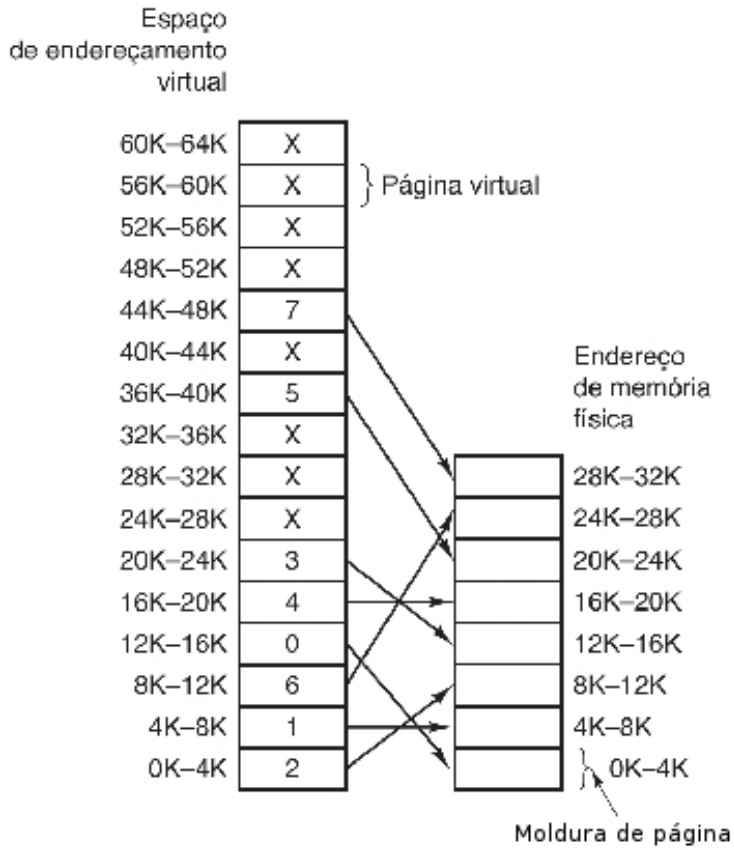
Aula 14 – Paginação

Norton Trevisan Roman
Clodoaldo Aparecido de Moraes Lima

13 de outubro de 2014

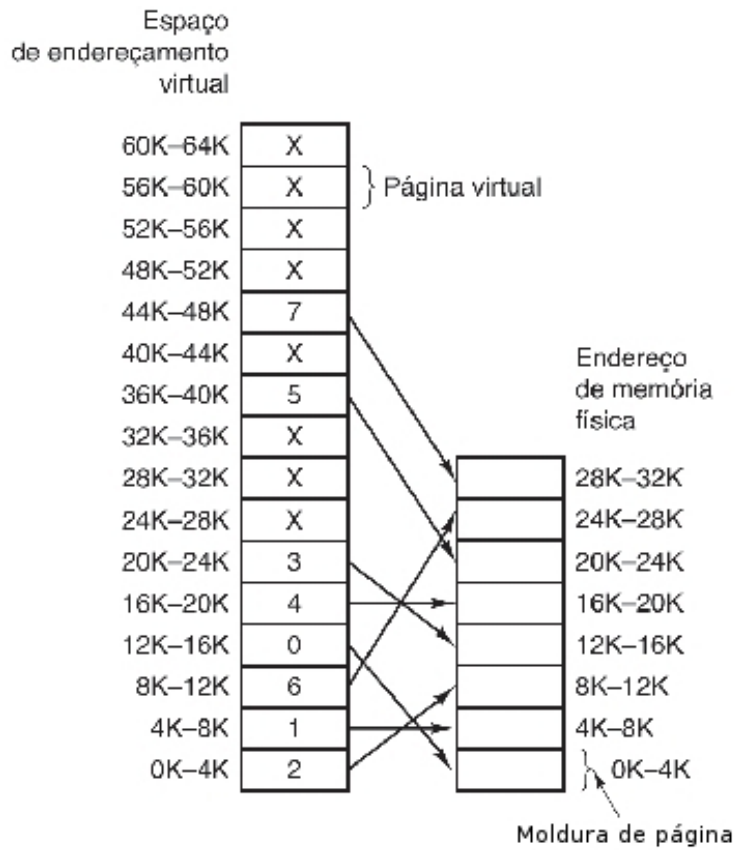
Memória Virtual – Paginação

- Ex:
  - Computador que gera até 64K de endereços virtuais (16 bits)
    - Tem apenas 32KB de memória física
    - Programas de 64KB podem ser escritos, mas não carregados inteiramente na memória
    - Uma cópia de 64KB deve estar no disco, para que partes possam ser carregadas à memória



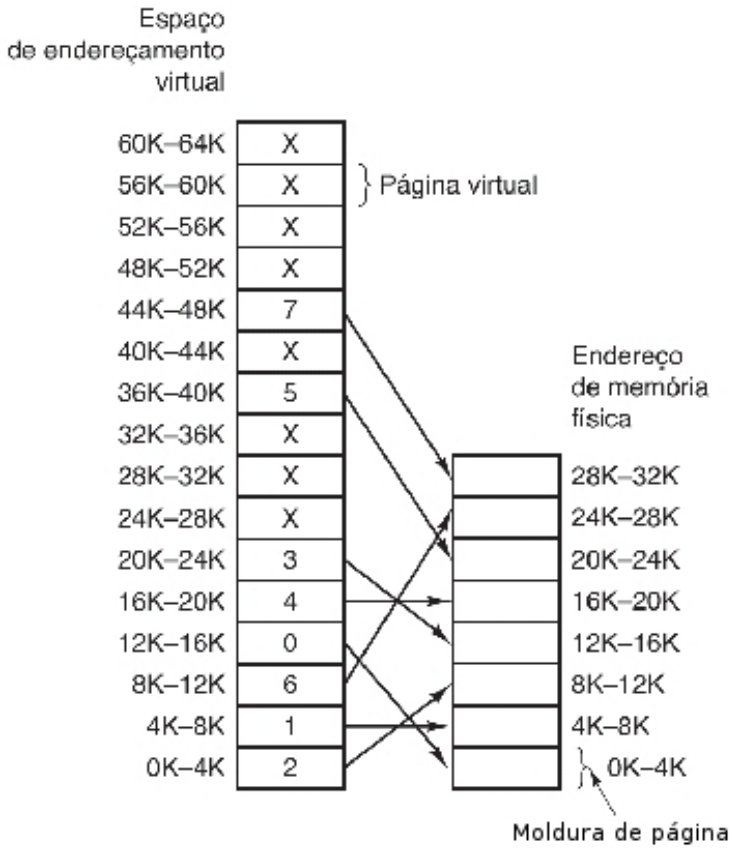
Memória Virtual – Paginação

- Divide-se o espaço de endereçamento virtual em unidades de tamanho fixo – as páginas
  - Nesse caso, Páginas de 4KB
    - 4096 bytes/endereços (0-4095)
  - As unidades correspondentes na memória física são as page frames
  - Páginas e molduras de página são em geral do mesmo tamanho
    - Alternativamente, o SO pode mapear 2 ou mais molduras a uma única página



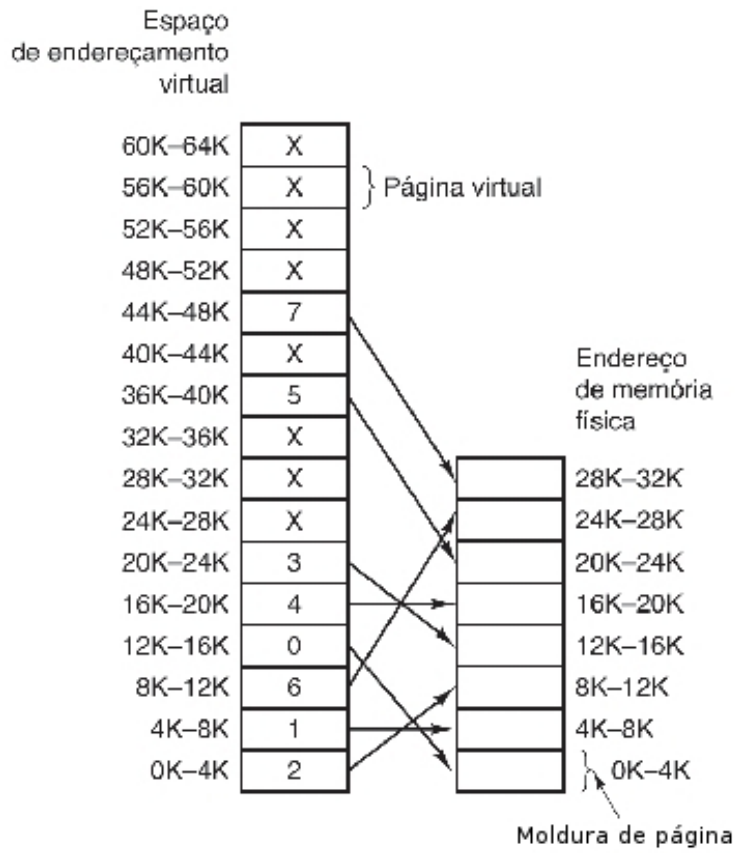
Memória Virtual – Paginação

- Embora tenha 32KB, o sistema age como se tivesse 64KB
- Ex: MOV REG,5
  - A MMU identifica que é a primeira página (5B acima da sua base $\rightarrow$ 0)
  - Ela está mapeada à terceira frame, que começa em 8k = 8192
    - O endereço enviado ao barramento é 5 + 8192 = 8197



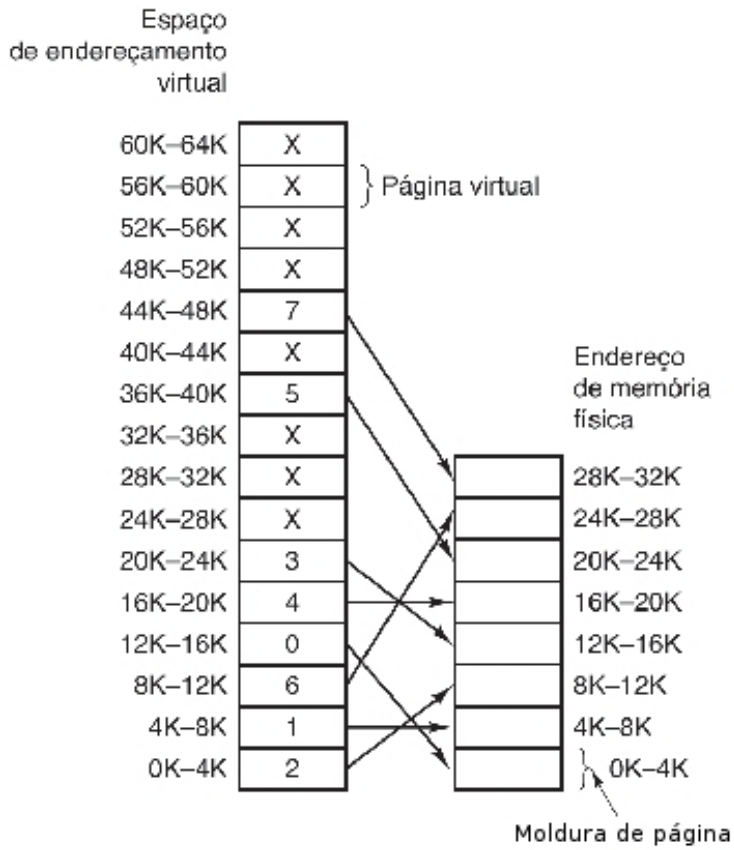
Memória Virtual – Paginação

- Como sabemos que páginas estão na memória efetivamente?
  - Se temos apenas 8 frames, somente 8 páginas (das 16) estão mapeadas
  - Solução:
    - Bit de presente/ausente em cada entrada da tabela de páginas
    - Identifica que páginas estão fisicamente presentes na memória



