Estruturas de Sistemas Computacionais

Por que estudar a arquitetura de sistemas computacionais?

Talvez porque o comportamento de um sistema operacional está ligado aos mecanismos de E/S de um computador.

ou

O sistema operacional deve garantir uma operação correta do sistema computacional. Para que os programas dos usuários não interfiram em uma operação correta do sistema, o hardware deve dispor de mecanismos apropriados.

Operação de Sistemas Computacionais

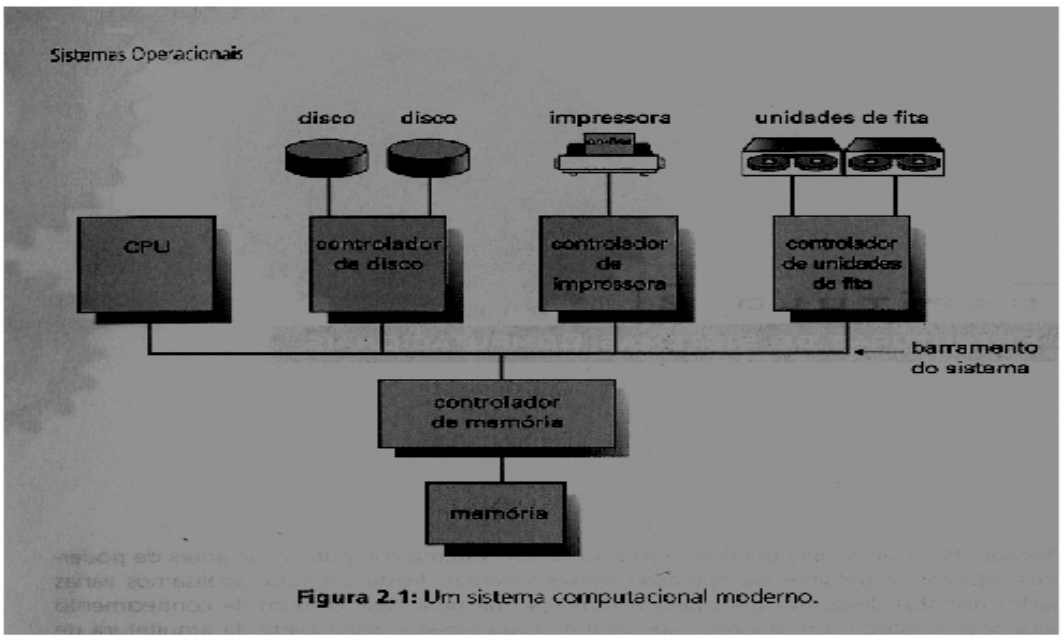


Figura 2.1: Um sistema computacional moderno.

Fonte:Sistemas Operacionais Conceitos – Silberschatz; Galvin

Um sistema computacional moderno, consiste em uma CPU e vários controladores de dispositivos conectados por um barramento comum que oferece acesso a uma memória compartilhada.

- Cada controlador é responsável por um tipo específico de dispositivo.
- A CPU e os controladores de dispositivos podem ser executados simultaneamente, competindo por ciclos de memória. Para assegurar um acesso ordenado á memória compartilhada, **existe um controlador de memória**, **cuja função é sincronizar os acessos a ela.**

Agora que vocês já sabem tudo sobre sistema computacional, será que é só ligar o computador e pronto.

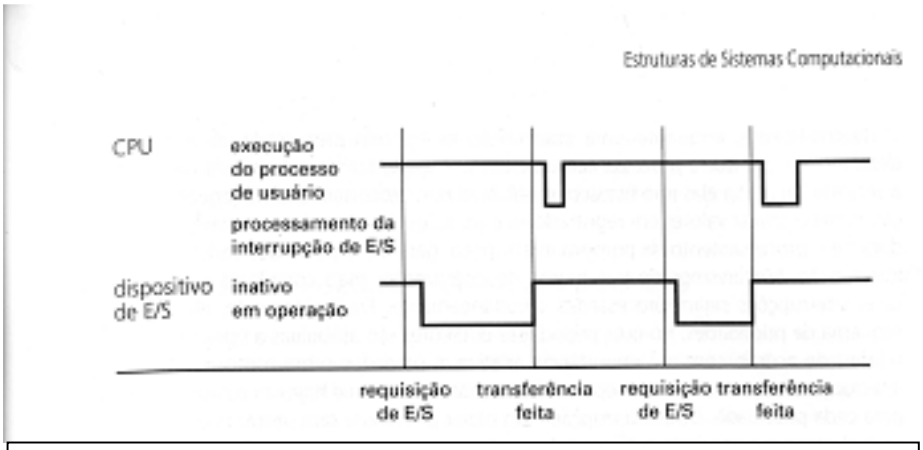
Não, é preciso existir um programa inicial a ser executado, ou programa de bootstrap.

