Paginação e memória virtual

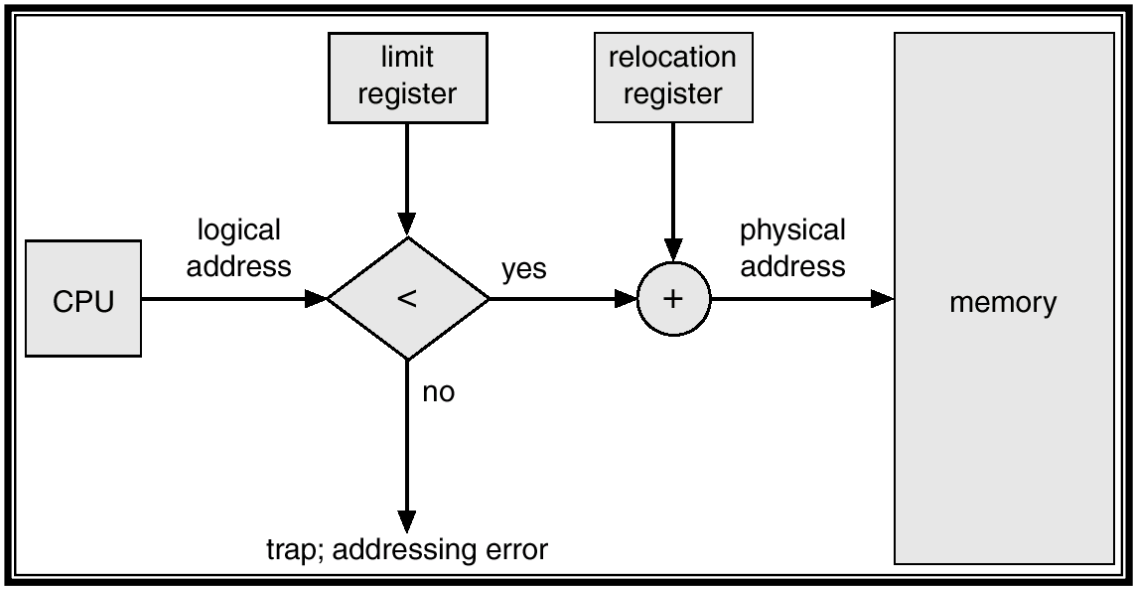
Secções 18.2 a 18.8.1

S.P. Dandamudi, *Fundamentals of Computer Organization and Design*, Ed. Springer, 2003

Espaço físico gerido por alocação contígua

- Cada imagem de um processo tem de ser carregada numa partição de dimensão maior ou igual à dimensão da imagem do processo
- Suporte hardware: registos base e limite
  - ✓ Asseguram a recolocação do programa e a protecção
  - ✓ O valor corrente dos registos base e limite estão associados ao processo

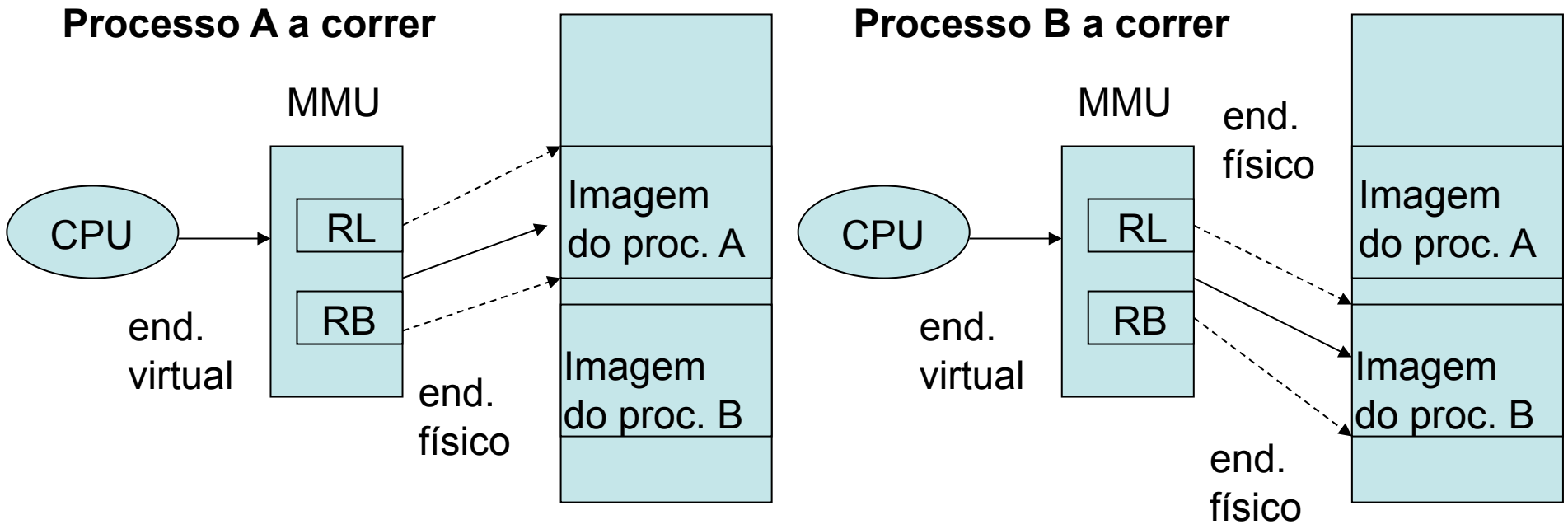
Suporte hardware para a transformação de endereços



Endereço virtual linear mapeado em partição de memória contígua

Hardware necessário: registos base e limite

- Suporte hardware: registos base e limite
  - ✓ Asseguram a recolocação do programa e a protecção
  - ✓ O valor corrente dos registos base (RB) e limite (RL) são guardados no descritor do processo

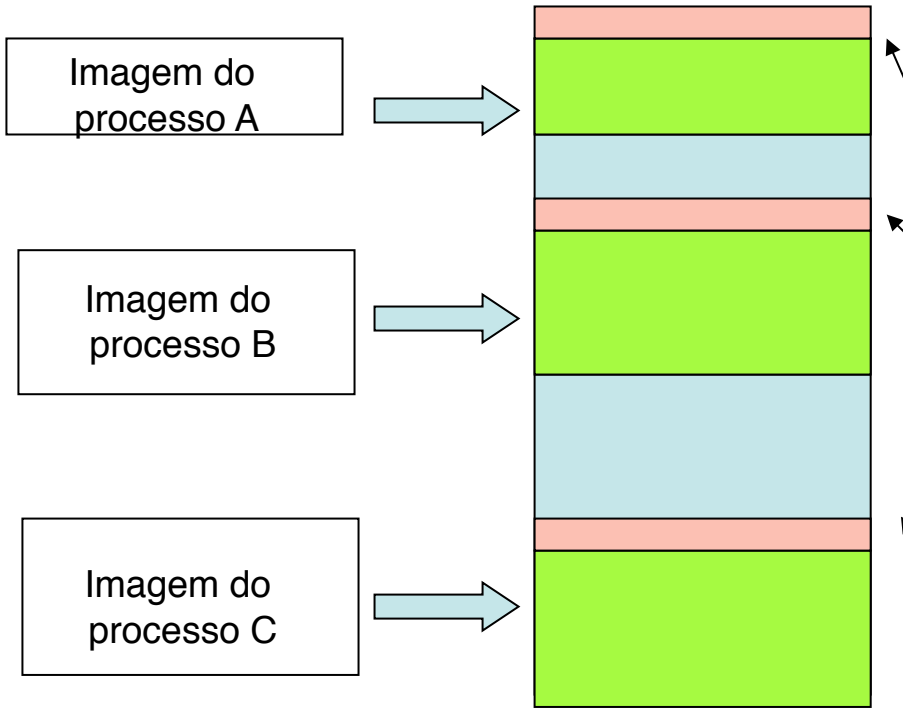


Espaço físico gerido por atribuição contígua

- Memória dividida em duas zonas:
  - ✓ SO.
  - ✓ Espaço para imagens de processos.
- Espaço do utilizador
  - ✓ Uma partição: não admite multiprogramação
  - ✓ Várias partições de tamanho fixo
    - ✓ Suporta multiprogramação, mas é inflexível
    - ✓ Fragmentação interna (espaço desperdiçado dentro da partição)
  - ✓ Número variável de partições e de dimensão variável
    - ✓ Partições criadas a pedido
    - ✓ Sofre de fragmentação externa (pode ser impossível juntar todo o espaço livre)

Espaço físico gerido por alocação contígua

- Cada imagem de processo tem de ser carregada numa partição de dimensão maior ou igual à dimensão da imagem do processo



Para carregar um processo tem de estar livre uma partição de dimensão adequada, ou então ser criada uma nova a pedido na altura

Espaço desperdiçado

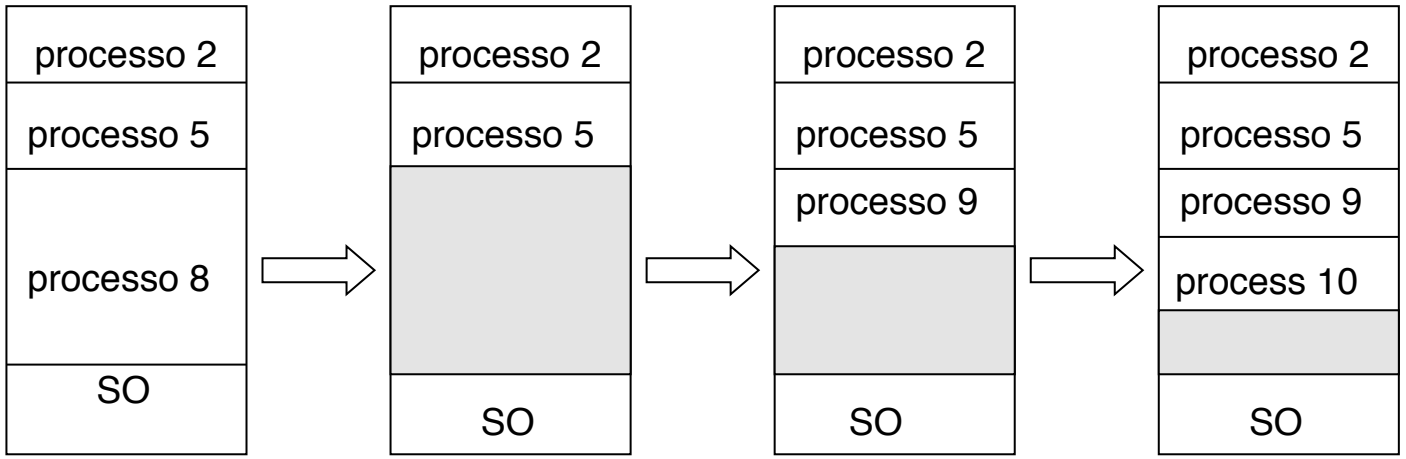
Memória física disponível para os processos

Formas de organizar as partições

- **Número de partições fixo e de dimensão fixa**
  - ✓ pouco flexível; apenas utilizável em ambientes em que as necessidades variam pouco;
  - ✓ nº de processos máximo igual ao nº de partições;
  - ✓ dado um processo cuja imagem tem dimensão, L se não houver uma partição de dimensão igual ou superior livre, o processo não pode ser carregado
- **Número de dimensões variável e com dimensão também variável**
  - ✓ existe um conjunto de memória livre e a partição é criada "à medida" do tamanho da imagem do processo a carregar

Inconvenientes da atribuição de memória por partições contíguas

- Zonas livres de diferentes dimensões estão espalhadas pela memória física.
- Quando um processo é criado, é necessário atribuir-lhe uma zona livre contígua suficientemente grande



Fragmentação

- **Fragmentação externa** – existe memória física livre em quantidade suficiente para carregar o processo, mas não está contígua.
- **Fragmentação interna** – espaço existente dentro de uma partição que não é usado porque a a dimensão da imagem é inferior ao da partição
- Operações de compactação reduzem o problema da fragmentação externa
  - ✓ Juntar a memória livre num bloco único.
  - ✓ Operação feita em tempo de execução; só possível com recolocação dos programas (ex: registos base e limite)
  - ✓ Operação que exige muitas operações de I/O e provoca a paragem temporária de todos os programas envolvidos

