

ORGANIZAÇÃO E ARQUITETURAS DE COMPUTADORES

Gerência de Memórias Cache (Mapeamento de Endereços)

Introdução a Caches

- **Funcionalidade**
 - Área de memória rápida e com informações dinâmicas
- **Cache só pode ter **parte** dos dados do nível mais abaixo**
 - Tamanho menor
- **Problemas**
 - Como identificar se o dado procurado está na cache?
 - Se estiver, como acessar de forma rápida?
 - Se não estiver, como buscar eficientemente de níveis inferiores?
 - Qual dado tirar da cache para colocar o novo dado?
- **Processador não sabe qual memória física tem o dado**
 - Gera apenas endereços e a hierarquia se encarrega de acessar a informação endereçada

Mapeamento de Endereços em Memória Cache

- Como fazer para pesquisar um dado na cache?
 - Fazer cache com todos endereços não faz sentido
 - Efetuar varredura seqüencial na cache leva muito tempo
- Solução
 - Fazer mapeamento de endereços
- Objetivo
 - Relacionar informações (dados e instruções) da memória principal com posições da cache
- Formas de mapeamento de memórias cache
 - Direto
 - Associativo
 - Conjunto associativo

Índice

1. Mapeamento de Endereços em Memória Cache

1.1 Mapeamento Direto

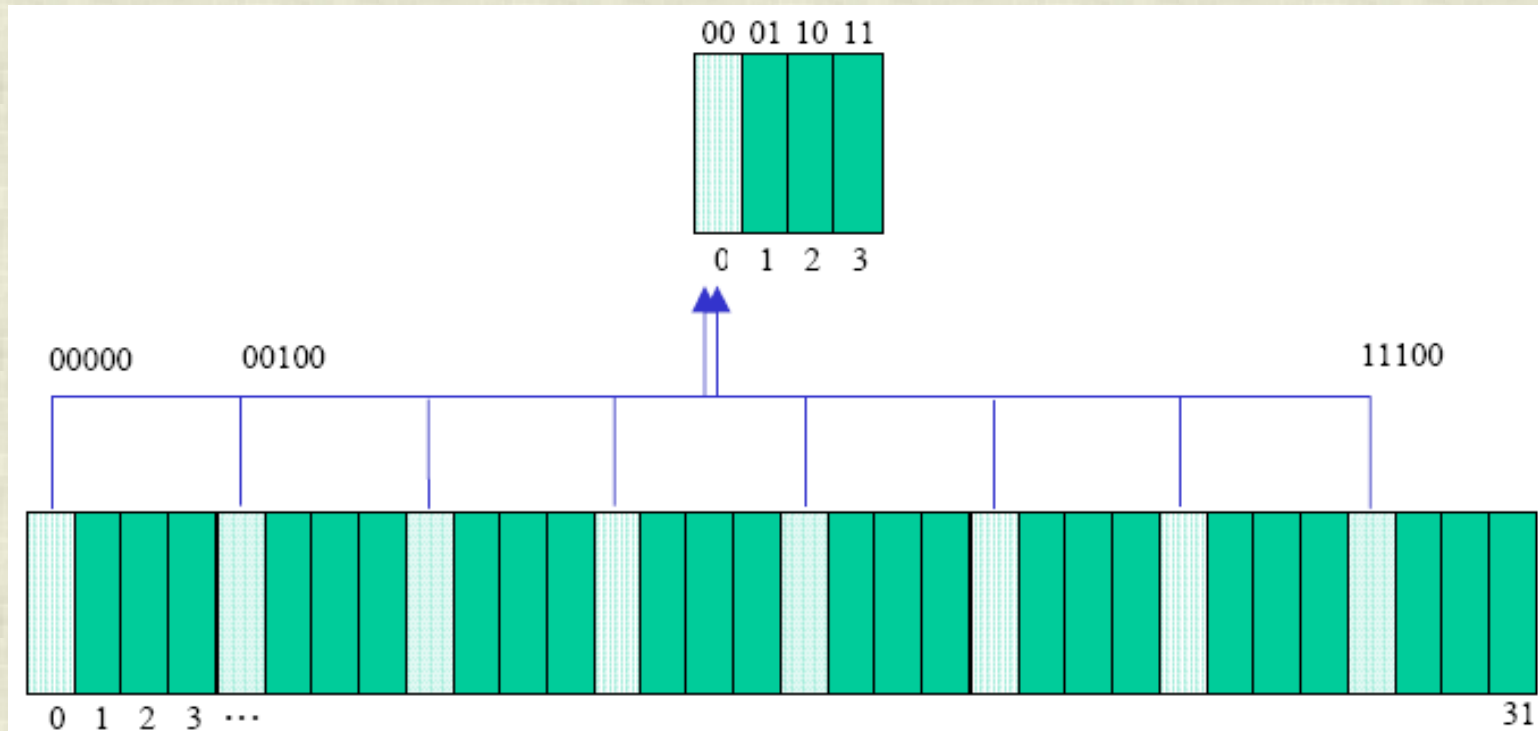
1.2 Mapeamento Associativo

1.3 Mapeamento Conjunto Associativo

Introdução ao Mapeamento Direto

- **Forma mais simples de mapeamento**
 - Posição na cache depende do endereço da palavra na memória principal (MP)
 - Cada palavra possui uma posição fixa
 - Grupo de palavras mapeado na mesma posição da cache
- **Exemplo**
 - Cache de 4 posições e MP de 32 endereços (palavra de 8 bits)
 - Cada posição da cache tem 1 de 8 posições da MP
 - Endereço obtido pelo resto da divisão inteira do número de posições da cache
 - Mapeamento utilizando os dois bits menos significativos do endereço

Esquema de Mapeamento Direto



- TAG (rótulo) para cada posição da cache → identifica qual das palavras está na cache
- Bit de validade → posição da cache está ocupada ou contém lixo
- Poderiam ser usados os bits mais significativos do endereçamento ao invés dos bits menos significativos? Qual a consequência?

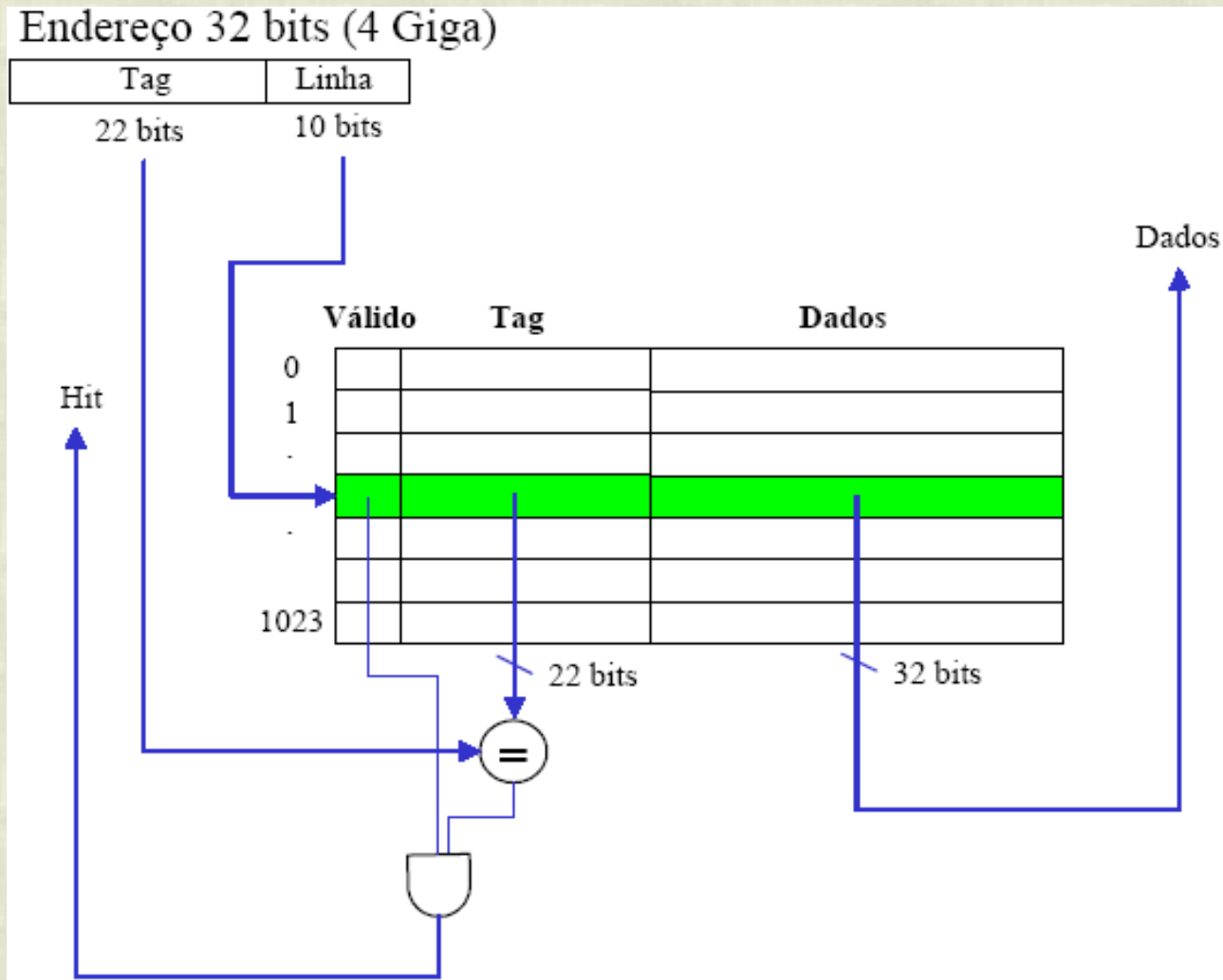
Bit de Validade	<u>Taq</u>	<u>Dado</u>
1	001	00110110
0		
0		
1	000	11100011

