Universidade Federal do Rio Grande do Sul
Instituto de Informática

Organização de Computadores

Aula 17

Memória cache
segunda parte
Organização de Memória Cache

Revisão última aula

- **Hierarquia de Memória**
- **Hit e Miss**
- **Localidade Temporal**
- **Localidade Espacial**
- **Impacto no Desempenho – Hit Ratio - Taxa de Acerto**
- **Organizações de Memória Cache - Mapeamento**

Memória cache
segunda parte

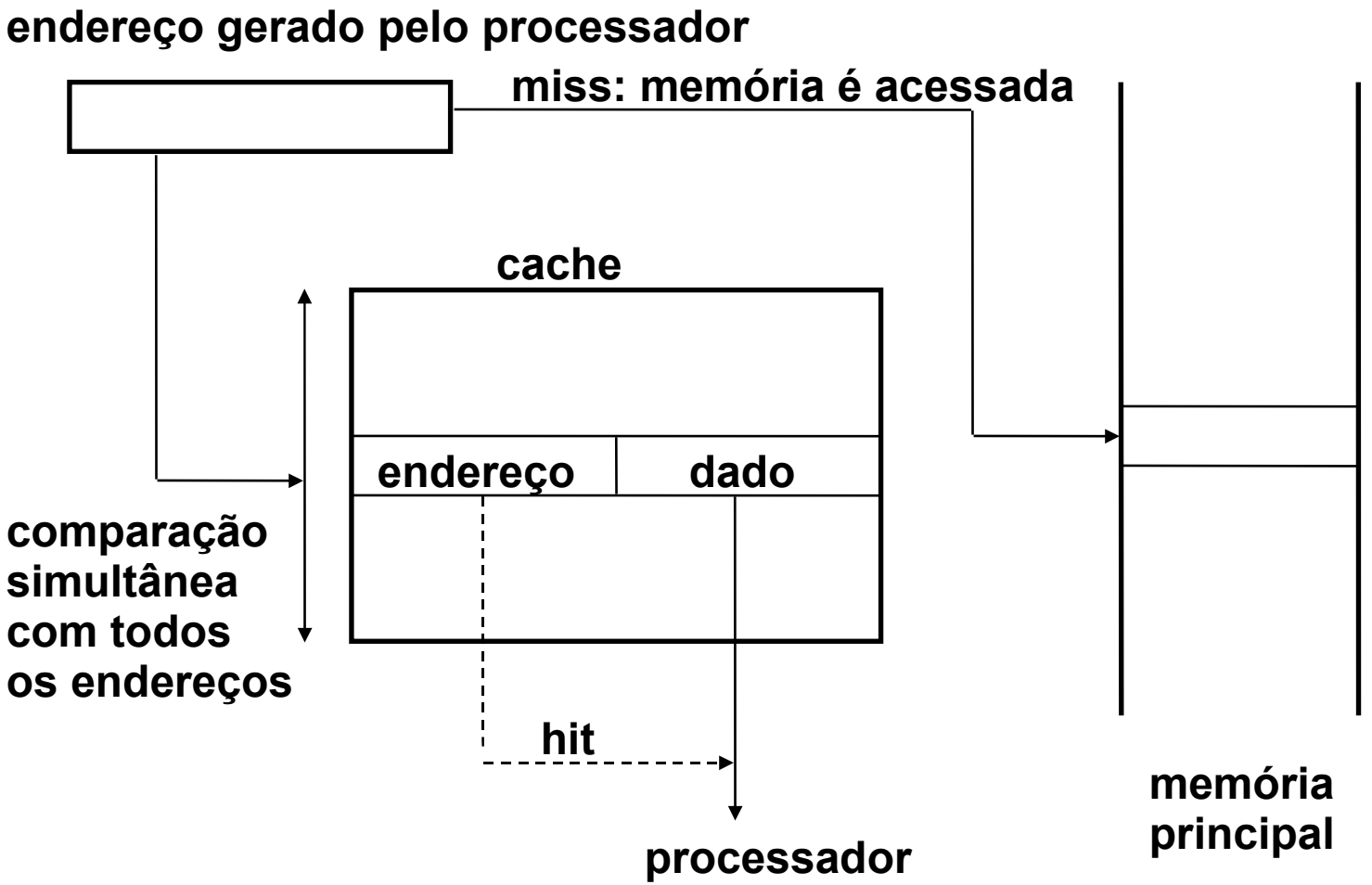
1. **Mapeamento completamente associativo**

2. **Mapeamento direto**

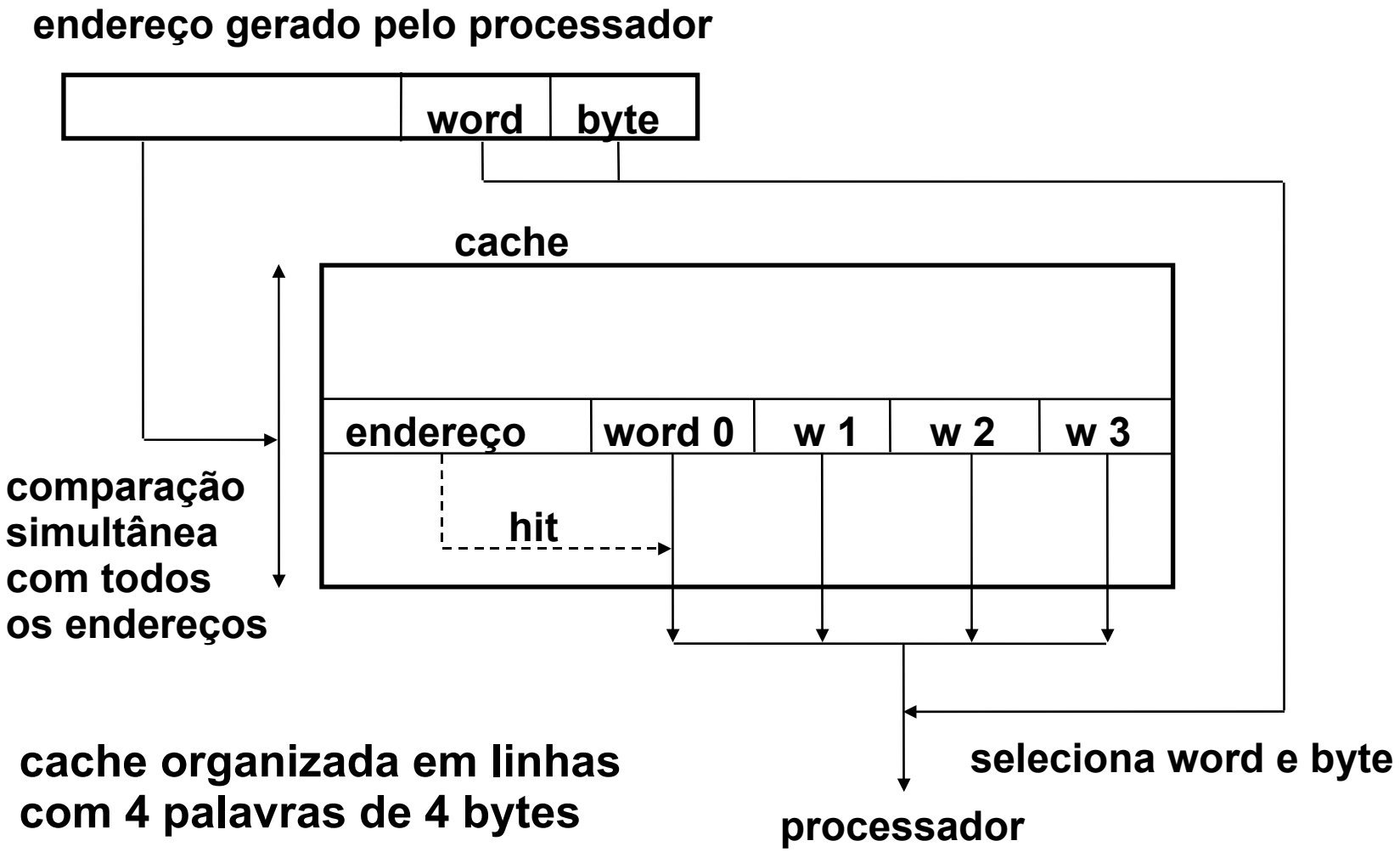
3. **Mapeamento conjunto - associativo**