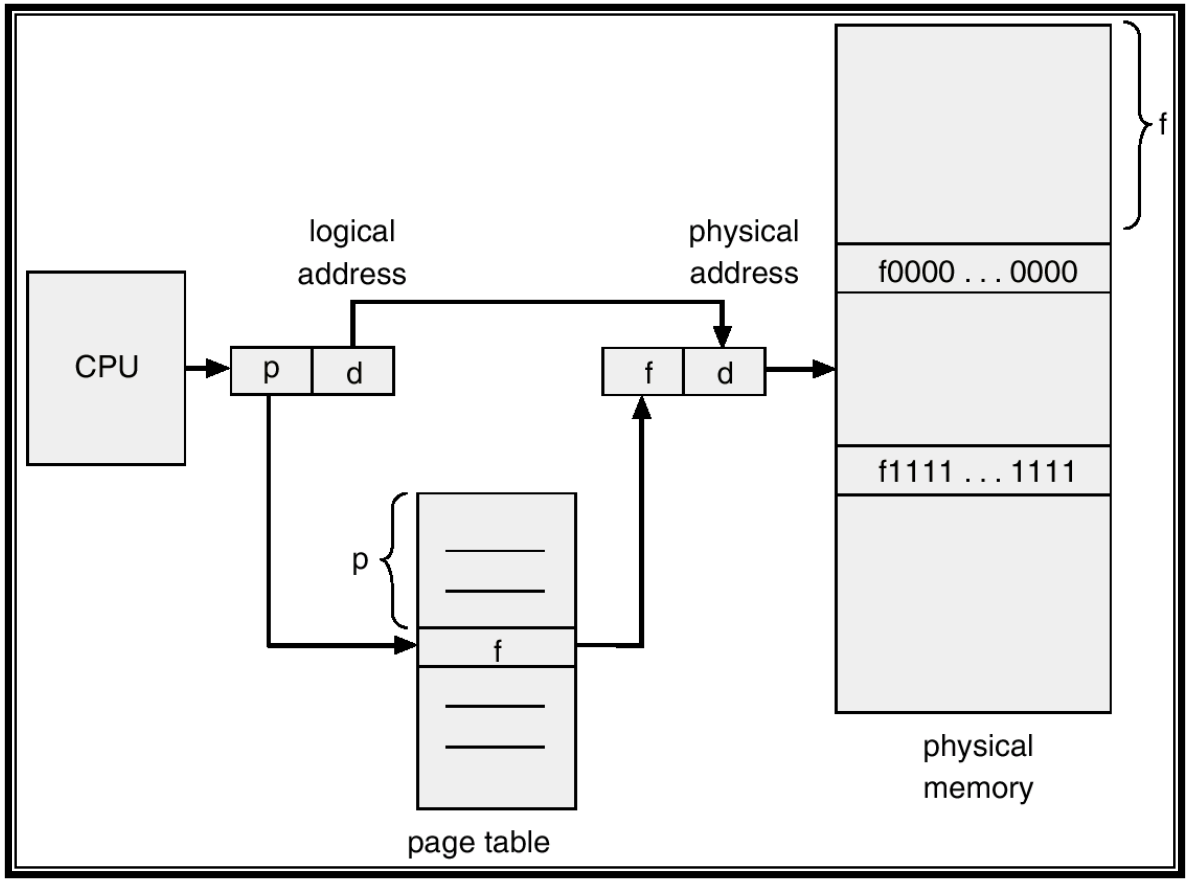
Paginação

- O espaço de endereçamento de um processo não necessita de ser contíguo – atribui-se memória física ao processo onde quer que ela esteja disponível
- Divide-se a memória física em blocos de tamanho fixo chamadas páginas físicas (**frames**) – tamanho é uma potência de 2 entre 512 bytes e 8192 bytes.
- Divide-se o espaço de endereçamento lógico em blocos do mesmo tamanho chamadas **páginas** (ou páginas virtuais).
- Manter informação sobre todas as frames livres.
- Para executar um programa com *n* páginas, é preciso encontrar *n* frames livres e carregar lá o programa.
- Preparar um tabela de páginas que traduz endereços virtuais em endereços físicos.
- Não há fragmentação externa
- Há fragmentação interna (em média ½ página)

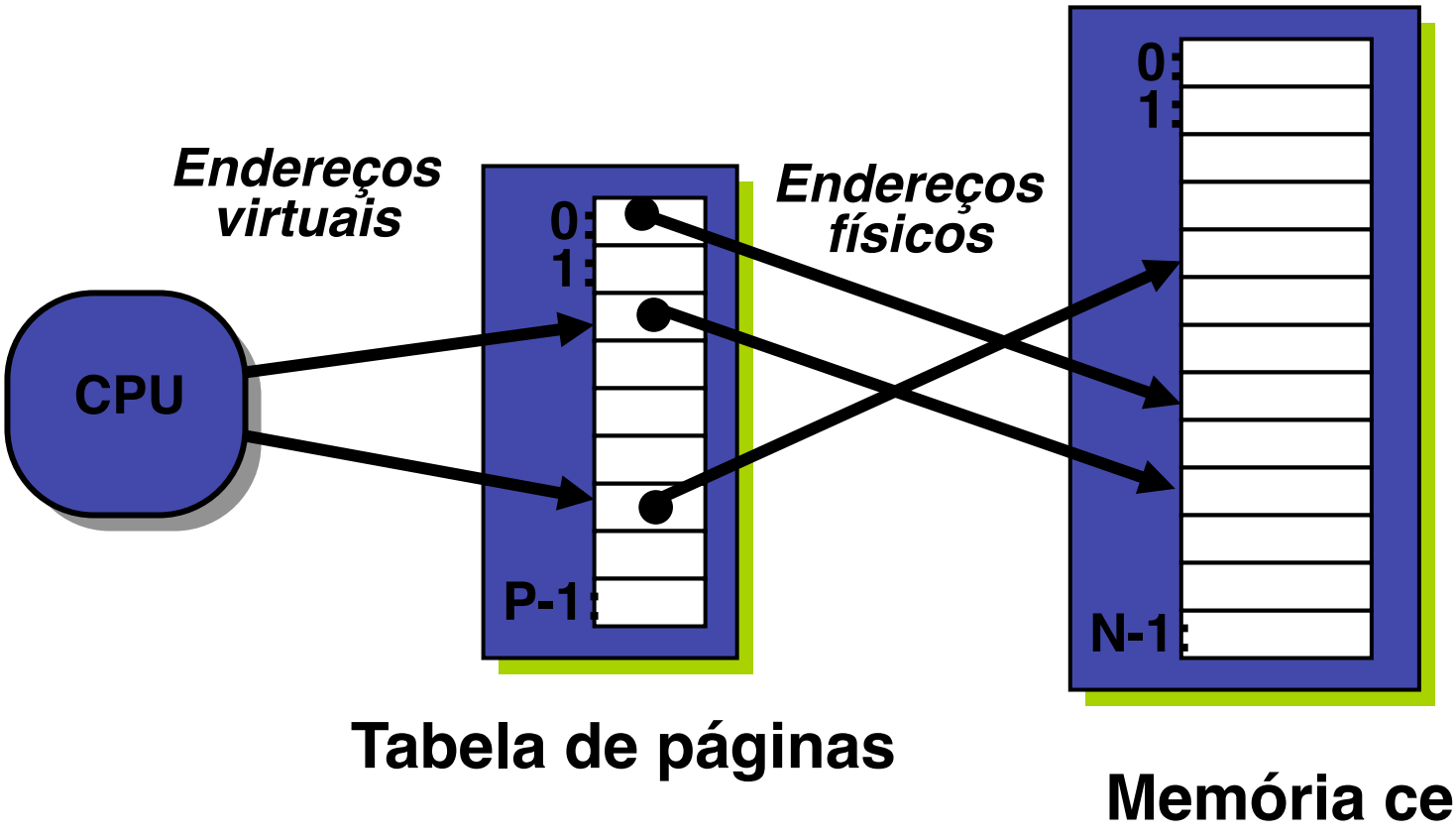
Transformação de endereços usada para paginação

- ✓ O endereço virtual (gerado pelo CPU) é composto por um número de página virtual (P) e um deslocamento (D)
- ✓ *P* : usado como índice na *tabela de páginas* que contém o endereço base de cada página na memória física.
- ✓ *D:* combinado com o endereço base define o endereço físico enviado para a memória

Transformação endereço virtual em físico



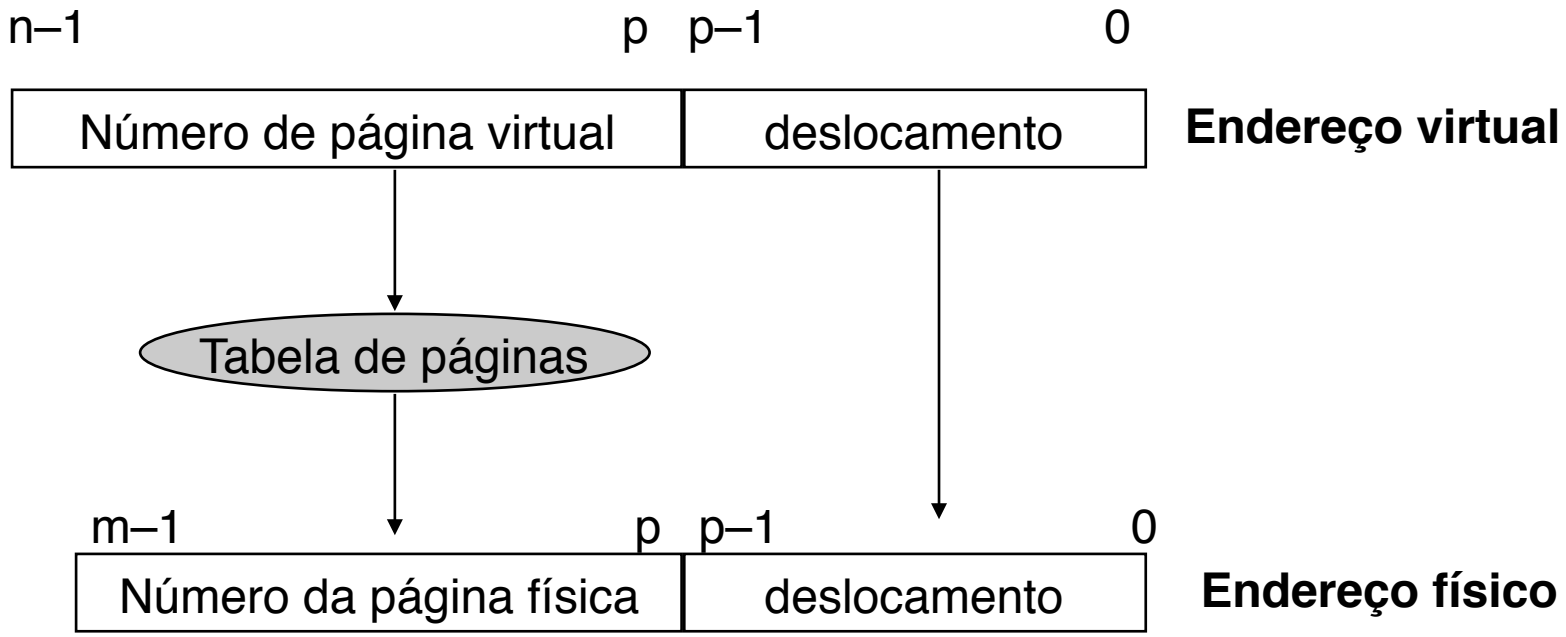
Gestão de memória física por páginas



Transformação de endereços: Hardware converte *endereços virtuais* em *endereços físicos* através de um tabela gerida pelo SO (*tabela de páginas*)

