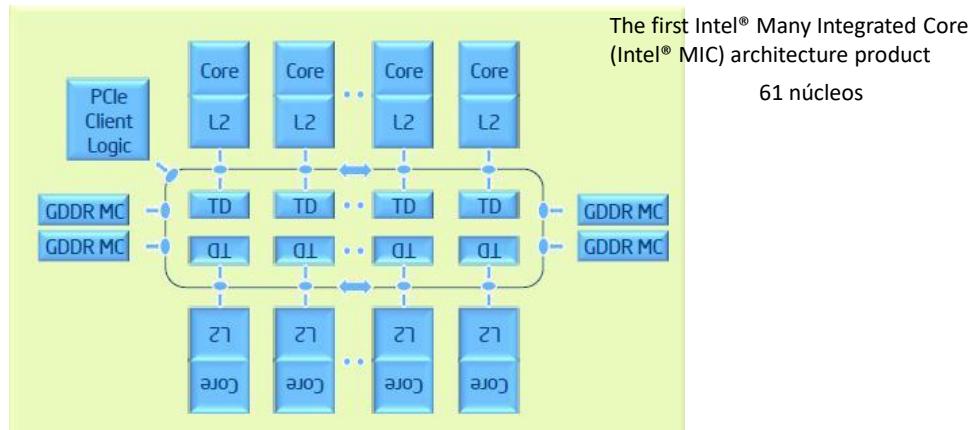
Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

245

245

Arquiteturas de Aceleradores Manycore

Intel Xeon Phi (Knights Corner)



<https://software.intel.com/content/www/us/en/develop/articles/intel-xeon-phi-coprocessor-codename-knights-corner.html>

2024

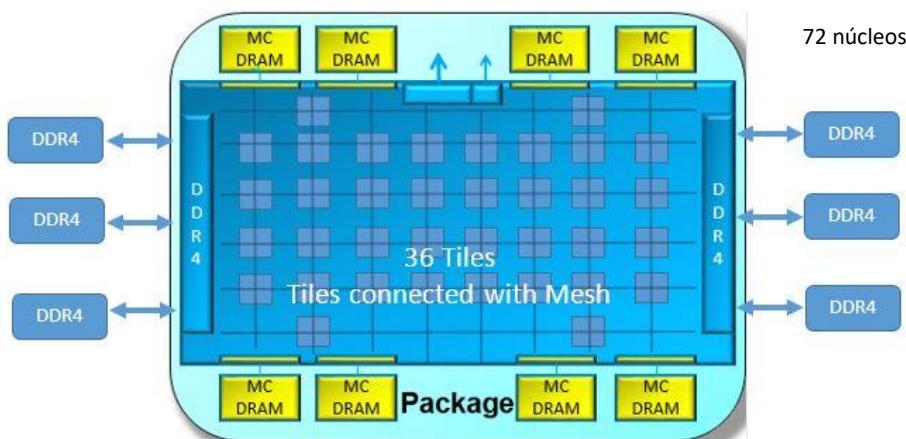
Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

246

246

Arquiteturas de Aceleradores Manycore

Intel Xeon Phi (Knights Landing)



<https://software.intel.com/content/www/us/en/develop/articles/intel-xeon-phi-x200-processor-memory-modes-and-cluster-modes-configuration-and-use-cases.html>

2024

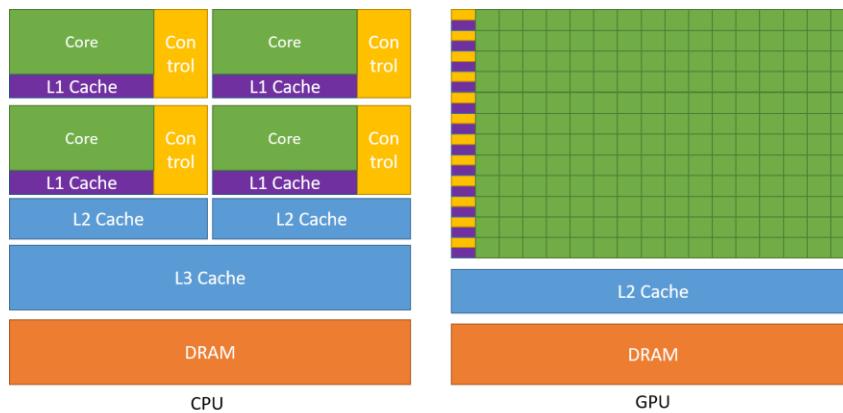
Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

247

247

Arquiteturas de Aceleradores Manycore

Graphics Processing Unit (GPU)



<https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html>

2024

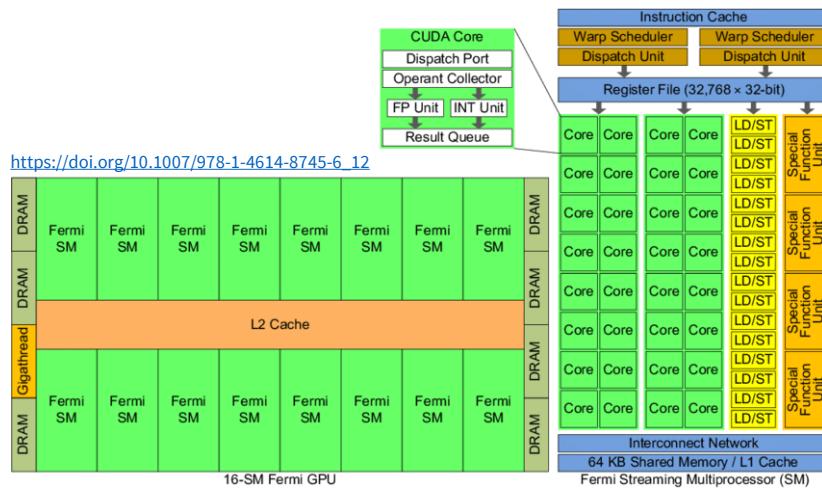
Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

248

248

Arquiteturas de Aceleradores Manycore

Graphics Processing Unit (GPU)



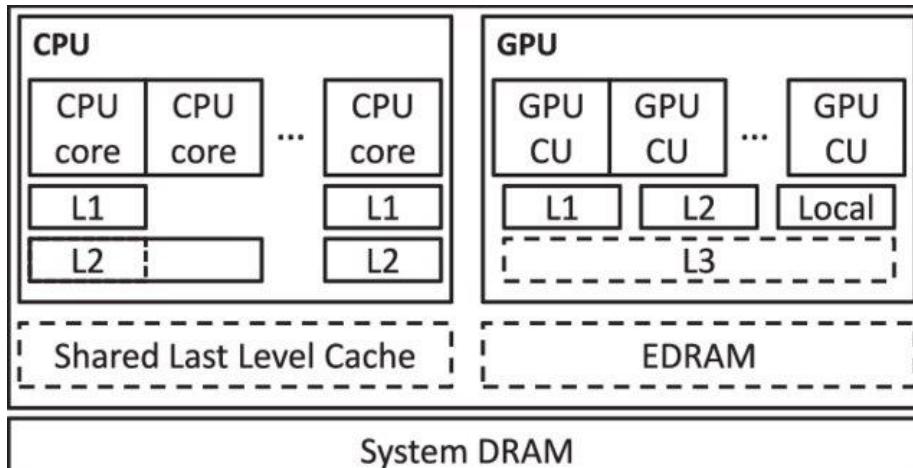
2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computaçao - PUC Minas

249

249

CPU e GPU integradas



Accelerated Processing Units
Ex: Intel Ivy Bridge and Haswell

<https://doi.org/10.1109/TPDS.2016.2586074>

2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computaçao - PUC Minas

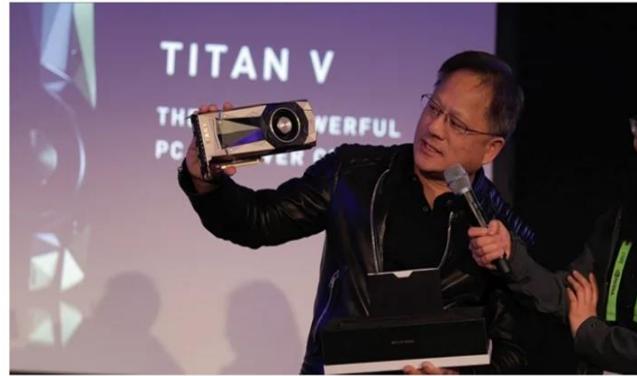
250

250

Revolução Manycore

- GTX 280 (2008) – 240 núcleos
- GTX 480 (2010) – 480 núcleos
- GTX 690 (2012) - 1536 núcleos
- GTX Titan (2014) – 2880 núcleos
- GTX Titan (2017) – 3840 núcleos
- Titan V (2019) – 5120 núcleos

Outros números generosos fazem parte do pacote. Encontramos 5.120 núcleos CUDA divididos em seis clusters, frequência de 1.200 MHz (1.455 MHz em boost), 640 núcleos tensor (usados em aprendizagem de máquina), 320 TMUs (unidades para texturização), além de TDP de 250 W.



<https://tecnoblog.net/noticias/2017/12/08/nvidia-gpu-titan-v/>

2024

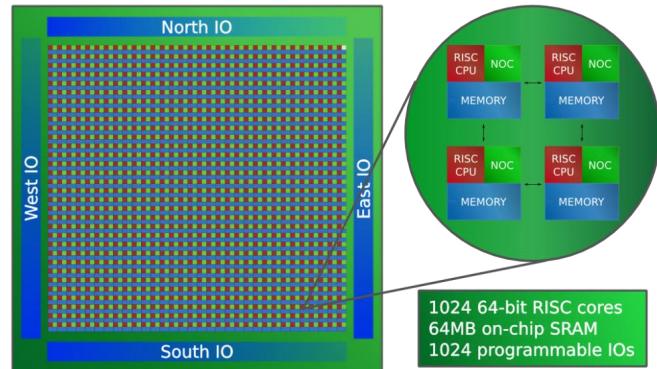
Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

251

251

Arquiteturas manycore e redes-em-chip

- Em linhas gerais, uma arquitetura *manycore* que demanda uma rede-em-chip, possui como característica o propósito geral de processamento, principalmente paralelo, com muitas comunicações coletivas. Por este motivo, é necessário ter uma rede capaz de oferecer diversos caminhos de interconexão entre os núcleos, memórias, entre outros dispositivos dentro do chip. Uma arquitetura de GPU não possui esta característica.



<https://www.parallel.org/2016/10/05/epiphany-v-a-1024-core-64-bit-risc-processor/>

2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

252

252

Antes do Manycore...

- Processador multi-core:
 - 2 núcleos: barramento
 - 4 núcleos: barramento ou chave crossbar
 - 6 núcleos: barramento ou chave crossbar
 - 8 núcleos: barramento ou chave crossbar
 - 12 núcleos: barramento ou chave crossbar
 - 16 núcleos: barramento ou chave crossbar!?
- 48 núcleos: como interconectar!?

