5.4.2 - Software do Disco - Cache de Rastreamento

★ o tempo requerido para uma operação de *seek* para um novo cilindro é usualmente muito maior que o tempo de transferência ou rotação;

- ... em outras palavras, uma vez que o *driver* tenha posicionado o braço em algum lugar, pouco importa o tempo gasto para ler um setor ou uma trilha inteira.

★ alguns *disk drivers* tiram proveito desta propriedade mantendo secretamente uma cache do rastreamento *(track-at-a-time cache)*, o que não é conhecido pelo *software* independente do dispositivo;

- ... se um setor é demandado e o mesmo encontra-se no cache, nenhuma transferência do disco é requerida.

✘ **cache de rastreamento** - complexidade do *software* e espaço req. de buffer;

✘ **cache de rastreamento** - transferência da cache para o programa que requisitou a operação deve ser feita pela CPU, usando um *loop* programado, ao invés de DMA.

... 5.4.2 - Software do Disco - Cache de Rastreamento

★ alguns controladores aprimoram este processo, e fazem o *track-at-a-time caching* na sua memória interna, transparente para o *driver*;

- ... assim a transferência entre o controlador e a memória pode usar DMA.

✍ Note que ambos controlador e *driver* podem ler ou escrever trilhas inteiras em um único comando, mas que o *software* independente do dispositivo não pode;

- ... a razão para isto é que ele considera o disco como uma sequência linear de blocos, sem considerar como eles são divididos em trilhas e cilindros.

... 5.4.2 - Software do Disco - Manipulação de Erros

★ **erros de seek** - são causados por problemas mecânicos no braço;

- ... mantendo o rastreamento da posição do braço internamente, para realizar um *seek*, o controlador emite uma série de pulsos para o motor do braço, um pulso por cilindro, para mover o braço para o novo cilindro;

- ... quando o braço chega no destino, o controlador lê o número do cilindro (escrito quando da formatação) gerando um erro de *seek* caso o braço esteja no lugar errado.

★ alguns computadores corrigem o erro de *seek* automaticamente, mas outros incluindo os IBM PC's apenas atribuem um bit de erro e abandonam o resto para o *driver*;

- ... o *driver* manipula este erro pela emissão de um comando **recalibrate**, que move o braço tão longe ele possa ir, e reinicia o controlador interno do cilindro corrente em 0, o que usualmente soluciona o problema;

- ... caso contrário, o dispositivo deve ser reparado.

... 5.4.2 - Software do Disco - Manipulação de Erros

★ resumidamente, o controlador é um pequeno computador especializado, contendo *software*, variáveis, *buffers*, e ocassionalmente *bugs*;

- ... algumas vezes uma sequência não usual de eventos, tal como uma interrupção sobre um dispositivo juntamente com um comando **recalibrate** em outro, dispara o *bug* e causa o controlador entrar em *loop* ou perder-se do estava fazendo.

- ... projetistas de controladores usualmente consideram a pior situação e provêem um pino no *chip*, o qual, quando em nível alto, força o controlador esquecer sua tarefa corrente, reiniciando a si mesmo;

- ... se tudo isso falhar, o *disk driver* pode reiniciar o controlador. Se isto também for em vão, o *driver* imprime uma mensagem e termina sua operação.

... 5.4.2 - Software do Disco - Manipulação de Erros

★ **erros de programação** - ocorrem quando o *driver* diz ao controlador para executar uma operação de *seek* em um cilindro não existente;

- ... ler de um setor não existente, usar uma cabeça não existente, ou transferir de ou para uma posição de memória inexistente.

★ **erros checksum** - transientes são causados por poeira no ar entre a cabeça e a superfície do disco provocando erros de leitura;

- ... em muitos casos eles podem ser eliminados pela repetição da operação algumas vezes. Se o erro persiste, o bloco deve ser marcado como defeituoso.