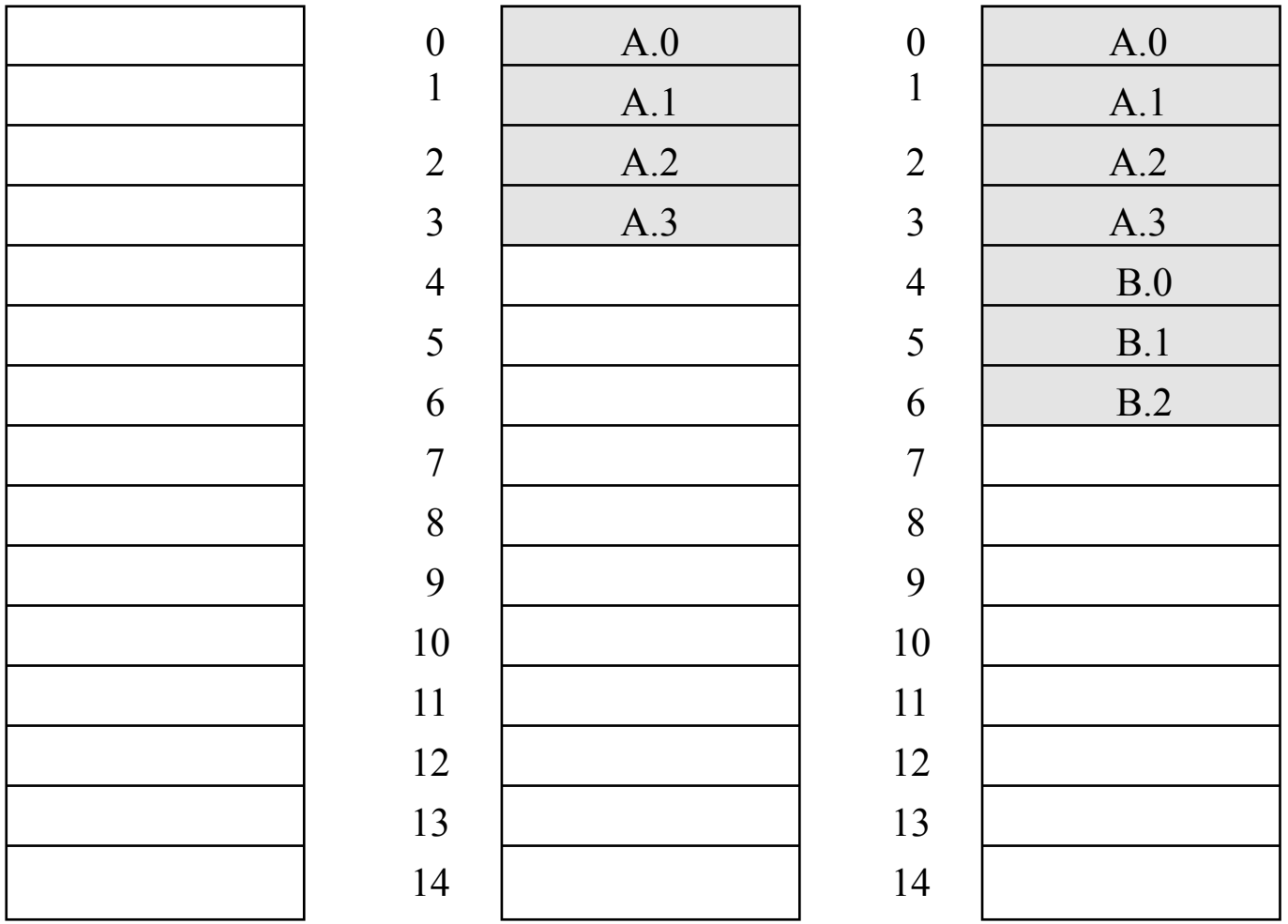
Transformação de endereços

- Parâmetros
  - ✓P = 2^p = tamanho da página (bytes).
  - ✓N = 2^n = Endereço virtual tem n bits
  - ✓M = 2^m = Endereço físico tem m bits

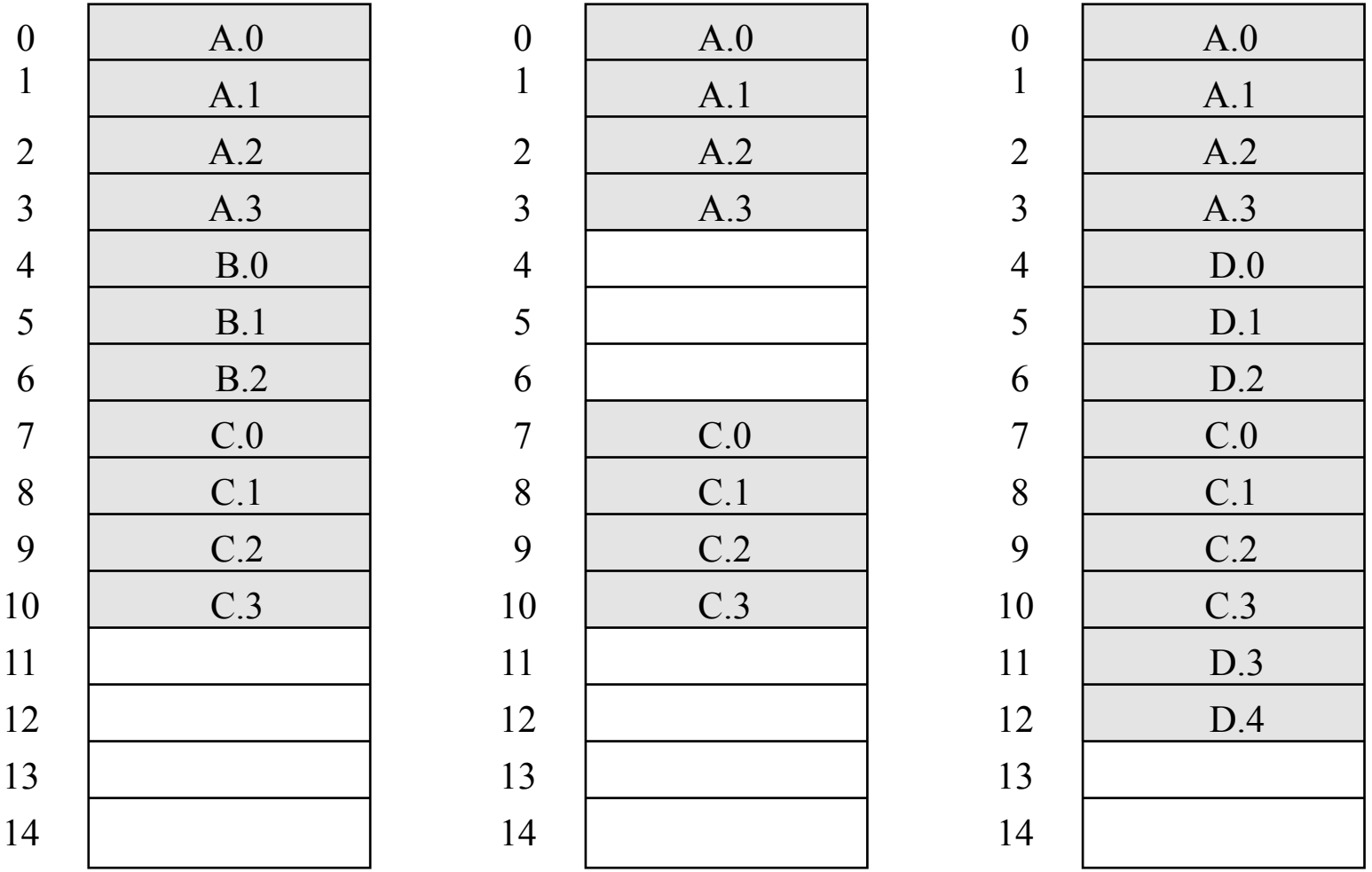


Note-se que os bits correspondentes ao deslocamento (dentro da página) não mudam durante a transformação

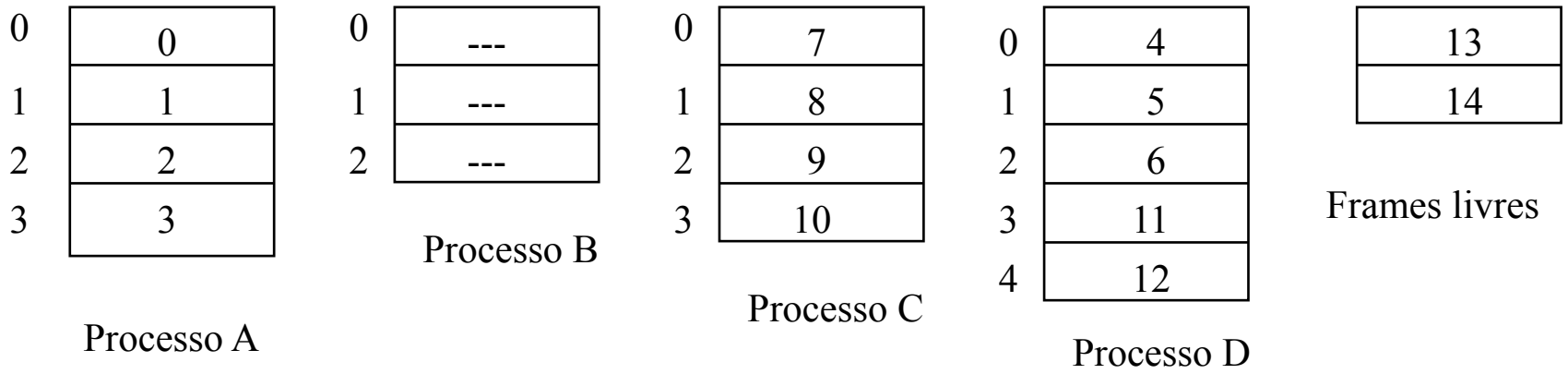
Carregando os processos A e B em memória



Paginação: atribuição não contígua



Tabelas de páginas para o exemplo



O SO mantém uma tabela de páginas para cada processo:

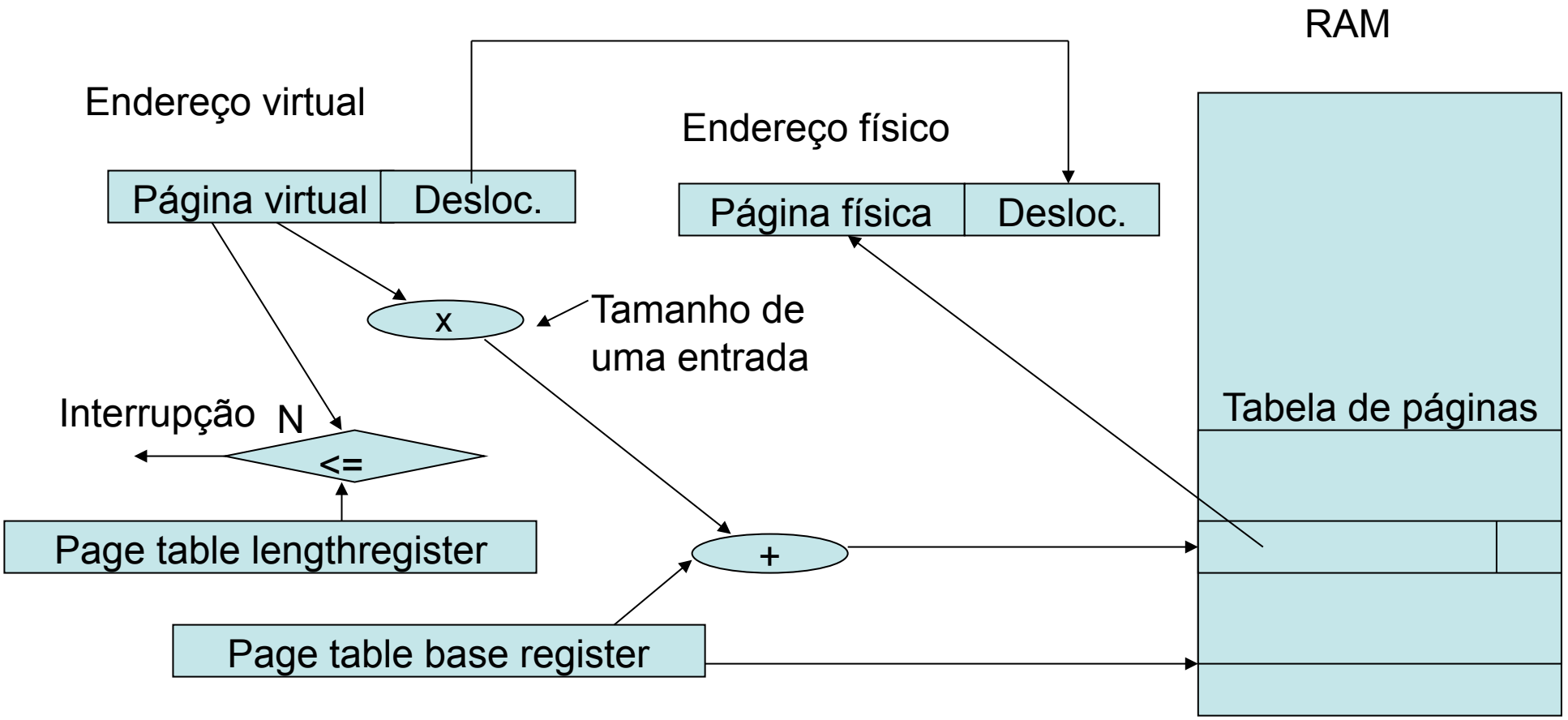
- Contém a “frame” correspondente a cada página do processo

Suporte da tabela de páginas

- Onde se guarda a tabela de páginas de um processo?
  - ✓ No caso do Pentium a tabela tem 2^{20} entradas; se cada uma tiver 4 bytes, são 4 Mbytes …
  - ✓ Não é tecnicamente viável guardar a tabela de páginas na MMU
  - ✓ Há uma tabela de páginas por processo …
- Solução:
  - ✓ Guardar a tabela de páginas em memória

Suporte da tabela de páginas

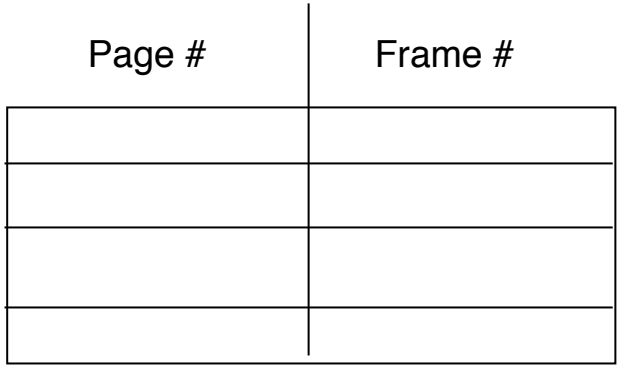
- Tabela de páginas em memória central (RAM).
- *Page-table base register (*PTBR) aponta para a base da tabela.
- *Page-table length register* (PRLR) indica o tamanho da tabela de páginas.