2024

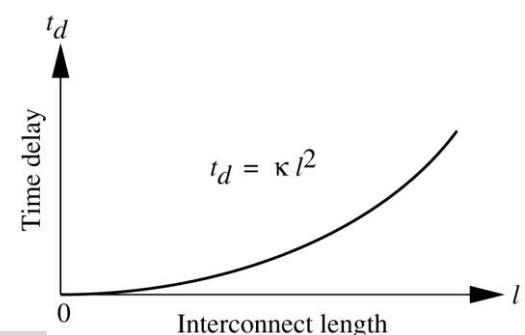
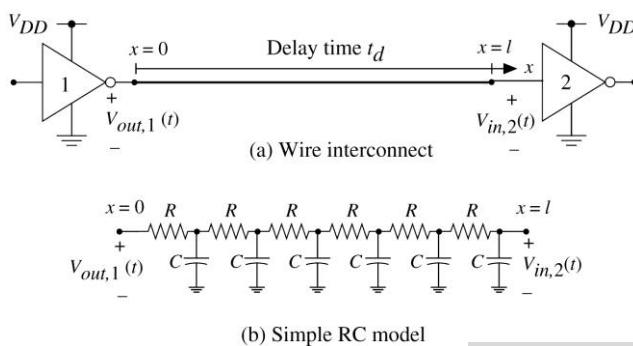
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

253

253

Processador Manycore

- O problema está no fio para interconectar os núcleos!



Uyemura, Sistemas Digitais, Uma abordagem Integrada, Thomson, 2002

2024

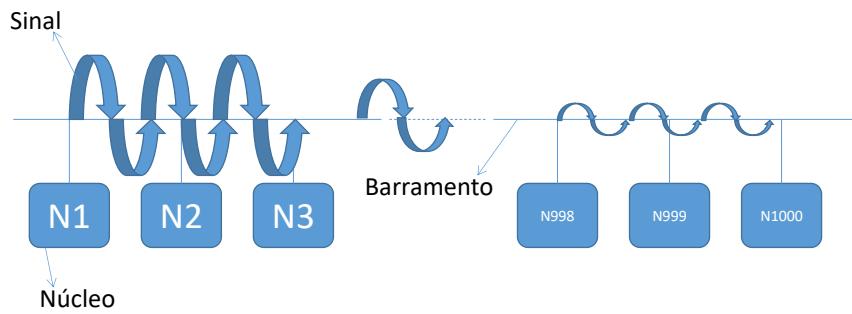
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

254

254

Processador Manycore

- O problema está no fio para interconectar os núcleos!
 - Problema 1: atenuação do sinal.
 - Perda dos dados.



2024

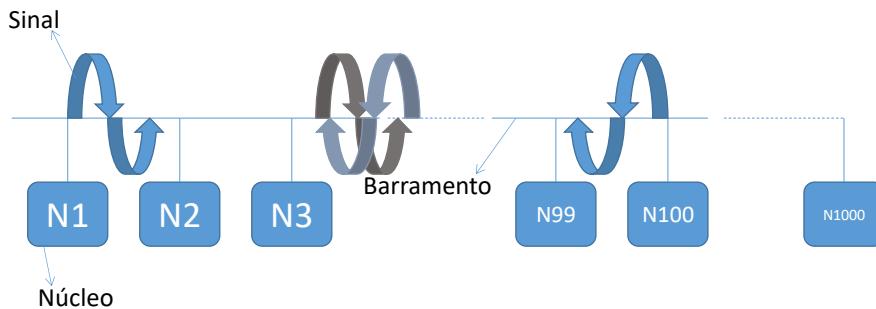
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

255

255

Processador Manycore

- O problema está no fio para interconectar os núcleos!
 - Problema 2: núcleos distantes não escutam sinal.
 - Colisão e perda de dados.



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

256

256

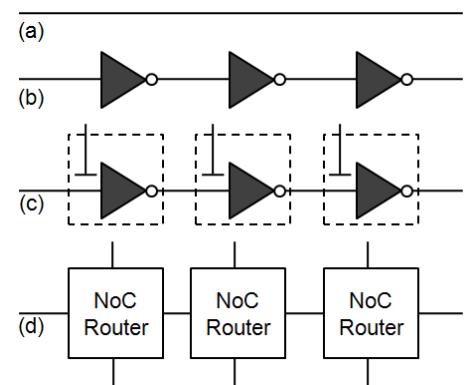
Se o problema está no fio...

- Vamos eliminar a influência do fio!
- Rede-em-Chip com fio, mas fio curto!
- Há também:
 - Rede-em-Chip sem fio!
 - Rede-em-Chip óptica!
 - Rede-em-Chip reconfigurável!

Do Barramento ao roteador

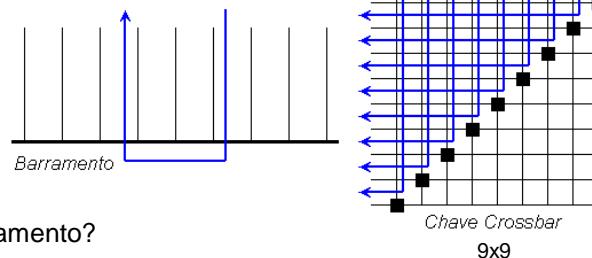
- Evolução da interconexão global para NoC.
- a) fio longo dominado pela resistência,
- b) adição de repetidores ou buffers,
- c) repetidores se tornam latches,
- d) latches evoluem para roteadores de NoC.

CIDON, I., KOLODNY, A., GINOSAR, R., The Elements of NoC, Tutorial (Slides), ACM/IEEE International Symposium on Networks-on-Chip (NOCS), San Diego, USA, 2009



Rede-em-Chip

- Interconexões não escaláveis:
 - Barramento e Chave Crossbar.
 - Por que?



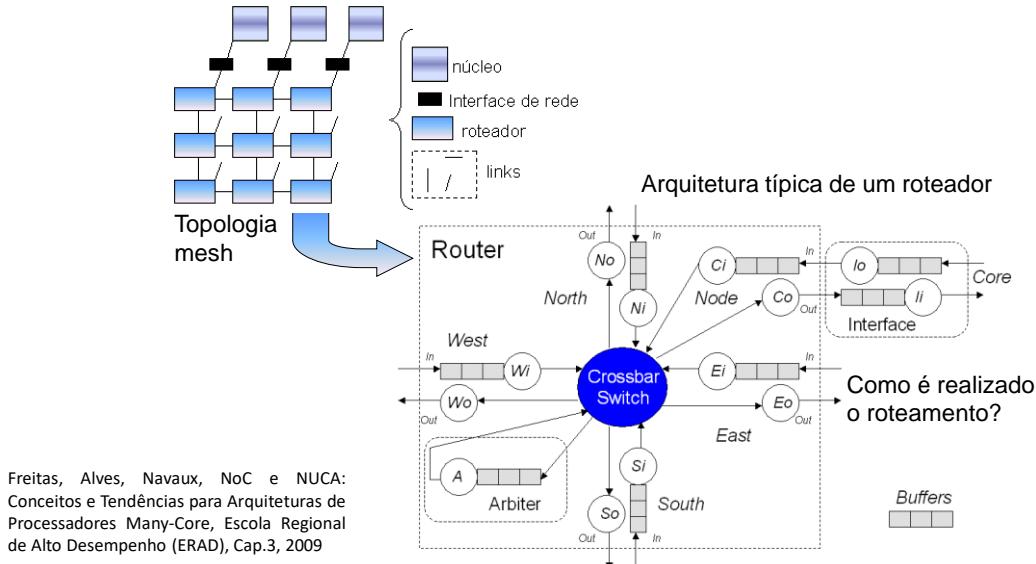
Qual a principal limitação do barramento?

Imaginem uma chave crossbar 99x99?
É viável? Por que?

Rede-em-Chip

- Principais características:
 - Composta por roteadores,
 - Possui pacotes de rede,
 - Trabalha com protocolo de roteamento,
 - Possui diversas topologias,
 - Trabalha com Qualidade-de-Serviço (QoS),
 - É tolerante a falhas,
 - É escalável!

Redes-em-Chip



2024

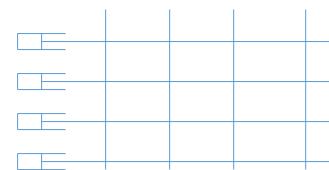
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

261

261

Tipos de Buffers

- Buffers de entrada: As técnicas de arbitragem são relativamente simples, possui uma melhor relação de área e potência, além de proporcionar um melhor desempenho para a chave crossbar.
- Buffers de saída: Em função de N entradas conectadas a cada um dos buffers de saída, a chave crossbar precisa ser N vezes mais rápida. A adoção de buffers de saída não é a mais adequada para alto desempenho. No entanto, existem vantagens em se tratando da eliminação do bloqueio de pacotes que não receberam permissão de envio porque o primeiro pacote da fila ainda não teve liberação de uma determinada saída. Este problema é conhecido como head of the line blocking e pode acontecer nas soluções com buffers de entrada.
- Buffers de crosspoint: Cada ponto de conexão da chave crossbar possui um buffer. É utilizada a técnica de roteamento chamada de self-routing. Neste caso, em cada crosspoint seria necessário além do buffer um decodificador para decisão de envio ou não do pacote. Esta solução aumenta o tamanho e a potência consumida da chave crossbar.



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

262

262

Preocupações no projeto de NoC

- Deadlock: é a representação de uma dependência cíclica. Neste caso, um pacote não consegue progredir e fica restrito a um subconjunto de estados ou roteadores.
- Livelock: é a representação de uma contínua retransmissão do pacote sem atingir o nó destino. Comum em protocolos de roteamento.
- Starvation: é a representação da não alocação de um recurso devido a postergação indefinida de acesso ao mesmo. Comum em protocolos de arbitragem.

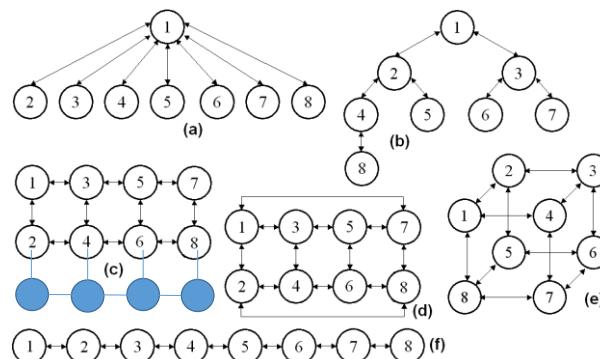
2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

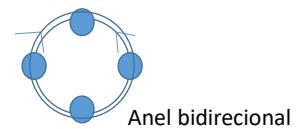
263

263

Redes-em-Chip (topologias)



FREITAS, H. C.; SANTOS, T. G. S.;
NAVAUX, P. O. A. NoC Architecture
Design for Multi-Cluster Chips, IEEE
International Conference on Field
Programmable Logic and Applications,
FPL, Heidelberg, p. 53-58. 2008



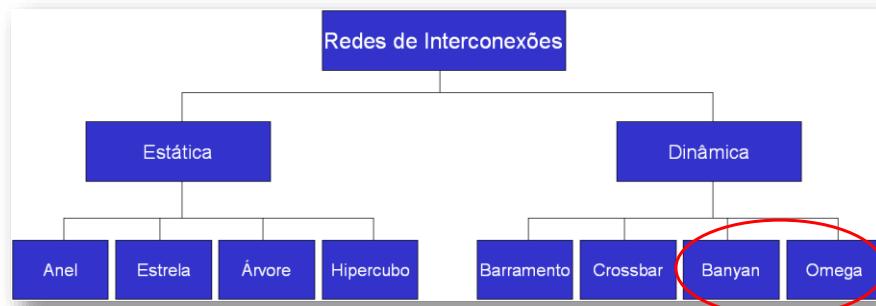
2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

264

264

Redes-em-Chip (Classificações)



DE ROSE, C., NAVAUX, P. O. A., Arquiteturas Paralelas, [S.I.], Sagra Luzzatto, 2003

2024

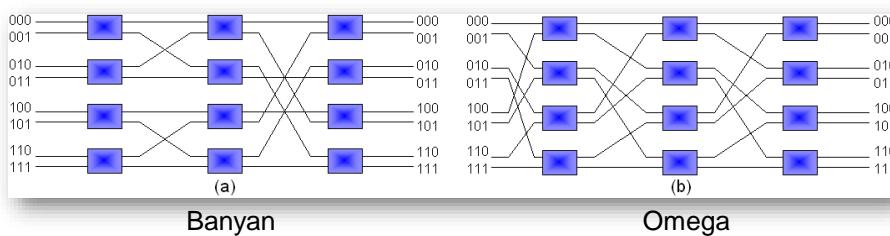
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

265

265

Redes Dinâmicas (multi-nível)

Um único caminho entre entrada e saída faz com que o roteamento seja eficiente, podendo ser feito de forma descentralizada.