1. Mapeamento Completamente Associativo

1. Mapeamento Completamente Associativo



Mapeamento completamente associativo



Mapeamento completamente associativo

- **Vantagem: máxima flexibilidade no posicionamento de qualquer palavra (ou linha) da memória principal em qualquer palavra (ou linha) da cache**

- **Desvantagens**
  - **custo em hardware da comparação simultânea de todos os endereços armazenados na cache**
  - **algoritmo de substituição (em hardware) para selecionar uma linha da cache como conseqüência de um miss**

- **Utilizado apenas em memórias associativas de pequeno tamanho**
  - **tabelas**

2. Mapeamento Direto

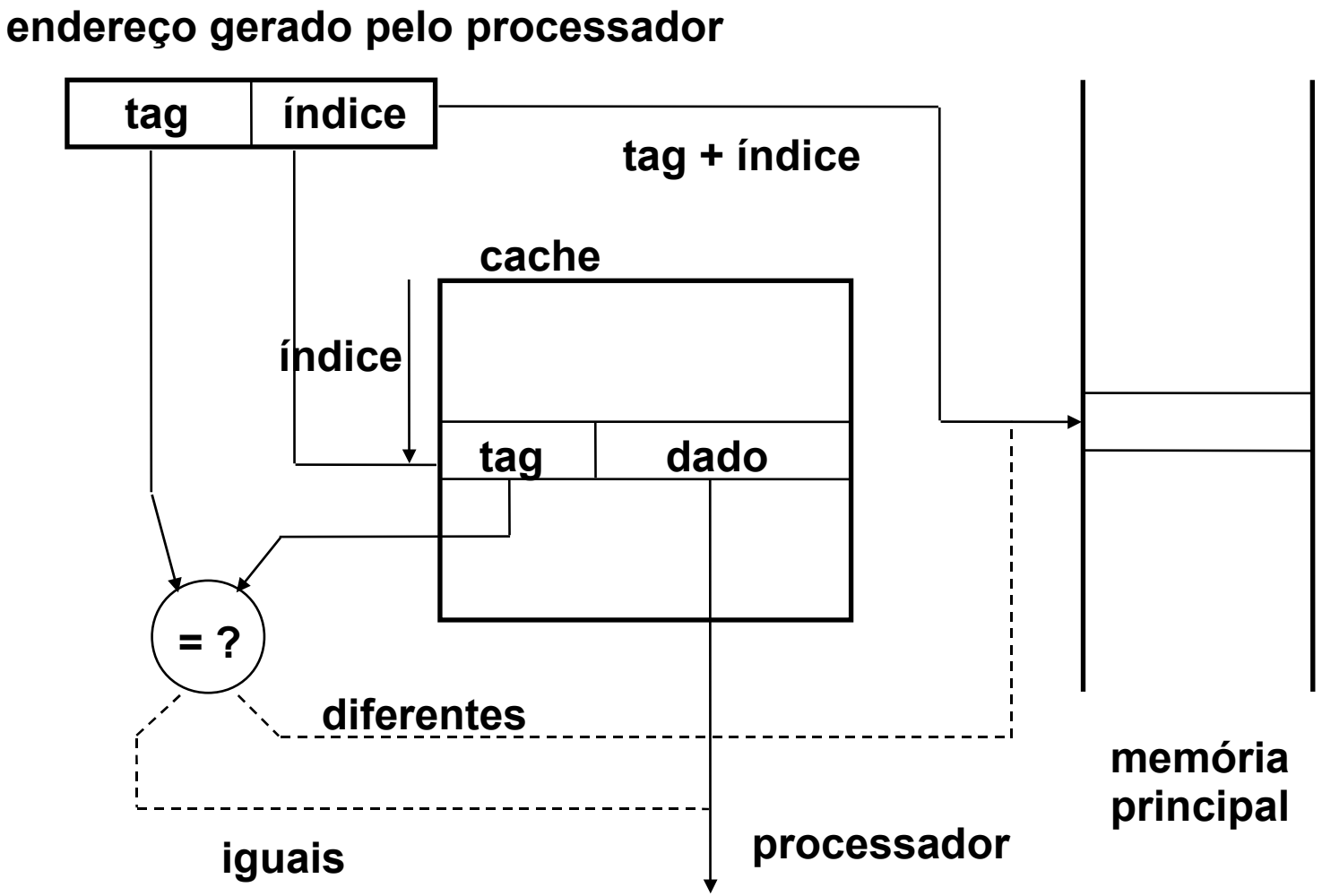
2. Mapeamento direto

- **Endereço é dividido em 2 partes**
  - **parte menos significativa: índice, usado como endereço na cache onde será armazenada a palavra**
  - **parte mais significativa: tag, armazenado na cache junto com o conteúdo da posição de memória**

