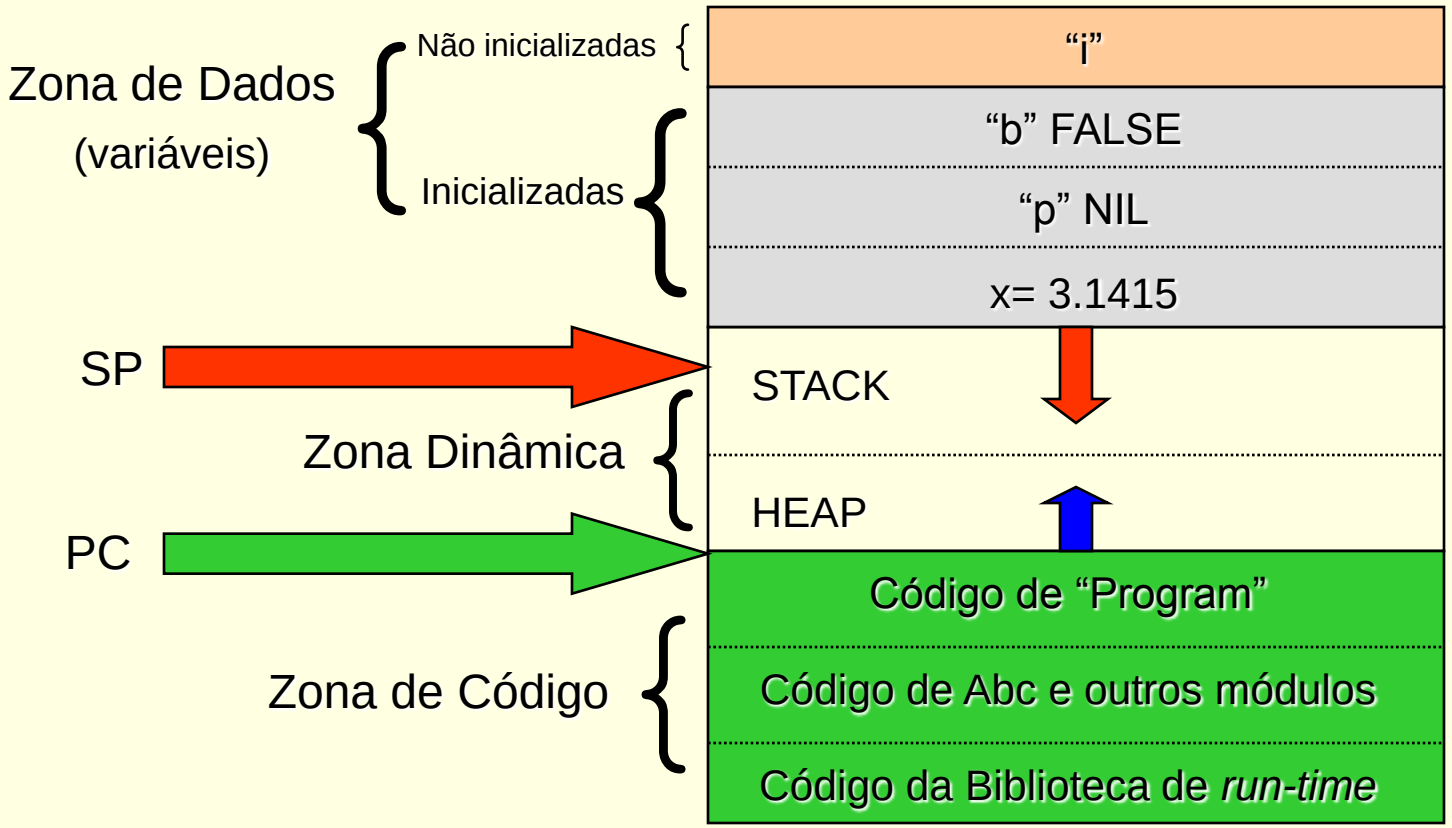
Fundamentos de Sistemas de Operação

Unix Windows NT Netware MacOS DOS/VS Vax/VMS Linux Solaris HP/UX AIX Mach Chorus

Parte II: Gestão de Memória

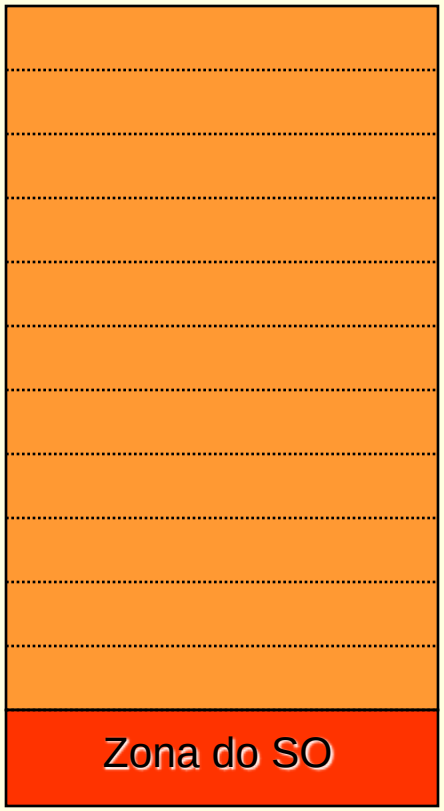
d) Paginação

Como colocar em RAM múltiplos Processos?



