Organização e Arquitetura de Computadores II

Gerência de Memórias *Cache*
(Mapeamento de Endereços)

Capítulo 2, 6 do Jean-Loup Baer
Capítulo 5 do Monteiro
Capítulo 4 do Stallings
Capítulo 5 do Hennessy e Patterson
Capítulo 7 do Patterson e Hennessy

Última alteração: 27/11/2017

Prof. Ney Laert Vilar Calazans

Baseado em notas de aulas originais do Prof. Dr. César Marcon

Motivação

- **Funcionalidade**
  - Área de memória rápida e com informações dinâmicas
- ***Caches* possuem parte dos dados de níveis inferiores**
  - Tamanho menor, acesso mais rápido, preço maior
- **Problemas → São 4 principais**
  1. Como identificar se o dado que se quer está na cache?
  2. Se estiver, como acessá-lo da forma mais rápida?
  3. Se não estiver, como buscá-lo (eficientemente) em níveis inferiores?
  4. Que dado tirar da cache para colocar o novo dado (se cache cheia)?
- **Processador não sabe qual nível tem o dado**
  - Apenas gera endereços do mapa de memória principal (MP) → Hw da hierarquia de memória obtém informação

Motivação

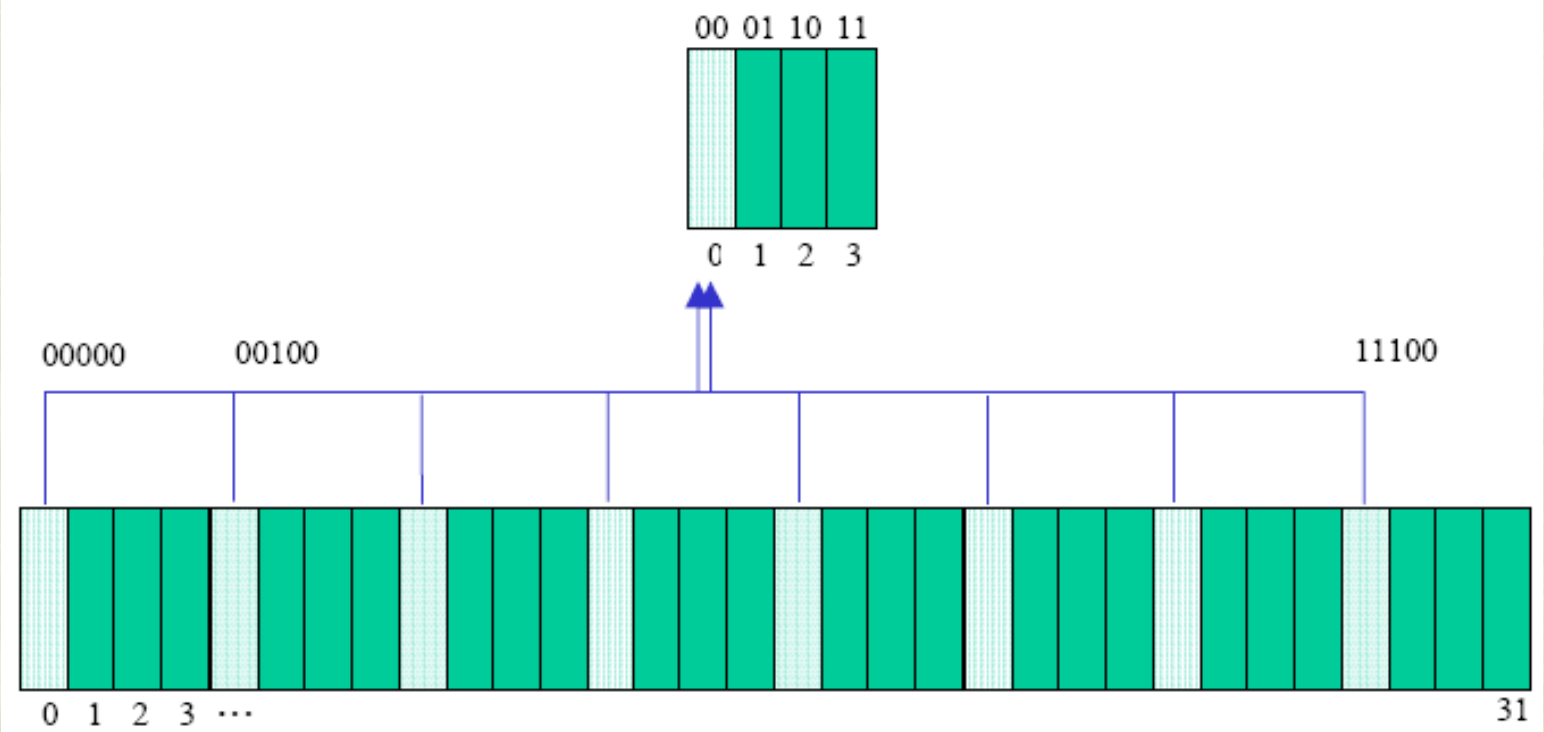
- **Como pesquisar se dado na *cache*?**
  - Cache com todos endereços da MP → não faz sentido
  - Varredura sequencial na *cache* → gasta tempo demais
- **Solução**
  - Algum esquema de mapeamento de endereços
- **Objetivo**
  - Relacionar informações (dados e/ou instruções) da memória principal com posições da *cache*
- **Formas de mapeamento clássicas**
  - Direto
  - Associativo
  - Conjunto Associativo

Índice

Introdução ao Mapeamento Direto

- **Mapeamento mais simples**
  - Posição na *cache* → depende do endereço da palavra na memória principal (MP)
    - Cada palavra → posição fixa
  - Grupo de palavras mapeado na mesma posição da *cache*
- **Exemplo simples**
  - *Cache* de 4 posições, MP de 32 endereços (endereçada a byte)
  - Cada posição da *cache* → 1 de 8 posições da MP
  - Endereço na *cache* → resto da divisão inteira (mod) do endereço na MP pelo tamanho da cache
    - i.e. → Mapeamento usa dois bits menos significativos (em inglês, *least significant bits* ou LS bits) do endereço na MP

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**
- **Poder-se-ia usar os bits mais significativos (em inglês, *most significant bits* ou MS bits) do endereço ao invés dos LS bits? Qual a consequência?**

Bit de Validade	Tag	Dado
1	001	00110110
0		
0		
1	000	11100011

Acesso a Cache com Mapeamento Direto

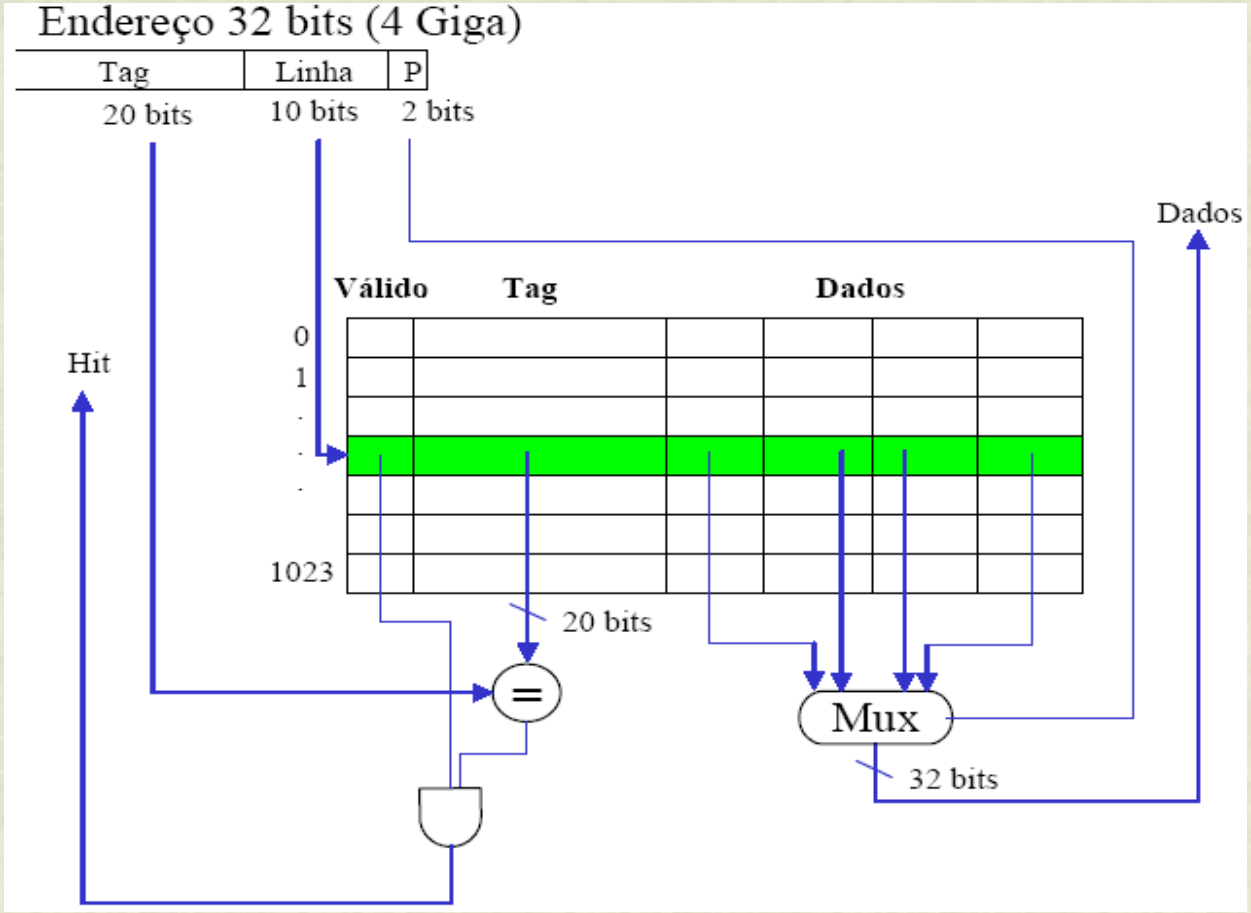
- **Passos para um acesso, dado um endereço X da MP**
  1. Endereço na *cache* = (X **mod** número de linhas da cache) (número de linha da cache = tamanho da cache → Se tamanho da cache é potência de 2, mod corresponde a alguns dos LS bits do endereço)
  2. Se bit de validade da linha da cache for válido, verificar Tag. Se Tag & Endereço da *cache* é endereço **X**, ir para 6.