- **Quando acesso é feito, índice é usado para encontrar palavra na cache**
  - **se tag armazenado na palavra da cache é igual ao tag do endereço procurado, então houve hit**

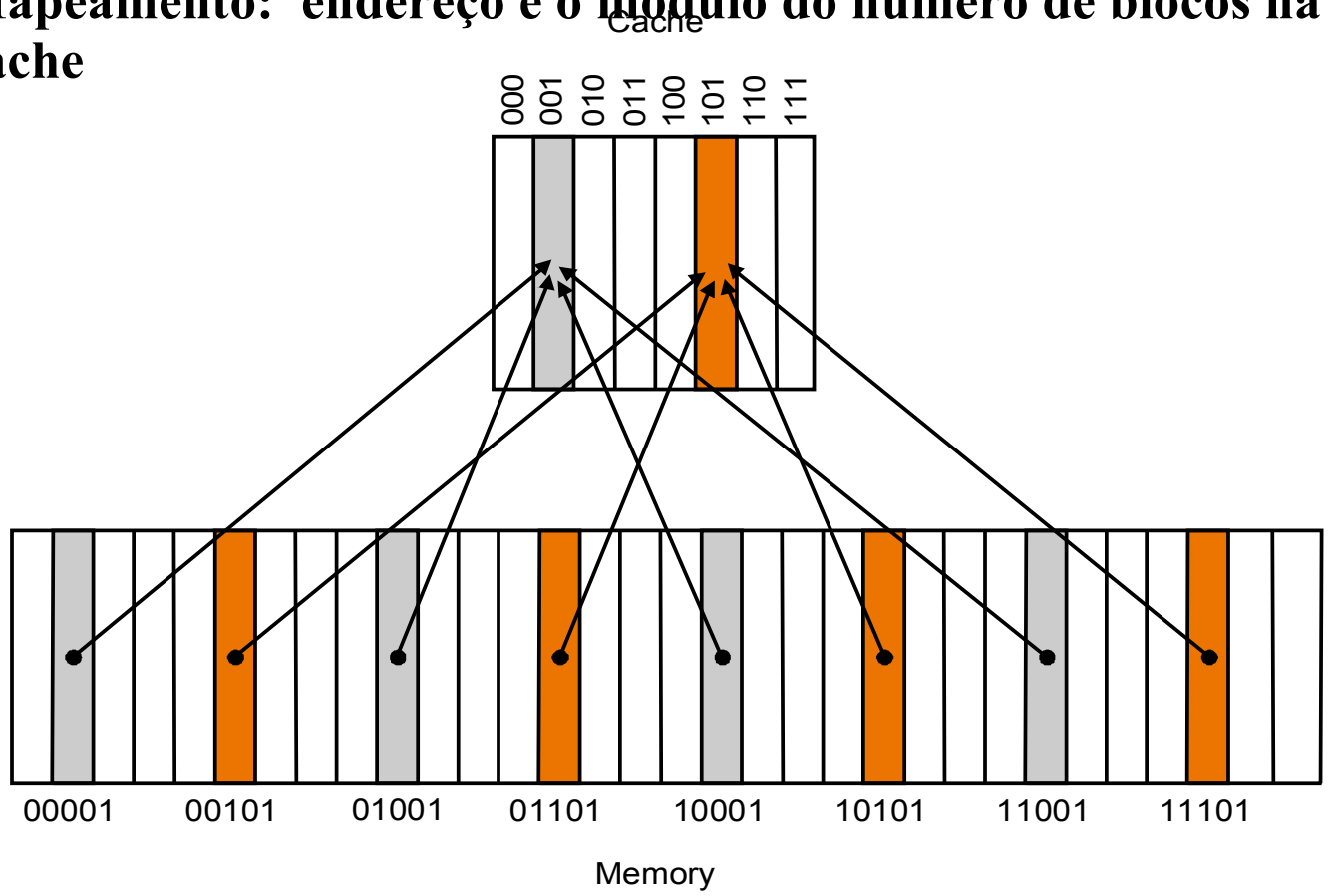
- **Endereços com mesmo índice são mapeados sempre para a mesma palavra da cache**

