Aula 5: Hierarquia de Memória - Parte II

Professor(a): Virgínia Fernandes Mota
http://www.dcc.ufmg.br/∼virginiaferm

OCS (TEORIA) - SETOR DE INFORMÁTICA

Memória Virtual

- Considere vários programas executados ao mesmo tempo em um computador
- Memória total exigida pode ser maior que memória disponível
- Apenas fração da memória usada ativamente em um dado momento
- Memória principal contém apenas memória usada
- Programas devem ser protegidos uns dos outros

Memória Virtual

- Ao compilarmos programas, não sabemos com quem eles irão compartilhar memória
- Próprios programas mudam dinamicamente durante execução
- Cada programa deve ter seu próprio espaço de endereçamento
  - Faixa distinta de locais de memória acessível apenas a esse programa
  - Memória virtual traduz espaço de endereçamento do programa para endereços físicos

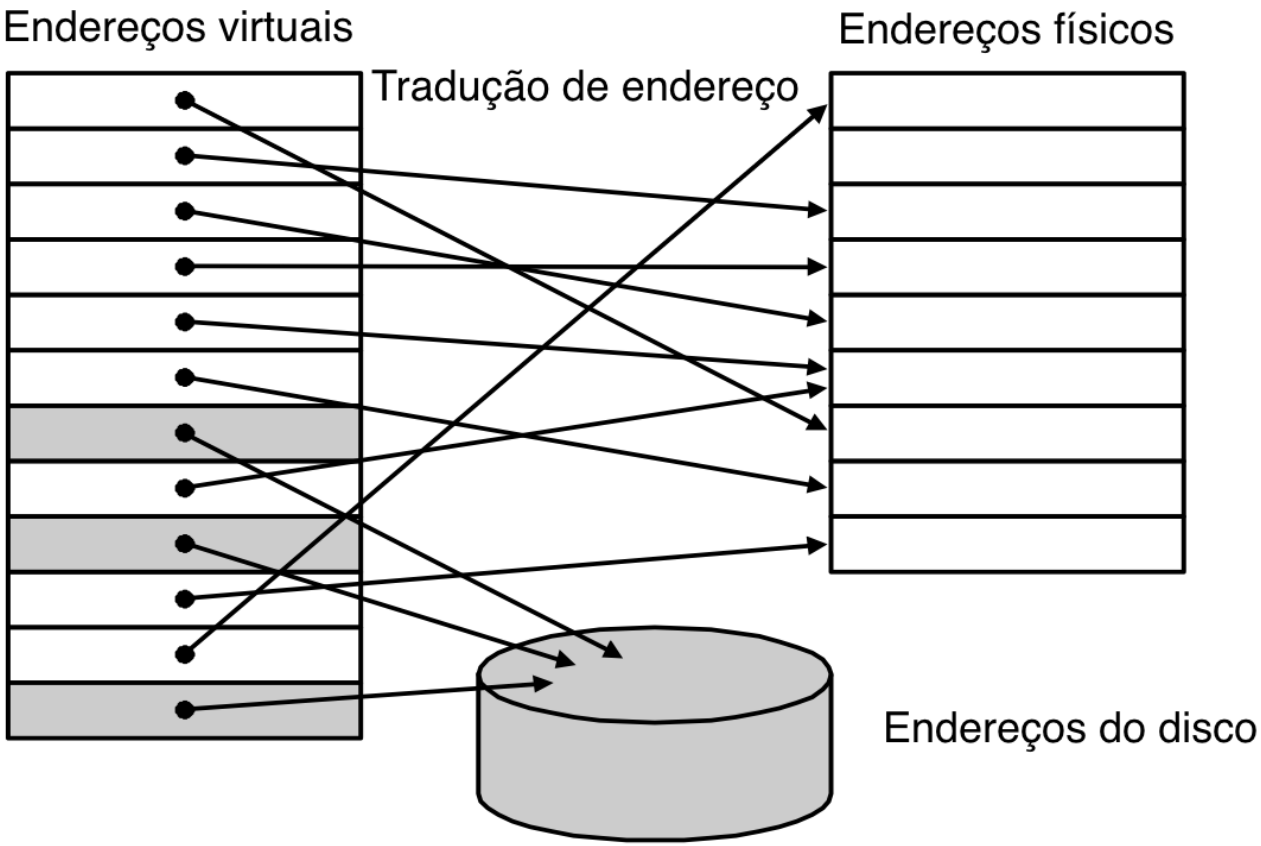
Memória Virtual

- Permitir que programa do usuário exceda tamanho da memória principal
- Técnica anterior: uso de overlays
  - Programador dividia programa em partes
  - Cada parte cabia na memória principal
  - Programa só acessava overlay carregado
- Memória virtual faz gerenciamento automático!

