Cruzeiro do Sul
Educacional

Gerenciamento de Memória

Conteudista
Prof. Me. Claudney Sanches Júnior

Revisão Textual
Aline de Fátima Camargo da Silva

Sumário

Objetivos da Unidade

- Entender a gerência da memória em SOs;
- Aprender sobre memória virtual, paginação e segmentação;
- Conhecer o gerenciador de memória e seus principais algoritmos.

Atenção, estudante! Aqui, reforçamos o acesso ao conteúdo *on-line* para que você assista à videoaula. Será muito importante para o entendimento do conteúdo.

Este arquivo PDF contém o mesmo conteúdo visto *on-line*. Sua disponibilização é para consulta *off-line* e possibilidade de impressão. No entanto, recomendamos que acesse o conteúdo *on-line* para melhor aproveitamento.

VOCÊ SABE RESPONDER?

Uma empresa comprou uma CPU com muita memória física, como solucionar esse problema?

Contextualização

Figura 1 – Suporte
Fonte: Vida de Suporte

#ParaTodosVerem: a imagem mostra uma história em quadrinho em quatro partes. No primeiro quadrinho há o título "Suporte – Baseado em história real enviada por Marlon Guilherme", com fundo preto. No segundo, há um homem, um programador, carrega um CPU; ao seu lado, uma mulher diz: "Por que, em vez de jogar fora, você não coloca esse computador para a funcionária nova?". No terceiro, um homem responde: "Ele é muito antigo. Inclusive tá até sem a memória e a fonte.". No quarto quadrinho, a mulher diz: "Mas ela só precisa do *Word*"; então, o homem derruba o CPU, com espanto. Fim da descrição.

Uma empresa comprou uma CPU com muita memória física. Faça um documento no *word* com os passos necessários para desabilitar no *Windows* a Memória Virtual. Demonstre, capturando as telas, que você consegue desabilitar e utilizar o SO. Isto é, ilustre, com imagens da tela, os passos necessários para a desabilitação da Memória Virtual.

Memória

Antes de começar a falar do gerenciamento de memória do SO, precisamos revisar os conceitos desta, bem como a memória principal e a memória secundária. O primeiro conceito a se guardar é que ao se reportar à memória, em computação, estamos nos referindo a todos os dispositivos que permitem a um computador guardar dados temporariamente ou permanentemente. O termo memória é genérico, servindo tanto para o armazenamento de dados como para o armazenamento de programas.

> As memórias de alta velocidade, localizadas no processador que guardam dados para uso imediato, são as mais velozes e caras de um sistema computacional, pois operam na mesma velocidade dos processadores e recebem o nome de registradores.

Como existe uma diferença de velocidade muito grande dos registradores em relação à memória principal, surgiu a necessidade de um tipo de memória interna que intermedia o processador e o dispositivo de armazenamento, normalmente, com um serviço que antecipa a probabilidade de dados serem usados novamente. Esta memória mais lenta que os registradores e mais rápida que a memória principal recebeu o nome de Cache.

Com o avanço tecnológico, várias caches foram desenvolvidas, sendo algumas conhecidas e outras não. As que daremos destaque são as caches L1, L2 e L3. As memórias caches são medidas por sua capacidade de armazenamento e sua latência. Latência é o tempo decorrido entre um ciclo de *clock* da máquina e o tempo de transferência de dados.

A latência é medida em nanossegundos ou em ciclos de processador, ou seja, ciclos de *clock* que o processador tem que rodar sem executar nenhuma operação, pois fica aguardando a memória. Assim, a Cache L1 demora 2 ou 3 ciclos para responder a uma solicitação, enquanto se um dado for solicitado a cache L2, demora 10 ciclos. Dessa forma, os algoritmos de previsão de uso são também de extrema importância.

A memória principal consiste em memória volátil de acesso aleatório (*Random Access Memory – RAM*). O termo aleatório vem do sentido de que os processos podem acessar dados em qualquer ordem. O que diferenciava os sistemas de armazenamento em fita é que os dados somente eram lidos em uma determinada sequência, mas foram os meios de armazenamento populares dos primeiros computadores.

