Sistemas de Computação

Mestrado Integrado em
Engenharia de Telecomunicações e Informática

2015/2016

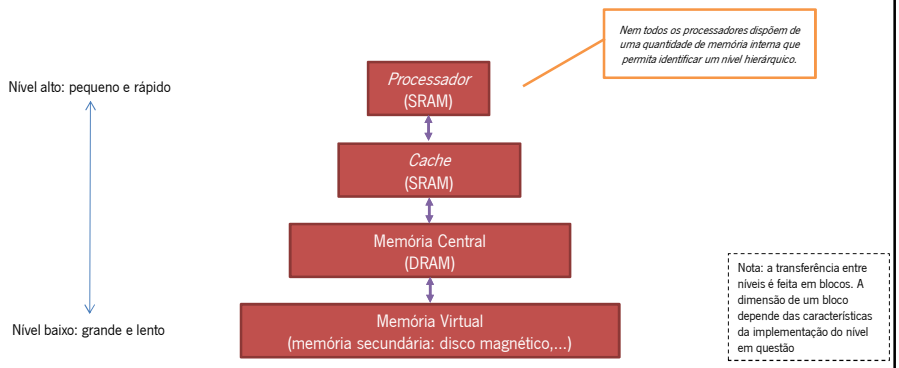
Hierarquia da memória (I)

- Princípios de programação estruturada conduzem ao conceito de localidade – programas acedem a um espaço de endereçamento limitado, em cada instante de tempo.
- **Localidade temporal** (resulta de ciclos): um item referenciado tem grande probabilidade de o ser novamente, num curto espaço de tempo;
- **Localidade espacial** (resulta da natureza sequencial dos programas): quanto um item é referenciado, há uma elevada probabilidade de os seus vizinhos o serem de seguida
- Programas e dados estão em memória.
  - Como conciliar grandes capacidades de memória com custos e um desempenho elevado?

Tecnologia	*Tempo de acesso (ms)*	*Custo/Mbyte (Eur)*
SRAM	2-5	1-5
DRAM	20-50	0.1-0.15
Discos magnéticos	7,000,000 – 15.000.000	0.0006-0.001

Hierarquia da memória (II)

- Como conciliar grandes capacidades de memória com custos e um desempenho elevado?
- Explorando o conceito de localidade e implementando uma hierarquia de memórias

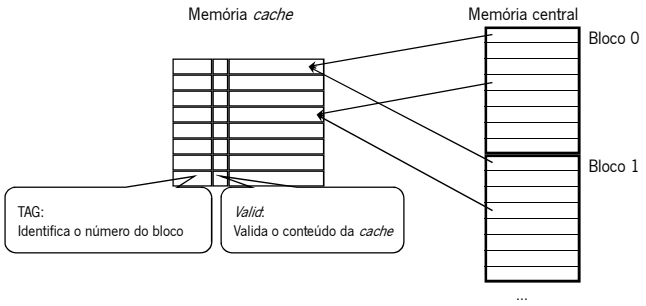


Hierarquia da memória (III)

- Definições
  - ***Hit*** - quando o processador acede a um item que se encontra no nível superior
  - ***Miss*** - por oposição ao Hit
  - ***Hit Rate*** - fracção de acessos à memória que se traduzem em Hits
  - ***Miss Rate*** - (1.0 - Hit Rate)
  - ***Hit Time*** - tempo de acesso ao nível superior, incluindo o tempo de procura
  - ***Miss penalty*** - tempo de actualização do nível superior, com um bloco do nível inferior
- *Hit Time* tem que ser muito menor que o *Miss Penalty* !!
- Estrutura dos programas tem um impacto muito grande na efectiva utilização desta hierarquia (compiladores)
- A gestão desta hierarquia é partilhada pelo hardware, pelo sistema operativo e, por vezes, pelas aplicações (memória virtual)

Hierarquia da memória (IV)

- *Cache*
  - Historicamente o nível entre o processador e a memória central. De uma forma mais genérica, designa qualquer meio de armazenamento implementado por forma a explorar a "localidade" dos programas.
  - Como controlar o conteúdo da *cache*? Uma solução usa "mapeamento directo", particularmente simples de implementar



Hierarquia da memória (V)

- Exemplo (apenas com leitura), para uma *cache* de 8 palavras e uma memória de 32 palavras. Sequência de acessos às posições de memória:
  22, 26, 22, 16, 18 e 26