Suporte da tabela de páginas

- Neste esquema cada acesso a dados ou código obriga a dois acessos à memória: um para tabela de páginas e outra para o dado / instrução.
- O problema dos dois acessos à memória pode ser resolvido por hardware especial de consulta rápida chamado *translation look-aside buffer (TLB)* que se baseia numa *memória associativa* ou interrogável pelo conteúdo

TLB - Memória associativa

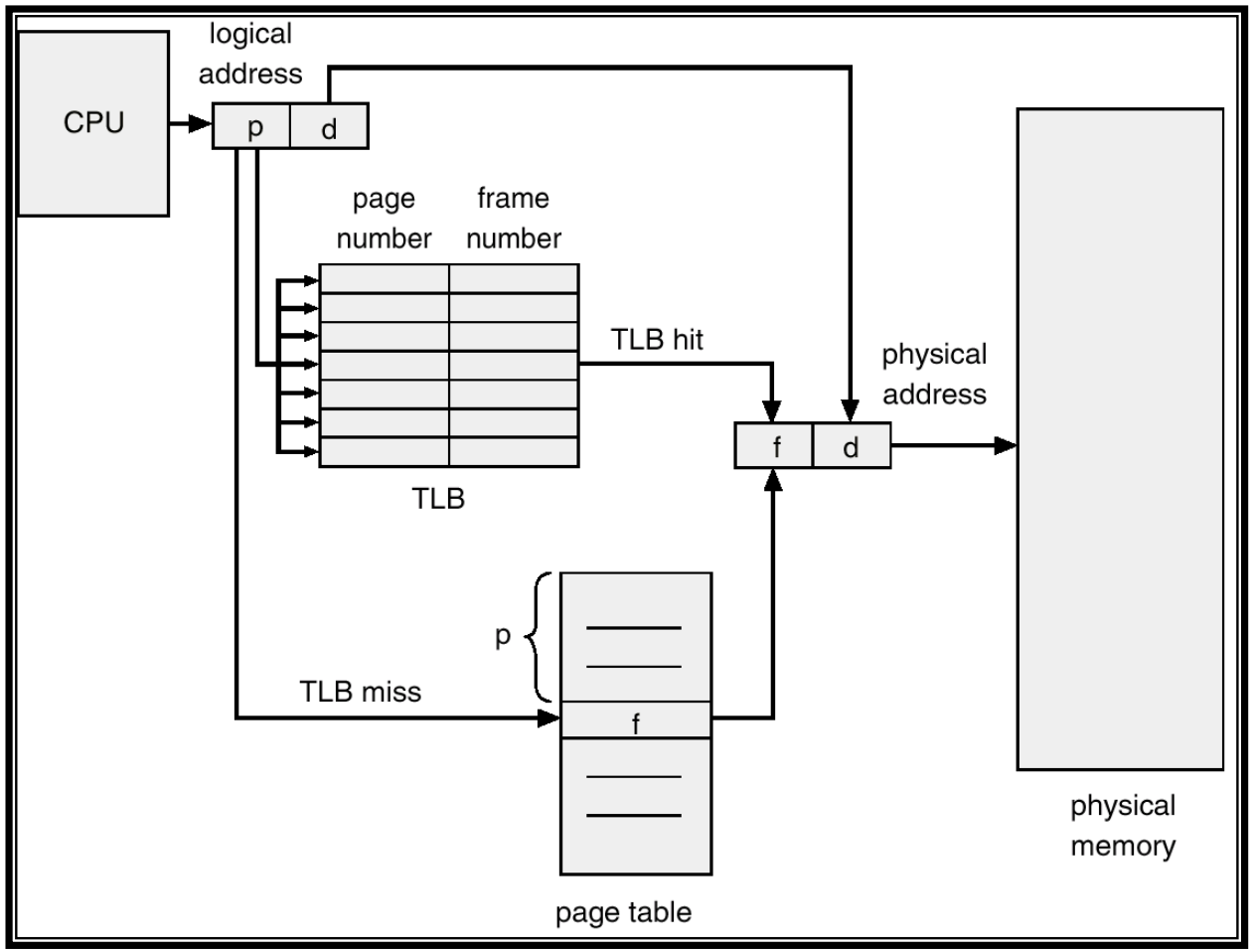


Transformação de endereços (A´, A´´)

- ✓ Se A´ está na memória associativa, TLB responde com o número da "frame" A´´.
- ✓ Senão obter o número da "frame" "da tabela de páginas na memória

Hardware de paginação com TLB

- "Translation Lookaside Buffer" (TLB)
  - pequena cache hardware na MMU
  - Faz a correspondência o nº página virtual em nº página física



Protecção de memória

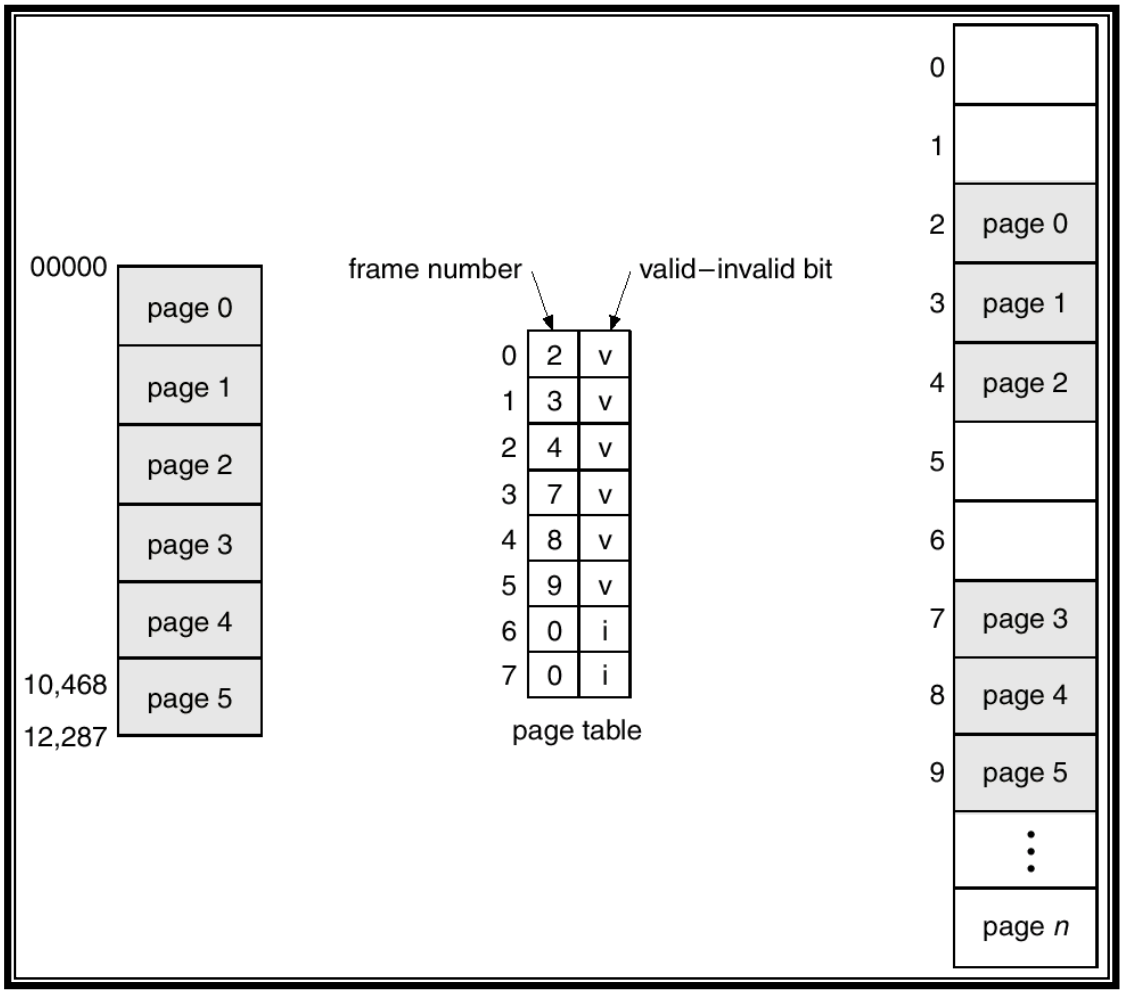
- A protecção de memória está associada ao preenchimento que o SO faz da tabela de páginas do processo

- *Valid-invalid* bit (bit V) associado a cada entrada da tabela de páginas:
  - ✓ "válido" indica que a página está no espaço de endereçamento lógico do processo – é uma página legal
  - ✓ "inválido" indica que a página não está no espaço de endereçamento virtual do processo.