... 5.4.2 - Software do Disco - Manipulação de Erros

★ **evitando blocos defeituosos** - uso de um programa especial que toma a relação destes blocos como entrada, e cria um arquivo contendo todos os blocos defeituosos;

- ... uma vez que este arquivo tenha sido criado, para o alocador do disco parecerá que estes blocos estão ocupados, não os alocando para outros arquivos;
- ... como o arquivo de blocos defeituosos nunca é lido, os blocos defeituosos permanecerão inertes no disco.

★ **tarefa árdua** - alguns controladores "inteligentes" reservam algumas poucas trilhas, não normalmente disponíveis para programas do usuário;

- ... quando um disco é formatado, o controlador determina quais blocos são defeituosos e automaticamente substitui por uma trilha de reserva;
- ... como a tabela contendo esse mapeamento é mantida na memória interna do controlador e no disco, a substituição é transparente para o *driver*.

... 5.4.2 - Software do Disco - Algoritmos de Escalonamento

★ quando existe vários dispositivos, uma tabela de requisições pendentes deve ser mantida para cada dispositivo separadamente;

- ... quando algum dispositivo está desocupado, um *seek* deve ser emitido para mover os seus braços para o cilindro onde será necessário (assumindo que o controlador permita *seeks* sobrepostos);

- ... quando a transferência corrente termina, um teste pode ser feito para verificar se algum dos dispositivos está posicionado no cilindro correto;

- ... se um ou mais estão, a próxima transferência pode ser iniciada no dispositivo que já está no cilindro correto (otimização);

- ... se nenhum está posicionado, o *driver* deve emitir novos *seeks* sobre os dispositivos que já completaram a transferência, e esperar a próxima interrupção para examinar em qual dispositivo o posicionamento do braço se completou.

5.4.2 - Software do Disco - Manipulação de Erros

★ Discos RAM não têm preocupação sobre *seek* ou na otimização rotacional: em algum instante todos os blocos podem ser lidos ou escritos sem nenhum movimento físico;

★ uma outra área na qual discos RAM são consideravelmente mais simples que discos rotativos é a manipulação de erros. Discos rotativos são submetidos por uma larga variedade de erros. Alguns dos mais comuns são:

- erros de programação — requisição para setor não existente;

- erro de *checksum* transiente — causado por sujeira na cabeça;

- erro de *checksum* permantente — bloco do disco fisicamente danificado;

- erro de *seek* — enviar o braço para o cilindro 6, mas ele vai para o 7;

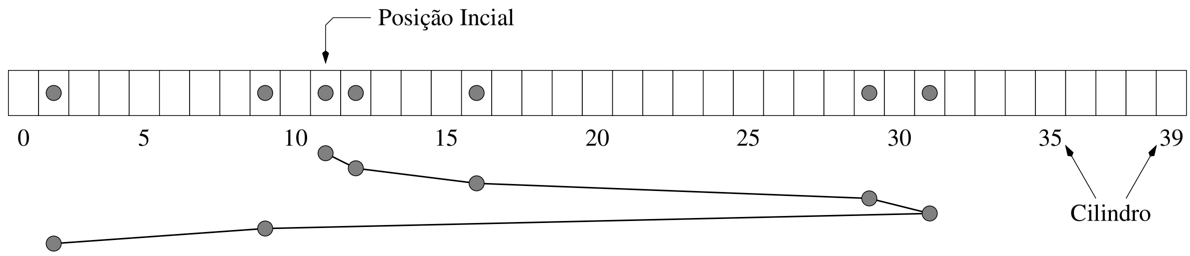
- erro de controlador —recusa do controlador em aceitar comandos.