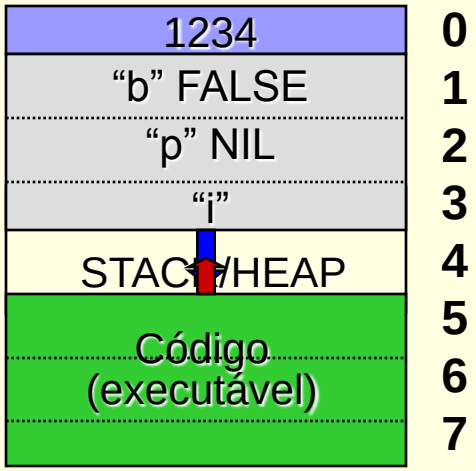
Paginação: I

- Frames (quadros, molduras):
  - Divide-se a memória física em blocos de tamanho fixo chamados páginas físicas (ou frames) de dimensão 2^f, tipicamente entre 512 bytes e 8192 bytes.



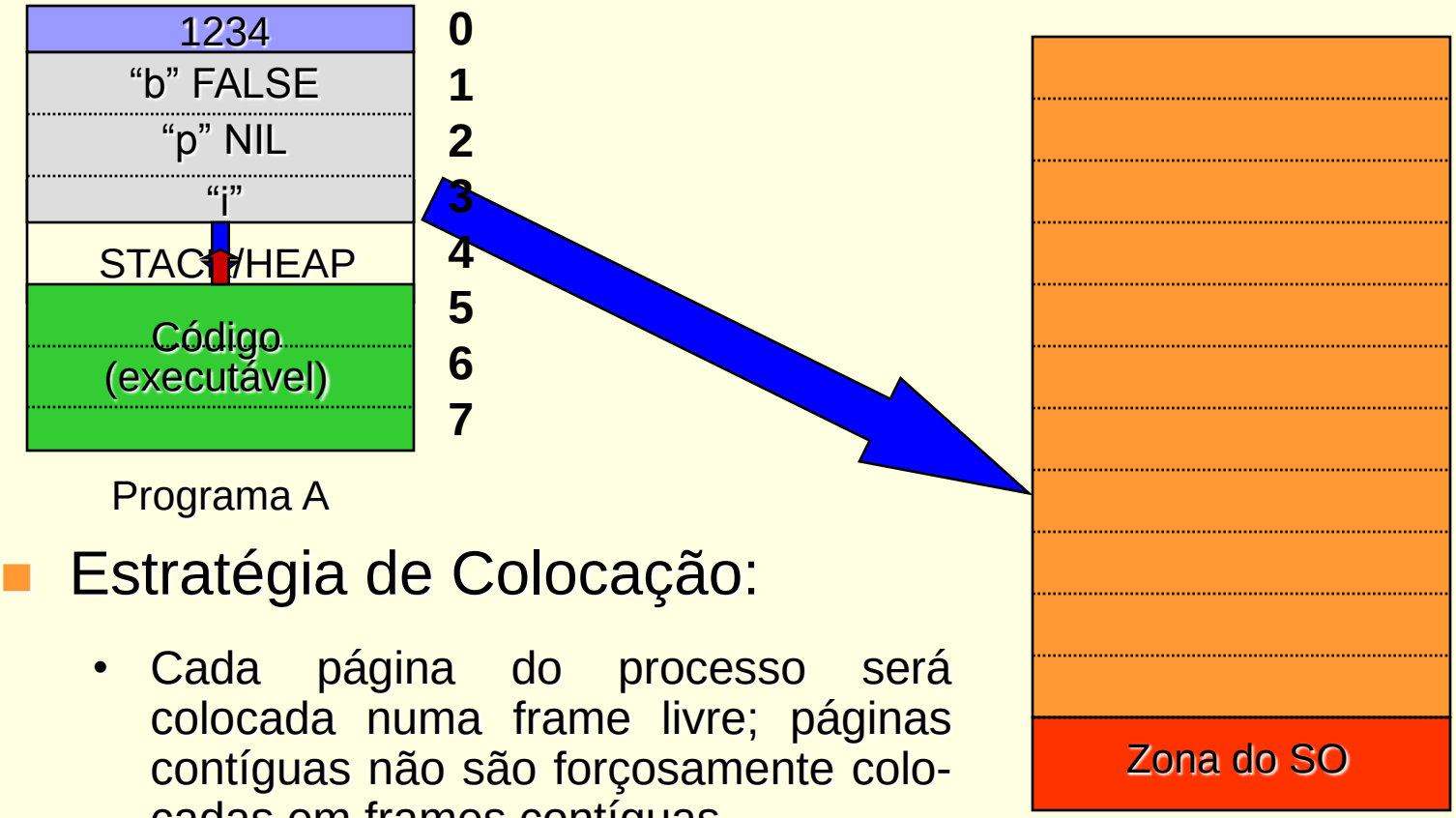
Paginação: II

- Páginas:
  - Divide-se o espaço de endereçamento do processo em blocos de tamanho fixo chamados páginas de dimensão igual à das frames.
  - Notar que a cada região (código, variáveis, constantes, *stack*,…) pode ser associado um conjunto de bits de protecção (rwx) que serão usados para validar os acessos às páginas dessa região.