Acesso a Cache com Mapeamento Direto

- **Passos para um acesso**
 1. **Calcular o mod do endereço pelo número de posições da cache → Ex. usar os bits menos significativos do endereço**
 2. **Se bit de validade da posição for válido**
 - Verificar Tag**
 - Senão**
 - Acusar miss**
 - Ir para 4**
 - 3. **Se Tag diferente de endereço**
 - Acusar miss**
 - Ir para 4**
 - Senão**
 - Ler posição (hit)**
 - Ir para 7**
 4. **Buscar dado no nível inferior**
 5. **Colocar na posição**
 6. **Efetuar leitura**
 7. **Fim**

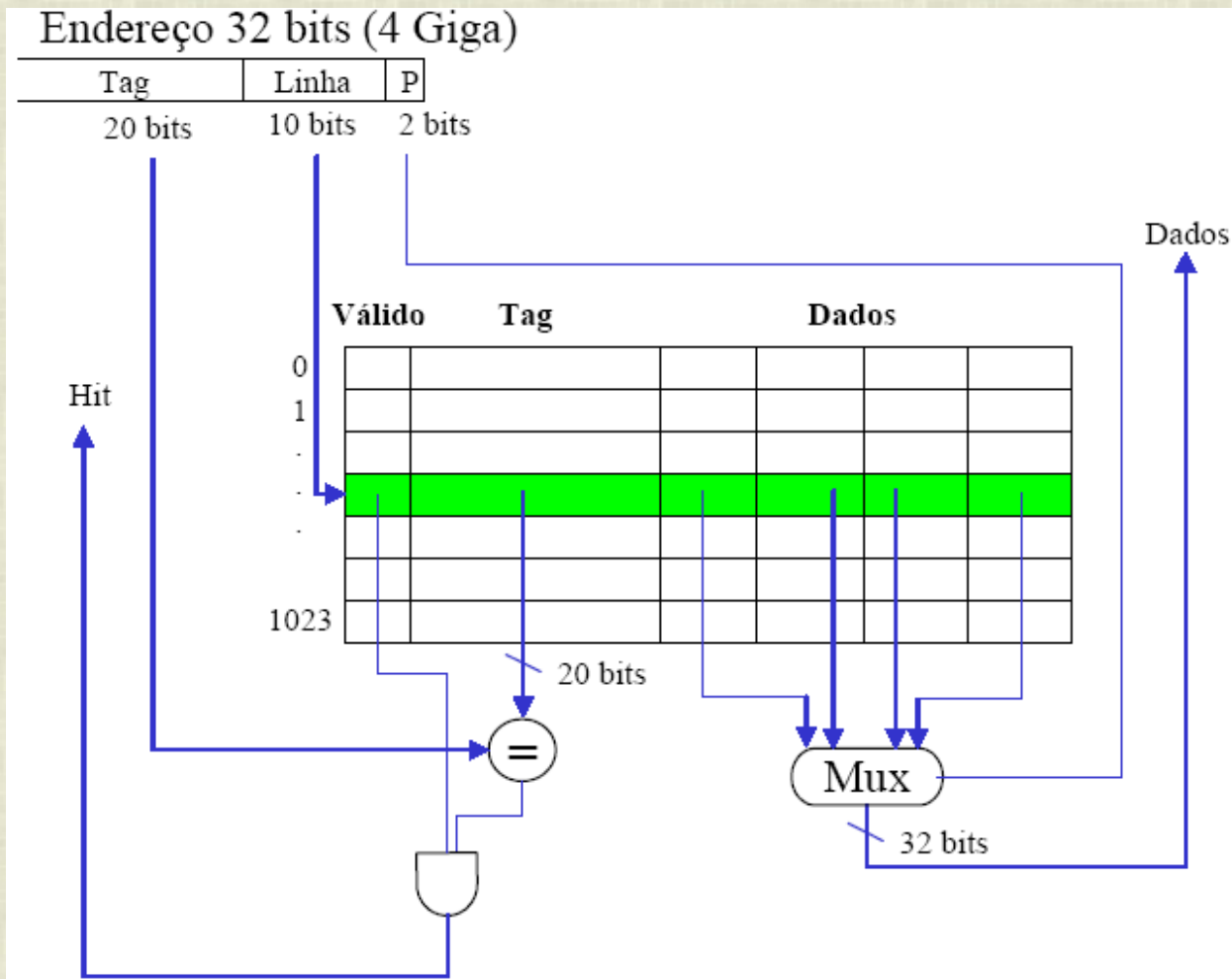
Mapeamento Direto – Bit validade e Tag

- Divisão de bits no registrador de endereçamento
 - Exemplo de uma cache com 1024 linhas (2^{10}) com palavra de 32 bits



Mapeamento Direto – Transferência de Blocos

- Transferência de blocos entre níveis de memória
 - Exemplo da divisão de blocos em uma cache com 1024 linhas (bloco com 4 palavras de 32 bits)



O que ganho ao utilizar blocos ao invés de palavras?

Mapeamento Direto - Exercícios

1. Considerando um espaço de endereçamento de 1 Giga. Como ficaria a divisão de bits para uma cache de 2048 posições que trabalhe com blocos de 8 palavras?

Endereço de 30 bits (1 Giga)		
16 (Tag)	11 (Linha)	3 (Palavra)

1. Quanto tem efetivamente de dados nessa cache, considerando palavras de 32 bits?

Linha da <i>Cache</i>		
1 (validade)	16 (tag)	8*32 (bloco de dados)

(Bit de validade + Tag + Dados) / Dados

$$(1 + 16 + (8*32)) / (8*32)$$

$$273 \text{ bits} / 256 \text{ bits}$$

$$256 / 273 * 100\% = 93.77\%$$

Mapeamento Direto - Exercícios

1. Supor as seguintes características:

- Memória principal de 2GBytes, com palavras de 32 bits
- Cache com 64Kbits
- Blocos de 16 palavras

Mostre como ficará distribuída a cache e o formato da palavra de endereço

Mapeamento Direto - Conclusões e Questões

- **Vantagens do mapeamento direto**
 - Hardware barato
 - Procura simples (posição fixa)
 - Não existe escolha da vítima (é dada pelo módulo)
 - Simplicidade / Velocidade
- **Desvantagens do mapeamento direto**
 - Pode ter mau aproveitamento das posições da cache (dependendo dos endereços gerados)
 - Usa parte da cache para controle
- **Como melhorar o mapeamento apresentado?**
- **Como retirar dependência entre endereço na memória e posição da cache sem comprometer desempenho da procura?**

