FUNDAMENTOS DE ARQUITETURAS DE COMPUTADORES

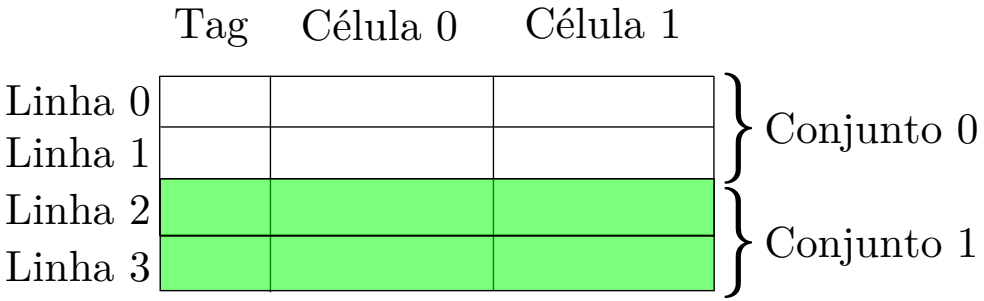
MEMÓRIA CACHE – CONTINUAÇÃO CAPÍTULO 5

Cristina Boeres

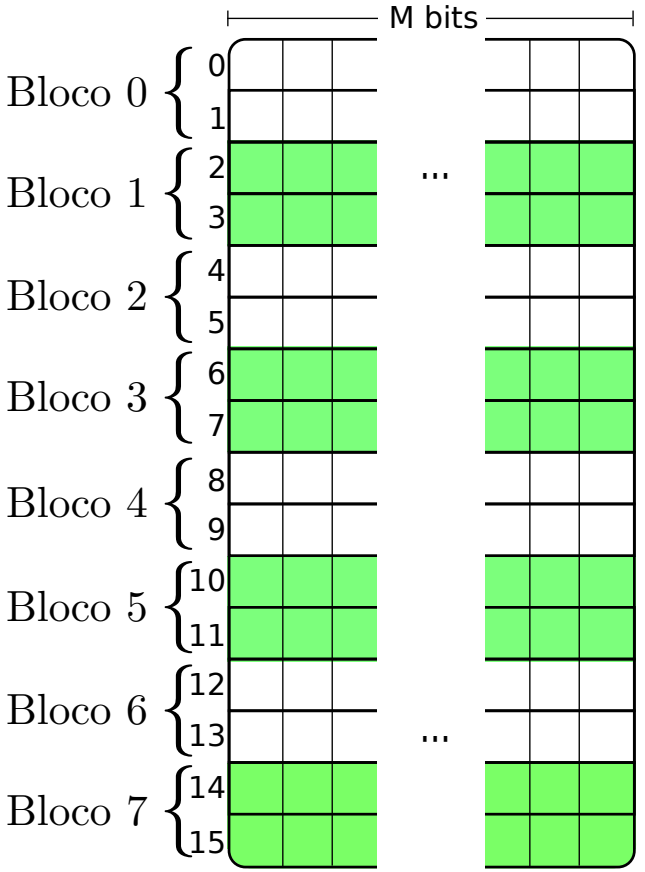
Mapeamento Associativo por Conjunto

- Tenta resolver o problema de
  - conflito de blocos na mesma linha (mapeamento direto)
  - custo da comparação do campo TAG(mapeamento associativo)
- Divide-se o espaço de linhas da cache em conjuntos de N linhas
  - Cada conjunto é tratado pelo sistema como método direto
  - Dentro de cada conjunto, o método é o associativo
- A maioria dos sistemas emprega mapeamento associativo por conjunto, variando o valor de N
  - Conjuntos de 4, 8, 16