Saiba Mais

Todos usamos *caching*. Guardamos coisas próximas em lugares estratégicos para ter acesso fácil e rápido. Por exemplo, deixar um lápis ou caneta e papel sobre a mesa do escritório é uma forma de *caching*. Contudo, os projetistas de SO têm que tomar muito cuidado ao utilizá-la. A cache em computação é uma cópia do dado armazenado, sendo que esta cópia sofrerá constante alteração e o original não. Portanto, o SO tem que frequentemente copiar os dados da memória cache para o original — esse processo é denominado esvaziamento de cache.

Toda memória principal ou RAM se inicia com uma classificação que pode ser *Single In-line Memory Module* (SIMM) ou *Double In-line Memory Module* (DIMM), dependendo do pente de memória, isto é, a base em que os *chips* são soldados. Para saber de qual se trata basta olhar o pente que pode ter *chips* de memórias RAM soldadas de apenas um lado (*Single*) ou de ambos os lados (D*ouble*) (Deitel, 2005).

Ainda, a memória principal pode ser classificada quando a frequência e sincronização com o barramento. Se ela trabalha na mesma frequência do processador obedecendo ao mesmo ciclo de *clock*, sincronizando a saída de dados com os demais componentes do computador, recebe o nome de *Synchronous Dynamic Random Access Memory* (SDRAM).

As memórias assíncronas ou simplesmente *Dynamic Random Access Memory* (DRAM) tem um custo mais baixo e conseguem armazenar mais dados no mesmo espaço, contudo devido ao suporte a múltiplos pentes das SDRAM suplantou a DRAM.

O suporte a múltiplos pentes propicia que, enquanto um atende uma solicitação de leitura de um dado, outro já pode enviar uma resposta a outra solicitação, de forma que o barramento tem uma alimentação contínua. Outra característica que diferencia a DRAM é que ela necessita de um circuito de *refresh* ou renovação do sinal algumas vezes em um milissegundo para não perder os dados, enquanto que a SDRAM não necessita de tal recurso (Deitel, 2005).

Para finalizar a classificação de memória principal, devemos pensar na entrega dos pentes de memória, a qual, por sua vez, poderia entregar o dobro de dados em uma só transmissão. Assim, surgiu o padrão *Double Date Rate* (DDR) que dobrou a taxa de transferência de dados. Depois, surgiram a DDR2 e DDR3, as quais aumentaram este fator para 4x e 8x, respectivamente. E já foi lançada a DDR4 que deverá chegar à produção em massa após 2.014.

A memória secundária é uma expansão da memória principal menos dispendiosa e mais lenta. A latência do armazenamento em disco (HD) é, normalmente, medida em milissegundos, em geral, um milhão de vezes mais lento do que as memórias cache que são aquelas colocadas próximas aos processadores com latência de 10 a 20 ciclos de *clock*.

A seguir, a Figura 2 apresenta a latência das memórias (Deitel, 2005).