     Senão, (i.e. se bit de validade inválido ou se [Tag&End Cache] não produzir **X**)

     Acusar **miss**
  4. Buscar dado no nível de memória inferior da hierarquia
  5. Colocar dado no Endereço da *cache*
  6. Efetuar operação (leitura/escrita)
  7. Fim

Mapeamento Direto – Divisão de bits no registrador de endereçamento

- **Exemplo da divisão de blocos em uma cache com 1024 (2^{10}) linhas e blocos com 4 palavras de 32 bits**
  - Bit de validade e Tag
  - Transferência de blocos entre níveis de memória



Qual a vantagem de utilizar blocos ao invés de palavras?

Mapeamento Direto - Exercícios

1. **Considere um espaço de endereçamento da MP de 1 Giga. Como fica a divisão de bits para uma cache de 2048 posições que trabalha com blocos de 8 palavras?**

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

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

Linha da *Cache*		
1 (validade)	16 (tag)	8*32 (bloco de dados)

(Bit de validade + Tag + Dados) / Dados

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

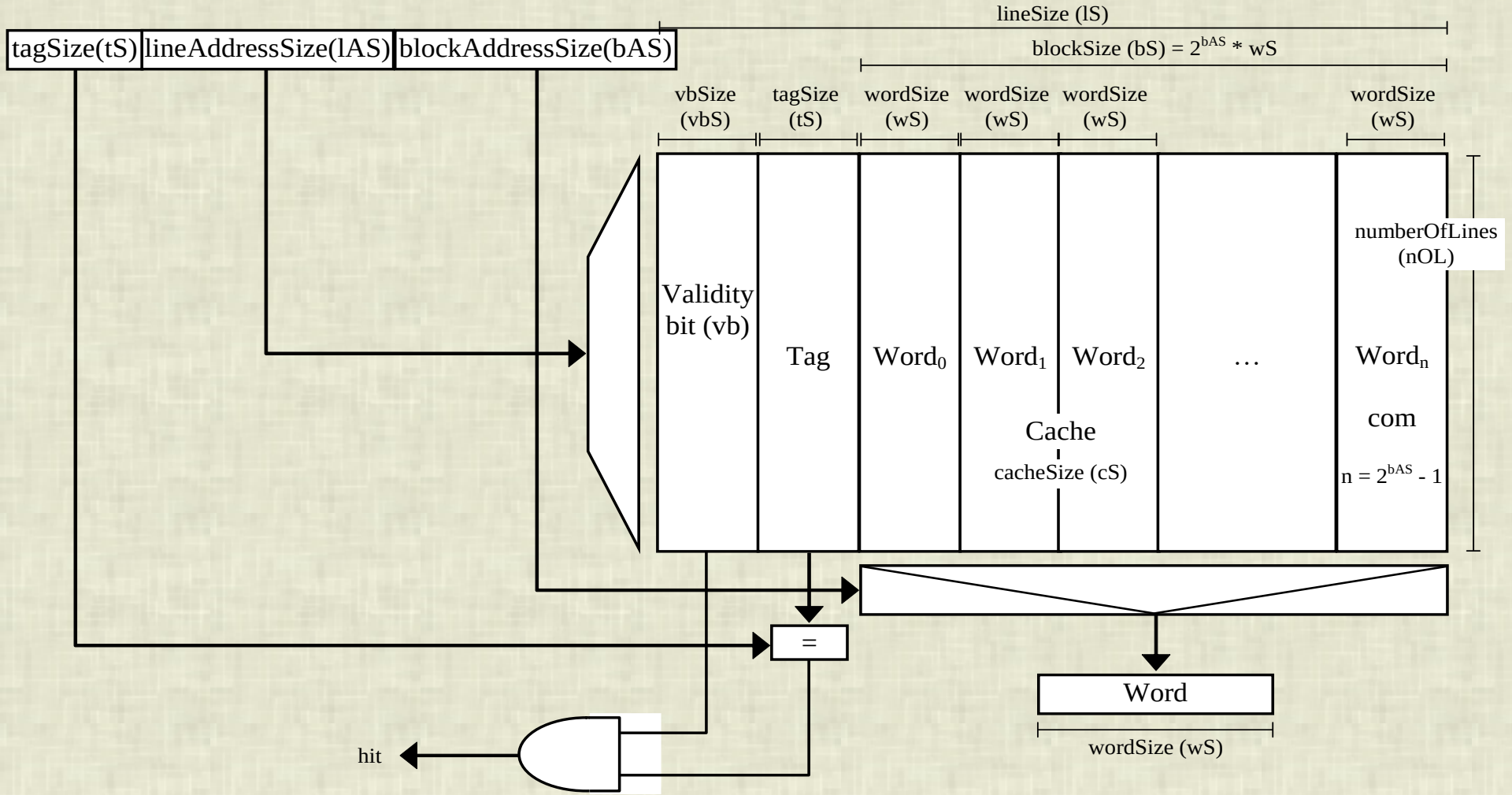
273 bits / 256 bits

256 / 273 * 100% = 93.77%

Mapeamento Direto - Exercícios

3. **Supor as seguintes características: (i) Memória principal de 2 GBytes, com palavras de 32 bits; (ii) Cache com 64 Kbits; (iii) Blocos de 16 palavras**

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



Mapeamento Direto
Obtendo o Formato do Endereçamento

Formato de Endereçamento → obtido via as seguintes relações

- cacheSize ≥ lineSize × numberOfLines
  - lineSize = numberOfWordInsideBlocks × wordSize + tagSize + bitValiditySize
  - numberOfLines = $2^{lineAddressSize}$
  - validityBitSize = 1
  - **cacheSize ≥ (numberOfWordInsideBlocks × wordSize + tagSize + 1)× $2^{lineAddressSize}$**
- addressSize = tagSize + lineAddressSize + blockAddressSize
  - addressSize = $\log_2$(memorySize)
  - blockAddressSize = $\log_2$(numberOfWordInsideBlocks)
  - $\log_2$(memorySize) = tagSize + lineAddressSize + $\log_2$(numberOfWordInsideBlocks)
  - **tagSize = $\log_2$(memorySize) - lineAddressSize - $\log_2$(numberOfWordInsideBlocks)**

Mapeamento Direto
Obtendo o formato do Endereçamento

Voltando ao problema

- Memória principal de 8GBytes, com palavras de 32 bits (word = 4 bytes)
  - **memorySize = 2GWords**
- Cache com 64Kbits
  - **cacheSize = 64KBits = 64 × 1024 = 65536 bits**
- Blocos de 16 palavras
  - **numberOfWordInsideBlocks = 16**
  - **blockAddressSize = $\log_2(16) = 4$**

Aplicando as relações, teremos

- **$cacheSize \geq (numberOfWordInsideBlocks \times wordSize + tagSize + 1) \times 2^{lineAddressSize}$**
  - $65536 \geq (16 \times 32 + tagSize + 1) \times 2^{lineAddressSize}$
  - $65536 \geq (512 + tagSize + 1) \times 2^{lineAddressSize}$
  - **$65536 \geq (513 + tagSize) \times 2^{lineAddressSize}$**

