Questionário - 14

Arthur C. M. Barcella e Matheus P. Salazar

Explique a diferença entre endereços lógicos e endereços físicos e as razões que justificam o uso de endereços lógicos.

Os processos em execução usam endereços lógicos para acessar memória de forma abstrata, já os endereços físicos representam a localização dos dados na memória física.

Os endereços lógicos permitem o gerenciamento de memória eficiente, fornecem proteção e isolamento entre processos e oferecem suporte a recursos avançados, como memória virtual, já os endereços físicos são necessários para permitir acesso direto à memória física, realizar tarefas de gerenciamento de memória, realizar operações de baixo nível e mapear endereços lógicos para endereços físicos reais durante o armazenamento.

O que é uma MMU – Memory Management Unit?

MMU é um componente que realiza o gerenciamento de memória. Ele é responsável por mapear os endereços virtuais usados pelos programas para endereços físicos na memória real do sistema.

Seria possível e/ou viável implementar as conversões de endereços realizadas pela MMU em software, ao invés de usar um hardware dedicado? Por que?

Sim, as traduções de endereços realizadas pela MMU podem ser implementadas em software, embora geralmente seja mais comum e econômico implementá-las em hardware dedicado.

A implementação de hardware oferece melhor desempenho, menos sobrecarga de CPU e maior flexibilidade na otimização de recursos específicos do sistema. Vale destacar que a segurança teria que ser reforçada, a fim de impedir possíveis ataques.

Explique as principais formas de organização de memória.

A memória pode ser dividida entre física e virtual, onde a memória virtual pode ser organizada de três modos: partições, segmentos e páginas.

As memórias virtuais por partições dividem a memória física em blocos, permitindo que vários programas sejam executados simultaneamente. Quando os programas são iniciados, alguns deles são carregados na memória física, enquanto o restante permanece na memória secundária.

Já na de segmento são alocados dinamicamente conforme necessário e podem ser expandidos ou contraídos durante a execução do programa. Por fim a memória por páginas funciona, quando um programa é iniciado, apenas as páginas necessárias são carregadas na memória física, enquanto as páginas não utilizadas permanecem na memória secundária.

Por que os tamanhos de páginas e quadros são sempre potências de 2?

Por motivos práticos, os tamanhos de página e quadro em sistemas de computador geralmente são 2. Essa opção permite o mapeamento eficiente de endereços de memória para páginas usando operações bit a bit simples.

Além disso, o uso de potências de 2 facilita o alinhamento da memória, garantindo que os dados residem em endereços que são múltiplos do tamanho da palavra.

Considerando a tabela de segmentos a seguir (com valores em decimal), calcule os endereços físicos correspondentes aos endereços lógicos 0:45, 1:100, 2:90, 3:1.900 e 4:200.

base	44	200	0	2000	1200
limite	810	200	1000	1000	410
limite do quadro	854	400	1000	3000	1610
segmento logico	0	1	2	3	4
posição logica	45	100	90	1900	200
valor fisico =	89	300	90	3900	1400
Endereço Válido	VERDADEIRO	VERDADEIRO	VERDADEIRO	FALSO	VERDADEIRO

Considerando a tabela de páginas a seguir, com páginas de 500 bytes , informe os endereços físicos correspondentes aos endereços lógicos 414, 741, 1.995, 4.000 e 6.633, indicados em decimal. Obs.: Um tamanho de página de 500 bytes permite fazer os cálculos mentalmente, sem a necessidade de converter os endereços para binário e vice-versa, bastando usar divisões inteiras (com resto) entre os endereços e o tamanho de página.

página	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
quadro	3	12	6	–	9	–	2	–	0	5	–	–	–	7	–	1

Considerando a tabela de páginas a seguir, com páginas de 500 bytes , informe os endereços físicos correspondentes aos endereços lógicos 414, 741, 1.995, 4.000 e 6.633, indicados em decimal. Obs.: Um tamanho de página de 500 bytes permite fazer os cálculos mentalmente, sem a necessidade de converter os endereços para binário e vice-versa, bastando usar divisões inteiras (com resto) entre os endereços e o tamanho de página.

valor inicial	414	741	1995	4000	6633
pagina	0	1	3	9	13
quadro	3	12	---	5	7
valor final	1914	6241	---	2500	3633

Considere um sistema com endereços físicos e lógicos de 32 bits, que usa tabelas de páginas com três níveis. Cada nível de tabela de páginas usa 7 bits do endereço lógico, sendo os restantes usados para o offset. Cada entrada das tabelas de páginas ocupa 32 bits. Calcule, indicando seu raciocínio:

a. O tamanho das páginas e quadros, em bytes.

b. O tamanho máximo de memória que um processo pode ter, em bytes e páginas.

c. O espaço é ocupado pela tabela de páginas para um processo com apenas uma página de código, uma página de dados e uma página de pilha. As páginas de código e de dados se encontram no inicio do espaço de endereçamento lógico, enquanto a pilha se encontra no final do mesmo.

d. Idem, caso todas as páginas do processo estejam mapeadas na memória.

Tem 7 bits de endereço lógico para cada nível da tabela de páginas, logo $2^7 = 128$ entradas, então 128*32 bits = 4096 bits = 512 bytes. Há três níveis, então 3*512 = 1536 bytes no total da tabela.

A) São 7 bits para cada nível, então o tamanho de páginas é de 32 - 3*7 = 11 bits de offset. Logo o tamanho da página será de $2^{11} = 2048$ bytes.
Tamanho quadro = tamanho de páginas.

