Arquitetura e Organização de Computadores II

Memória compartilhada centralizada

Prof. Nilton Luiz Queiroz Jr.

Arquiteturas de memória centralizada

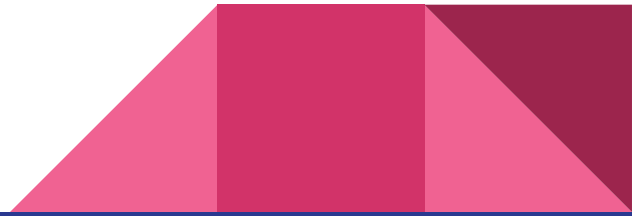
- Grandes caches com múltiplos níveis podem reduzir as demandas por largura de banda de memória;
- A observação desse fator que motiva multiprocessadores de memória centralizada;

Arquiteturas de memória centralizada

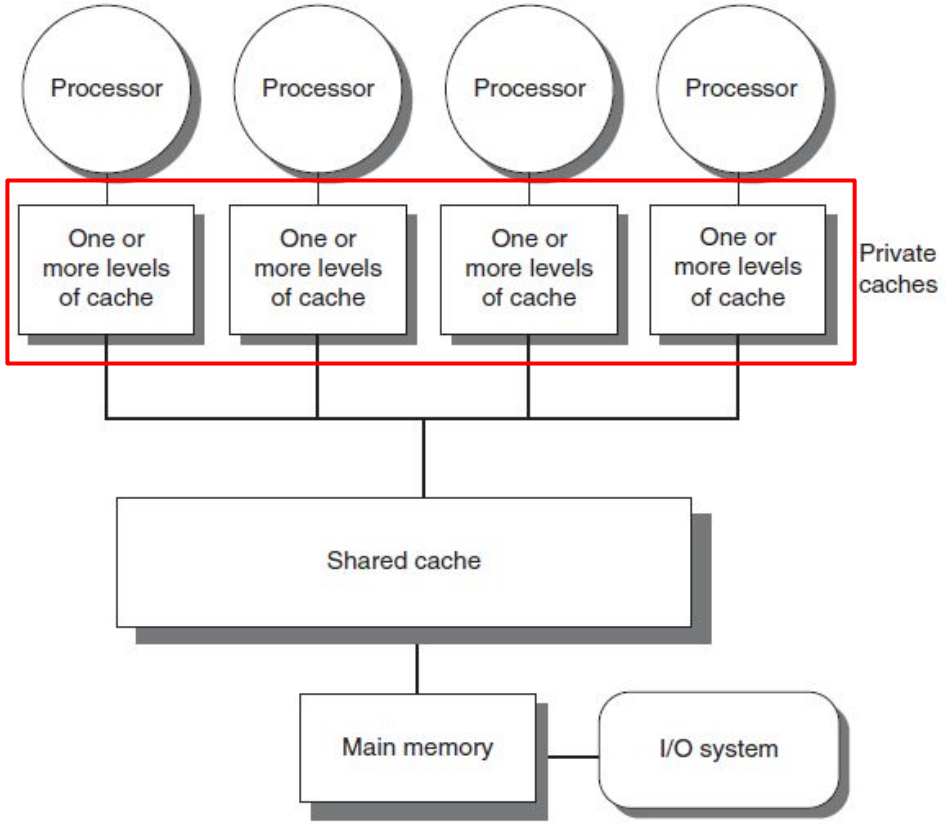
- Máquinas simétricas de memória compartilhada normalmente admitem dois tipos de cache de dados:
  - Privado;
  - Compartilhado;

Cache privada

- Cada multiprocessador tem sua cache privada;
  - Cache L1 e L2 por exemplo;
- A cache de dados privados é usada somente por um processador;
- Quando itens privados são colocados em cache existem dois fatores que são impactados:
  - O tempo de acesso a esse item;
  - A largura de banda requerida;



