Memória Cache

Marcos Monteiro Junior

Tratando falhas de cache

- A unidade de controle precisa encontrar a falha de cache e processa-la buscando os dados requisitados na memória.

- Se ocorrer um acerto nada deve ser feito

- Como o processador deve trabalhar em caso de falha?

Tratando falhas de cache

- O tratamento de falha é feito pela unidade de controle do processador e um controlador separado que inicia o acesso à memória e preenche novamente a cache.
- O processamento de uma falha causa um *Stall*, assim como no *pipeline.*
- Congela o código até a requisição seja completa.
- Alguns processadores mais modernos conseguem continuar o processamento de outros campos do código.

Tratando falhas de cache

- Falhas de instrução e de dados são tratados.
  - Se um acesso à instrução resultar em falha o conteúdo do registrador de instrução será invalido
  - Logo o contador de programa regride para que seja estabelecida a memória cache.

Tratando falhas de cache

- Segue as etapas
  1. Enviar o valor de PC (atual -4) para a memória
  2. Instruir a memória principal a realizar uma leitura e esperar que a memória complete seu acesso
  3. Escrever na entrada da cache, colocando os dados na memória na parte dos dados da entrada, escrevendo os bits mais significativos do endereços (vindo da ALU) no campo tag e ligando o bit de validade.
  4. Reiniciar a execução da instrução na primeira etapa, o que buscara novamente a instrução, desta vez encontrando-a na cache.

Tratando escritas

- Escritas funcionam de maneira diferente
- Suponha que seja realizada um *store* o valor sera colocado na cache
- Logo existirá um valor na cache que não está na memória principal. Dizemos que a cache e a principal estão ***inconsistentes***.

Tratando escritas

- Manter os dados consistentes consiste em escrever os dados na cache e na memória principal, isso é chamado de ***write-through.***
- Essa é uma solução simples, porém:
  - Os dados recém buscados para a cache podem ser reescritos pelo usuário
  - É um processo muito lento já que um processo de escrita pode levar até 100 ciclos de clock

Tratando escritas

- Uma solução é:
  - Usar buffer de escrita, esse armazena os dados até serem escritos na memória
  - Se ocorra um número de escrita maior que a capacidade do buffer o processador deverá sofrer um *stall.*
- Outra alternativa é o esquema chamado *write-back*
  - Quando ocorre uma escrita o novo valor é escrito somente no bloco da cache
  - O bloco só é escrito no nível inferior quando o bloco da cache for substituído.

Exemplo

- Intrinsity FastMath, é um processador de arquitetura MIPS
- Possui um exemplo didático
- Oferece tanto write-through quanto write-back