Índice

1. Mapeamento de Endereços em Memória Cache

1.1 Mapeamento Direto

1.2 Mapeamento Associativo

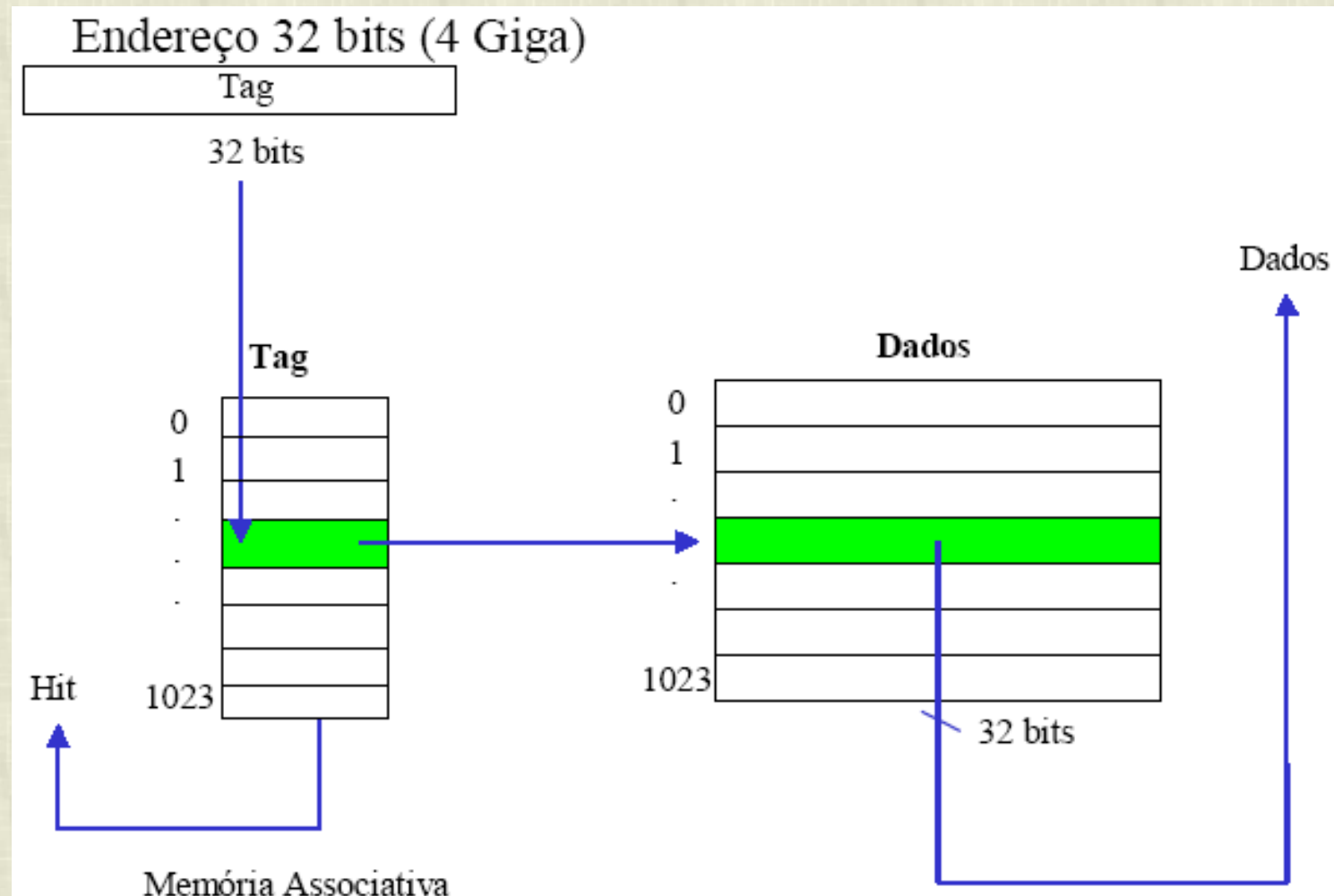
1.3 Mapeamento Conjunto Associativo

Introdução ao Mapeamento Associativo

- **Endereço da MP em qualquer posição da cache**
 - Tag não fica mais na cache e sim em memória especial (memória associativa)
 - Pode ter bit de validade ou usar tag com valor inválido para determinar se posição tem uma informação válida
- **Conseqüências**
 - Necessita fazer procura de dado
 - Necessita política de substituição
 - Quando ocorre miss
 - Buscar no nível abaixo e, caso a cache esteja com todas posições ocupadas, quem tirar para abrir lugar?
- **Solução para procurar**
 - Procurar em paralelo
 - Usar memória associativa
 - Memória cara e de tamanho limitado

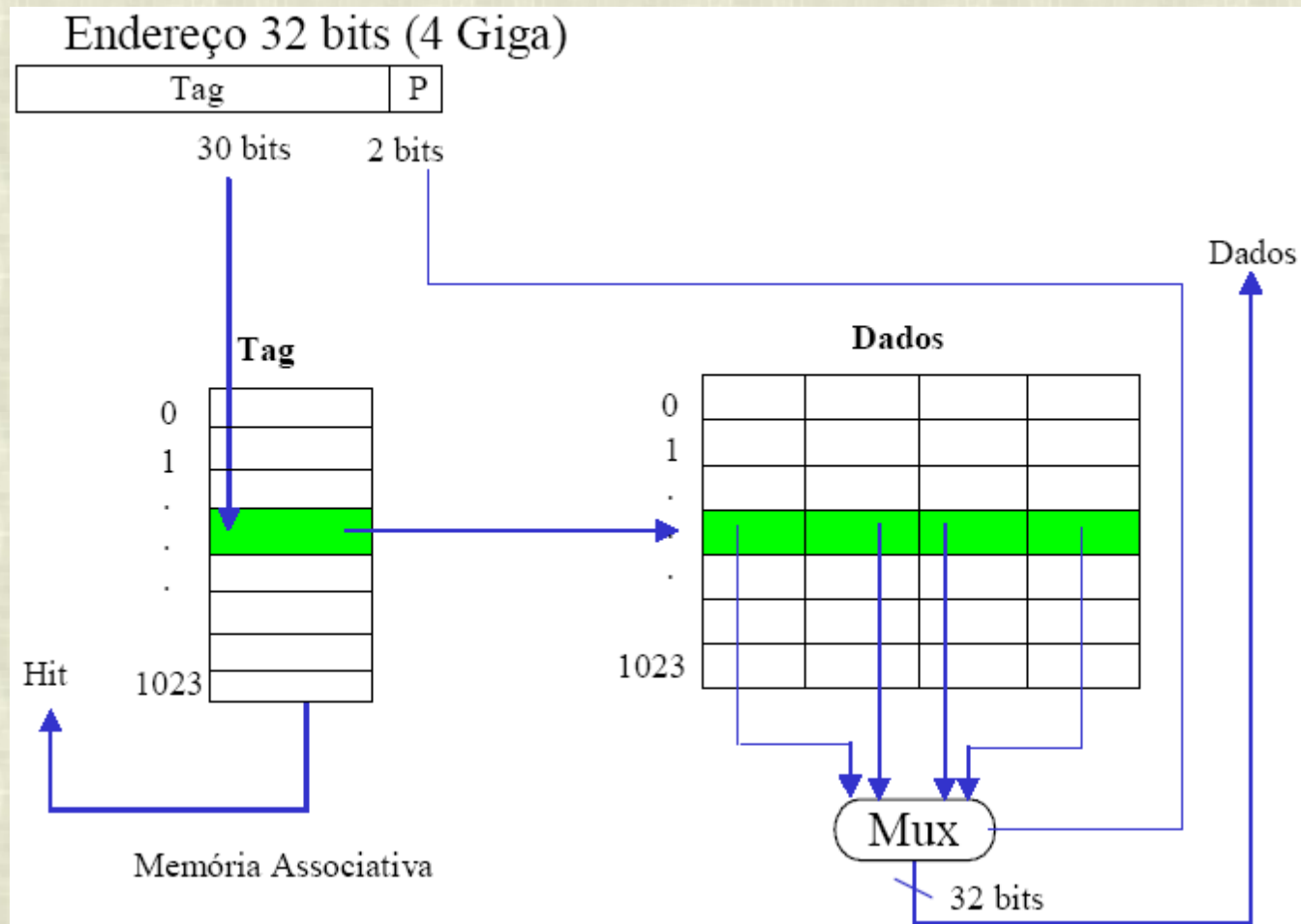
Esquema Básico de Mapeamento Associativo - Tag

- Divisão de bits no registrador de endereçamento
 - Exemplo de uma cache com 1024 posições (2^{10}) com palavra de 32 bits



Mapeamento Associativo – Transferência de Blocos

- Transferência de blocos entre níveis de memória
 - Exemplo da divisão de blocos em cache de 1024 linhas (bloco com 4 palavras de 32 bits)



Mapeamento Associativo – Substituição de Dados na Cache

- Usar políticas

- **Randômica**

- Escolhe aleatoriamente posição a ser substituída → Simples, mas sujeito a aumentar número de caches miss

- **Contador**

- Um contador baseado em algum clock aponta para a próxima posição a ser substituída → simples, mas igualmente à política randômica, está sujeita a aumentar o número de misses

- **LFU (Least Frequent Used)**

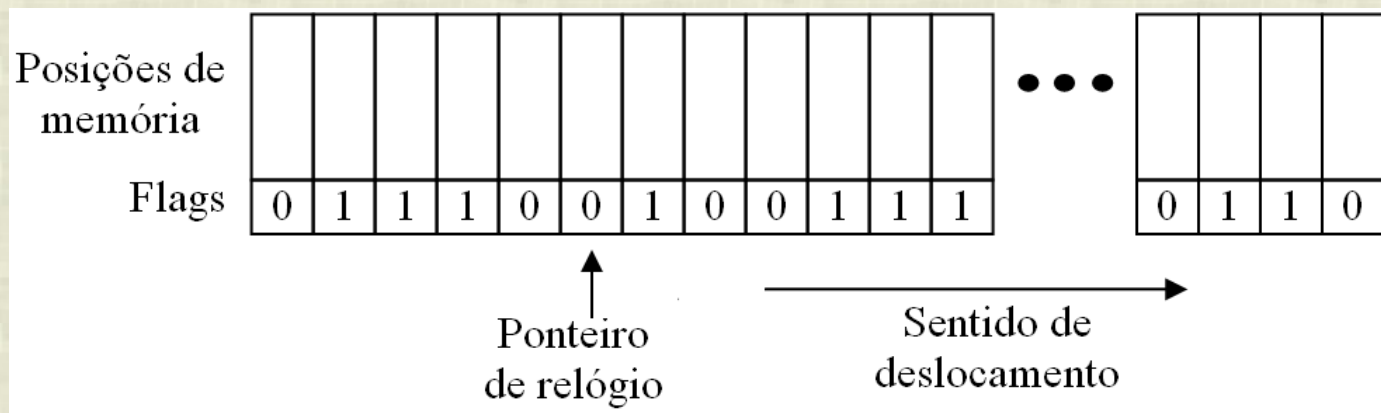
- Posição da cache menos usada é substituída → Escolha precisa mecanismo de contagem para cada acesso

- **LRU (Least Recent Used)**

- Posição da cache usada a mais tempo será substituída → Escolha precisa variável de tempo a cada acesso e comparação