- **$tagSize = \log_2(memorySize) - lineAddressSize - \log_2(numberOfWordInsideBlocks)$**
  - $tagSize = \log_2(2\ Giga) - lineAddressSize - 4$
  - $tagSize = 31 - lineAddressSize - 4$
  - **$tagSize = 27 - lineAddressSize$**

Mapeamento Direto
Obtendo o formato do Endereçamento

Finalizando

Substituindo o termo da equação na inequação, teremos:

- **$65536 \geq (513 + 27 - lineAddressSize) \times 2^{lineAddressSize}$**
- **$65536 \geq (540 - lineAddressSize) \times 2^{lineAddressSize}$**

A solução da inequação pode ser feita por tentativa e erro, para chegar ao valor que mais se aproxima da solução

- **lineAddressSize = 6 $\Rightarrow$**
  - **$65536 \geq (540 - 6) \times 26$**
  - **$65536 \geq 34176$**
- **tagSize = 27 - lineAddressSize $\Rightarrow$**
  - **tagSize = 27 - 6**
  - **tagSize = 21**

Assim, o formato do endereçamento fica

Mapeamento Direto - Conclusões e Questões

- **Vantagens do mapeamento direto**
  - Simplicidade / Velocidade
    - Hardware barato
    - Procura simples (posição fixa)
    - Não existe seleção da vítima (deterministicamente dada pela operação mod)
- **Desvantagens do mapeamento direto**
  - Pode aproveitar mal espaço da *cache* (depende de ordem em que endereços são gerados)
- **Como melhorar o mapeamento apresentado?**
- **Como retirar dependência entre endereço da MP e posição da *cache* sem comprometer desempenho da procura?**

Exercícios

1. **(POSCOMP 2011 - 30)** Um sistema de computador possui um mapa de memória de 4 Gbytes, usando endereçamento a byte e uma memória cache com organização de mapeamento direto. A cache tem capacidade de armazenar até 1.024 palavras de 32 bits provenientes do mapa de memória. Assuma que a cache sempre é escrita de forma atômica com quatro bytes vindos de um endereço de memória alinhado em uma fronteira de palavra de 32 bits, e que ela usa 1 bit de validade por linha de cache. Neste caso, as dimensões do rótulo (*tag*) da cache, do índice e o tamanho da cache são, respectivamente

   a) 12 bits, 18 bits e 54.272 bits

   b) 14 bits, 18 bits e 56.320 bits

   c) 20 bits, 10 bits e 54.272 bits

   d) 20 bits, 12 bits e 54.272 bits

   e) 22 bits, 10 bits e 56.320 bits

Resposta de Exercícios

1. **(POSCOMP 2011 - 30)** Um sistema de computador possui um mapa de memória de 4 Gbytes, usando endereçamento a byte e uma memória cache com organização de mapeamento direto. A cache tem capacidade de armazenar até 1.024 palavras de 32 bits provenientes do mapa de memória. Assuma que a cache sempre é escrita de forma atômica com quatro bytes vindos de um endereço de memória alinhado em uma fronteira de palavra de 32 bits, e que ela usa 1 bit de validade por linha de cache. Neste caso, as dimensões do rótulo (tag) da cache, do índice e o tamanho da cache são, respectivamente

a) 12 bits, 18 bits e 54.272 bits

b) 14 bits, 18 bits e 56.320 bits

c) 20 bits, 10 bits e 54.272 bits

d) 20 bits, 12 bits e 54.272 bits

e) 22 bits, 10 bits e 56.320 bits

Exercícios

Considere uma memória principal de 256 bytes, endereçada a byte, 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 sequências de acesso à memória principal tabeladas abaixo, preencha a cache para o caso de mapeamento direto. Diga, também, quantos misses e quantos hits aconteceram.

Sequência de acessos (posição 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

Hits =

Misses =

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

Resposta de Exercícios

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

(continuação)

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, Misses = 6

Índice

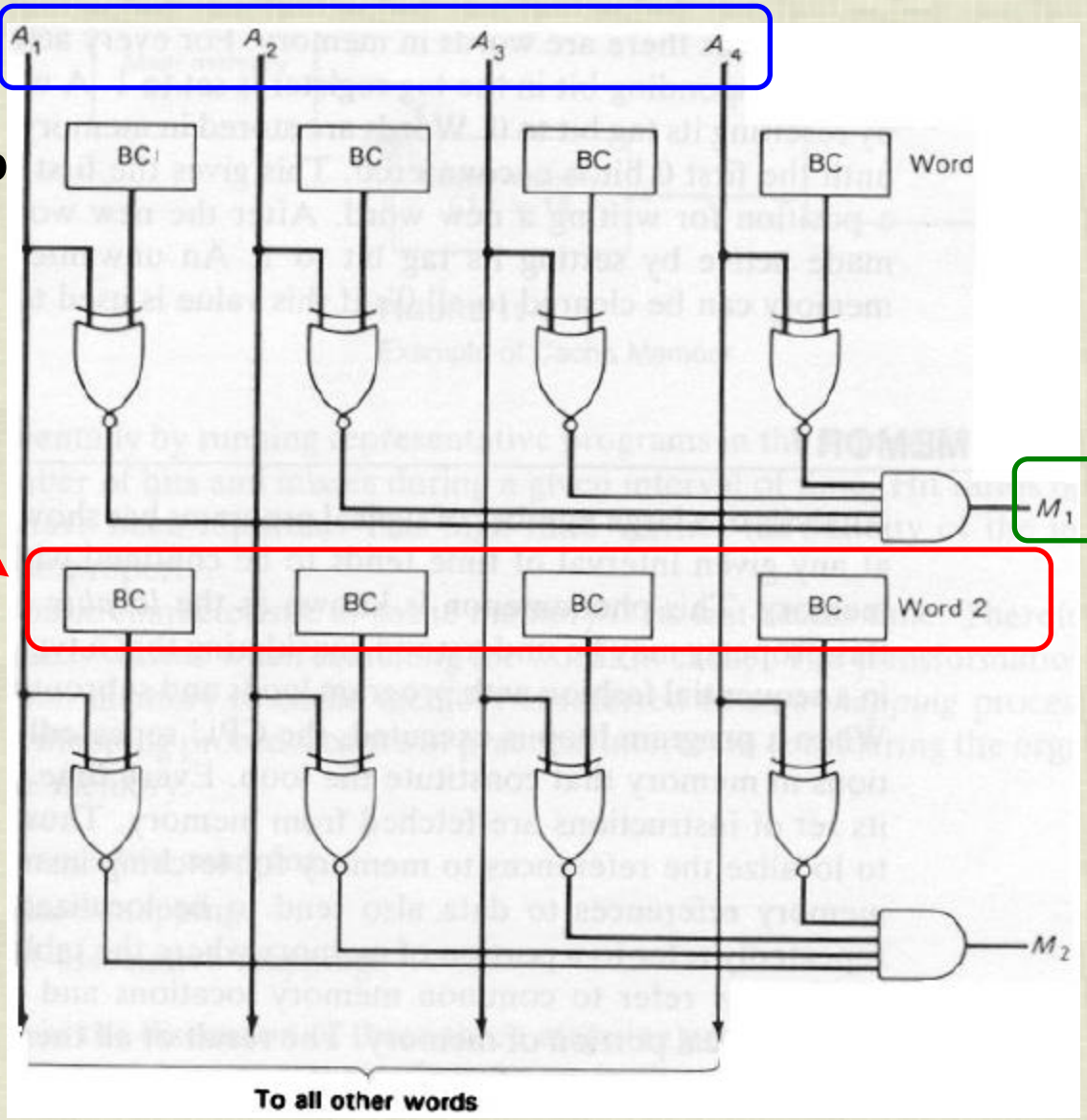
Introdução ao Mapeamento Associativo

- **Ideia → Endereço da MP pode ser carregado em qualquer posição da *cache***
  - *Tag* fora da *cache* → usa-se memória auxiliar especial (associativa)
- **Consequências**
  - Necessita-se de um algoritmo de procura (tem que ser rápido → Hw)
  - Necessita política de substituição (rápida → Hw)
    - Quando ocorre *miss*, deve buscar em nível (níveis) inferior(es); caso *cache* com todas posições ocupadas, como abrir lugar?
- **Solução para procura**
  - Paralelismo
  - Memória associativa → dado valor, retorna posições onde se encontra
    - Memórias associativas → muito cara → → tamanho pequeno