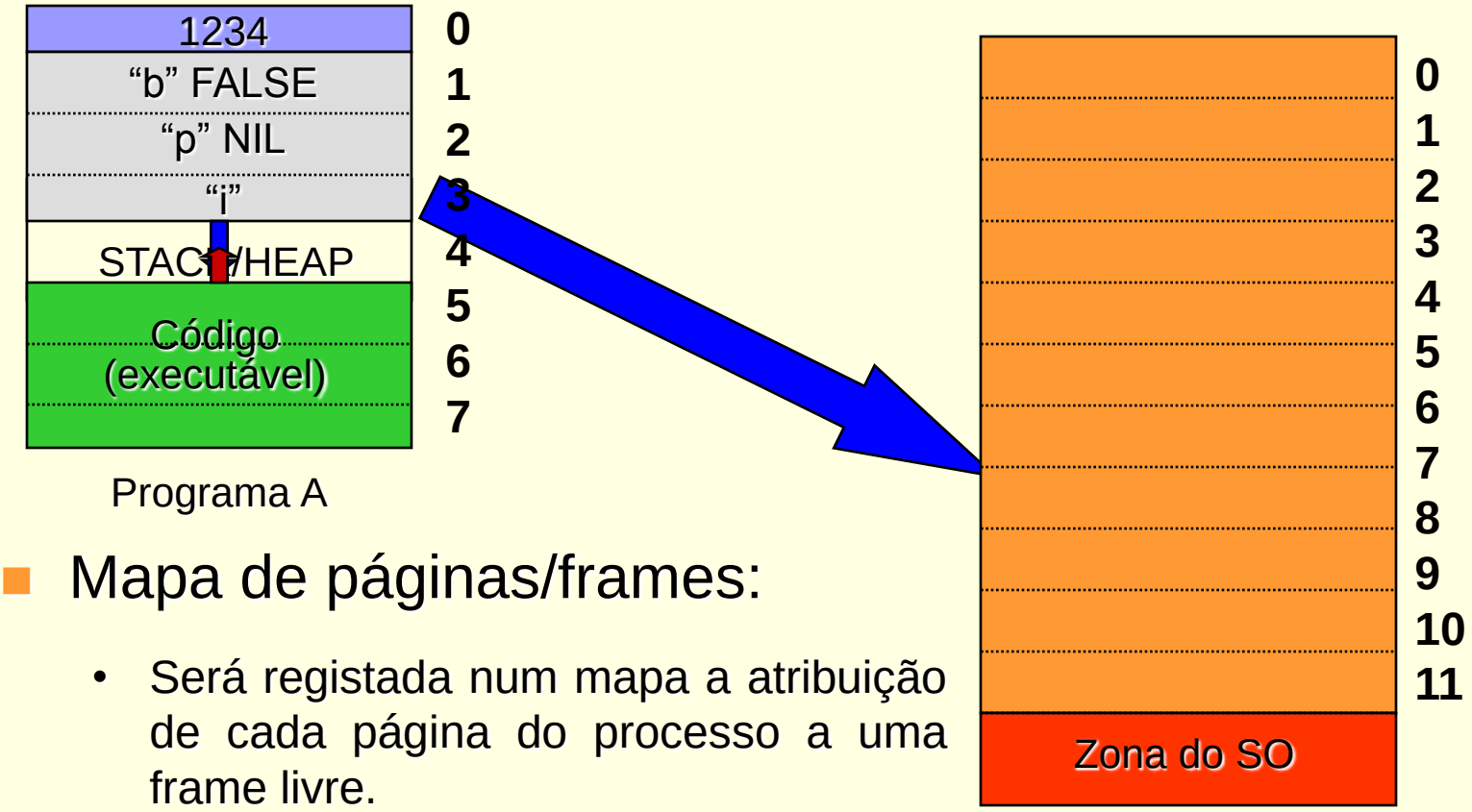
Programa A

Paginação: III



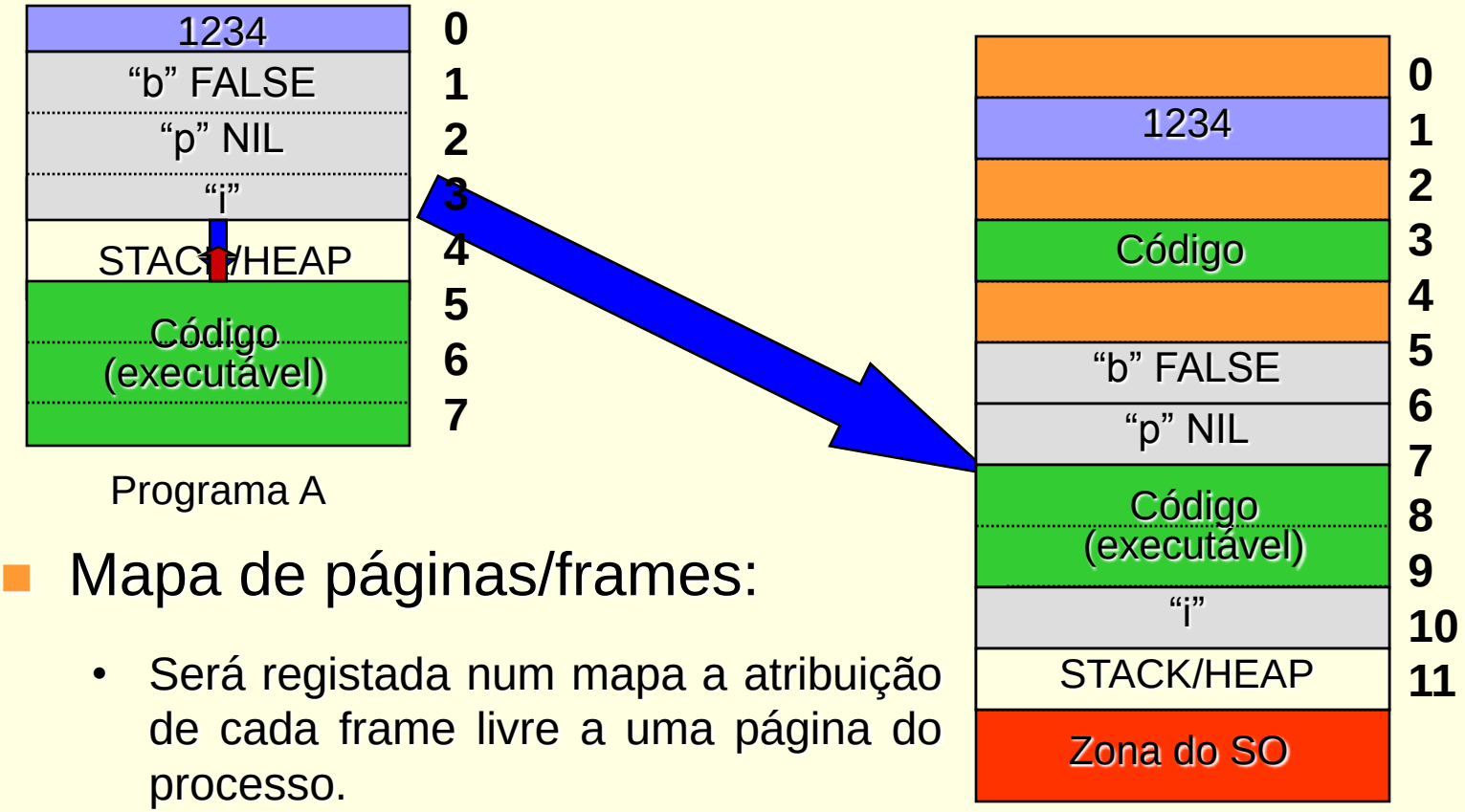
- Estratégia de Colocação:
  - Cada página do processo será colocada numa frame livre; páginas contíguas não são forçosamente colocadas em frames contíguas.

Paginação: IV