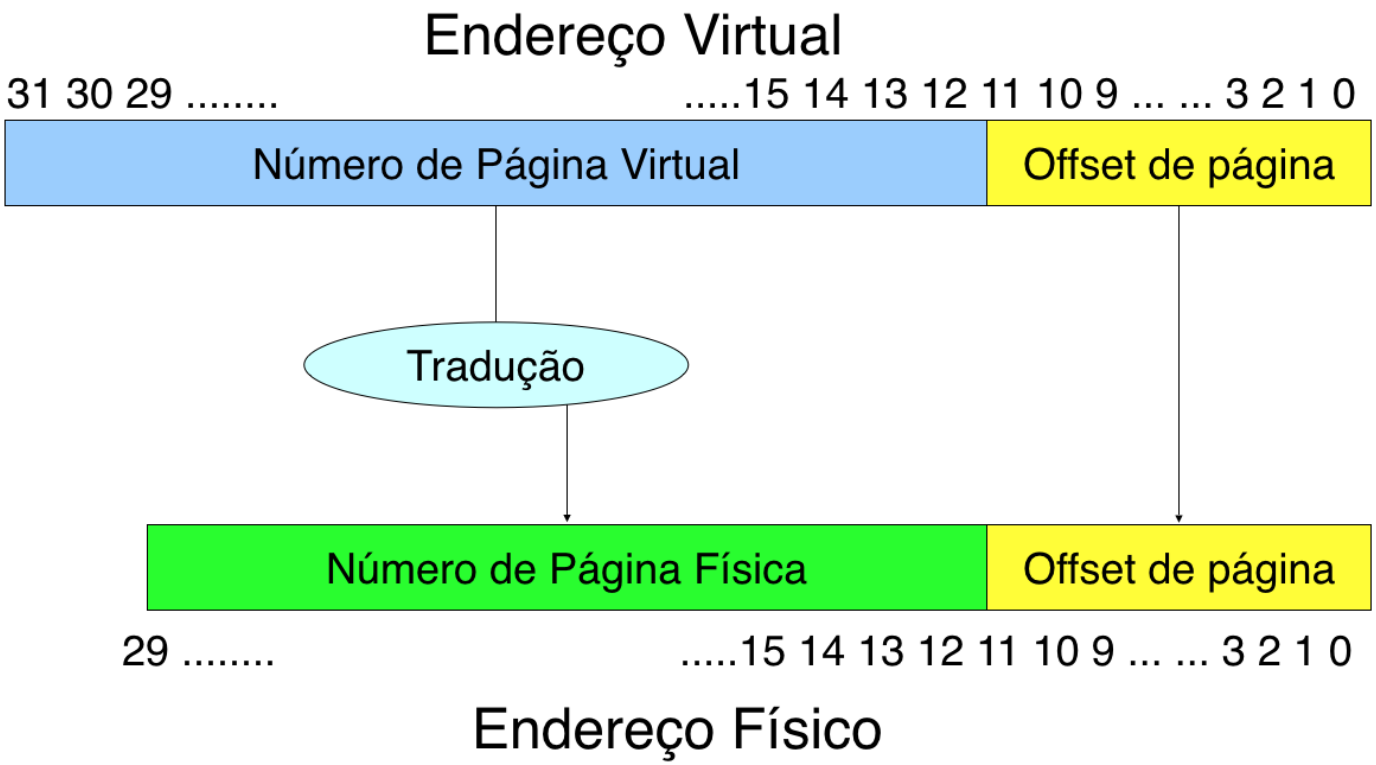
Terminologia

- **Página**: bloco de memória virtual
- **Falha de página**: quando página acessada não está presente na memória principal
- **Endereço virtual**: endereço que corresponde a um local no espaço virtual e é traduzido pelo mapeamento de endereço para um endereço físico quando a memória é acessada
- **Relocação**: mapeia endereços virtuais usados por um programa para diferentes endereços físicos antes que os endereços sejam usados para acessar a memória
- Permite carregar programas em qualquer parte da memória
- Usados blocos de tamanho fixo
- **Segmentação**: esquema de mapeamento de endereço de tamanho variável em que um endereço consiste de duas partes: número de segmento, e um offset do segmento