Mapeamento Associativo – Algoritmo do Relógio

- Técnica simples para pesquisar endereço de memória cache visitado há mais tempo. Permite a implementação do critério de substituição por tempo
- A técnica, na verdade, não retorna a posição há mais tempo visitada, mas é simples, fornecendo posições visitadas há mais tempo
- Cada posição da cache tem associado um bit (flag) que informa se a mesma foi ou não recentemente visitada. Quando posição de memória é acessada, a flag é escrita
- Um ponteiro (*relógio*) que caminha nos endereços da cache em uma determinada base de tempo, vai apagando as flags na medida em que passa por elas
- Se uma posição de memória não tem a flag escrita é considerada que ela foi acessada há mais tempo e pode ser descartada. Esta informação fica em uma pequena lista no gerente da hierarquia de memória



- O algoritmo pode ser melhorado substituindo flags por contadores. Cada vez que o relógio passa, contador é decrementado, e cada vez que a posição de memória é acessada, contador volta para valor máximo

Acesso a Cache com Mapeamento Associativo

- **Passos para um acesso**
 1. Alimentar memória associativa com Tag procurado (comparação)
 2. Se Tag está na memória associativa
 - (Hit) Acessar **memória cache** com **índice** fornecido pela **memória associativa**
Ir para 7
 - Senão
Acusar miss
 1. Se não existir posição livre na cache
 - Escolher endereço para substituir de acordo com política estabelecida (e.g. LRU)
 1. Buscar dado no nível inferior
 2. Colocar na posição livre ou escolhida da **memória cache**
 3. Cadastrar posição na **memória associativa** para pesquisas futuras
 4. Efetuar leitura
 5. Fim

Mapeamento Associativo - Exercícios

1. Considerando espaço de endereçamento de 256 Mega palavras. Como ficaria a divisão de bits do endereço para uma **cache** de 2048 posições e que trabalhe com blocos de 8 palavras?

Endereço de 28 bits (256 Mega)	
25 (Tag)	3 (Palavra)

1. Quanto tem efetivamente de dados nessa cache?

100% (não considerando bit de validade)

1. Qual o tamanho da **memória associativa**?

$$\begin{aligned}\text{Tamanho} &= 2048 * 25 \text{ (tag)} \\ &= 51200 \text{ bits} / 8 \\ &= 6400 \text{ bytes} / 1024 \\ &= 6,25 \text{ Kbytes}\end{aligned}$$

Mapeamento Associativo - Conclusões e Questões

- **Vantagens do mapeamento associativo**
 - Melhor distribuição da informação na cache
 - Melhor aproveitamento da cache → Praticamente 100% de aproveitamento
 - Tags não ocupam espaço da cache (estão na memória associativa)
 - **Desvantagens**
 - Memória associativa tem alto custo e tamanho limitado
 - Limita número de linhas da cache
 - Necessita política de substituição
 - Pode ocorrer escolhas inadequadas
 - Gasta tempo
-
- **Limite de tamanho** da cache devido pesquisa na memória associativa é uma restrição muito forte → **Tendência é aumentar a cache**
 - Qual seria outra possibilidade?

Índice

1. Mapeamento de Endereços em Memória Cache

1.1 Mapeamento Direto

1.2 Mapeamento Associativo

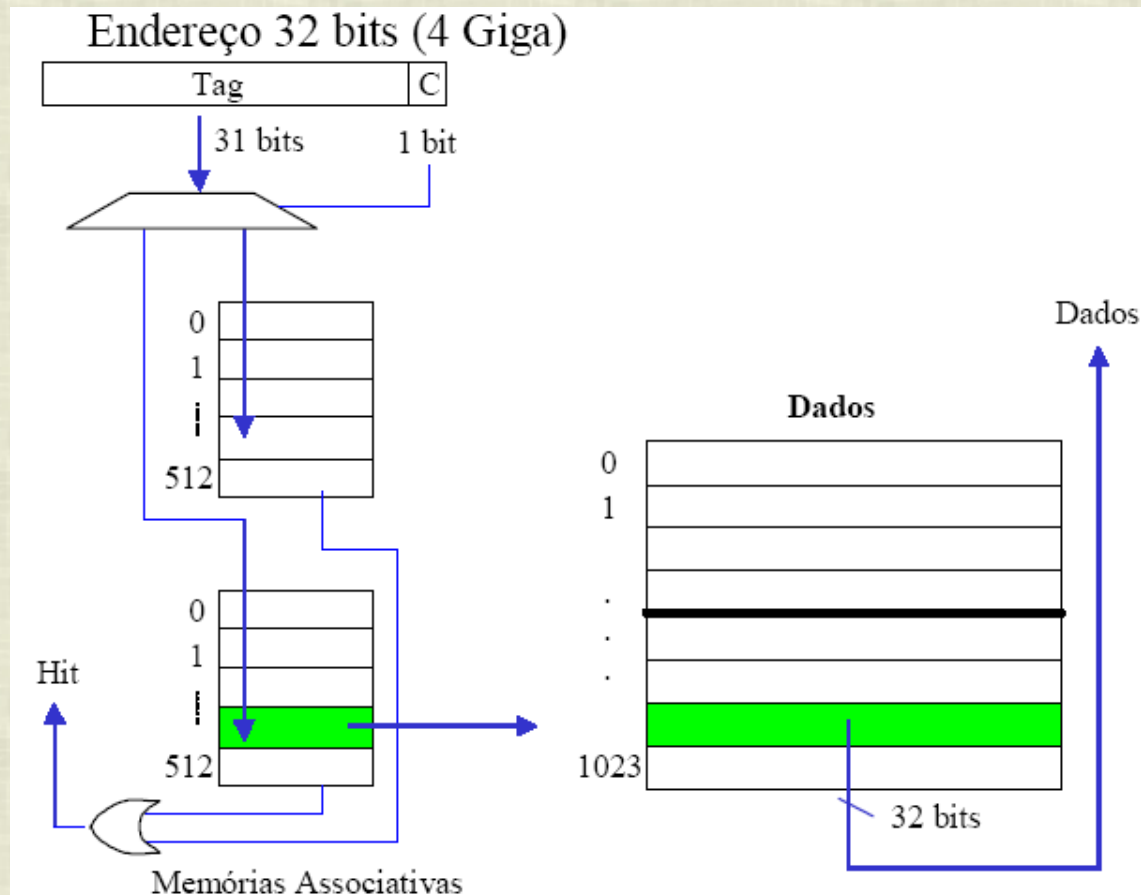
1.3 Mapeamento Conjunto Associativo

Introdução ao Mapeamento Conjunto Associativo

- Compromisso entre mapeamento direto e totalmente associativo
- Cache dividida em **S** conjuntos de **N** blocos/palavras
 - Se **S** = 1 → mapeamento associativo
 - Se **S** = número de blocos/palavras da cache → mapeamento direto
- Pesquisa dentro do conjunto
 - Endereço **i**, da memória principal, pode mapear para endereços no conjunto **i mod S**, da cache
- Necessita de política de substituição quando ocorre cache miss
 - Caso conjunto esteja cheio quem tirar?
 - A substituição deve ser necessariamente dentro do conjunto

Mapeamento Conjunto Associativo - Tag

- Divisão de bits no registrador de endereçamento
- Exemplo de cache com 1024 posições (2^{10}) com palavra de 32 bits e 2 conjuntos (S=2)



Acesso com Mapeamento Conjunto Associativo

- **Passos para um acesso**
 1. Calcular resto da divisão inteira do endereço pelo número de conjuntos **S** → Ex. usar bits menos significativos de endereço
 2. Alimentar memória associativa do conjunto com Tag (Comparação)
 3. Se Tag está na **memória associativa**
 - (Hit) Acessar **memória cache** com índice fornecido pela **memória associativa**
 - Ir para 8
 - Senão
 - Acusar miss
 1. Se não existir posição livre no conjunto cache
 - Escolher endereço para substituir de acordo com política estabelecida
 1. Buscar dado no nível inferior
 2. Colocar dado na posição livre ou escolhida da **cache**
 3. Cadastrar posição na **memória associativa** para pesquisas futuras
 4. Efetuar leitura
 5. Fim

Mapeamento Conjunto Associativo - Exercícios

- Considerando uma cache com 1024 posições (2^{10}) com palavra de 32 bits, um espaço de endereçamento de 4 Gigabytes (2^{32}), dois conjuntos associativos e Tag com endereço inválido para marcar dados inválidos, responda as perguntas que seguem