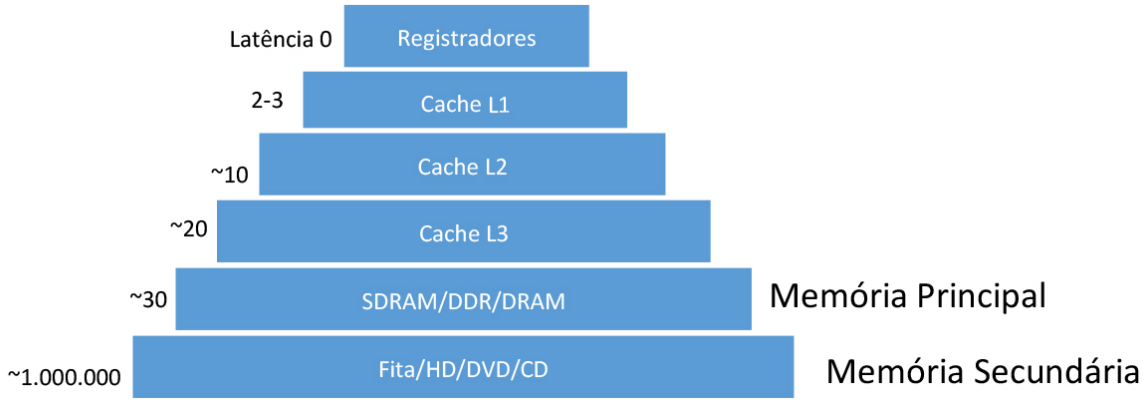

Figura 2 – Latência das memórias
Fonte: Elaborada pelo conteudista

#ParaTodosVerem: a imagem mostra um gráfico em formato pirâmide, em que há 6 níveis formados por linhas azuis, as quais constam, de cima para baixo: "Fita/HD/DVD/CD"; "SDRAM/DDR/DRAM"; "Cache L3"; "Cache L2" "Cache L1"; e "Registradores". Ao lado esquerdo, paralelo a cada linha, há os valores "-1.000.000' "-30"; "-20"; "-10"; "2-3"; e "Latência 0". No lado direito, na base, estão escritas "Memória Secundária" e "Memória Principal". Fim da descrição.

Uma vantagem dos dispositivos de armazenamento secundário é que eles possuem grande capacidade de armazenamento e os dados são guardados em meio persistente, portanto, preservados quando se retira a fonte de energia do dispositivo (Tanenbaum, 2009).

Gerência de Memória

O que todo programa deseja é dispor de maneira infinita, rápida, não volátil e a um baixo custo de uma memória que pudesse conter todo seu conteúdo. A função do SO é abstrair a hierarquia e latência das memórias existentes em um modelo útil, e, então, gerenciar essa abstração (Tanenbaum, 2009).

A parte do SO que gerencia parcialmente a memória do computador é denominada Gerenciador de Memória. Entre suas tarefas, podemos citar a de alocar memória para os processos quando estes precisam, a de liberar memória quando um processo termina e tratar a tarefa de tratar do problema de *swapping*. O gerenciador de memória é o componente do SO que se preocupa com o esquema de organização da memória e com a estratégia do gerenciamento.

O Gerenciador de Memória tem como premissa otimizar a utilização da memória principal. Isso era fácil nos primeiros SO, que simplesmente usavam a memória física disponível (1960-1970). Nesse período, não era possível rodar mais de um programa por vez, pois o SO entregava a gerência dos endereços ao programa que estava em execução e o programa do usuário podia acessar os endereços que estavam alocados para o próprio SO.

Importante

Programas tendem a se expandir e ocupar toda a memória disponível. Lei de *Parkinson*.

Para permitir que mais de um programa execute simultaneamente, o SO deve resolver dois problemas: o de segurança e o de realocação dos endereços atribuídos ao programa. A solução é criar um espaço de endereçamento para cada processo. Assim, o endereço 28 na memória de um processo é diferente do endereço 28 em outro processo.

Uma forma simples de criar os espaços é, via *hardware*, criar dois registradores um registrador-base e outro registrador-limite e cada processo precisará somar o registrador-base em seus endereços e comparar o resultado com o registrador-limite a fim de observar se está dentro do espaço de endereçamento reservado para ele. Apenas o SO poderá atribuir valores para esses registradores.

O primeiro PC 8088 utilizava parte desta solução denominada de realocação dinâmica, pois tinha apenas o registrador-base (Tanenbaum, 2009). O maior problema dessa solução é a fragmentação da memória, conforme ilustrado na Figura 3 que acaba desperdiçando memória, por isso, surgiu a necessidade de tentar outra solução.