Mapeamento Associativo por Conjunto



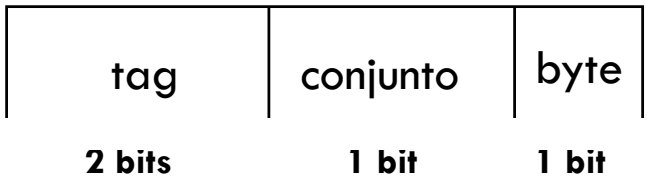
Mapeamento de Endereços	
Conjunto 0	Conjunto 1
0000	0010
0001	0011
0100	0110
0101	0111
1000	1010
1001	1011
1100	1110
1101	1111



Mapeamento Associativo por Conjunto

- MP de 16B com blocos de 2bytes
  - 16B são 16 endereços: cada endereço é especificado por 4 bits
  - Número de blocos = 16bytes/2bytes = 8 blocos
- Memória cache possui
  - 4 linhas de capacidade com 2 bytes por linha
  - 2 conjuntos de 2 linhas cada

  → 1 bit para especificar qual conjunto
- Quantos blocos por conjunto (TAG) ?
  - 8 blocos / 2 conjuntos = 4 blocos por conjunto = 2^2
  - TAG tem 2 bits



Acesso a Cache com Mapeamento Associativo por Conjunto

- Endereço da MP é interpretado pelo sistema de controle da cache

- Valor do campo conjunto é identificado
- Verifica se bloco está na cache
  - Valor do campo tag é “replicado” em todos os elementos de comparação
    - 1 por linha do conjunto
  - As comparações são feitas simultaneamente dentro de um conjunto

Acesso a Cache com Mapeamento Associativo por Conjunto

- Endereço da MP é interpretado pelo sistema de controle da cache

- Se for igual → hit
  - Valor do byte é passado para processador pelo BD
- Senão → miss
  - Sistema inicia a localização do bloco na MP para transferir cópia para a o conjunto específico
    - A linha dentro do conjunto é escolhida de acordo com a política de substituição de linhas

Substituição de Dados na Cache

- Qual dos blocos armazenados deve ser substituído por um novo bloco?
- Decisão necessária quando método de mapeamento é associativo
- Algoritmos:
  - LRU – Least Recently Used
  - FIFO – First-In, First-Out
  - LFU – Least Frequently Used
  - Escolha aleatória

LRU – Least Recently Used

- Escolhe o bloco que não é usado há mais tempo
  - certo custo para guardar essa informação
- Pode se usar um bit indicando que linha foi usada pelo processador
- Em caches associativas por conjuntos de 2 linhas por conjunto
  - simples de implementar
  - quando uma das linhas for acessada, bit é setado (1) e o bit da outra linha do conjunto é zerado
- Quando aumenta a associatividade, problema se torna maior
  - difícil implementar processo de forma exata

Algoritmos de Substituição de Dados na Cache

- FIFO – First-In, First-Out
  - Esquema de fila
    - Primeiro a chegar é o primeiro a sair
  - Escolha independe da frequência de uso do bloco pelo processador
- LFU – Least Frequently Used
  - Bloco que teve menos acessos pelo processador é escolhido
- Escolha aleatória
  - Bloco é escolhido aleatoriamente, independente de seu uso pelo processador

Algoritmos de Substituição de Dados na Cache

- Estudos sobre memórias cache, baseados em simulações, indicam que a escolha aleatória reduz muito pouco o desempenho do sistema em comparação com os demais algoritmos
  - Política aleatória → Simples de implementar
- Quando associatividade aumenta (conjuntos com 4, 8, ou mais linhas)
  - LRU e aleatório quase se equivalem em desempenho
  - Aleatório é mais simples e barato em termos de hardware

Política de Escrita da Cache

- Operações de escrita do processador são feitas em cache
  - Necessário atualizar MP para sistema manter correção e integridade
- Algumas considerações:
  - MP pode ser acessada pela cache ou por dispositivos de E/S (DMA – Direct Memory Access)
    - Cache por ter sido alterada e MP ainda não
    - MP pode ter sido alterada e cache está desatualizada
  - MP pode ser acessada por vários processadores, cada um com sua cache
    - MP pode ser alterada e outras caches estarem desatualizadas