1. Quanto tem efetivamente de dados nessa cache?

100%

1. Qual tamanho de cada memória associativa?

$$\text{Tamanho} = 1024 \text{ (linhas)} / 2 \text{ (cj.)} * 31 \text{ (tag)}$$

$$= 512 * 31 \text{ bits} = 15872 \text{ bits}$$

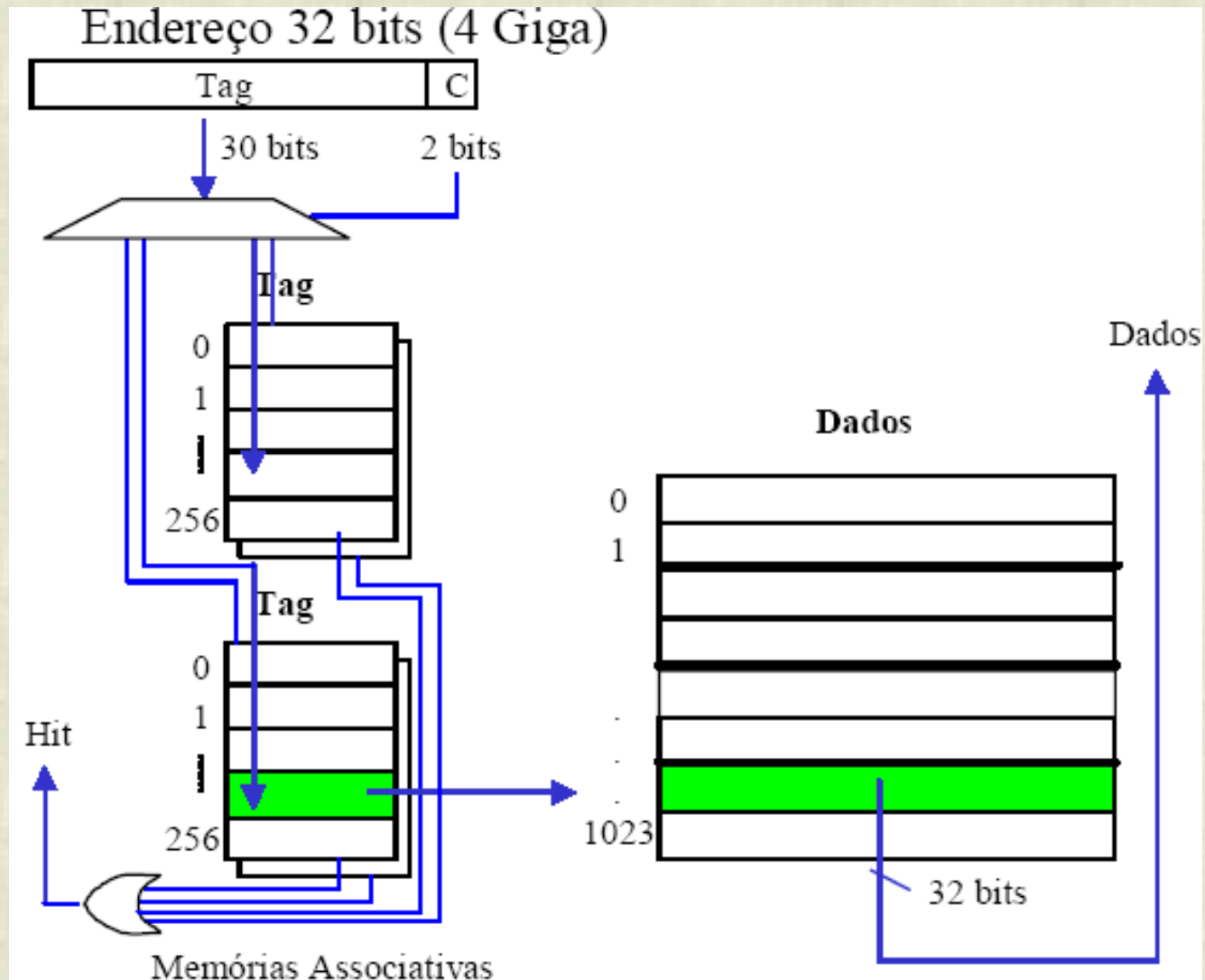
$$= 15872 \text{ bits} / 8 = 1984 \text{ bytes}$$

$$= 1984 \text{ bytes} / 1024 = 1,93 \text{ Kbytes}$$

1. Como ficaria a mesma cache com 4 conjuntos ($S=4$)?

Mapeamento Conjunto Associativo - Exercícios

- Esquema com 4 conjuntos associativos



Mapeamento Conjunto Associativo - Exercícios

1. Qual tamanho de cada uma das 4 memórias associativas?

$$\begin{aligned}\text{Tamanho} &= 256 * 30 \text{ (tag)} \\ &= 7680 \text{ bits} / 8 \\ &= 960 \text{ bytes} / 1024 \\ &= 0,94 \text{ Kbytes}\end{aligned}$$




1. Como ficaria divisão do endereço de cache de 4 conjuntos com 2048 posições (**bloco** com 4 palavras de 32 bits)?

Endereço de 32 bits (4 Giga)

28 (Tag)	2 (Conjunto)	2 (Palavra)
----------	--------------	-------------

Mapeamento Conjunto Associativo – Níveis de Associatividade

- Associatividade da cache é dada pelo número de vias (ways)
- Exemplo: cache de 8 posições pode ter 1 a 8 vias

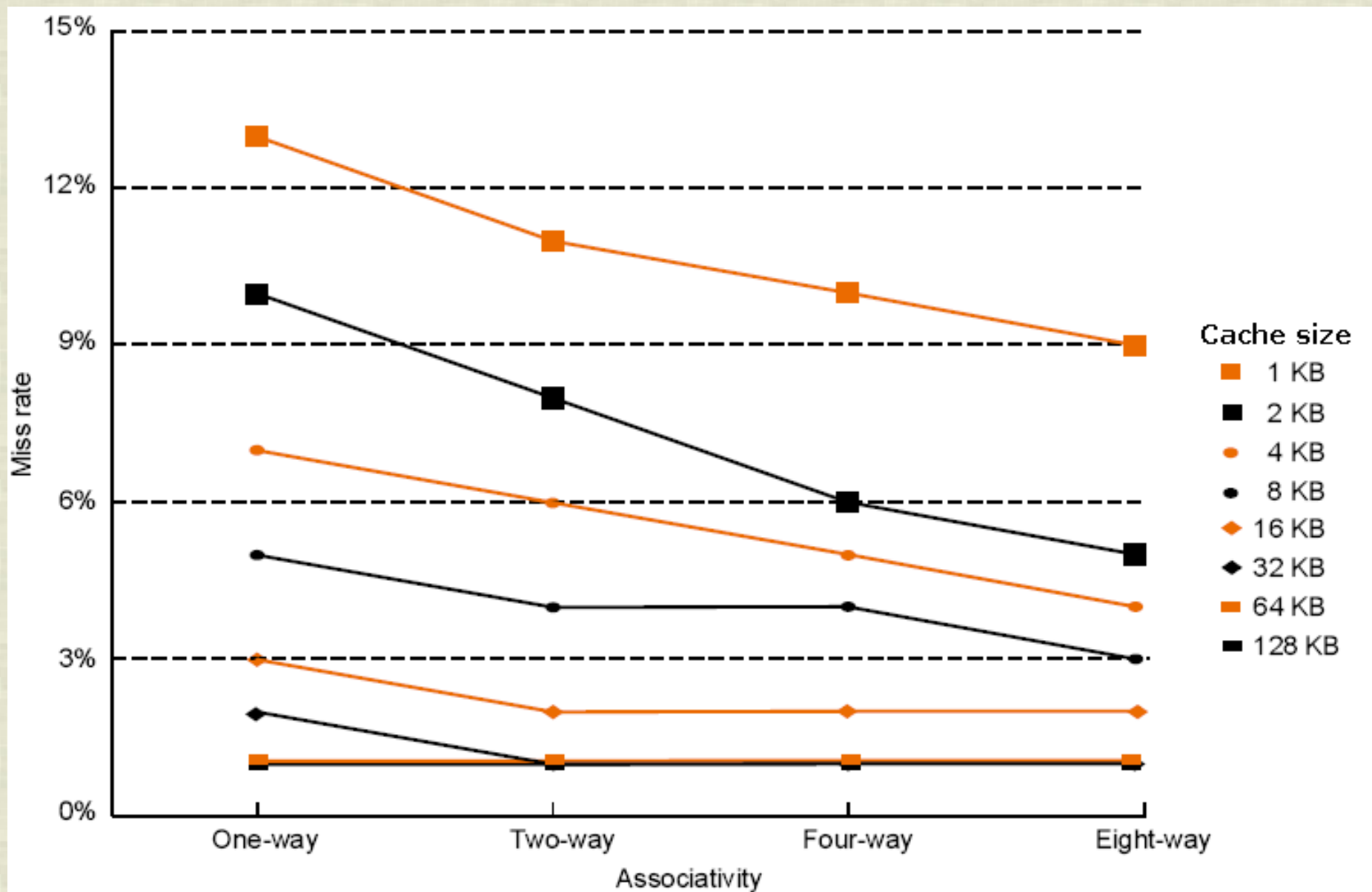
Vias (Ways)	Mapeamento	Esquema
1	Direto	
2	Conjunto-associativo (4 cj)	
4	Conjunto-associativo (2 cj)	

- Quantos conjuntos Associativa cache L1 4-Way de 64 Kbytes e com processador Ultra Sparc III (assumir bloco de 32 palavras de 64 bits)?

