Paginação

Sistemas Operacionais

Prof. Pedro Ramos
pramos.costar@gmail.com

Pontifícia Universidade Católica de Minas Gerais
ICEI - Departamento de Ciência da Computação

PAGINAÇÃO: MOTIVAÇÃO E FUNCIONALIDADES

- Regra 90/10: Processos gastam 90% do tempo acessando 10% do seu espaço na memória.
  => Mantenha apenas as partes de um processo na memória que estão sendo realmente utilizadas.

- Páginas simplificam o problema de ajuste de fragmentos.

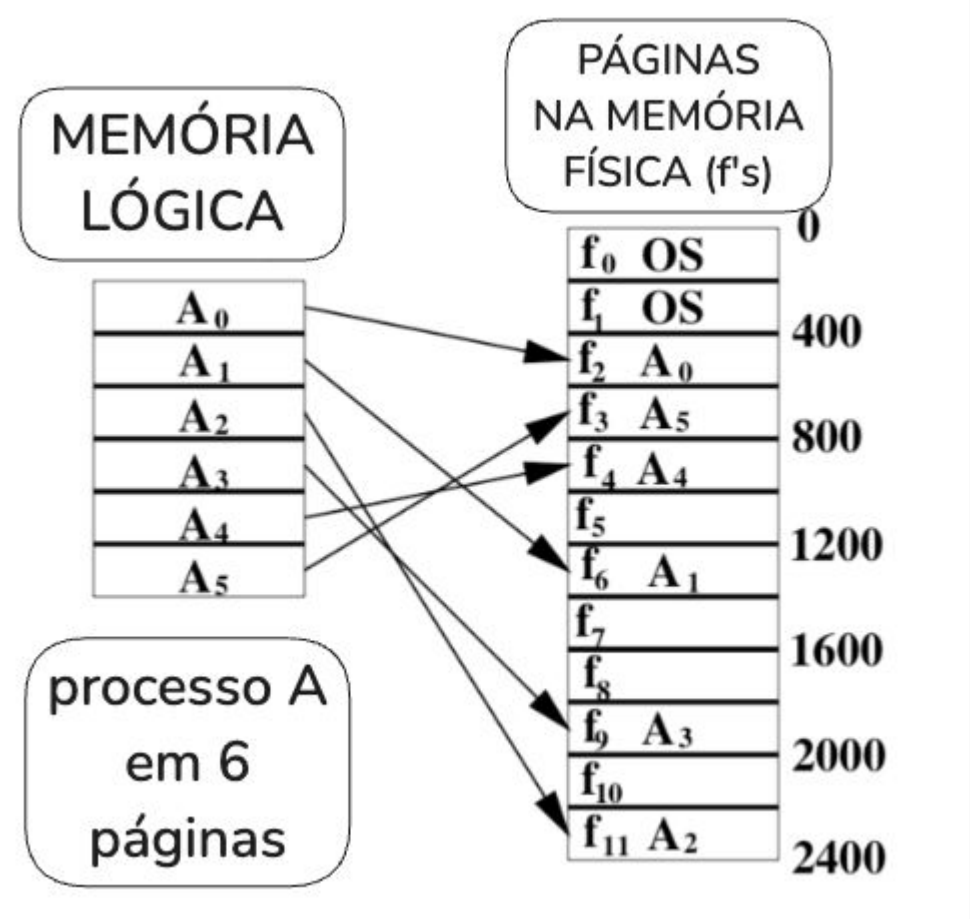
- A memória lógica (virtual) do processo é contígua, mas **as páginas não precisam ser alocadas contiguamente na memória**.

- Dividindo a memória em páginas de tamanho fixo, podemos ***eliminar a fragmentação externa***.

- A paginação não elimina a fragmentação interna (*1/2 página por processo, em média*).

PAGINAÇÃO - EXEMPLO

Qual o papel do SO?

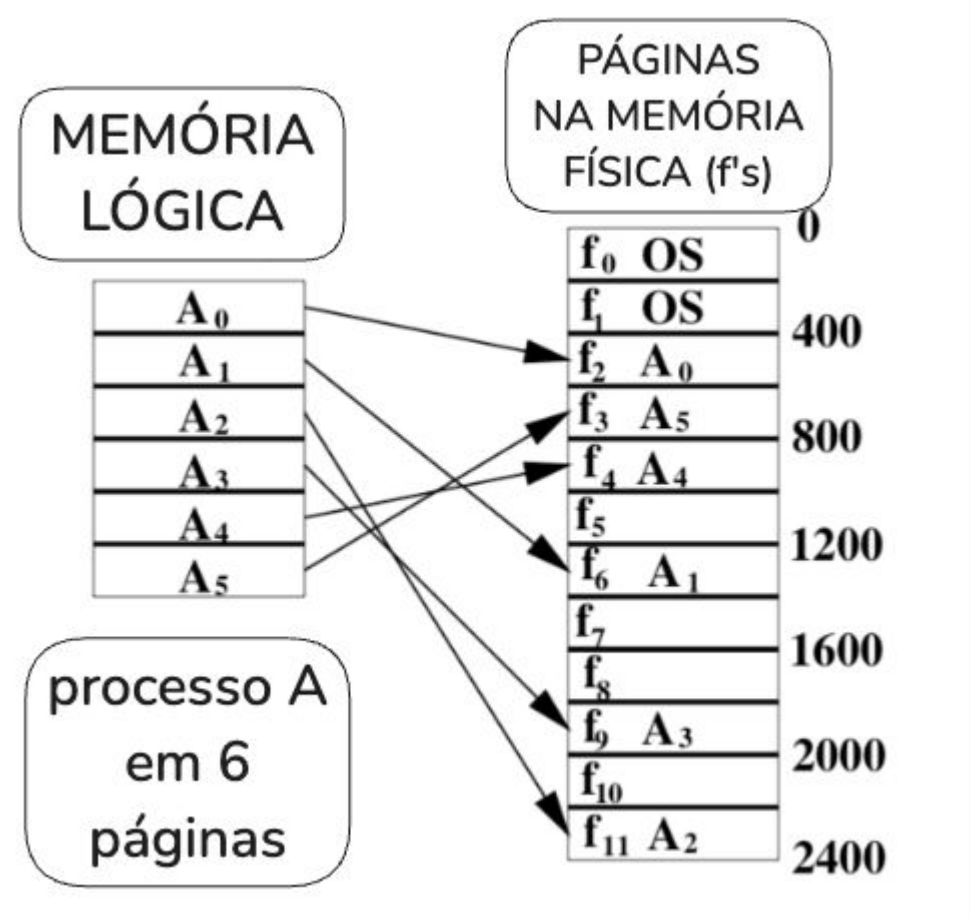


PAGINAÇÃO - EXEMPLO

Qual o papel do SO?

GERENCIAR A TABELA DE PÁGINAS.

TRADUZIR ENDEREÇOS DAS PÁGINAS PARA ENDEREÇOS FÍSICOS.



HARDWARE DE PAGINAÇÃO

COMO ENCONTRAR ENDEREÇOS QUANDO AS PÁGINAS NÃO ESTÃO ALOCADAS CONTIGUAMENTE EM MEMÓRIA?