- Focaremos apenas em **paginação**!

Terminologia



Mapeamento de Endereço Virtual em Endereço Físico



Mapeamento de Endereço Virtual em Endereço Físico

- Endereço virtual desmembrado em: Número e Offset.
- Número de página virtual:
  - Bits mais significativos
  - Número de página endereçáveis não precisa corresponder com o endereço físico
  - Quantidade maior é a base para ilusão de memória ilimitada
  - Ex: Página virtuais: 20 bits => 4GB; Páginas Físicas: 18 bits => 1 GB
- Offset de página:
  - Bits menos significativos
  - Não alterados
  - Quantidade de bits determina o tamanho da página
  - Ex: 12 bits => 4 KB

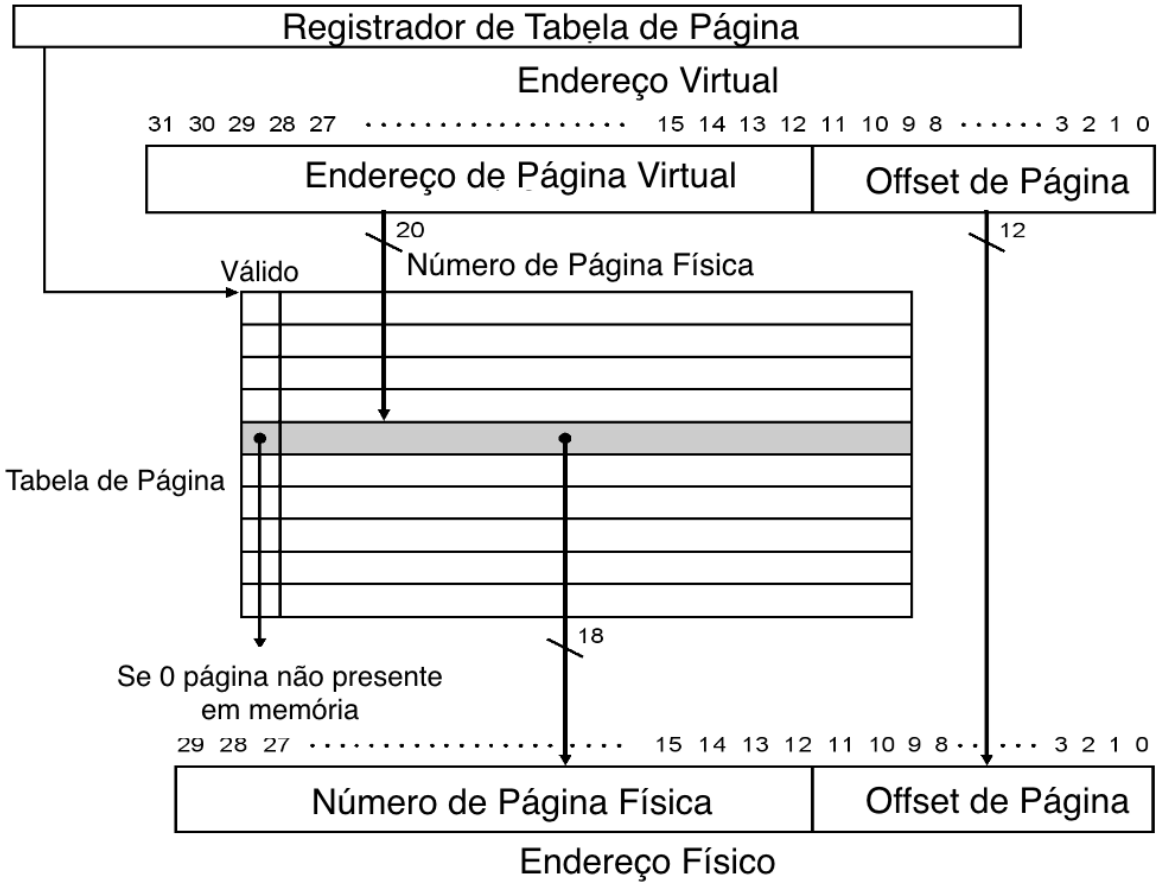
Mapeamento de Endereço Virtual em Endereço Físico