Mapeamento Direto

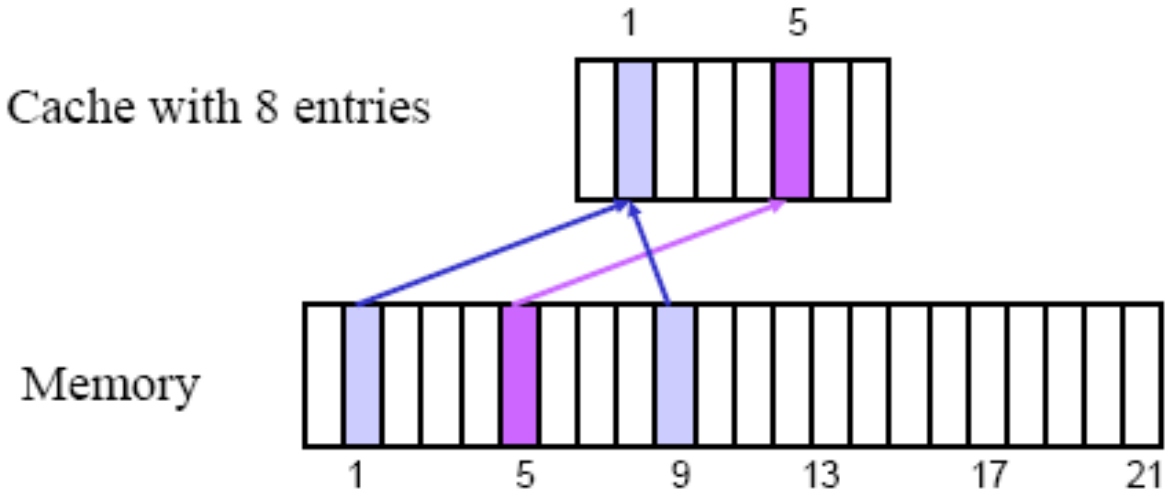


Mapeamento Direto

- **Mapeamento: endereço é o módulo do número de blocos na cache**

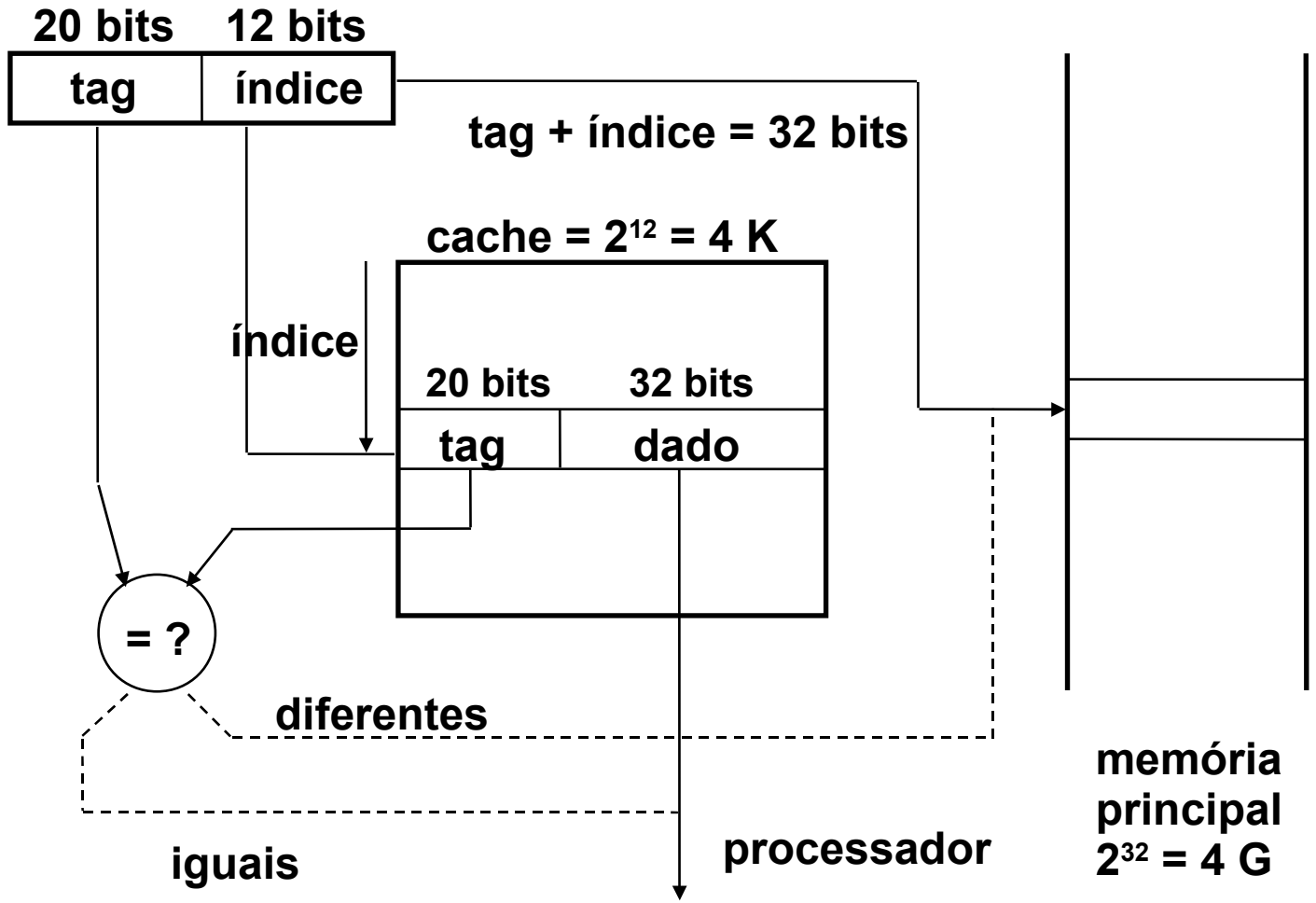


Mapeamento Direto

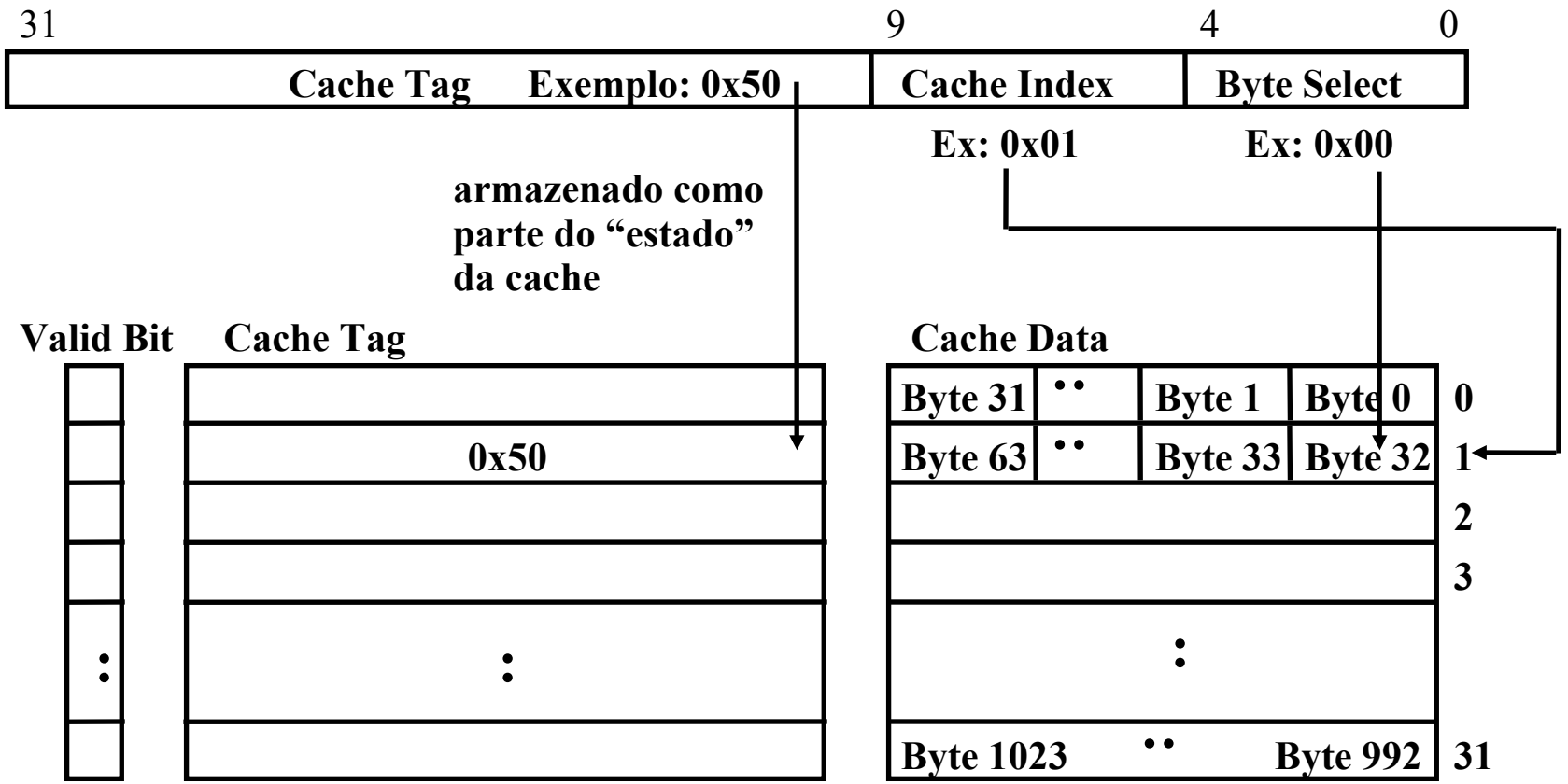


What cache block does memory address 9 map to?
Cache block word address = 9 mod 8 = 1
9 is 1001b, so cache block is 001b (1)

