4.3 - Memória Virtual

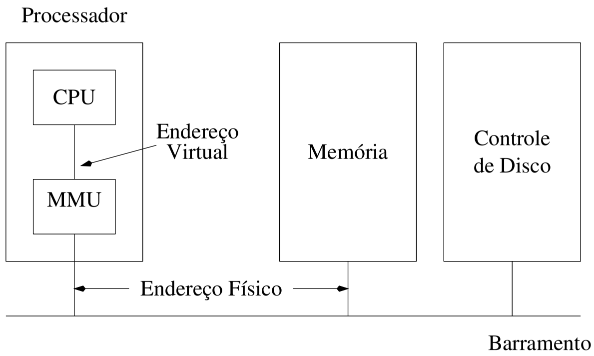
- desde os primórdios da informática, o tamanho dos programas vem superando a quantidade de memória disponível para abrigá-los;
- ... a solução usualmente adotada era dividir o programa em partes, chamados *overlays* e após a execução de cada *overlay* o próximo é executado;
- ... alguns sistemas com *overlay* eram relativamente complexos, permitindo múltiplos *overlays* na memória por vez, embora mantidos em disco e permutados para a memória principal pelo sistema operacional.
- **problema** - o particionamento do programa era deixada a cargo do programador.
- **proposta** - deixar o particionamento do programa a cargo do sistema operacional, ou seja, a combinação do tamanho do programa, dados e pilha, pode exceder a quantia de memória física disponível = **memória virtual**.

... 4.3 - Memória Virtual

- ... o sistema operacional mantém aquelas partes do programa correntemente em uso na memória principal e o resto no disco;
- ... por exemplo, um programa de tamanho 10M bytes pode ser executado em uma máquina com 1M bytes de memória escolhendo-se cuidadosamente quais dos 1M será mantido na memória a cada instante.
- memória virtual pode também ser utilizada em um sistema multiprogramado, por exemplo, 08 programas de 1 MB podem ser alocados em partições de 256 KB numa memória de 2 MB, com cada programa operando sobre 256 KB.
- ... de fato, memória virtual e multiprogramação estão intimamente relacionadas;
- ... enquanto um programa está esperando que parte de seu espaço de endereçamento seja trazido à memória, o programa é bloqueado até que a operação de E/S seja completada, e neste intervalo a CPU pode ser direcionada a outro processo.