- **Define os valores iniciais necessários ao funcionamento do sistema:**
  - **Valores dos registradores da CPU.**
  - **Valores utilizados pelos controladores de dispositivos.**
  - **Valores de determinadas posições da memória.**
  - **Deve ainda saber como carregar o sistema operacional e como iniciar a execução deste.**
    - **Para isso, deve localizar e carregar na memória o núcleo do sistema operacional.**

Nota Importante

O sistema Operacional inicia a execução de seu primeiro processo e espera pela ocorrência de algum evento.

- ✓ **A ocorrência de um evento é em geral sinalizada por uma interrupção de hardware ou de software (trap).**
  - **O hardware pode causar uma interrupção a qualquer momento, enviando um sinal para a CPU, normalmente por meio do barramento do sistema.**
  - **Os softwares podem causar uma interrupção pela execução de uma operação especial: uma chamada ao sistema.**
  - **Para cada uma dessas interrupções existe uma rotina responsável pelo tratamento do evento que causou a interrupção, chamada de rotina de tratamento de interrupção.**
  - **Quando ocorre uma interrupção, a CPU para o que está fazendo e transfere imediatamente a execução para uma posição fixa, predeterminada para essa interrupção. Essa posição contém normalmente o endereço inicial da rotina de tratamento dessa interrupção. Essa rotina é executada e, quando termina, a CPU retorna a execução do processo computacional que havia sido interrompido.**



Cronograma de interrupções para um único processo realizando operação de saída de dados.

Mecanismo de interrupções

Constitui parte importante da arquitetura de um computador.

- Ao ocorrer uma interrupção, o controle deve ser transferido para a rotina de tratamento de interrupção apropriada.
- As interrupções devem ser processadas rapidamente e, como existe um número predefinido de possíveis interrupções, pode ser usada uma tabela com endereços iniciais das rotinas de interrupção.
- A rotina de interrupção é chamada por meio da tabela, sem necessidade de uma rotina intermediária.
- A tabela com endereços de tratadores de interrupção é armazenada em posições iniciais da memória(por volta das primeiras cem posições).
- Cada rotina de tratamento, correspondente a cada tarefa de cada dispositivo do sistema, tem seu endereço inicial armazenado em uma das entradas da tabela. Essa tabela de endereços, ou vetor de interrupções, é então indexada por um valor determinado por um número de dispositivo e uma identificação da requisição ou evento que provocou a interrupção, fornecendo o endereço da rotina de tratamento da interrupção correspondente a esse dispositivo.
- MS-DOS e o UNIX utilizam esse mecanismo para o início do tratamento de interrupções.
- O mecanismo de tratamento de interrupções deve também armazenar o endereço da instrução que foi interrompida.
- Enquanto uma interrupção está sendo processada, as outras ficam desabilitadas, até que o processamento dessa interrupção termine. Se não

ficassem desabilitadas, o processamento da segunda interrupção poderia gravar valores em registradores e posições de memória que estivessem em uso durante o processamento da primeira interrupção, gerando perda dos dados da primeira interrupção.

- Mecanismos de tratamento de interrupções mais complexos permitem que duas interrupções sejam processadas simultaneamente.
    - Usam um esquema de prioridades, no qual prioridades diferentes são atribuídas a tipos diferentes de tarefas, **de acordo com sua importância relativa.**
    - Uma interrupção com maior prioridade será processada mesmo que uma de menor prioridade esteja sendo processada.
    - Sistemas operacionais modernos são baseados no uso de interrupções.

Nota Importante	**Quando ocorre uma interrupção, o hardware transfere o controle para o sistema operacional. O sistema operacional preserva o estado da CPU, armazenando o conteúdo dos registradores e o contador de instruções. Depois, determina o tipo de interrupção que ocorreu, o que pode acabar exigindo o uso de polling – comunicação com todos os dispositivos de E/S para determinar qual foi o serviço requisitado – ou pode ser o resultado natural do uso de um mecanismo de vetor de interrupções.**

Estrutura de Sistemas de E/S

Um controlador de dispositivos possui uma área local de armazenamento e um conjunto de registradores de propósito específico. Ele é responsável pela transferência de dados entre os dispositivos periféricos que ele controla e sua área de armazenamento local. O tamanho dessa área varia de um controlador para outro, dependendo do dispositivo que estiver sendo controlado.