... 5.4.2 - Software do Disco - Algoritmos de Escalonamento

★ para ilustrar o algoritmo do elevador, assuma que o bit de direção esteja inicialmente em UP e que a ordem na qual os cilindros são servidos é 12, 16, 34, 36, 9, e 1;

- ... esta ordem gera movimentos do braço de 1, 4, 18, 2, 27, e 8, num total de 60 cilindro, ou seja, ligeiramente melhor que SSF, embora seja usualmente pior;

- ... uma propriedade interessante do algoritmo do elevador é que dada uma coleção de requisições, o limite superior para o total de movimentos é fixado: ele é apenas duas vezes o número de cilindros.



... 5.4.2 - Software do Disco - Algoritmos de Escalonamento

★ alguns controladores de disco provêem um modo do *software* para inspecionar o número de setores correntes sob a cabeça, o que possibilita uma outra otimização;

- ... se duas ou mais requisições para o mesmo cilindro estão pendentes, o *driver* pode emitir a requisição para o setor que passará sob a cabeça do próximo cilindro;

- ... note que quando trilhas múltiplas estão presentes num cilindro, requisições consecutivas podem ser conduzidas para diferentes trilhas com nenhuma penalidade;

- ... o controlador pode selecionar alguma das cabeças instantâneamente, uma vez que seleção de cabeça não involve movimento dos braços nem atraso rotacional.

... 5.4.2 - Software do Disco - Algoritmos de Escalonamento

- **Problema no SSF** - suponha que mais requisições estão chegando enquanto as requisições 12, 9, 16, 1, 34 e 36 estão sendo processadas:

- ... por exemplo, se, após chegar ao cilindro 16, uma nova requisição para o cilindro 8 está presente e, assim, terá prioridade sobre o cilindro 1;

- ... se a requisição for para o cilindro 13, o braço irá atender esta requisição (vai p/ o cilindro 13) ao invés de ir para o cilindro 1.

★ com disco muito carregado, o braço tende a permanecer no meio do disco a maior parte do tempo, prejudicando assim as requisições das extremidades;

- ... requisições distantes do meio são em média mais demoradas, colocando o objetivo de mínima resposta no tempo e equitatividade em conflito.

... 5.4.2 - Software do Disco - Algoritmos de Escalonamento

★ **alternativa** - reconciliar os objetivos conflitantes entre a eficiência e equitatividade constitui-se em manter o movimento do braço na mesma direção até não haver mais requisições pendentes naquela direção, e então o movimento do braço é mudado.

★ este algoritmo (*algoritmo do elevador*), requer que o *software* mantenha 1 bit para indicar a direção corrente, *UP* ou *DOWN*;

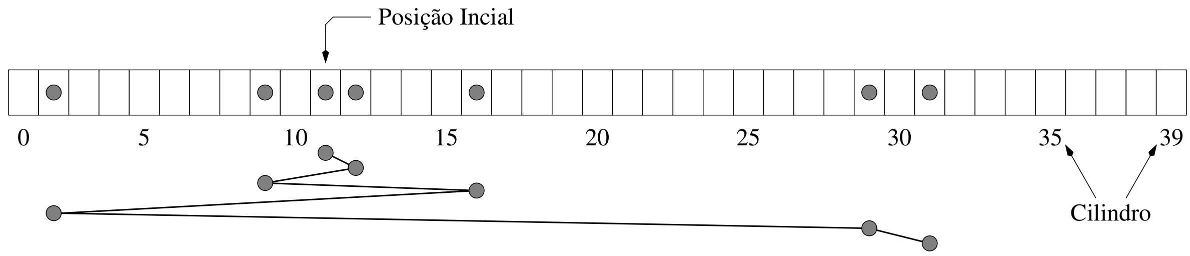
- ... quando a requisição termina, o *disk driver* testa o bit, se é *UP*, o braço é movido para a próxima requisição pendente de posições mais altas, se houver;

- se não há requisições pendentes para posições mais altas, o bit de direção é revertido (mudando o sentido) para a próxima requisição de posição mais baixa, se houver.