Próxima aula: **Paginação**

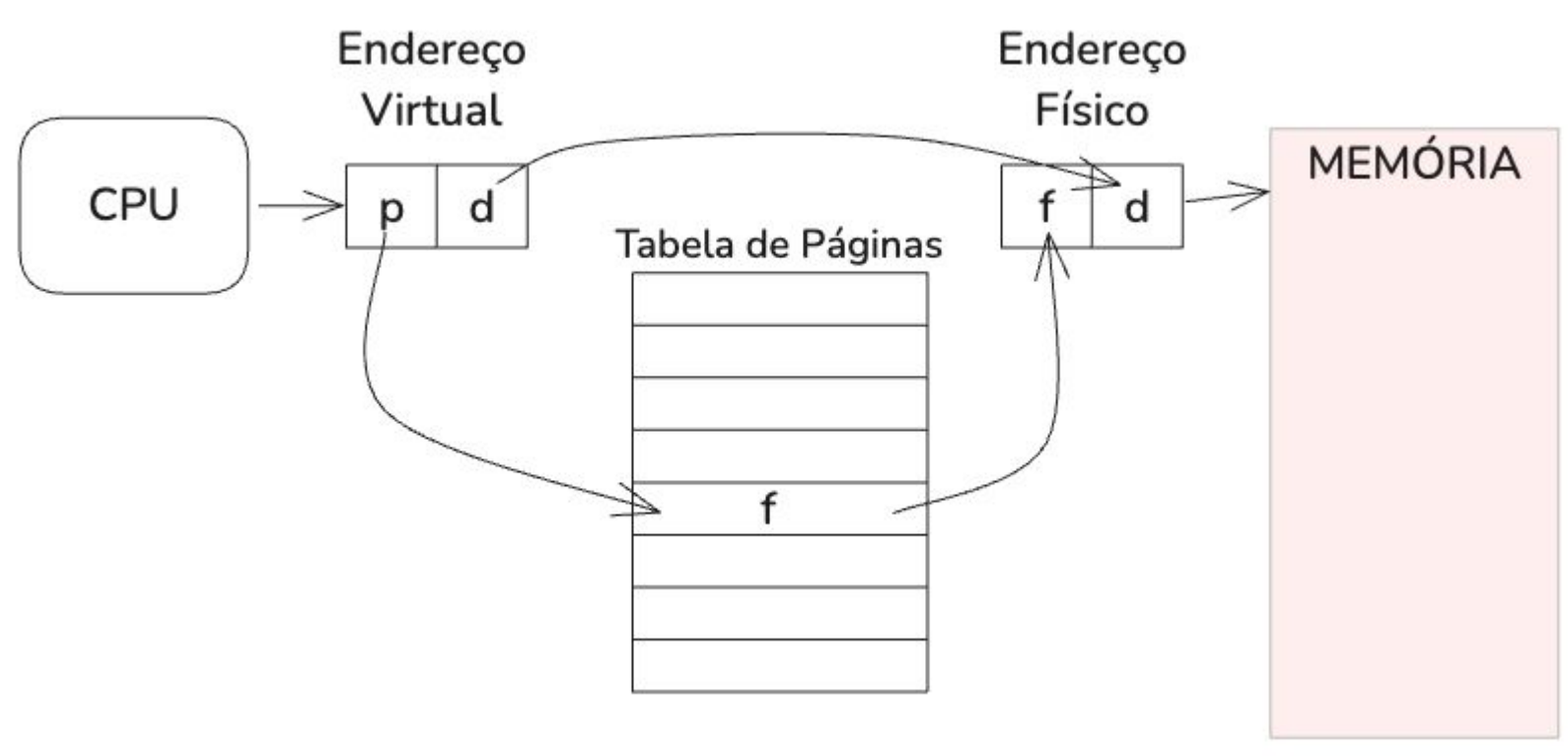
HARDWARE DE PAGINAÇÃO

Problema: Como encontramos endereços quando as páginas não estão alocadas de forma contígua na memória?

Endereço Virtual:

- O compilador gera endereços virtuais (lógicos) para os processos na memória.
- O processo gera ***endereços virtuais contíguos, de 0 até o tamanho do processo***.
- O SO organiza o processo em páginas, e o hardware de paginação traduz os endereços virtuais para endereços físicos reais na memória.
- Na paginação, o endereço virtual identifica a página e o deslocamento (offset) dentro da página.
- A **tabela de páginas rastreia o quadro de página na memória onde cada página está localizada**.

HARDWARE DE PAGINAÇÃO



HARDWARE DE PAGINAÇÃO

- A paginação é uma forma de **relocação dinâmica**, onde cada endereço virtual é associado pelo hardware de paginação a um endereço físico.

- Pense na tabela de páginas como um conjunto de registradores de relocação, um para cada quadro (*FRAME ou 'registro'*) de página.

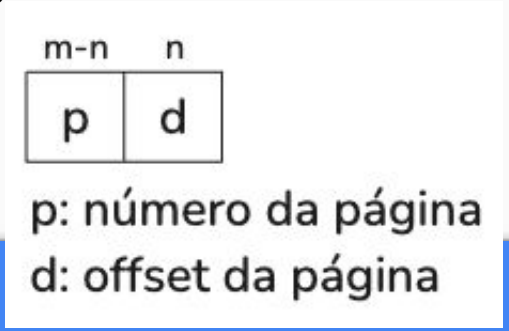
- **O mapeamento é invisível para o processo**; o SO mantém o mapeamento e o hardware realiza a tradução.

- A proteção é fornecida com os mesmos **mecanismos usados na relocação dinâmica (endereço base/limite por página)**.

HARDWARE DE PAGINAÇÃO: DETALHES DE IMPLEMENTAÇÃO

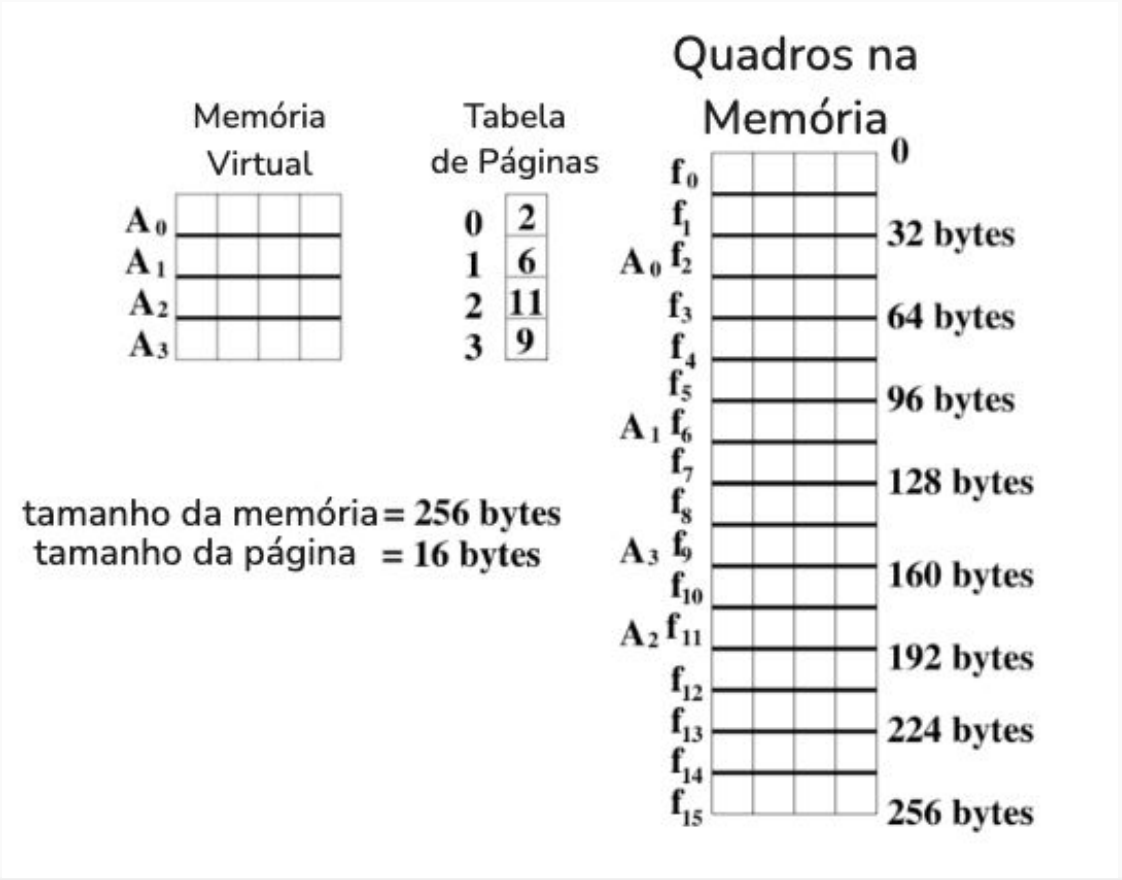
- O tamanho da página (tamanho dos quadros) normalmente é uma potência de 2, variando entre **512 bytes e 8192** bytes por página.