Esquema de Memória Associativa – Pesquisa Paralela

Endereço proveniente do processador

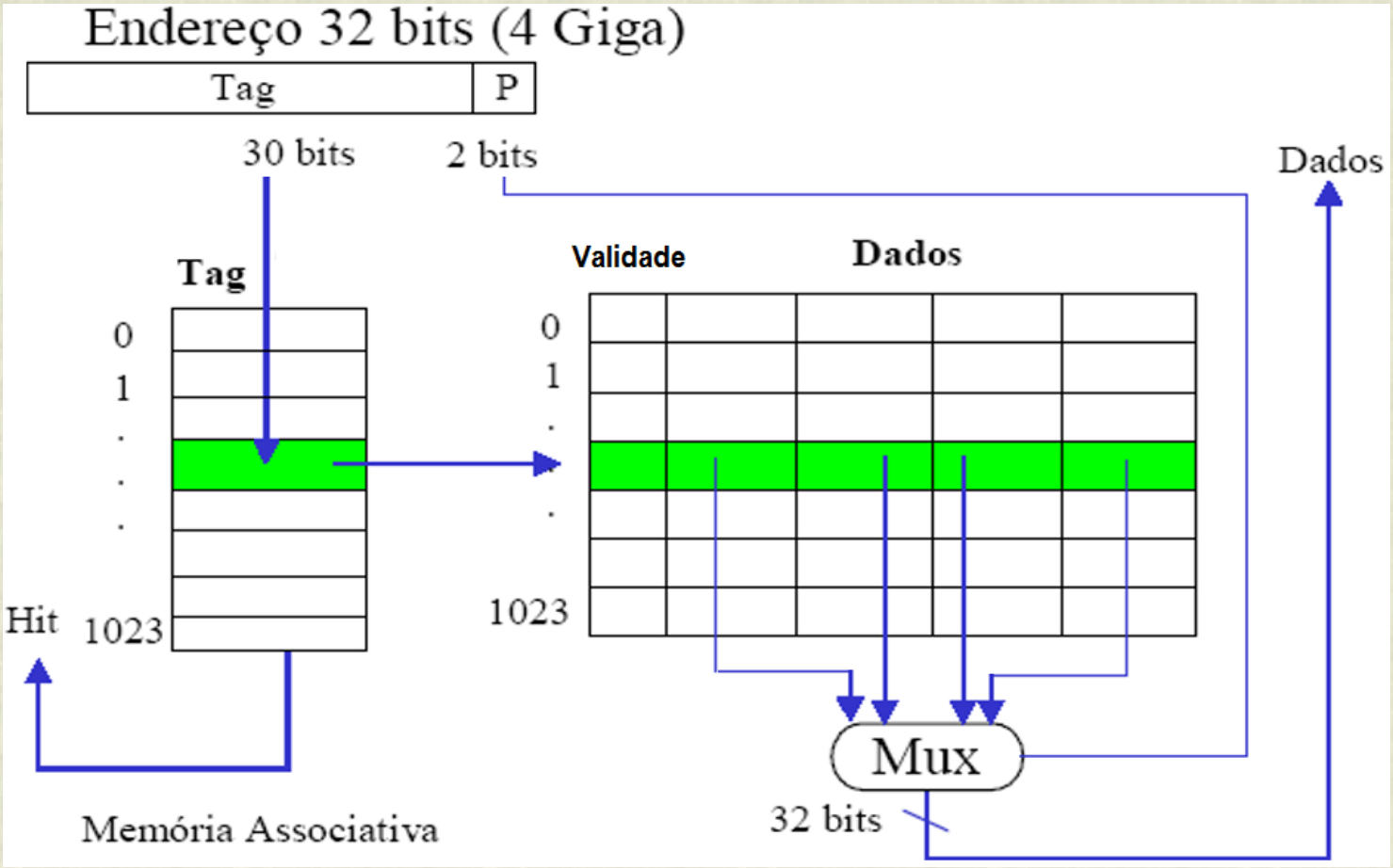
Conteúdo da memória associativa



Se todos os bits dos endereços forem iguais, M_1=1

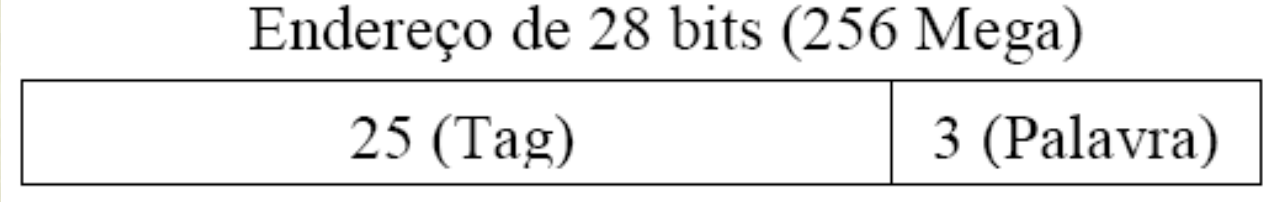
Mapeamento Associativo – Divisão de bits no registrador de endereçamento

- **Exemplo da divisão de blocos em uma cache com 1024 (2^{10}) linhas e bloco com 4 palavras de 32 bits**
  - Bit de validade e Tag
  - Transferência de blocos entre níveis de memória



Mapeamento Associativo - Exercícios

1. Considere um espaço de endereçamento de 256 Megawords (2^{28}). Como ficaria a divisão de bits do endereço para uma cache associativa de 2048 posições e blocos de 8 palavras?



2. Quanto tem efetivamente de dados nessa cache?

   100% (não considerando bit de validade)

3. Qual o tamanho da memória associativa?

   Tamanho = 2048 * 25 (tag)
   = 51200 bits / 8
   = 6400 bytes / 1024 (só para ter resultado em Kbytes)
   = 6,25 Kbytes

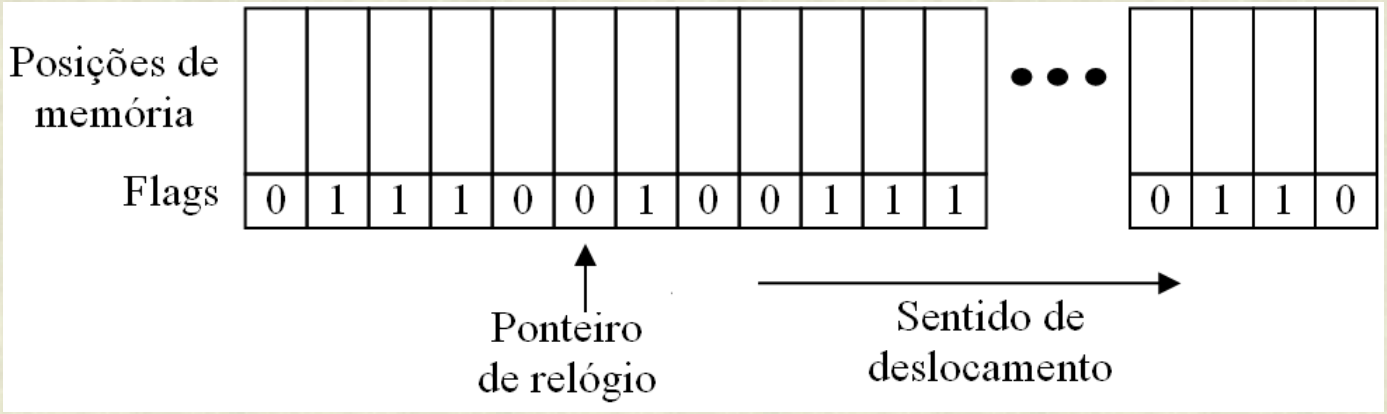
Observação: 1 bit de cache associativa gasta muito mais Hw que 1 bit de memória (RAM) convencional!!

Mapeamento Associativo – Substituição de Dados na Cache

- **Exemplos de Políticas**
  - Aleatória (*Random*)
    - Escolhe aleatoriamente posição a ser substituída → Simples, mas sujeito a aumentar número de *cache misses*
  - Contador
    - Contador baseado em algum tipo de relógio aponta próxima posição a ser substituída → simples, mas como na política aleatória, sujeita a aumentar o número de *misses*
  - LFU (*Least Frequently Used*)
    - Posição da cache menos frequentemente usada é substituída → Escolha precisa de mecanismo de contagem para cada acesso
  - LRU (*Least Recently Used*)
    - Posição da *cache* usada há mais tempo substituída → Escolha precisa controle de tempo a cada acesso e comparação

Mapeamento Associativo – Algoritmo do Relógio

- **Técnica simples → determina endereço de cache visitado há mais tempo. Implementa critério de substituição por tempo**
- **Na verdade, não retorna a posição visitada há mais tempo exatamente, mas é simples, fornecendo alguma das 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, o *flag* é escrito**
- **Um ponteiro (controlado por *relógio*) visita endereços da *cache,* apagando flags à medida em que passa por estas**
- **Se posição de memória não tem *flag* em 1 considera-se que foi acessada há mais tempo e pode ser descartada. Informação fica em lista (pequena) no gerente da hierarquia de memória**