- Mapa de páginas/frames:
  - Será registada num mapa a atribuição de cada página do processo a uma frame livre.

Paginação: V

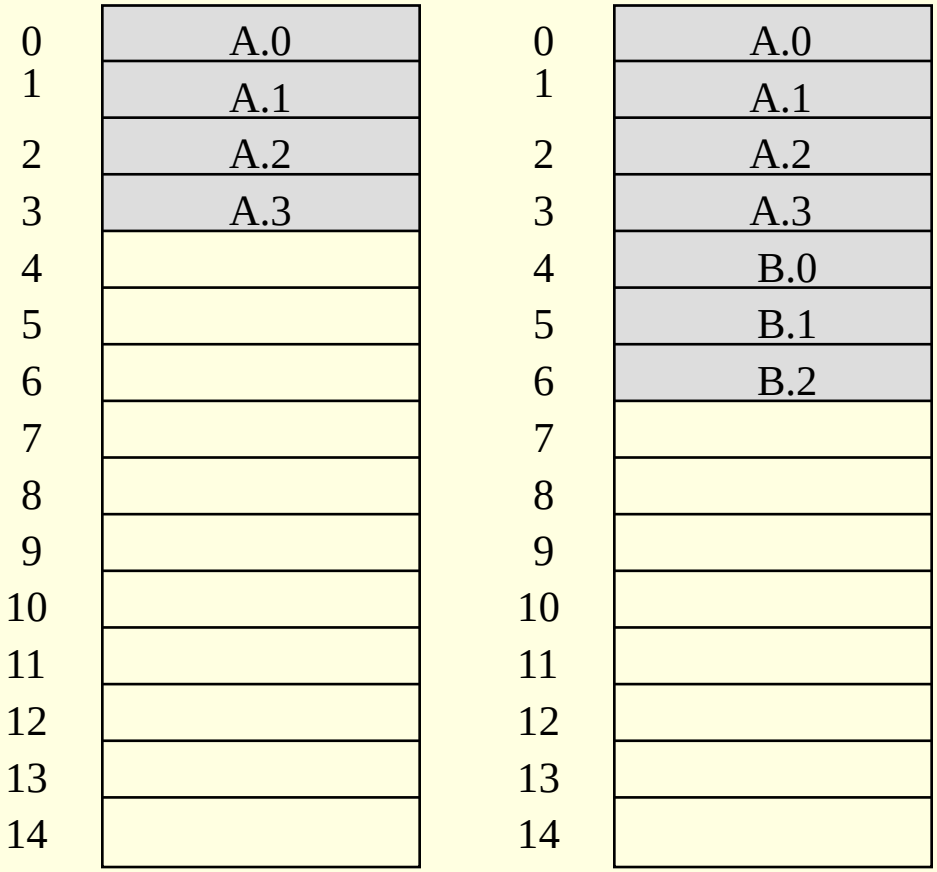


- Mapa de páginas/frames:
  - Será registada num mapa a atribuição de cada frame livre a uma página do processo.