... 5.4.2 - Software do Disco - Algoritmos de Escalonamento

★ em um disco de 40 cilindros, uma requisição chega para ler um bloco no cilindro 11. Enquanto a busca para o cilindro 11 está em progresso, novas requisições chegam para os cilindros 1, 36, 16, 34, 9, e 12, nesta ordem;

- ... elas são inseridas na tabela de requisições pendentes, tendo cada cilindro um lista separada conforme apresentado na figura abaixo.



... 5.4.2 - Software do Disco - Algoritmos de Escalonamento

★ quando a requisição corrente termina (cilindro 11), o *disk driver* tem que escolher qual será a próxima requisição a ser atendida;

- ... usando *first come, first served*, ele irá para o cilindro 1, então para o 36, e assim por diante — este algoritmo requer movimentos do braço de 10, 35, 20, 18, 25, e 3, respectivamente, num total de 111 cilindros.

★ alternativamente, a próxima requisição pode ser manuseada a fim de minimizar o tempo de *seek*, ou seja, com a sequência 12, 9, 16, 1, 34, e 36, os movimentos do braço são 1, 3, 7, 15, 33, e 2, num total de 61 cilindros;

- ... este algoritmo, *Small Seek First* (SSF), diminuiu o total de movimentos do braço pela metade, comparado com o FCFS (*First Come, First Served*).

5.4.2 - Software do Disco

★ dentre as características genéricas relacionadas com os *disk drivers*, o tempo de leitura ou escrita de um bloco do disco é um fator determinante:

- ... tempo de *seek* - tempo para mover o braço para o cilindro desejado;
- ... atraso rotacinal - tempo para o setor desejado ficar sob a cabeça de leitura/escrita;
- ... tempo de transferência do bloco.

★ para muitos discos, o tempo de *seek* domina, assim reduzindo-se o tempo de *seek*, podemos melhorar substancialmente o desempenho do sistema.

... 5.4.2 - Software do Disco - Algoritmos de Escalonamento

★ se o *disk driver* aceita uma requisição por vez e a executa nesta ordem, isto é, *First-Come, First-Served (FCFS)*, pouco pode ser feito para otimizar o tempo de *seek* uma vez não há a possibilidade de pré-processamento;

- ... **estratégia alternativa** - é provável que enquanto o braço está executando um *seek* na metade de uma requisição, uma outra requisição de disco pode ter sido gerada por outro processo;
- ... muitos *disk drivers* mantém uma tabela, indexada pelo número do cilindro, com todas as requisições pendentes para cada cilindro, encadeadas juntas numa lista;
- ... para este tipo de estrutura de dados (fila de requisições), podemos melhorar o algoritmo de escalonamento *First-Come, First-Served.*

... 5.4.1 - Hardware do Disco

- ... muitos controladores podem também ler ou escrever em um dispositivo enquanto executam uma busca em um ou mais de um dispositivos, mas nenhum pode ler ou escrever em dois dispositivos no mesmo tempo;

- ... isto decorre do fato de que ler ou escrever requer que o controlador mova bits na faixa de μs, assim uma transferência usa muito de sua capacidade computacional;

- ... a habilidade de realizar duas ou mais buscas ao mesmo tempo pode reduzir o tempo médio de acesso de modo considerável.

★ **IBM PC** - são parâmetros padrões para unidades de disquetes de 5.25", dupla-face/dupla-densidade e capacidade para 360 KB.