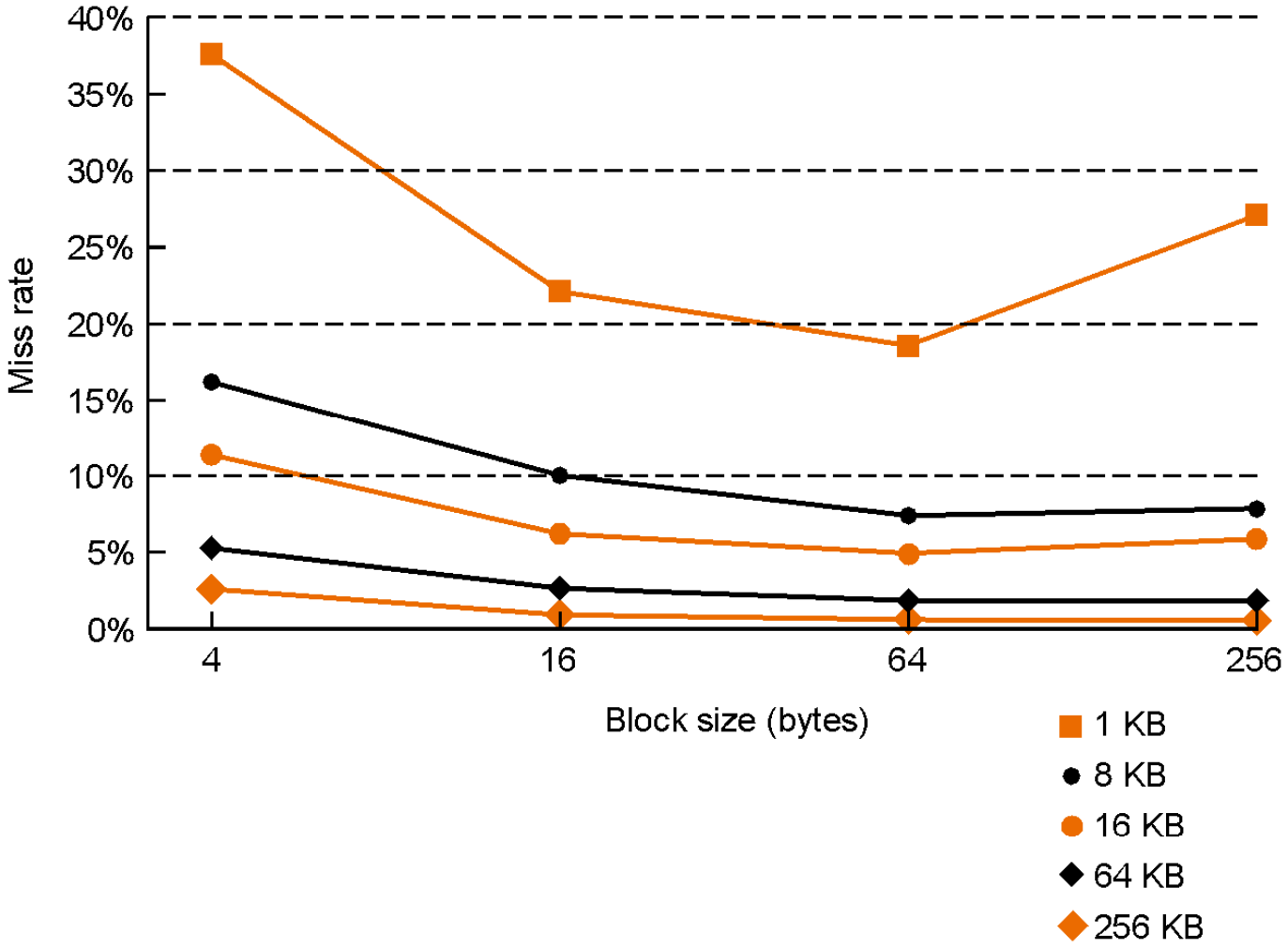
Mapeamento Direto – exemplo



Mapeamento Direto – uso de linhas



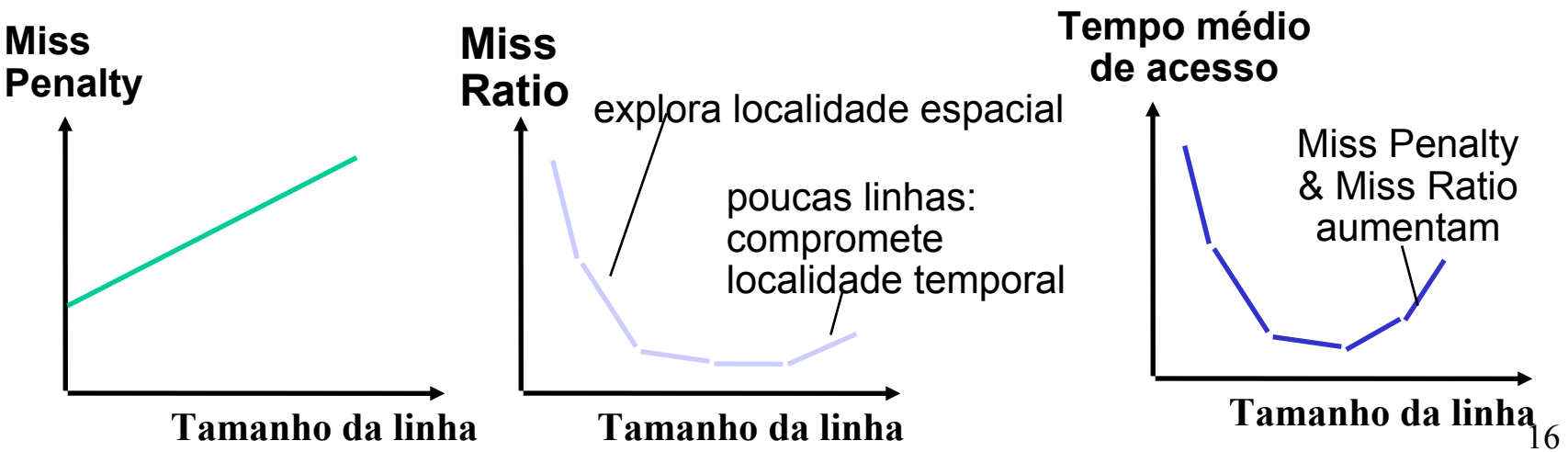
Tamanho da linha x *miss ratio*



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, endereços de 32 bits**

- **64 KB -> 16 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} x (32 + 32 -14 -2 +1) = 2^{14} x 49 = 784 x 2^{10} = 784 Kbits