- Escolhas no projeto nos sistemas de memória virtual motivados pelo alto custo de uma falha de página
- Milhões de ciclos para ser processada
- Páginas devem ser grandes o suficiente para amortizar o longo tempo de acesso
  - Atualmente 4KB a 16KB; novos sistemas: 32KB a 64KB - Mais?
- Uso de organizações que reduzam falha de páginas
  - Totalmente associativa
- Falhas de páginas tratadas por software
  - Custo adicional (overhead) pequeno comparado ao tempo de acesso ao disco
- Uso de *copy-back* (com bit de modificação)

Posicionando uma Página e a Encontrando Novamente

- Dificuldade do posicionamento totalmente associativo: localizar entrada
- Tabela de páginas usada para localizar páginas
- Tabela indexada pelo número da página do endereço virtual para descobrir número da página física correspondente
- Cada programa possui sua própria tabela
- Local da tabela de página indexado por um registrador (registrador de tabela de páginas)

Posicionando uma Página e a Encontrando Novamente



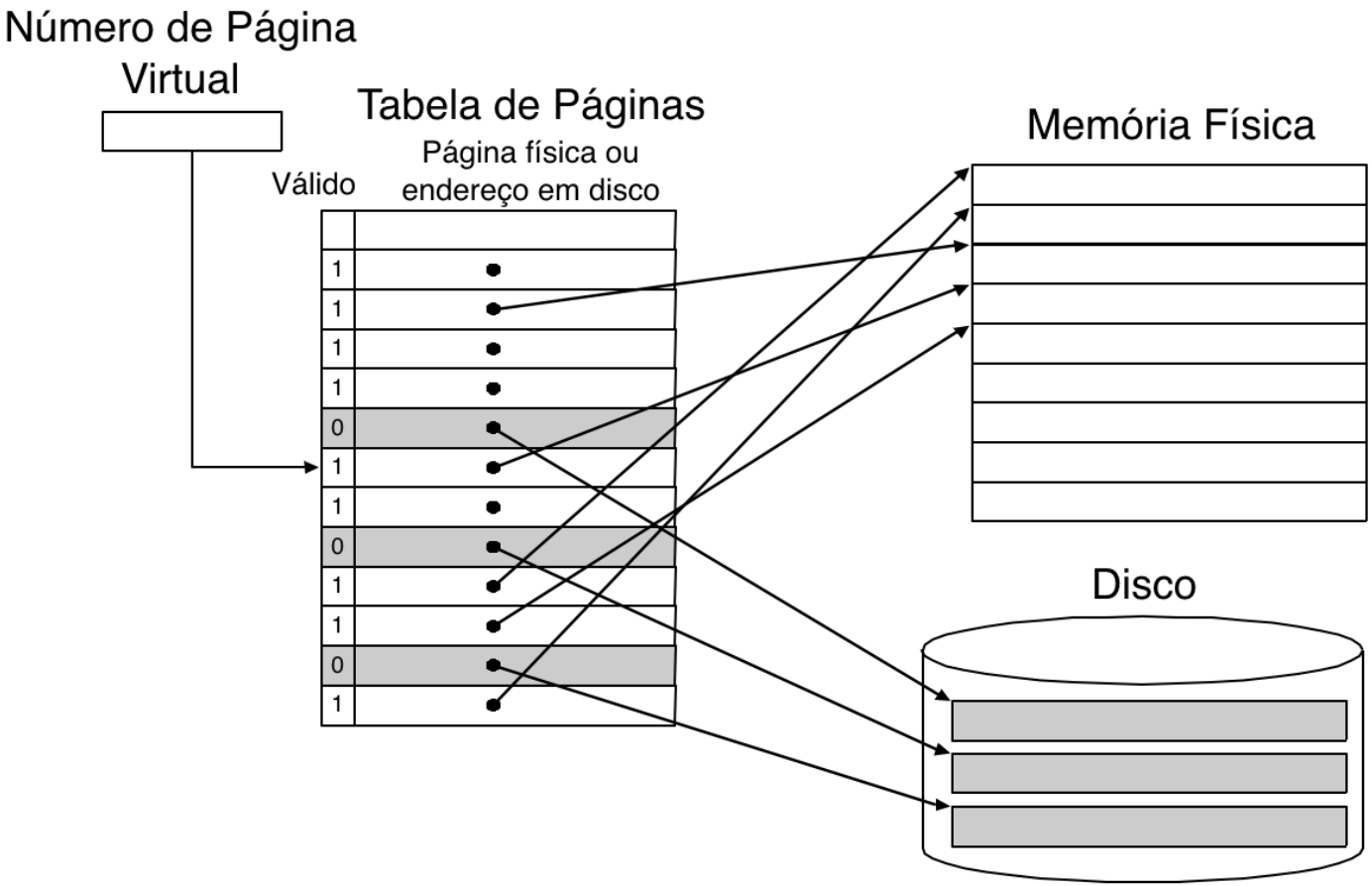
Falha de Página

- Se bit de validade desligado, ocorre **falha de página**
- SO deve receber o controle
- Transferência feita pelo mecanismo de exceção
- SO encontra a página e decide onde colocá-la na memória principal
  - Endereço virtual não diz imediatamente onde está a página no disco