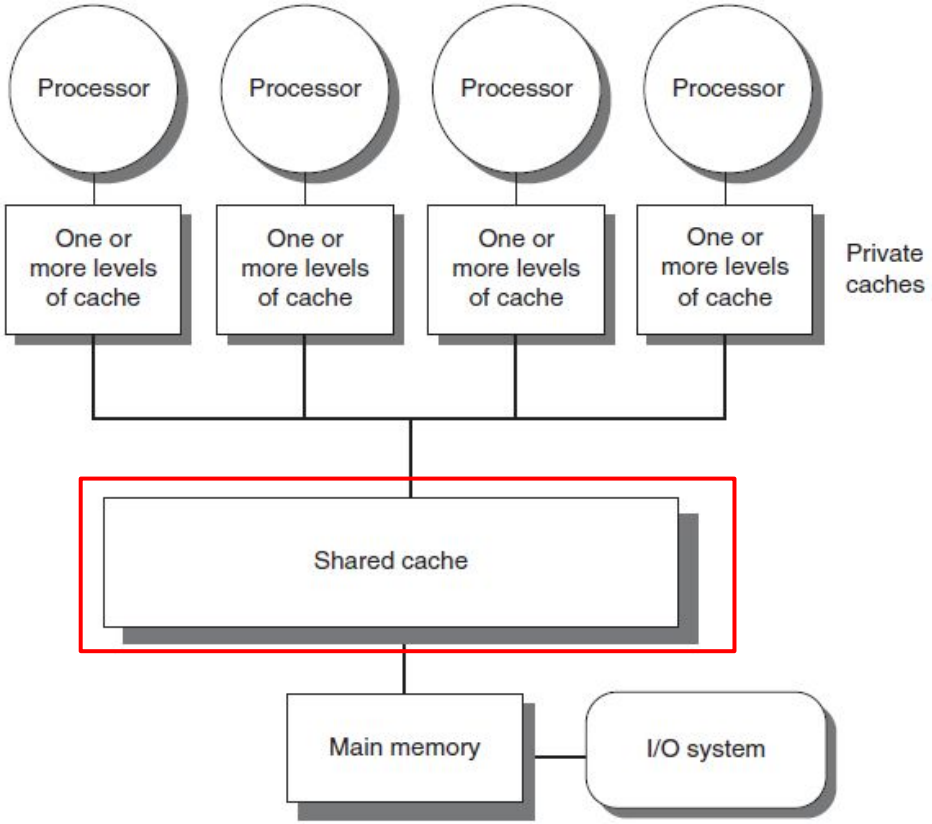
Cache privada



Cache compartilhada

- Usados por diversos processadores;
  - Cache L3 por exemplo;
- Proveem comunicação entre os processadores através de leituras e escritas de dados compartilhados;
- Quando dados compartilhados são colocados na cache o valor compartilhado pode ser replicado nas caches privadas;
  - Reduz a latência de acesso;
  - Reduz largura de banda exigida;
  - Pode reduzir disputa nos dados que estão sendo lidos simultaneamente por multiplos processadores;
- Pode causar problemas de coerência de cache;

Cache compartilhada



Coerência de cache

- Caching compartilhada introduz o problema de coerência de cache pois a visão de dois processadores distintos depende de suas caches;
  - Se não forem tomadas medidas adequadas alguns problemas podem ocorrer;
- Em um determinado estado, dois processadores podem ter valores diferentes para o mesmo endereço de memória, o que geraria erros de execução;

Time	Event	Cache contents for processor A	Cache contents for processor B	Memory contents for location X
0				1
1	Processor A reads X	1		1
2	Processor B reads X	1	1	1
3	Processor A stores 0 into X	0	1	0

Coerência de cache

- Pode-se dizer que temos dois aspectos diferentes do comportamento do sistema de memória:
  - Coerência;
  - Consistencia;
- Ambos são essenciais para escrita de programas corretos de memória compartilhada;

Coerência de cache

- A coerência é o aspecto que define quais valores podem ser retornados por uma leitura;
- Diz-se que um sistema de memória é coerente se:
  1. Uma leitura por um processador P em um local X que feita após uma escrita por P em um local X, sem escritas de outros processadores em X, sempre retornará o valor escrito por P;
  2. Uma leitura por um processador em um local X que é feita após uma escrita por outro processador no mesmo local X, retorna o valor escrito, se a leitura e escrita estão suficientemente separadas no tempo, e nenhuma outra escrita ocorreu no intervalo;
  3. Escritas no mesmo local por dois processadores são serializadas, ou seja, duas escritas no mesmo local por dois processadores são vistas na mesma ordem por todos processadores;

