CURSO TÉCNICO EM INFORMÁTICA
SISTEMAS OPERACIONAIS II

MEMÓRIA VIRTUAL

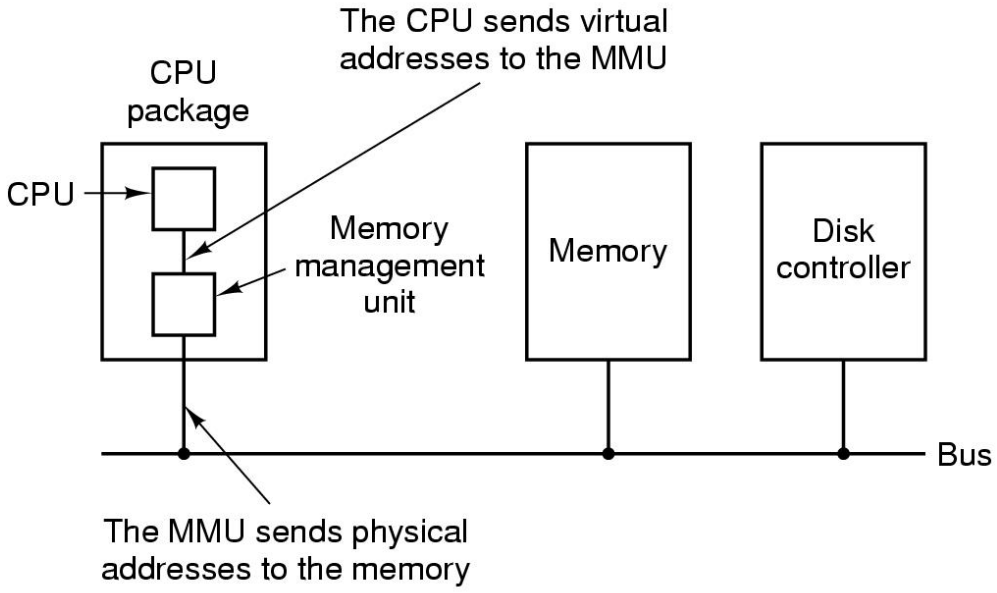
- O Processo passa a possuir um espaço de endereçamento virtual. Este espaço de endereçamento pode ser maior que a memória física.
- S.O. gerencia o espaço da memória alocando partes deste espaço para cada um dos processos.
- Cada processo possui o seu próprio endereçamento, sendo este endereço virtual convertido para um endereço real pelo S.O. ou Hardware (MMU).
- O S.O. realiza SWAP de partes dos processos, mantendo na memória principal as áreas necessárias, e em disco, as áreas não utilizadas no momento.
- Duas técnicas são utilizadas para viabilizar a memória virtual: Paginação ou Segmentação

PAGINAÇÃO

É a técnica de memória virtual mais utilizada.

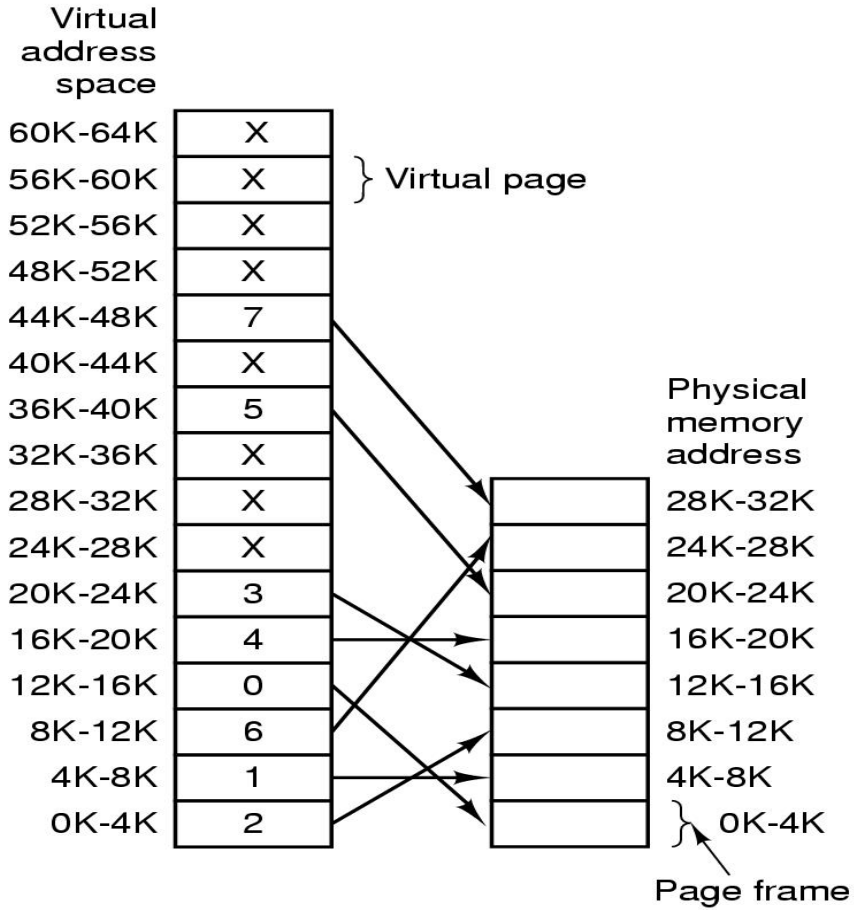
Quando um processo acessa um endereço de memória, este endereço é correspondente ao espaço de endereçamento do processo. O S.O. (ou o Hardware) deve converter este endereço em um endereço na memória principal.