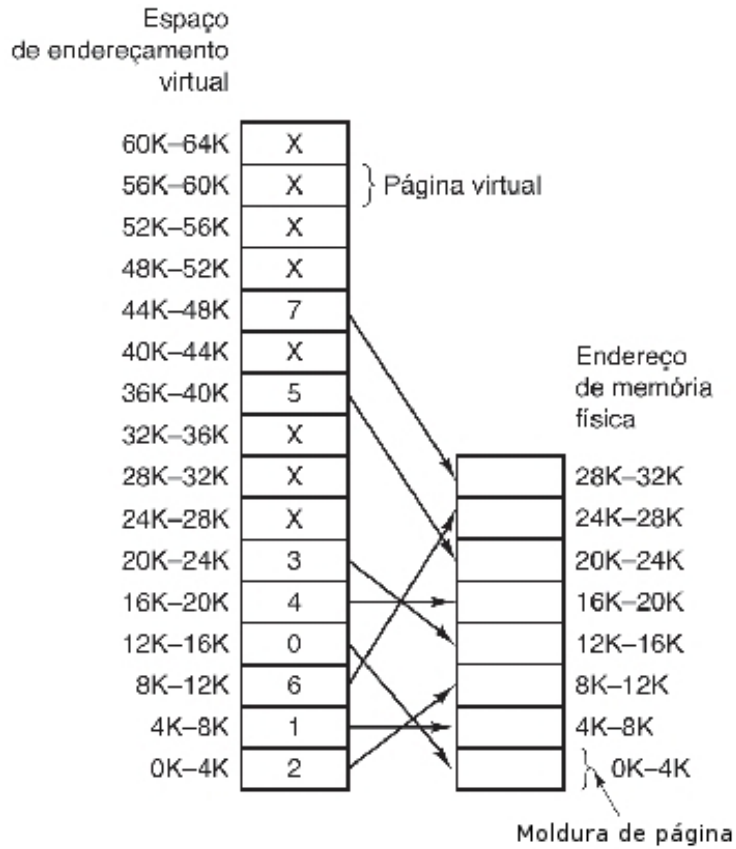
Memória Virtual – Paginação

- E se um programa referenciar um endereço não mapeado?
  - Ex: MOV REG, 32780
    - Byte 12 da página 8
  - A MMU verifica que a página não está mapeada
    - Através do bit na tabela
    - Força o desvio da CPU para o S.O. (trap) – page fault



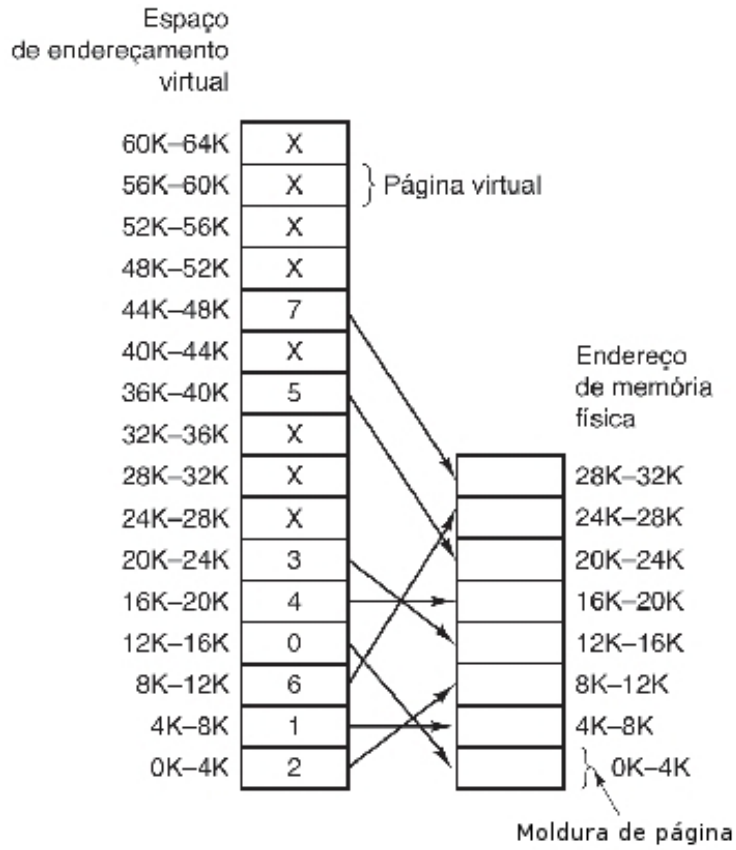
Memória Virtual – Paginação

- O S.O. toma uma moldura pouco usada e escreve seu conteúdo no disco
- Carrega a página recém referenciada na moldura (frame) recém liberada
  - Atualiza o mapeamento na tabela de páginas
  - Reinicia a instrução aprisionada na trap (que causou a trap)



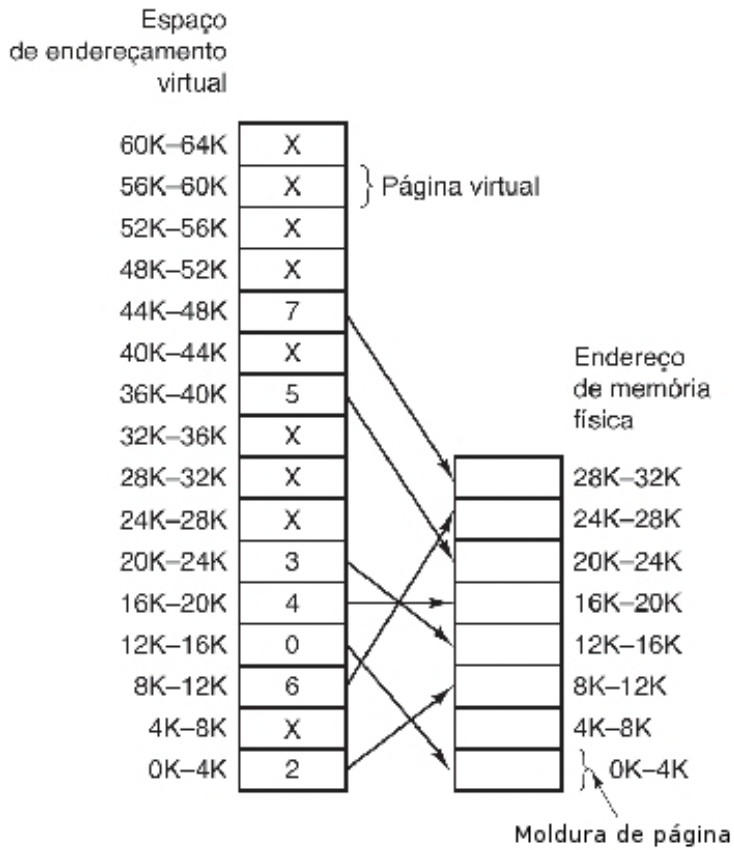
Memória Virtual – Paginação

- Ex: MOV REG, 32780 (página 8)
  - Se o SO decidir escolher a moldura 1, deverá carregar a página 8 a partir do endereço físico 4096
  - Na MMU, marca a entrada, na tabela de páginas, da página 1 como "não mapeada"



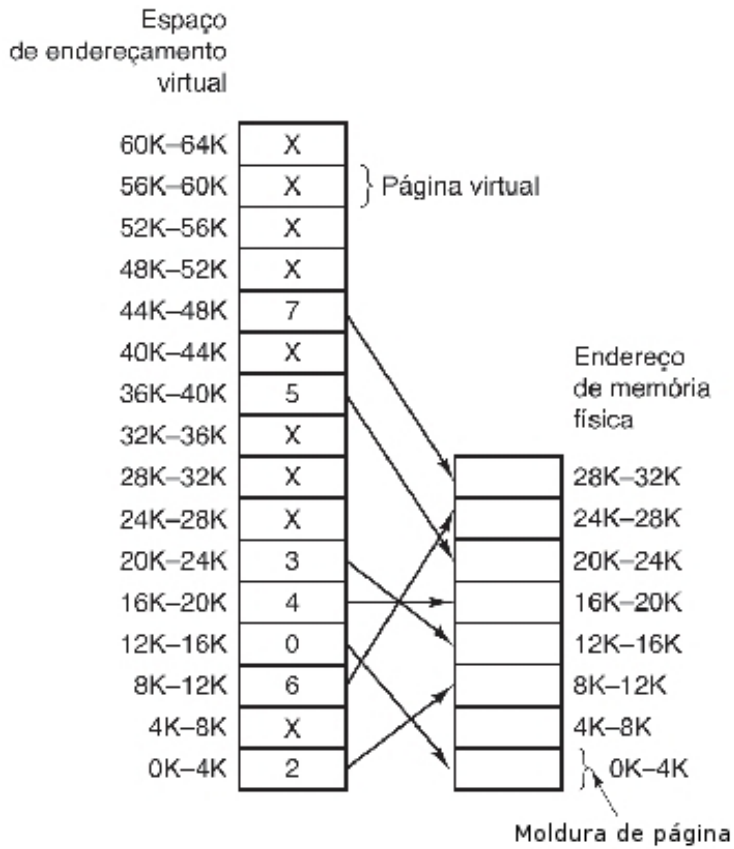
Memória Virtual – Paginação

- Ex: MOV REG, 32780 (página 8)
  - Se o SO decidir escolher a moldura 1, deverá carregar a página 8 a partir do endereço físico 4096
  - Na MMU, marca a entrada, na tabela de páginas, da página 1 como "não mapeada"



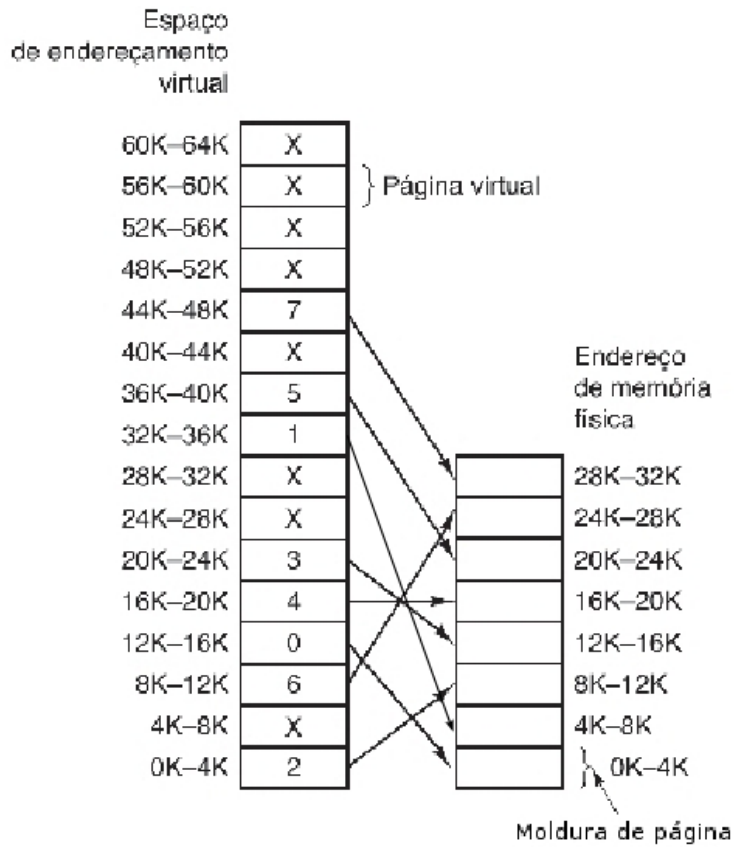
Memória Virtual – Paginação

- Ex: MOV REG, 32780 (página 8)
  - Em seguida marca, na tabela de páginas, a entrada da página 8 como "mapeada"



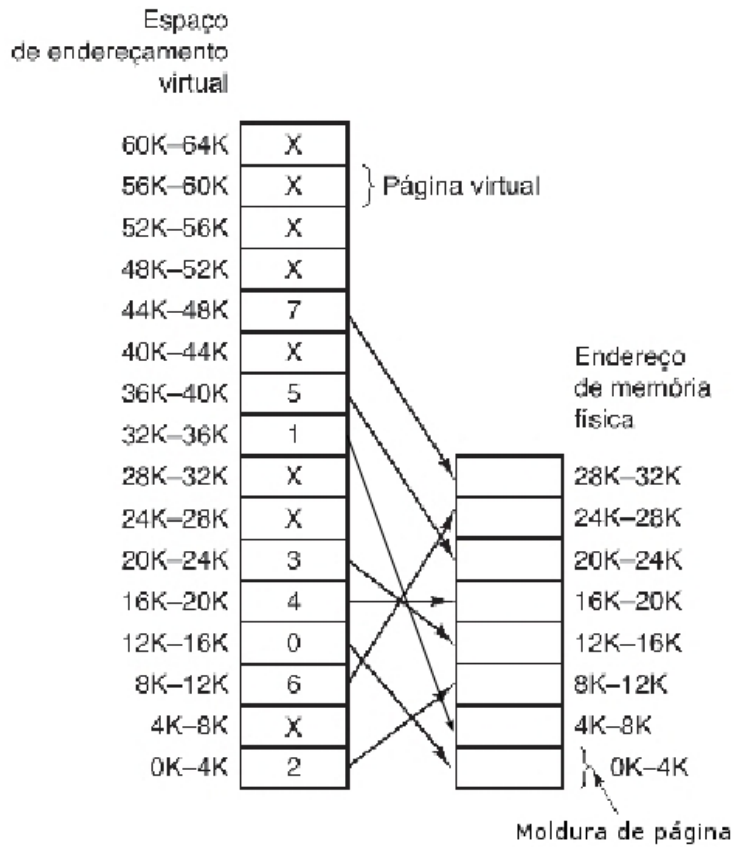
Memória Virtual – Paginação

- Ex: MOV REG, 32780 (página 8)
  - Em seguida marca, na tabela de páginas, a entrada da página 8 como "mapeada"