Exemplo

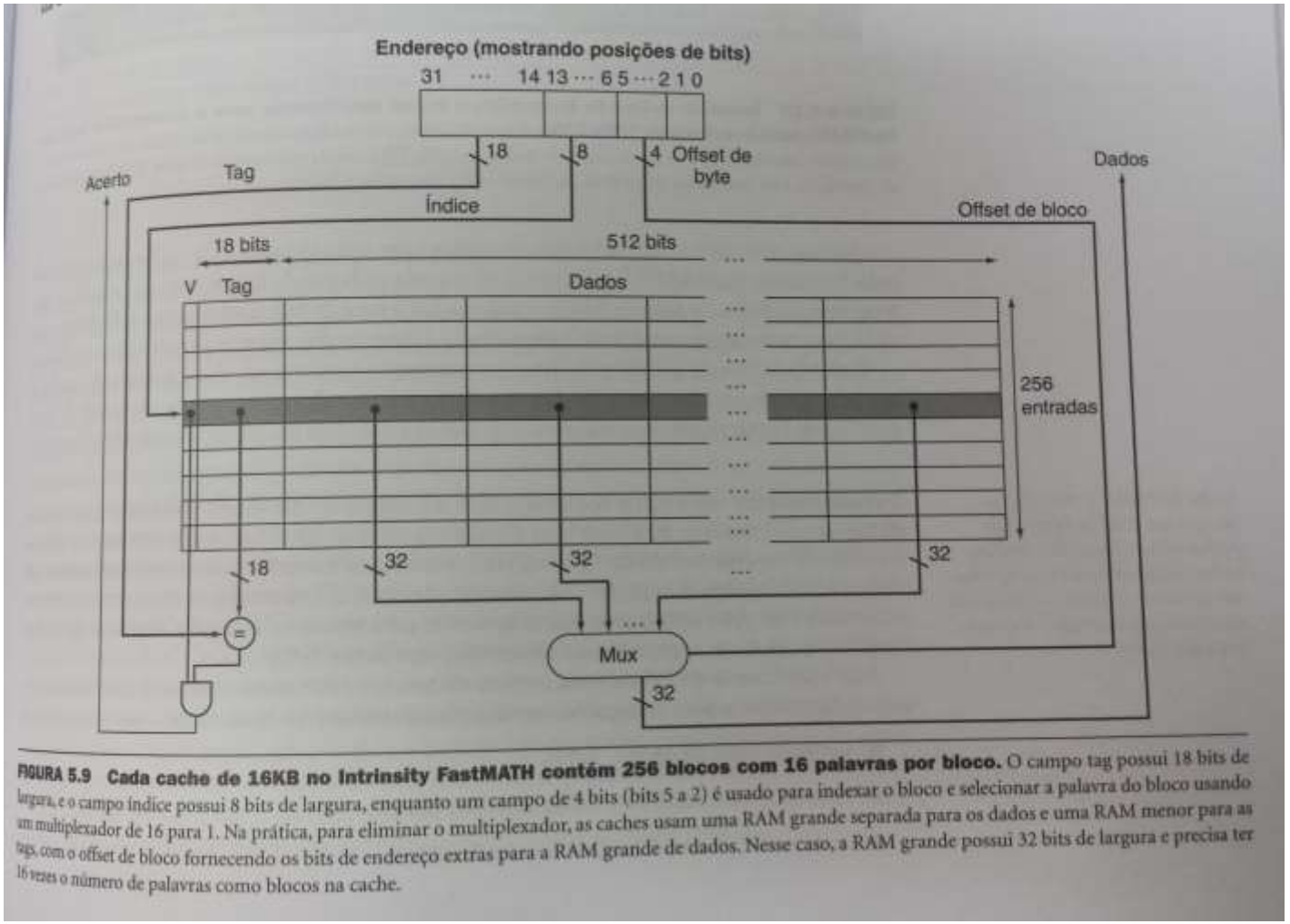

FIGURA 5.9 Cada cache de 16KB no Intrinsity FastMATH contém 256 blocos com 16 palavras por bloco. O campo tag possui 18 bits de largura, e o campo índice possui 8 bits de largura, enquanto um campo de 4 bits (bits 5 a 2) é usado para indexar o bloco e selecionar a palavra do bloco usando um multiplexador de 16 para 1. Na prática, para eliminar o multiplexador, as caches usam uma RAM grande separada para os dados e uma RAM menor para as tags, com o offset de bloco fornecendo os bits de endereço extras para a RAM grande de dados. Nesse caso, a RAM grande possui 32 bits de largura e precisa ter 16 vezes o número de palavras como blocos na cache.

Exemplo

- Esse processador possui pipeline de 12 estágios
- Possui cache de instrução e cache de dados separadas.
- Etapas
  1. Enviar o endereço à cache apropriada. O end. do PC ou da ALU.
  2. Se a cache sinalizar acerto, a palavra requisitada estará disponível nas linhas de dados. Como existem 16 palavras no bloco desejado, precisamos selecionar a palavra correta. Um campo índice de bloco é usado para controlar o multiplexador, que seleciona a palavra requisitada das 16 palavras do bloco.
  3. Se a cache sinalizar falha, enviaremos o endereço para a memória principal. Quando a memória retornar com os dados, nós os escrevemos na cache e, então os lemos para atender à requisição

Projetando sistemas de memória para suportar caches

- As falhas de cache são satisfeitas pela memória principal, que construída com DRAMs.

- O processador normalmente é ligado ao memória por meio de barramentos. A velocidade de clock do barramento geralmente é muito mais lenta do que a do processador. A velocidade desse barramento afeta a penalidade de falha.