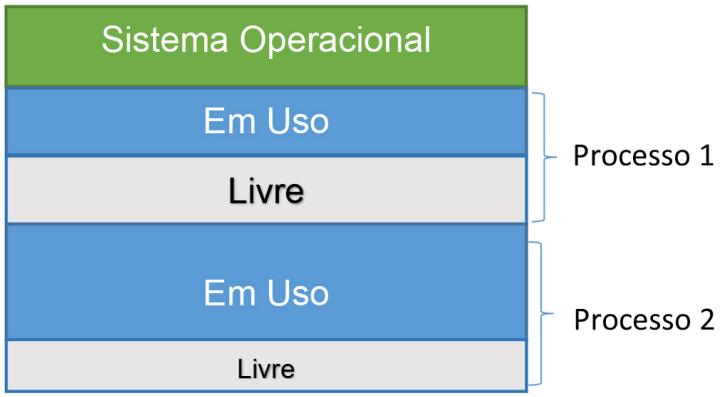

Figura 3 – Fragmentação da memória
Fonte: Elaborada pelo conteudista
#ParaTodosVerem: a imagem mostra um quadro dividido em 5 linhas. A primeira, na cor verde, contém o termo "Sistema Operacional"; a segunda, na cor azul, com o termo "Em uso"; e, embaixo, uma linha cinza, com o termo "Livre". As duas últimas linhas seguem a mesma estrutura. Fim da descrição.

O grande problema dessa solução que adotava limites fixos de memória era a fragmentação e a limitação de processos ativos, visto que o SO dividia toda a memória quando era carregado e alocava partes iguais aos processos. Adotando-se um número variável para o espaço da memória um número maior de processos podem ser alocados. Essa solução demanda do Gerenciador de Memória um algoritmo de alocação da memória livres. Desse modo, os novos processos sempre poderão ser alocados.

Elaboraram-se três métodos de seleção de uma região livre: Melhor Escolha (*best fit*) , Pior Escolha (*worst fit*) e Primeira Escolha (*first fit*). O método melhor escolha coloca o processo na menor região livre; a pior escolha coloca o processo com a maior região livre; e a primeira escolha aloca a primeira região livre para o processo.

Veja a Figura 4 que exibe um processo que demanda 14 *kbytes* em uma memória com 5 processos em uso.

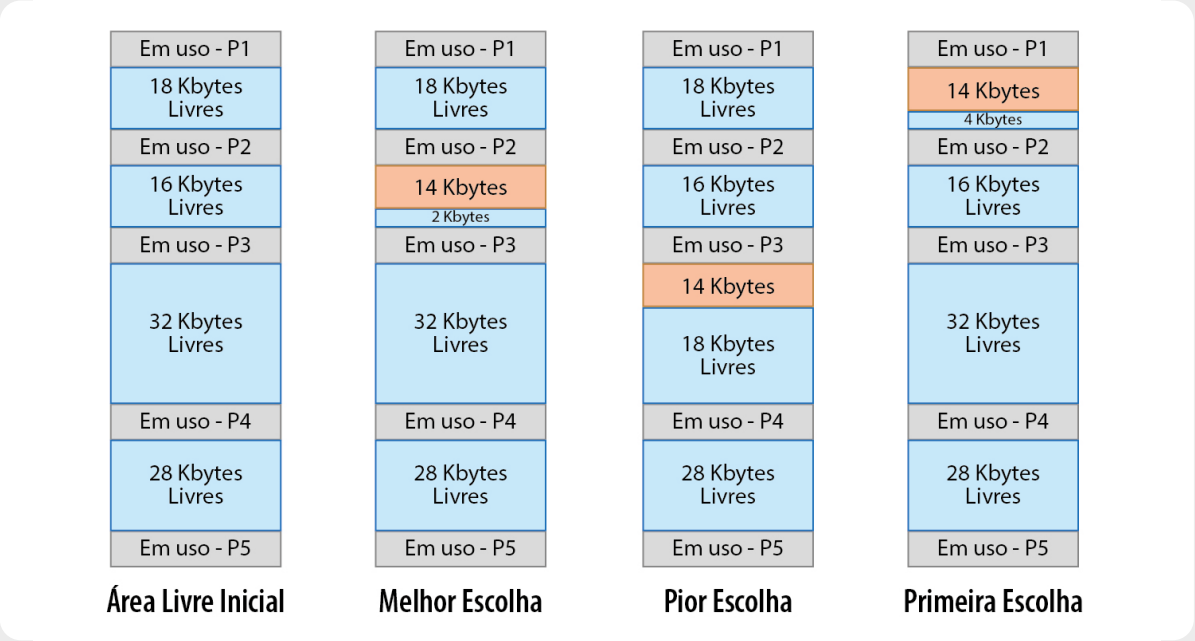

Figura 4 – Gerência de Memória - Alocação da Região Livre
Fonte: Elaborada pelo conteudista

#ParaTodosVerem: a imagem mostra quatro colunas lado a lado, cada uma dividida em linhas nas cores cinza, azul claro e laranja; nelas, há diferentes quantidades de memórias. As colunas estão classificadas em "Área Livre Inicial", "Melhor Escolha", "Pior Escolha" e "Primeira Escolha". Fim da descrição.