O MMU – Memory Management Unit – é o chip responsável pela conversão endereço-virtual/e



O MMU deve ter uma tabela interna que realiza o mapeamento dos endereços.

Exemplo: Se tivermos um computador com apenas 32K de memória e que gere endereços virtuais de até 64K (endereços de 16 bits) e considerando que cada página deva ter 4K, teremos a seguinte situação:



Baseando-se nesse modelo, podemos visualizar alguns procedimentos de conversão aplicados a algumas instruções de movimento de dados entre a memória e um registrador.

1. Instrução: MOVE REG,0

   O endereço virtual 0 (zero) é mandado ao MMU
   O MMU identifica que este endereço está na página 0
   Esta página está mapeada no frame 2 da memória física
   O MMU transforma o endereço virtual 0 no endereço real 8192
   A instrução final será: MOVE REG,8192

2. Instrução: MOVE REG,8192

   O endereço virtual 8192 é mandado ao MMU
   O MMU identifica que este endereço está na página 2
   Esta página está mapeada no frame 6 da memória física
   O MMU transforma o endereço virtual 8192 no endereço real 24576
   A instrução final será: MOVE REG,24576

3. Instrução: MOVE REG,20500

   O endereço virtual 20500 é mandado ao MMU
   O MMU identifica que este endereço está na página 5
   Esta página está mapeada no frame 3 da memória física
   O MMU transforma o endereço virtual 20500 (20K + 20) no endereço real 12308 (12K + 20)
   A instrução final será: MOVE REG,12308

4. Instrução: MOVE REG,12300

   O endereço virtual 12300 é mandado ao MMU
   O MMU identifica que este endereço está na página 3
   Esta página está mapeada no frame 0 da memória física
   O MMU transforma o endereço virtual 12300 (12K + 12) no endereço real 12
   A instrução final será: MOVE REG,12

Como neste exemplo a memória principal não é suficiente para armazenar todo o processo, ele estará com todas as suas páginas virtuais em uma área na memória secundária.

Se a operação desejada fosse MOVE REG,32780 ocorreria um evento chamado "PAGE FAULT", que indicaria que a página acessada não está atualmente na memória principal.

Após a ocorrência de um "PAGE FAULT" o S.O. escolhe um page frame (por exemplo: o menos utilizado), escreve para a memória secundária, carrega no seu espaço a página virtual necessária e reinicializa a instrução.

É comum a tabela de páginas do MMU conter um "bit de presença" que indica a presença ou não daquela página na memória principal.