Freitas, Alves, Navaux, NoC e NUCA: Conceitos e Tendências para Arquiteturas de Processadores Many-Core, Escola Regional de Alto Desempenho (ERAD), Cap.3, 2009

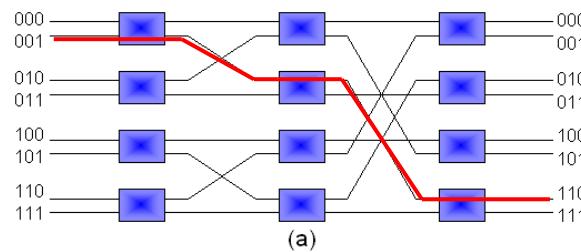
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

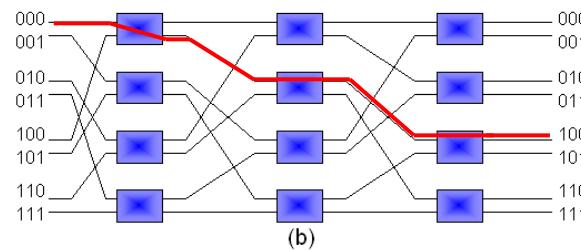
266

266

Redes Dinâmicas (multi-nível)



(a)



(b)

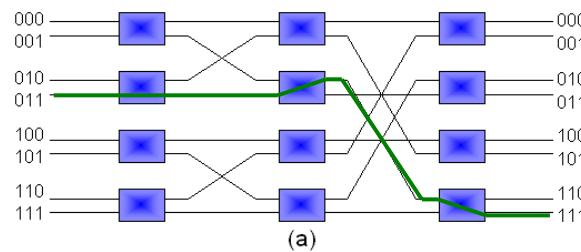
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

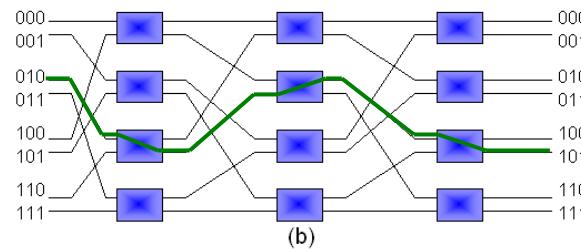
267

267

Redes Dinâmicas (multi-nível)



(a)



(b)

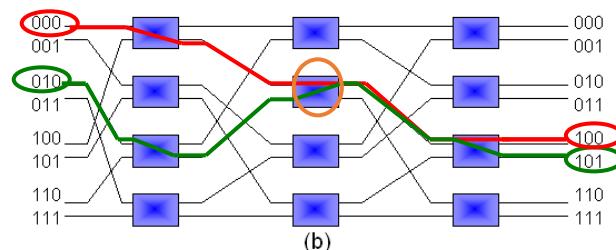
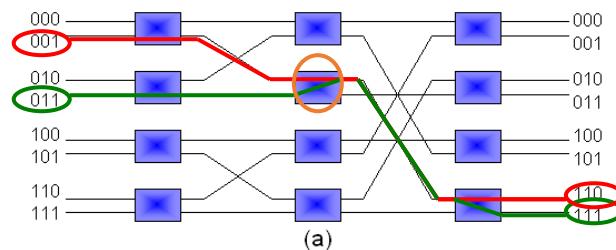
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

268

268

Redes Dinâmicas (multi-nível bloqueante)



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

269

269

Redes-em-Chip (Classificações)

Topologia	Estratégia de Transferência	Método de Controle	Estrutura do Caminho
Estrela	Indireta	Roteamento centralizado	Dedicado
Árvore	Indireta	Roteamento Descentralizado	Dedicado
Mesh	Direta	Roteamento Descentralizado	Dedicado
Torus	Direta	Roteamento Descentralizado	Dedicado
Hipercubo	Direta	Roteamento Descentralizado	Dedicado
Pipeline	Direta	Roteamento Descentralizado	Dedicado

Exemplo compartilhado?

Adaptado de:

AHMADI, H., DENZEL, W. E., A Survey of Modern High-Performance Switching Technique, IEEE Journal on Selected Areas in Communications, v.7, n.7, p.1091-1103, September 1989.

ANDERSON, G. A., JENSEN, E. D., Computer Interconnection Structures: Taxonomy, Characteristics, and Examples, ACM Computing Surveys, v.7, n.4, p.197-213, December 1975.

DUATO, J., YALAMANCHILI, S., NI, L., Interconnection Networks, [S.l.], Morgan Kaufmann, 2002.

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

270

270

Viabilidade das NoCs

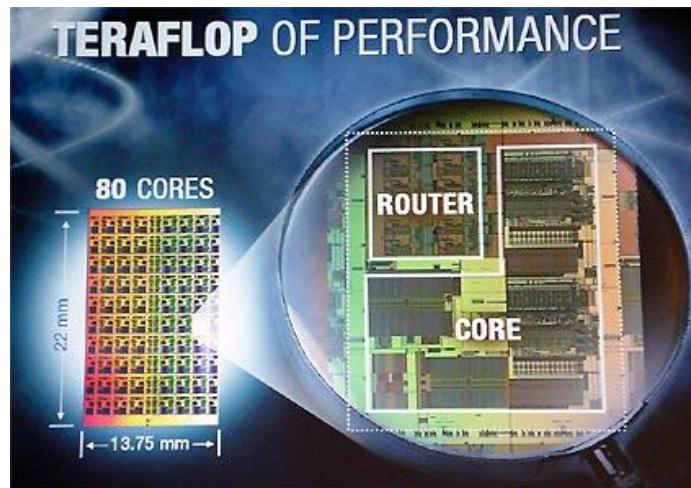
Tipo de Intercexão		Prós (+) e Contras (-)
Barramento	Fio	O aumento do fio aumenta a resistência degradando o desempenho. -
Chave Crossbar	Fio	O aumento do fio aumenta a resistência degradando o desempenho. -
<i>Network-on-Chip</i>	Fio	Os fios são ponto-a-ponto entre roteadores e o desempenho não degrada em função do aumento de nós. +
Barramento	Árbitro	O árbitro é um gargalo à medida que o número de nós aumenta. -
Chave Crossbar	Árbitro	O árbitro pode ser centralizado ou descentralizado e não é o fator principal para degradação do desempenho em função do aumento dos nós. +-
<i>Network-on-Chip</i>	Árbitro	As decisões de roteamento são distribuídas e não representam um gargalo. +
Barramento	Largura de banda	A largura de banda é limitada e compartilhada por todos os nós. -
Chave Crossbar	Largura de banda	Cada interconexão é independente e a largura de banda de comunicação por conexão não é afetada pelas demais. +
<i>Network-on-Chip</i>	Largura de banda	A largura de banda não é afetada pelo aumento da rede. +
Barramento	Latência	Latência é afetada pelo fio. +
Chave Crossbar	Latência	Latência é afetada pelo fio. +
<i>Network-on-Chip</i>	Latência	Latência é afetada pelas contenções em roteadores. -
Barramento	Compatibilidade	Em sua maioria são compatíveis com qualquer IP (<i>Intellectual Property</i>) incluindo os softwares. +
Chave Crossbar	Compatibilidade	Em sua maioria são compatíveis com qualquer IP (<i>Intellectual Property</i>) incluindo os softwares. +
<i>Network-on-Chip</i>	Compatibilidade	São necessários adaptadores (<i>wrappers</i>) entre os IPs e os softwares precisam de sincronização em sistemas <i>multi-core</i> . -
Barramento	Complexidade	Conceitos simples e bem compreendidos. +
Chave Crossbar	Complexidade	Conceitos simples e bem compreendidos. +
<i>Network-on-Chip</i>	Complexidade	Projetistas precisam de uma reeducação em função dos novos conceitos. -

Adaptado de Bjerregaard, A Survey of Research and Practices of Network-on-Chip, ACM Computing Surveys 2006

Protocolos

- Políticas e estratégias de transporte de dados em uma NoC é de responsabilidade dos protocolos.
- A definição do protocolo descreve as principais características de funcionamento da rede.
- Os protocolos precisam ser capazes de:
 - Garantir a entrega de dados.
 - Confiabilidade da rede.
 - A melhor rota.
 - Melhor desempenho, entre outros.

Research Manycore Chip



https://www.legitreviews.com/an-overview-of-intels-teraflops-research-chip_460

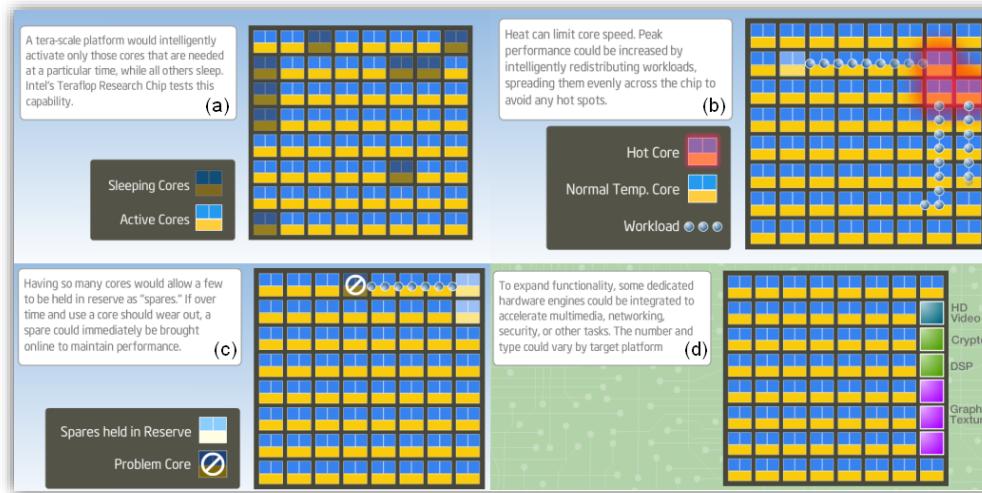
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

273

273

Research Manycore Chip



<https://www.youtube.com/watch?v=TAKGOUvtzpE>
https://www.youtube.com/watch?v=We_PRTRfiNs

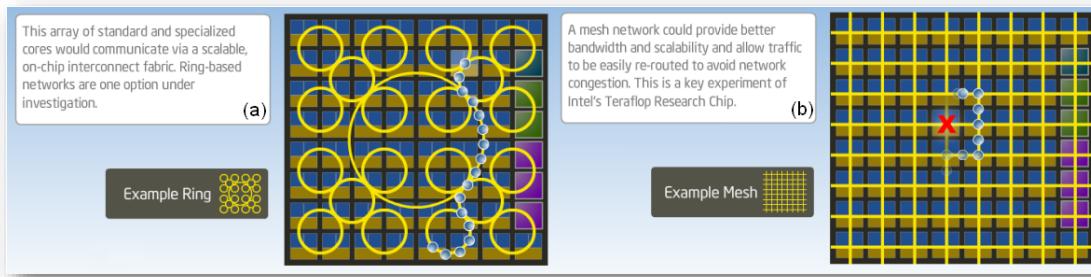
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

274

274

Research Manycore Chip



<https://www.youtube.com/watch?v=TAKG0UvtzpE>
https://www.youtube.com/watch?v=We_PRtRfiNs

Por que a Intel estuda dois tipos de topologias?