B) O processo tem 32 bits, onde 3*7 = 21 bits são para índice da tabela. Logo o número máximo de endereços são $2^{21} = 2097152$ páginas.
A página tem tamanho de 2048 bytes, então a memória do processo é de 2048*2097152 = 4294967296 bytes.

C) As entradas de página na tabela valem 32 bits, 1 byte = 8 bits, então cada entrada 4 bytes. No problema temos 3 páginas, código, dados e pilha, portanto 3*4 = 12 bytes de espaço.

D) Número de páginas * número de bytes no total da tabela = 2097152 * 1536 = 3221225472 bytes.

Explique o que é TLB, qual a sua finalidade e como é seu funcionamento.

TLB, em inglês "advance translate buffer", é uma memória cache especial usada em sistemas de computador para acelerar a conversão de endereços virtuais em endereços físicos. Sua finalidade é melhorar o desempenho do acesso à memória reduzindo o tempo de tradução de endereços.

Sobre as afirmações a seguir, relativas à alocação por **páginas**, indique quais são incorretas, justificando sua resposta:

a. Um endereço lógico com N bits é dividido em P bits para o número de página e N − P bits para o deslocamento em cada página.

b. As tabelas de páginas multiníveis permitem mais rapidez na conversão de endereços lógicos em físicos.

c. O bit de referência R associado a cada página ~~é "ligado"~~ é usado pela MMU para identificar quais páginas foram acessadas recentemente ~~sempre que a página é acessada~~.

d. O cache TLB é usado para manter páginas frequentemente usadas na memória.

e. O bit de modificação M associado a cada página é "ligado" pelo núcleo sempre que um processo modificar o conteúdo da mesma.

f. O cache TLB ~~deve~~ não precisa ser esvaziado a cada troca de contexto entre processos.

Por que é necessário limpar o cache TLB após cada troca de contexto entre processos? Por que isso não é necessário nas trocas de contexto entre threads?

O cache TLB deve ser liberado durante uma troca de contexto entre processos para garantir a consistência da conversão de endereço virtual.

Por outro lado, no caso de troca de contexto entre threads, essa limpeza é desnecessária porque as traduções permanecem válidas para as threads subsequentes no mesmo processo.

Um sistema de memória virtual paginada possui tabelas de página com três níveis e tempo de acesso à memória RAM de 100 ns. O sistema usa um cache TLB de 64 entradas, com taxa estimada de acerto de 98%, custo de acerto de 10 ns e penalidade de erro de 50 ns. Qual o tempo médio estimado de acesso à memória pelo processador? Apresente e explique seu raciocínio.

Começamos calculando o tempo médio de acesso a TLB, onde é necessário saber a taxa de acerto e o custo para isso mais a penalidade de erro.
tmédio = taxa acerto * custo acerto + (1 - taxa acerto) * penalidade erro
tmédio = 0,98 * 10 ns + (1 - 0,98) * 50 ns
tmédio = 9,8 ns + 1 ns
tmédio = 10,8 ns

Depois disso precisamos saber o tempo médio total, que será o de acesso a memória RAM e o tempo médio de acesso a TLB.
tmédio tot = tempo acesso RAM + tmédio acesso TLB
tmédio tot = 100 ns + 10,8 ns
tmédio tot = 110,8 ns

Considerando um sistema de 32 bits com páginas de 4 KBytes e um TLB com 64 entradas, calcule quantos erros de cache TLB são gerados pela execução de cada um dos laços a seguir. Considere Somente os acessos à matriz buffer (linhas 5 e 9), ignorando páginas de código, heap e stack. Indique seu raciocínio.

```
1  unsigned char buffer[4096][4096] ;
2
3  for (int i=0; i<4096; i++) // laço 1
4      for (int j=0; j<4096; j++)
5          buffer[i][j] = 0 ;
6
7  for (int j=0; j<4096; j++) // laço 2
8      for (int i=0; i<4096; i++)
9          buffer[i][j] = 0 ;
```

Considerando um sistema de 32 bits com páginas de 4 KBytes e um TLB com 64 entradas, calcule quantos erros de cache TLB são gerados pela execução de cada um dos laços a seguir. Considere Somente os acessos à matriz buffer (linhas 5 e 9), ignorando páginas de código, heap e stack. Indique seu raciocínio.

TLB = 64 entradas || 4096 (Ciclos) de 4096 (Acessos)

1º Caso (acesso escalonado a cada página):
4096 - 64 = 4032 (Erros por ciclo)
4032*4096 = 16.498.944 (Erros de cache TLB)

2°Caso (acesso sequencial varrendo a página):
4096 - 64 = 4032 (Erros de cache TLB)
Neste caso não são contados erros por ciclo, pois uma vez que a TLB armazenou a posição da página (prefixo), a página é varrida completamente antes de passar para a próxima consulta.

Considerando um sistema com tempo de acesso à RAM de 50 ns, tempo de acesso a disco de 5 ms, calcule quanto tempo seria necessário para efetuar os acessos à matriz do exercício anterior nos dois casos (laço 1 e laço 2). Considere que existem 256 quadros de 4.096 bytes (inicialmente vazios) para alocar a matriz e despreze os efeitos do cache TLB.

RAM = 50ns || DISCO = 5ms

1º Caso (acesso escalonado a cada página):
Acesso sequencial para leitura das posições 4096*5= 20,48uS (Por ciclo)
20,48*4096 = 83,88608ms (Tempo total para leitura de todos os dados).

2°Caso (acesso sequencial varrendo a página):
4096*5= 20,48uS
Neste caso não são contados tempos em ciclos, pois uma vez que a TLB armazenou a posição da página (prefixo), a página é varrida completamente antes de passar para a próxima consulta.

Questionário - 14

Arthur C. M. Barcella e Matheus P. Salazar