CES-33
Projeto: Gerenciador de Memória Virtual

Este projeto consiste em escrever um programa que traduza endereços lógicos para endereços físicos de um espaço de endereçamento virtual de 2^{16}=65.536 bytes. Seu programa lerá de um arquivo contendo endereços lógicos e, usando tanto um TLB quanto uma tabela de páginas, traduzirá cada endereço lógico para seu correspondente endereço físico, dando saída no valor do byte armazenado no endereço físico resultante. O objetivo deste projeto é a simulação dos passos envolvidos na tradução de endereços lógicos para físicos, bem como treinar as ideias de threads e sincronismo.

Especificidades
Seu programa lerá um arquivo contendo vários números inteiros de 32 bits que representam endereços lógicos. No entanto, você precisa se preocupar apenas com endereços de 16 bits; portanto, você deve mascarar os 16 bits da extrema direita de cada endereço lógico. Esses 16 bits são divididos em (1) um número de página de 8 bits e (2) um deslocamento de página de 8 bits.
Outras especificidades incluem:

- 2^8 entradas na tabela de páginas;
- Tamanho da página de 2^8 bytes;
- 16 entradas na TLB;
- Tamanho da moldura de 2^8 bytes;
- 256 molduras;
- Tamanho da memória física de 65.536 bytes (256 molduras x tamanho de quadro de 256 bytes).

Além disso, seu programa precisa se preocupar apenas com a leitura de endereços lógicos e sua tradução para seus endereços físicos correspondentes. Você não precisa dar suporte à gravação no espaço de endereçamento lógico.

Tradução de Endereços
Seu programa traduzirá endereços lógicos para físicos usando uma TLB e uma tabela de páginas como visto na teoria. Primeiro, o número da página é extraído do endereço lógico e a TLB é consultada. Em caso de sucesso da TLB, o número da moldura é obtido a partir da TLB. Em caso de omissão da TLB, a tabela de páginas é consultada. No último caso, o número da moldura é obtido na tabela de páginas ou ocorre um erro de página (Page Fault). A Figura 9.34 a seguir ilustra o processo. (Obs: quadro=moldura).

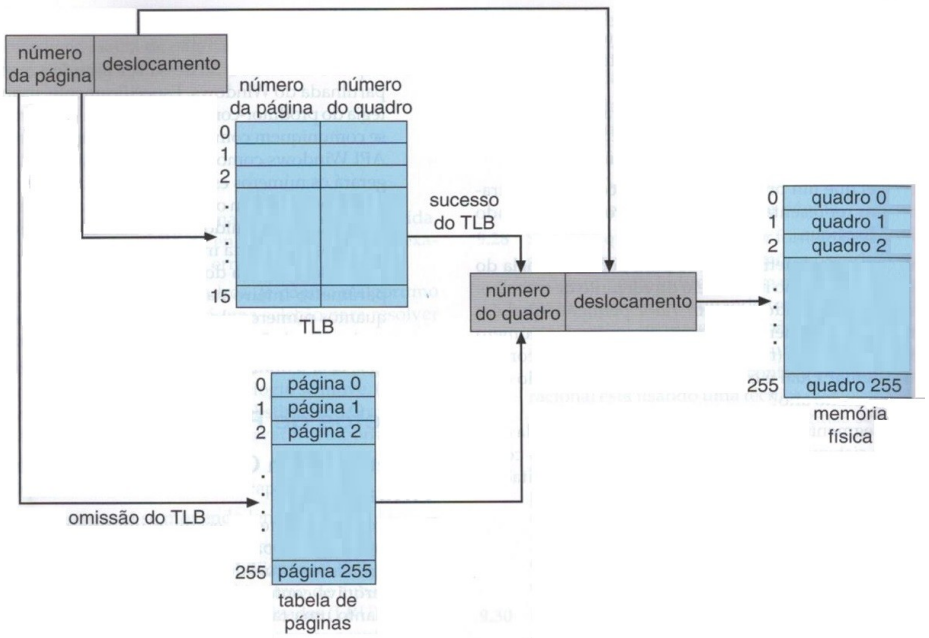

Figura 9.34 Uma representação do processo de tradução de endereços.