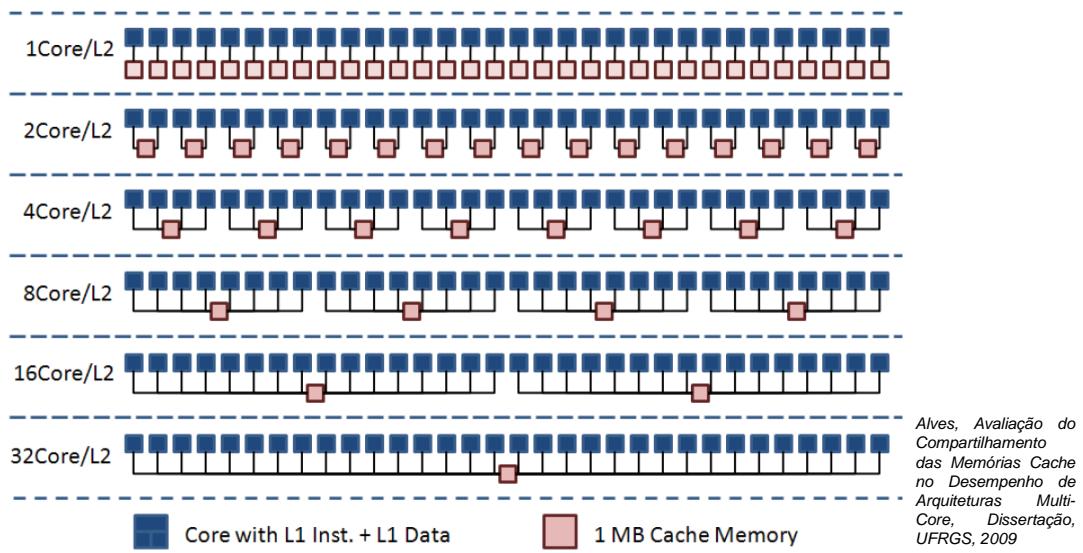
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

275

275

Processadores manycore - cache compartilhada



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

276

276

A interface hardware / software

Arquitetura de Computadores III

Arquiteturas paralelas, classificações, clusters e supercomputadores

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

277

277

Arquiteturas monoprocessadas

- Os processos compartilham o mesmo processador.
- S.O. pode ser implementado com o conceito de monoprogramação ou multiprogramação.

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

278

278

Arquiteturas monoprocessadas

- Monoprogramação: recursos do computador alocados para uma única tarefa até o seu término.
- Multiprogramação: processador alterna a execução de vários processos.

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

279

279

Arquiteturas multiprocessadas

- Multiprocessada: vários elementos de processamento.
- Tipos de arquiteturas multiprocessadas:
 - Memória compartilhada
 - Memória distribuída

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

280

280

Classificações de Flynn

Classificação de Flynn (*Flynn, 1972*) segundo o
fluxo de instruções e fluxo de dados.

	SD (Single Data)	MD (Multiple Data)
SI (Single Instruction)	SISD Máquinas von Neumann	SIMD Máquinas Array
MI (Multiple Instruction)	MISD Sem representante até agora	MIMD Multiprocessadores e multicamputadores

De Rose, Navaux, Arquiteturas Paralelas (2003)

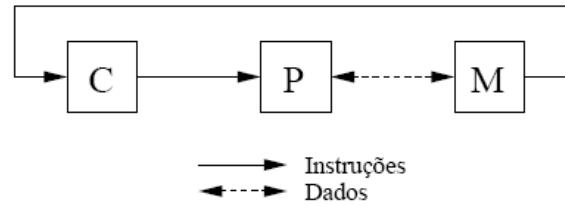
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

281

281

Classificações de Flynn



SISD

De Rose, Navaux, Arquiteturas Paralelas (2003)

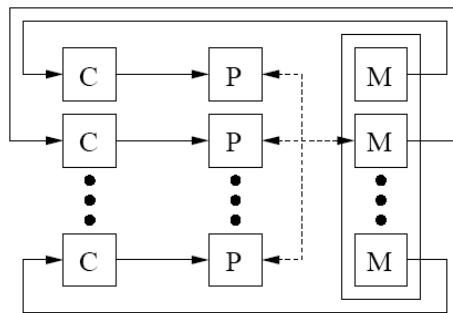
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

282

282

Classificações de Flynn



De Rose, Navaux, Arquiteturas Paralelas (2003)

MISD

2024

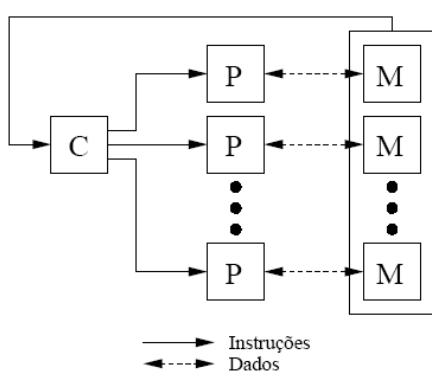
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

283

283

Classificações de Flynn

- Processadores vetoriais:



SIMD

De Rose, Navaux, Arquiteturas Paralelas (2003)

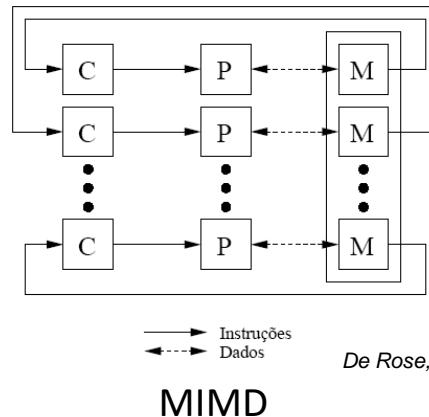
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

284

284

Classificações de Flynn



MIMD

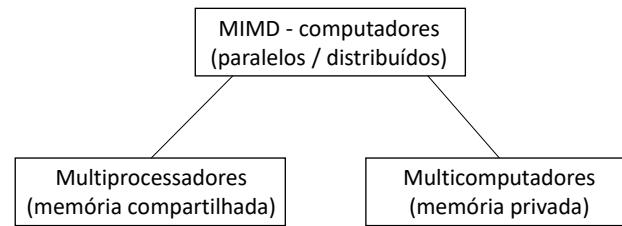
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

285

285

Subdivisão da classe MIMD



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

286

286

Classificações segundo compartilhamento de memória

- UMA: Uniform Memory Access
- NUMA: Non-Uniform Memory Access
- CC-NUMA: Cache-Coherent NUMA
- NCC-NUMA: Non-Cache-Coherent NUMA
- SC-NUMA: Software-Coherent NUMA
- COMA: Cache-Only Memory Architecture
- NORMA: Non-Remote Memory Access

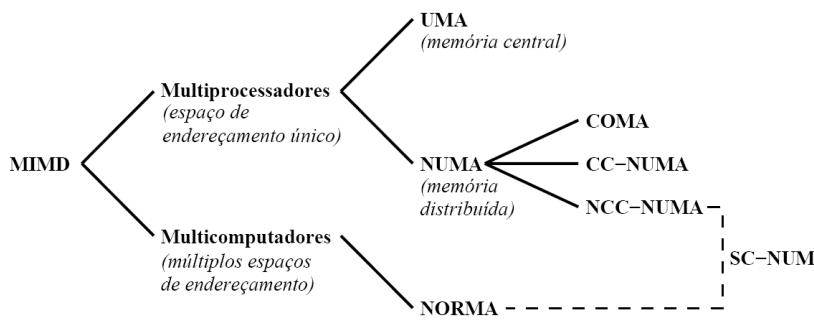
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

287

287

Classificações segundo compartilhamento de memória



De Rose, Navaux, Arquiteturas Paralelas (2003)

A linha tracejada indica que através de um software que implemente coerência de *cache*, as máquinas NCC-NUMA e NORMA podem se transformar em máquinas SC-NUMA.

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

288

288

Memória Compartilhada

- Elementos de processamento compartilham a mesma memória.
- Programação realizada através de variável compartilhada. Maior facilidade na construção de programas paralelos.
- Neste tipo de arquitetura existe uma limitação de número de nós.
- Escalabilidade não é total. **Por que?**
- Programação em memória compartilhada é realizada com threads.
- Exemplos:
 - Pthreads
 - OpenMP

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

289

289

Memória Compartilhada

- O desempenho neste tipo de sistema é maior se consideramos no seu projeto o uso de memória cache.
- Surge o problema de coerência de cache.
- Solução em software ou em hardware.