- Potências de 2 facilitam a tradução de endereços virtuais em endereços físicos. Por exemplo, dado:
  - um espaço de endereços virtuais de tamanho 2^m bytes e uma página de tamanho 2^n,
  - os *m-n* bits mais significativos de um endereço virtual selecionam a página,
  - os n bits menos significativos selecionam o deslocamento dentro da página.



TRADUÇÃO DE ENDEREÇO - EXEMPLO

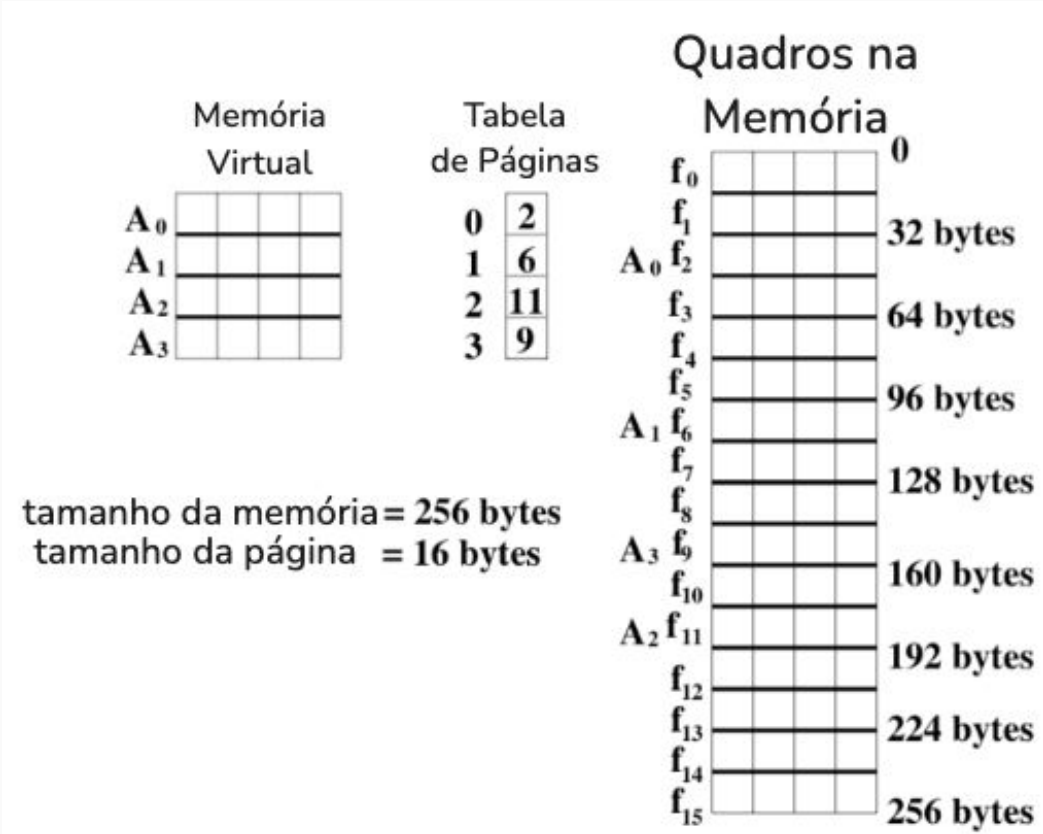
O quão grande é a tabela de páginas?



TRADUÇÃO DE ENDEREÇO - EXEMPLO

O quão grande é a tabela de páginas?

16 páginas (256 bytes em memória física / 16 bytes por página)

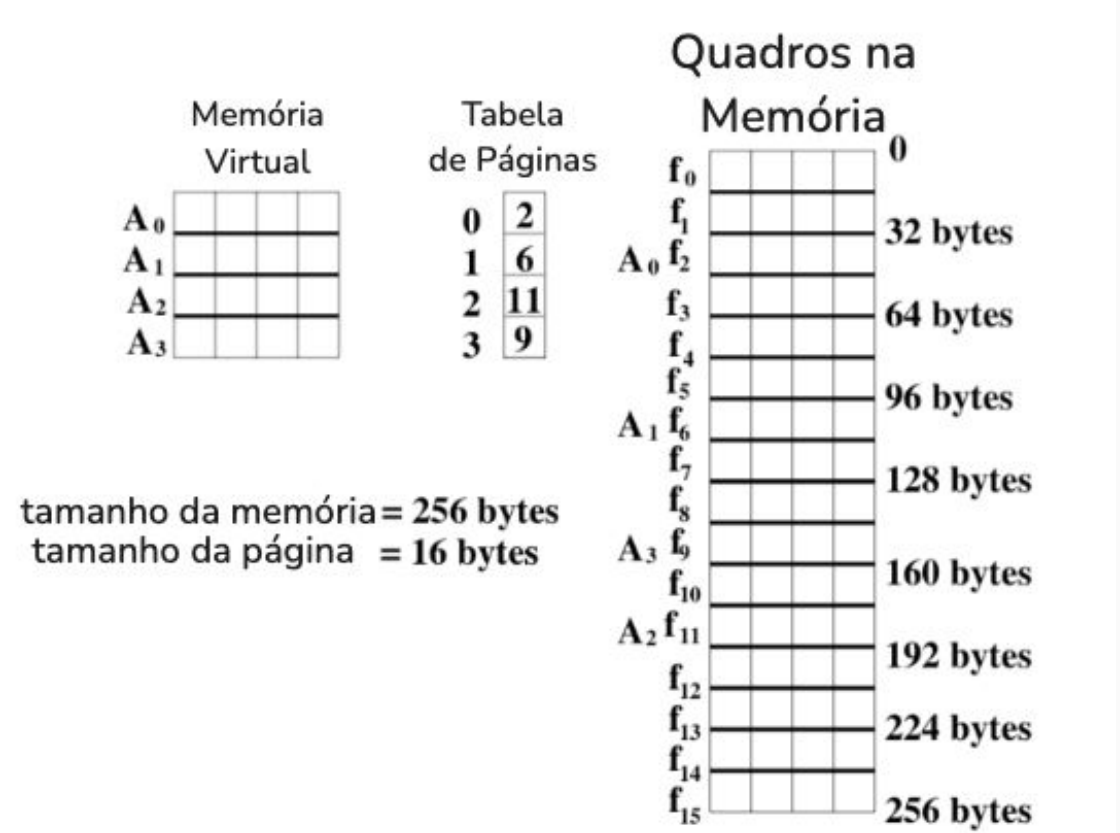


TRADUÇÃO DE ENDEREÇO - EXEMPLO

O quão grande é
a tabela de páginas?

16 páginas (256 bytes em memória física / 16 bytes por página)

Quantos bits preciso para endereçar a memória física inteira?