  - Página deve ser monitorada

Falha de Página

- SO cria espaço no disco para todas as páginas de um processo no momento em que este é criado: **Área de swap**
- Estrutura criada para registrar onde cada página virtual está armazenada no disco: Pode ser parte da tabela de páginas ou estrutura de dados auxiliar

Falha de Página



Falha de Página

- SO também cria estrutura que controla quais processos e quais endereços virtuais usam cada página física
- Caso todas as páginas de memória estejam em uso quando falha de página ocorre, SO deve escolher que página substituir
  - Página que supostamente não será usada no futuro escolhida
  - Passado usado para prever futuro: (LRU - *last recently used*)
  - Bit de referência usado

Reduzindo o Tamanho da Tabela de Páginas

- Endereço de 32 bits, páginas de 4KB e 4 bytes por entrada da tabela de página
- Número de entradas = 232 / 212 = 220
- Tamanho da tabela = 220 x 4 bytes = 4MB
- 4 MB por programa em execução!
- Várias técnicas para reduzir armazenamento necessário

Reduzindo o Tamanho da Tabela de Páginas

- Uso de registrador limite: Tabela se expande somente quando número de página virtual for maior que conteúdo do registrador
- Dividir tabela em duas: Cada uma cresce em uma direção (pilha e heap)
- Uso de hashing: Tabela precisa ser no máximo do tamanho da memória física