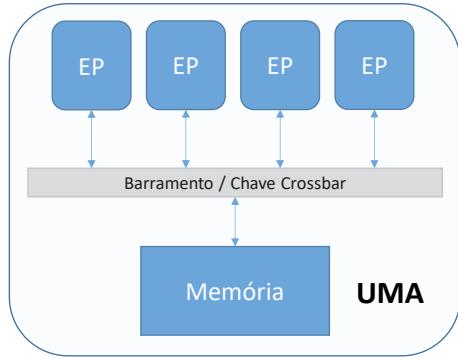
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

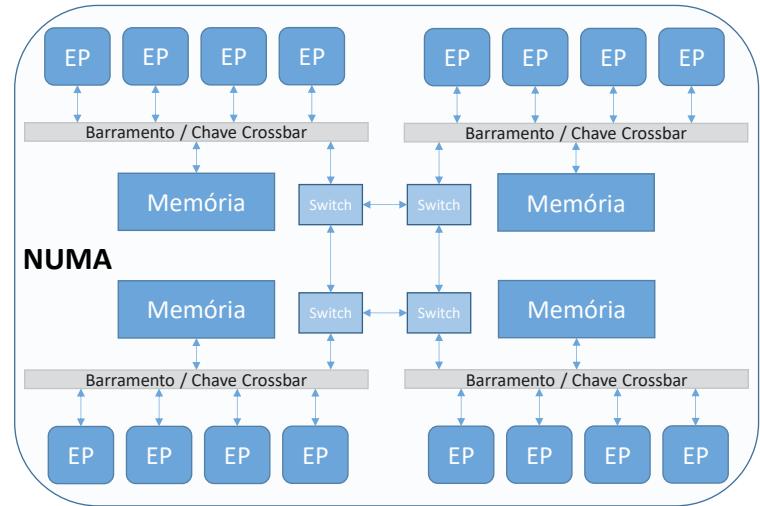
290

290

UMA vs. NUMA



EP: Elemento de Processamento



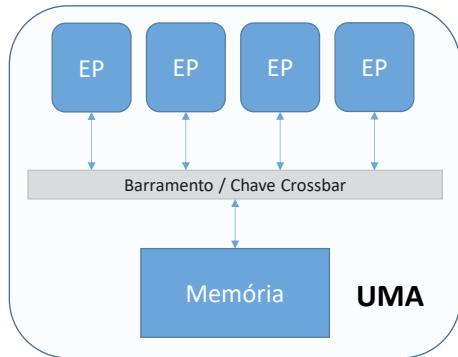
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

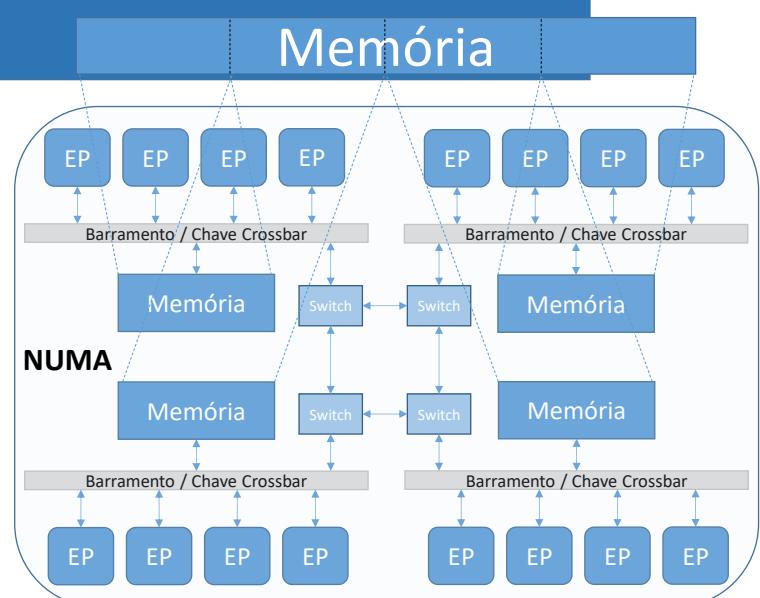
291

291

UMA vs. NUMA



EP: Elemento de Processamento



2024

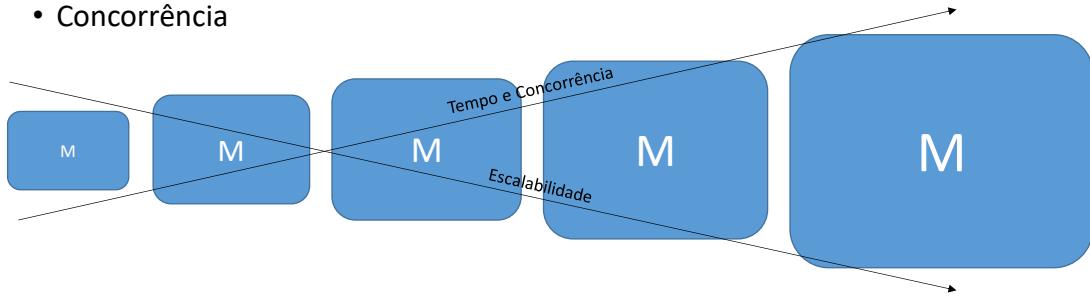
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

292

292

Qual a vantagem de usar uma máquina NUMA?

- Restrições existentes em relação à memória (M) centralizada que podem afetar o desempenho e o consumo de energia:
 - Escalabilidade
 - Tempo de acesso
 - Concorrência

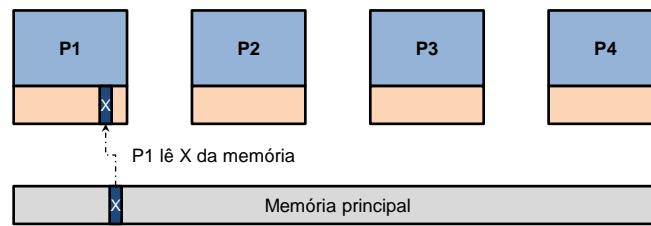


Uso compartilhado da memória

- A maioria dos processadores atuais possuem múltiplos núcleos
- Diversos núcleos compartilham o acesso a dados em uma mesma unidade de memória
- Outro problema: garantir a **coerência** do dado, o que é resolvido com algoritmos específicos para este fim

Problema da coerência

- Multicores: Múltiplos processadores e caches
- Múltiplas cópias dos mesmos dados em caches diferentes



Souza, M. A., Way-Replacement Algorithms for Multicore Processors Based on Coherence and Sharing States with Reinforcement Learning, Tese, PUC Minas, 2021

2024

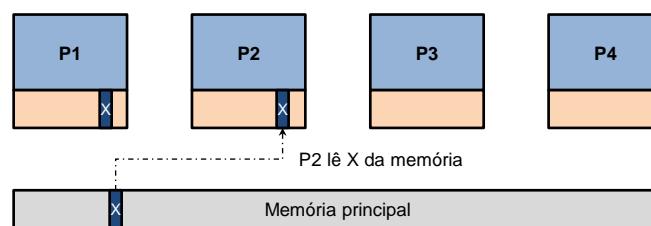
Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

295

295

Problema da coerência

- Multicores: Múltiplos processadores e caches
- Múltiplas cópias dos mesmos dados em caches diferentes



Souza, M. A., Way-Replacement Algorithms for Multicore Processors Based on Coherence and Sharing States with Reinforcement Learning, Tese, PUC Minas, 2021

2024

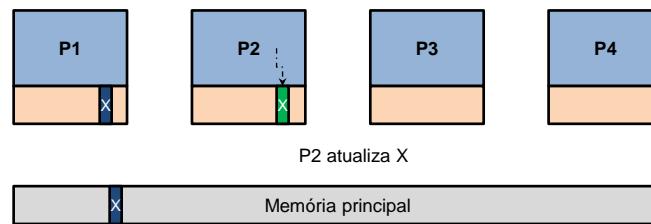
Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

296

296

Problema da coerência

- Multicores: Múltiplos processadores e caches
- Múltiplas cópias dos mesmos dados em caches diferentes



Souza, M. A., Way-Replacement Algorithms for Multicore Processors Based on Coherence and Sharing States with Reinforcement Learning, Tese, PUC Minas, 2021

2024

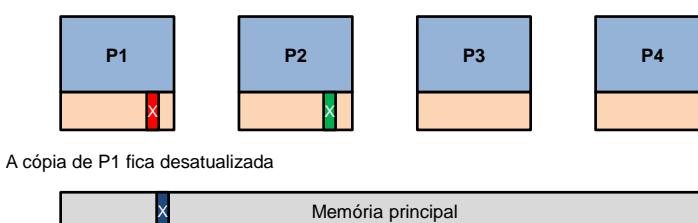
Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

297

297

Problema da coerência

- Multicores: Múltiplos processadores e caches
- Múltiplas cópias dos mesmos dados em caches diferentes



Souza, M. A., Way-Replacement Algorithms for Multicore Processors Based on Coherence and Sharing States with Reinforcement Learning, Tese, PUC Minas, 2021

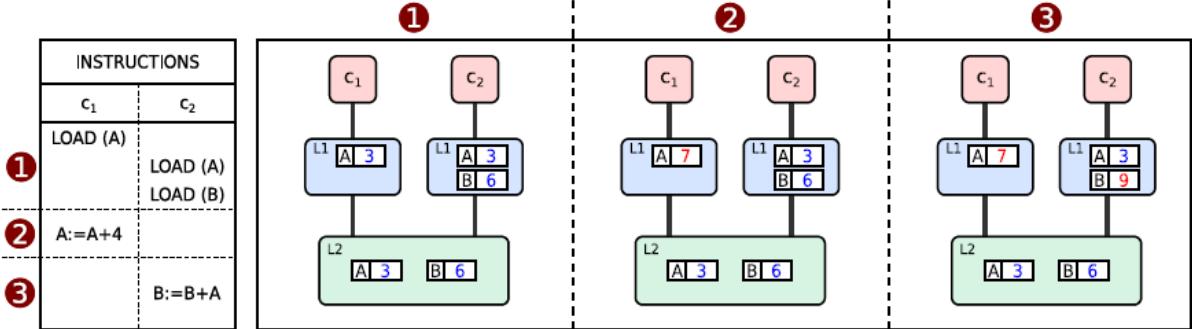
2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

298

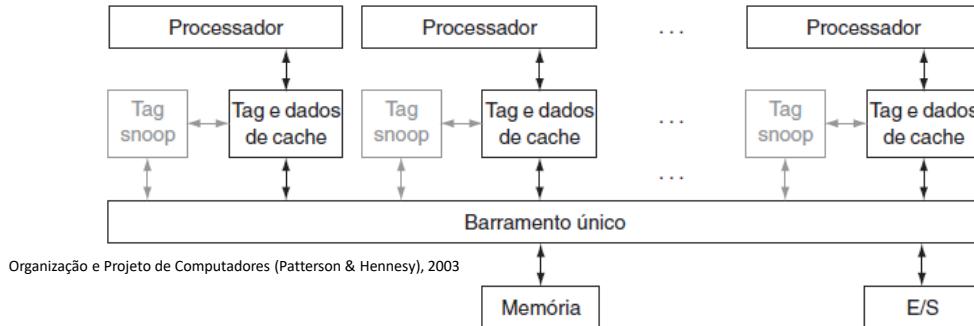
298

Problema da coerência



Souza, M. A., Way-Replacement Algorithms for Multicore Processors Based on Coherence and Sharing States with Reinforcement Learning, Tese, PUC Minas, 2021

Coerência de cache (Snooping)



Coerência de cache com snooping. Um método para manter coerência de cache em que todos os controladores de cache monitoram o barramento para determinar se possuem ou não uma cópia do bloco desejado.

Write-invalidate. Um tipo de protocolo de snooping em que o processador de escrita faz com que todas as cópias em outras caches sejam invalidadas antes de mudar sua cópia local, o que permite atualizar os dados locais até que outro processador os solicite.

Write-update. Um tipo de protocolo de snooping em que o processador de escrita faz com que todas as cópias em outras caches sejam atualizadas.

Coerência de cache (Diretório)

- Cada módulo de memória possui um diretório local que armazena as informações sobre onde as cópias dos blocos estão residentes.
- Os protocolos baseados em diretório enviam comandos de consistência seletivamente para aquelas caches que possuem uma cópia válida do bloco de dados compartilhado.