- **Algoritmo pode ser melhorado substituindo flags por contadores com mais de 1 bit. Cada vez que o relógio passa, contador é decrementado, e cada vez que a posição é acessada, contador volta ao seu valor máximo**

Acesso a Cache com Mapeamento Associativo

- **Passos para um acesso**
  1. Pesquisar Tag na memória associativa (comparação)
  2. Se Tag na memória associativa

     (Hit) Acessar memória cache com índice fornecido pela memória associativa

     Ir para 8

     Senão

     Acusar miss
  4. Escolher endereço para substituir, de acordo com política estabelecida (e.g., LRU)
  5. Buscar dado em nível inferior
  6. Colocar na posição livre ou escolhida da memória cache
  7. Cadastrar posição na memória associativa para pesquisas futuras
  8. Efetuar acesso
  9. Fim

Mapeamento Associativo - Conclusões e Questões

- **Vantagem do mapeamento associativo**
  - Melhor distribuição da informação na cache
- **Desvantagens**
  - Memória associativa cara → tamanho limitado
    - Limita número de linhas de *cache*
  - Necessita política de substituição
    - Podem ocorrer escolhas inadequadas
    - Gasta tempo
    - Requer hardware extra para controle de qual posição deve ser substituída

---

- **Limite de tamanho da cache devido pesquisa na memória associativa é restrição muito forte → Tendência é aumentar a cache**
- **Que opções existem?**

Exercícios

(POSCOMP 2013, Questão 48) Sobre memória cache, considere as afirmativas a seguir.

I. No mapeamento associativo, cada bloco da memória principal pode ser carregado em qualquer linha da cache.

II. No mapeamento direto, cada bloco da memória principal é mapeado a apenas uma linha de cache.

III. No mapeamento direto, o acesso repetido a diferentes blocos de memória mapeados na mesma linha de cache resultará em uma alta taxa de acerto.

IV. A técnica de mapeamento associativo é simples e pouco dispendiosa para se implementar.

Assinale a alternativa correta.

a) Somente as afirmativas I e II são corretas.

b) Somente as afirmativas I e IV são corretas.

c) Somente as afirmativas III e IV são corretas.

d) Somente as afirmativas I, II e III são corretas.

e) Somente as afirmativas II, III e IV são corretas.

Resposta de Exercícios

(POSCOMP 2013, Questão 48) Sobre memória cache, considere as afirmativas a seguir.

I. No mapeamento associativo, cada bloco da memória principal pode ser carregado em qualquer linha da cache.

II. No mapeamento direto, cada bloco da memória principal é mapeado a apenas uma linha de cache.

III. No mapeamento direto, o acesso repetido a diferentes blocos de memória mapeados na mesma linha de cache resultará em uma alta taxa de acerto.

IV. A técnica de mapeamento associativo é simples e pouco dispendiosa para se implementar.

Assinale a alternativa correta.

a) Somente as afirmativas I e II são corretas.

b) Somente as afirmativas I e IV são corretas.

c) Somente as afirmativas III e IV são corretas.

d) Somente as afirmativas I, II e III são corretas.

e) Somente as afirmativas II, III e IV são corretas.

Exercícios

Considere uma memória principal de 256 bytes, endereçada a byte, 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 sequências de acesso à memória principal tabeladas abaixo, preencha a caches para o caso de mapeamento associativo. Considere que a política para substituição é LRU. Diga quantos misses e quantos hits aconteceram. Compare a implementação de mapeamento direto com a de mapeamento associativo.

Sequência de acessos (posição 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

MA (5 bits)

BV	0	1	2	3	4	5	6	7

Hits =

Misses =

Resposta de Exercícios

(continuação)

Mapeamento Direto (exercício anterior)

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)
00010, 11001
00011
10010
10011

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

Hits = 15, Miss = 5

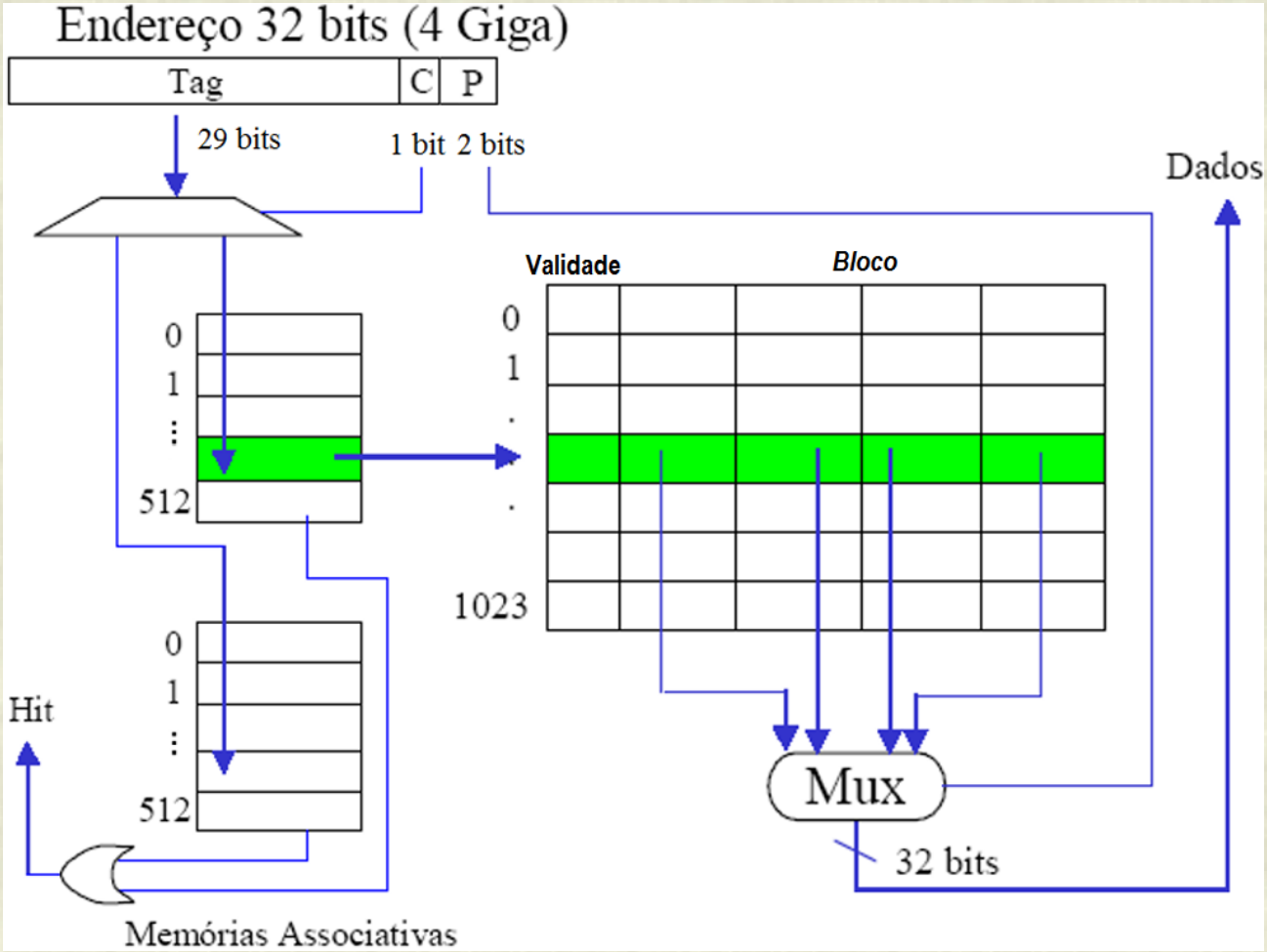
Índice

Introdução ao Mapeamento Conjunto Associativo

- **Compromisso entre mapeamento direto e totalmente associativo**
- **Cache dividida em S conjuntos de N linhas**
  - Se S = 1 → mapeamento associativo
  - Se S = número de linhas 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, que elemento escolher p/ substituir?
  - *A substituição deve ser necessariamente dentro do conjunto*

Mapeamento Conjunto Associativo – Divisão de bits no registrador de endereçamento

- **Exemplo de cache com 1024 linhas (2^{10}), blocos com 4 palavras, palavra de 32 bits e 2 conjuntos (S=2)**



Acesso à Cache 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
  4. Escolher endereço para substituir de acordo com política estabelecida
  5. Buscar dado no nível inferior
  6. Colocar dado na posição livre ou escolhida da *cache*
  7. Cadastrar posição na memória associativa para pesquisas futuras
  8. Efetuar acesso
  9. 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, responda as perguntas que seguem**

1. **Quanto tem efetivamente de dados nessa cache?**

   100% (sem considerar o bit de validade)

2. **Qual tamanho de cada memória associativa?**

   Tamanho = 1024 (linhas) / 2 (cj.) * 31 (tag)