... 5.4.1 - Hardware do Disco

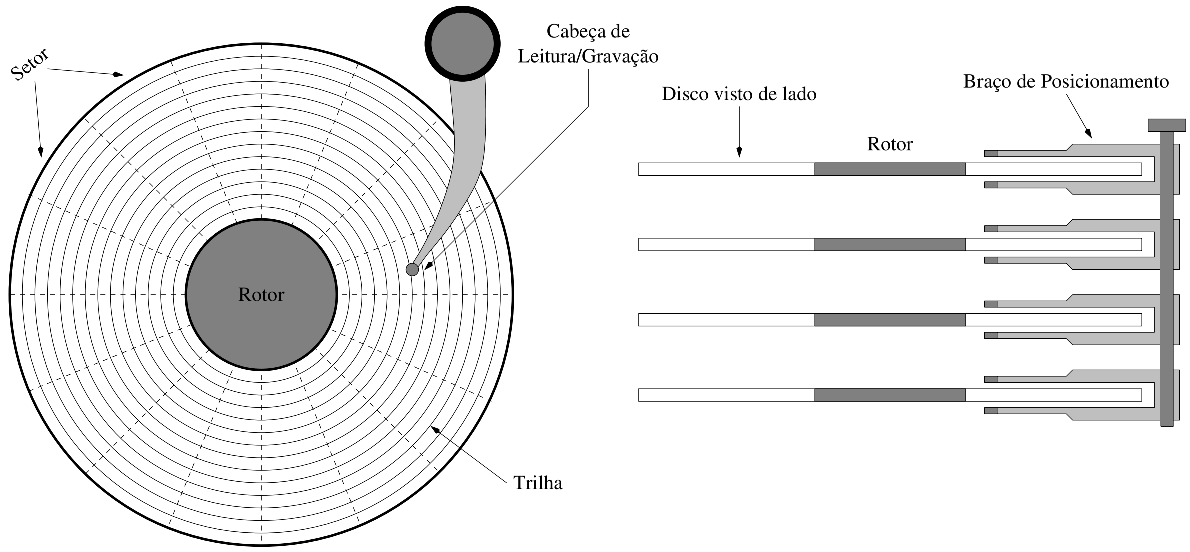
★ os parâmetros de dispositios para discos flexíveis do IBM PC são mostrados abaixo:

- ... esses são os parâmetros padrões para unidades de disquetes de 5.25", dupla-face/dupla-densidade e capacidade para 360 Kbytes.

número de cilindros	40	tempo de busca (cilindros adjacentes)	6 ms
trilhas por cilindro	2	tempo de busca (médio)	77 ms
setores por trilha	9	tempo de rotação	200 ms
total de setores	720	tempo de partida/parada	250 ms
bytes por setor	512	tempo para transferir um setor	22 ms
bytes por disquete	368640		

5.4 - Discos Rotativos

★ elementos básicos de um disco rotacional: cabeça de leitura/gravação, rotor, braço de posicionamento, cilindro, trilha e setor.



5.4.1 - Hardware do Disco

★ todos os discos rotativos são organizados em cilindros, cada qual contendo tantas trilhas quanto cabeças empilhadas verticalmente;

- ... as trilhas são divididas em setores, com um número de setores na circunferência, tipicamente entre 8 a 32 setores por trilha;

- ... todos os setores contém o mesmo número de bytes, embora setores no anel externo do disco sejam fisicamente maiores que aqueles no anel interno, mas isto não permite que o espaço extra seja aproveitado.

★ um aspecto que tem importante implicações no *disk driver* é a possibilidade do controlador fazer buscas *(seek)* em dois ou mais dispositivos ao mesmo tempo, que por sua vez são conhecidas como busca entrelaçada *(overlapped seek)*;

- ... enquanto o controlador e o *software* estão esperando uma busca se completar em um dispositivo, o controlador pode iniciar uma busca em outro;