❖ Interrupções das operações de E/S

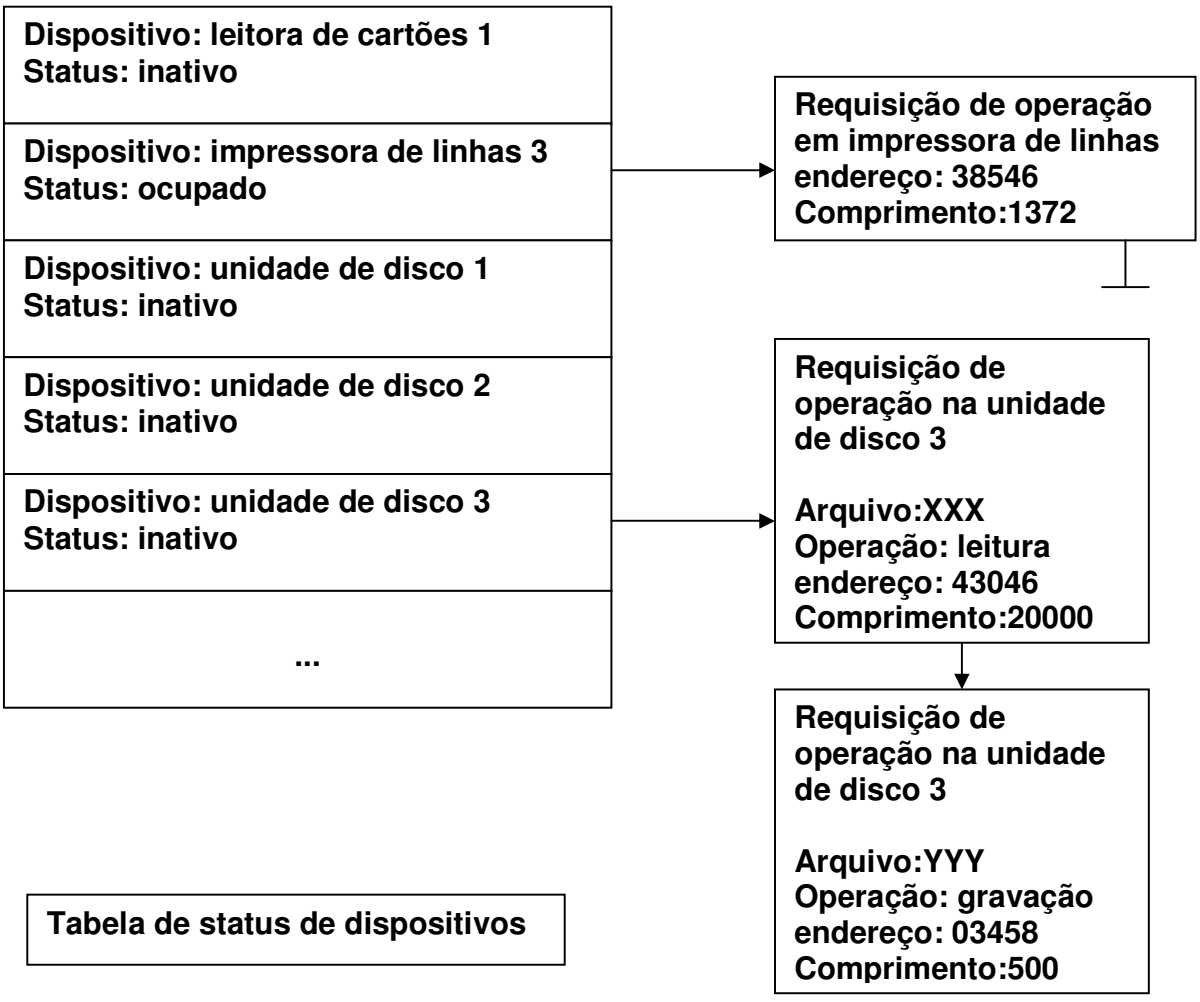
- Para iniciar uma operação de E/S, a CPU carrega determinados valores em registradores apropriados no controlador do dispositivo correspondente à operação de E/S. Esse controlador, por sua vez, examina o conteúdo desses registradores para determinar a ação a ser tomada. Por exemplo, se ele encontra uma requisição de leitura de dados, inicia a transferência de dados do dispositivo para sua área de armazenamento local. Quando todos os dados tiverem sido transferidos, o controlador do dispositivo informa à CPU que a operação está concluída, o que provoca uma interrupção.

- Quando a operação de E/S se inicia, há duas possibilidades:
  - Síncrona:
    - A operação de E/S é iniciada e, ao término desta, o processo do usuário volta a ser executado.
  - Assíncrona:
    - O programa do usuário continua sendo executado sem esperar que a operação de E/S seja concluída.

A espera pela realização de uma operação de E/S pode ocorrer de duas maneiras:

- Instrução especial de espera(wait), que mantém a CPU inativa até que ocorra a próxima interrupção.
- Equipamentos que não têm essa instrução podem ficar em um ciclo de espera:

É preciso manter informações sobre requisições simultâneas para a realização de operações de E/S. Para isso, o sistema operacional usa uma tabela, que contém uma entrada para cada dispositivo de E/S, chamada de tabela de status de dispositivo.



Cada entrada da tabela indica o tipo, o endereço e o estado(inativo, ocupado ou não funcionando) de um dispositivo. Se o dispositivo estiver ocupado, devido a alguma requisição, o tipo dessa requisição e outros parâmetros são armazenados na entrada da tabela correspondente a esse dispositivo. Como outros processos podem fazer requisições a um mesmo dispositivo, o sistema operacional mantém ainda uma fila de espera.