Tamanho da linha da cache = $32 * 64 \text{ b} = 2048 \text{ b} = 256 \text{ B} = 0,25 \text{ KB}$

$64 \text{ KB} / 0,25 \text{ KB} = 256 \text{ linhas} / 4 \text{ (4-Way} \rightarrow 4 \text{ linhas por cj)} = 64 \text{ cj}$

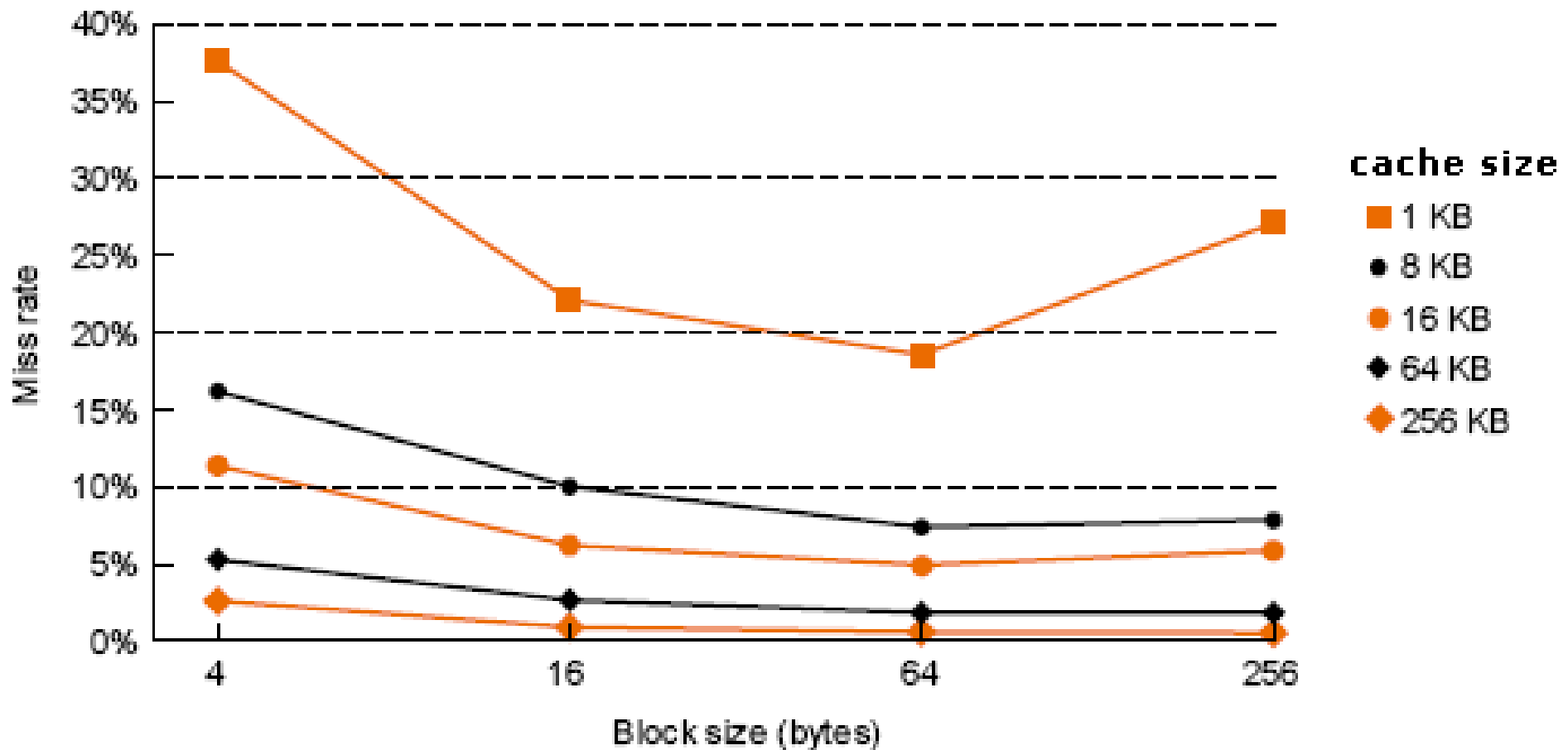
Mapeamento Conjunto Associativo – Níveis de Associatividade e Miss Rate



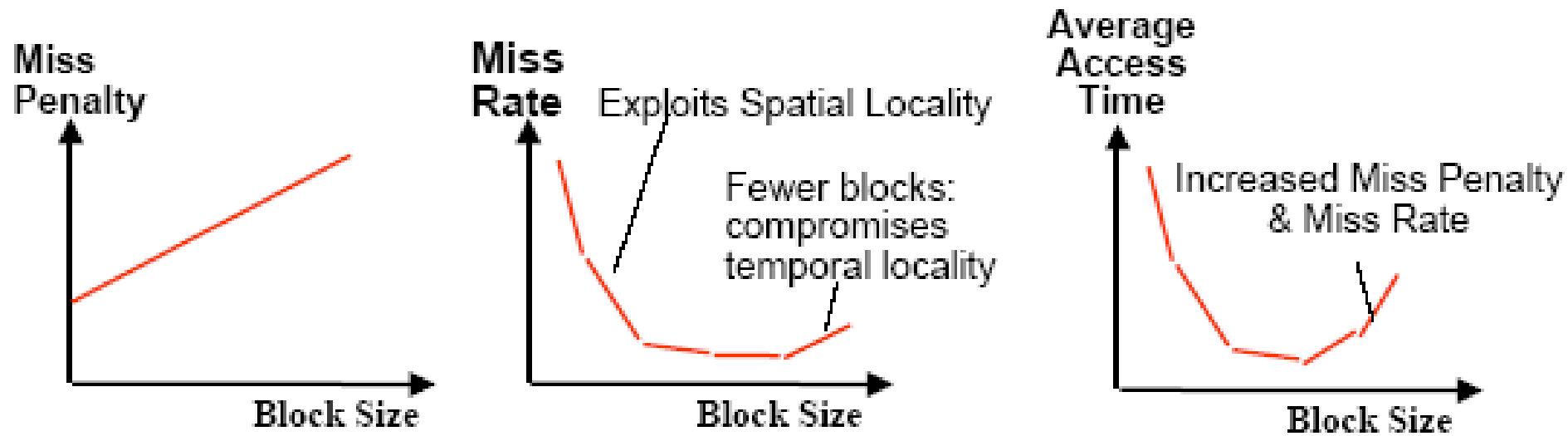
Mapeamento Conjunto Associativo – Vantagens e Desvantagens

- **Vantagens**
 - Aumenta tamanho da cache mantendo tamanho da memória associativa (limitação tecnológica ou custo)
 - Pode usar a totalidade da cache para dados
 - Tags estão nas memórias associativas
 - Bit de validade pode ser substituído por endereço inválido
- **Desvantagens**
 - Memória associativa tem alto custo e tamanho limitado
 - Necessita política de substituição
 - Gasta tempo
 - Pode escolher mal

Mapeamento Conjunto Associativo – Miss Rate versus Tamanho do Bloco



Mapeamento Conjunto Associativo – O Efeito do Tamanho do Bloco



Exercícios:

1. Analise o comportamento das três curvas
2. Porque, a partir de um determinado tamanho, quando o bloco aumenta também aumenta o tempo de acesso?

Exercícios I

1. Cite alguns problemas básicos do uso de memória cache e comente
2. Porque não é necessário para o processador saber onde estão fisicamente os dados quando gera o endereçamento?
3. Quais são as três formas básicas de mapeamento de memórias cache? Comente sobre vantagens e desvantagens das mesmas
4. Explique como funciona o mapeamento direto. Para que serve a tag, e para que serve o bit de validade?
5. Comente sobre o passo 4 (buscar dado no nível inferior) do algoritmo de acesso à memória cache com mapeamento direto. Fale com relação a tempo para sua execução. Detalhe melhor este passo
 1. Calcular o módulo do endereço pelo número de posições da *cache* (ou usar os bits menos significativos do endereço);
 2. Verificar o bit de validade da posição da *cache* correspondente e se for inválido acusar *miss* (ir para 4), senão verificar o Tag;
 3. Se Tag diferente do endereço procurado acusar *miss* (ir para 4), senão ocorre *hit*. Ler a posição (fim);
 4. Buscar dado no nível inferior. Colocar na posição e efetuar a leitura (fim)
6. Porque o mapeamento direto é tão rápido/simples para acessar um dado? Comente, descrevendo os mecanismos necessários para o seu funcionamento