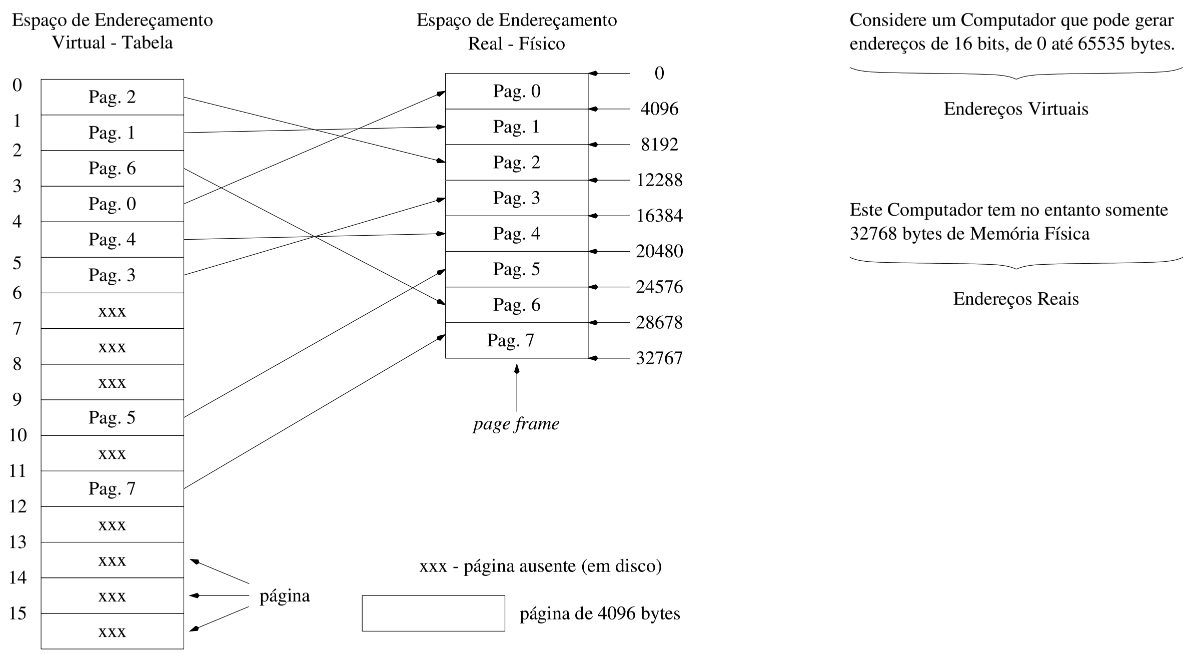
4.3.1 - Paginação

- ... endereços gerados pelos programas são chamados *endereços virtuais* e formam o *espaço virtual de endereçamento* do processo;
- ... em computadores sem memória virtual, o endereço virtual representa o próprio endereço físico e, portanto, não se faz necessário o mapeamento;
- ... quando memória virtual é usada, os endereços de memória não vão diretamente para o barramento de memória, ao invés disso, eles vão à unidade de gerenciamento de memória denominada MMU ou *Memory Management Unit*;



... 4.3.1 - Paginação

- vejamos como esse mapeamento funciona com o auxílio da figura abaixo:



... 4.3.1 - Paginação

✯ O que acontece se o programa tentar usar uma página não mapeada ?

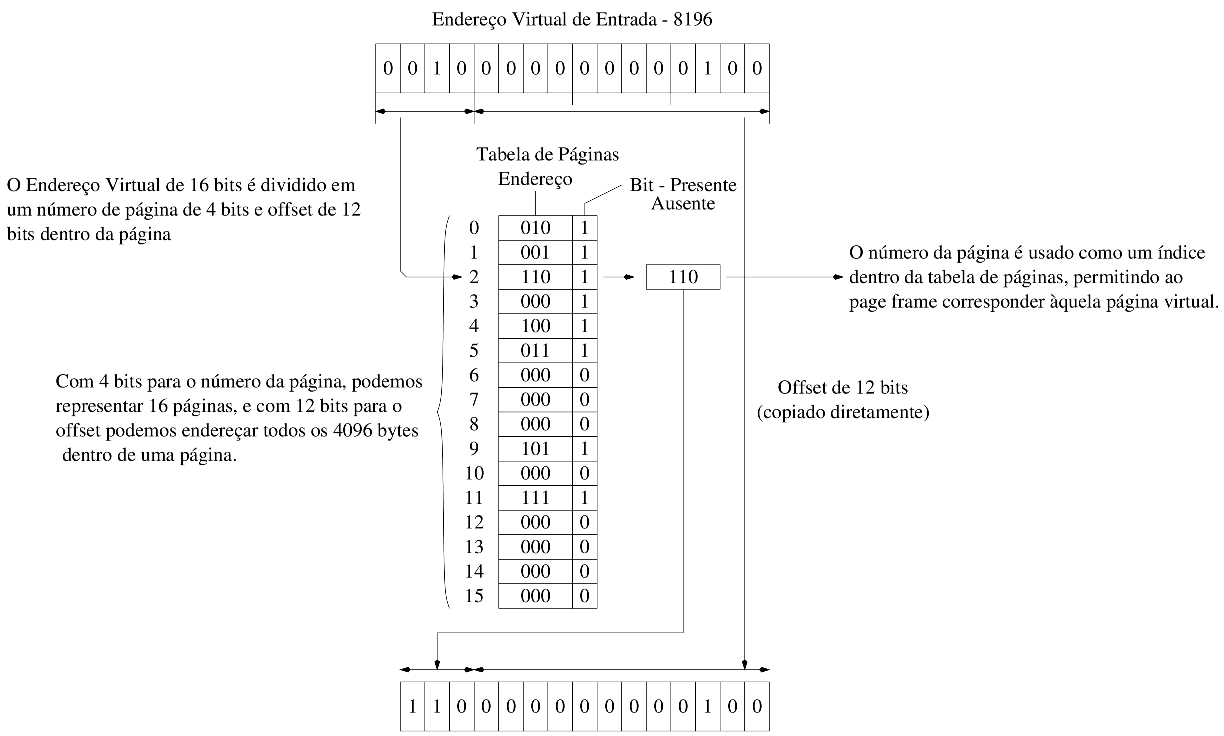
- ... como exemplo, seja uma instrução que referencia o byte número 12 dentro da página virtual 8 (começando em 32768) — a MMU observa que a página está sem mapeamento e força a CPU a causar um *trap* no sistema operacional;