- **98 KB para 64 KB de dados, ou 50% a mais**

Mapeamento Direto

- **Vantagens**
  - **não há necessidade de algoritmo de substituição**
  - **hardware simples e de baixo custo**
  - **alta velocidade de operação**

- **Desvantagens**
  - **desempenho cai se acessos consecutivos são feitos a palavras com mesmo índice**
  - ***hit ratio* inferior ao de caches com mapeamento associativo**

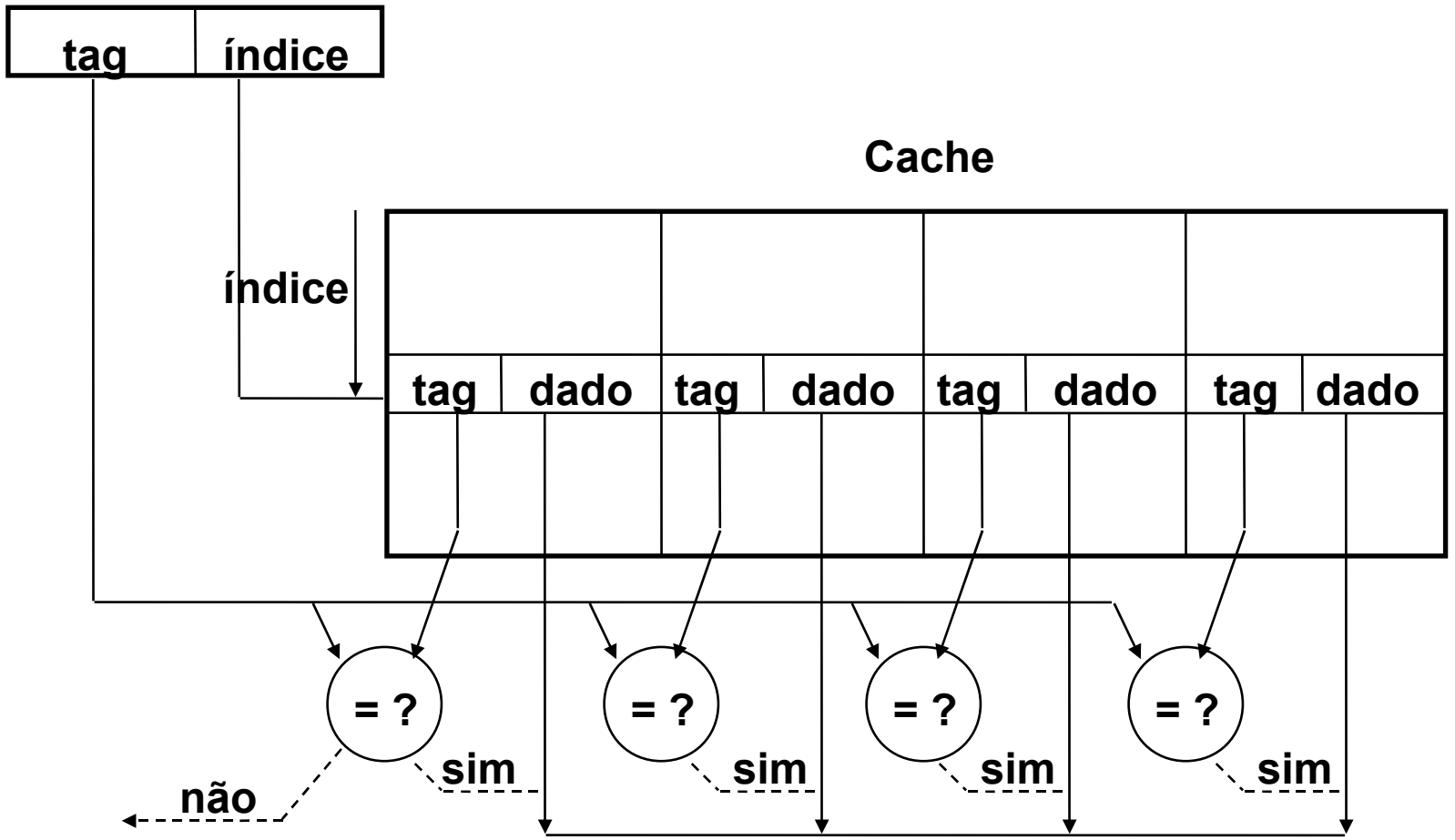
- **Demonstra-se no entanto que *hit ratio* aumenta com o aumento da cache, aproximando-se de caches com mapeamento associativo**
  - **tendência atual é de uso de caches grandes**

3. Mapeamento Conjunto – Associativo

3. Mapeamento conjunto – associativo

- **Mapeamento direto: todas as palavras armazenadas na cache devem ter índices diferentes**
- **Mapeamento associativo: linhas podem ser colocadas em qualquer posição da cache**
- **Compromisso: um nº limitado de linhas, de mesmo índice mas diferentes tags, podem estar na cache ao mesmo tempo (num mesmo conjunto)**
- **nº de linhas no conjunto = associatividade**