   = 512 * 31 bits = 15872 bits

   = 15872 bits / 8 = 1984 bytes

   = 1984 bytes / 1024 = 1,93 Kbytes

3. **Como ficaria a mesma cache com 4 conjuntos (S=4)?**

Mapeamento Conjunto Associativo - Exercícios

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

   Tamanho = 256 * 30 (tag)
   = 7680 bits / 8
   = 960 bytes / 1024
   = 0,94 Kbytes

5. **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)

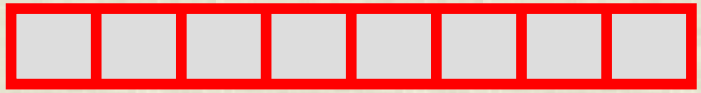
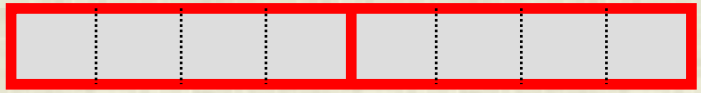
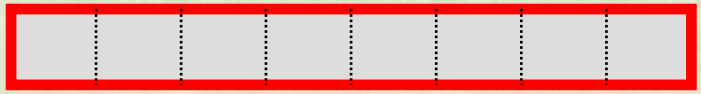
28 (Tag)	2 (Conjunto)	2 (Palavra)

Mapeamento Conjunto Associativo – Vantagens e Desvantagens

- **Vantagem**
  - Aumenta tamanho da cache mantendo tamanho da memória associativa (limitação tecnológica)
- **Desvantagens**
  - Memória associativa tem alto custo e tamanho limitado
  - Somente faz substituição dentro do conjunto
    - Pode aumentar o número de miss, se comparado com o mapeamento associativo
  - Necessita política de substituição
    - Gasta tempo
    - Pode escolher mal
    - Requer hardware extra para controle de qual posição deve ser substituída

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

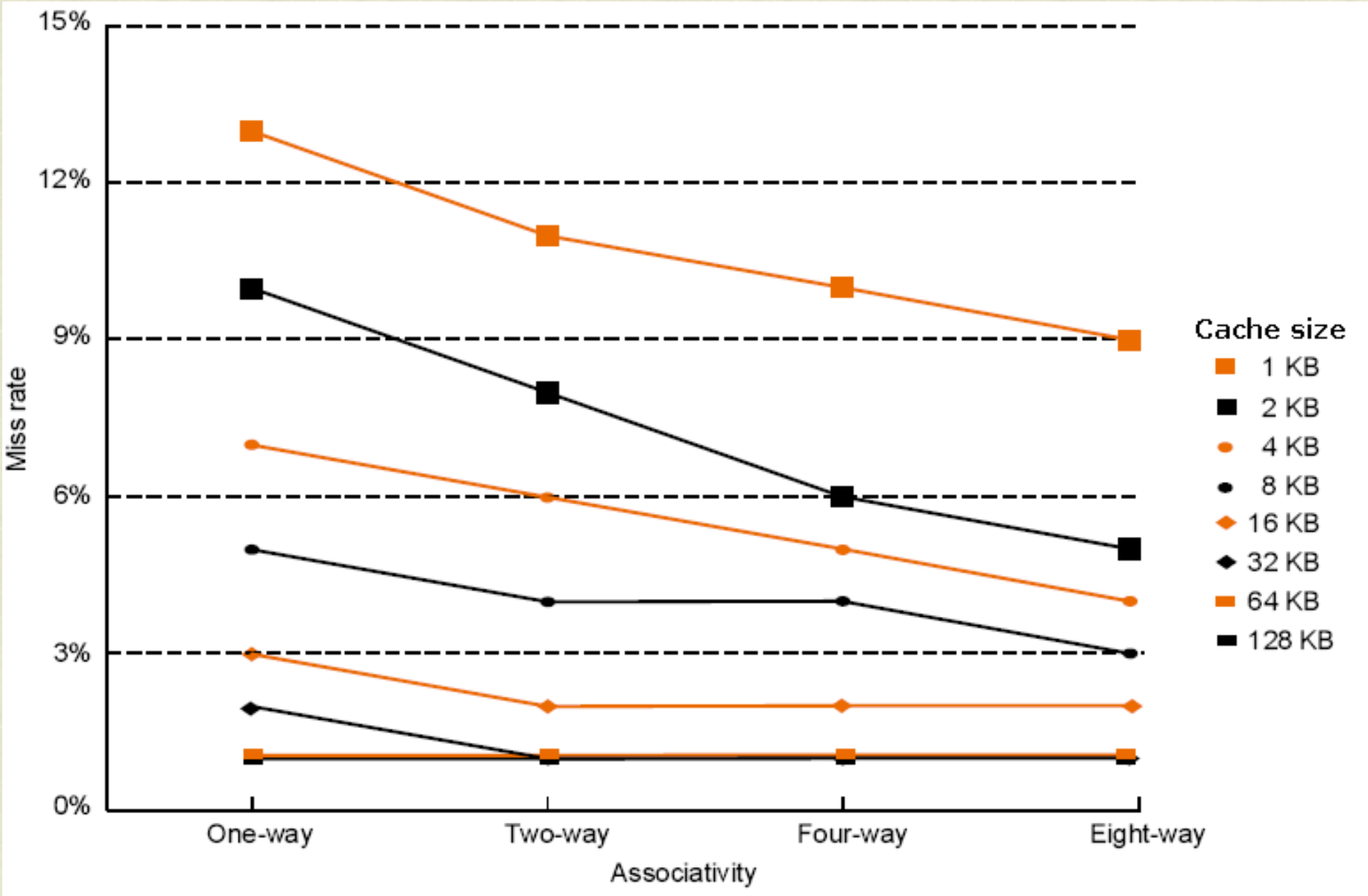
Vias (Ways)	Mapeamento	Esquema
1	Direto	
2	Conjunto-associativo (4 cj)	
4	Conjunto-associativo (2 cj)	
8	Associativo	

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

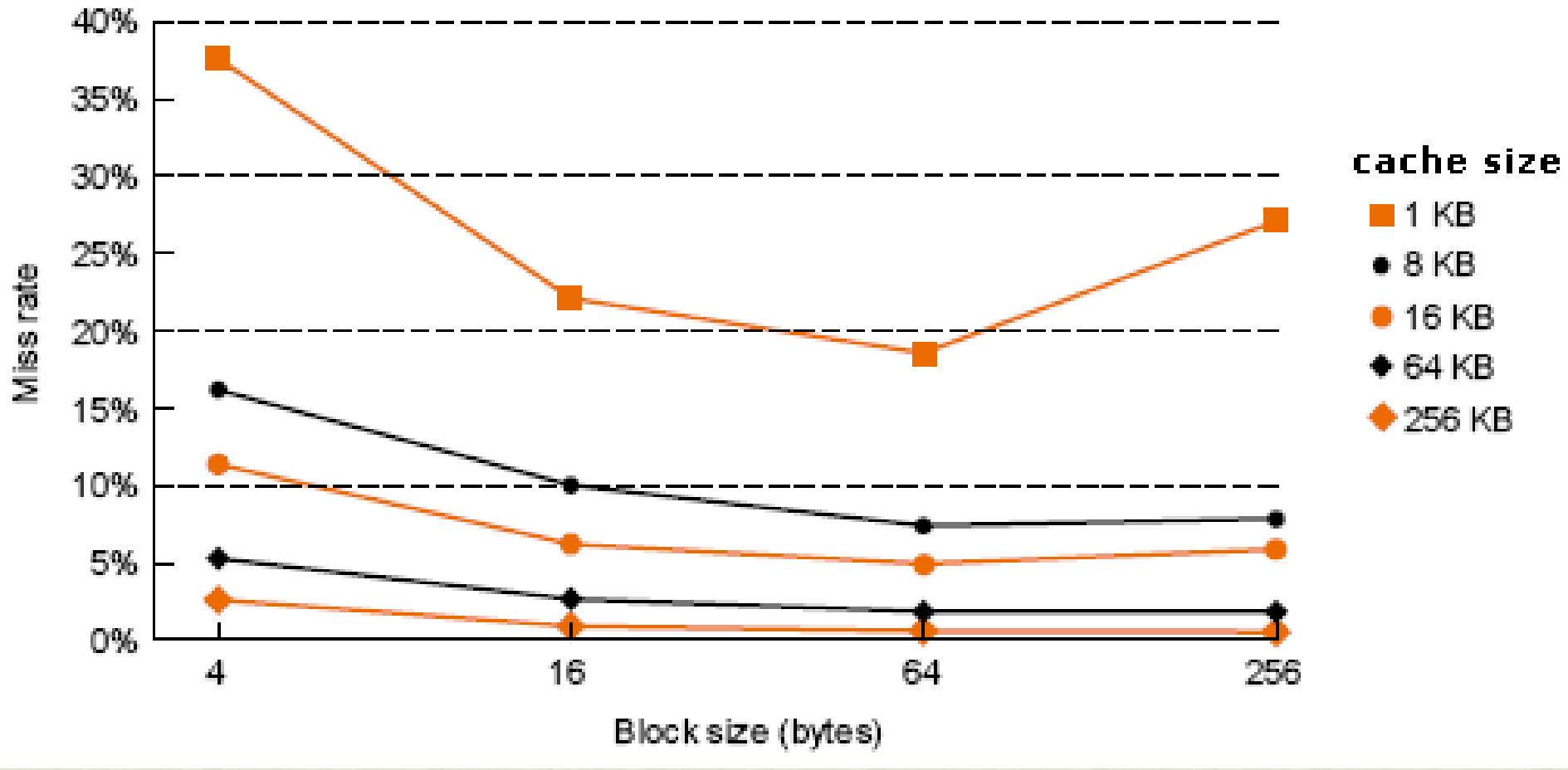
  Tamanho da linha da cache = 32 * 64 b = 2048 b = 256 B = 0,25 KB