Exercícios II

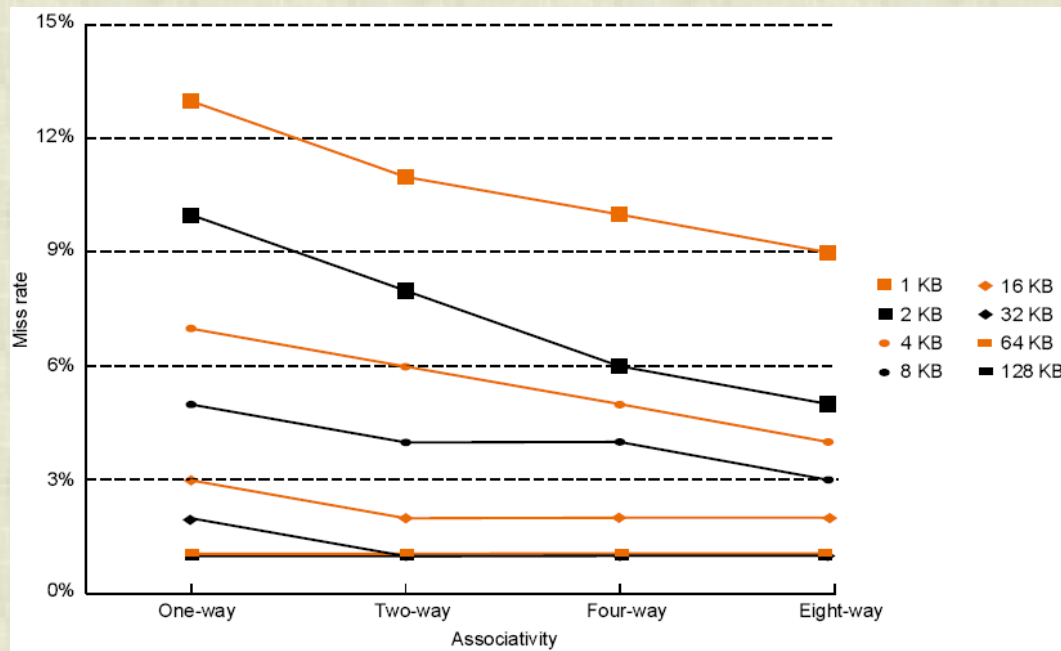
1. Qual a necessidade do multiplexador no mapeamento direto com blocos?
2. Comente a seguinte frase: “Depois de realizado vários testes, verificamos que cache, com mapeamento direto, somente é interessante se os dados que compartilham as mesmas áreas de cache estiverem bastante distantes na memória física”
3. Comente a afirmativa: “Para ter melhor desempenho, a escolha do modo de endereçamento depende da aplicação”. Descreva aplicações onde o mapeamento direto pode ser interessante
4. Existe como a abordagem de mapeamento direto ser realizada dinamicamente? Ou seja, os dados não têm endereços fixos conhecido em tempo de projeto, mas sim durante a execução do programa este endereço é calculado
5. Considerando possível a pergunta acima, quais as conseqüências desta modalidade de mapeamento direto? Comente analisando características como: velocidade de acesso, custos de implementação, flexibilidade da memória cache.
6. Qual a característica básica de uma memória associativa? Responda comentando sobre a organização deste tipo de memória
7. Descreva o funcionamento do mapeamento associativo. Cite uma grande vantagem? Cite uma grande desvantagem?
8. Porque no mapeamento associativo pode não ser necessário o uso de bit de validade?

Exercícios

1. Cite as três políticas básicas para substituição de dados em uma memória associativa. Explique como funciona cada uma delas. Diga vantagens e desvantagens. Sugira alguma outra política e compare com as anteriores
2. Diga qua(l/is) a(s) diferença(s) básica(s) entre o acesso a dados em cache com mapeamento direto e mapeamento associativo
3. Comente a afirmação: "A grande vantagem do mapeamento associativo frente ao mapeamento direto está no fato que toda a memória pode ser utilizada eficientemente como cache. Já no mapeamento direto esta eficiência é praticamente inalcançável"
4. Descreva o funcionamento do mapeamento conjunto associativo
5. Dado S como número de conjuntos de um mapeamento conjunto associativo. Porque se $S = 1$ o mapeamento conjunto associativo se assemelha ao mapeamento associativo, e se $S = \text{número de blocos/palavras da cache}$ o mapeamento conjunto associativo se assemelha ao mapeamento direto.
6. Descreva como funcionam os níveis de associatividade
7. Qual a diferença básica entre o mapeamento associativo e o mapeamento conjunto associativo? Apresente vantagens e desvantagens.

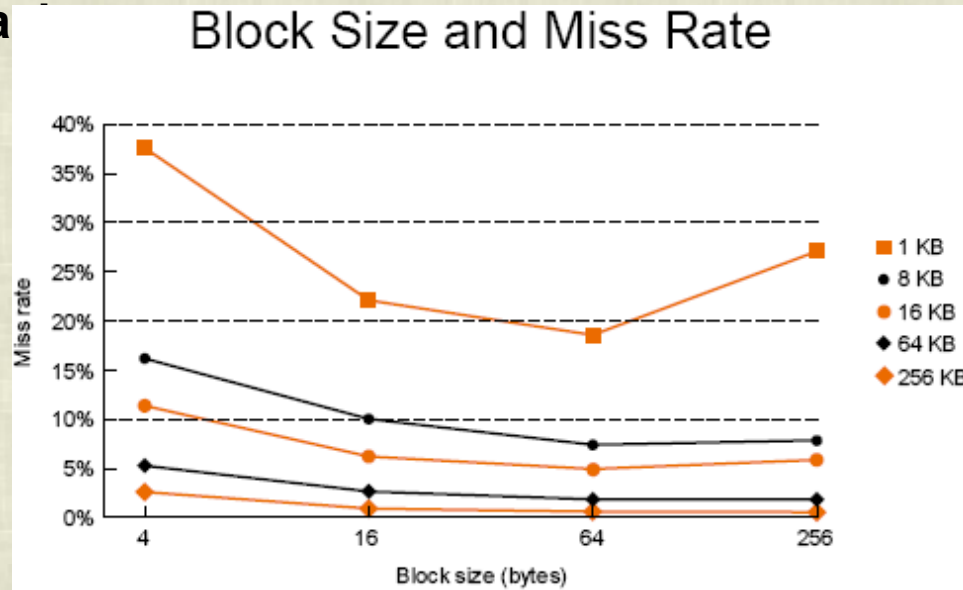
Exercícios

1. Compare com relação a vantagens e desvantagens, o mapeamento direto e o mapeamento conjunto associativo
2. O que diferencia a memória associativa das memórias convencionais? Qual a aplicação das memórias associativas? Faça um diagrama da arquitetura de uma memória associativa, explicando o seu funcionamento
3. Na figura abaixo, que relaciona taxa de associatividade com miss-rate, é mostrado que para, caches pequenas, o miss-rate reduz à medida que aumenta a associatividade (número de vias). Porque isto acontece? Porque o mesmo efeito não é verificado com caches maiores?