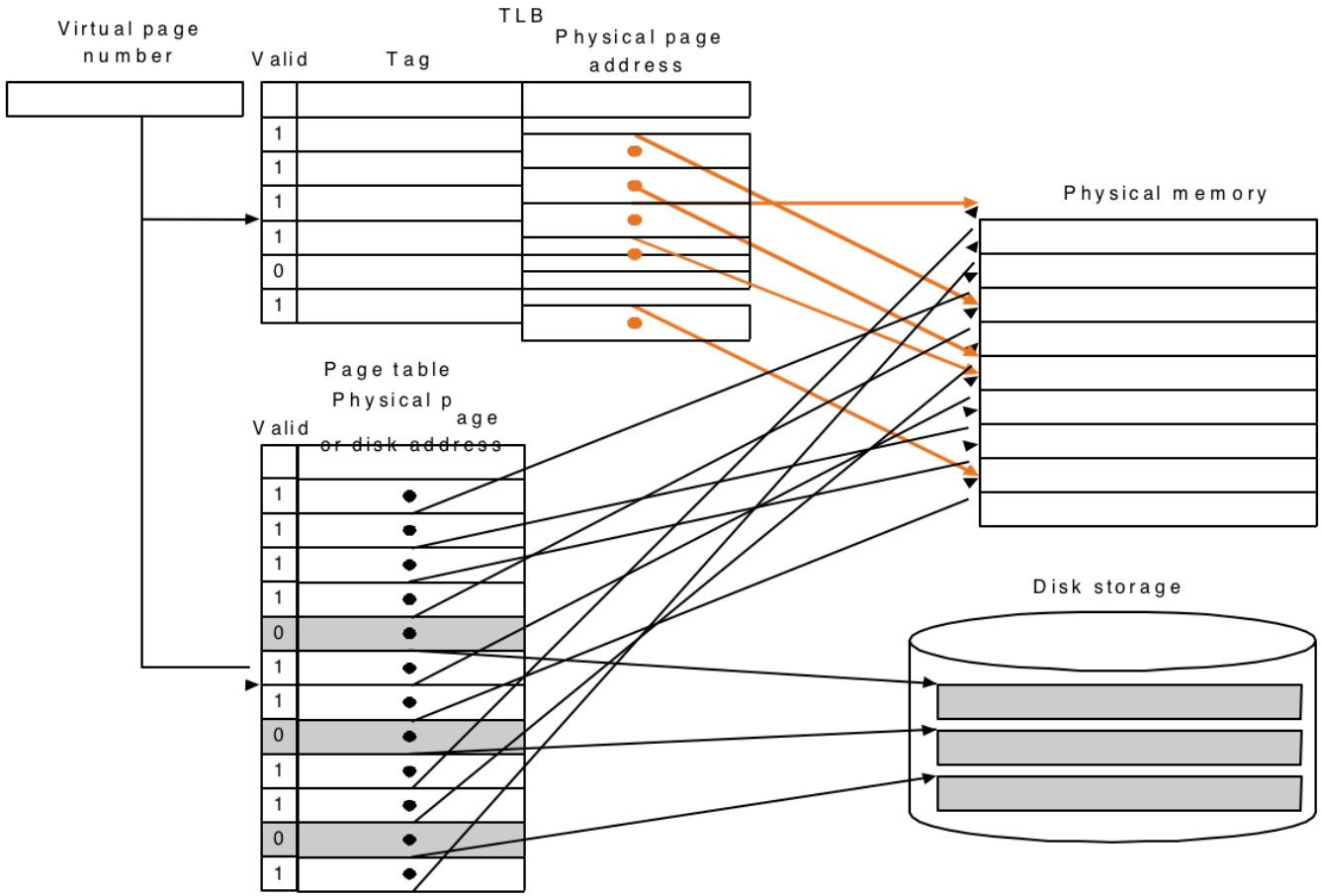
Reduzindo o Tamanho da Tabela de Páginas

- Uso de múltiplos níveis de tabela
- Primeiro nível mapeia grandes blocos de tamanho fixo
- Cada entrada na tabela indica se alguma página do bloco está alocada, e, se estiver, aponta para a tabela de páginas do bloco
- Uso de paginação para as tabelas de páginas