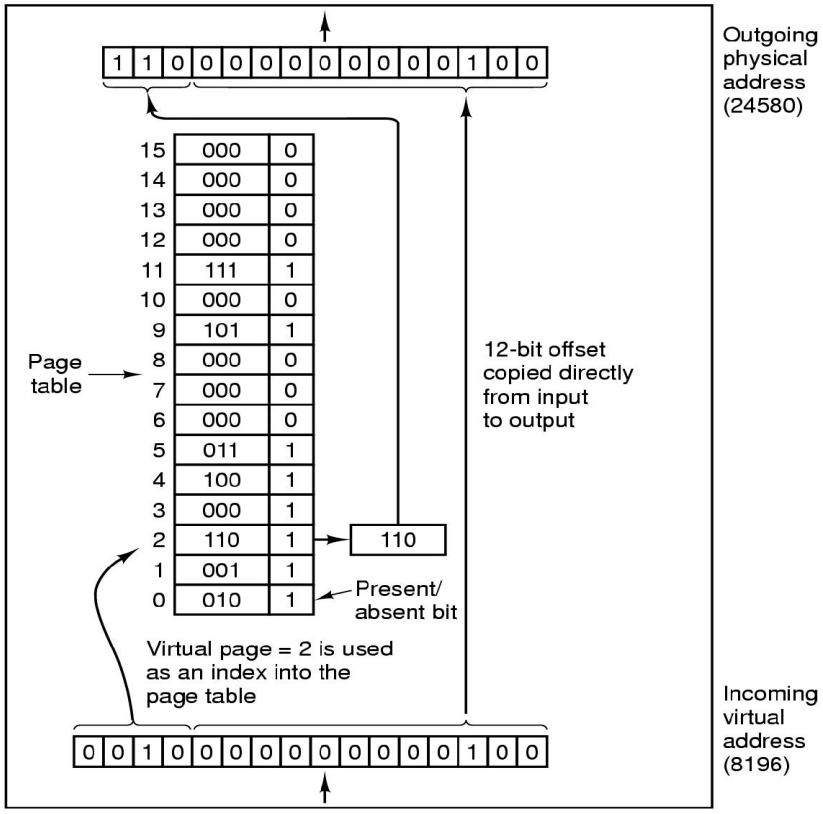
Cada processo possui sua própria tabela de páginas.

Esta figura demonstra um esquema possível para o funcionamento interno do MMU.

É passado ao MMU o endereço virtual: 8196 (em 16 bits) para ser convertido no endereço real correspondente.

Os 4 bits de mais alta ordem do endereço virtual identificam a página na tabela onde está armazenado o endereço da frame

O endereço final é composto dos 3 bits da frame mais os 12 bits restantes (deslocamento) do endereço original (em 15 bits)



Paginação - Como funciona

1. SO traz para a MP algumas páginas do programa;
2. conjunto de páginas de um processo presentes na MP é chamado de conjunto residente;
3. Quando é necessário um endereço que não está presente na MP uma interrupção é gerada (Page Fault) e o processo é colocado no estado "bloqueado";
4. Enquanto o SO busca a página necessária (acesso a disco), outro processo é despachado para usar a CPU;
5. Se a MP estiver toda ocupada, o SO deverá selecionar uma frame a ser liberada para receber a nova página;
6. Se a página que estava ocupando a frame escolhida para ser liberada tiver sido modificada (dados alterados), o SO deverá atualizá-la na memória virtual antes de proceder a liberação da frame, evitando que os novos dados sejam perdidos;
7. Quando a página é trazida para a memória, o primeiro processo passa para a fila dos "prontos"
8. Todas essas operações são transparentes para o processo (programa)

Algoritmos de Troca de Página

1. NRU (Not Recently Used) - página não utilizada recentemente

- Associa a cada página dois bits (R e M)
  - R => página sofreu uma operação de leitura
  - M => página sofreu uma operação de escrita
- Esses bits estão na tabela de páginas
- Dividem as páginas em 4 classes:
  - classe 0: não referenciada - não modificada
  - classe 1: não referenciada - modificada
  - classe 2: referenciada - não modificada
  - classe 3: referenciada - modificada
- No início todas as páginas são de classe 0
- S.O. de tempos em tempos limpa o bit de referência, mudando as pag.para classes 0 e 1
- Libera páginas de menor classe
- Algoritmo fácil, eficiente e com boa performance

2. FIFO

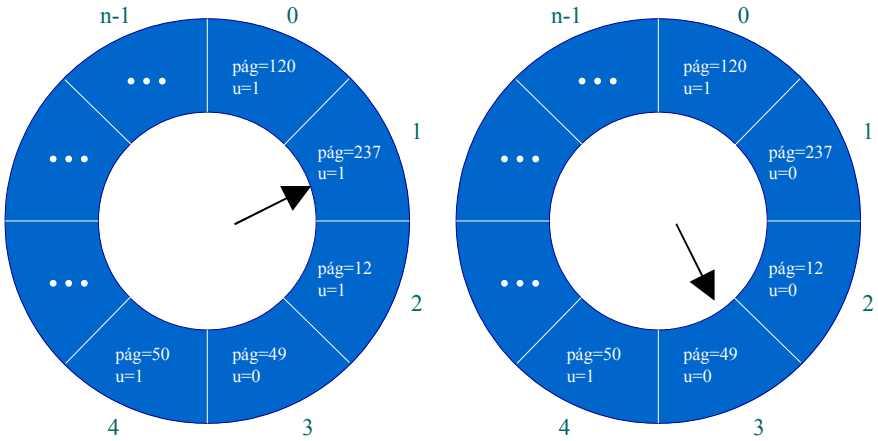
- S.O. mantém um lista com todas as páginas em memória ordenada pela ordem de chegada
- A mais antiga será sempre a página escolhida para ser substituída
- Não é eficiente, pois a página mais antiga pode ser também a mais acessada