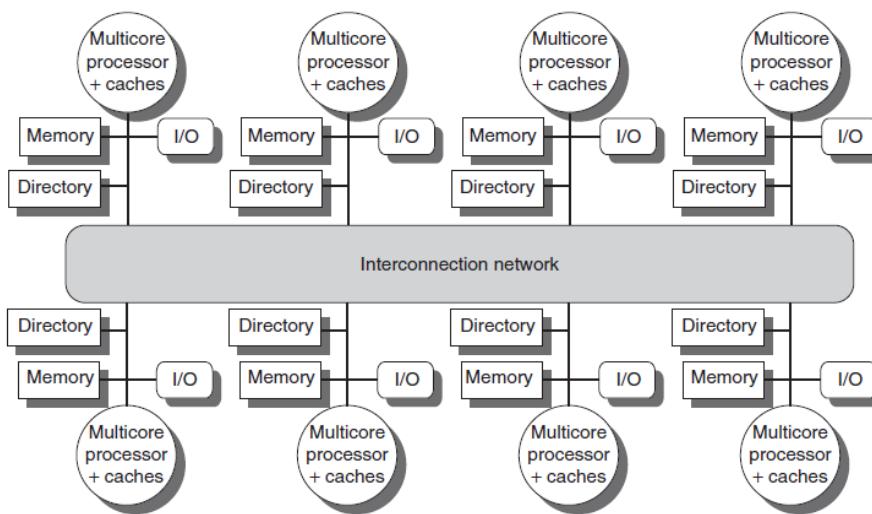
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

301

301

Coerência de cache (Diretório)



HENNESSY, John L.,
PATTERSON, David A.,
Computer Architecture
A Quantitative Approach

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

302

302

Coerência de cache (MSI, MESI, MOESI)

- Estados que cada bloco na memória cache possui:
 - Inválido (I): bloco inválido na memória cache.
 - Shared (S) ou Compartilhado: bloco só foi lido e pode haver cópias em outras memórias cache.
 - Modificado (M): apenas essa cache possui cópia do bloco e a memória principal não está atualizada.
 - Exclusivo (E): Apenas essa cache possui cópia do bloco e a memória principal está atualizada.
 - Owner (O) : Essa cache supre o dado em caso de leitura com falha no barramento uma vez que a memória não está atualizada. Outras caches podem ter cópia do dado.

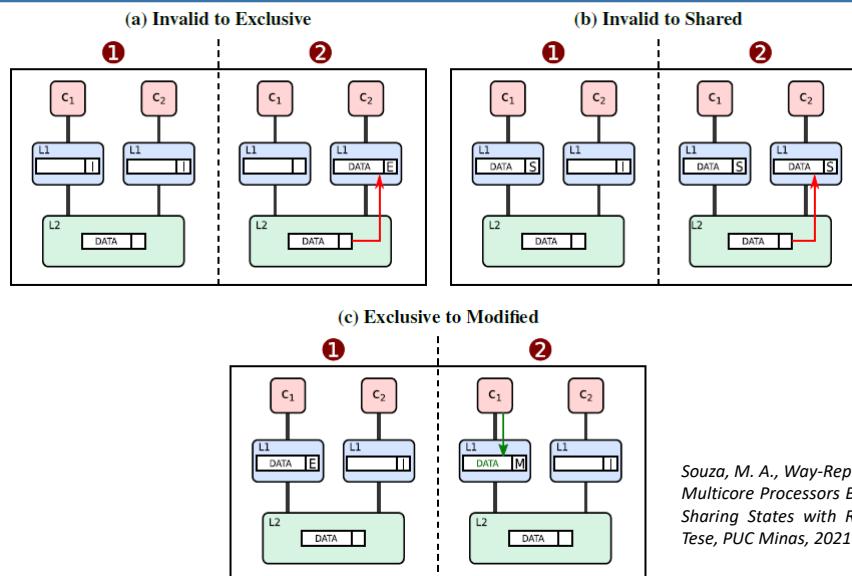
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

303

303

Protocolo MESI



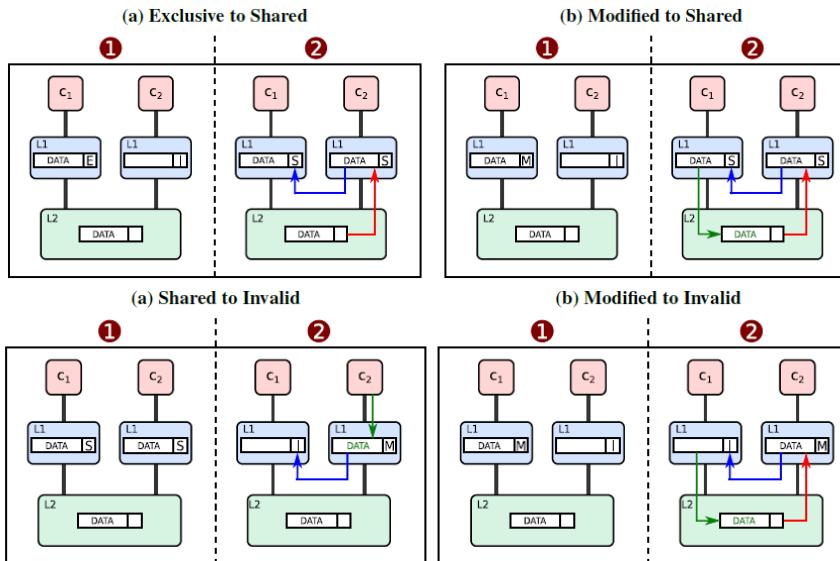
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

304

304

Protocolo MESI



Souza, M. A., Way-Replacement Algorithms for Multicore Processors Based on Coherence and Sharing States with Reinforcement Learning, Tese, PUC Minas, 2021

2024

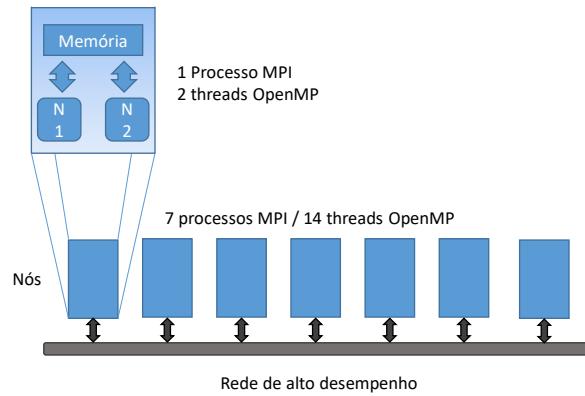
Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

305

305

Memória Distribuída

- Grupo de computadores autônomos (nós) que trabalham juntos como um recurso único.
- Os nós são interconectados através de redes de alto desempenho.



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

306

306

Memória Distribuída

- Escalabilidade absoluta e incremental.
- Alta disponibilidade.
- Excelente custo benefício.
- Comunicação realizada através de passagem de mensagens.
 - MPI (Message Passing Interface) ou
 - PVM (Parallel Virtual Machine).

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

307

307

Memória Distribuída

- Cluster ou aglomerado de computadores.
 - São usados em gerenciadores de bancos de dados, com servidores WEB.
 - São usados principalmente com processamento paralelo.
- Grids ou grades computacionais.

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

308

308

Cluster de Computadores



<https://hpcf.umbc.edu/system-description-maya/>

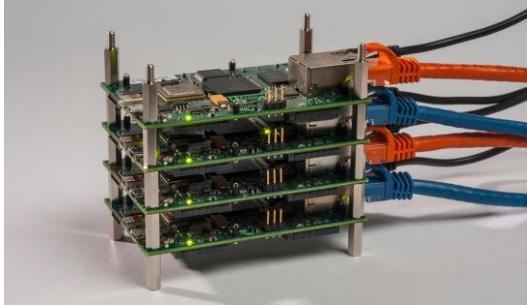
2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

309

309

Cluster Parallelia Board



<https://www.parallelia.org/2013/08/21/parallelia-hardware-update/>



<https://www.parallelia.org/2014/04/30/cases-and-cooling/>

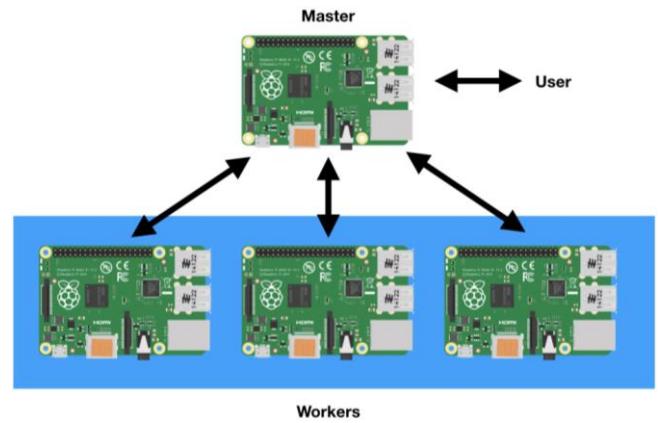
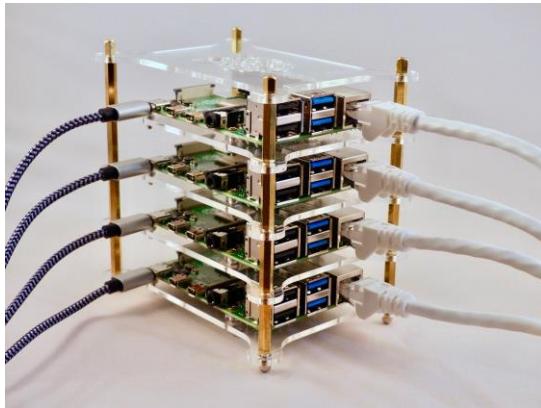
2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

310

310

Cluster Raspberry Pi 4



<https://magpi.raspberrypi.org/articles/build-a-raspberry-pi-cluster-computer>

2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

311

311

Cluster Raspberry Pi 2 Model B



Photo courtesy of CArT

2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

312

312

Jaguar – Cray XT5-HE Opteron Six Core 2.6 GHz



<https://phys.org/news/2009-11-oak-ridge-jaguar-supercomputer-world.html>

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

313

313

Clusters de Computadores (top500.org)