O uso do algoritmo de seleção de memória livre melhor escolha deixa o menor resto, porém após um longo período de processamento deixará espaços muito pequenos na memória para serem úteis a algum processo. O algoritmo pior escolha deixa o maior espaço após cada locação, mas tende a espalhar as porções não usadas sobre área não contínua de memória tornando difícil alocar grandes processos.

O algoritmo primeira escolha tende a ser um meio termo com característica adicional de fazer com que os espaços vazios migrem para o final da memória. O gerenciador de memória deverá, com qualquer uma dessas abordagens, manter uma lista de chamada "Lista Livre" dos blocos disponíveis com informações sobre o seu tamanho. Antes de retornar um bloco à lista livre, o gerenciador de memória deve verificar se o bloco liberado está próximo a outros blocos, a fim de que possam ser combinados, formando um bloco maior.

Memória Virtual

A maioria do SO enfrenta a dificuldade da falta de memória física para a demanda dos processos ativos. Por exemplo, facilmente ao ligar o seu computador com o SO *Windows* ou SO Linux, uns 50 – 200 MB serão alocados. A estratégia de manter um processo na memória principal até sua finalização recebe o nome de *swapping*

que pode ser traduzida por troca ou permuta. O Gerenciador de Memória precisará trocar os dados que estão na memória principal (RAM) por dados que estão na memória secundária (HD) e utilizar a memória secundária como uma extensão da memória principal.

Com o tempo, o Gerenciador de Memória foi aperfeiçoado a fim de antecipar a necessidade das trocas, permitindo que vários processos permanecessem na memória principal ao mesmo tempo. Os novos algoritmos de *swapping* trocam um processo somente quando outro precisar daquele espaço de memória, visto que a memória secundária é mais lenta. Com uma quantidade de memória principal suficiente, esses sistemas reduzem muito o tempo gasto nas trocas. Os sistemas de *swapping* do início da década de 1960, levaram a uma nova estratégia — a memória virtual com paginação.

A estratégia da Memória Virtual é permitir que a memória física e a secundária sejam combinadas em uma nova e única memória. A ideia básica é que cada programa tenha seu espaço de endereçamento dividido em blocos chamados de páginas. Cada uma destas é uma série contínua de endereços. Elas têm seus endereços mapeados principalmente na memória física, de forma que ao acessar um endereço que tenha o equivalente em memória física o sistema apenas passa o endereço físico correspondente.

Caso o endereço seja da parte gravada na memória secundária, o sistema irá solicitar a troca de forma que os dados que estão na memória física serão gravados na memória secundária e os dados da memória secundária serão carregados para a memória física. Quando o processo finalizar as trocas, o sistema informa o endereço físico correspondente (Deitel, 2005).

Glossário

- **Espaço de Endereçamento Virtual**: são os endereços que o programa pode referenciar;
- **Espaço de Endereçamento Físico**: são os endereços reais de memória;
- **Tabela de Páginas**: relaciona os endereços virtuais com os endereços físicos.

A paginação tem como premissa que existe memória secundária disponível e suficiente para manter o programa completo com seus dados. A cópia do programa na memória secundária pode ser considerada o original; enquanto que as partes trazidas da memória principal, de vez em quando, podem ser consideradas cópias.

Quando modificações são feitas na memória principal, elas devem ser refletidas na memória secundária. O espaço de endereçamento virtual é dividido em páginas de tamanhos iguais. Sempre que um processo solicitar o acesso a um endereço, ele está solicitando por meio de um endereço virtual, o qual precisa ser traduzido em um endereço físico.

Isso acontece com tanta frequência, que solicitar ao processador fazer a tradução custaria na performance, de modo que o sistema de memória virtual necessita ter um *hardware* para este propósito especial.