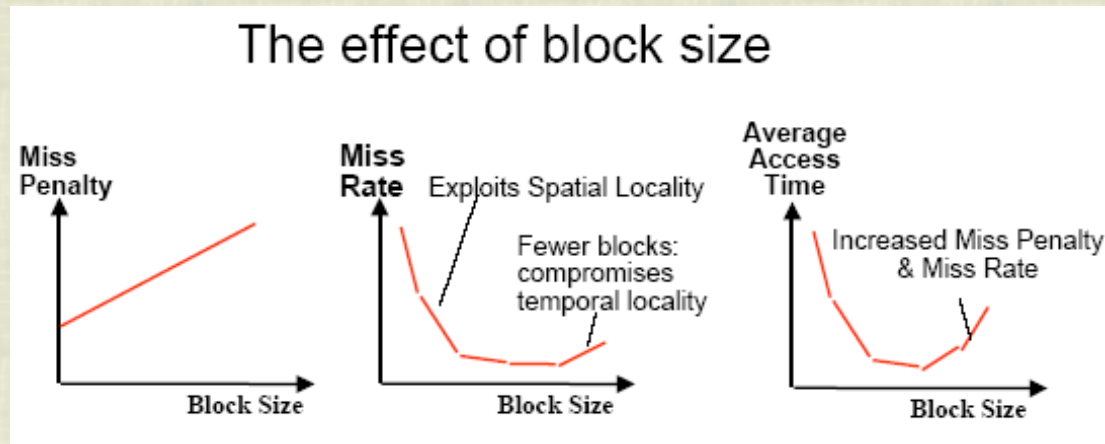


Exercícios

1. Comente as curvas que relacionam tamanho do bloco com miss-rate e tamanho de cache

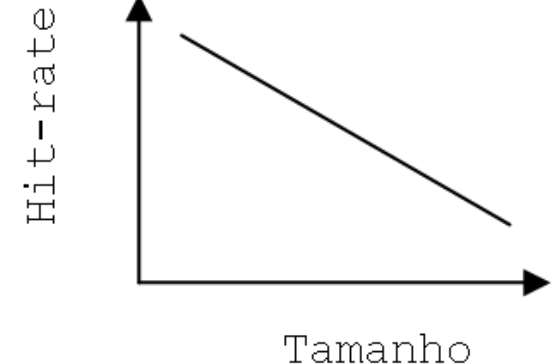
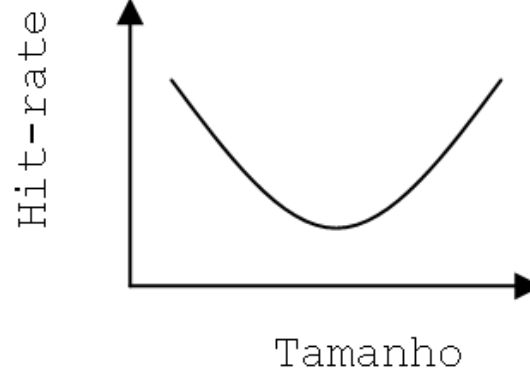
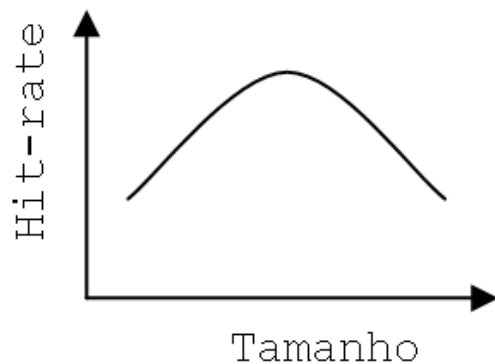
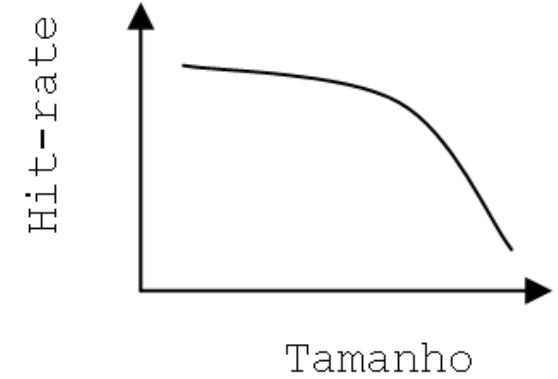
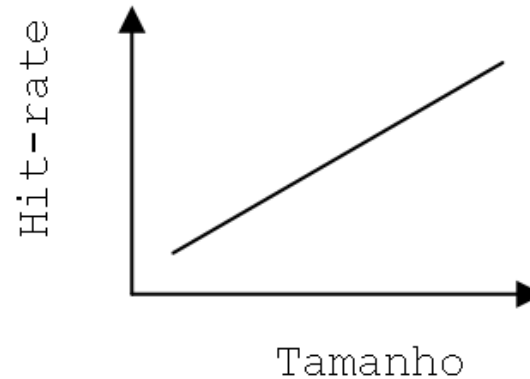
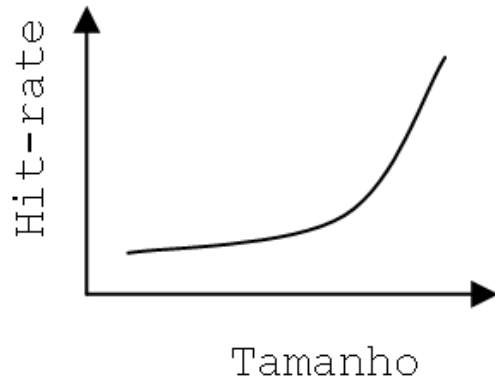


1. Comente as três figuras que relacionam o efeito do tamanho do bloco no tempo médio de acesso e no miss-rate



Exercícios

1. Diferencie localidade espacial de localidade temporal. Explique porque os sistemas de hierarquia de memórias são baseados no princípio de localidade
2. Aponte qual é o gráfico que melhor representa o efeito do tamanho da memória cache no hit rate e diga por quê



Exercícios

1. Considere a seguinte estrutura de memória:

- Memória principal: 1 MByte
- Memória cache: 16 Kbytes
- Tamanho do bloco na cache: 32 palavras
- Tamanho da palavra: 1 Byte
 - Quantos blocos têm a memória cache? Mostre o cálculo e explique
 - Como é formado o endereço para o mapeamento direto? Explique com diagramas
 - Qual a área necessária para armazenar os TAGs, em Bytes?
 - Qual a desvantagem do mapeamento direto?
 - Como seria a formação para o mapeamento conjunto associativo 4-way (4 blocos em cada conjunto)? Explique com diagramas

1. O que pode ser alterado na arquitetura de memórias cache para diminuir o miss-rate? Explique as possibilidades que forem citadas

Exercícios

1. Considere uma memória principal de 256 bytes, e caches de 4 linhas, cada linha contendo um bloco de 8 bytes, além do espaço para colocar os bits de controle (bit de validade e tag), se forem necessários. Dadas as seqüências de acesso à memória principal tabeladas abaixo, preencha as caches para o caso de mapeamento direto e associativo. Considere que no mapeamento associativo, a política para substituição é LRU. Compare ambas as implementações. Diga quantos miss e quantos hits aconteceram em cada caso

Seqüência de acessos (acessão representada em hexadecimal)

Acesso	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Posição	16	17	18	19	16	17	18	19	96	97	98	99	1A	1B	C8	C9	CA	1B	1C	1D

Tabela auxiliar com a numeração da memória principal (256 bytes, blocos de 8 bytes) (linha)

00 - 07	08 - 0F	10 - 17	18 - 1F
20 - 27	28 - 2F	30 - 37	38 - 3F
40 - 47	48 - 4F	50 - 57	58 - 5F
60 - 67	68 - 6F	70 - 77	78 - 7F
80 - 87	88 - 8F	90 - 97	98 - 9F
A0 - A7	A8 - AF	B0 - B7	B8 - BF
C0 - C7	C8 - CF	D0 - D7	D8 - DF
E0 - E7	E8 - EF	F0 - F7	F8 - FF

Exercícios

1. (continuação)

Mapeamento Direto

BV	TAG (3 bits)	0	1	2	3	4	5	6	7

Hits =

Miss =

Mapeamento Associativo

MA (5 bits)	BV	0	1	2	3	4	5	6	7

Hits =

Miss =

Resposta de Exercícios

1. (continuação)

Mapeamento Direto

BV	TAG (3 bits)	0	1	2	3	4	5	6	7
0									
0, 1	110	C8	C9	CA	CB	CC	CD	CE	CF
0, 1	000, 100	10,90	11,91	12,92	13,93	14,94	15,95	16,96	17,97
0, 1	000, 100, 000	18,98,18	19,99,19	1A,9A,1A	1B,9B,1B	1C,9C,1C	1D,9D,1D	1E,9E,1E	1F,9F,1F

Hits = 14, Miss = 6

Mapeamento Associativo

MA (5 bits)	BV	0	1	2	3	4	5	6	7
00010, 11001	0,1	10, C8	11, C9	12, CA	13, CB	14, CC	15, CD	16, CE	17, CF
00011	0,1	18	19	1A	1B	1C	1D	1E	1F
10010	0,1	90	91	92	93	94	95	96	97
10011	0,1	98	99	9A	9B	9C	9D	9E	9F

Hits = 15, Miss = 5

Exercícios

1. (POSCOMP 2005 - 23) Das afirmações a seguir, sobre memória cache, quais são verdadeiras?
- I. Numa estrutura totalmente associativa, um bloco de memória pode ser mapeado em qualquer slot do cache.
 - II. O campo tag do endereço é usado para identificar um bloco válido no cache, junto com o campo de índice.
 - III. Um cache de nível 2 serve para reduzir a penalidade no caso de falta no nível 1.
 - IV. O esquema de substituição LRU é o mais usado para a estrutura de mapeamento direto.
- a. Somente as afirmações (I), (III) e (IV).
 - b. Somente as afirmações (II), (III) e (IV).
 - c. Somente as afirmações (I) e (II).
 - d. Somente as afirmações (I), (II) e (III).
 - e. Somente as afirmações (II) e (III).

Resposta de Exercícios

1. (POSCOMP 2005 - 23) Das afirmações a seguir, sobre memória cache, quais são verdadeiras?
- I. Numa estrutura totalmente associativa, um bloco de memória pode ser mapeado em qualquer slot do cache.
 - II. O campo tag do endereço é usado para identificar um bloco válido no cache, junto com o campo de índice.
 - III. Um cache de nível 2 serve para reduzir a penalidade no caso de falta no nível 1.
 - IV. O esquema de substituição LRU é o mais usado para a estrutura de mapeamento direto.
- a. Somente as afirmações (I), (III) e (IV).
 - b. Somente as afirmações (II), (III) e (IV).
 - c. Somente as afirmações (I) e (II).
 - d. Somente as afirmações (I), (II) e (III).
 - e. Somente as afirmações (II) e (III).

Exercícios

1. (POSCOMP 2008 - 55) Analise as seguintes afirmativas

- I. O processador que apresenta o melhor desempenho é sempre aquele que tem a frequência de relógio mais alta.
- II. A técnica de pipeline é utilizada para aumentar o desempenho em processadores. Dessa forma, o pipeline alivia o tempo de latência das instruções.
- III. A maneira mais simples de aumentar a taxa de acertos em memória cache é aumentar a sua capacidade.
- IV. Em arquiteturas superescalares, os efeitos das dependências e antidependências de dados são reduzidos na etapa de renomeação de registradores.

A análise permite concluir que:

- a) Todas as afirmativas são verdadeiras.
- b) Somente as afirmativas II e III são verdadeiras.
- c) Somente as afirmativas III e IV são verdadeiras.
- d) Somente as afirmativas II, III e IV são verdadeiras.
- e) Nenhuma das afirmativas é verdadeira.

Resposta de Exercícios

1. (POSCOMP 2008 - 55) Analise as seguintes afirmativas

- I. O processador que apresenta o melhor desempenho é sempre aquele que tem a frequência de relógio mais alta.
- II. A técnica de pipeline é utilizada para aumentar o desempenho em processadores. Dessa forma, o pipeline alivia o tempo de latência das instruções.
- III. A maneira mais simples de aumentar a taxa de acertos em memória cache é aumentar a sua capacidade.
- IV. Em arquiteturas superescalares, os efeitos das dependências e antidependências de dados são reduzidos na etapa de renomeação de registradores.

A análise permite concluir que:

- a) Todas as afirmativas são verdadeiras.
- b) Somente as afirmativas II e III são verdadeiras.
- c) Somente as afirmativas III e IV são verdadeiras.
- d) Somente as afirmativas II, III e IV são verdadeiras.
- e) Nenhuma das afirmativas é verdadeira.