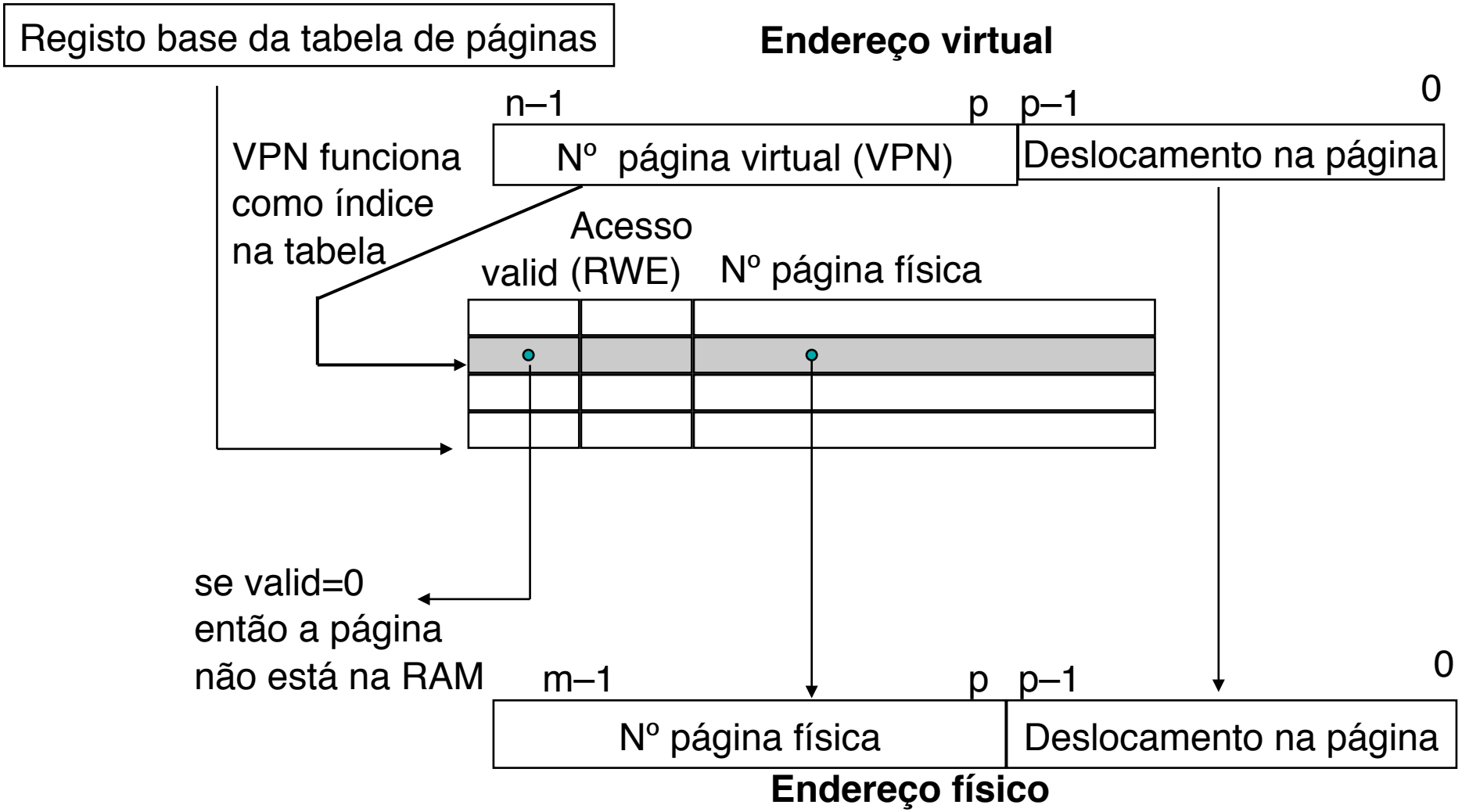
Informação para cada página

- A entrada I da tabela de páginas contém para a página I a seguinte informação:
  - ✓ Bit de validade V (página pertence ou não ao espaço de endereçamento do processo)
  - ✓ Número da frame F (se V==1)
  - ✓ Bits que definem as possibilidades de acesso: READ-ONLY, WRITE, EXECUTE
    - ✓ Páginas de código tem EXECUTE e READ-ONLY
    - ✓ Páginas de dados tem READ/WRITE
    - ✓ ...
  - ✓ Outros bits usados para gestão
    - ✓ Dirty bit (página alterada)

Bit de validade na tabela de páginas



Transformação de endereços usando a tabela de páginas



Memória virtual

- **Memória virtual (VM)** – separação da memória lógica (virtual) da memória física
  - ✓ Só parte dos programas precisam de estar carregados em memória.
  - ✓ Espaços de endereçamento virtuais podem ser muito maiores do que os físicos.
  - ✓ Permite partilha de partes do espaço de endereçamento entre processos.
  - ✓ Permite uma criação de processos mais eficiente.
  - ✓ É completamente transparente ao utilizador/programador

- VM pode ser suportada através **paginação a pedido**
  - ✓ Todas as páginas virtuais da imagem do processo estão em disco, chamado *disco de paginação* ou *disco de swap*
  - ✓ As páginas só vêm para memória central (RAM) quando são referenciadas, i.e. quando há um endereço virtual emitido pelo CPU em que os bits mais significativos correspondem a esse número de página virtual

Disco de paginação

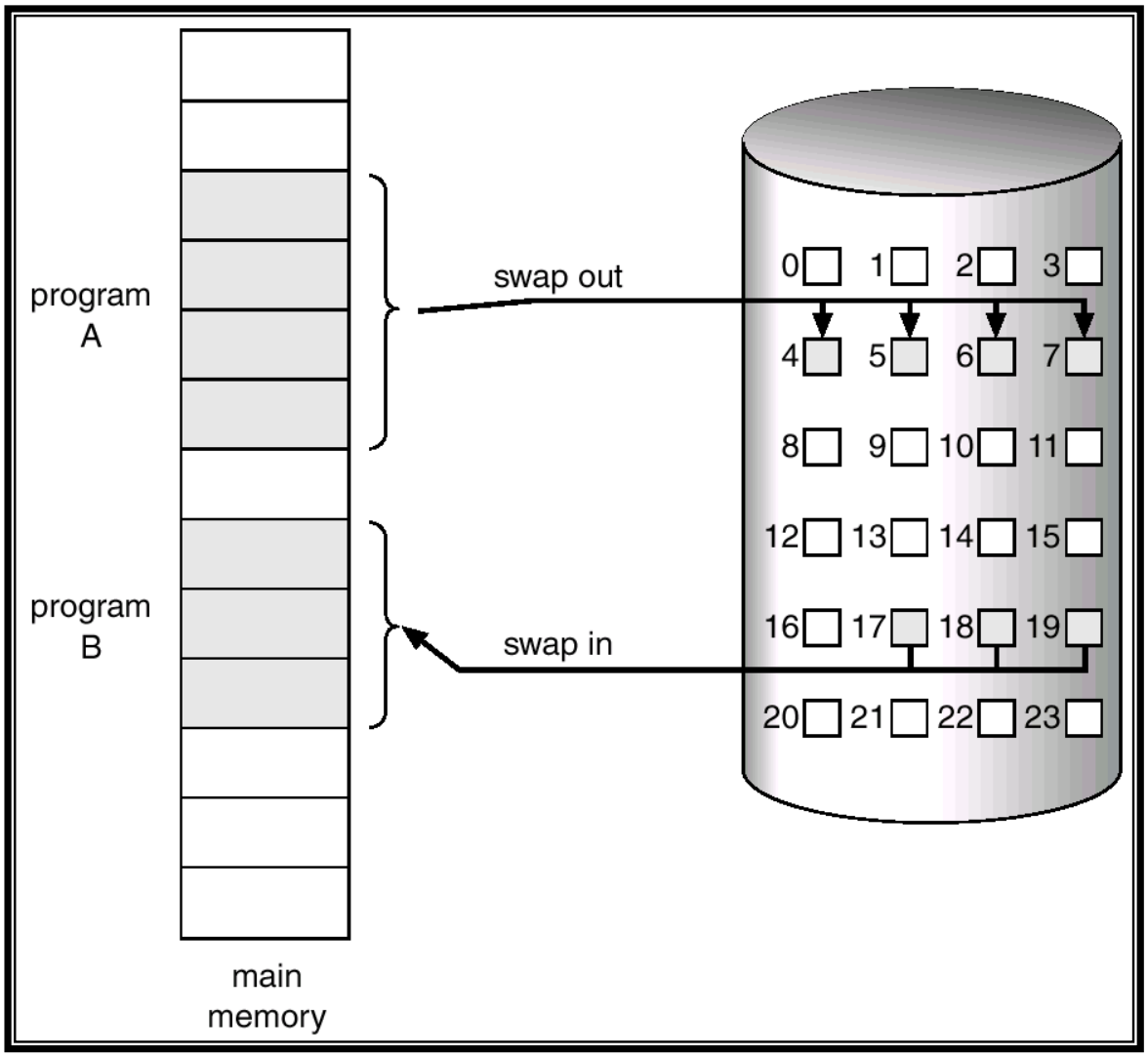
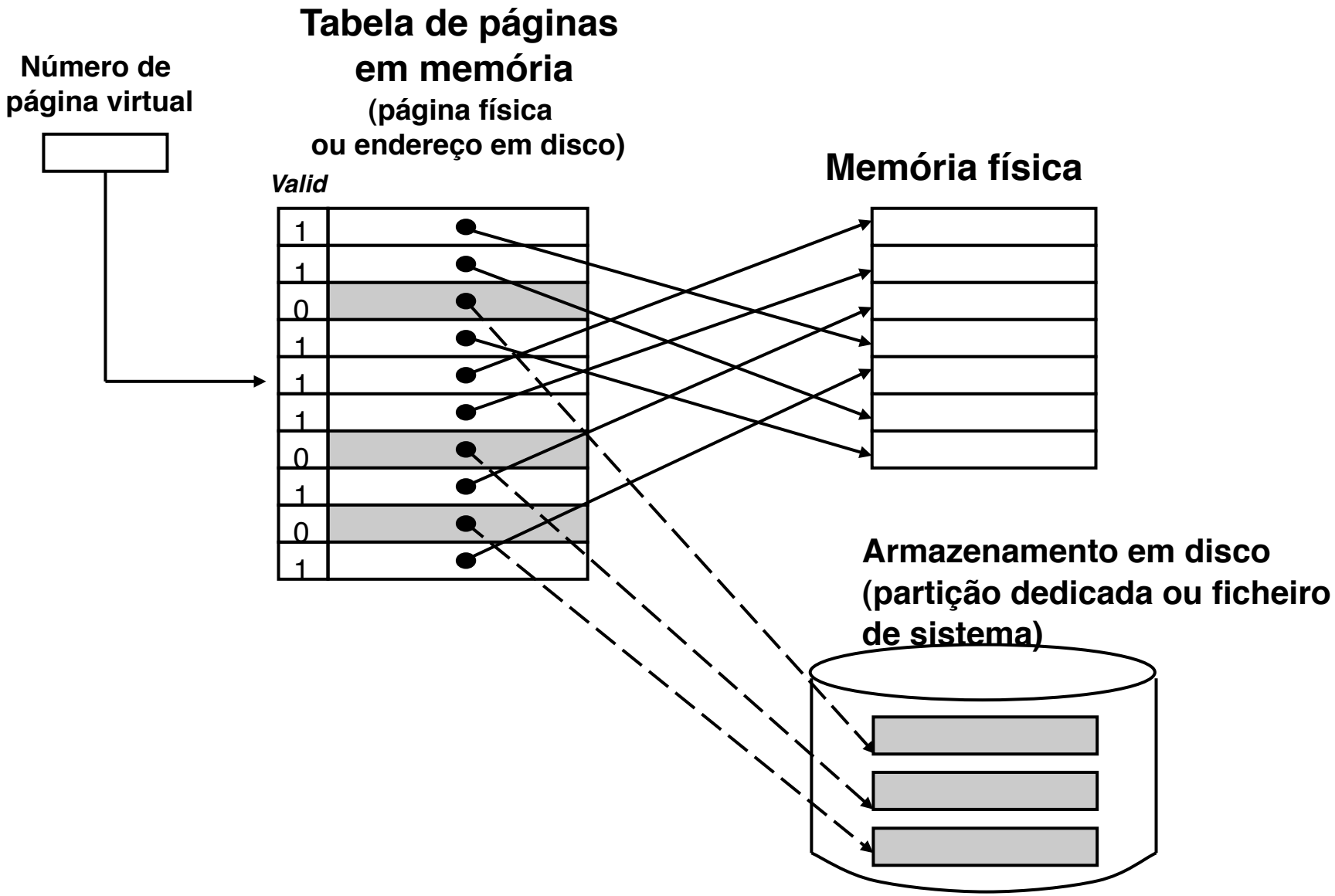


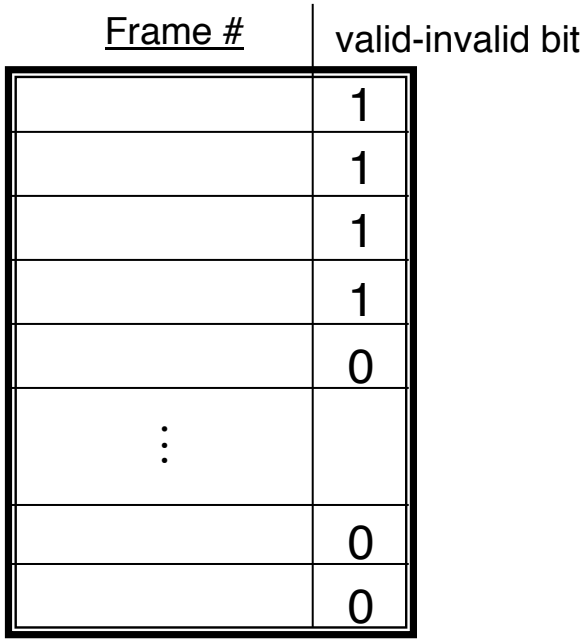
Tabela de páginas



Bit de Validade

- Cada página tem associado um bit de Validade (V)
  (V==1 ⇒ em RAM, V==0 ⇒ em disco)
- Inicialmente V==0 em todas as páginas.
- Na transformação de endereços, se V na entrada da tabela de páginas está a 0 ⇒ page fault (falta de página).