Há a unidade de gerenciamento de memória (*Memory Management Unit* – MMU) que mapeia rapidamente endereços virtuais para endereços reais e o mecanismo de tradução dinâmica de endereços (*Dynamic Address Translation* – DAT) que convertem endereços virtuais em físicos durante a execução. A tradução de um endereço de memória virtual para um endereço físico é feita como na Figura 5. O sistema mantém uma tabela de mapas de blocos para cada processo.

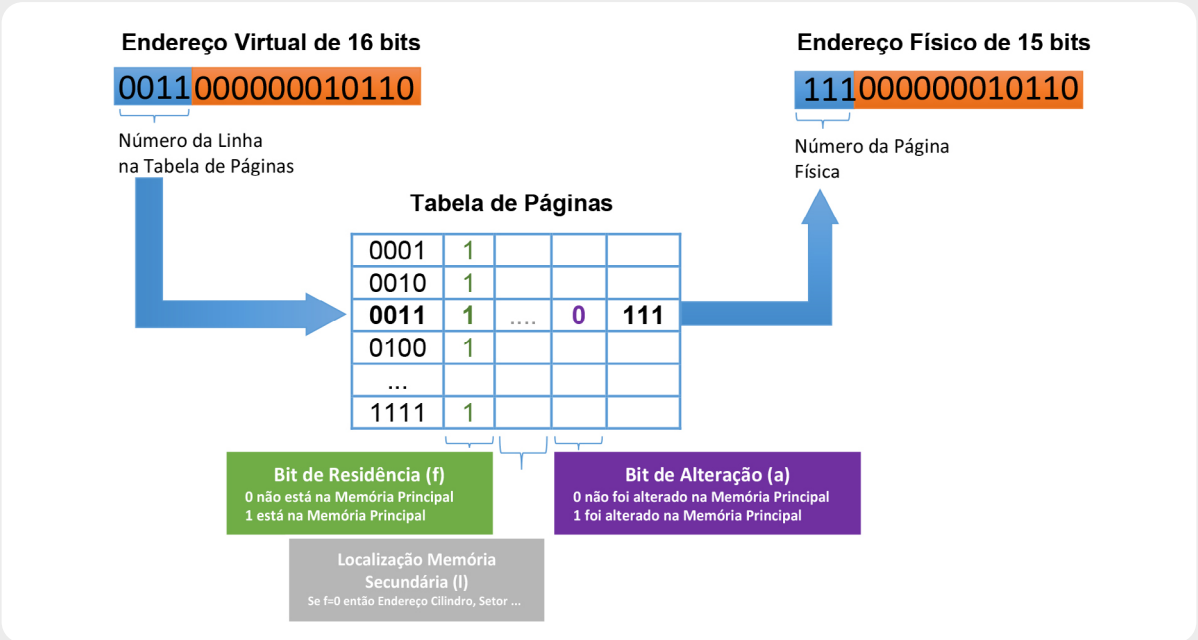

Figura 5 – Tradução de memória
Fonte: Elaborada pelo conteudista

#ParaTodosVerem: a imagem mostra um diagrama formado por uma tabela e três blocos. A tabela, ao centro, apresenta a tabela de páginas, abaixo dela há um bloco verde, referente a "*Bit* de Resistência (f), um bloco roxo, referente a "*Bit* de Alteração (a)" e um bloco cinza, referente a "Localização Memória Secundária (T)". Nas extremidades direita e esquerda, acima da tabela, há dois endereços: "Endereço Virtual de 16 bits" e "Endereço Físico de 15 *bits*". Fim da descrição.

O uso dessa solução pode levar a tabelas muito extensas e tornar a MMU muito lenta. Para ilustrar: de 32 *bits* de endereços e páginas de 4K tem-se 1 milhão de entradas na tabela. Para solucionar tal problema, a ideia básica é manter todas as tabelas na memória, optando pelo uso de dois apontadores e um deslocamento. Assim, o primeiro apontador é do Diretório (p) e o segundo (t) aponta para os quadros e o terceiro para o deslocamento (d) (Deitel, 2005).