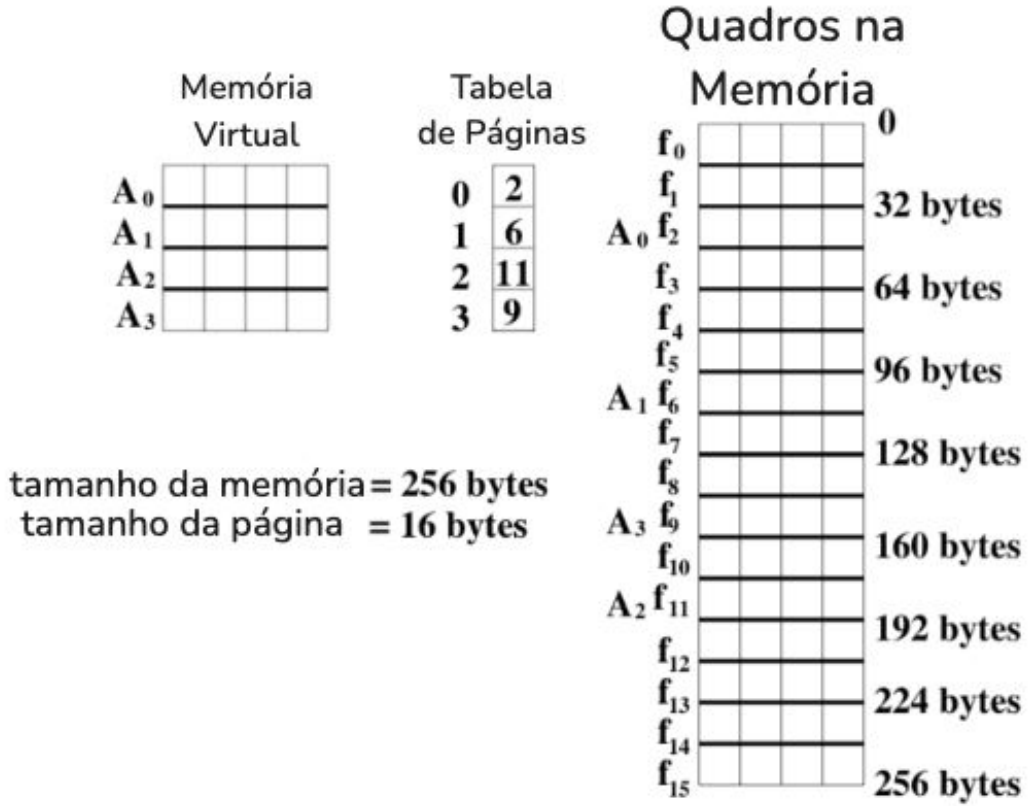
TRADUÇÃO DE ENDEREÇO - EXEMPLO

O quão grande é
a tabela de páginas?

16 páginas (256 bytes em memória física / 16 bytes por página)

Quantos bits preciso para endereçar a memória física inteira?

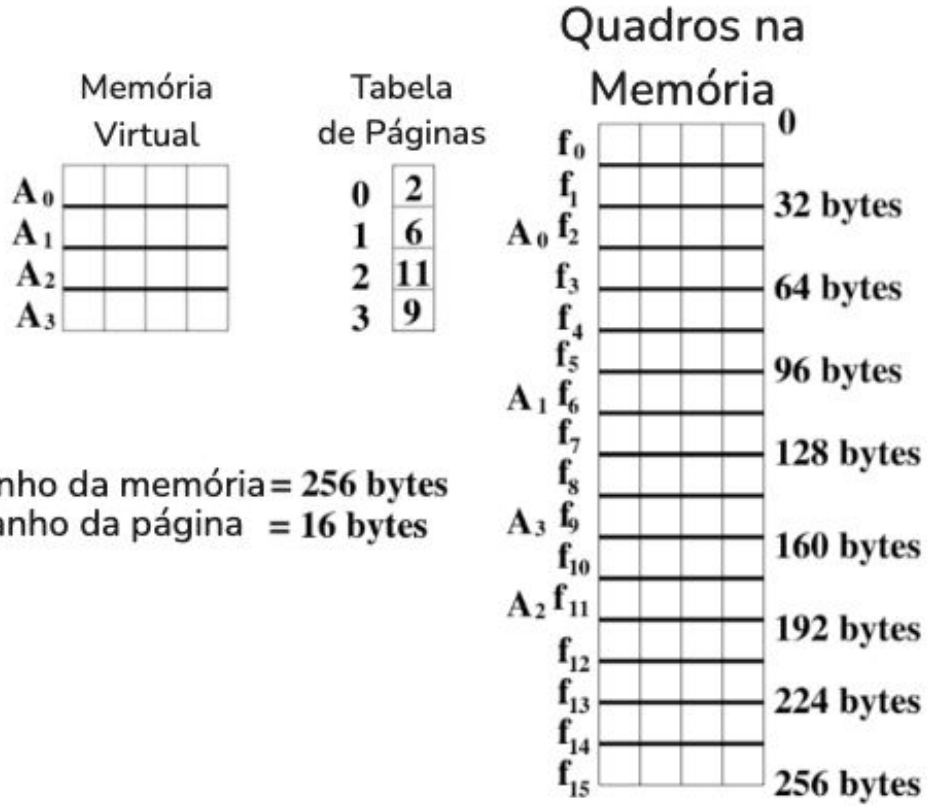
1 byte = 8 bits (2^8)



TRADUÇÃO DE ENDEREÇO - EXEMPLO

1 byte para endereçar 256 bytes.

Qual parte é p e qual parte é d?



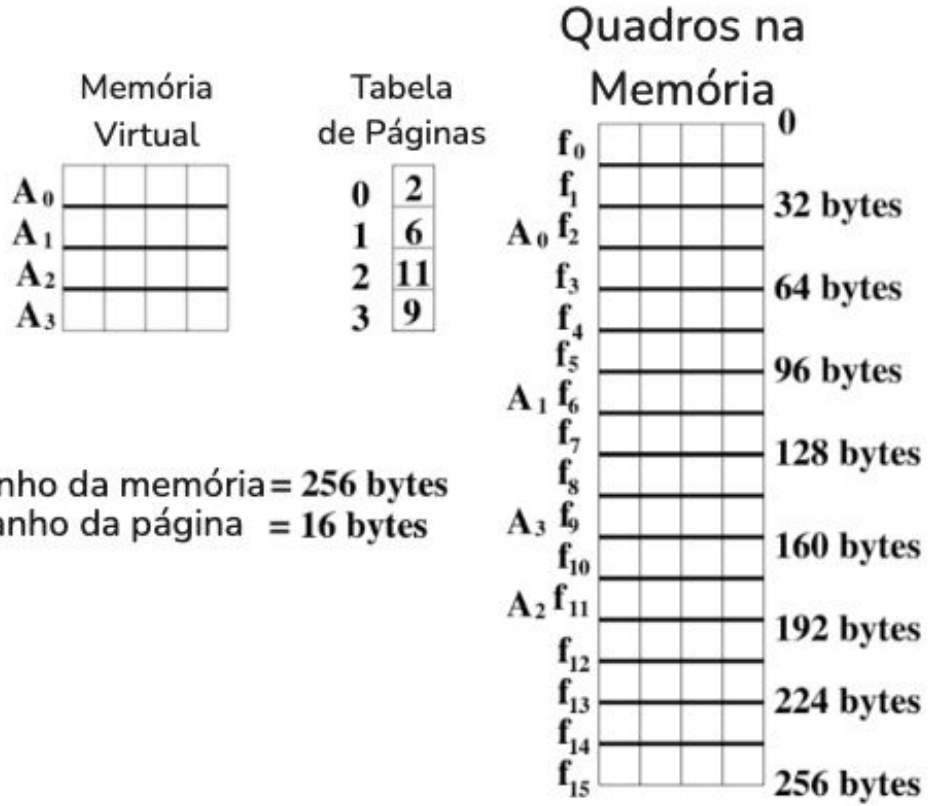
TRADUÇÃO DE ENDEREÇO - EXEMPLO

1 byte para endereçar 256 bytes.