Paginação - Exemplo com 4 processos: I

- *Carregando os processos:*

  A e B

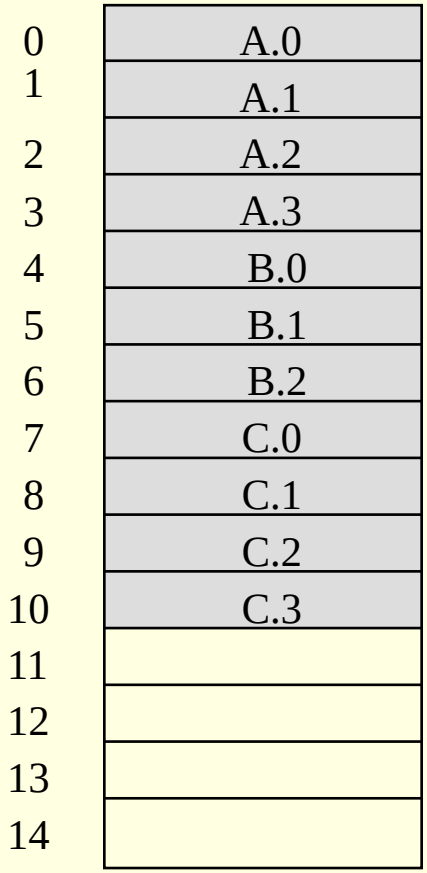


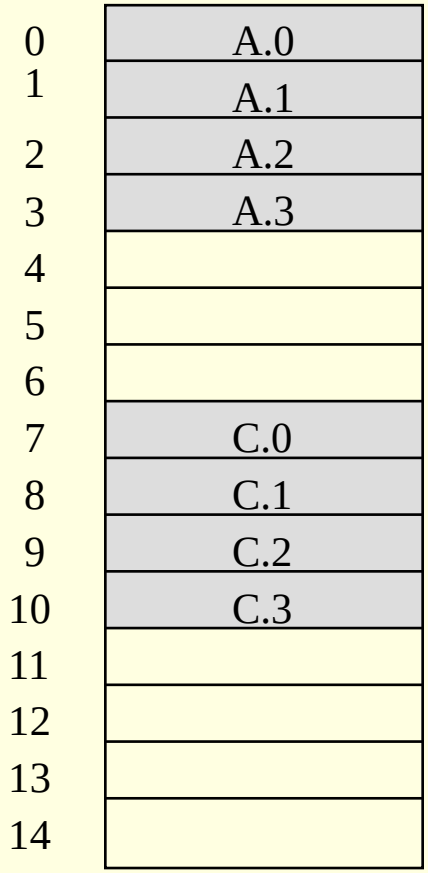
Paginação - Exemplo com 4 processos: II

- *Carrega-se o processo*

  C,

- *Termina B*





Paginação - Exemplo com 4 processos: III

- *Carrega-se o processo*

  D

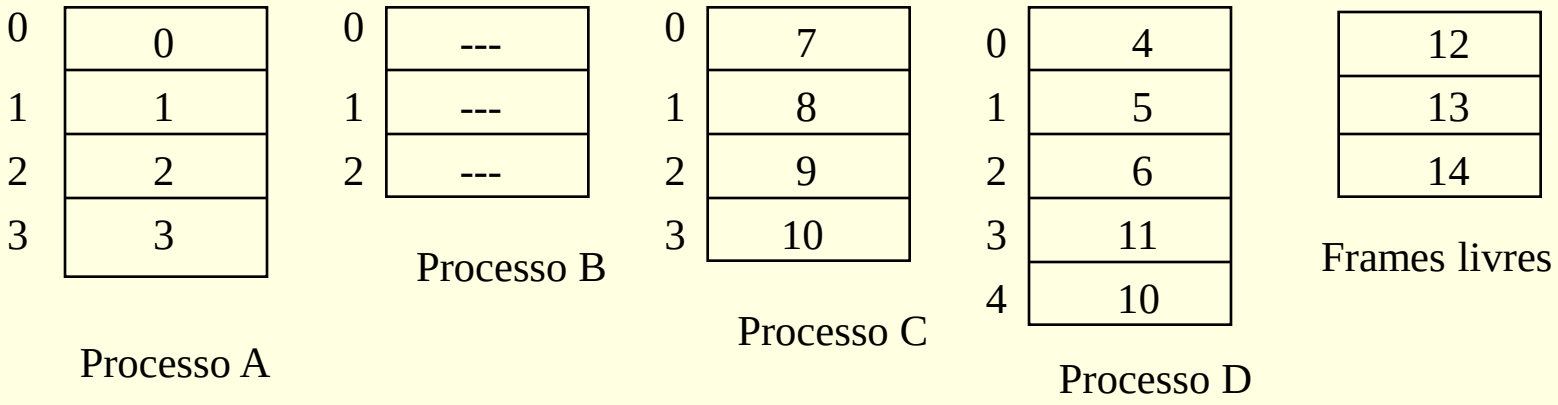
- *D é colocado de forma não--contígua...*