Tornando a Tradução Rápida: TLB

- Cada acesso à memória por um programa pode levar o dobro do tempo
  - Acesso à tabela de páginas para obter endereço
  - Acesso à memória, para buscar dado
- Cache usada para monitorar endereços recentemente usados
  - Translation Lookaside Buffer (TLB)
  - Evita acesso à tabela de páginas

Tornando a Tradução Rápida: TLB



Tornando a Tradução Rápida: TLB

- Valores típicos para uma TLB:
- Tamanho: 16 a 512 entradas
- Tamanho do bloco: 1 a 2 entradas da tabela de página
- Tempo de acerto: 0,5 a 1 ciclo de clock
- Penalidade de falha: 10 a 100 ciclos de clock
- Taxa de falhas: 0,01% a 1%

Integrando Memória Virtual, TLBs e Caches

- Sistemas de memória virtual e de cache funcionam em conjunto como uma hierarquia
  - Dados não podem estar na cache a não ser que estejam presentes na memória principal
- SO desempenha importante papel na manutenção dessa hierarquia
  - Conteúdo da página removida da cache
  - Tabela de dados e TLB modificadas de modo que tentativa de acessar dados gere falha de página
- Melhor caso: Endereço virtual é traduzido pela TLB e enviado para a cache, onde dados são encontrados

Tratando Falhas de TLB e Falhas de Páginas

- Tratamento de falhas de TLB e de páginas complexo.
- Falha de TLB pode indicar duas possibilidades:
  - Página presente na memória: Precisamos criar entrada na TLB
  - Página ausente: transferir controle para SO
- Como saber qual situação ocorreu?

Tratando Falhas de TLB e Falhas de Páginas

- Se entrada na tabela de páginas tiver bit de validade desligado, a página correspondente não está em memória
  - Falha de página
- Se o bit estiver ligado, temos apenas que recuperar entrada desejada

Estrutura Comum para Hierarquias de Memória

- Muitos aspectos das hierarquias de memória diferem quantitativamente:

Recurso	Valores Típicos para Caches L1	Valores Típicos para Caches L2	Valores Típicos para Memória Paginada	Valores Típicos para TLB
Tamanho total em blocos	250 - 2000	4000 – 250.000	16.000 – 250.000	16 – 512
Tamanho total em kilobytes	16 – 64	500 – 8.000	250.000 – 1.000.000.000	0,25 – 16
Tamanho do bloco em bytes	32 – 64	32 – 128	4.000 – 64.000	4 – 32
Penalidade da perda em clocks	10 – 25	100 – 1.000	10.000.000 – 100.000.000	10 – 1000
Taxas de perda (global para L2)	2% - 5%	0,1% - 2%	0,00001% - 0,0001%	0,01% - 2%

Estrutura Comum para Hierarquias de Memória

- Questões comuns a todos os níveis de uma hierarquia de memória:
  1. 1: onde um bloco pode ser colocado?
  2. 2: como um bloco é encontrado?
  3. 3: que bloco deve ser substituído em uma falha de cache?
  4. 4: o que acontece em uma escrita?

Estrutura Comum para Hierarquias de Memória

- **Questão 1**: onde um bloco pode ser colocado?

Nome do Esquema	Número de Conjuntos	Blocos por Conjunto
Diretamente mapeado	Número de blocos na cache	1
Associativo por conjunto	$\frac{\text{Número de blocos na cache}}{\text{Associatividade}}$	Associatividade (normalmente 2 a 16)
Totalmente associativo	1	Número de blocos na cache

- Aumento da associatividade normalmente reduz taxa de falhas: Redução das falhas que disputam o mesmo local

- **Questão 2**: como um bloco é encontrado?
- Depende do esquema de posicionamento do bloco

Associatividade	Método de Localização	Comparações Necessárias
Diretamente mapeado	Indexação	1
Associativo por conjunto	Indexação do conjunto, pesquisa entre os elementos	Grau de associatividade
Totalmente associativo	Pesquisa de todas as entradas de cache	Tamanho da cache
	Tabela de consulta separada	0