... 5.3 - Discos RAM

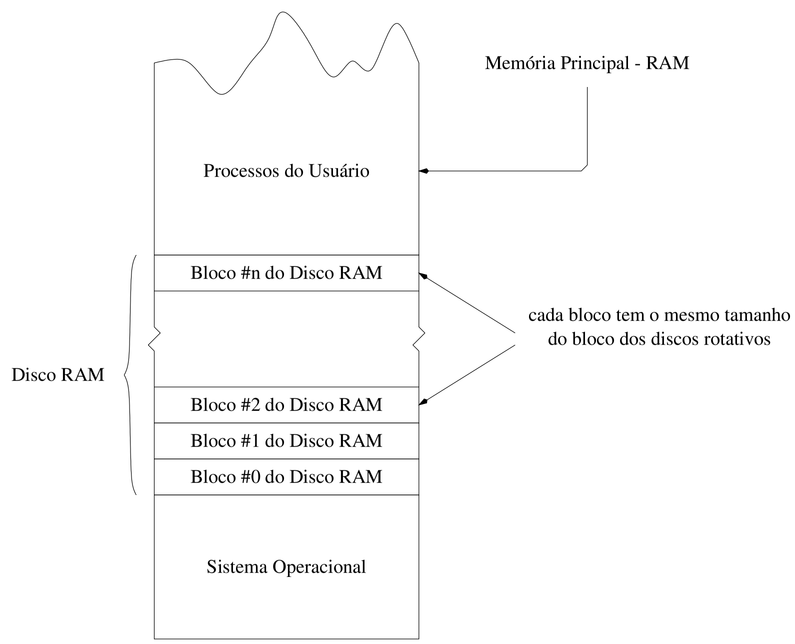
- ★ **idéia por detrás dos discos RAM** - o disco RAM é dividido em n blocos, dependendo de quanto de memória foi alocado para ele. Cada bloco tem o mesmo tamanho que o tamanho do bloco usado em discos rotativos.

- ★ quando o *driver* recebe um bloco para ler ou escrever, ele apenas computa onde na memória de disco da RAM o bloco está, e lê ou escreve nesta posição de memória;

- ... normalmente, a transferência deve ser feita pela chamada de um procedimento em linguagem *assembly*, que copia os dados gerados pelo programa do usuário com a máxima velocidade com a qual o *hardware* é capaz de operar.

... 5.3 - Discos RAM

- ★ região pré-alocada da memória principal utilizada para armazenamento de blocos.



5.3 - Discos RAM

- ✶ dispositivos do tipo bloco são armazenadores que aceitam dois comandos: ler e escrever um bloco que por sua vez é armazenado em memória rotativa, tal como nos discos rígidos e flexíveis;

- ✶ **Disco RAM** - é uma região pré-alocada da memória principal utilizada para o armazenamento de blocos;

- ... apresenta a vantagem de propiciar acesso muito rápido tornando-o apropriado para armazenamento de programas e dados que são utilizados com frequência.

... 5.3 - Discos RAM

- ✍ **montagem de dispositivos** - um dispositivo raiz está sempre presente em uma localização fixa, e sistemas de arquivos removíveis (i.e. discos) podem ser montados (*mounted*) na árvore de arquivos como forma de integrar o sistema de arquivos.

- ✍ em contraste, sistemas como o MS-DOS requerem que o usuário especifique a localização de cada arquivo, de forma explícita como em *B:ARQUIVO*, ou usando certos *defaults* (dispositivos correntes, diretórios correntes, e outros);

- ... com somente um ou dois discos flexíveis, esta carga pode ser gerenciável, mas em grandes computadores, com dezenas de discos, deixar o controle de todos os dispositivos para o usuário é intolerável.

... 5.2.5 - Software no Nível do Usuário

★ uma outra importante categoria é o sistema *spooling*, que não consiste de procedimentos da biblioteca; e sim o modo de negociação com os dispositivos dedicados de entrada/saída em um sistema com multiprogramação.

- ... embora seja fácil permitir que algum processo do usuário abra o arquivo especial associado à impressora, suponha que o processo permaneça com o dispositivo requisitado por várias horas — nenhum outro processo poderá imprimir.
- ... ao invés disso, o que é feito é criar um processo especial, chamado *daemon*, e um diretório especial, chamado *spooling directory*;
- ... para imprimir um arquivo, o aplicativo recebe o arquivo inteiro para ser impresso e o coloca no *spooling directory* para depois o *daemon* transferí-lo do *spooling directory* para a impressora;
- ... protegendo-se os arquivos especiais contra o uso direto por usuários, o problema de se ter alguém monopolizando-os é eliminado.