Mapeamento conjunto – Associativo



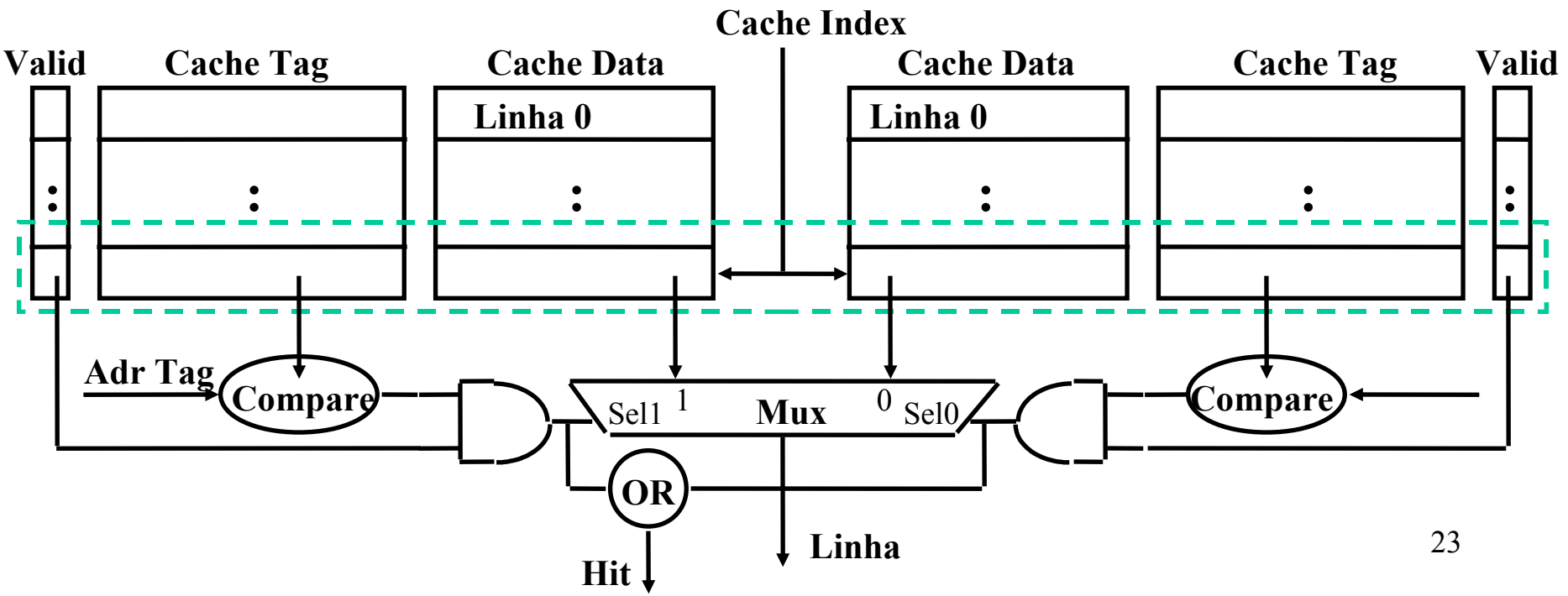
Mapeamento conjunto – Associativo

- **Vantagem em relação ao mapeamento completamente associativo: comparadores são compartilhados por todos os conjuntos**
- **Algoritmo de substituição só precisa considerar linhas dentro de um conjunto**
- **Muito utilizado em microprocessadores**
  - **Motorola 68040: 4-way set associative**
  - **Intel 486: 4-way set associative**
  - **Pentium: 2-way set associative**

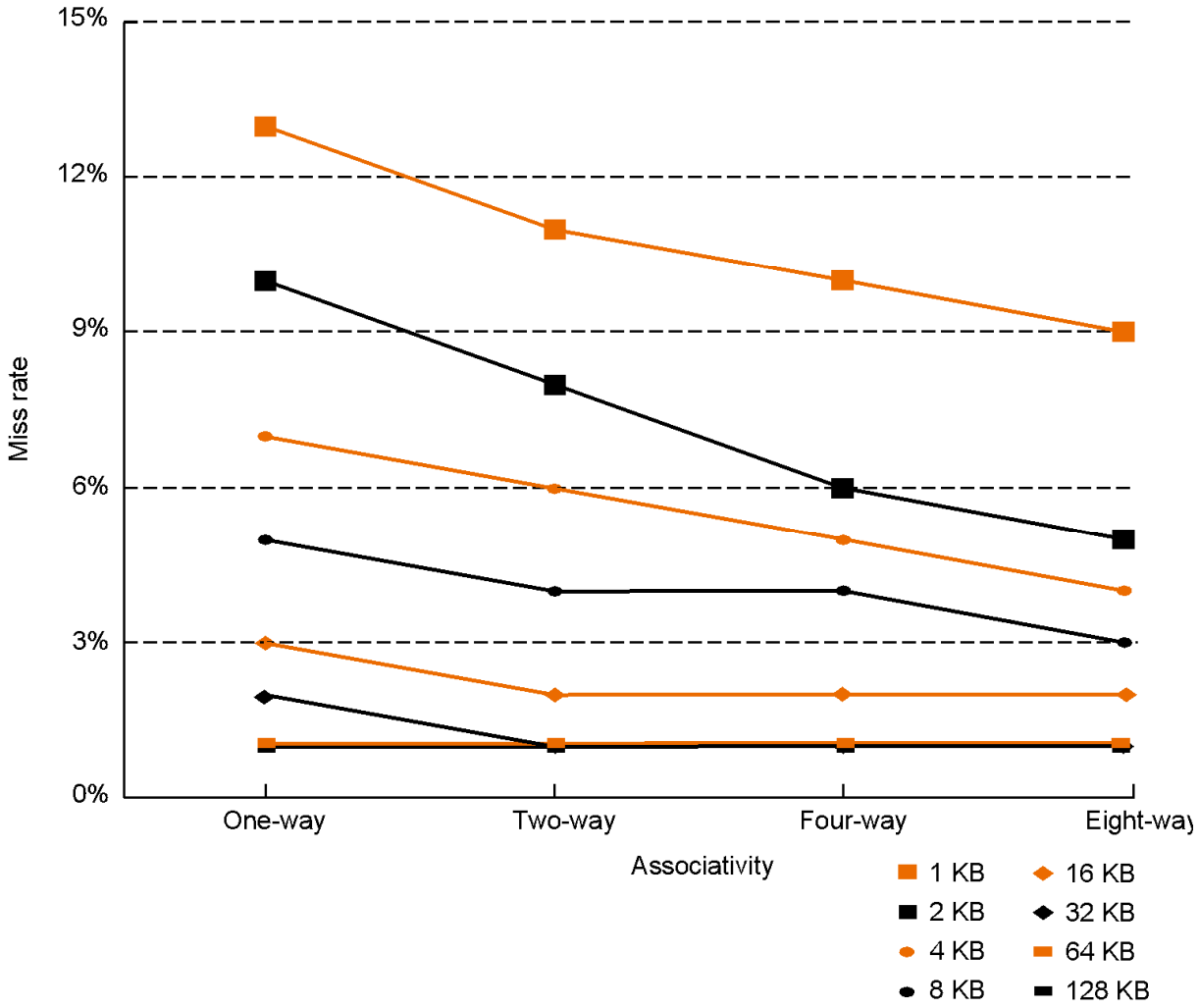
Desvantagem da cache conjunto-associativo