Estrutura Comum para Hierarquias de Memória

- Escolha entre diretamente mapeado, associativo por conjunto ou totalmente associativo depende do custo da falha comparado com o custo de implementar a associatividade
  - Em termos de hardware extra e em termos de tempo
- Uso de cache L2 permite associatividade mais alta

Estrutura Comum para Hierarquias de Memória

- Sistema de memória virtual
  - Tabela de mapeamento separada mantida para indexar memória: Custo extra para armazenar tabela
  - Uso do índice exige acesso extra à memória
  - Escolha da associatividade total motivada por:
    - Falhas serem muito caras
    - Software de substituição sofisticado para reduzir taxa de falhas
    - Mapa completo dispensa pesquisa ou hardware extra
    - Tamanho de página maior significa menor custo adicional do tamanho da tabela de página

- **Questão 3**: que bloco deve ser substituído em uma falha de cache?
  - Que bloco substituir em cache associativa?
    - Substituição aleatória
    - LRU
  - Implementação LRU mais onerosa
    - Com mais do que um pequeno grau de associatividade (geralmente, 2 a 4)
    - LRU normalmente aproximado para 4 vias

Estrutura Comum para Hierarquias de Memória

- Substituição aleatória possui taxa de falhas 1,1 vezes mais alta que LRU
  - Em cache associativa de duas vias
  - Implementação mais simples
  - Com caches maiores, taxa cai para ambas as caches e diferença absoluta se torna pequena

Estrutura Comum para Hierarquias de Memória

- **Questão 4**: o que acontece em uma escrita?
  - *Write-through*
    - Falhas mais simples e baratas porque nunca exigem que um bloco seja escrito de volta no nível inferior
    - Mais fácil de ser implementado, apesar de necessitar de buffer de escrita
  - *Copy-back* (ou *write-back*)
    - Palavras escritas pelo processador na velocidade em que cache, e não memória, pode aceitar
    - Diversas escritas dentro de bloco exigem apenas uma escrita no nível inferior da hierarquia: Uso efetivo da banda

Origem das Falhas em uma Hierarquia de Memória

- Falhas compulsórias: falhas causadas pelo primeiro acesso a um bloco (falhas de partida a frio)
- Falhas de capacidade: cache não pode conter todos os blocos necessários durante a execução de um programa
- Falhas de conflito: vários blocos competem pelo mesmo conjunto (falhas de colisão)
  - Cache associativa por conjunto ou diretamente mapeada
  - Eliminada em caches totalmente associativa do mesmo tamanho

Origem das Falhas em uma Hierarquia de Memória

- Falhas de conflito podem ser reduzidas aumentando associatividade
  - Pode aumentar tempo de acesso
- Falhas de capacidade podem ser reduzidas aumentando a cache
  - Cautela necessária para evitar aumento no tempo de acesso: Cache de primeiro nível cresce lentamente ou nem isso
- Falhas compulsórias podem ser reduzidas aumentando o tamanho do bloco
  - Reduz número de referências necessárias: Programa consiste de menos blocos
  - Aumento na penalidade de falha

Falácias e Armadilhas

- Armadilha: esquecer-se de considerar o endereço em bytes ou o tamanho de bloco de cache ao simular uma cache
- Ex: cache diretamente mapeada de 32 bytes, com tamanho de bloco de 4 bytes. Onde mapeia endereço 36?
  - Endereço em blocos: 36 é o endereço do bloco 9. $9 \bmod 8 = 1$
  - Se 36 for endereço em palavras: $36 \bmod 8 = 4$

Falácias e Armadilhas

- Armadilha: Ignorar o comportamento do sistema de memória ao escrever programas ou gerar código em um compilador
- Armadilha: Usar o tempo médio de acesso à memória para avaliar a hierarquia de memória de um processador com execução fora de ordem

Próxima aula

Exercícios para a prova