Exemplo hipotético

- 1 ciclo de clock de barramento de memória para enviar o endereço
- 15ciclos de clock de barramento de memória para cada acesso a DRAM
- 1 ciclo de clock de barramento de memória para enviar uma palavra de dados
- Se tivermos um bloco de cache de quatro palavras e um bloco de DRAM com largura de uma palavra, a penalidade de falha seria 1+4X15+4X1=65ciclos de clock

Projetos de memórias

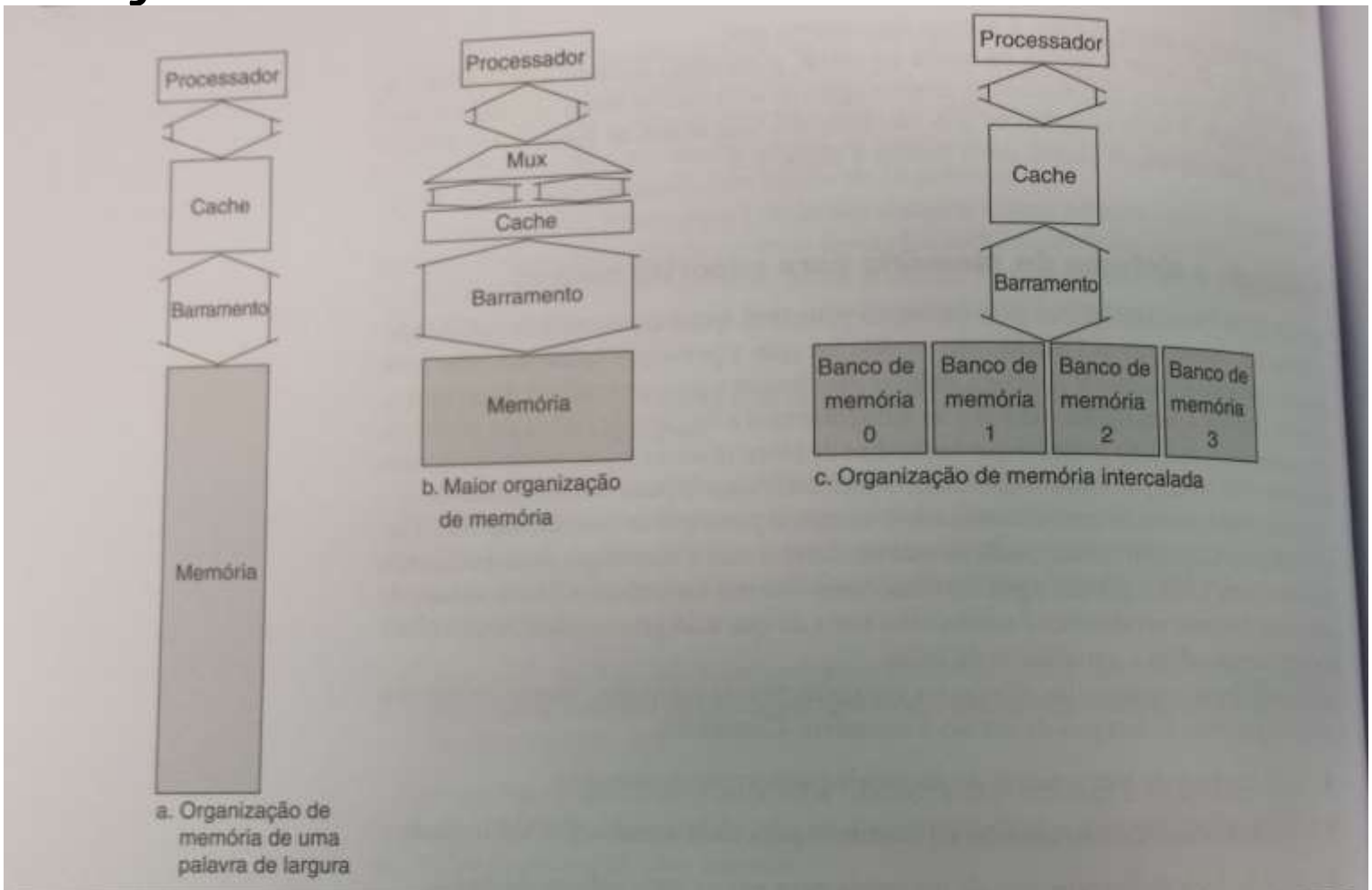

FIGURA 5.11 O principal método para obter largura de banda de memória mais alta é aumentar a largura física ou lógica do sistema de memória. Nesta figura, a largura de banda da memória é melhorada de duas maneiras. O projeto mais simples (a) usa uma memória na qual todos os componentes possuem uma palavra de largura; (b) mostra uma memória, um barramento e uma cache mais largos; enquanto (c) mostra um barramento e uma cache mais estreitos com uma memória intercalada. Em (b), a lógica entre a cache e o processador consiste em um multiplexador usado em leituras e lógica de controle para atualizar as palavras apropriadas da cache nas escritas.