Política de Escrita pela Memória Cache

- *Write through* – escrita em ambas
- *Write back* – escrita somente no retorno
- *Write once* – escrita uma vez

Política de Escrita pela Memória Cache

- *Write through* – escrita em ambas
  - Cada escrita em cache acarreta escrita em MP
  - Caso existam outros processadores, estes também alteram suas caches
  - Mesmo conteúdo sempre nas duas memórias
- *Write back* – escrita somente no retorno
  - Escrita só é atualizada em MP quando bloco for substituído e se foi atualizado
  - Bit adicional é setado (1) se bloco foi atualizado em cache
  - Quando for substituído, se bit correspondente estiver setado, escrita é feita na MP

Política de Escrita pela Memória Cache

- *Write once* – escrita uma vez
  - Apropriada para sistema multiprocessados (cada um com sua cache) compartilhando mesmo barramento
  - Controlador da cache atualiza MP quando o bloco for atualizado pela primeira vez (*write through*) e alerta outros componentes que compartilham o barramento
  - Eles são notificados sobre alteração e impedem o uso da palavra específica
  - Próximas alterações são feitas somente em cache local e o bloco só é atualizado em MP quando for substituído (*write back*)

Comparação entre políticas de escrita

- *Write through*
  - Pode haver grande quantidade de escritas desnecessárias em MP → reduzindo o desempenho
- *Write back*
  - Não há escritas desnecessárias, porém a MP pode ficar desatualizada para uso de outros dispositivos (E/S) → obrigando a acessar o dado através da cache
- *Write once*
  - Conveniente para sistemas multiprocessados
- Estudos indicam que a porcentagem de escrita na cache é pequena (da ordem de 15%) → Política simples de write through

Níveis de Cache

- Nível 1 (Level 1) ou L1
  - Sempre localizada no interior do processador
  - Cache primária
    - Cache L1 de instruções e cache L1 de dados
- Nível 2 (Level 2) ou L2
  - normalmente localizada no exterior do processador (placa mãe)
    - Cache secundária
  - Alguns processadores têm L2 interna a pastilha do processador
- Nível 3 (Level 3) ou L3
  - Quando processador possui L1 e L2 interna, é a cache externa ao processador (placa mãe)

Tamanho da Memória Cache

- Definição do tamanho adequado depende de vários fatores:
  - Tamanho da memória principal
  - Relação acertos/faltas
  - Tempo de acesso da MP e das caches L1 e L2
  - Custo médio por bit da MP e das caches L1 e L2
  - Natureza do programa em execução
- Exemplo: Intel Core
  - L1: 32KB para dados e 32KB para instruções
  - L2: 256KB por núcleo
  - L3: 8MB compartilhada

Tamanho da Memória Cache

- Largura de linha ótima é uma decisão difícil
- O tamanho da linha está naturalmente associado ao princípio da localidade espacial
  - Como se sabe que acessando um determinado endereço, os acessos seguintes tem grande probabilidade de serem endereços contíguos
  - Uma linha com muitos bytes atende a este princípio
- Porém aumentar a linha não indica que o número de cache hits aumenta!
  - Existem desvios nos programas (if-then-else)
  - Linhas grandes diminuem o número de linhas
- É comum encontrar caches com linhas entre 8 e 32 bytes

Tamanho da Memória Cache

- Suponha um computador com 4 GB de memória RAM
  - Memória é endereçada em células de 1 byte
  - Cache possui linhas com 32 células cada (além do campo tag)
  - No total, a cache possui 1024 linhas
- Determine o formato dos endereços de memória nos seguintes tipos de mapeamento:
  - Mapeamento direto
  - Mapeamento associativo
  - Mapeamento associativo por conjunto com 16 conjuntos (com o mesmo número de linhas cada)