Em um sistema com compartilhamento de tempo, o sistema operacional pode iniciar a execução de um outro processo que estiver pronto para ser executado.

- **Estrutura de acesso direto à memória**

Considere um controlador simples de um terminal de entradas de dados. Quando uma operação de leitura de uma linha desse terminal for realizada, o primeiro caractere digitado será enviado ao computador. Ao receber esse caractere, o dispositivo de comunicação assíncrona ao qual o terminal está conectado interromperá a CPU. Quando a requisição para interromper a CPU ocorrer, ela estará prestes a executar alguma instrução(Se a CPU estiver no meio da execução de uma instrução, geralmente a interrupção ficará pendente até que a CPU conclua a execução dessa instrução).

O endereço da instrução interrompida é armazenado e o controle é transferido para a rotina apropriada de tratamento da interrupção.

Essa rotina de tratamento armazena os valores de todos os registradores da CPU que serão usados por ela. Ela verifica se há quaisquer condições de erro que possam ter resultado da última operação de entrada. Armazena então o caractere recebido do terminal em uma área de armazenamento local. A rotina de interrupção deve ainda ajustar o valor dos ponteiros, de modo a assegurar que o próximo caractere lido seja armazenado na próxima posição da área de armazenamento local. A rotina de interrupção modifica a seguir o valor de uma determinada posição na memória, indicando para as outras partes do sistema operacional que dados novos foram recebidos. As outras partes são responsáveis por processar os dados contidos na área de armazenamento local e pela transferência dos caracteres para o programa que está solicitando a entrada. A rotina de tratamento de interrupção restaura os valores de todos os registradores que foram usados por ela. O controle da execução volta em seguida para a instrução interrompida.

Se os caracteres estiverem sendo digitados em um terminal de 9600 bauds, o terminal pode aceitar e transferir aproximadamente um

caractere a cada milissegundo. Uma rotina de tratamento de interrupção pode requerer dois microssegundos por caractere, deixando então 998 de cada 1000 microssegundos para uso da CPU e para tratamento de outras interrupções.

Devido a essa disparidade, operações de E/S assíncronas normalmente recebem uma prioridade de interrupção baixa, permitindo que outras interrupções mais importantes sejam processadas primeiro ou mesmo que a rotina de interrupção que está sendo executada no momento seja interrompida por uma outra. **Entretanto, um dispositivo de alta velocidade, como uma fita, um disco ou uma rede de comunicações, pode ser capaz de transmitir informações a velocidades próximas da de acesso à memória; a CPU precisaria de 2 microssegundos para responder a cada interrupção, com interrupções chegando a cada 4 microssegundos. Isso não deixaria muito tempo para a execução de processos.**

Nota Importante	**Dispositivos de E/S de alta velocidade usam acesso direto à memória (DMA – direct memory access).** **Depois de preparar áreas de armazenamento, ponteiros e contadores para o dispositivo de E/S, um controlador de DMA transfere um bloco de dados inteiro diretamente para ou de sua própria área de armazenamento local para a memória, sem intervenção da CPU.** **Apenas uma interrupção é gerada por bloco, em vez de uma interrupção por byte, gerada para dispositivos de baixa velocidade.**

Um programa de usuário, ou o próprio sistema operacional, pode solicitar transferências de dados. O sistema operacional aloca uma área para armazenamento de dados(vazia no caso de entrada e cheia para saída), de um pool de áreas de memória reservadas para armazenamento. Uma parte do sistema operacional chamada de rotina de controle de dispositivo(device driver) atribui valores para os registradores do controlador de acesso direto à memória, para que os endereços de origem e destino e o tamanho do bloco de dados apropriados sejam usados para a transferência. O controlador de DMA é então instruído a iniciar a operação de E/S. Enquanto o controlador de DMA está efetuando a transferência dos dados, a CPU está livre para efetuar outras tarefas.

Como a memória em geral só consegue transferir uma palavra de cada vez, o controlador de DMA rouba ciclos de memória da CPU. Esses ciclos roubados podem reduzir a velocidade de processamento da CPU enquanto uma transferência com acesso direto à memória está sendo realizada. O controlador de DMA interrompe a CPU quando a transferência termina.

Estrutura de Sistemas de Armazenamento

Para serem executados, os programas devem estar armazenados na memória. A memória principal é a única grande área de armazenamento à qual o processador pode acessar diretamente.

O ideal seria que os programas e dados ficassem permanentemente na memória principal. Isso não é possível pelas duas seguintes razões:

- Muito pequena para armazenar permanentemente todos os programas e dados.
- Dispositivo de armazenamento volátil: os valores armazenados na memória são perdidos quando a energia acaba ou é desligada.

A maioria dos sistemas computacionais conta com uma forma de memória secundária como uma extensão da memória principal. O principal requisito a ser satisfeito pelas memórias secundárias é a capacidade de armazenar de maneira permanente grandes quantidades de dados.