- ... o sistema operacional remove o *page frame* menos usado e escreve seu conteúdo de volta no disco. Ele então busca a página referenciada para o *page frame* liberado, atualiza o mapa, e retorna à instrução interrompida.

✯ Agora, vejamos como o gerenciador de memória trabalha e porque temos de escolher um tamanho de página como uma potência de 2.

- ... para isto considere o endereço virtual 8196 (0010 0000 0000 0100) sendo mapeado usando o mapa ou tabela de páginas do gerenciador de memória.

... 4.3.1 - Paginação



4.3.2 - Segmentação

- Paginação provê uma técnica para implementação de um grande espaço endereçável numa memória física limitada, no entanto, para algumas aplicações, um espaço endereçável bi-dimensional é mais conveniente.

- Idealmente, cada programa deve ter um número muito grande de segmentos (i.e. $2^{32} = 4G$), cada um consistindo de grande númeto de bytes (i.e. também 2^{32});

- ... os primeiros 64K de segmentos podem ser revervados para procedimentos, dados, e pilhas, pertencentes ao programa em execução;

- ... os segmentos restantes podem conter um arquivo por segmento, tal que processos possam diretamente endereçar todos seus arquivos, sem ter que abrí-los e usar primitivas especiais de entrada/saída para lê-los e escrevê-los;

- ... cada arquivo cresce ou reduz-se completamente, independentemente dos outros nesta disposição, com cada byte da memória sendo endereçado por um par formado pelo segmento e offset.

... 4.3.2 - Segmentação

- Projeto MULTICS introduziu a idéia de memória segmentada que sobrevive até hoje.

- ... uma implementação típica provê suporte de *hardware* para até 16 processos, cada um com espaço de endereçamento virtual de 1024 páginas de 2K ou 4K;

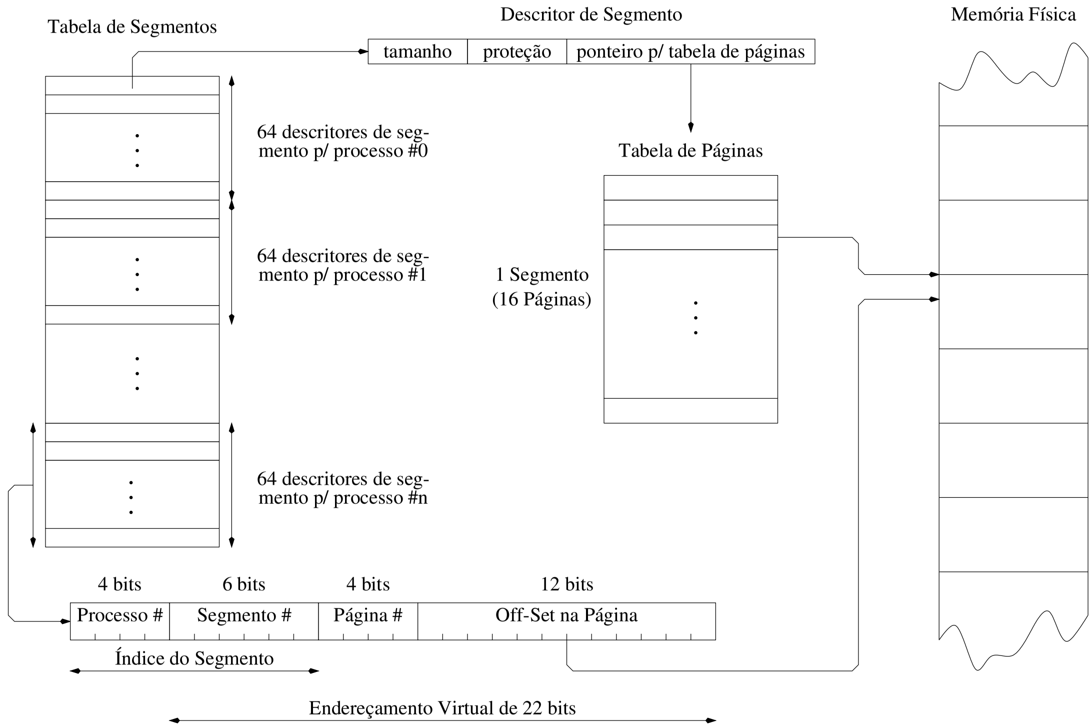
- ... se introduzirmos páginas de 4K para este exemplo, cada processo terá um espaço de endereçamento virtual de 4M, consistindo de 1024 páginas de 4K cada.