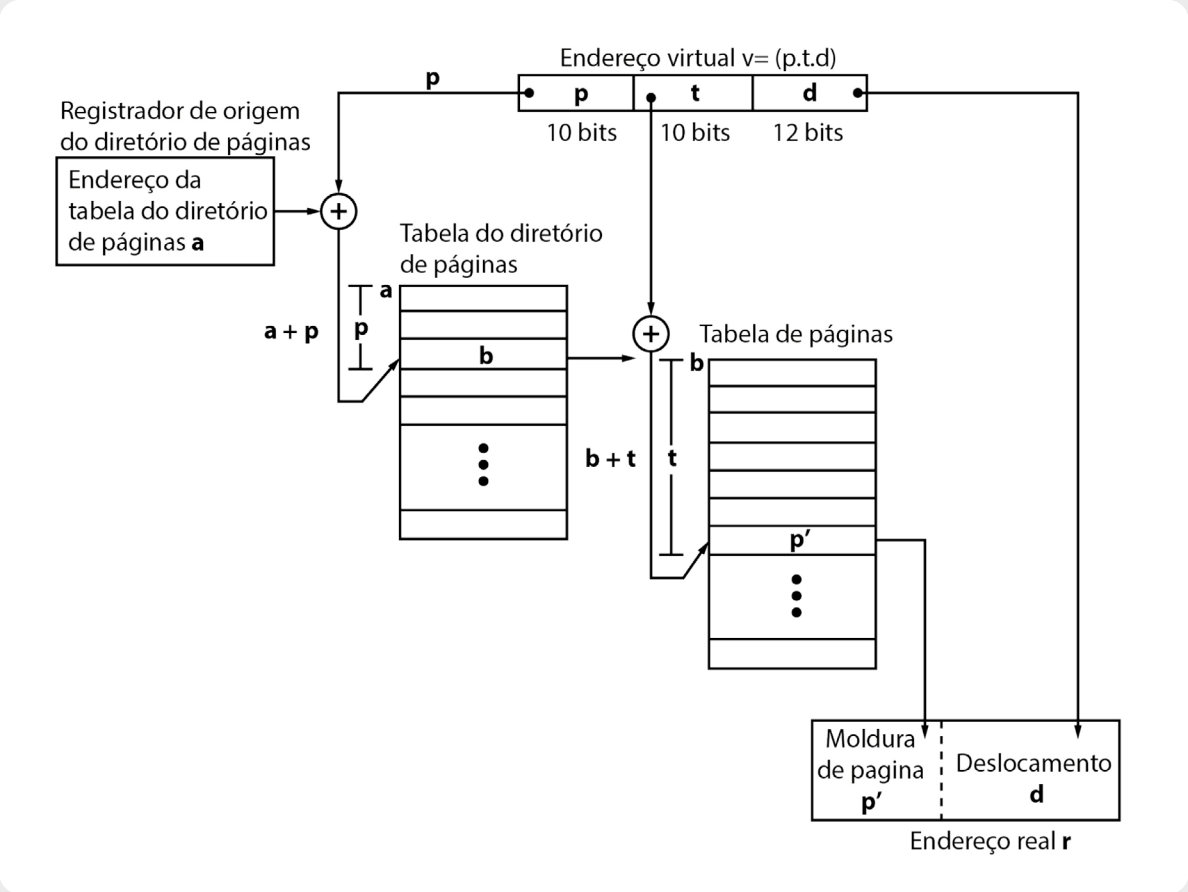

Figura 6 – *Memory Management Unit* MMU

Fonte: Elaborada pelo conteudista

#ParaTodosVerem: a imagem mostra um diagrama formado por linhas pretas finas. Ao centro, há dois quadros divididos em linhas, lado a lado, referentes a Tabelas de páginas, a elas estão ligados três outros quadros, o de cima, referente a "Endereço virtual"; outro na extremidade superior esquerda, referente a "Registrador de origem do diretório de páginas"; e o último quadro, na extremidade inferior direita. Fim da descrição.