O dispositivo mais comum de memória secundária é o disco magnético, que permite o armazenamento tanto de dados quanto de programas. A maioria dos programas ficam armazenados em um disco até que sejam carregados na memória. Muitos programas usam o disco tanto como fonte quanto destino de informações de processamento. Por isso, um gerenciamento apropriado da memória de discos é de importância fundamental em um sistema computacional.

A estrutura de armazenamento básica, registradores, memória principal e discos magnéticos, são apenas parte das estruturas de armazenamento, ainda temos, memória cachê, CD-ROM, fitas magnéticas, entre outras.

As principais diferenças entre os vários sistemas de armazenamento são relativas a velocidade, custo, tamanho e volatilidade.

❖ Memória Principal

A memória principal e os registradores existentes internamente no processador são os únicos dispositivos de armazenamento aos quais a CPU tem acesso direto. Portanto, qualquer instrução em execução e quaisquer dados que estejam sendo usados pelas instruções devem estar em desses dispositivos de armazenamento de acesso direto. Se os dados não estiverem na memória, eles precisam ser ali colocados para que a CPU possa realizar suas operações sobre esses dados.

Para permitir um acesso mais eficiente aos dispositivos de E/S, muitas arquiteturas de computadores dispõem de uma forma de E/S chamada **E/S mapeada em memória(memory-mapped I/O).** Nesse caso, uma faixa de endereços de memória é reservada e associada aos registradores de dispositivos de E/S. Ler e escrever nesses endereços de memória faz com que os dados sejam transferidos dos registradores dos dispositivos para a memória e vice-versa. Esse método é apropriado para dispositivos com pequeno tempo de resposta, como controladores de vídeo. Nos PCs IBM, cada posição na tela é mapeada em uma posição de memória. Mostrar um texto na tela é quase tão fácil quanto escrevê-lo nas posições correspondentes mapeadas em memória.

❖ Discos Magnéticos

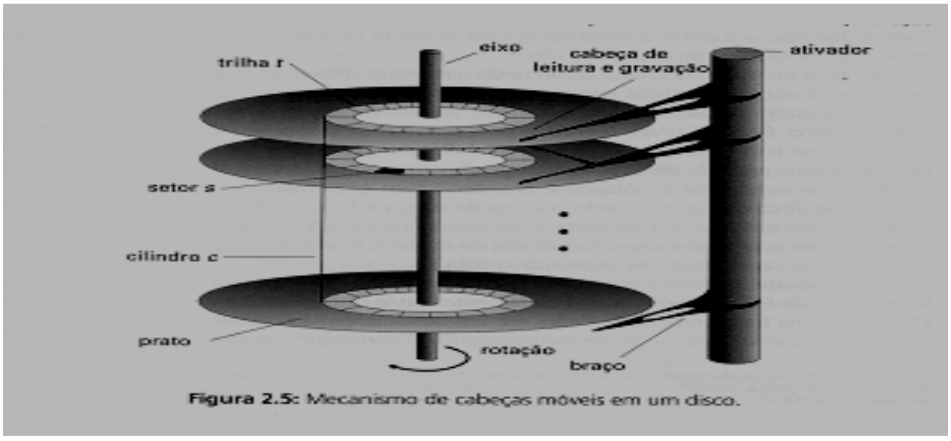


Figura 2.5: Mecanismo de cabeças móveis em um disco.

Grande parte da memória secundária de computadores modernos é formada por discos magnéticos. Os dados são armazenados quando as informações são registradas magneticamente sobre os pratos.

As transferências de dados em um barramento são realizadas por processadores eletrônicos especiais chamados de controladores. O controlador hospedeiro é o controlador que fica na extremidade do

barramento ligada ao computador. Um controlador de disco é montado no interior de cada unidade de disco. Para realizar uma operação de E/S em um disco, o computador envia um comando ao controlador hospedeiro, em geral usando portas de E/S mapeadas em memória. Em seguida, o controlador hospedeiro envia o comando por meio de mensagens para o controlador do disco, e este opera o hardware da unidade de disco para executar o comando. Os controladores de discos geralmente têm uma memória cachê embutida. As transferências de dados na unidade de disco ocorrem entre a memória cachê e a superfície do disco e as transferências de dados para o hospedeiro, em altas velocidades, entre a memória cachê e o controlador hospedeiro.

- **Fitas Magnéticas**

  Os sistemas computacionais mais antigos usavam fitas magnéticas como meio de armazenamento secundário. Embora uma fita magnética seja um meio relativamente permanente e possa armazenar uma grande quantidade de dados, o tempo de acesso a uma fita é grande se comparado ao da memória principal. Acessos aleatórios a fitas magnéticas são cerca de mil vezes mais lentos do que no caso de discos magnéticos, de modo que as fitas não são muito úteis como memória secundária. As fitas são usadas principalmente para armazenamento de cópias de dados e programas (backup), para armazenamento de informações pouco usadas e como um meio para transferir informações de um sistema para outro.