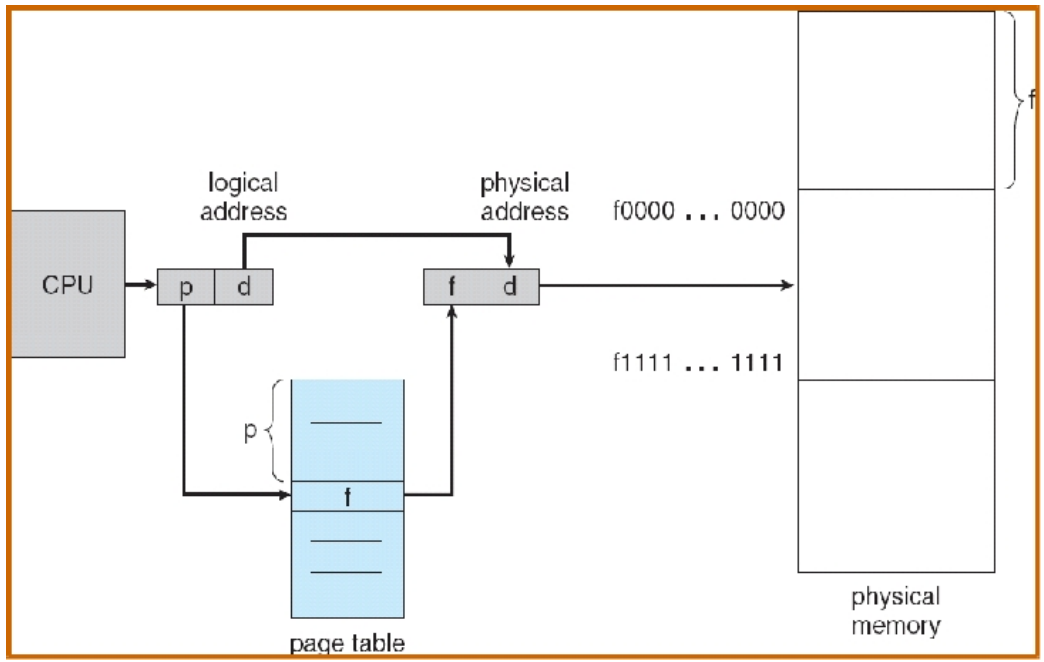
Memória Virtual – Paginação

- Ex: MOV REG, 32780 (página 8)
  - Em seguida marca, na tabela de páginas, a entrada da página 8 como "mapeada"
  - Quando a instrução que causou a interrupção for reexecutada, a MMU transformará o endereço virtual 32780 no físico 4108 (4096 + 12)



Memória Virtual – Tabela de Páginas

- Uma vez que o mapeamento é feito pela MMU, como a MMU busca um endereço?
  - Ideal: usar parte do endereço virtual como índice na tabela, onde está o endereço-base da moldura correspondente na memória



Memória Virtual – Tabela de Páginas

- E como fazer isso?
  - Use tamanhos de página que sejam potências de 2
  - $4K = 4096 = 2^{12}$
    - 0k a 4095B $\rightarrow$ 000000000000000 a 000111111111111
    - 4k a 8191B $\rightarrow$ 001000000000000 a 001111111111111
    - 8k a 12287B $\rightarrow$ 010000000000000 a 010111111111111
    - 12k a 16383B $\rightarrow$ 011000000000000 a 011111111111111
    - ...
  - Os bits mais significativos (vermelho) são o número da página

Memória Virtual – Tabela de Páginas

- E endereços dentro da página?
  - Use os bits além do limite da página (em preto, no exemplo anterior)
  - Ex: 8196
    - Múltiplo de 4k mais próximo $\rightarrow 2 = 8192$
      001000000000000
    - $$\begin{array}{lll} & 001000000000100 & (8196) \\ \hline & 001000000000000 & (8192 - \text{base da página}) \\ + & 000000000000100 & (4 - \text{deslocamento}) \end{array}$$
    - Apenas copiou-se bits – nenhuma operação aritmética envolvida

Memória Virtual – Tabela de Páginas

- Adicionar o deslocamento não muda os bits da base da página
  - Se elas forem potência de 2
  - Ex: Páginas de 4KB
    - Permitem endereços internos à página de 0 a 4095

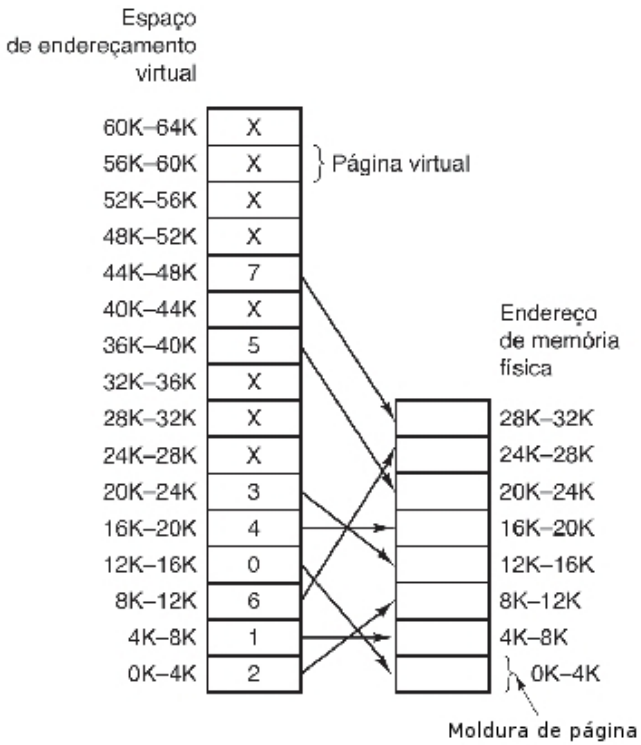
	000111111111110	(4094)
+	000000000000001	(1)
	000111111111111	(4095)
+	000000000000001	(1)
	001000000000000	(4096) – mudou de página

Memória Virtual – Tabela de Páginas

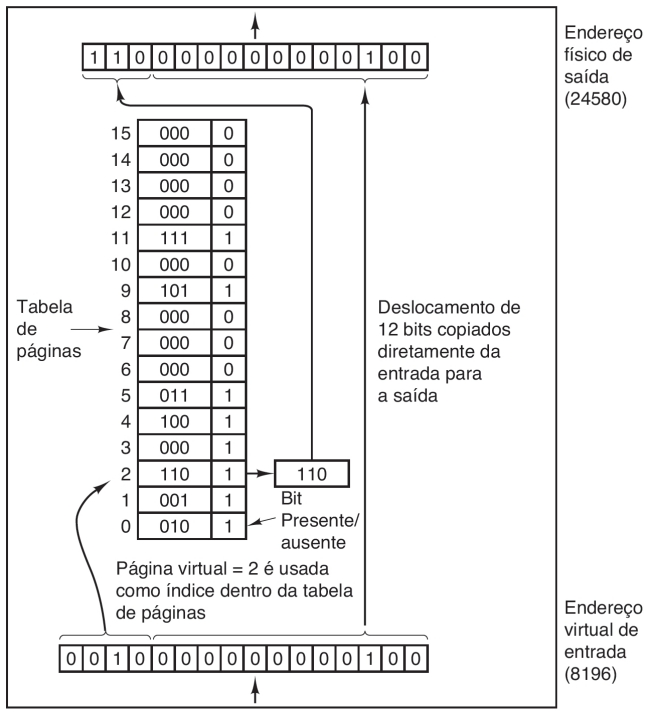
- Parte em vermelho:
  - Endereço-base da página (montado zerando-se a parte em preto)
  - Varia, na hora de mapear páginas a suas respectivas molduras
- Parte em preto:
  - Deslocamento (offset)
  - Não varia $\rightarrow$ um endereço que estava n bytes acima da base da página estará os mesmos n bytes acima da base da moldura

Memória Virtual – Tabela de Páginas

- Ex: Suponha uma MMU que implemente o seguinte mapeamento de páginas:



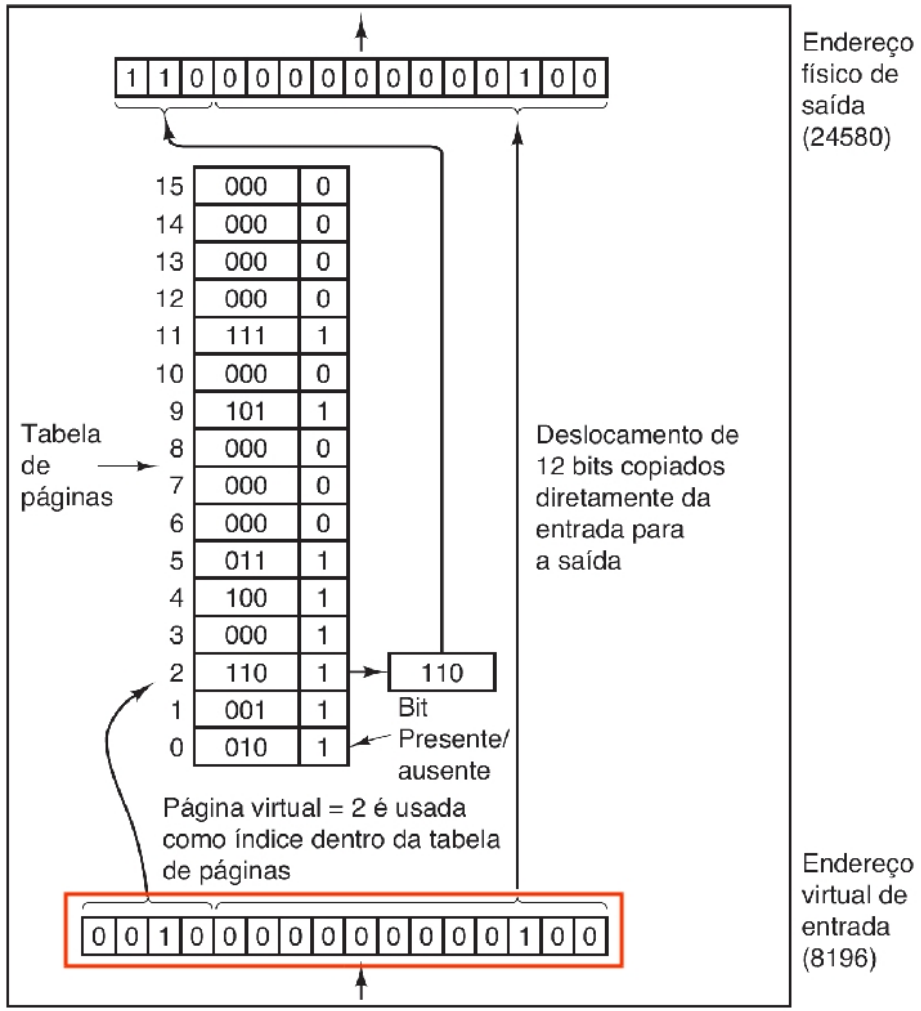
Mapeamento



MMU

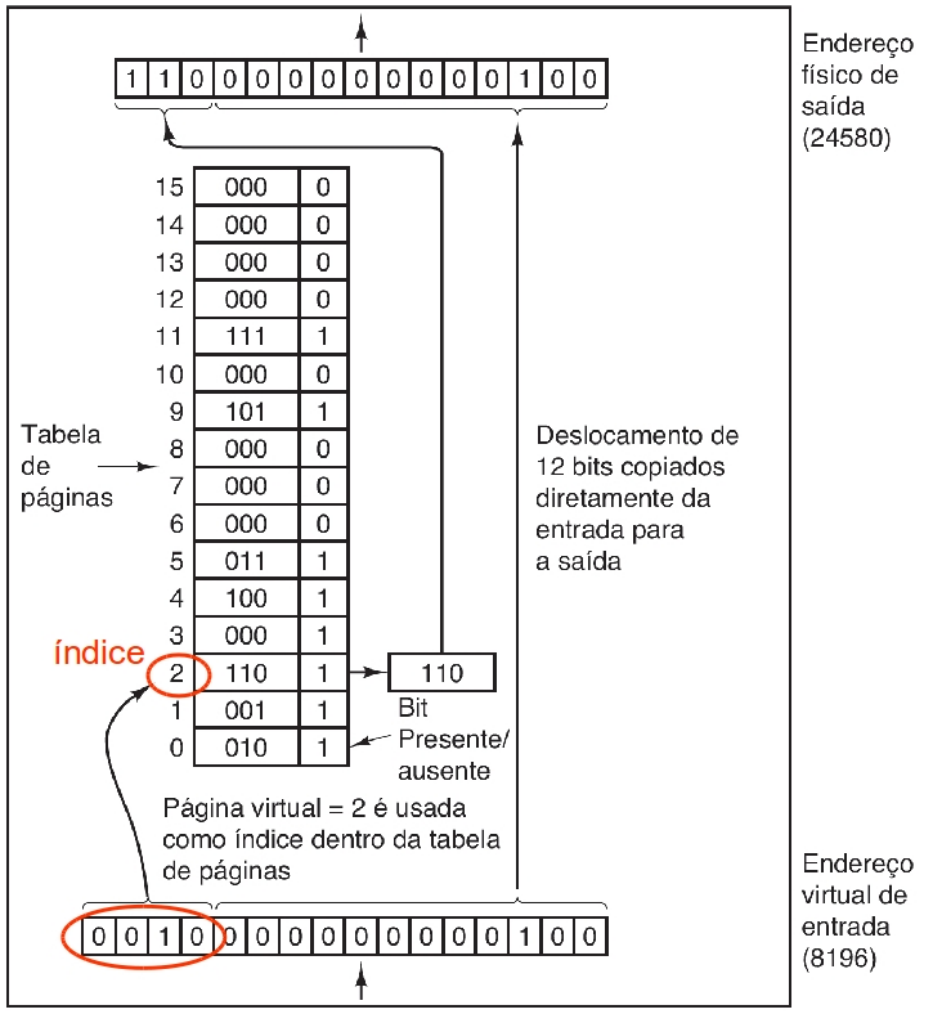
Memória Virtual – Tabela de Páginas

- Ex: Endereço 8196
  - 0010000000000100
  - MMU com 16 páginas de 4KB
    - Endereço virtual de 16 bits (65536 B)
    - A tabela tem 16 entradas (0000 a 1111)
  - Hardware com 8 frames (de 4KB)
    - Endereço físico de 15 bits