Coerência de cache

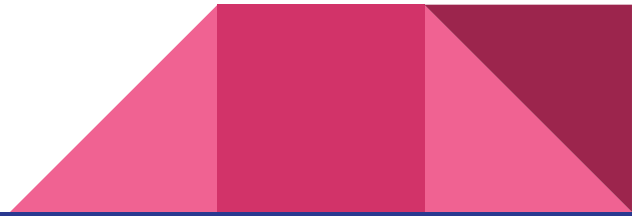
- A propriedade 1 simplesmente preserva a ordem do programa;
- A segunda propriedade define a noção do que significa ter uma visão coerente de memória;
  - Impedir que processadores possam ler valores “antigos” evita incoerências da memória;
- A terceira propriedade é a mais sutil, porém é igualmente importante:
  - Imagine se a escrita não fosse serializada;
    - Suponha duas escritas em X por dois processadores P1 e P2, sendo P1 o primeiro a escrever, e P2 o segundo.
    - Caso as escritas não fossem serializadas algum processador poderia ver primeiro a escrita de P2 e em seguida a de P1 mantendo assim o valor escrito por P1;

Consistencia

- Quando um valor será visto também é uma importante questão;
- Não se pode exigir que uma leitura de um local X veja instantaneamente o valor escrito em X por um outro processador;
    - Uma escrita de X em um processador preceder uma leitura de X de um outro processador e o intervalo de tempo entre tais eventos for muito curto, não é garantido que a alteração seja propagada;

Coerência e Consistência

- Coerência e Consistência são complementares;
  - Coerência define o comportamento de escritas e leituras no mesmo local de memória;
  - Consistência define o comportamento de leituras e escritas com respeito ao acesso de outras localizações de memória;



Coerência de cache

- Para questões de coerência, a princípio, iremos assumir as duas afirmações a seguir:
  - Uma escrita não está completa até que todos processadores vejam suas mudanças;
  - A ordem das escritas não são alteradas em relação a outras instruções de acesso a memória;
- Tais condições implicam que:
  - Se um mesmo processador escreve em um local A, e em seguida num outro local B, todos processadores que veem as mudanças feitas em B obrigatoriamente verão as mudanças em A;
    - Com isso, leituras podem ser reordenadas, porém as escritas devem ser feitas em ordem;

Coerência de cache

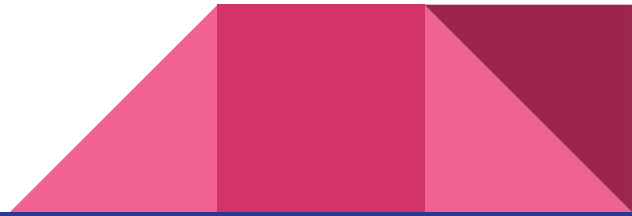
- Programas que executam em mais de um processador normalmente terá cópia dos seus dados nas caches dos processadores que o executam;
- Multiprocessadores coerentes oferecem migração e replicação dos dados compartilhados;
- O suporte a migração e replicação é essencial para a performance em acessar dados compartilhados;

Coerência de cache

- Migração:
  - Um dado pode ser movido para uma cache local e usado lá de uma maneira transparente;
  - Reduz a latência de acesso aos dados compartilhados;
  - Reduz a largura de banda necessária para os dados compartilhados;
- Replicação:
  - Um dado pode ser simultaneamente lido;
    - É feita uma cópia do dado na cache local;
  - Reduz latência de acesso aos dados compartilhados;
  - Reduz a disputa por dados compartilhados;

Coerência de cache

- Para evitar a necessidade de resolver esses problemas no software, foram adotados protocolos para solução de tais problemas em hardware;
  - Protocolos de coerência de cache;
- Atualmente usam-se duas classes de protocolos:
  - Baseado em diretório;
  - Snooping;