Rank	Site	Computer/Year Vendor	Cores	R _{max}	R _{peak}	Power	
1	Oak Ridge National Laboratory United States	Jaguar - Cray XT5-HE Opteron Six Core 2.6 GHz / 2009 Cray Inc.	224162	1759.00	2331.00	6950.60	MPP
2	DOE/INSA/LANL United States	Roadrunner - BladeCenter QS22/LS21 Cluster, PowerCell 8i 3.2 Ghz / Opteron DC 1.8 Ghz, Voltaire Infiniband / 2009 IBM	122400	1042.00	1375.78	2345.50	Cluster
3	National Institute for Computational Sciences/University of Tennessee United States	Kraken XT5 - Cray XT5-HE Opteron Six Core 2.6 GHz / 2009 Cray Inc.	98928	831.70	1028.85		MPP
4	Forschungszentrum Juelich (FZJ) Germany	JUGENE - Blue Gene/P Solution / 2009 IBM	294912	825.50	1002.70	2268.00	MPP
5	National SuperComputer Center in Tianjin/NUDT China	Tianhe-1 - NUDT TH-1 Cluster, Xeon E5540/E5450, ATI Radeon HD 4870 2, Infiniband / 2009 NUDT	71680	563.10	1206.19		Cluster
6	NASA/Ames Research Center/NAS United States	Pleiades - SGI Altix ICE 8200EX, Xeon QC 3.0 Ghz/Nehalem EP 2.93 Ghz / 2009 SGI	56320	544.30	673.26	2348.00	MPP
7	DOE/INSA/LNL United States	BlueGene/L - eServer Blue Gene Solution / 2007 IBM	212992	478.20	596.38	2329.60	MPP
8	Argonne National Laboratory United States	Blue Gene/P Solution / 2007 IBM	163840	458.61	557.06	1260.00	MPP
9	Texas Advanced Computing Center/Univ. of Texas United States	Ranger - SunBlade x6420, Opteron QC 2.3 Ghz, Infiniband / 2008 Sun Microsystems	62976	433.20	579.38	2000.00	Cluster
10	Sandia National Laboratories / National Renewable Energy Laboratory United States	Red Sky - Sun Blade x6275, Xeon X55xx 2.93 Ghz, Infiniband / 2009 Sun Microsystems	41616	423.90	487.74		Cluster

Novembro de 2009

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

314

314

Clusters de Computadores (top500.org)

Rank	Site	Computer
1	RIKEN Advanced Institute for Computational Science (AICS) Japan	K computer, SPARC64 Vllifx 2.0GHz, Tofu interconnect Fujitsu
2	National Supercomputing Center in Tianjin China	Tianhe-1A - NUDT TH MPP, X5670 2.93Ghz 6C, NVIDIA GPU, FT-1000 8C NUDT
3	DOE/SC/Oak Ridge National Laboratory United States	Jaguar - Cray XT5-HE Opteron 6-core 2.6 GHz Cray Inc.
4	National Supercomputing Centre in Shenzhen (NSCS) China	Nebulae - Dawning TC3600 Blade, Intel X5650, NVidia Tesla C2050 GPU Dawning
5	GSIC Center, Tokyo Institute of Technology Japan	TSUBAME 2.0 - HP ProLiant SL390s G7 Xeon 6C X5670, Nvidia GPU, Linux/Windows NEC/HPC
6	DOE/INSA/LANL/SNL United States	Cielo - Cray XE6 8-core 2.4 GHz Cray Inc.
7	NASA/Ames Research Center/NAS United States	Pleiades - SGI Altix ICE 8200EX/8400EX, Xeon HT QC 3.0/Xeon 5570/5670 2.93 Ghz, Infiniband SGI
8	DOE/SC/BNL/NERSC United States	Hopper - Cray XE6 12-core 2.1 GHz Cray Inc.
9	Commissariat à l'Energie Atomique (CEA) France	Tera-100 - Bull bulx super-node S6010/S6030 Bull SA
10	DOE/INSA/LANL United States	Roadrunner - BladeCenter QS22/LS21 Cluster, PowerXCell 8i 3.2 Ghz / Opteron DC 1.8 GHz, Voltaire Infiniband IBM

Junho de 2011

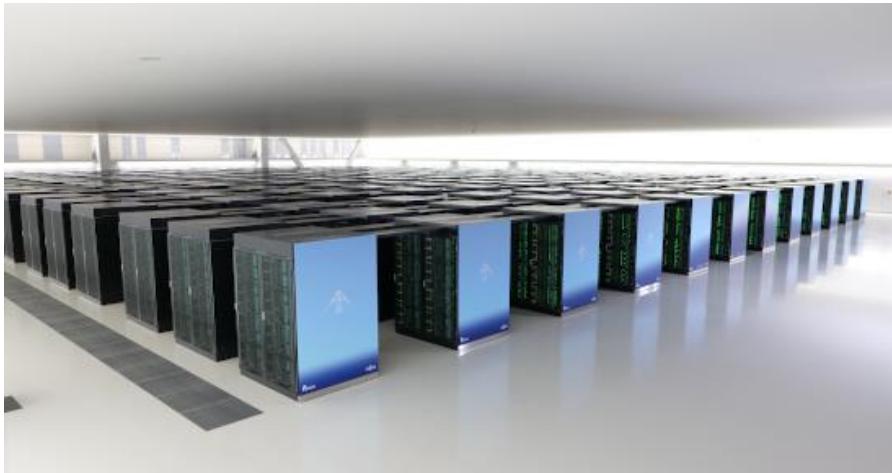
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

315

315

Cluster de Computadores



https://www.riken.jp/en/news_pubs/news/2020/20200623_1/

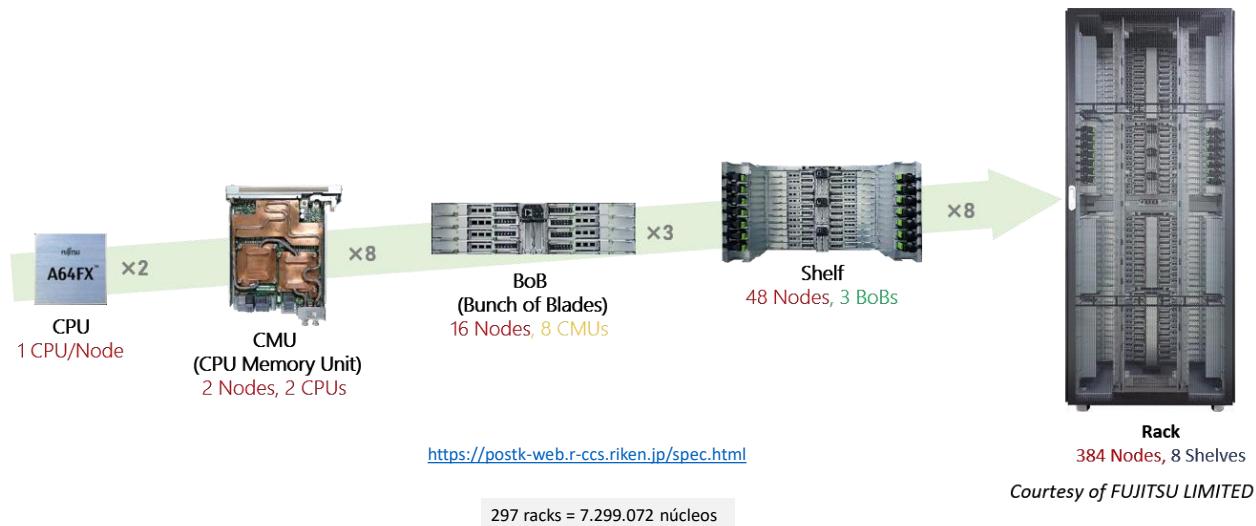
2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

316

316

Supercomputador Fugaku (Fujitsu, Japão)



2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

317

317

Clusters de Computadores (top500.org)

Rank	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	Supercomputer Fugaku - Supercomputer Fugaku, A64FX 48C 2.2GHz, Tofu interconnect D, Fujitsu RIKEN Center for Computational Science Japan	7,299,072	415,530.0	513,854.7	28,335
2	Summit - IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM DOE/SC/Oak Ridge National Laboratory United States	2,414,592	148,600.0	200,794.9	10,096
3	Sierra - IBM Power System AC922, IBM POWER9 22C 3.1GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM / NVIDIA / Mellanox DOE/NNSA/LLNL United States	1,572,480	94,640.0	125,712.0	7,438
4	Sunway TaihuLight - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway, NRCPC National Supercomputing Center in Wuxi China	10,649,600	93,014.6	125,435.9	15,371
5	Tianhe-2A - TH-IVB-FEP Cluster, Intel Xeon E5-2692v2 12C 2.2GHz, TH Express-2, Matrix-2000, NUDT National Super Computer Center in Guangzhou China	4,981,760	61,444.5	100,678.7	18,482

TOP500 LIST - JUNE 2020

2024

318

318

Clusters de Computadores (top500.org) - Brasil

Rank	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
56	Atlas - Bull 4029GP-TVRT, Xeon Gold 6240 18C 2.6GHz, NVIDIA Tesla V100, Infiniband EDR, Atos Petróleo Brasileiro S.A Brazil	91,936	4,376.0	8,848.5	547
82	Fênix - Bull 4029GP-TVRT, Xeon Gold 5122 4C 3.6GHz, NVIDIA Tesla V100, Infiniband EDR, Atos Petróleo Brasileiro S.A Brazil	60,480	3,161.0	5,371.8	390
240	Santos Dumont (SDumont) - Bull Sequana X1000, Xeon Gold 6252 24C 2.16Hz, Mellanox InfiniBand EDR, NVIDIA Tesla V100 SXM2, Atos Laboratório Nacional de Computação Científica Brazil	33,856	1,849.0	2,727.0	
395	Ogbon Cimatec/Petrobras - Bull Sequana X1000, Xeon Gold 6240 18C 2.6GHz, Mellanox InfiniBand EDR, NVIDIA Tesla V100 SXM2, Atos SENAI CIMATEC Brazil	27,768	1,605.0	2,323.3	

TOP500 LIST - JUNE 2020

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

319

319

Clusters de Computadores

Santos Dumont Supercomputer



<https://sdumont.lncc.br/>

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

320

320

Clusters de Computadores (top500.org)

2024

321

Rank	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	Supercomputer Fugaku - Supercomputer Fugaku, A64FX 48C 2.2GHz, Tofu interconnect D, Fujitsu RIKEN Center for Computational Science Japan	7,630,848	442,010.0	537,212.0	29,899
2	Summit - IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM DOE/SC/Oak Ridge National Laboratory United States	2,414,592	148,600.0	200,794.9	10,096
3	Sierra - IBM Power System AC922, IBM POWER9 22C 3.1GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM / NVIDIA / Mellanox DOE/NNSA/LLNL United States	1,572,480	94,640.0	125,712.0	7,438
4	Sunway TaihuLight - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway, NRCPC National Supercomputing Center in Wuxi China	10,649,600	93,014.6	125,435.9	15,371
5	Perlmutter - HPE Cray EX235n, AMD EPYC 7763 64C 2.45GHz, NVIDIA A100 SXM4 40 GB, Slingshot-10, HPE DOE/SC/LBNL/NERSC	761,856	70,870.0	93,750.0	2,589

TOP500 LIST - NOVEMBER 2021

321

Clusters de Computadores (top500.org) - Brasil

2024

322

55	Dragão - Supermicro SYS-4029GP-TVRT, Xeon Gold 6230R 26C 2.1GHz, NVIDIA Tesla V100, Infiniband EDR, Atos Petróleo Brasileiro S.A Brazil	188,224	8,983.0	14,006.5	943
107	Atlas - Bull 4029GP-TVRT, Xeon Gold 6240 18C 2.6GHz, NVIDIA Tesla V100, Infiniband EDR, Atos Petróleo Brasileiro S.A Brazil	91,936	4,376.0	8,848.5	547
125	IARA - NVIDIA DGX A100, AMD EPYC 7742 64C 2.25GHz, NVIDIA A100 SXM4 40 GB, Infiniband, Nvidia SiDi Brazil	24,800	3,657.0	4,130.4	
146	Fênix - Bull 4029GP-TVRT, Xeon Gold 5122 4C 3.6GHz, NVIDIA Tesla V100, Infiniband EDR, Atos Petróleo Brasileiro S.A Brazil	60,480	3,161.0	5,371.8	390
388	Santos Dumont (SDumont) - Bull Sequana X1000, Xeon Gold 6252 24C 2.1GHz, Mellanox InfiniBand EDR, NVIDIA Tesla V100 SXM2, Atos Laboratório Nacional de Computação Científica Brazil Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas	33,856	1,849.0	2,727.0	