Tabela de acessos

Endereço	*Hit/Miss*	*Posição na cache*
22 – 10110	Miss	10100 mod 8 = 110
26 – 11010	Miss	11010 mod 8 = 010
22 – 10110	Hit	10110 mod 8 = 110
16 – 10000	Miss	10000 mod 8 = 000
18 – 10010	Miss	10010 mod 8 = 010
26 – 11010	Miss	11010 Mod 8 = 010

Cache

Endereço	*V*	*Tag*	*Dado*
000			
001			
010			
011			
100			
101			
110			
111			

Hierarquia da memória (VI)

- Operações de escrita
  - Evitar a inconsistência entre a cache e a memória central
  - Esquema mais simples é garantir que as operações de escrita na memória afectam tanto a cache como a memória central (sincrono) – ***write-through***
  - Outra solução é usar ***write-back***: a escrita inicial é feita na cache e a escrita na memória central ocorre apenas quando o bloco de dados da cache é modificado ou substituido por outro conteúdo. (*copy-back cache*)

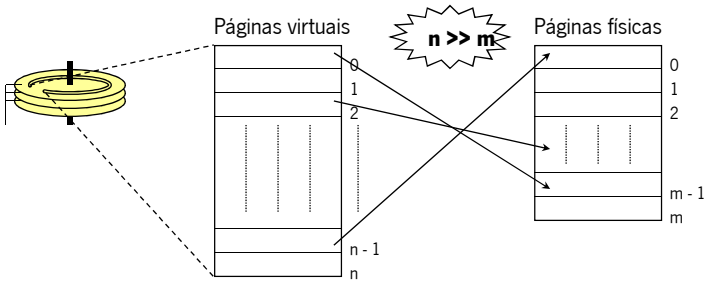
Memória Virtual (I)

- Memória virtual:
  - Método para aumentar, virtualmente, a quantidade de memória central
  - A memória virtual implementa a tradução do espaço de endereçamento do programas para os endereços fisicos. Desta forma existe protecção ao espaço de memória de cada programa!
  - Vantagens:
    - Programas maiores do que a memória disponível
    - Maior eficiência na partilha do processador (multitasking)
  - Mecanismos semelhantes aos utilizados para a cache: bloco é designado por página; o *miss* é designado de *page fault*.
  - *Virtual Address Space*: espaço de endereçamento virtual disponível para cada aplicação (4Gbytes na plataforma wintel – 32 bits)
  - Problemas de implementação: elevado custo dos page fault (centenas de milhares de ciclos de clock!).
    - O tamanho das páginas deve amortizar o tempo de acesso (4Kbytes, 16kBytes, 32KBytes, 64KBytes)
    - Reduzir a taxa de ocorrências de *page faults*

Memória Virtual (II)

- Memória virtual:
  - Gestão das páginas:
    - Que páginas carregar e para onde?
    - Como libertar as páginas ocupadas na memória central?
  - Operações de escrita
    - Write-through: esta técnica implica um tempo de acesso proibitivo!
  - Transformação de endereços virtuais em endereços reais: em tempo-real, dentro do próprio processador – *TLB-Translation Look-ahead Buffer*

Memória Virtual (III)



Mapeamento flexível facilita a gestão da memória
(carregamento de programas e a gestão do espaço livre)

Memória Virtual (IV)

- Flexibilidade no mapeamento:
  - O sistema operativo pode substituir qualquer página na memória central
  - Mecanismo de transformação – ***page table***
    - Uma *page table* por cada programa
  - Substituição de páginas: algoritmo LRU (*Least Recently Used*) – segundo o princípio da localidade temporal, a página utilizada há mais tempo é a melhor candidata para substituição
  - Zona *swap* armazena as páginas temporariamente removidas, pelo sistema operativo, da memória central

Hierarquia de memória – Memória Virtual

- Conclusões
  - Hierarquia da memória procura minimizar o efeito da memória central ser constituída por circuitos DRAM (lentos e de capacidade "limitada"), explorando o principio da localidade (espacial e temporal). Mas...
  - Velocidade das CPUs continua a aumentar a um ritmo mais elevado do que o da diminuição do tempo de acesso das memórias (ou discos):
    - Caches multinível (actualmente são implementados 2 níveis, sendo um deles interno ao circuito da CPU)
    - Desenvolver melhores estruturas de memória DRAM
    - Melhorar o desempenho dos compiladores, explorando melhor a hierarquia da memória
      - Reorganizar os programas de forma a evidenciarem melhor localidade
      - Utilizando *prefetching* - o compilador pode antever que blocos de memória são necessários, e desencadear a sua transferência para níveis mais baixos da hierarquia antes de serem referenciados.