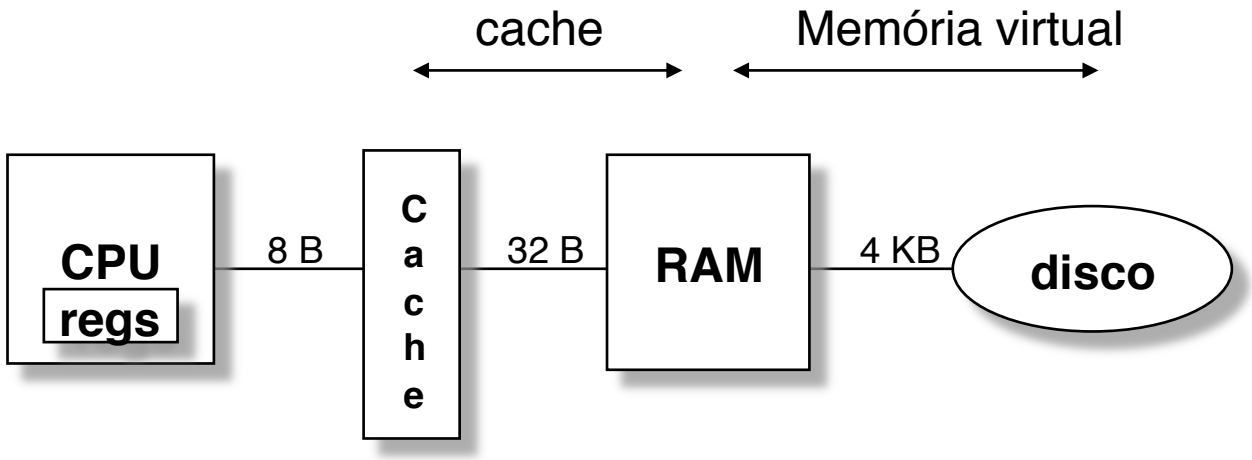
Tabela de páginas



Motivações para a memória virtual

- Usar a RAM como Cache para o Disco
  - ✓ O espaço de endereçamento do processo pode exceder a dimensão da memória. A soma dos espaços de endereçamento dos vários processos pode exceder o tamanho da memória física
- Simplificar a gestão de memória
  - ✓ Múltiplos processos residem em memória central cada um com o seu próprio espaço de endereçamento. Só o código e dados "activos" é que estão em RAM; mais RAM pode ser atribuída se necesário.
  - ✓ Fornece protecção – isto já era conseguido a partir da tabela de páginas mesmo sem memória virtual.

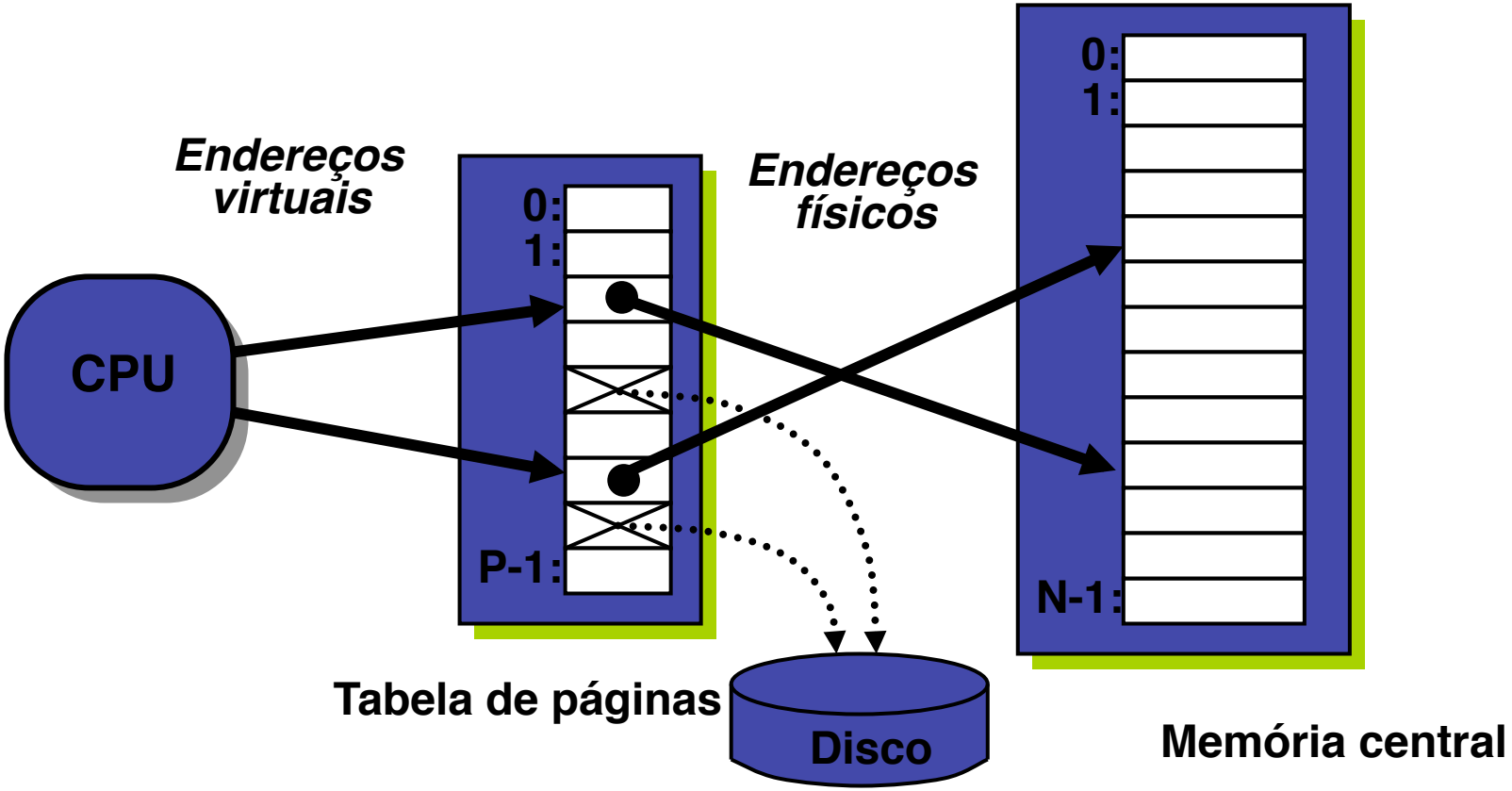
Níveis na hierarquia de memória



	Registos	Cache	RAM	Disco
tamanho:	32 B	32 KB-4MB	128 MB	30 GB
velocidade:	3 ns	6 ns	60 ns	8 ms
$/Mbyte:		$100/MB	$1.25/MB	$0.008/MB
tamanho da linha:	8 B	32 B	4 KB	

Maior, mais lenta, mais barata

Sistemas com memória virtual



Transformação de endereços: Hardware converte *endereços virtuais* em *endereços físicos* através de um tabela gerida pelo SO (*tabela de páginas*)

Paginação a pedido

- A página só é trazido para memória quando é referenciada.
  - ✓ Menos I/O
  - ✓ Menos RAM utilizada
  - ✓ Tempo de resposta menor
  - ✓ Mais programas em RAM

- Página é necessária ⇒ referencia-se; o SO sabe qual dos casos seguintes se trata:
  - ✓ Referência inválida ⇒ abortar a execução
  - ✓ Não-em-memória ⇒ é preciso trazer a página virtual para RAM

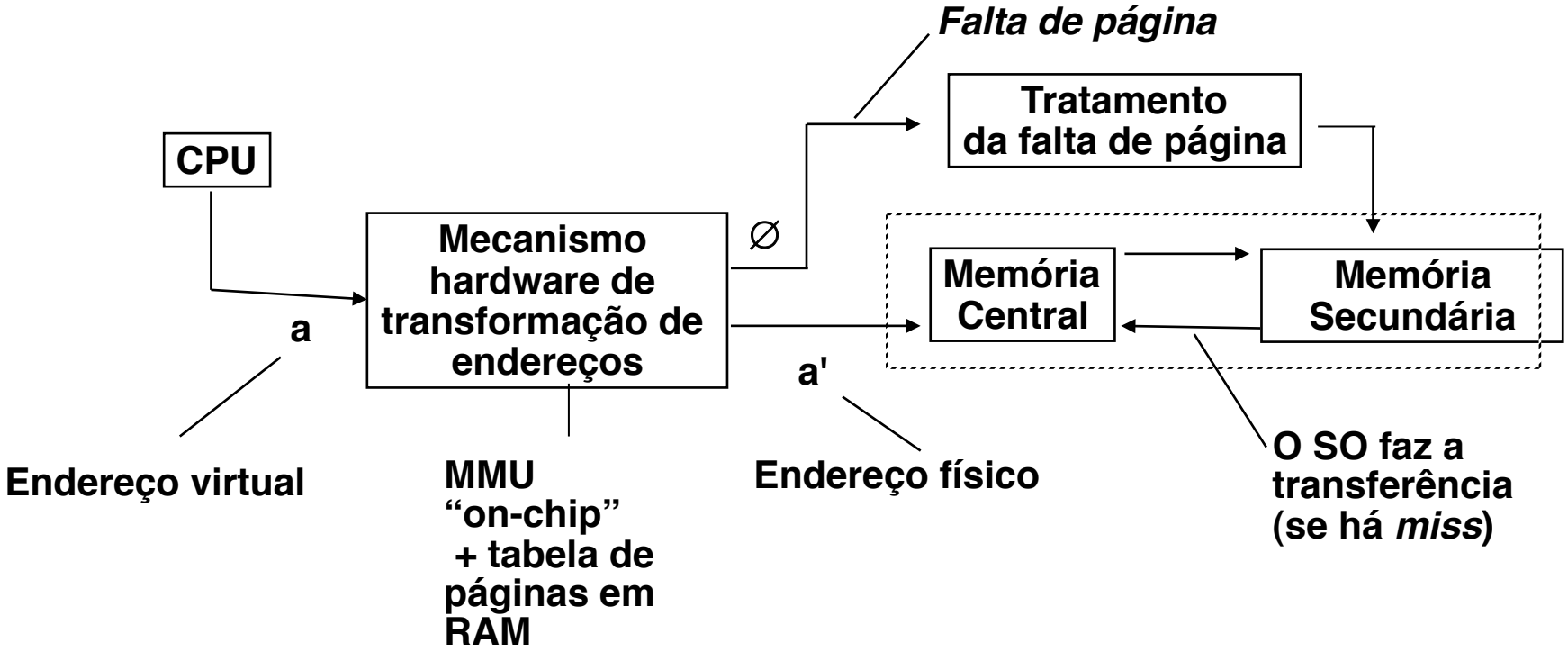
Transformação de endereços

N > M

V = {0, 1, . . . , N–1} espaço de endereçamento virtual
P = {0, 1, . . . , M–1} espaço de endereçamento físico

MAP: V → P U {∅} função de correspondência

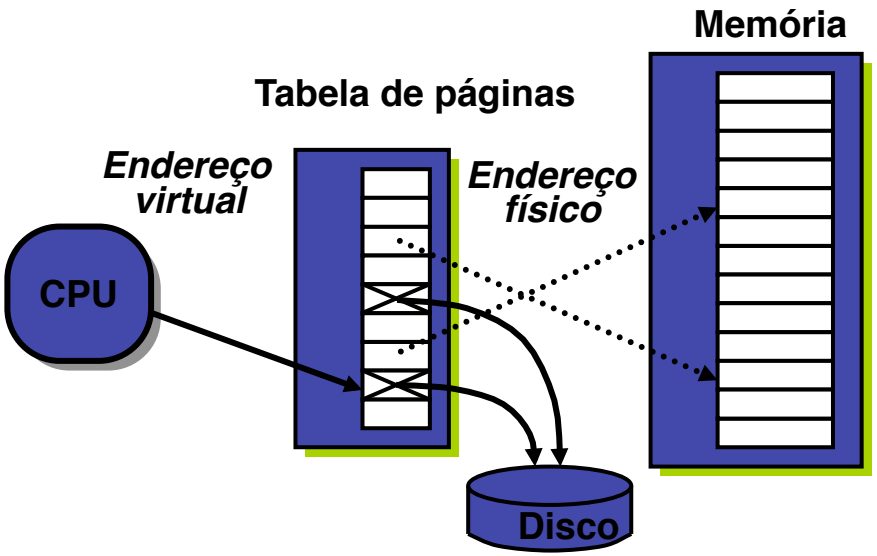
MAP(a) = a' se o dado no endereço a está presente no endereço físico a' em P
= ∅ se o dado no endereço virtual a não está presente em P