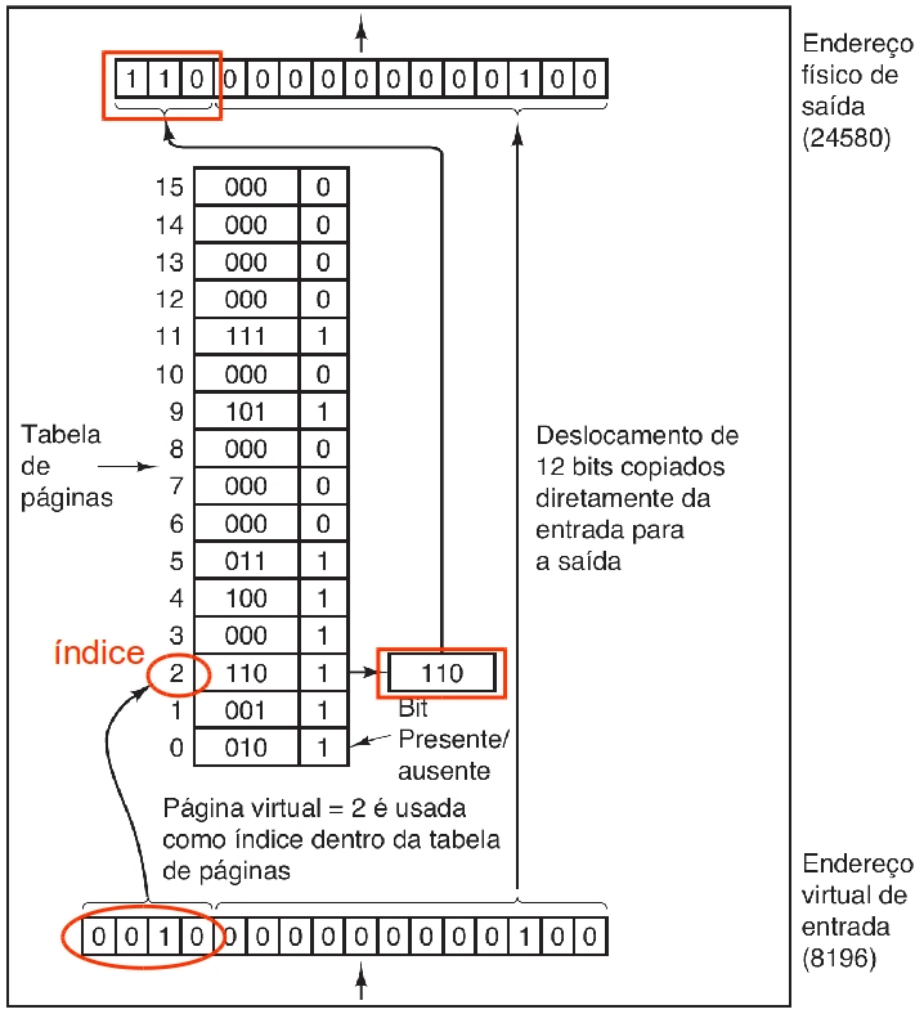
Memória Virtual – Tabela de Páginas

- Usamos os 4 bits mais altos do endereço virtual como índice na tabela (a base da página)
  - É o número da página
  - Se página estiver na RAM
    - (Bit presente/ ausente = 1)
    - O endereço físico pode ser montado



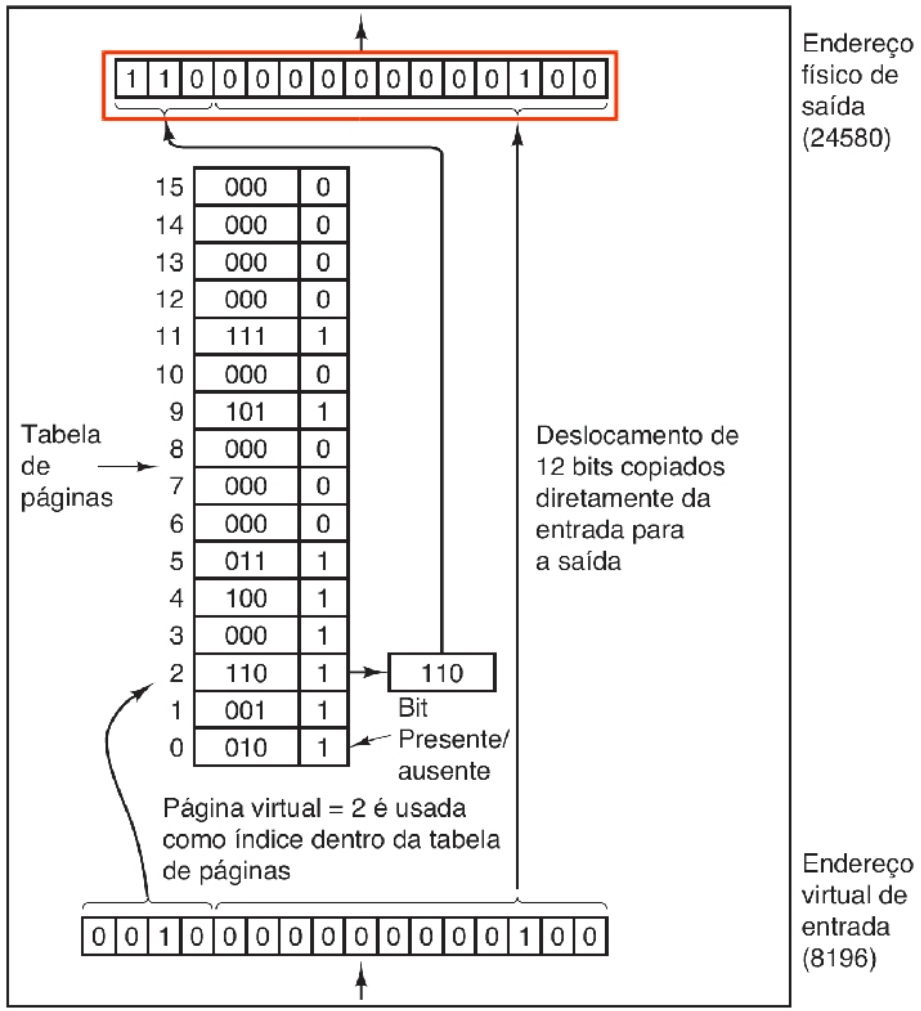
Memória Virtual – Tabela de Páginas

- Montando o endereço físico
  - O nº da moldura de página (110) é copiado para os três bits mais significativos do endereço de saída (real, de 15 bits), juntamente com o deslocamento (sem alteração)



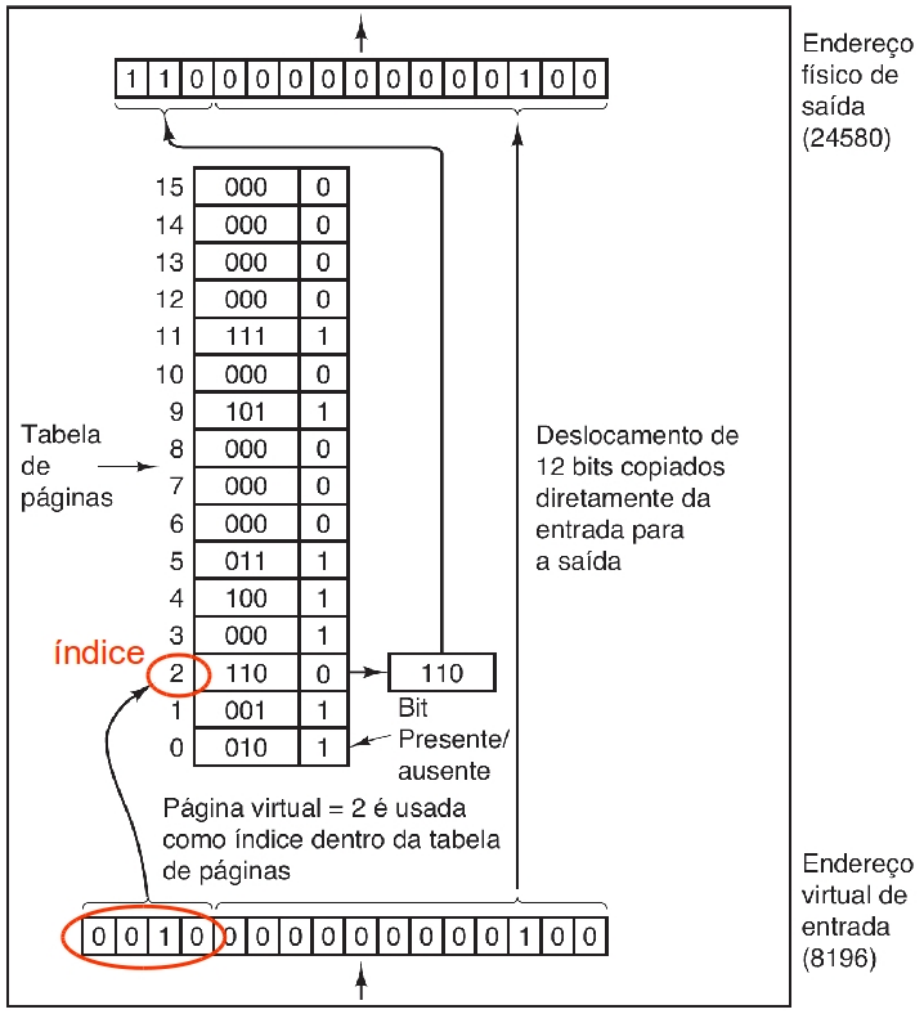
Memória Virtual – Tabela de Páginas

- Montando o endereço físico
  - O registrador de saída envia então esse endereço à memória, via barramento



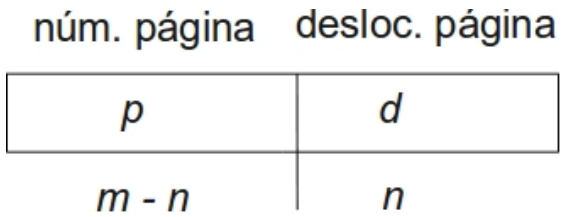
Memória Virtual – Tabela de Páginas

- Se página não estiver na RAM
  - (Bit presente/ausente = 0)
  - Executa uma trap (page fault) → Desvia ao S.O.