  64 KB / 0,25 KB = 256 linhas / 4 (4-Way → 4 linhas por cj) = 64 cj

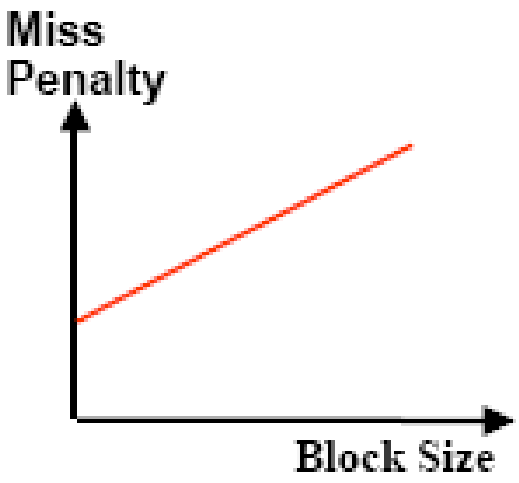
Mapeamento Conjunto Associativo – Níveis de Associatividade e Miss Rate

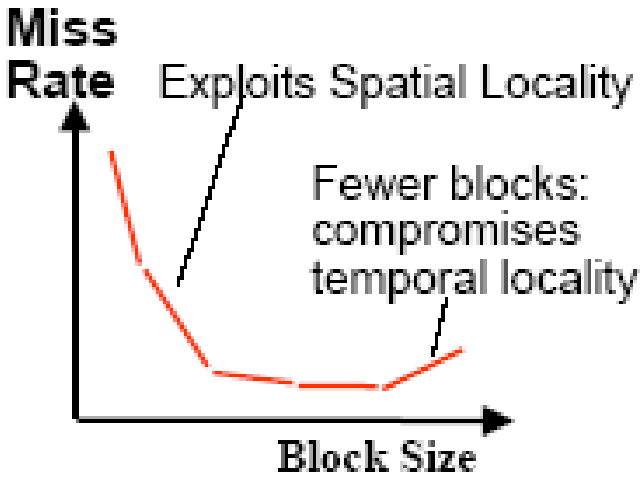


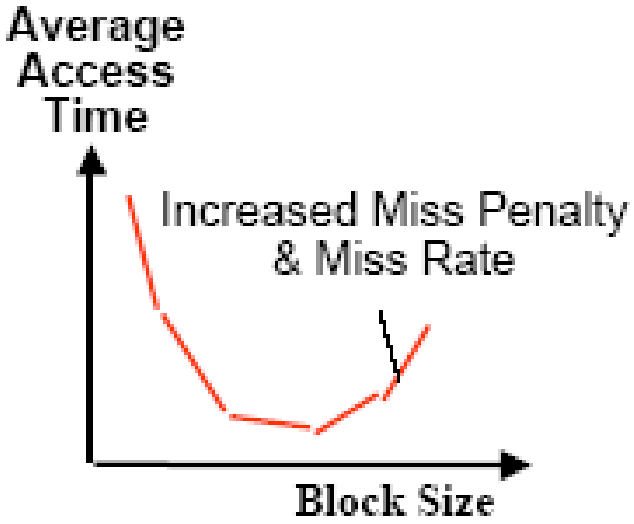
Mapeamento Conjunto Associativo – Miss Rate versus Tamanho do Bloco



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

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

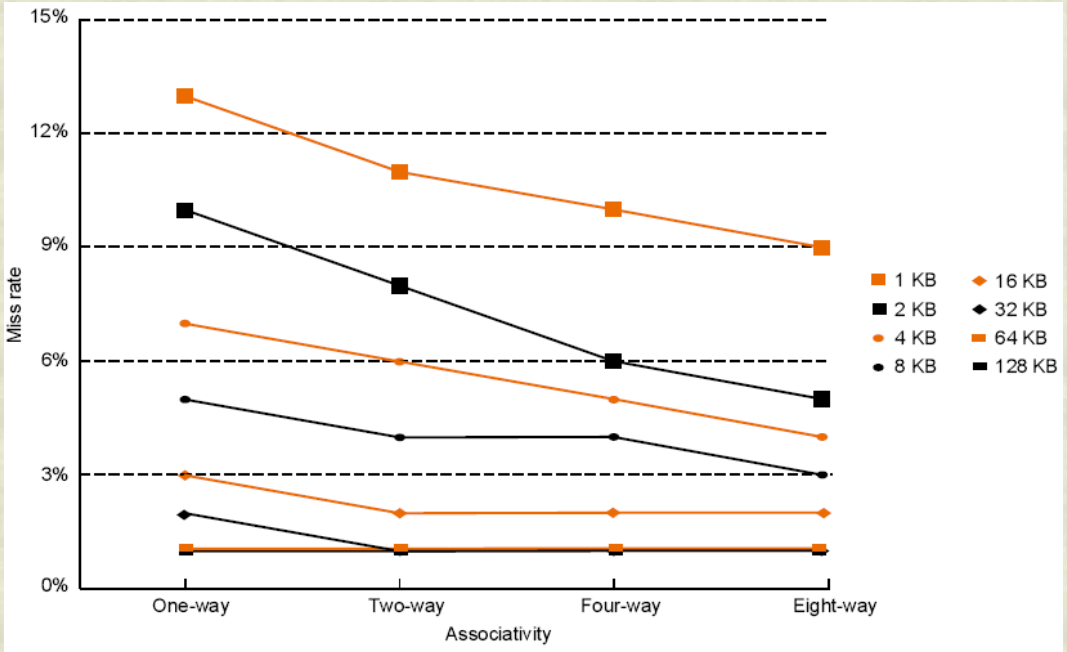
7. Qual a necessidade do multiplexador no mapeamento direto com blocos?
8. Comente a seguinte frase: “Depois de realizado vários testes, verificamos que cache, com mapeamento direto, somente é interessante se os dados que compartilham as mesmas áreas de cache estiverem bastante distantes na memória física”
9. 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
10. 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
11. Considerando possível a pergunta acima, quais as consequências desta modalidade de mapeamento direto? Comente analisando características como: velocidade de acesso, custos de implementação, flexibilidade da memória cache.
12. Qual a característica básica de uma memória associativa? Responda comentando sobre a organização deste tipo de memória
13. Descreva o funcionamento do mapeamento associativo. Cite uma grande vantagem? Cite uma grande desvantagem?
14. Porque no mapeamento associativo pode não ser necessário o uso de bit de validade?

Exercícios

15. 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
16. Diga qua(l/is) a(s) diferença(s) básica(s) entre o acesso a dados em cache com mapeamento direto e mapeamento associativo
17. 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”
18. Descreva o funcionamento do mapeamento conjunto associativo
19. 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 = número de linhas da cache o mapeamento conjunto associativo se assemelha ao mapeamento direto?
20. Descreva como funcionam os níveis de associatividade
21. Qual a diferença básica entre o mapeamento associativo e o mapeamento conjunto associativo? Apresente vantagens e desvantagens.