Baseado em diretório

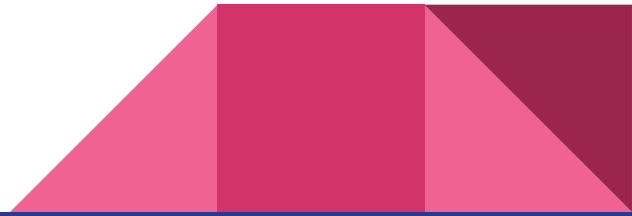
- A coerência de cache baseada em diretório mantém o estado de compartilhamento de um bloco de memória física;
  - Esse local é chamado de diretório;
- Pode-se usar um único diretório centralizado associado com a memória ou um outro tipo de ponto único de centralização;
  - Como a cache mais externa de multiprocessadores;
- Não faz muito sentido o uso de diretórios centralizados em memória distribuída compartilhada;
  - Criaria um único ponto de disputa e tornaria difícil escalar para diversos processadores;
  - Esquemas de diretório para DSM são mais complexos;

Snooping

- A coerência feita por snooping usa a ideia de que cada cache que tem uma cópia do bloco é responsável por verificar seu status;
- Em processadores simétricos as caches são geralmente todas acessíveis por um meio de broadcast;
- Usa controladores de cache para determinar se eles tem ou não uma cópia do bloco requisitado;
- Podem ser usados como protocolos de coerência para multiprocessadores multichip;
    - O snoop pode ser aplicado com uma estrutura de diretório em cada multiprocessador

Protocolos snooping

- Existem duas maneiras de implementar coerência:
  - Atualizar todas as cópias nas diferentes caches;
    - Chamado de protocolo de atualização de escrita, ou protocolo de broadcast de escrita;
  - Garantir que somente um processador terá acesso a um item antes de escrever nele;
    - Chamado de protocolo de invalidação de escrita;



Protocolo de broadcast de escrita

- A memória está sempre atualizada;
  - Miss em loads não tem problema;
- Sempre que uma escrita é feita deve ser usado o barramento para atualizar todos os valores;

Protocolo de invalidação de escrita

- O protocolo de invalidação de escrita não permite outras escritas ou leituras de um item quando a escrita ocorre;
  - Todas as outras cópias do item são invalidades;
  - A escrita deve ser exclusiva para impedir escritas simultâneas;
  - A leitura deve ser exclusiva para não ler dados errados;

Processor activity	Bus activity	Contents of processor A's cache	Contents of processor B's cache	Contents of memory location X
				0
Processor A reads X	Cache miss for X	0		0
Processor B reads X	Cache miss for X	0	0	0
Processor A writes a 1 to X	Invalidation for X	1		0
Processor B reads X	Cache miss for X	1	1	1

Protocolo de invalidação de escrita

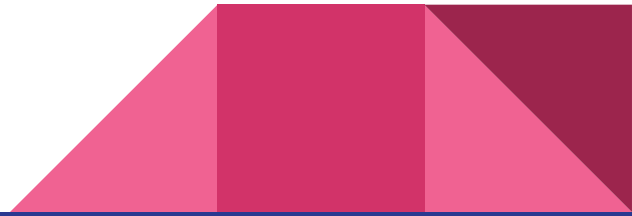
- Quando ocorrer a tentativa de duas escritas simultâneas apenas um consiguirá escrever;
  - Isso irá invalidar o dado do outro, assim terá um miss na cache, e terá que obter o novo valor para a escrita;
  - O processador que fez a primeira escrita pode ser o responsável por responder o miss na cache que o segundo processador teve;
    - Pode-se também atualizar a memória;;
      - Atualizar “instantaneamente”;
      - Atualizar quando o bloco for substituído;
        - Requer monitoramento do “proprietário” da mudança;
          - Responsável por atualizar todas as outras caches e também a memória;

Protocolo de invalidação de escrita

Processor activity	Bus activity	Contents of processor A's cache	Contents of processor B's cache	Contents of memory location X
				0
Processor A reads X	Cache miss for X	0		0
Processor B reads X	Cache miss for X	0	0	0
Processor A writes a 1 to X	Invalidation for X	1		0
Processor B reads X	Cache miss for X	1	1	1

Coerência de cache

- A maior parte dos processadores implementam o protocolo de invalidação de escrita;
  - O protocolo de invalidação consome uma quantidade menor de banda que o protocolo de broadcast;



Referências

HENNESSY, John L.;PATTERSON, David A. Computer architecture: a quantitative approach. Elsevier, 2011.