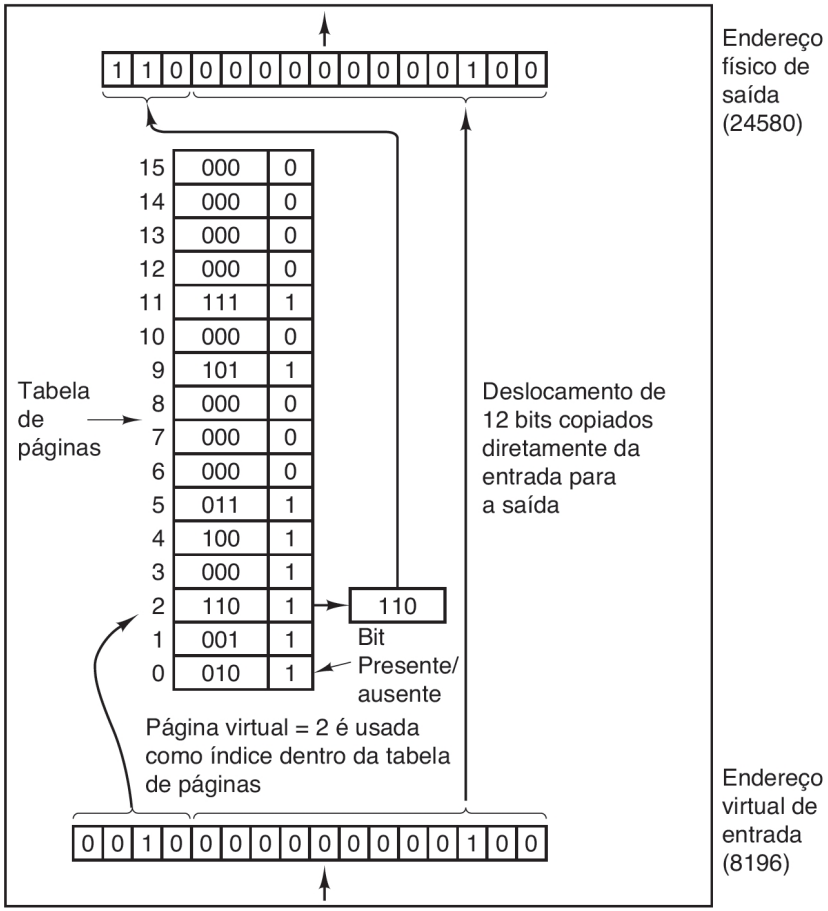
Memória Virtual – Tabela de Páginas

- O endereço virtual é então dividido em:
  - Número de página (p) – usado como índice para a tabela de páginas, que contém o endereço da base da moldura correspondente na memória física
  - Deslocamento de página (d) – combinado com o endereço da base para definir o endereço de memória física, que é enviado ao barramento de memória



  - Para determinado espaço de endereço lógico 2^m e tamanho de página 2^n

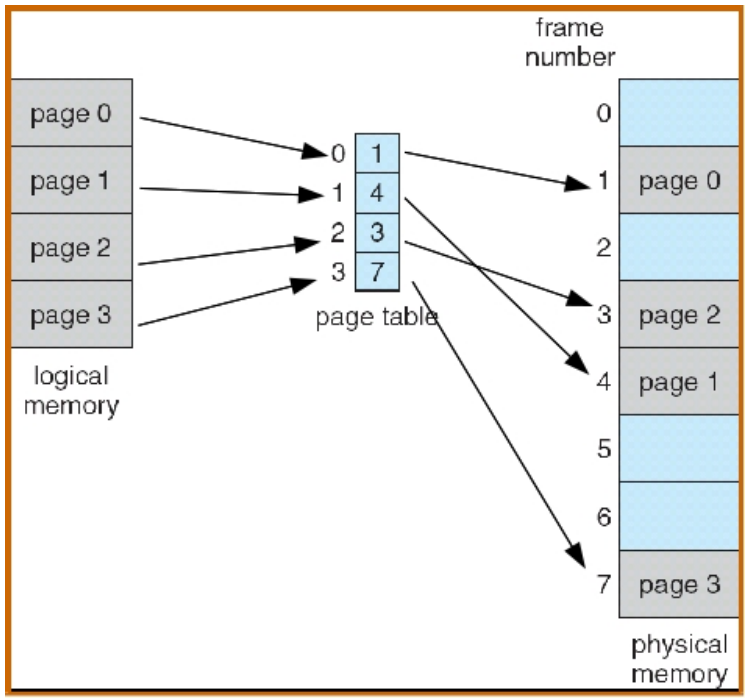
Memória Virtual – Tabela de Páginas



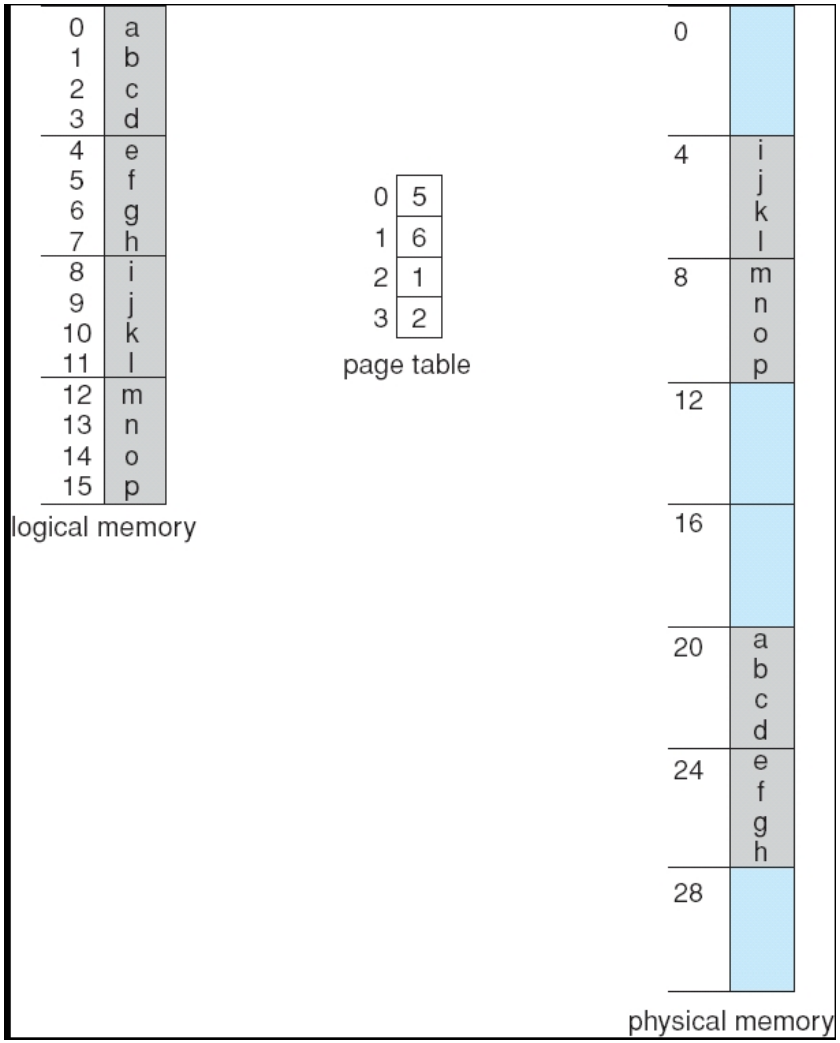
- O deslocamento é então o quanto acima da base da página o endereço está
  - Ex: Endereço 8296
    - 10000001101000
  - $8296 = 8192 + 104$:

$$\begin{array}{rr} & 010000000000000 \\ + & 000000001101000 \\ \hline & 010000001101000 \end{array}$$

Memória Virtual – Tabela de Páginas



Note que, embora o mapeamento seja diretamente entre páginas e molduras, todo o conteúdo dentro delas é indiretamente mapeado também



Paginação – Em Suma

- Divida a memória física em blocos de tamanho fixo, denominados molduras
- Divida a memória lógica em blocos do mesmo tamanho, denominados páginas
- Acompanhe todas as molduras livres
- Configure uma tabela de páginas para traduzir de endereços lógicos para físicos

Memória Virtual – Paginação

- Problemas:
  - Fragmentação interna – sobra espaço na página
  - Definição do tamanho das páginas;
    - Geralmente a MMU que define e não o SO
    - Transferências com o disco são feitas geralmente uma página por vez
    - Páginas maiores: leitura mais eficiente (do disco), tabela menor, mas maior fragmentação interna
    - Páginas menores: leitura menos eficiente, tabela maior, mas menor fragmentação interna
  - Problema:
    - Geralmente queremos páginas enormes com acesso rápido

Memória Virtual – Tabela de Páginas

- Espaço virtual $\times$ tamanho de página

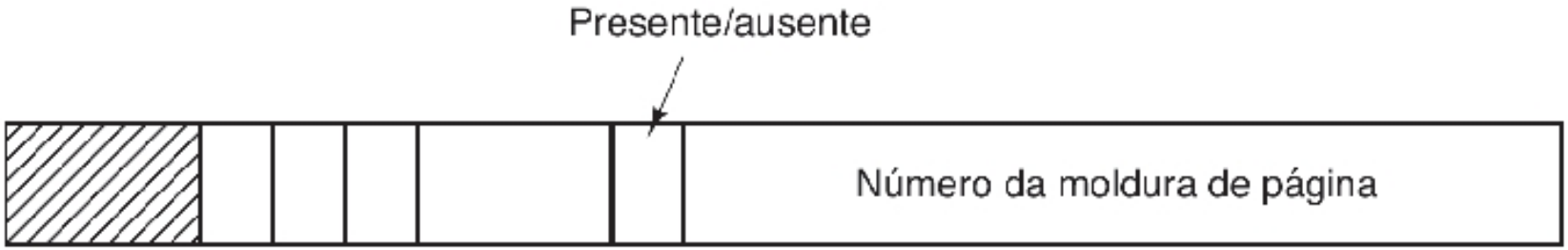
Espaço de Endereçamento Virtual	Tamanho da página	Número de páginas	Número de entradas nas tabela de páginas
2^{32} endereços	512 bytes	2^{23}	2^{23}
2^{32} endereços	4 kbytes	2^{20}	2^{20}
2^{64} endereços	4 kbytes	2^{52}	2^{52}
2^{64} endereços	64 kbytes	2^{48}	2^{48}

Entrada na Tabela de Páginas

- Depende muito do hardware
- Em geral, 32 bits, divididos da seguinte maneira:
  - Page frame number:
    - Identifica a moldura correspondente à página
    - Campo mais importante

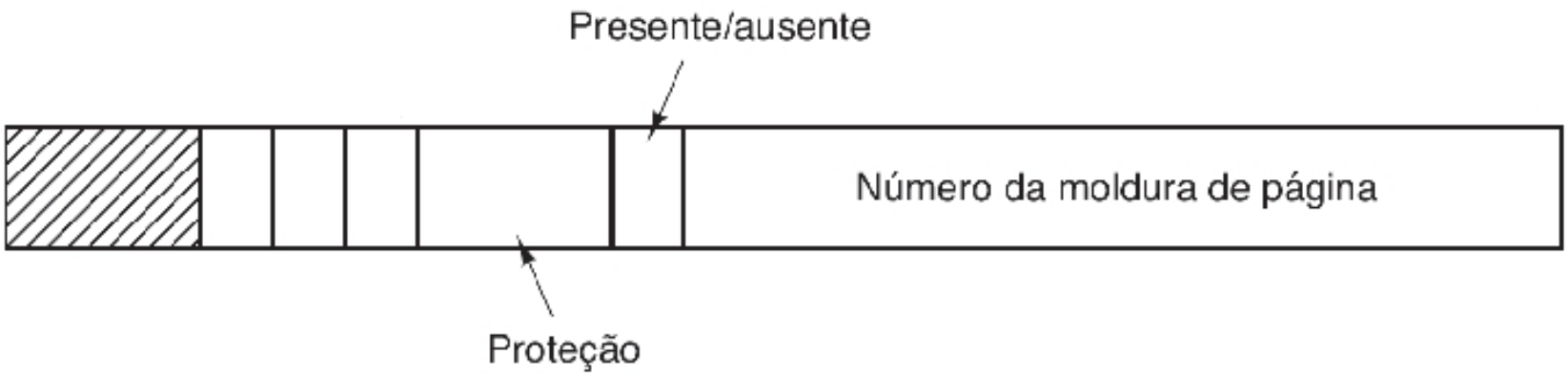
Entrada na Tabela de Páginas

- Em geral, 32 bits, divididos da seguinte maneira:
  - Bit de Residência (Presente/ausente):
    - Se valor igual 1, então entrada válida para uso
    - Se valor igual 0, então entrada inválida, pois a página virtual correspondente não está na memória (acessá-la causará page fault)



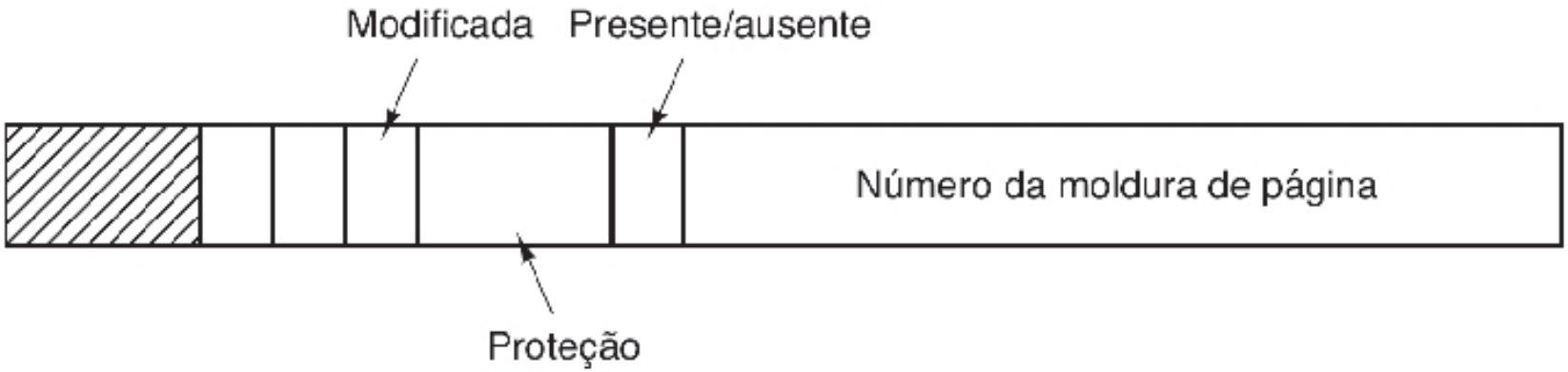
Entrada na Tabela de Páginas

- Em geral, 32 bits, divididos da seguinte maneira:
  - Bits de Proteção:
    - Indicam tipos de acessos permitidos à página:
      1 bit $\rightarrow$ 0 – leitura/escrita
      1 – leitura
      3 bits $\rightarrow$ 1^o bit – Leitura
      2^o bit – Escrita
      3^o bit – Execução