TOP500 LIST - NOVEMBER 2021

322

Lei de Amdahl

- O ganho ideal é limitado pela fração do Código não melhorada.
- O ganho real é a razão dos tempos de execução:

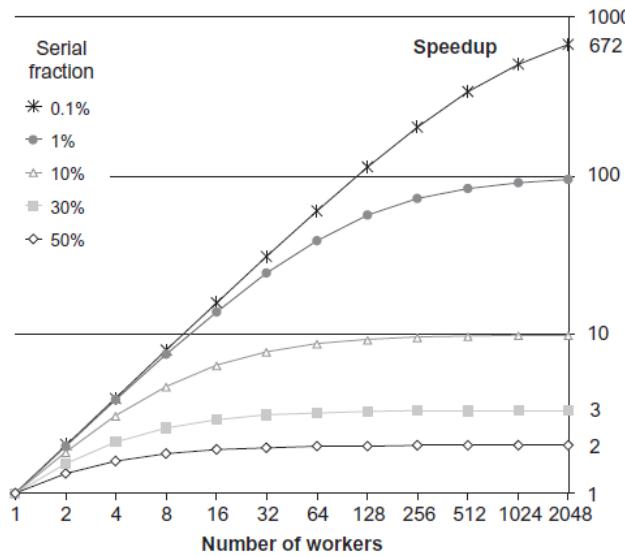
$$\text{Speedup}_{\text{overall}} = \frac{\text{Execution time}_{\text{old}}}{\text{Execution time}_{\text{new}}} = \frac{1}{(1 - \text{Fraction}_{\text{enhanced}}) + \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}}}$$

- Suponha que nós queremos melhorar o processador usado para serviço Web. O nome processador é 10 vezes mais rápido em computação do que o processador original. Assumindo que o processador está 40% do tempo ocupado com computação e outros 60% esperando por I/O. Qual é o ganho obtido pelo melhoramento?

$$\text{Fraction}_{\text{enhanced}} = 0.4; \text{Speedup}_{\text{enhanced}} = 10; \text{Speedup}_{\text{overall}} = \frac{1}{0.6 + \frac{0.4}{10}} = \frac{1}{0.64} \approx 1.56$$

HENNESSY, John L.
PATTERSON, David A.,
Computer Architecture
A Quantitative Approach

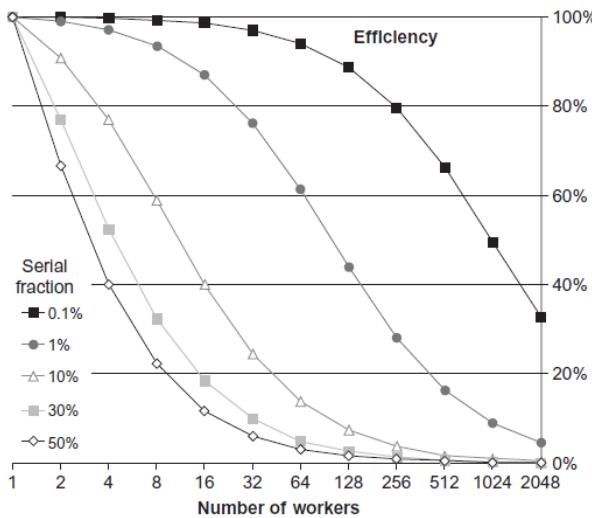
Lei de Amdahl



- A escalabilidade da paralelização é limitada pela fração serial do código. Esta fração é a parte não paralelizada.

PACHECO, Peter S., An introduction to parallel programming

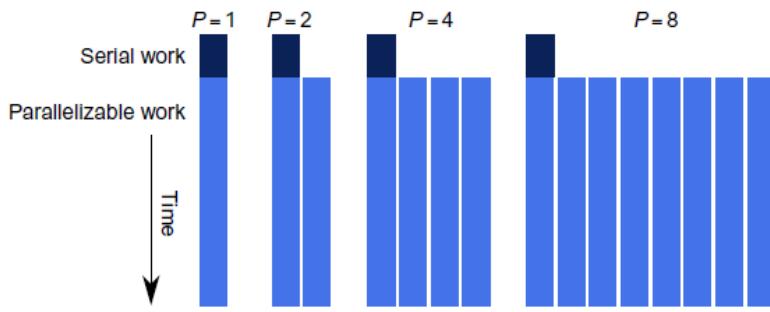
Lei de Amdahl



- Mesmo quando ganhos são possíveis, a eficiência pode facilmente ser ruim em função do trecho não paralelizado.

PACHECO, Peter S., An introduction to parallel programming

Lei de Gustafson-Barsis'



- Se o tamanho do problema aumenta em P enquanto o trecho serial cresce lentamente ou se mantém fixo, o ganho cresce se os nós de processamento aumentam em quantidade.

PACHECO, Peter S., An introduction to parallel programming

Strong and Weak Scalability (escalabilidade forte e fraca)

- **Strong scalability:** uma forma de escalabilidade que mede como o desempenho aumenta quando se faz uso de nós adicionais, mas com o tamanho do problema fixo.
 - A lei de Amdahl considera o ganho variando número de nós e problema fixo.
- **Weak scalability:** uma forma de escalabilidade que mede como o desempenho aumenta quando se faz uso de nós adicionais ao mesmo tempo que se aumenta o tamanho do problema, em taxas iguais.
 - A lei de Gustafson-Barsis' Law assume que o tamanho do problema cresce conforme aumenta-se o número de nós.

PACHECO, Peter S., An introduction to parallel programming

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

327

327

Eficiência e Ganho

- Duas importantes métricas relacionadas ao desempenho e paralelismo são ganho e eficiência. O **ganho** compara o tempo para resolver um problema computacional idêntico em uma unidade de processamento versus P unidades:

$$\text{speedup} = S_P = \frac{T_1}{T_P}$$

- onde T_1 é o tempo de um programa com uma unidade de processamento e T_P é o tempo em P unidades. **Eficiência** é o ganho dividido pelo número de unidades de processamento utilizadas:

$$\text{efficiency} = \frac{S_P}{P} = \frac{T_1}{P T_P}$$

PACHECO, Peter S., An introduction to parallel programming

2024

Arquitetura de Computadores III - Engenharia e Ciência da Computação - PUC Minas

328

328

Ganho Linear Superlinear

- Um algoritmo que executa P vezes mais rápido em P processadores possui **ganho linear**.
- **Ganho Superlinear:** uma eficiência maior do que 100%.
 - Como isso é possível?
 - Utilização de cache?

PACHECO, Peter S., An introduction
to parallel programming

2024

Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

329

329

Desempenho de uma aplicação paralela

- O que pode impactar o desempenho de uma aplicação paralela reduzindo sua escalabilidade?
 - Rede. Por que?
 - Carga de trabalho desbalanceada. Por que?
 - Regiões sequencias de códigos paralelos. Por que?

2024

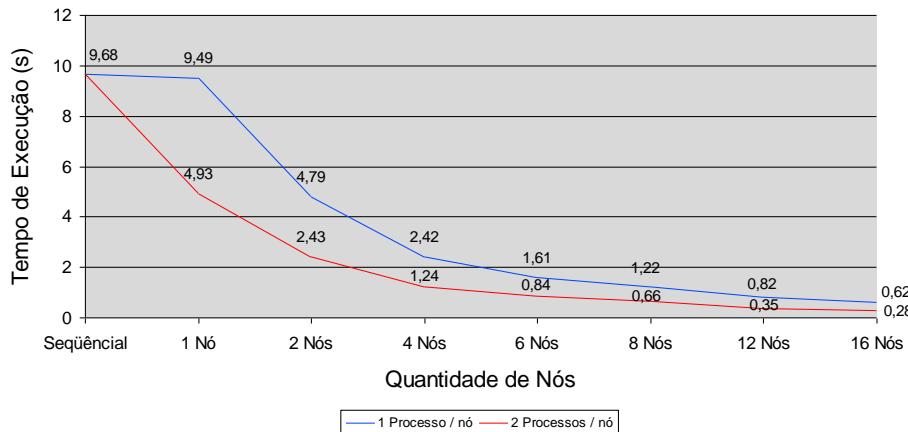
Arquitetura de Computadores III - Engenharia e Ciéncia da Computação - PUC Minas

330

330

Avaliação de desempenho

Resultados de Desempenho

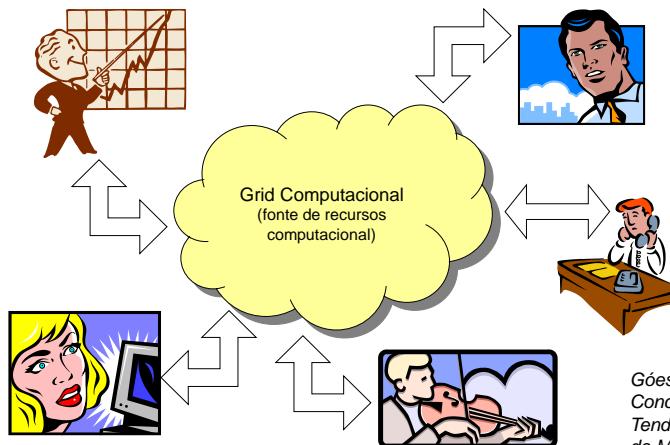


Grid Computacional

- Uma plataforma para execução de aplicações paralelas
 - Amplamente distribuída
 - Heterogênea
 - Compartilhada
 - Sem controle central
 - Com múltiplos domínios administrativos
- Diferença para computação em nuvem (*cloud computing*)?
- O que é *fog computing*?

Grid Computacional

- Analogia com rede elétrica



Góes et al., *Computação em Grade: Conceitos, Tecnologias, Aplicações e Tendências*, Escola Regional de Informática de Minas Gerais (ERI-MG), 2005

Grid Computacional

- SMPs
- MPPs
- NOWs
- Grids



- SMP: Symmetric Multiprocessor (memória compartilhada)
- MPP: Massively Parallel Processors
- NOW: Network of Workstations

Grid Computacional

- TeraGrid (de 2001 a 2011)
 - 4 centros de supercomputação norte-americanos
 - Cada centro com milhares de processadores dedicados ao TeraGrid
 - Canais de altíssima velocidade (40 GBits/s)
 - Poder agregado de 13,6 TeraFlops
- SETI@home (desde 1999)
 - Ciclos ociosos de 1.6 milhões de processadores espalhados em 224 países
 - Computa a uma taxa superior a 10 Teraflops
- Grid5000 (desde 2003)
 - Instrumento científico para estudo de sistemas paralelos e distribuídos de larga escala.
 - O objetivo inicial era alcançar 5000 processadores, atualizado para núcleos e alcançado no inverno de 2008-2009.
 - São 9 sites na França (Grenoble, Lille, Luxembourg, Lyon, Nancy, Nantes, Rennes, Sophia-Antipolis, Toulouse).