Hierarquia de Armazenamento

A ampla variedade de sistemas de armazenamento em um sistema computacional pode ser organizada de forma hierárquica, de acordo com critérios de velocidade e custo. Os níveis mais altos contêm dispositivos mais caros, mas também mais rápidos. Descendo na hierarquia, o custo por bit geralmente diminui, enquanto o tempo de acesso em geral aumenta.

Muitos dos primeiros dispositivos de armazenamento, incluindo memórias de fita de papel e de anéis de ferrite, estão relegados a museus, agora que as memórias que usam fitam magnéticas e semicondutores tornaram-se mais rápidas e mais baratas.

Além da velocidade e do custo dos vários sistemas de armazenamento, há também a questão da volatilidade do armazenamento. Uma memória volátil perde seu conteúdo quando a energia para o dispositivo acaba ou é desligada.

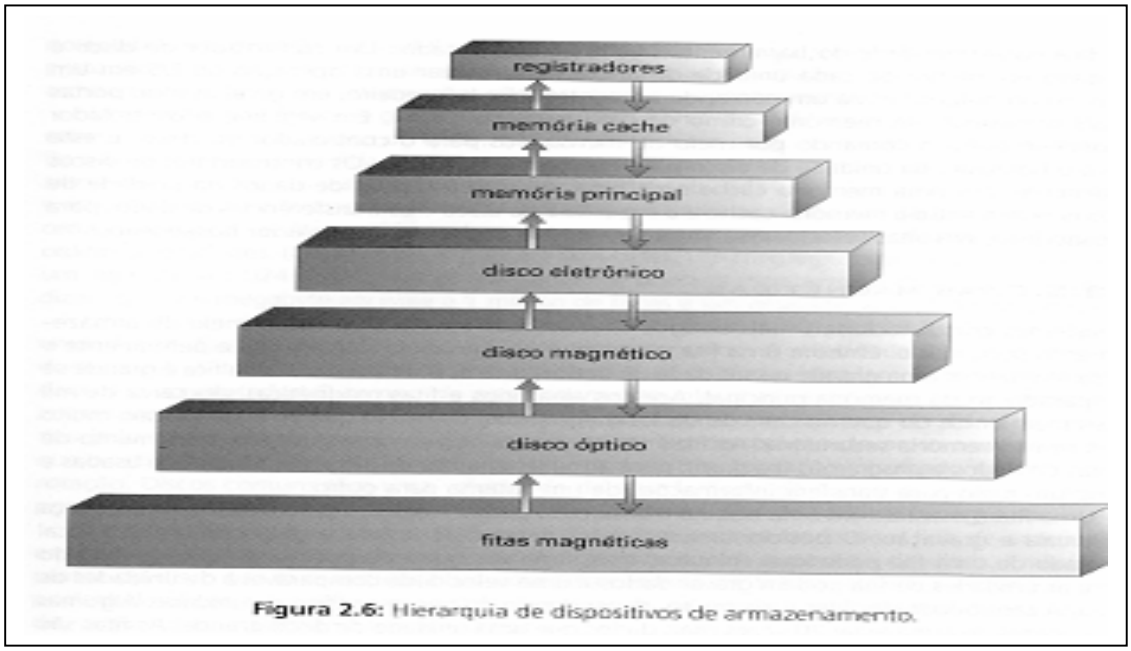


Figura 2.6: Hierarquia de dispositivos de armazenamento.

Os sistemas de armazenamento posicionados acima dos vários discos são voláteis, ao contrário daqueles situados abaixo da memória principal.

O projeto de um sistema completo de memória deve levar em conta todos esses fatores: usar apenas o necessário de memórias mais caras e fornecer o máximo possível de memória mais barata, não-volátil. Memórias cachê podem ser instaladas para diminuir o efeito de diferenças de desempenho entre as memórias sempre que houver uma disparidade grande de tempo de acesso ou na taxa de transferência entre dois componentes.

- **Princípio das Memórias Cachê**

  O uso de memórias cachê envolve um princípio importante dos sistemas computacionais. As informações são normalmente mantidas em algum sistema de armazenamento. Um valor que representa uma informação que desejamos usar é em geral copiado temporariamente em um sistema de armazenamento mais rápido - a memória cache. Quando precisamos de uma determinada informação, primeiro verificamos se ela pode ser encontrada na memória cachê. Em caso afirmativo, usamos diretamente a memória cachê; caso contrário, utilizamos o sistema de armazenamento principal, colocando uma cópia na memória cachê sob a hipótese de que existe uma grande probabilidade de que será necessário usar essa informação novamente.

Estendendo essa visão, podemos ver registradores internos, tais como registradores indexadores, como uma memória cachê de alta velocidade para a memória principal. Um programador, ao implementar algoritmos de alocação de registradores e de substituição de variáveis por registradores, decide quais informações devem ser mantidas em registradores e quais devem ser mantidas na memória principal. Existem também memórias cachê que são totalmente implementadas em hardware. Por exemplo, a maioria dos sistemas possui uma memória cachê para armazenar instruções que podem ser as próximas a serem executadas. Sem essa memória cachê, a CPU precisaria esperar por vários ciclos, enquanto uma instrução estivesse sendo trazida da memória principal. Por razões similares, a maioria dos sistemas tem uma ou mais memórias cachê de dados de alta velocidade na hierarquia de memórias.