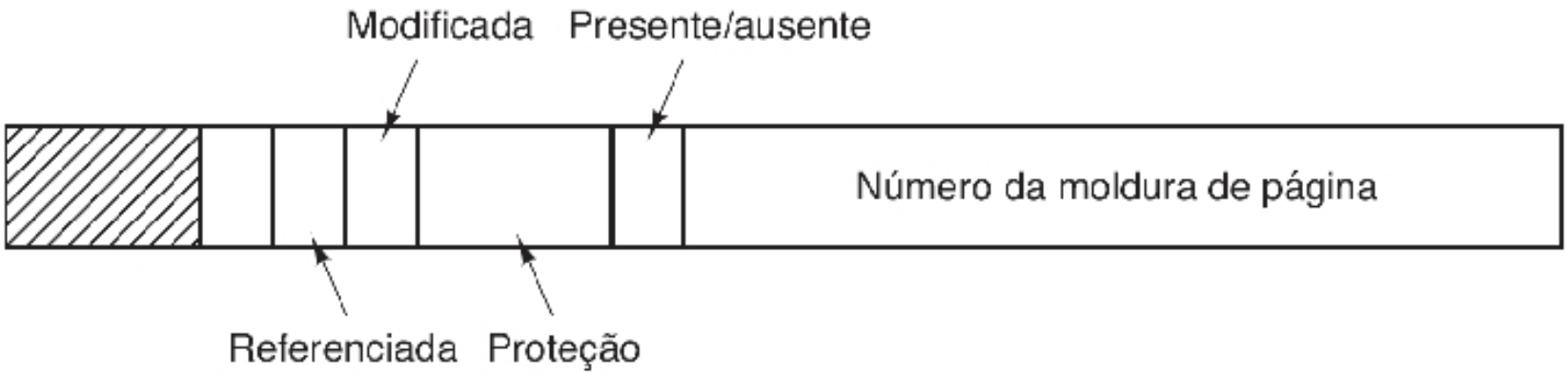
Entrada na Tabela de Páginas

- Em geral, 32 bits, divididos da seguinte maneira:
  - Bit de Modificação (ou bit sujo):
    - Controla o uso da página
    - Se página foi escrita, valor igual a 1 – a moldura deve copiada para o disco, caso seja removida da memória
    - Se valor igual a 0, página não foi modificada (está limpa) – a página é abandonada, já que a cópia em disco é válida



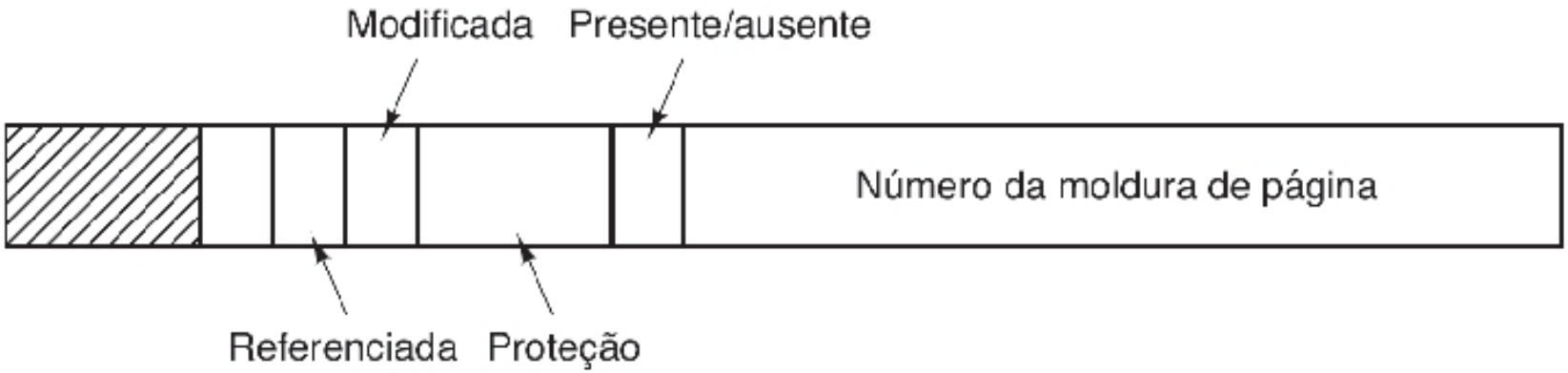
Entrada na Tabela de Páginas

- Em geral, 32 bits, divididos da seguinte maneira:
  - Bit de Referência:
    - Também controla o uso da página
    - Quando a página é referenciada (para leitura ou escrita), o hardware faz o bit ser 1



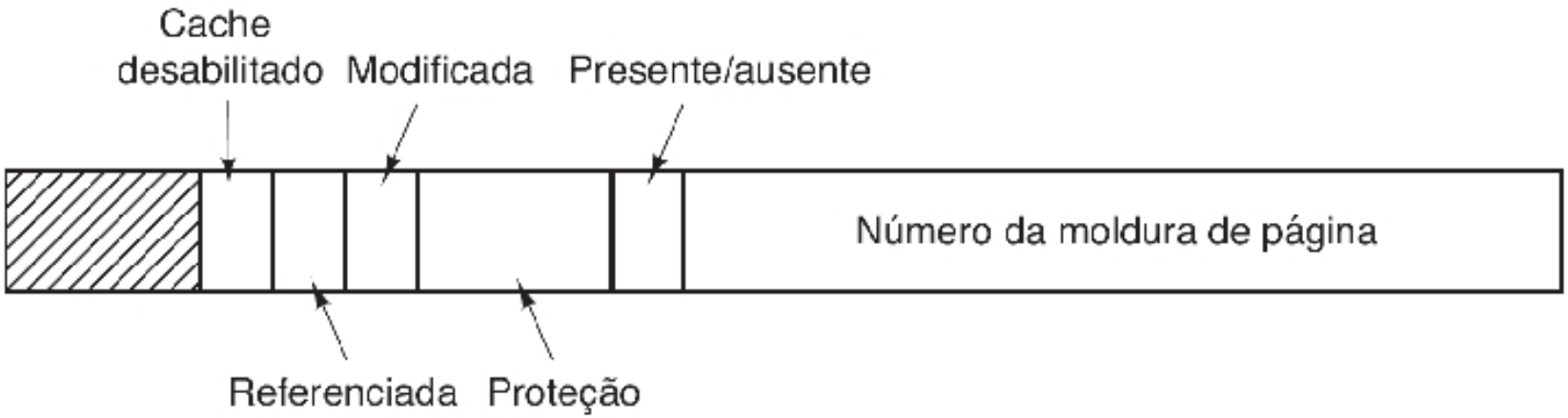
Entrada na Tabela de Páginas

- Em geral, 32 bits, divididos da seguinte maneira:
  - Bit de Referência:
    - Auxilia o SO na escolha da página que deve deixar a RAM, em caso de page fault – molduras que não estão em uso são melhores candidatas a sair
    - Em um dado intervalo de tempo, uma interrupção do clock faz o bit ser 0 (para todas as entradas) – apenas páginas referenciadas dentro do intervalo de clock são marcadas



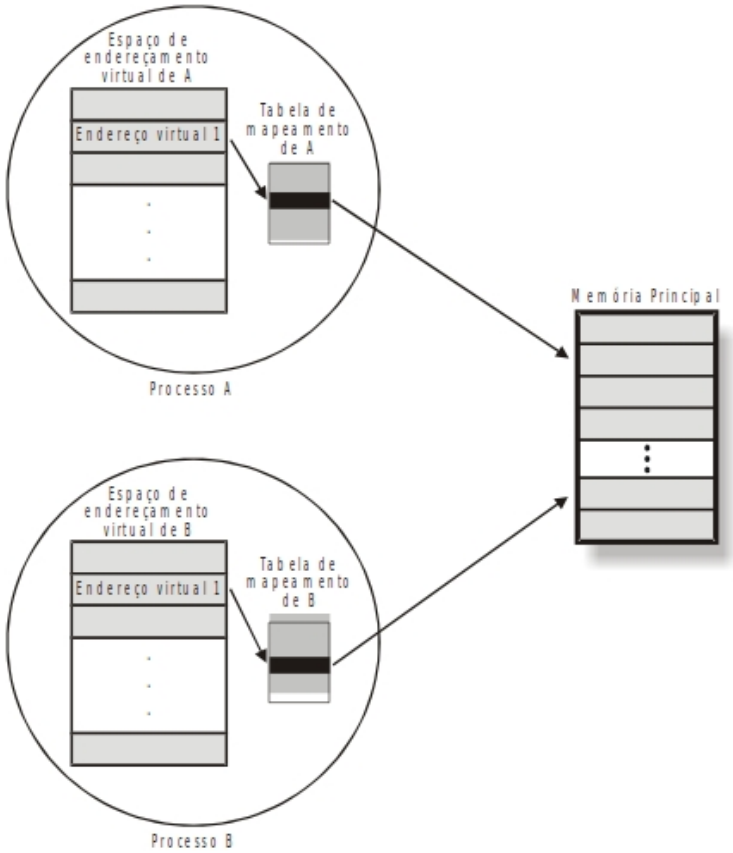
Entrada na Tabela de Páginas

- Em geral, 32 bits, divididos da seguinte maneira:
  - Bit de Cache:
    - Permite desabilitar o mecanismo de cache para a página
    - Necessário para páginas que acessam registradores de dispositivos, em vez da memória – O hardware deve acompanhar diretamente o dispositivo (Ex: E/S), e não usar uma cópia em cache



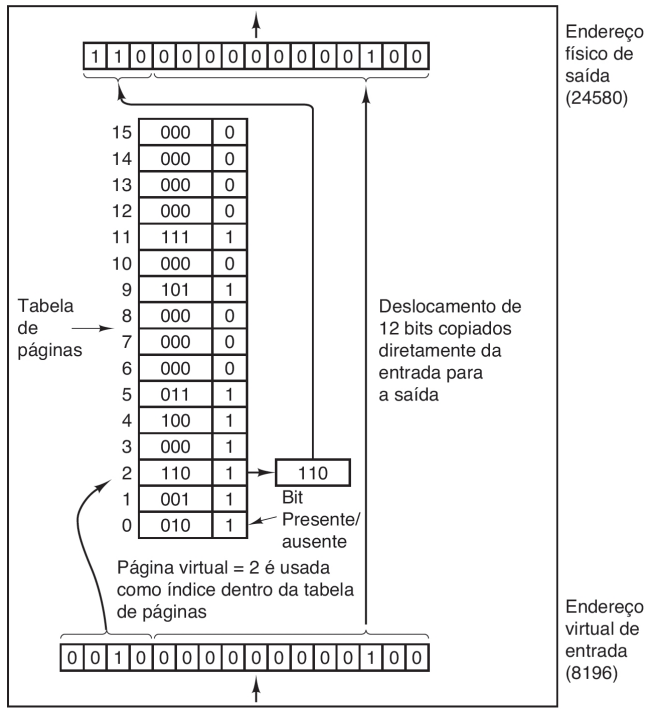
Memória Virtual – Tabela de Páginas

- Cada processo tem sua própria tabela
  - Cada um tem seu próprio espaço de endereçamento
  - Cada um acha que começa em uma mesma posição
  - Deve residir no BCP do processo
    - De fato, ponteiros para uma estrutura no kernel



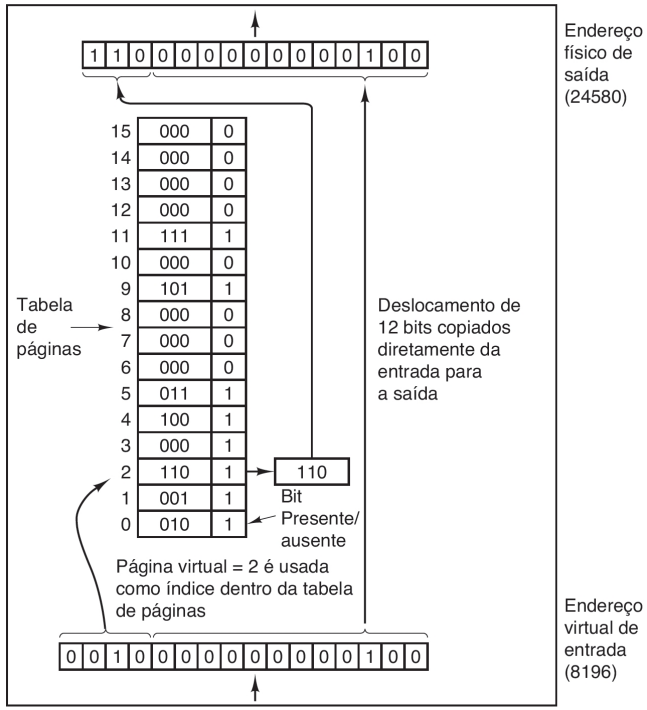
Memória Virtual – Tabela de Páginas

- Uma vez que a tabela reside no espaço do processo, e o mapeamento é feito pela MMU, como se dá a ligação entre eles?
  - Uma possibilidade é ter a tabela na forma de um arranjo de registradores, na MMU
    - Com uma entrada para cada página virtual
    - Indexado pelo número da página
    - Lembre que a tabela tem tamanho fixo pré-definido