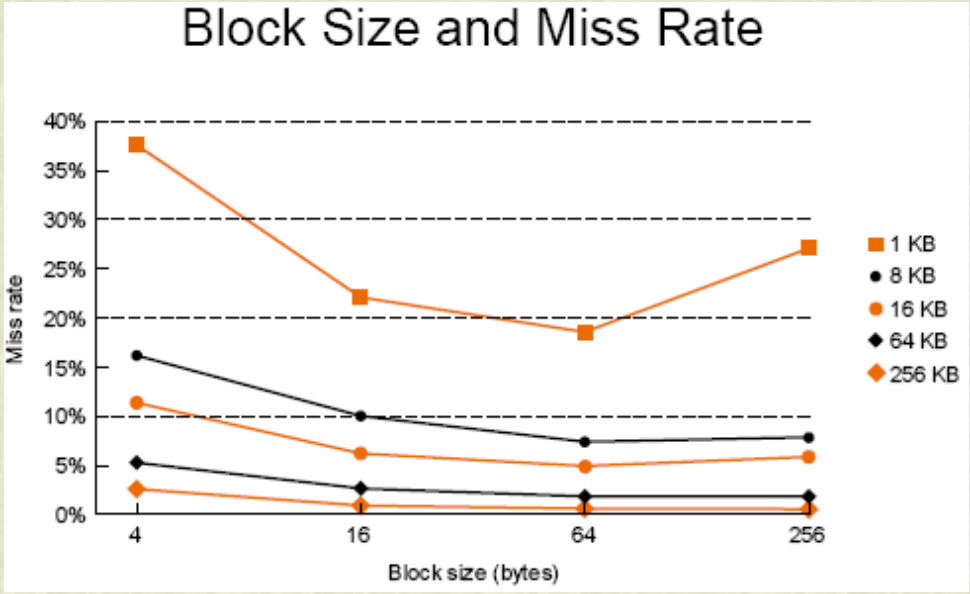
Exercícios

22. Compare com relação a vantagens e desvantagens, o mapeamento direto e o mapeamento conjunto associativo
23. 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
24. 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?

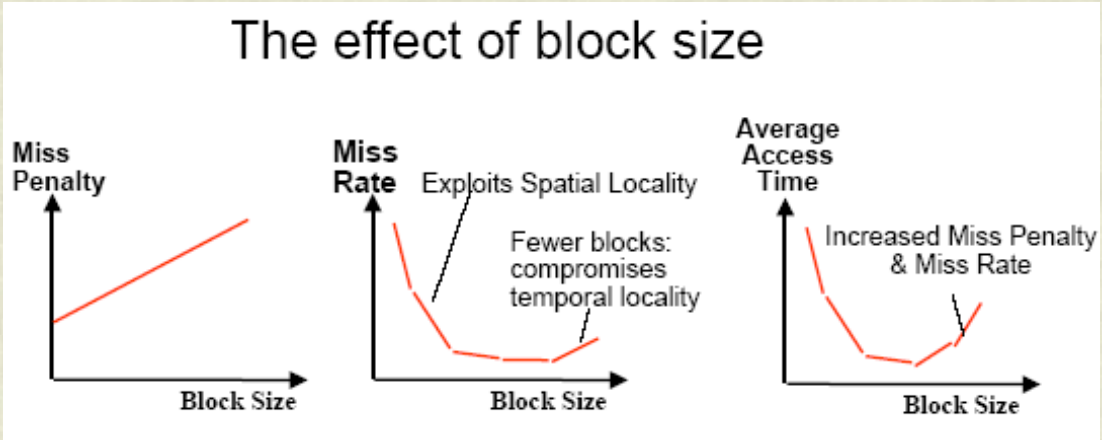


Exercícios

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

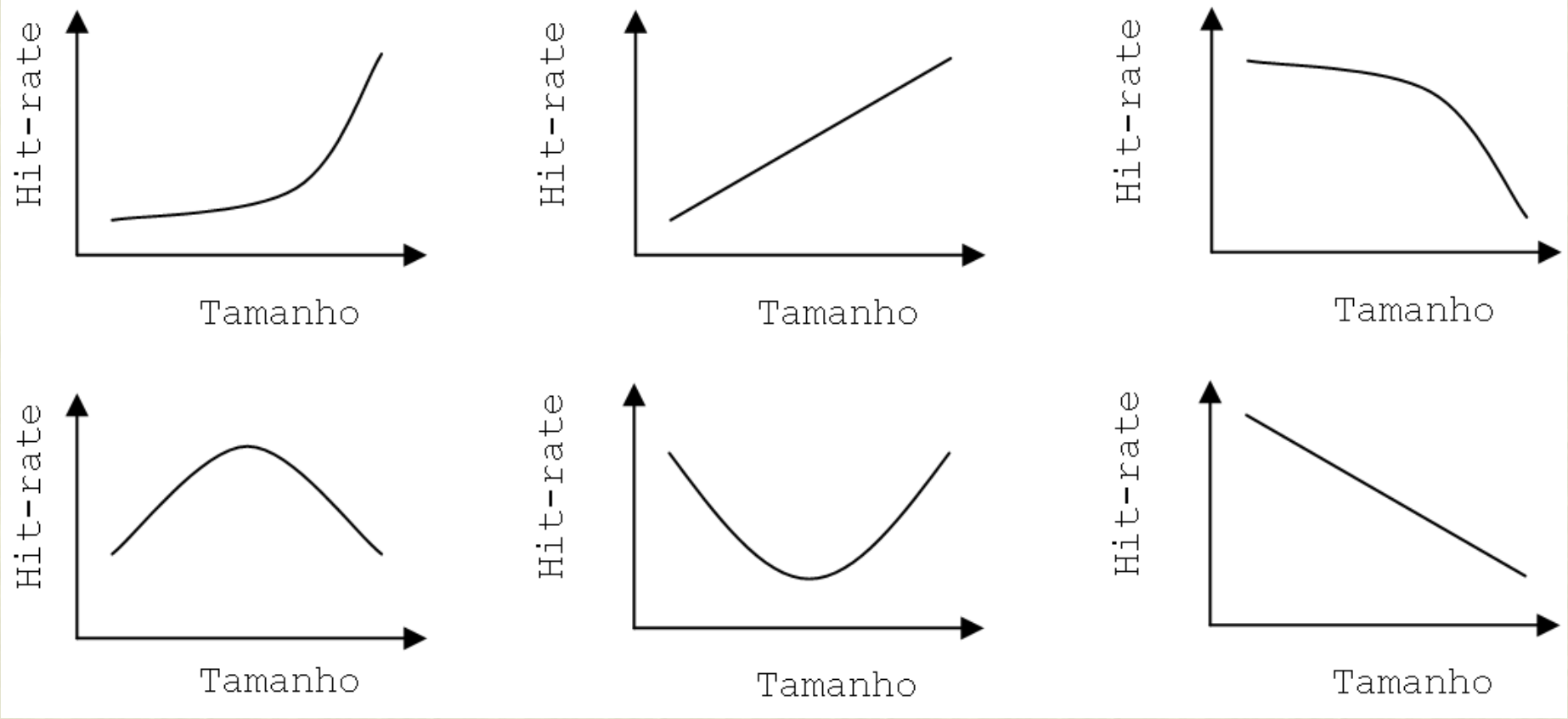


26. Comente as três figuras que relacionam o efeito do tamanho do bloco no tempo médio de acesso e no miss-rate



Exercícios

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



Exercícios

29. 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
30. O que pode ser alterado na arquitetura de memórias cache para diminuir o miss-rate? Explique as possibilidades que forem citadas

Exercícios

31. **(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

31. (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

32. (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

32. (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.

Exercícios

33. (POSCOMP 2010 - 33) Um computador apresenta um sistema de memória organizado em quatro níveis: memórias cache níveis 1 e 2, memórias RAM principal e secundária. Programas prontos para execução são trazidos da memória secundária e transformados em processos na memória principal. Uma instrução para acessar dados na memória fornece o endereço real de memória onde se localiza a informação desejada. A informação é então buscada na cache nível 1. Se lá não for encontrada, ela é buscada no segundo nível de cache. Não sendo encontrada, a informação é finalmente buscada na memória principal.

Qual o modo de endereçamento utilizado?

a) Imediato

b) Indireto

c) Direto

d) Implícito

e) Relativo

Resposta de Exercícios

33. (POSCOMP 2010 - 33) Um computador apresenta um sistema de memória organizado em quatro níveis: memórias cache níveis 1 e 2, memórias RAM principal e secundária. Programas prontos para execução são trazidos da memória secundária e transformados em processos na memória principal. Uma instrução para acessar dados na memória fornece o endereço real de memória onde se localiza a informação desejada. A informação é então buscada na cache nível 1. Se lá não for encontrada, ela é buscada no segundo nível de cache. Não sendo encontrada, a informação é finalmente buscada na memória principal.

Qual o modo de endereçamento utilizado?

a) Imediato

b) Indireto

c) Direto

d) Implícito

e) Relativo

Exercícios

Considere uma memória principal de 256 bytes, endereçada a byte, 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 sequências de acesso à memória principal tabeladas abaixo, preencha a cache para o caso de ***mapeamento conjunto associativo***, com ***2 conjuntos***. Considere a política para substituição é LRU. Diga quantos *misses* e quantos *hits* aconteceram.

Sequência de acessos (posição 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

Hits = , Miss =

MA (4 bits)

BV	0	1	2	3	4	5	6	7