Qual parte é p e qual parte é d?

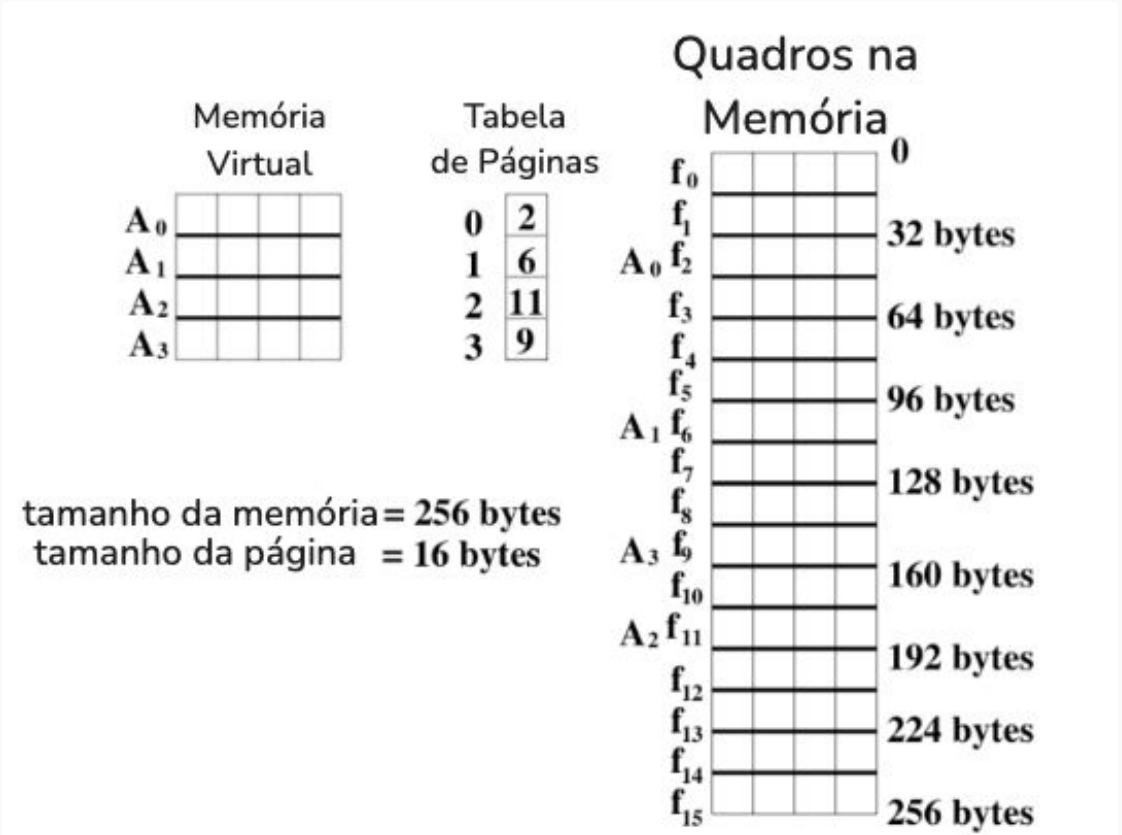
4 bits para a página (16 páginas = 2^4)

4 bits para o offset dentro da página.



TRADUÇÃO DE ENDEREÇO - EXEMPLO

Dado o endereço virtual 24, calcule seu endereço físico.



TRADUÇÃO DE ENDEREÇO - EXEMPLO

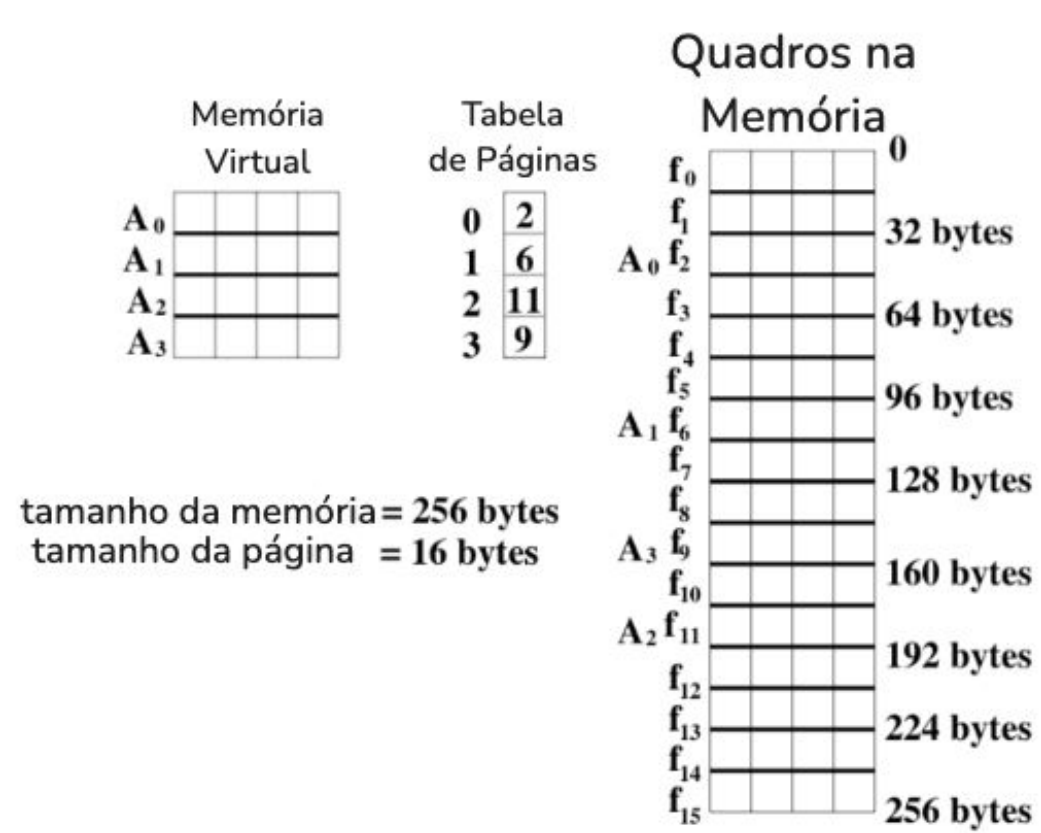
Dado o endereço virtual 24, calcule seu endereço físico.

24/16 = 1
(divisão inteira pelo tamanho da página)

Está na página 1.

p = 1

E o offset?
d => resto da divisão
(mod '%' do endereço virtual com o tamanho da página)



TRADUÇÃO DE ENDEREÇO - EXEMPLO

Dado o endereço virtual 24, calcule seu endereço físico.

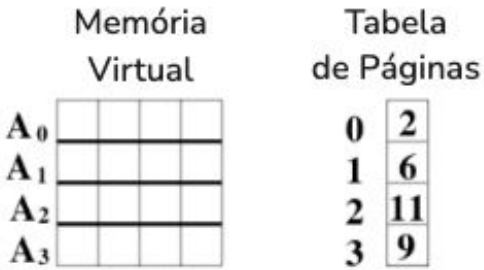
24/16 = 1
(divisão inteira)

Está na página 1.