Memória Virtual – Tabela de Páginas

- Quando um processo é inicializado, o SO carrega os registradores com a tabela de páginas desse processo
  - Vinda da memória
- Vantagem: não requer referências à memória durante o mapeamento (após carregar a tabela)
- Desvantagens:
  - Método caro, se a tabela for grande
  - A cada troca de contexto deve-se carregar toda a tabela – problema de desempenho



Memória Virtual – Tabela de Páginas

- Outra possibilidade seria manter a tabela inteiramente na memória
  - O hardware só precisa de um único registrador, que aponte para o início da tabela
    - Em uma mudança de contexto, muda-se apenas esse registrador
    - Algumas vezes há 2 registradores
  - Desvantagem:
    - Requer referências extras à memória (para ler as entradas na tabela) durante a execução de cada instrução
    - Mais lenta
  - Vantagem:
    - Acomoda tabelas grandes

Tabela de Páginas na Memória Principal

- Com 2 registradores associados:
  - Registrador de base da tabela de página (PTBR)
    - Page table base register
    - Aponta para o início da tabela de página, indicando o endereço físico de memória onde a tabela está alocada
    - No x86, tipicamente, os 20 bits mais altos do registrador CR3 são usados como PTBR (usa um esquema de páginas múltiplas – veremos mais adiante)
  - Registrador de tamanho da tabela de página (PTLR)
    - Existente apenas em alguns sistemas
    - Page-table length register
    - Indica tamanho da tabela de página (número de entradas da tabela $\rightarrow$ número de páginas)

Tabela de Páginas na Memória Principal

- Nesse esquema, cada acesso de dado/instrução exige dois acessos à memória:
  - Um para a tabela de página e um para o dado/instrução
  - Cada acesso à memória, feito no programa, se transforma em 2
  - Contudo, a maioria dos programas tende a fazer um grande número de referências a um pequeno número de páginas
    - Apenas uma fração pequena das entradas na tabela são lidas com grande frequência
    - O que fazer?

Memória Associativa (TLB)

- Solução:
  - O problema dos dois acessos à memória pode ser solucionado pelo uso de um cache de hardware especial para pesquisa rápida
    - Geralmente localizado dentro da MMU
    - Chamado memória associativa ou translation lookaside buffers (TLBs)
    - Hardware especial para mapear endereços virtuais para endereços físicos sem ter que passar pela tabela de páginas na memória principal
    - Consiste em um pequeno número de entradas, cada uma com informações sobre uma página

Memória Associativa (TLB)

Valida $= 1 \rightarrow$ a página está em uso

Válida $= 0 \rightarrow$ não está em uso

Usado para gerenciamento da TLB: se o programa usar menos páginas que as entradas disponíveis na TLB, marca-se algumas inválidas

Válida	Página virtual	Modificada	Proteção	Moldura da página
1	140	1	RW	31
1	20	0	R X	38
1	130	1	RW	29
1	129	1	RW	62
1	19	0	R X	50
1	21	0	R X	45
1	860	1	RW	14
1	861	1	RW	75

Memória Associativa (TLB)

Modificada $= 1 \rightarrow$ a página foi modificada

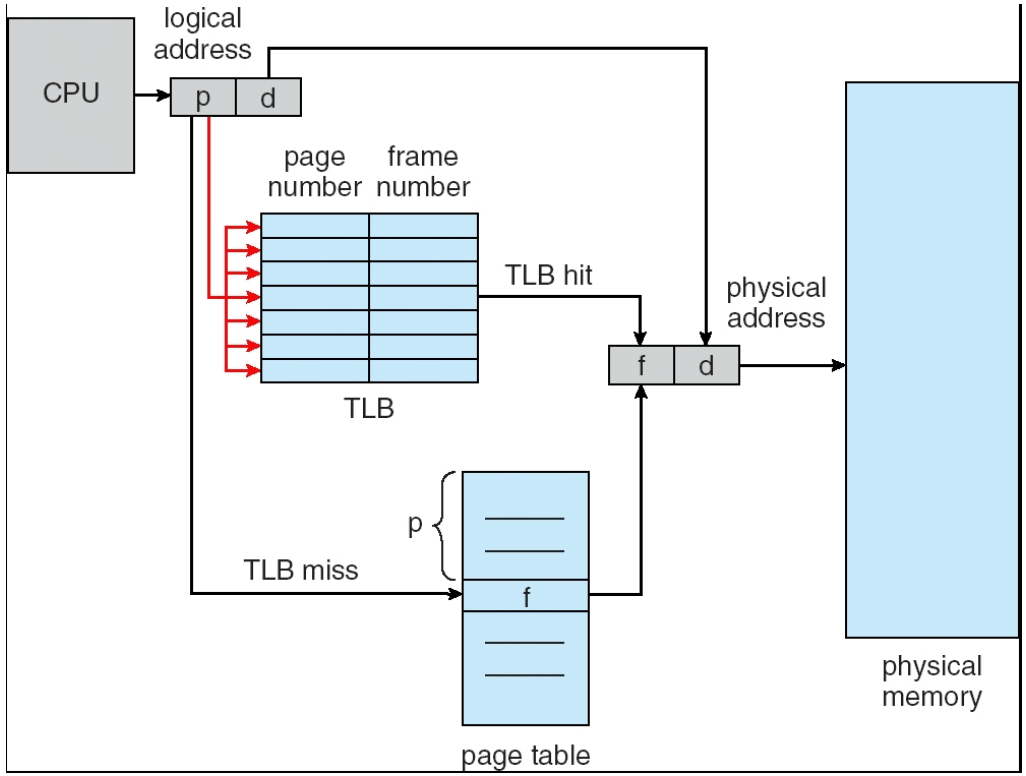
Página virtual: número da página virtual

Com exceção desse último e de "válida", todos os demais também estão na tabela de páginas

Válida	Página virtual	Modificada	Proteção	Moldura da página
1	140	1	RW	31
1	20	0	R X	38
1	130	1	RW	29
1	129	1	RW	62
1	19	0	R X	50
1	21	0	R X	45
1	860	1	RW	14
1	861	1	RW	75

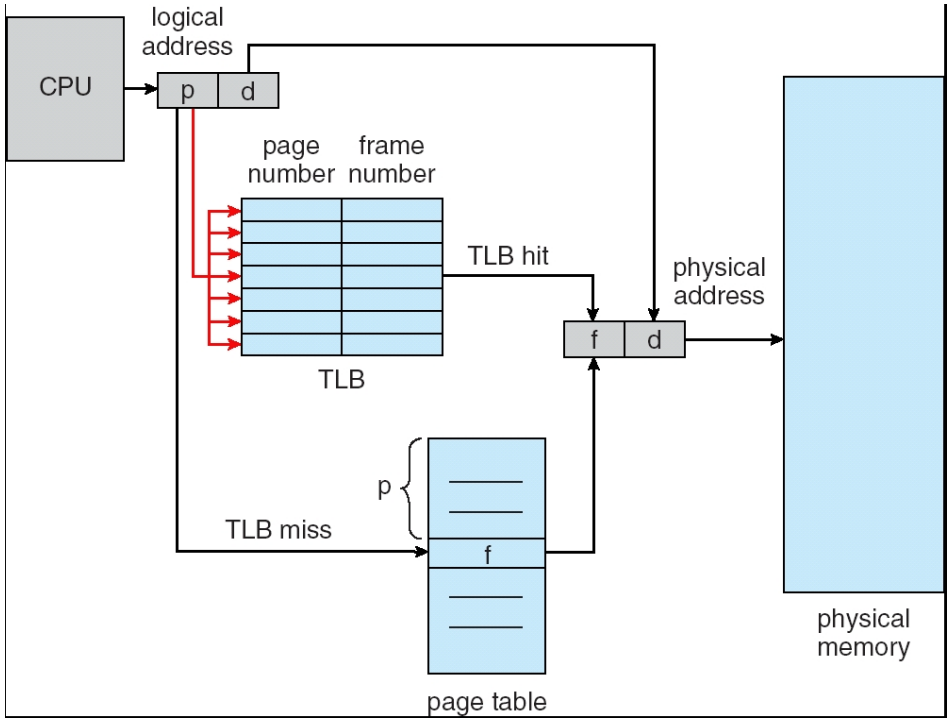
Memória Associativa (TLB)

- Funcionamento:
  - Quando um endereço virtual chega à MMU, o hardware verifica se sua página virtual está na TLB
    - Compara a todas as entradas simultaneamente (operação feita no hardware)



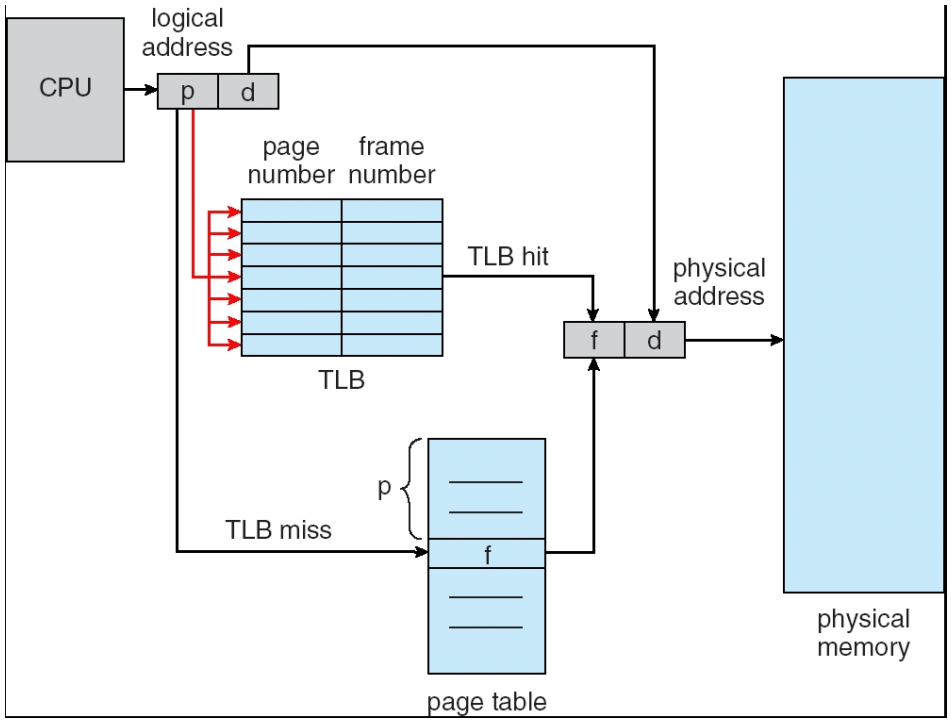
Memória Associativa (TLB)

- Funcionamento:
  - Se estiver na TLB (hit), e os bits de proteção não forem violados, a moldura é obtida da TLB, sem passar pela tabela de páginas
  - Se violar algum dos bits de proteção é gerada uma falha de proteção (protection fault)
    - Desvia ao S.O. via trap



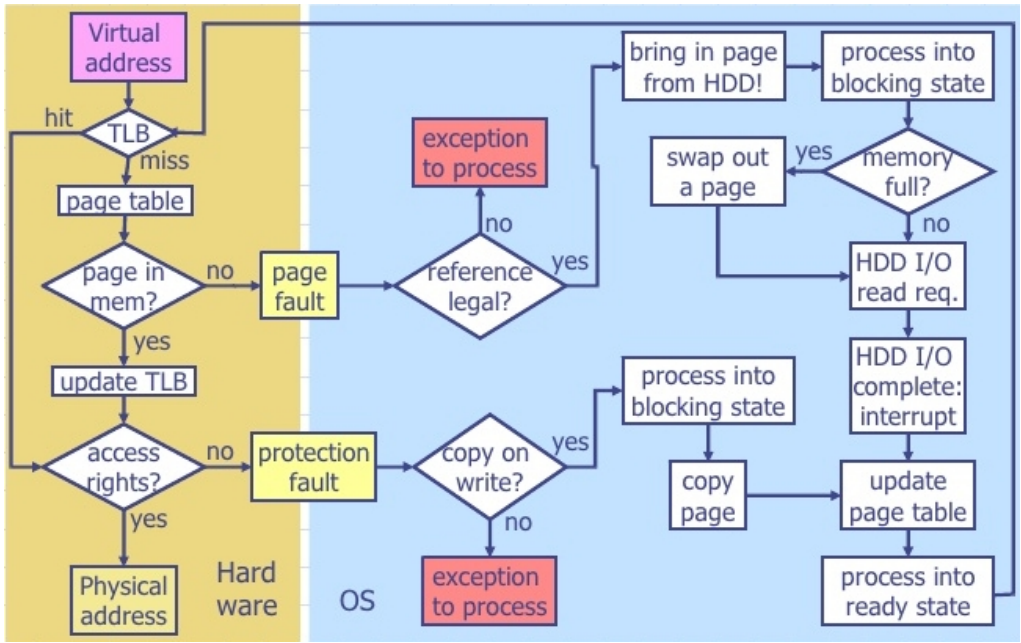
Memória Associativa (TLB)