Manipulando erros de página
Seu programa implementará a paginação por demanda (Só traz a página para a memória quando ela é necessária). A memória de retaguarda (swap) é representada pelo arquivo BACKING_STORE.bin, um arquivo binário de 65.536 bytes. Quando ocorrer um erro de página, você lerá uma página de 256 bytes a partir do arquivo BACKING_STORE.bin e a armazenará em uma moldura disponível na memória física. Por exemplo, se um endereço lógico com o número de página 15 resultar em um erro de página, seu programa lerá a página 15 em BACKING_STORE.bin e armazenará em uma moldura na memória física. Uma vez que esse quadro seja armazenado, acessos subsequentes à página 15 serão resolvidos pela TLB ou pela tabela de páginas.
Você tratará BACKING_STORE.bin como um arquivo de acesso aleatório para que possa pesquisar aleatoriamente certas posições do arquivo para leitura. (Use as funções de C fopen(), fread(), fseek() e fclose().A princípio, o tamanho da memória física é igual ao tamanho do espaço de endereçamento virtual, logo, na primeira etapa, você não precisa se preocupar com substituições de página durante um page fault.

Arquivo de Texto
O arquivo addresses.txt contém valores inteiros representando endereços lógicos que variam de 0 a 65.535 (o tamanho do espaço de endereçamento virtual). Seu programa abrirá esse arquivo, lerá cada endereço lógico, o traduzirá para o físico e dará saída no valor do byte sinalizado contido no endereço físico.

Como começar
Primeiro, escreva um programa simples que extraia o número e o deslocamento da página a partir dos números inteiros a seguir:

1, 256, 32768, 128, 65534, 33153

Talvez a maneira mais fácil de fazer isso seja usando os operadores de mascaramento de bits e de deslocamento de bits.Uma vez que você possa estabelecer corretamente o número e o deslocamento da página a partir do número inteiro, estará pronto para começar.
Inicialmente, sugere-se que você ignore o TLB e use apenas a tabela de páginas. Você pode integrar a TLB, uma vez que sua tabela de páginas esteja funcionando apropriadamente. Lembre-se que a tradução de endereços pode funcionar sem uma TLB; a TLB apenas a torna mais rápida. Quando você estiver pronto para implementar a TLB, não se esqueça de que ela tem somente 16 entradas; assim, você terá que usar uma estratégia de substituição quando atualizar uma TLB cheia. Você pode usar uma política FIFO ou LRU para atualizar a TLB.

Como Executar Seu Programa
Seu programa deve ser executado como descrito a seguir:

./a.out addresses.txt

Seu programa lerá o arquivo addresses.txt que contém 1000 endereços lógicos variando de 0 a 65.535. Seu programa deve traduzir cada endereço lógico para um endereço físico e determinar o conteúdo do byte sinalizado armazenado no endereço físico correto. (sugere-se que use o tipo char do C que ocupa um byte de memória).
O programa deve dar saída nos seguintes valores:

1. O endereço lógico que está sendo traduzido (o valor inteiro que está sendo lido em addresses.txt);
2. O endereço físico correspondente (aquele para o qual seu programa traduziu o endereço lógico);
3. O valor do byte sinalizado armazenado no endereço físico resultante.

O arquivo correct.txt contém os valores de saída corretos para o arquivo addresses.txt. Você deve usar esse arquivo para determinar se o seu programa está traduzindo corretamente endereços lógicos para físicos.

Estatísticas
Após concluído, seu programa deve relatar as seguintes estatísticas:

1. Taxa de erros de página - o percentual de referências de endereços que resultaram em erros de página;
2. Taxa de sucesso do TLB - o percentual de referências de endereços que foram resolvidas na TLB.

Já que os endereços lógicos em addresses.txt foram gerados aleatoriamente e não refletem quaisquer localidades de acesso à memória, não espere obter uma alta taxa de sucesso de TLB.

Modificações

Até este ponto, a saída de seu programa, chamado **Versão1**, será comparada com o arquivo correct.txt.
Mais duas modificações são pedidas no projeto.
A **Versão 2** deve conter as seguintes modificações:

1. Esse projeto presume que a memória física tenha o mesmo tamanho do espaço de endereçamento virtual. Na prática, a memória física é tipicamente muito menor do que um espaço de endereçamento virtual. Use um espaço menor: 128 quadros de páginas ao invés de 256. Essa alteração demandará a modificação do seu programa para que ele controle os quadros de páginas livres, assim como a implementação de uma política de substituição de páginas usando FIFO ou LRU.
2. Apenas para treino, utilize em seu programa 2 threads: uma é disparada para pesquisar a TLB, outra para pesquisar a tabela de páginas. No caso real, usar thread para isso não é compensador, pois a TLB é muito mais rápida que a RAM e as tarefas não aconteceriam de fato simultaneamente. Porém, como este é apenas um simulador, para efeito de treino de programação de threads, vale a pena.

No dia 24/Maio haverá uma apresentação preliminar e no dia 07/Jun a apresentação final com entrega de relatório.
O projeto valerá 11 pontos para quem implementar FIFO e LRU.