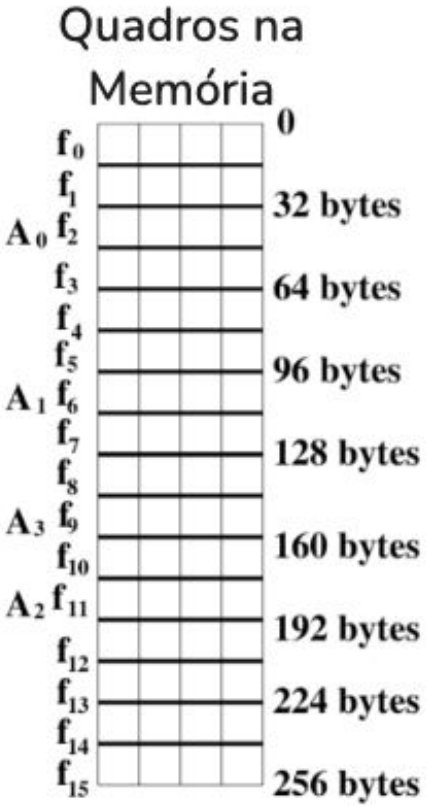
p = 1
d = 8

00011000
p d

Em qual quadro (frame)?



tamanho da memória = 256 bytes
tamanho da página = 16 bytes



TRADUÇÃO DE ENDEREÇO - EXEMPLO

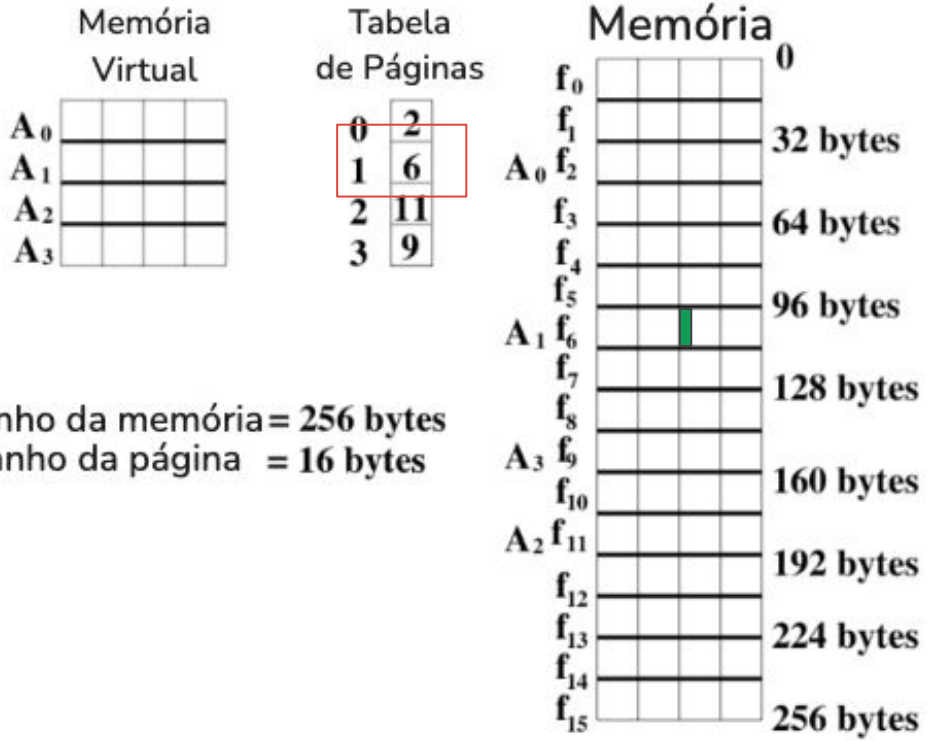
Dado o endereço virtual 24, calcule seu endereço físico.

24/16 = 1
(divisão inteira)

Está na página 1.

p = 1 **f = 6**
d = 8 **d = 8**

00011000 **01101000**
p **d** **f** **d**



tamanho da memória = 256 bytes
tamanho da página = 16 bytes

TRADUÇÃO DE ENDEREÇO - EXEMPLO

Em uma arquitetura típica de 32 bits, endereços tem 4 bytes cada.

Ainda temos uma memória de tamanho = 256 bytes.

Quantos *frames (ou 'quadros')* na memória teremos se considerarmos palavras de tamanho => 4 bytes?

256/4 => 64 frames

Podemos utilizar então 16 páginas não contíguas para referenciar 64 endereços contiguos.

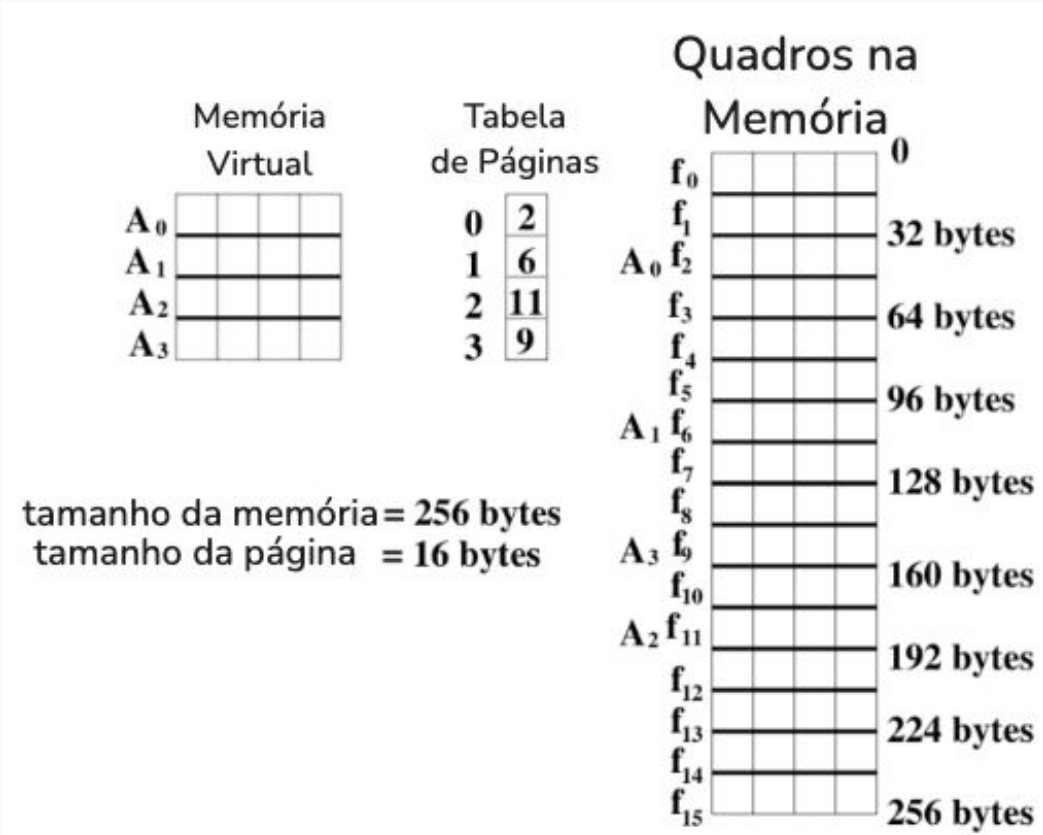
TRADUÇÃO DE ENDEREÇO - EXEMPLO

Agora: 32 bits (4 bytes) por endereço.

Quantos bits preciso para endereçar 1 posição na memória?
64 endereços -> 2^6
⇒ *6 bits!*

Qual parte é p, e d ?