Segmentação

A segmentação parte do princípio que um programa pode ser divido em dados e instruções e que estes podem ser armazenados em blocos chamados segmentos. Os segmentos não precisam ser do mesmo tamanho, nem ocupar posições adjacentes na memória principal. Um segmento que corresponda a um *array* e tão grande quanto o *array*. O segmento gerado para conter um código é do tamanho do código.

O sistema gerenciador de memória segmentado mantém na memória principal apenas os segmentos necessários para a execução em um determinado instante. Um processo pode executar enquanto suas instruções e dados estiverem localizados na memória principal. Se o processo referenciar um segmento que não está na memória principal, o gerenciador deverá recuperar o segmento solicitado. Um segmento que chegar poderá ser alocado para qualquer área disponível na memória principal que for grande suficiente para contê-lo (Deitel, 2005).

Há muitas estratégias para implementar a tradução de endereços de segmentação, já que um sistema é capaz de empregar mapeamento direto, associativo ou mapeamento combinado/associativo. Algumas vantagens da segmentação são:

- Facilidade de compilação e ligações entre os procedimentos separados em segmentos;
- A mudança do tamanho de um segmento não afetará os demais;
- A segmentação facilita partilhar dados entre vários processos;
- O programador tem ciência do conteúdo do segmento e pode protegê-lo.

A desvantagem é que os segmentos dependem dos processos, mas os blocos de memória são recursos de *hardware* e seu tamanho é dependente do sistema. Uma solução para este problema é a combinação de segmentação com paginação.

Material Complementar

Sites

Com as Máquinas Virtuais (VM) que você instalou nas unidades anteriores, inicie no SO *Windows* a gerência de memória virtual. O SO *Windows* apresenta uma interface amigável para gerenciar Memória Virtual. Pesquise os comandos, teste e experimente modificar e desabilitar a Memória Virtual.

Memória Virtual
https://bit.ly/46J2jgV

Comandos para Alterar o Tamanho
https://bit.ly/3FxLJ7S

Atividades de Fixação

1 – O que representa a memória em um contexto de computação?

a. A velocidade de processamento da CPU.
b. A capacidade de armazenar informações permanentemente em um disco rígido.
c. O número de núcleos em uma CPU.
d. A capacidade de armazenar temporariamente dados e instruções para processamento pela CPU.
e. A quantidade de armazenamento disponível na nuvem.

2 – O que é memória virtual em sistemas de computação?

a. Uma forma de armazenamento permanente, como um disco rígido, usado para armazenar backups de dados.
b. Uma área de armazenamento temporária que armazena dados exclusivamente na memória RAM.
c. Um sistema de gerenciamento de arquivos em nuvem que armazena todos os dados on-line.
d. Um mecanismo que combina a memória RAM com o armazenamento em disco para aumentar a capacidade de armazenamento temporário.
e. Uma tecnologia que permite a virtualização de CPUs para executar diversas instâncias de um sistema operacional.

Atenção, estudante! Veja o gabarito desta atividade de fixação no fim deste conteúdo.

Referências

DEITEL, H. M. **Sistemas Operacionais**. 3. ed. São Paulo: Pearson Prentice Hall, 2005.

TANENBAUM, A. S. **Sistemas Operacionais Modernos**. 3. ed. São Paulo: Pearson Prentice Hall, 2009.

Gabarito

Questão 1

d) A capacidade de armazenar temporariamente dados e instruções para processamento pela CPU.

Justificativa: em computação, a memória representa a capacidade do sistema de armazenar temporariamente dados e instruções que são necessárias para processamento pela Unidade Central de Processamento (CPU). Isso inclui uma memória Memória de Acesso Aleatório (RAM), que é usada para armazenar dados temporários e instruções que estão sendo usadas pelo computador.

Questão 2

d) Um mecanismo que combina a memória RAM com o armazenamento em disco para aumentar a capacidade de armazenamento temporário.

Justificativa: a memória virtual é um mecanismo usado em sistemas de computação para aumentar a capacidade de armazenamento temporário, combinando a memória RAM com o armazenamento em disco. Isso permite que o sistema utilize espaço em disco como extensão da memória RAM, tornando possível executar programas maiores do que caberiam na memória RAM física.