- *e D* ***partilha*** *uma pagina juntamente com C...*

0	A.0
1	A.1
2	A.2
3	A.3
4	D.0
5	D.1
6	D.2
7	C.0
8	C.1
9	C.2
10	C.3/D.4
11	D.3
12	
13	
14	

Paginação - Exemplo com 4 processos: IV

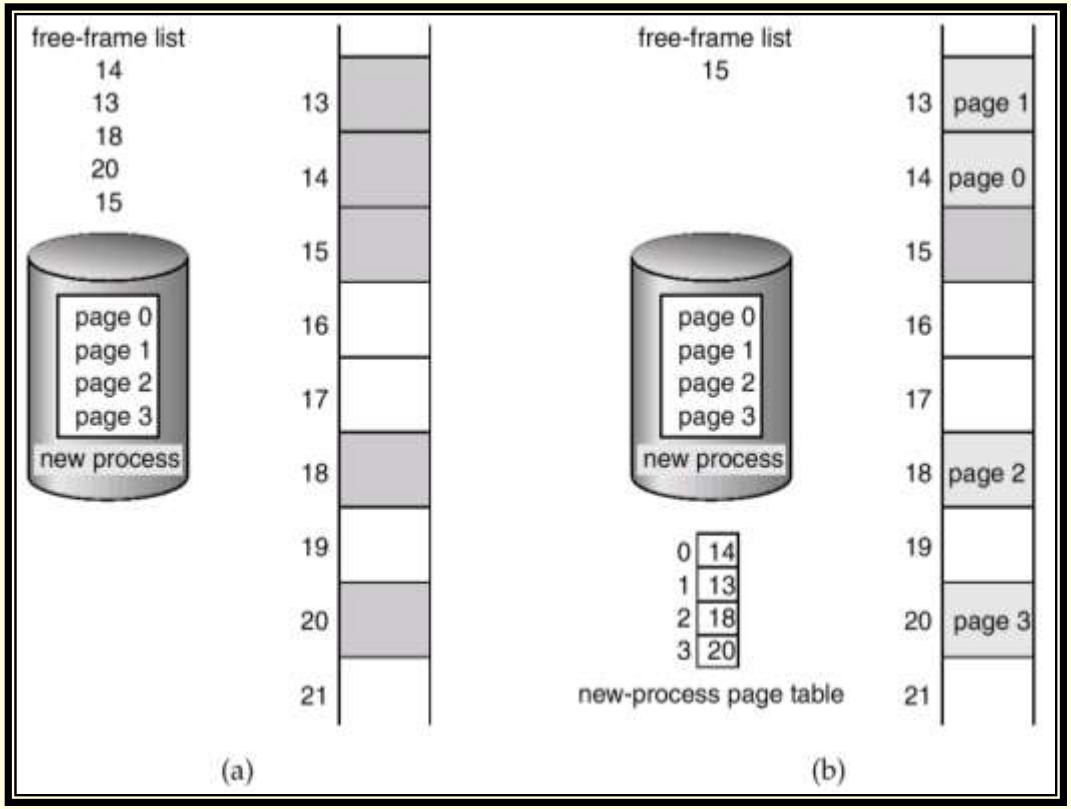
- As tabelas de páginas criadas pelo SO para cada um dos processos são:



- Notar: partilha da *frame* 10 pelos processos C e D

Paginação

- O Processo, as Páginas, as Frames e a Tabela de Páginas

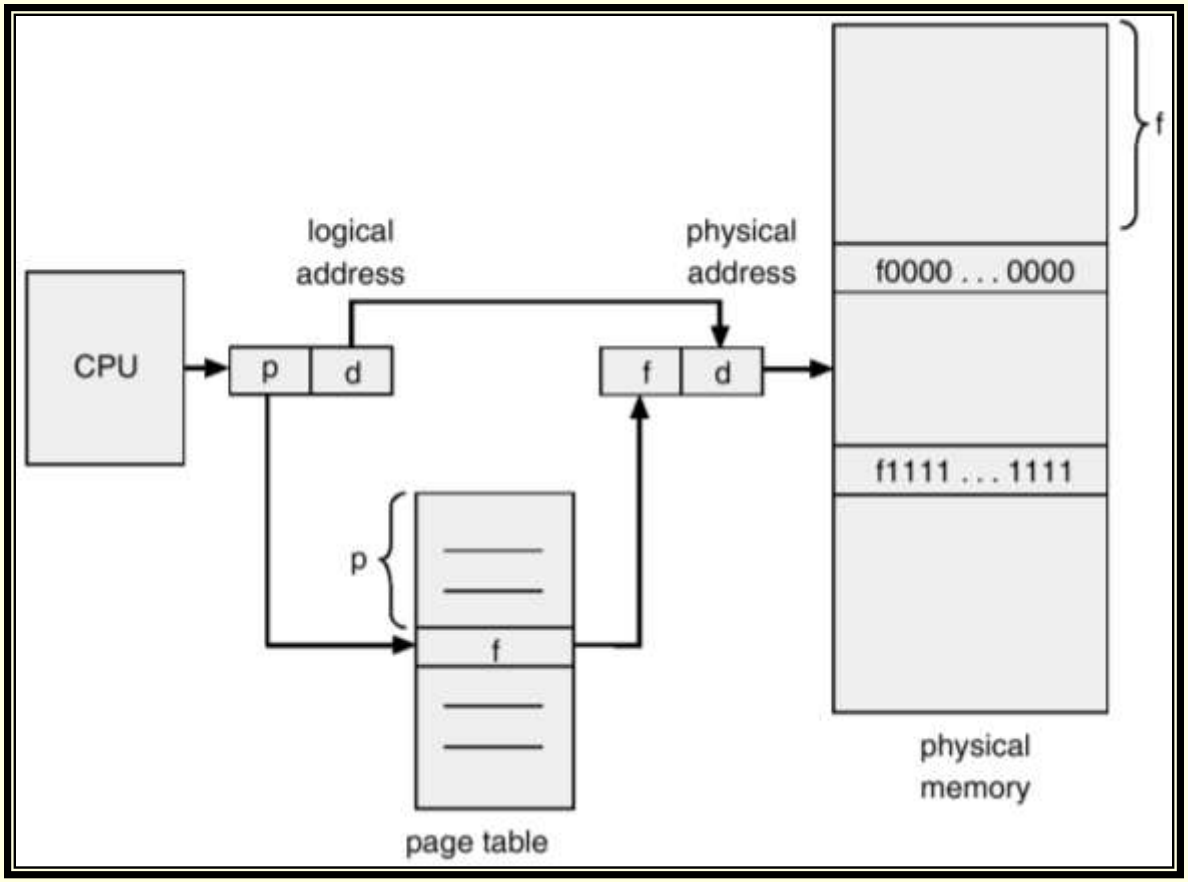


Fonte: OSC

Paginação – Tradução de Endereços: I

- Estrutura do endereço lógico (ou virtual):
  - 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.