- Este esquema poderia ser implementado dando a cada processo sua própria tabela com 1024 números de *page frames*, no entanto, esta técnica raramente é empregada;

- ... em vez disto, o *hardware* da MMU contém uma tabela com 16 seções, uma para cada um dos 16 processos e cada uma com 64 descritores de segmento;

- ... então o espaço de endereçamento para cada processo de 4M é dividido em 64 segmentos, cada um contendo 16 páginas de 4K.

... 4.3.2 - Segmentação



... 4.3.2 - Segmentação

- ... cada um dos descritores de segmento contém um tamanho de segmento (0 à 16 páginas), bits de proteção dizendo se um segmento pode ser lido ou escrito, e um ponteiro para a própria tabela de páginas;

- ... cada uma das tabelas de páginas contêm 16 entradas, cada entrada apontando para uma *page frame* na memória (guardando o número da *page frame*);

- ... quando o sistema operacional começa um processo, ele carrega um número de 4 bits do processo num registrador especial do *hardware*;

✯ ... quando que um referência a memória é feita, o gerenciador de memória traduz o endereço virtual como se segue:

❶ toma-se o número de 4 bits do processo e os 6 bits de mais alta ordem dos 22 que compõem o endereço virtual (necessários para o endereço de 4M);

... 4.3.2 - Segmentação

- ... ao combiná-los, gera-se um único número 10 bits usado para indexar a tabela de segmentos e localizar o descritor de segmento relevante — note que existem exatos $2^{10} = 1024$ segmentos na tabela: 64 segmentos/processo X 16 processos;

❷ verifica-se os bits de proteção para ver se o acesso é permitido e, em sendo, a MMU compara o número da página extraída do endereço virtual com o tamanho do segmento para verificar se o segmento é grande o bastante;

- ... caso seja, o número da página é usado como um índice para a tabela de páginas, cujo endereço é fornecido no descritor de segmento (todas tabelas de páginas são mantidas numa memória rápida especial dentro da MMU);

- ... uma vez que o número da estrutura de página é encontrado, ele é combinado com o deslocamento do endereço virtual para formar o endereço físico da memória, o qual é então posto no barramento de endereço.

... 4.3.2 - Segmentação

✔ **característica chave** - quando o sistema operacional escolhe um processo, tudo o que ele tem de fazer é mudar o registrador do número de 4 bits do processo e, assim, não precisa recarregar todas as tabelas de segmento ou de página;

- ... 02 ou mais processos podem dividir um segmento se seus descritores de segmentos apontarem para a mesma tabela de páginas, assim, qualquer mudança feita numa página deste segmento é automaticamente visível para os outros processos.

✔ **expansão do espaço de endereçamento** - para espaços de endereço mais largos paga-se o preço de uma tabela maior dentro da MMU;

- ... para o exemplo apresentado, se tivéssemos quatro vezes mais memória para as tabelas de páginas dentro da MMU e seus conteúdos com 4 processos em vez de 16, o MMU poderiamos suportar 64 segmentos de 1 MB por processo.