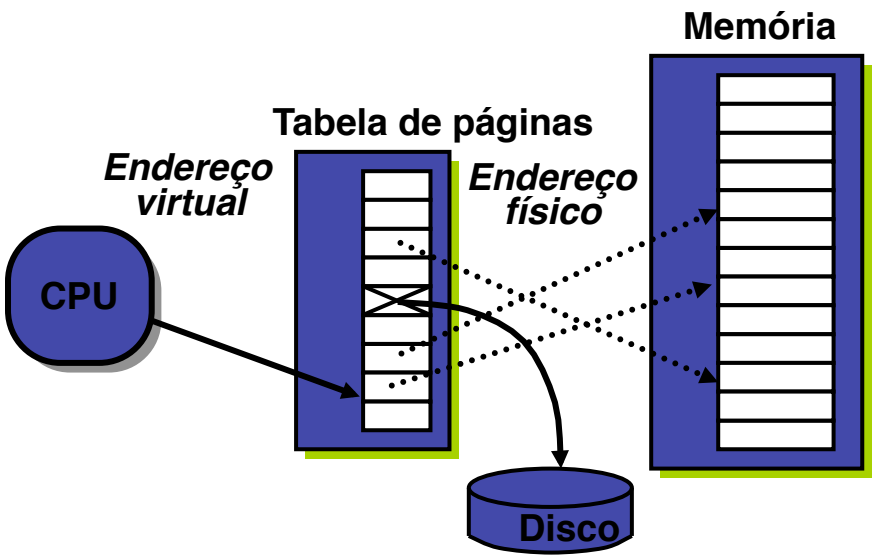
Falta de página

- ✓ Tabela de páginas indica que o conteúdo referenciado pelo endereço virtual não está em RAM
- ✓ É invocado um procedimento de excepção para trazer os dados para memória
  - ✓ o proceso corrente é suspenso, outros podem continuar
  - ✓ o SO tem controlo completo sobre a localização das páginas

Antes das falta

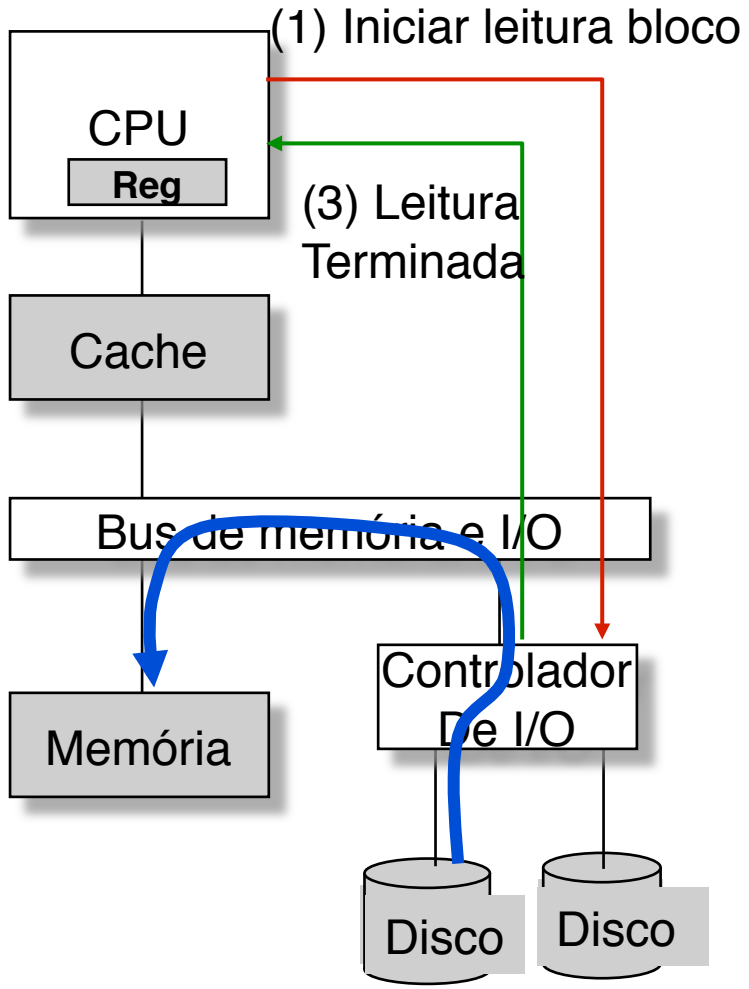


Depois da falta



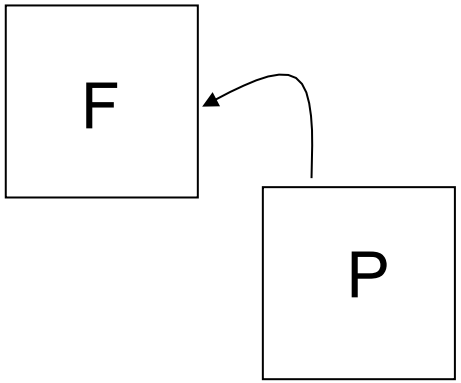
Atendimento da falta de página

- CPU pede ao controlador para ler um bloco de dim. P a partir do end. X e escrever na RAM começando no end. Y
- Ocorre a leitura
  - ✓ Direct Memory Access (DMA)
  - ✓ sob controlo do controlador do disco
- Controlador do disco assinala o fim
  - ✓ Interrupção
  - ✓ SO actualiza a tabeça de páginas
  - ✓ Processo pode continuar



Falta de página(Page Fault)

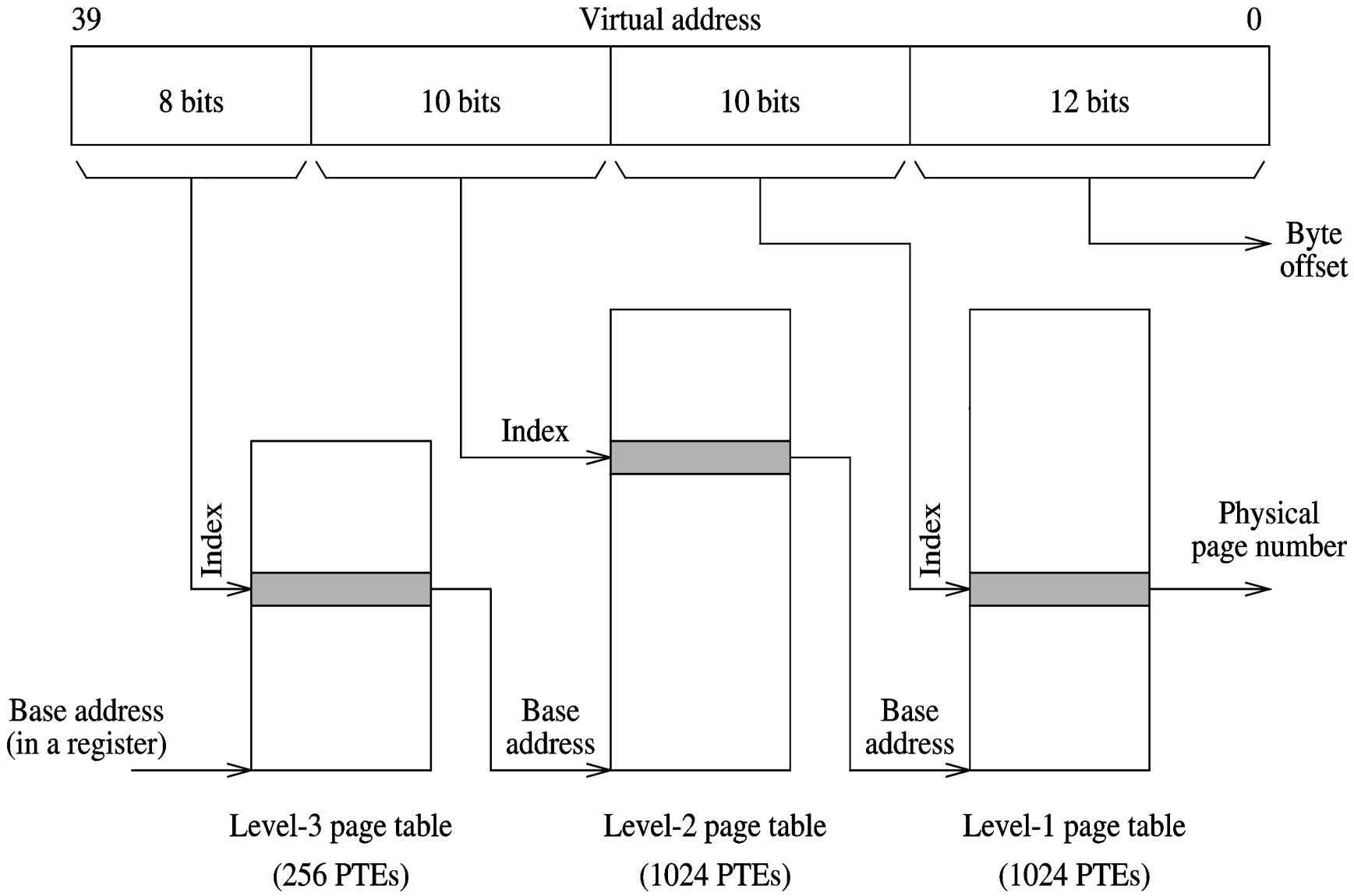
- A primeira referência a uma página P provoca uma excepção (fault) e o OS intervem
- OS olha para a tabela de páginas para decidir se é:
  - ✓ Referência inválida ⇒ aborta o programa.
  - ✓ Falta de página.
- Obtem uma "frame" vazia F.
- Carrega a página nessa "frame".
- Modifica a tabela de páginas
  - ✓ Tab_paginas[P].frame = F
  - ✓ Tab_paginas[P].V = 1.
- Retoma a instrução que provocou a excepção



Suporte da tabela de páginas

- Um espaço de endereçamento virtual grande precisa de tabelas de páginas enormes
- Colocado no espaço de endereçamento virtual do processo (área de sistema)
  - ✓ Uma vez que num dado instante só é necessária uma parte da tabela, pode-se usar uma estratégia hierárquica
    - ✓ Dividir a tabela de páginas em páginas
    - ✓ Trazer as tabelas à medida que são precisas
    - ✓ Um primeiro nível de tabelas pode apontar para tabelas de um segundo nível

Suporte da tabela de páginas(2)



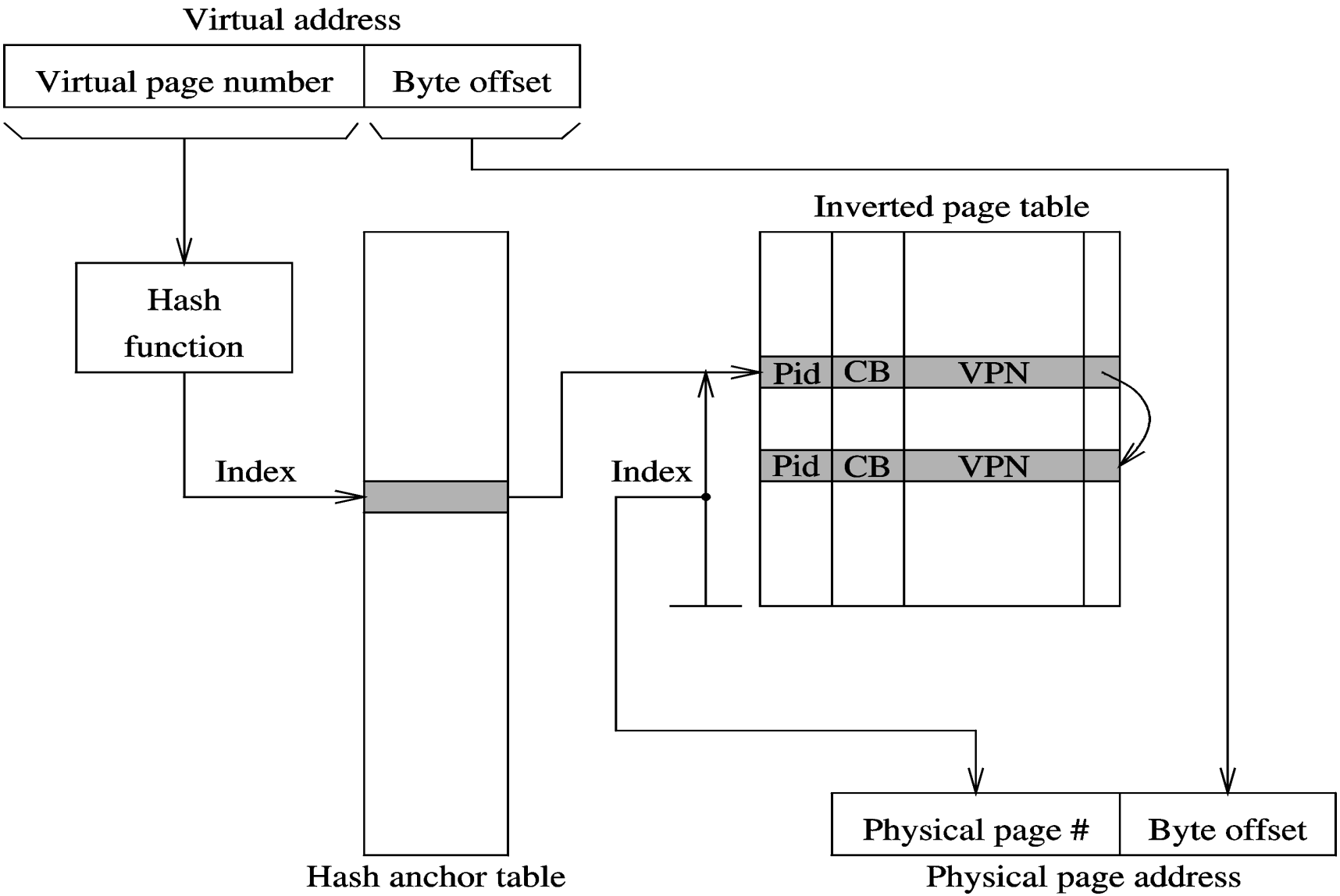
Suporte da tabela de páginas (3)