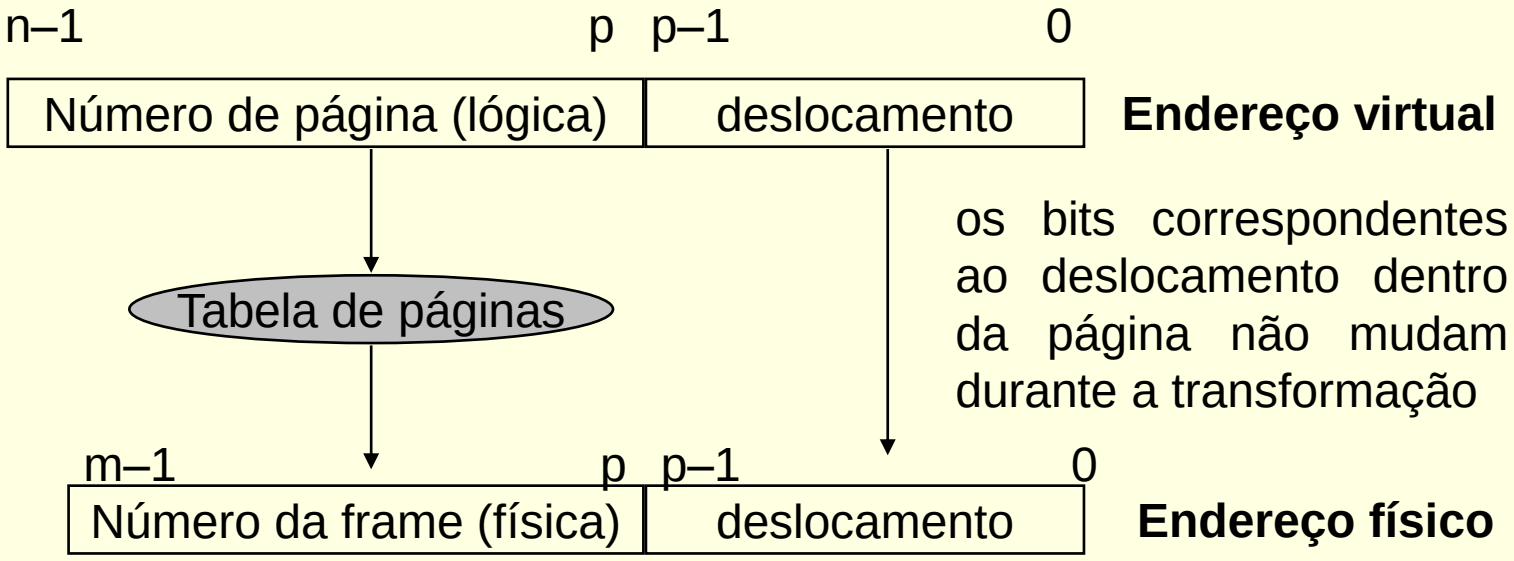
Paginação – Tradução de Endereços: II



Fonte: OSC

Paginação – Tradução de Endereços: III

- P = 2^p, tamanho da página (bytes).
- N = 2^n, endereço virtual de n bits.
- M = 2^m, endereço físico de m bits



Paginação – Protecção de regiões do processo

- Cada página tem associado um conjunto de bits de "protecção" (rwx) que caracterizam o tipo de acesso permitido; por exemplo,
  - As páginas de código têm o atributo x (execute); as páginas de variáveis têm os atributos rw (ler/escrever); as páginas de constantes têm o atributo r (ler); etc.

- Na transformação de endereços, a intenção (r, ou w, ou x) da instrução que "tenta" aceder a um dado endereço, é comparada com os atributos de protecção da página; por exemplo,
  - Se se tenta um JMP para uma página com o atributo x, ele é permitido, mas se for para uma página "de dados" somente com atributos rw, a execução é abortada...

Tabela de páginas

Frame #	bits
	001
	001
	110
	100
	000
…	
	000
	111

Paginação

- Hardware e Estruturas de dados para a GM:
  - Lista ligada (?) para gerir as frames livres, ordenada (?) por dimensão da região (frames livres contíguas) ou simplesmente um bitmap (?).
  - Tabela de páginas (por processo) em RAM.
  - Um novo registo no CPU, o *Page-table base register* (PTBR), aponta para a base da tabela de páginas (do processo em execução).
  - Um novo registo no CPU, o *Page-table length register* (PTLR) indica o tamanho da tabela de páginas (do processo em execução).

Paginação

- Algoritmos para a GM:
  - Estratégia de colocação: encontrar as *frames* livres necessárias.
  - Carregar cada página do processo numa das *frames* livres seleccionadas; associar a cada página de uma dada região do processo (código, variáveis, constantes, *stack*,…) os bits de protecção adequados.
  - Actualizar Tabela de Páginas, incluindo os bits de protecção, e as cópias dos registos PTBR, PTLR (estas são guardadas no PCB do processo).