Como uma memória cache tem um tamanho limitado, o gerenciamento da memória cache é um problema importante no projeto de sistemas computacionais. Uma seleção cuidadosa do tamanho da memória e da política de substituição dos dados pode fazer com que 80 a 99 por cento de todos os acessos sejam acessos à memória cachê, obtendo-se assim um desempenho extremamente alto.

A transferência de informações entre níveis de uma hierarquia de memórias pode ser explícita ou implícita, dependendo do projeto do hardware e do software de controle do sistema operacional. Por exemplo, a transferência de dados de uma memória cache para a CPU e para os registradores é usualmente função do hardware, sem qualquer intervenção do sistema operacional. Por outro lado, transferências de dados de discos para a memória são normalmente controladas pelo sistema operacional.

- **Coerência e Consistência**

Em uma estrutura de armazenamento hierárquica, valores que representam uma mesma informação podem aparecer em diferentes níveis do sistema de armazenamento. Por exemplo, considere uma variável A que armazena um valor inteiro, residente originalmente em um arquivo B, por sua vez armazenado em um disco magnético. Considere que o valor contido em A vai ser incrementado de 1. A operação de incrementar inicia primeiramente uma operação de E/S para copiar para a memória principal o bloco do disco no qual A reside. Essa operação é seguida por uma possível cópia do valor de A para a memória cachê e por outra cópia desse valor para um registrador interno. Portanto, uma cópia do valor de A aparece em diversos lugares. Como a operação de incrementar acontece no registrador interno, logo após esse incremento o valor de A não é o

mesmo nos vários sistemas de armazenamento. O valor de A torna-se o mesmo em todos os sistemas somente após o novo valor de A ter sido gravado de volta no disco magnético.

Em um ambiente multitarefa, no qual o controle da CPU passa de um processo a outro, deve ser tomado um cuidado extremo para assegurar que, se vários processos desejam acessar o valor contido em A, cada um deles obterá o valor que corresponde à modificação mais recente.

A situação fica mais complicada em um ambiente multiprocessador no qual, além de manter registradores internos, a CPU tem uma memória cache local. Nesse ambiente, uma cópia de A pode existir simultaneamente em diversas memórias cachê. Como as várias CPUs podem todas estar sendo executadas simultaneamente, devemos ter certeza de que uma modificação do valor de A em uma memória cachê é refletida imediatamente em todas as outras memórias cachê em que existe uma cópia do valor de A. Esse problema, chamado de **coerência de memórias cachê**, é normalmente uma função do hardware.

Em um ambiente distribuído, a situação fica ainda mais complexa. Nesse ambiente podem existir várias cópias de um mesmo arquivo em diferentes computadores, localizados em lugares diferentes. Como acessos e atualizações de valores correspondentes a essas várias cópias podem ser feitos simultaneamente, devemos assegurar que, quando uma cópia é atualizada em algum lugar, todas as outras deverão ser atualizadas assim que possível, **consistência**.

Proteção pelo Hardware

Os primeiros sistemas computacionais eram sistemas de um único usuário, operados pelo programador. Quando os programadores operavam o computador a partir de um console, eles tinham um controle completo sobre o sistema. Com o desenvolvimento dos sistemas operacionais, entretanto, esse controle foi dado ao sistema operacional. A começar pelo monitor residente, o sistema operacional passou a realizar muitas das funções(especialmente E/S) que eram anteriormente responsabilidade do programador.

Além disso, para melhorar a utilização do sistema, o sistema operacional passou a compartilhar os recursos do sistema entre os vários programas. Com o mecanismo de spooling, um programa podia estar em execução enquanto outros processos realizavam operações de E/S; o disco armazenava simultaneamente dados de muitos processos. A multiprogramação colocava diversos programas na memória ao mesmo tempo.

Esse compartilhamento tanto melhorou a utilização quanto aumentou os problemas.

- **Quando não havia compartilhamento, um erro em programa podia causar problemas apenas para aquele programa que estava em execução.**

- **Quando há compartilhamento, muitos processos podem ser afetados de forma adversa devido a um erro em um programa.**

Quando não há um mecanismo de proteção contra erros, ou o computador deve executar apenas um processo de cada vez ou todos os resultados estarão sob suspeita. Um sistema bem projetado deve assegurar que um programa incorreto não prejudique a execução de outros programas.

Sempre que ocorrer um erro em programas de usuários, o sistema operacional deverá terminar anormalmente o programa. Essa situação recebe o mesmo código que uma terminação anormal requisitada diretamente no programa do usuário. Uma mensagem de erro apropriada é emitida e é feito um dump de todo o conteúdo da memória, no instante em que ocorreu o erro ou a terminação anormal. Esse dump é normalmente feito em um arquivo para que o usuário possa examina-lo para talvez corrigir ou reiniciar o programa.