- A estrutura hierárquica de tabelas de páginas é conhecida por **tabela de páginas directa** (*forward-mapped page table)*
  - ✓ Tradução do endereço virtual em físico parte do nº página virtual para saber o nº de página físico
  - ✓ Um acesso à memória por cada nível (claro que há o TLB)
- Exemplo
  - ✓ Pentium (IA-32)
    - ✓ Hierarquia de dois níveis
    - ✓ Detalhes à frente

Tabela de páginas invertida

- As tabelas de páginas directas têm o inconveniente de o número de entradas da tabela ser muito grande
  - ✓ Problema agrava-se nos CPUs de 64-bit
  - ✓ Porquê?
    - ✓ O nº de página virtual é usado como índice
- Para reduzir o número de entradas
  - ✓ O nº da página física é usada como índice
    - ✓ A tabela cresce com o aumento da memória física
- Uma só tabela de páginas global (*system-wide*)
  - ✓ Ao nº da página virtual (VPN) é aplicada uma função (hashed) para gerar o índice na index tabela
    - ✓ É preciso tratar as situações em que números de páginas virtuais diferentes conduzem à mesma entrada na tabela (*colisões*)

Tabela de páginas invertida



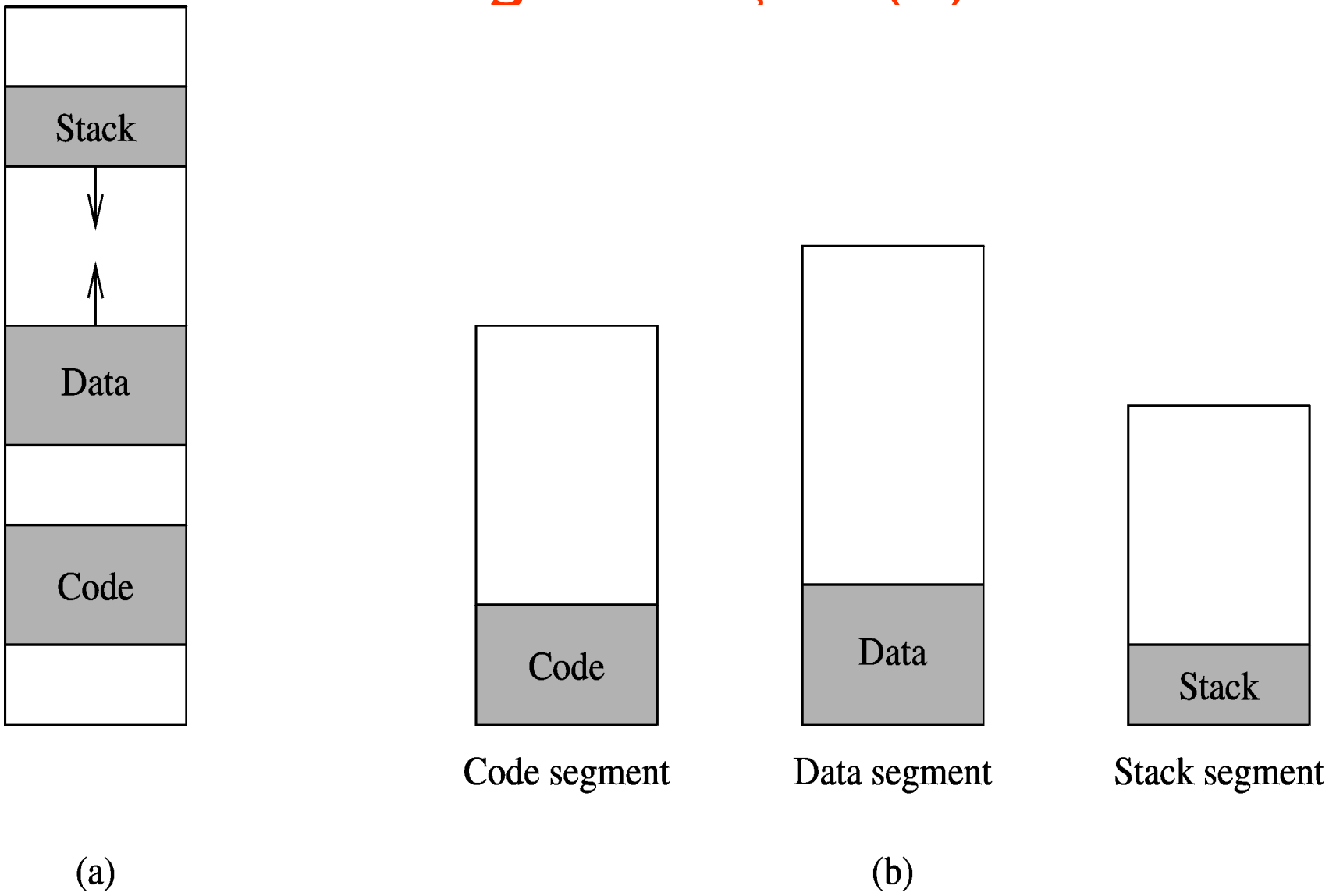
Espaço virtual organizado por segmentos

- Até agora temos admitido que o espaço de endereçamento virtual é linear (endereços de 0 a MAX -1)
  - ✓ Uma alternativa é considerar que o espaço de emdereçamento de um processo está dividido em várias partes
- Cada processo tem vários espaços de endereçamento virtuais separados
  - ✓ Cada espaço é chamado um ***segmento***
  - ✓ Exemplo: Pentium
    - ✓ Os segmentos podem ter 4 GB
- Os endereços têm duas partes
  - ✓ Número do segmento
  - ✓ Deslocamento dentro do segmento

Segmentação (2)

- O Pentium e outros CPUs suportam memória virtual segmentada
  - ✓ A paginação não é vista pelo programador
    - ✓ A segmentação é
      - ✓ Na programação em assembly do Pentium é visível ao programador que há três espaços distintos
        - ✓ dados, código, e pilha (podem ser 3 segmentos …)
- A segmentação oferece algumas vantagens importantes
  - ✓ Protecção
    - ✓ Pode ser especificado ao nível do segmento
  - ✓ Múltiplos espaços de endereçamento
  - ✓ Partilha
    - ✓ Os segmentos podem ser partilhados entre processos

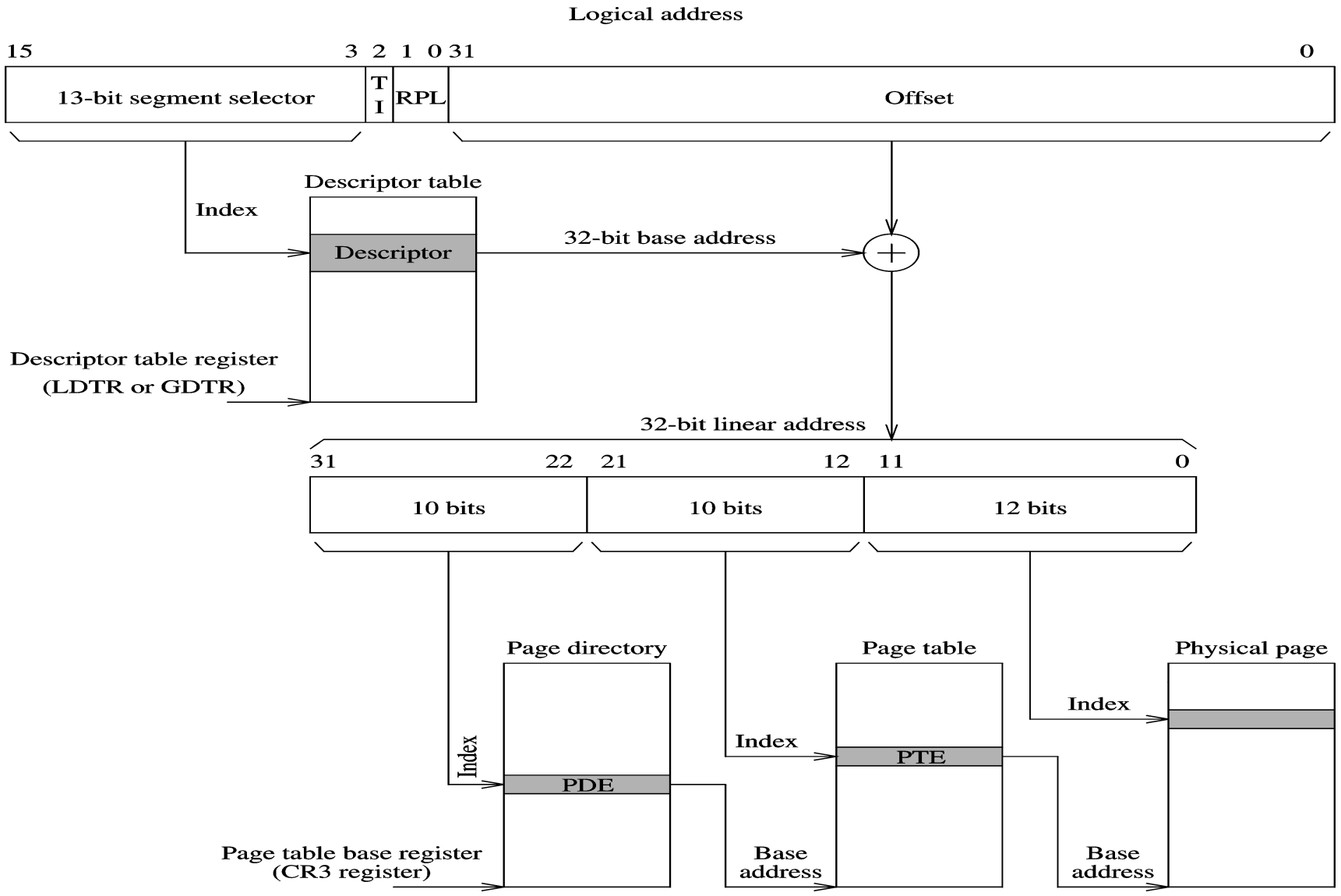
Segmentação (3)



Utilização conjunta de segmentação e paginação

- Pentium
  - ✓ Suporta paginação ou segmentação
    - ✓ Paginação pode ser desligada
    - ✓ Segmentação pode ser desligado
  - ✓ A segmentação transforma um endereço virtual de 48 bits (16 bits para o número do segmento + 32 bits para o deslocamento) num endereço linear com 32 bits
    - ✓ Se a paginação está ligada
      - ✓ Traduz o endereço linear de 32 bits (20 bits para o nº da página e 12 para o deslocamento num endereço físico de 32 bits
    - ✓ Se a paginação está ligada
      - ✓ O endereço linear é tratado como um endereço físico

Pentium - conversão de endereços virtuais em endereços físicos



Pentium - conversão de endereços virtuais em endereços físicos (2)