3. Algoritmo de segunda chance

- Aprimoramento do FIFO
- Utiliza um bit (R) para verificar se a página foi acessada
- Se a página mais antiga está com o bit R = 0, então ela não foi acessada => é retirada
- Se a página mais antiga está com o bit R = 1, então ela foi acessada => o bit é zerado e ela é colocada no final da fila (como se acabasse de ser carregada)

4. Algoritmo do relógio

- Melhoramento da técnica de 2a.chance
  - Mantém uma fila circular das páginas,
  - Dispensa o movimento de páginas para o final da fila
- Algoritmo bom e rápido



5. LRU (Least Recently Used) - página usada há mais tempo

- Presume que a página menos utilizada no espaço de tempo anterior será a menos utilizada a seguir
- Esse método exige a implementação de um controle de uso das páginas que pode encarecer significativamente o mecanismo.
- Existem algumas implementações de hardware relativamente simples para viabilizar esse controle:
  - Contador: existe um contador de páginas de 64 bits que é incrementado a cada instrução. Após cada referência à memória, o valor do contador é armazenado na tabela de páginas correspondente à página que acabou de ser referenciada.
    Se ocorrer uma falta de páginas o SO examina todos os valores dos contadores armazenados na tabela de páginas para encontrar o mais baixo. A página cujo contador tiver o menor valor é a usada há mais tempo.
  - Matriz N x N: Para uma máquina com N molduras de página, o hardware do LRU pode manter uma matriz N x N bits, inicialmente todos zerados.
    Sempre que uma moldura k for referenciada, o hardware coloca todos os bits da linha k em 1 e, depois, zera todos os bits da coluna k. Em cada instante, a linha com o menor valor binário armazenado será correspondente à página usada há mais tempo.
- O NFU (Not Frequently Used) é uma alternativa de implementação desse mecanismo somente através de software. Utiliza os mesmos bits R de outros algoritmos e, somando-os em um contador por página tem aproximadamente o número de acessos às páginas.

CONCEITOS IMPORTANTES

- WORKING SET: Conjunto de páginas ativas do processo
- THRASHING: Ocorre quando a tarefa de páginação (swap) gasta mais tempo que o processamento dos processos
- PÁGINAS COMPARTILHADAS: Páginas que são utilizadas por 2 ou mais processos ao mesmo tempo (ex.: componentes do núcleo referenciados pelos processos).

LRU - Matriz N x N – Exemplo:

Considerando 4 (quatro) molduras de páginas e a seguinte ordem de referência de páginas:
0 1 2 3 2 1 0 3 2 3, teremos a seguinte situação:
Obs.: moldura *k* referenciada: linha *k* ligada e coluna *k* zerada.

	0	1	2	3
0	0	1	1	1
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0

(a)

	0	1	2	3
0	0	0	1	1
1	1	0	1	1
2	0	0	0	0
3	0	0	0	0

(b)

	0	1	2	3
0	0	0	0	1
1	1	0	0	1
2	1	1	0	1
3	0	0	0	0

(c)

	0	1	2	3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0

(d)

	0	1	2	3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	1
3	1	1	0	0

(e)

	0	1	2	3
0	0	0	0	0
1	1	0	1	1
2	1	0	0	1
3	1	0	0	0

(f)

	0	1	2	3
0	0	1	1	1
1	0	0	1	1
2	0	0	0	1
3	0	0	0	0

(g)

	0	1	2	3
0	0	1	1	0
1	0	0	1	0
2	0	0	0	0
3	1	1	1	0

(h)

	0	1	2	3
0	0	1	0	0
1	0	0	1	0
2	1	1	0	1
3	1	1	0	0

(i)

	0	1	2	3
0	0	1	0	0
1	0	0	0	0
2	1	1	0	0
3	1	1	1	0

(i)