- **Operação em Modo Dual**

  Para assegurar uma operação apropriada, devemos proteger o sistema operacional e todos os outros programas e seus dados de qualquer programa incorreto. Todos os recursos compartilhados precisam ser protegidos. A abordagem adotada é fornecer suporte em hardware que permita uma diferenciação entre os vários modos de execução. Necessitamos de no mínimo **dois modos de operação: modo usuário e modo monitor (modo supervisor ou modo previlegiado).**

  Durante a inicialização (boot) do sistema, o modo de operação é o modo monitor. O sistema operacional é então carregado e inicia a execução de processos de usuários no modo usuário. Sempre que ocorre uma interrupção de software ou de hardware, o hardware muda o modo de operação de modo usuário para modo monitor.

  O modo dual de operação nos oferece uma maneira de proteger o sistema operacional e outros programas de programas incorretos. Essa proteção é implementada reservando-se algumas instruções que poderiam causar danos como

instruções privilegiadas. O hardware permite a execução de instruções privilegiadas apenas no modo monitor.

Caso tenha sido feita alguma tentativa de execução de uma instrução privilegiada no modo usuário, o hardware não executa a instrução; em vez disso, trata-a como ilegal e provoca uma interrupção de software que desvia o controle para o sistema operacional.

MS-DOS – foi desenvolvido para plataforma 8088, que não tinha bit indicador de modo de operação e portanto nenhum modo dual.
Versões mais recentes da CPU da Intel, como o 80486, fornecem operação em modo dual.
WindowsNT e OS/2, aproveitam esse recurso para proporcionar maior proteção ao sistema operacional.

- **Proteção às Operações de E/S**

Os programas de usuários podem interromper uma operação normal do sistema por meio da execução de instruções de E/S ilegais, do acesso às posições de memória ocupadas pelo próprio sistema operacional ou pela recusa em transferir o controle da CPU. Podemos usar vários mecanismos para garantir que tais procedimentos não ocorram.

Para evitar que um usuário realize operações ilegais de E/S, definimos todas as instruções de E/S como instruções privilegiadas. Assim, os usuários não podem enviar instruções de E/S diretamente: eles devem faze-lo por meio do sistema operacional. Para que a proteção seja completa, devemos ter certeza de que um programa de usuário não possa em momento algum obter o controle do computador no modo monitor.

- **Proteção da Memória**

Deve-se proteger o vetor de interrupções contra qualquer modificação por programas de usuários. Além disso, devemos proteger contra tais modificações também as rotinas de tratamento de interrupção no sistema operacional.

A modificação de rotinas de tratamento de interrupção provavelmente interferiria em operação apropriada do sistema computacional e de seus mecanismos de armazenamento.

Deve existir um mecanismo para proteção a memória, pelo menos para o vetor de interrupções e para as rotinas de tratamento de interrupção do sistema.

Essa proteção deve ser fornecida pelo hardware.

- **Proteção da CPU**

A terceira peça do quebra-cabeças de proteção é assegurar que o sistema operacional mantenha controle sobre o processo de execução de programas. Não se pode permitir que programas de usuários fiquem em ciclos infinitos de execução, o que faria com que o controle nunca retornasse ao sistema operacional. Para alcançar esse objetivo, podemos usar um temporizador, que pode ser preparado para causar interrupções a cada intervalo de tempo predeterminado.

Um temporizador gerador de interrupções em intervalos variáveis é geralmente implementado por meio de um temporizador gerador de interrupções em intervalos fixos e um contador.

O sistema operacional estabelece um valor inicial para o contador. A cada unidade de tempo determinada pelo temporizador, o valor armazenado no contador diminui. Quando o contador chega a 0, ocorre uma interrupção.

Antes de o controle retornar ao usuário, o sistema operacional certifica-se de que o temporizador está preparando para causar interrupções. Quando o temporizador causa uma interrupção, o controle é transferido automaticamente para o sistema operacional, que pode tratar a interrupção como um erro fatal ou pode alocar mais tempo para o programa.

Instruções que modificam a operação do temporizador são claramente instruções privilegiadas.

Arquitetura Geral do Sistema

- **O desejo de melhorar a utilização de sistemas computacionais levou ao desenvolvimento da multiprogramação e do compartilhamento de tempo, nos quais os recursos do sistema computacional são compartilhados por muitos programas e processos diferentes.**

- **O mecanismo de compartilhamento de tempo ocasionou modificações diretas na arquitetura básica dos computadores, para permitir ao sistema operacional manter o controle sobre o sistema computacional, e principalmente sobre a E/S. Para fornecer uma operação correta, consistente e continuada do sistema como um todo, um controle deve ser mantido sobre a realização de cada operação.**

- **Modo dual.**

- **As instruções de E/S e as instruções para armazenar um valor no temporizador ou em registradores usados para gerenciamento da memória são instruções privilegiadas.**