4 bits por página (16 páginas)
2 bits pro offset



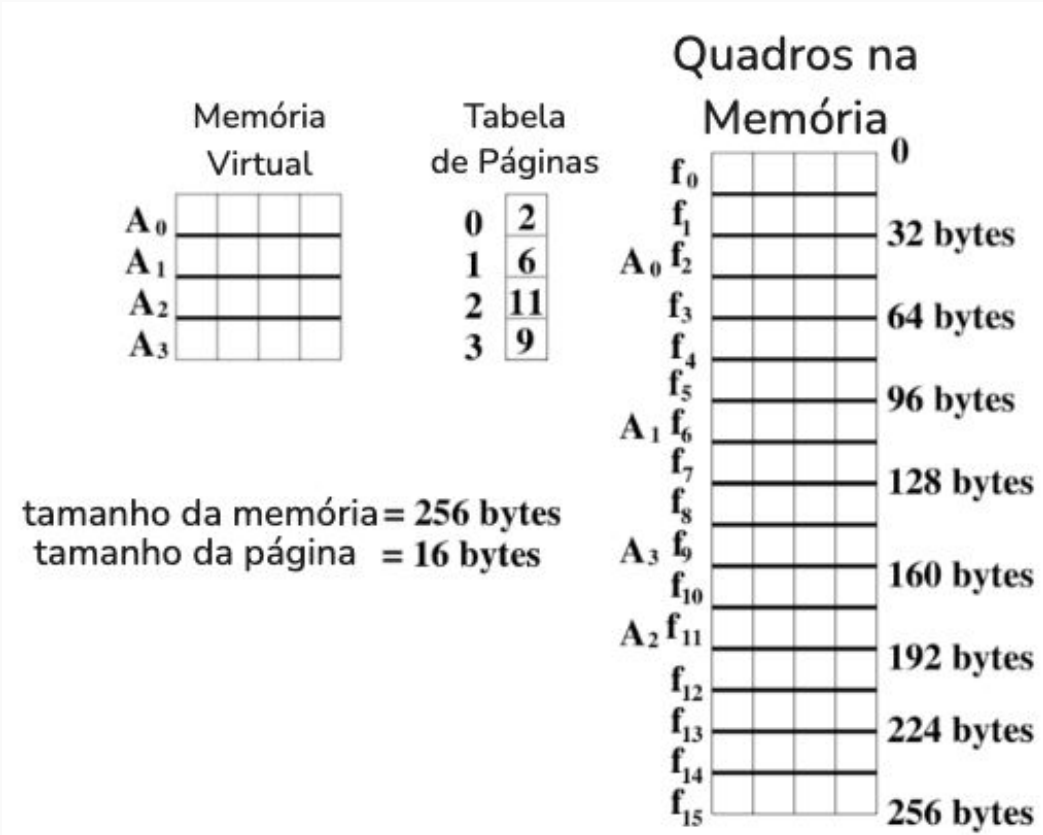
TRADUÇÃO DE ENDEREÇO - EXEMPLO

Agora: 32 bits (4 bytes) por endereço.

Quantos bits preciso para endereçar 1 posição na memória?
64 endereços -> 2^6
⇒ *6 bits!*

DADO O ENDEREÇO VIRTUAL 13, FAÇA A TRADUÇÃO PARA ENDEREÇO FÍSICO:

13 -> ?



TRADUÇÃO DE ENDEREÇO - EXEMPLO

Agora: 32 bits (4 bytes) por endereço.

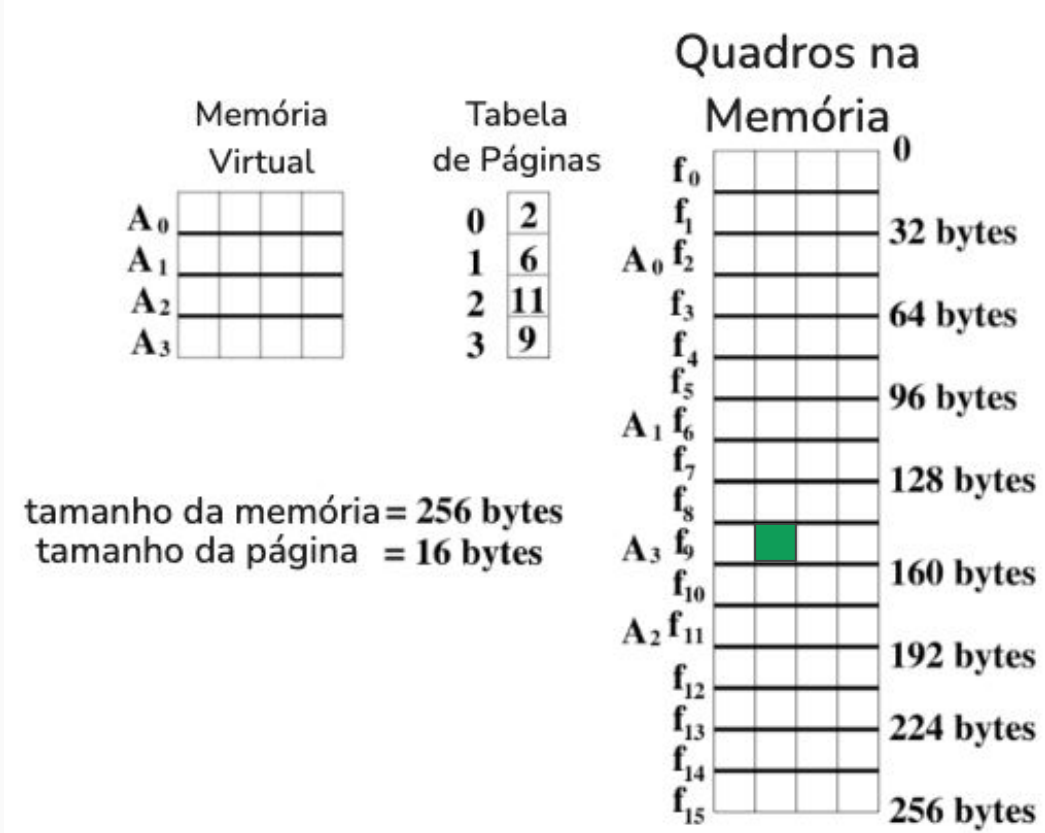
DADO O ENDEREÇO VIRTUAL 13, FAÇA A TRADUÇÃO PARA ENDEREÇO FÍSICO:

13 ->

Antes, uma página (16 bytes) tinha 16 endereços. Agora tem 4 (1 palavra => 4 bytes)

13/4 -> 3, resta 1
(divisão inteira pelo tamanho da página)

p = 3 f = 9
d = 1 d = 1



TRADUÇÃO DE ENDEREÇO - EXEMPLO

- Quais informações precisam ficar no PCB para possibilitar a troca de contexto entre processos?