Paginação: Vantagens

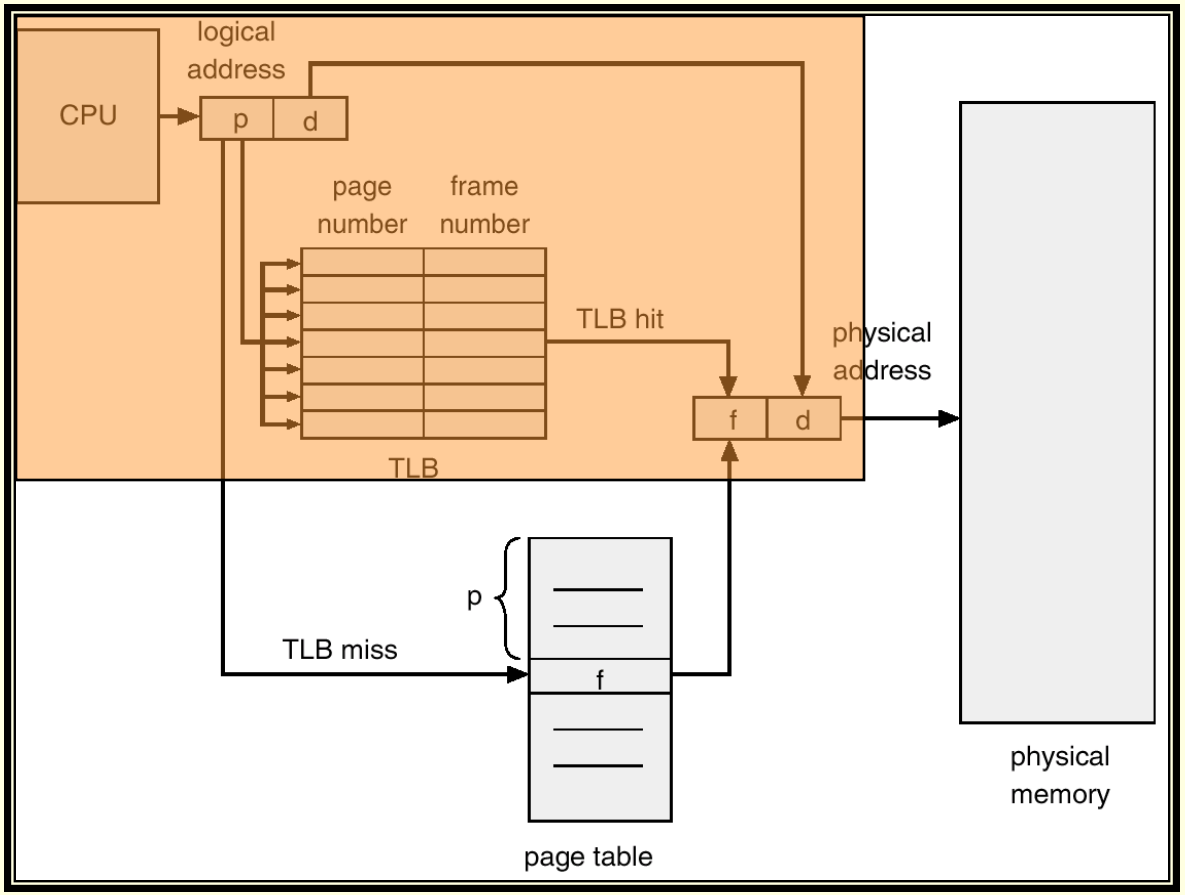
- Melhor Gestão da Memória:
  - Apenas a última página de uma dada região do processo (código, variáveis, constantes, *stack*,…) pode sofrer de "fragmentação" interna.
  - A utilização de memória partilhada entre diversos processos (mesmo programa ou diferentes programas usando as mesmas bibliotecas partilhadas) permite diminuir a ocupação da RAM.
  - O mecanismo de protecção com atributos rwx permite uma detecção de violações que doutra forma poderiam passar despercebidas (facilita, portanto, o *debugging* do programa)

Paginação: Desvantagens

- *Overhead*:
  - Cada acesso a dados ou código obriga a dois acessos à memória (ver pág. 14) - um para tabela de páginas e outro para acesso ao dado / instrução.
  - O referido *overhead* pode ser resolvido/minimizado usando um *hardware* especial de consulta rápida, chamado *translation look-aside buffer* (TLB), que se baseia numa memória associativa, também designada interrogável pelo conteúdo. O TLB mantém "no circuito" as traduções mais usadas (funciona como *cache*).
  - Seja como for, há sempre um atraso nos acessos a memória quando comparado com uma arquitectura com uma ATU mais simples (RB/RL, por exemplo).

Paginação – Tradução de Endereços com TLB

Nos processadores modernos a ATU/ MMU e o TLB estão no mesmo *chip* do CPU, onde está ainda a *cache* de primeiro nível…



Paginação: Tempo de acesso efectivo

- *Tempo de acesso efectivo*:
  - Pesquisa associativa = ε unidades de tempo.
  - O Tempo de acesso à RAM é T unidades de tempo.
  - Hit rate – percentagem das vezes em que o número da frame está num registo do TLB.
  - Hit Rate = α
  - Tempo de acesso efectivo (TAE)

    $TAE = (T + \varepsilon)\ \alpha + (2T + \varepsilon)(1 - \alpha) = (2 - \alpha)T + \varepsilon$