- Funcionamento:
  - Se a página virtual não estiver na TLB
    - A MMU busca-a na tabela de páginas
    - Remove uma das entradas da TLB, devolvendo-a à tabela na memória (de fato, apenas o bit de modificada, pois o resto – exceto "válida" – já está lá)
    - Coloca nessa entrada a página que acabou de buscar
    - Se essa página for usada novamente, estará na TLB



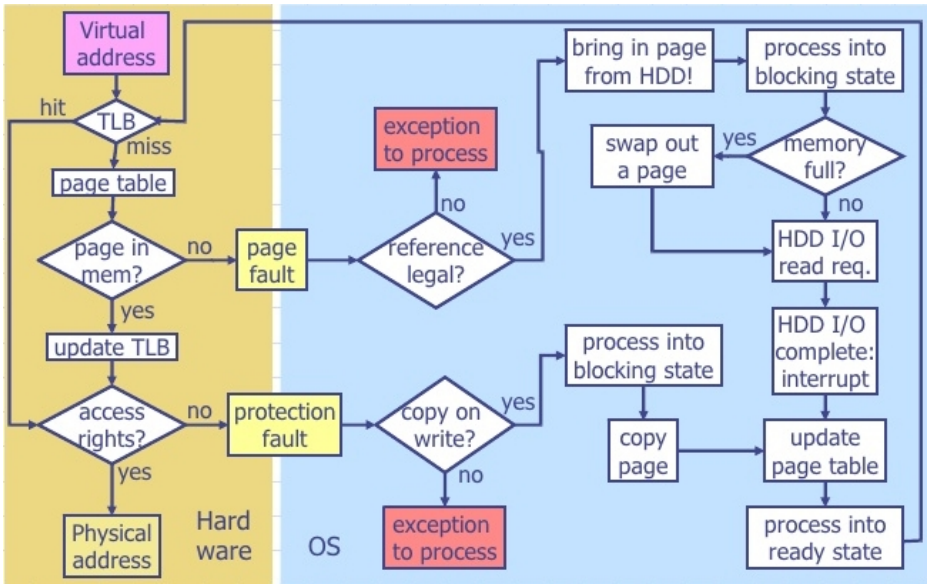
Memória Associativa (TLB)

- O gerenciamento da TLB pode ser:
  - Por hardware (X86)
    - Maior rapidez
    - Ocupa espaço físico que poderia ser disponibilizado para outras funções (cache etc)



TLB – Gerenciamento por Hardware

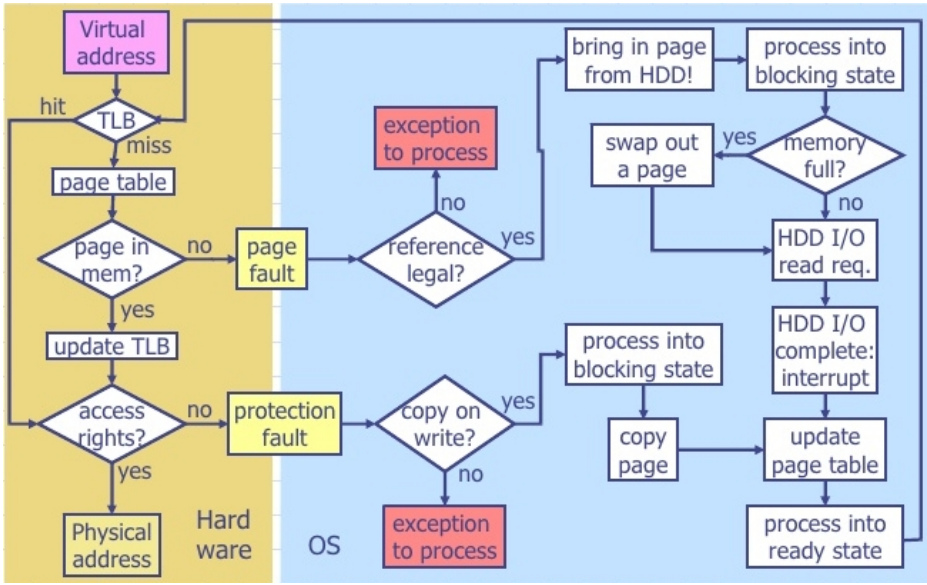
- Página inválida:
  - MMU gera uma interrupção de proteção e aciona o sistema operacional



  - Se a página estiver fora do espaço de endereçamento do processo, o processo é abortado
  - Se a página ainda não foi carregada na memória principal, ocorre uma falta de página (page fault)

TLB – Gerenciamento por Hardware

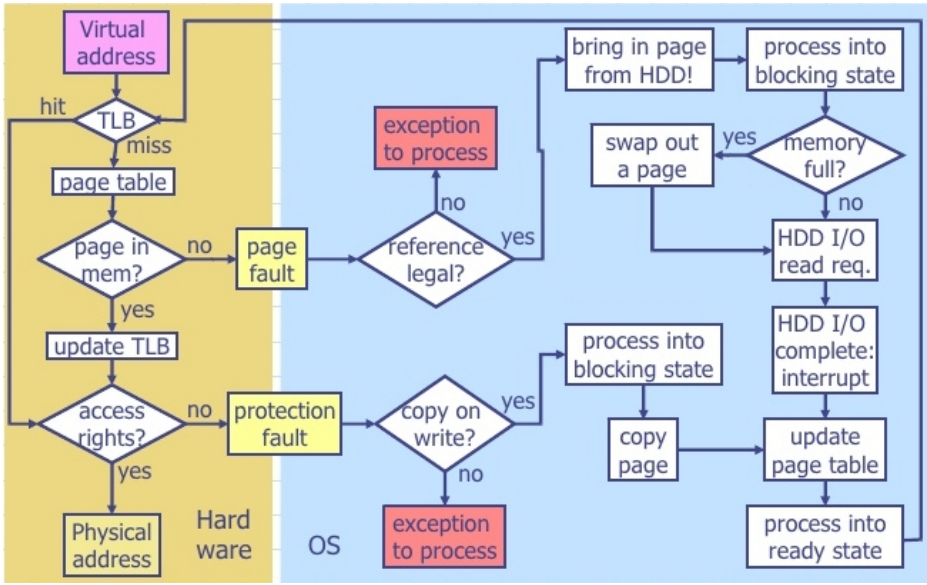
- Página inválida:
  - O processo é suspenso e seu descritor é inserido na fila dos processos esperando uma página virtual



  - Uma moldura livre (na memória) deve ser alocada
  - A página virtual acessada deve ser localizada no disco
  - Operação de leitura de disco
    - Indicando o endereço da página virtual no disco e o endereço da moldura alocada

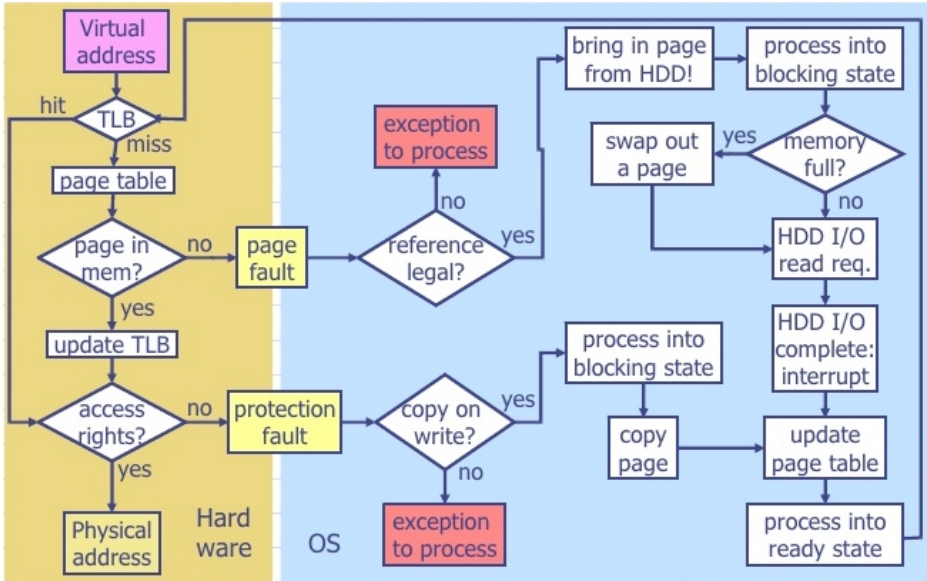
TLB – Gerenciamento por Hardware

- Página inválida:
  - Após a leitura do disco:
    - A tabela de páginas do processo é corrigida para indicar que a página virtual agora está válida e está na moldura alocada
    - Tarefa executada pelo Pager (rotina que carrega páginas especificas de um processo do disco para a memória principal)
    - O descritor do processo é retirado da fila especial e colocado na fila de prontos (para execução)



TLB – Gerenciamento por Hardware

- Copy on write?
- Quando um processo faz um fork:
  - As páginas na memória que podem ser modificadas tanto pelo pai quanto pelo filho são marcadas como C.o.W.
    - Não há, inicialmente, múltiplas cópias delas na memória

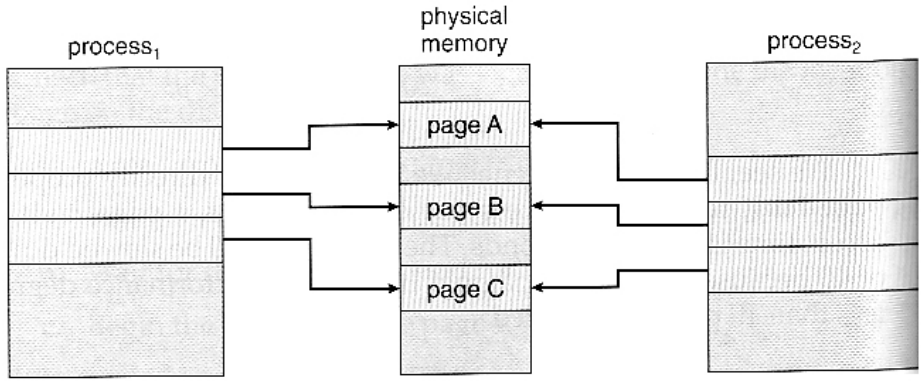


Copy on Write

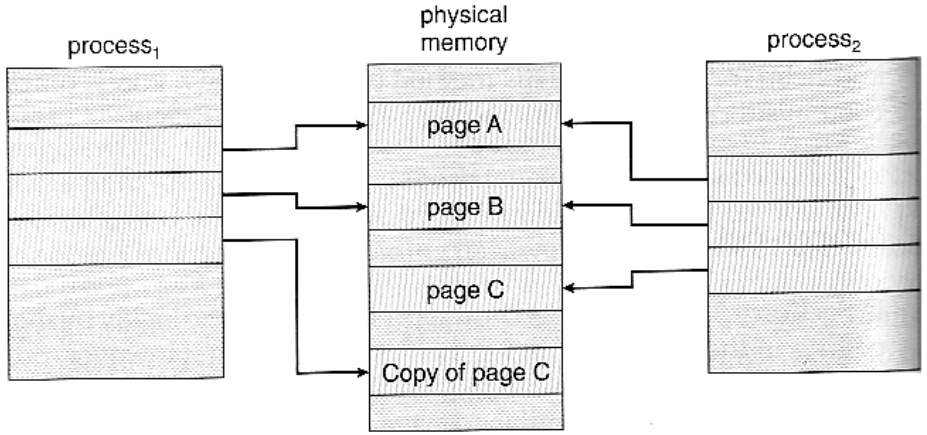
- Quando um processo modifica a memória, o SO faz a cópia efetiva
  - Mudanças em um processo não são visíveis ao outro
  - Copia ao filho somente as páginas necessárias
- Implementada pela notificação da MMU que certas páginas do processo são read-only
  - Daí a falha de proteção ao se escrever nelas

Copy on Write

Antes de modificar C



Depois de modificar C



Referências Adicionais

- http://en.wikipedia.org/wiki/Control_register
- http://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/9_VirtualMemory.html
- http://en.wikipedia.org/wiki/Copy-on-write