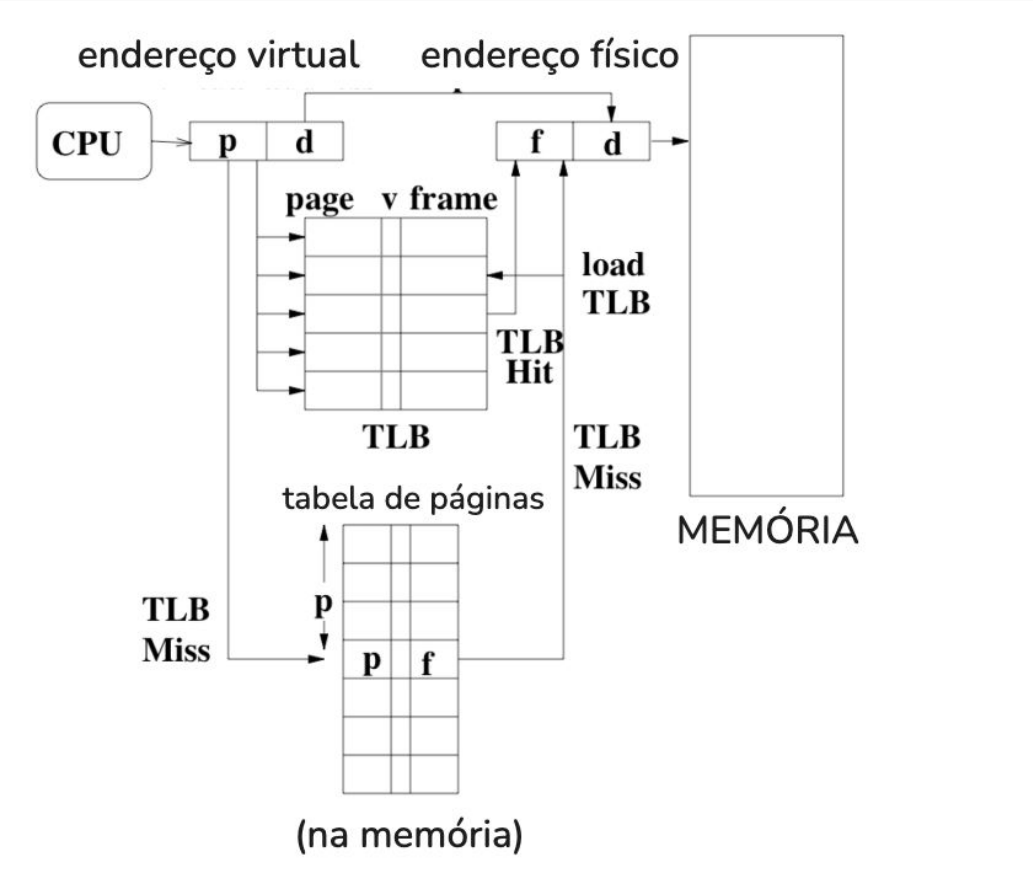
COMO TORNAR PAGINAÇÃO EFICIENTE

- Como armazenar a tabela de páginas?
  - Em Registradores: Vantagens? Desvantagens?
  - Em Memória: Vantagens? Desvantagens?

COMO TORNAR PAGINAÇÃO EFICIENTE

- Como armazenar a tabela de páginas?
  - Em Registradores: Vantagens? Desvantagens?
  - Em Memória: Vantagens? Desvantagens?
  - TLB: Uma memória rápida e associativa que armazena os números das paginas (chaves) aos quadros na memória física (valor). ⇒ *tabela hash / O(1)*
    - Se acessos de memória tem localidade, tradução de endereços também tem problemas de localidade
    - Tipicamente: TLB's tem até 2048 entradas.

TLB - TRANSLATION LOOK-ASIDE BUFFER



- v: bit de validade que diz se a entrada na tabela está atualizada
  - *dois processos podem estar usando a página 5, mas cada um tem um conteúdo diferente.*

- regra 90/10: TLB garante boa performance, mesmo com falhas (*miss*)

CUSTO DE UTILIZAR O TLB

- **Qual é o custo efetivo em termos de acessos à memória (am) se a tabela de páginas está na memória?**
  - custo_efetivo = 2 * ma

- **Qual é o custo efetivo em termos de acessos à memória com o TLB?**

- Probabilidade ou taxa de TLB acertos (*hits*): p
- custo_efetivo = (ma + TLB) * p + (2 * ma + TLB) * (1-p)

UM TLB **MAIOR**, AUMENTA A **TAXA DE ACERTO** e **DIMINUI CUSTO DE ACESSOS À MEMÓRIA.**

INICIALIZANDO A MEMÓRIA AO INICIAR O PROCESSO

1. Um processo que precisa de **k páginas** chega.
2. Se **k quadros de página estão livres**, aloque estes quadros livres para as páginas.
   Caso contrário, libere quadros que não são mais necessários.
3. O SO coloca cada página em um quadro e, em seguida, registra o número do quadro na entrada correspondente na tabela de páginas.
4. O SO invalida todas as entradas da TLB (limpa a TLB).
5. O SO inicia o processo.
6. Enquanto o processo é executado, o SO carrega as entradas da TLB à medida que cada página é acessada, substituindo uma entrada existente se a TLB estiver cheia.

SALVANDO / RESTAURANDO MEMÓRIA EM UMA MUDANÇA DE CONTEXTO

• O Bloco de Controle do Processo (PCB) precisa ser extendido para conter:
  - A tabela de páginas
  - Possivelmente uma cópia da TLB

• Em uma troca de contexto:
1. Copie o valor do registrador base da tabela de páginas para o PCB.
2. Copie a TLB para o PCB (*opcional*).
3. Limpe a TLB.
4. Restaure o registrador base da tabela de páginas.
5. Restaure a TLB, se ela tiver sido salva.

• **Paginação em Níveis:** *Se o espaço de endereço virtual é muito grande, as tabelas de páginas ficam grandes demais, e muitos sistemas utilizam uma estratégia de paginação em múltiplos níveis (consulte OSC para mais detalhes).*

COMPARTILHAMENTO

A paginação permite o compartilhamento de memória entre processos, pois a memória usada por um processo não precisa ser contígua.

O código compartilhado deve ser **reentrante**, o que significa que os processos que o utilizam não podem alterá-lo (isto é: sem dados (escrita/leitura) no código reentrante).

O compartilhamento de páginas é semelhante à forma como as threads compartilham texto e memória entre si.

Uma página compartilhada pode estar em partes diferentes do espaço de endereçamento virtual de cada processo, mas os endereços virtuais mapeiam para o mesmo endereço físico.
- O programa do usuário (ex.: emacs) marca o segmento de texto de um programa como reentrante com uma chamada de sistema.
- O sistema operacional rastreia o código reentrante

COMPARTILHAMENTO

A paginação permite o compartilhamento de memória entre processos, pois a memória usada por um processo não precisa ser contígua.

O programa do usuário (ex.: emacs) marca o segmento de texto de um programa como **reentrante com uma chamada de sistema.**

O sistema operacional rastreia o código reentrante disponível na memória e **o reutiliza caso um novo processo solicite o mesmo programa.**

Isso pode reduzir significativamente os requisitos de memória para aplicativos de uso frequente.

RESUMO

- **A paginação é uma grande melhoria em relação à relocação:**
  - Elimina o problema de **fragmentação externa** e a necessidade de compactação.
  - Permite o **compartilhamento de páginas de código** entre processos, reduzindo o consumo total de memória.
  - Permite que processos sejam executados mesmo estando apenas **parcialmente carregados** na memória principal.

- **No entanto, a paginação tem seus custos:**
  - A **tradução** de um endereço virtual para um endereço físico é mais **demorada**.
  - A paginação exige **suporte de hardware**, na forma do TLB, para ser eficiente.
  - A paginação exige um SO mais complexo para **gerenciar a tabela de páginas**.

EXERCÍCIOS

Um sistema operacional implementa memória virtual por paginação com 32 páginas contíguas, mapeadas **em memória física de tamanho igual a 2048 bytes (2KB).** Considere o tamanho da memória virtual igual ao tamanho da memória física.

Para uma arquitetura de 64 bits, calcule:

1) o **tamanho de cada página**;
2) o **endereço físico** relativo ao endereço virtual **156**.
3) o **endereço físico** relativo ao endereço virtual **85**.
4) o **endereço físico & o endereço virtual** relativos à **penúltima palavra da página 6**.
5) o **endereço físico & o endereço virtual** relativo à **terceira palavra da página 10**.

Tabela de Páginas

página	byte
6	128
1	1088
19	1728
10	448

PERGUNTAS?

REFERÊNCIAS

- **TANENBAUM, Andrew.** Sistemas operacionais modernos.
- **SILBERSCHATZ, Abraham et al.** Fundamentos de sistemas operacionais: princípios básicos.
- **MACHADO, Francis; MAIA, Luiz Paulo.** Arquitetura de Sistemas Operacionais.
- **CARISSIMI, Alexandre et al.** Sistemas operacionais.