- **Conjunto-associativa N-way X mapeamento direto**
  - **dado tem atraso extra do multiplexador**
  - **dado vem DEPOIS da decisão *Hit/Miss* e da seleção do conjunto**

- **Numa cache com mapeamento direto, linha da cache está disponível ANTES da decisão *Hit/Miss***
  - **possível assumir um *hit* e continuar. Recuperar depois se for *miss*.**



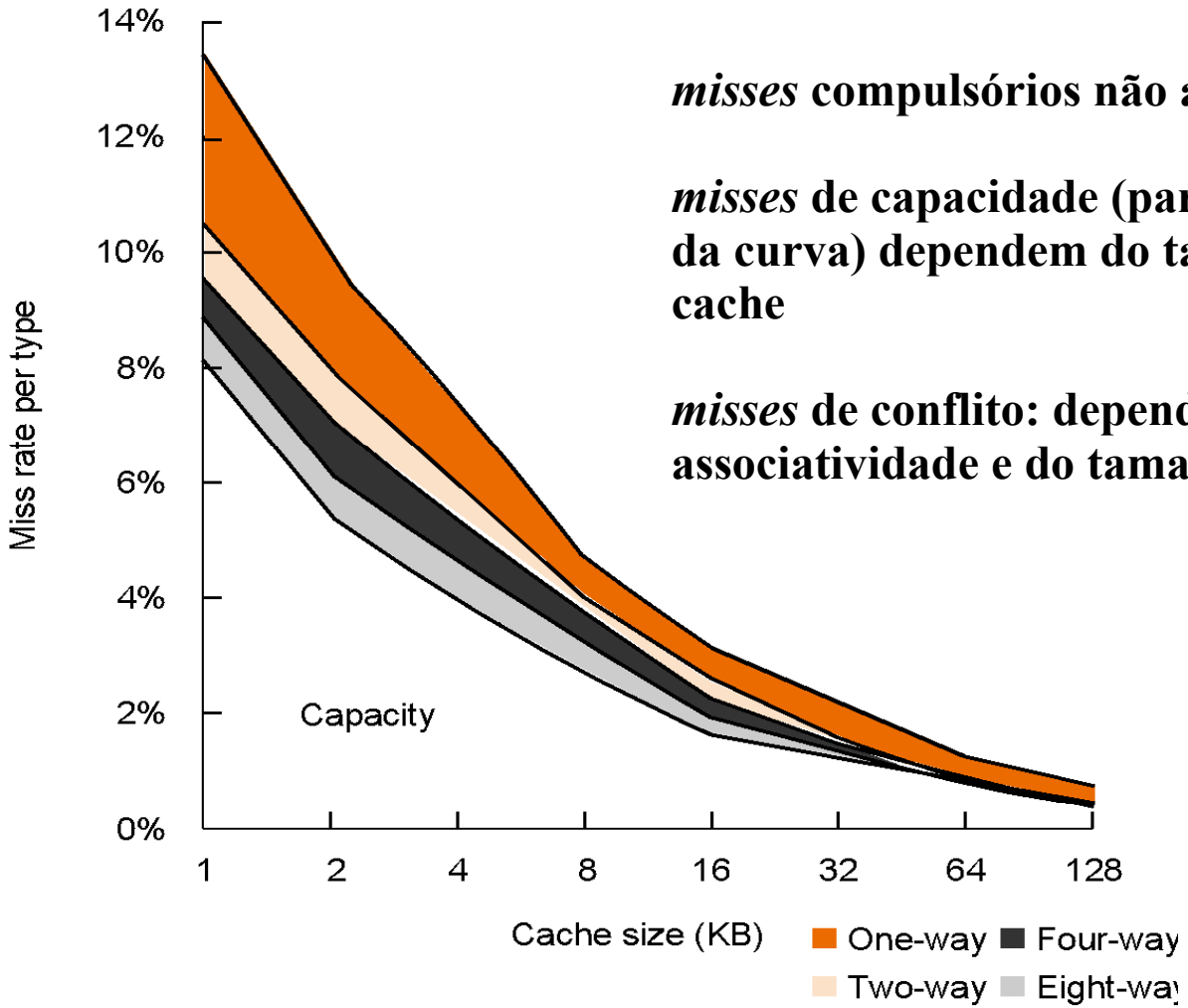
Impacto da associatividade da cache



Fontes de Misses

- **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**

Fontes de *Misses*



***misses* compulsórios não aparecem**

***misses* de capacidade (parte "branca" da curva) dependem do tamanho da cache**

***misses* de conflito: dependem da associatividade e do tamanho da cache**

Quantidade de *Misses* segundo a fonte

	Mapeam. direto	**Conj.-associat. N-way**	**Complet. associativa**
Tamanho da cache	Grande	Médio	Pequeno
***Misses* compulsórios**	Mesmo	Mesmo	Mesmo
***Misses* de conflito**	Alto	Médio	Zero
***Misses* de capacidade**	Baixo	Médio	Alto
***Misses* de invalidação**	Mesmo	Mesmo	Mesmo

4. Impacto no Desempenho

4. 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
H = Hit Ratio

Tce = Tc + (1 – h) Tm

se	**Tc = 1 ns,**	**Tm = 20 ns**			
	h =	**0.85**	**0.95**	**0.99**	**1.0**
		⇩	⇩	⇩	⇩
então	**Tce =**	**4 ns**	**2 ns**	**1.2 ns**	**1 ns**

Impacto no Desempenho

Tempo gasto com um *cache miss*, em número de instruções executadas

Proc.	IPC, Freq.	Latência/Clock =	Ciclos x IPC		Latência em Inst.
1º Alpha,	2 , 200MHz	- 340 ns / 5.0 ns =	68 clks x 2 instr.	ou	136 instruções
2º Alpha,	4 , 300MHz	- 266 ns / 3.3 ns =	80 clks x 4 instr.	ou	320 instruções
3º Alpha,	6 , 600MHz	- 180 ns / 1.6 ns =	108 clks x 6 instr.	ou	648 instruções

1/2 X latência x 3 X freqüência clock x 3 X instruções/clock $\Rightarrow \approx$ 5 X

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* e resultem numa penalidade de 50 ciclos**

CPI = CPI ideal + nº médio de stalls por instrução

= 1.1 ciclos + 0.30 acessos à memória / instrução

x 0.10 misses / acesso x 50 ciclos / miss

= 1.1 ciclos + 1.5 ciclos

= 2. 6

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**

FIM