... 5.2.5 - Software no Nível do Usuário

★ *spooling* não é somente usado para impressoras, mas também em outras situações;

- ... por exemplo, transferência de arquivo na rede frequentemente usa um *daemon* de rede capaz de ler um arquivo do diretório de *spooling* da rede e transmití-lo;

... 5.2.4 - Software Independente do Dispositivo

- ✶ **alocação de blocos nos dispositivos** - uando um arquivo é criado e preenchido com dados, novos blocos de disco têm que ser alocados para o arquivo;

- ... para realizar esta alocação, o sistema operacional precisa de uma lista ou mapa de bits dos blocos livres no disco, mas o algoritmo para localizar um bloco livre é independente do dispositivo e pode ser implementado acima do nível do *driver*.

- ✶ **dispositivos dedicados** - é o sistema operacional que examina a requisição para usar o dispositivo, aceitando dependendo da sua disponibilidade.

- ✶ **gerenciamento de erros** - um erro típico é a não leitura de um bloco ruim;

- ... após o *driver* tentar ler o bloco várias vezes, ele informa ao *software* independente do dispositivo a razão, permitindo o tratamento do erro;‘

- ... se o erro é crítico, o sistema operacional deve apresentar uma mensagem e, eventualmente, terminar sua execução.

5.2.5 - Software no Nível do Usuário

- ✶ embora muito do *software* de E/S esteja embutido no sistema operacional, uma pequena porção dele consiste de bibliotecas ligadas juntamente com programas do usuário, e até mesmo com programas inteiros executando fora do núcleo;

- ... chamadas de sistema, incluindo chamadas do subsistema de entrada/saída, são normalmente feitas por procedimentos da biblioteca;

- ... por exemplo, um programa em C que contém a chamada abaixo descrita, terá a biblioteca **fread** ligada com o arquivo objeto gerado na compilacção;

```
bytes_lidos = fread(buffer, tam_item, n_itens, arquivo);
```

- ... a coleção de todos estes procedimentos da biblioteca é parte desse sistema.

... 5.2.4 - Software Independente do Dispositivo

- ★ a função básica do *software* independente do dispositivo é realizar as funções de entrada/saída que são comuns a todos os dispositivos, e prover uma **interface uniforme** para o *software* do usuário.

- ★ **identificação dos dispositivos** - encarregar do mapeamento simbólico dos nomes dos dispositivos para os seus *drivers* apropriados;

- ... no UNIX, por exemplo, cada *device driver* tem a ele associado um *inode*, o qual armazena informações necessárias à localização do *driver* e do dispositivo associados;

- ... estes dispositivos estão dispostos no diretório */dev* e o usuário tem permissão limitada para operá-los.

- ★ **proteção dos dispositivos** - em muitos microcomputadores não há nenhuma proteção, enquanto que em alguns *mainframes* e superminis, acessos a dispositivos de entrada/saída é completamente proibido a usuários.

... 5.2.4 - Software Independente do Dispositivo

- ★ **software independente do dispositivo** - deve encobrir este fato e prover um tamanho de bloco uniforme para camadas superiores, por exemplo, pelo tratamento de vários setores como um simples bloco lógico;

- ... deste modo, os níveis superiores somente negociam com dispositivos abstratos que usam o mesmo tamanho de bloco, independente do tamanho físico do setor.

- ★ **bufferização** - tanto para dispositivos de blocos como para de caracter;

- ... para dispositivos de bloco, o *hardware* executa escrita e leitura de blocos inteiros, mas o processo do usuário está livre para ler ou escrever unidades arbitrárias;

- ... para dispositivos de caracter, usuários podem escrever dados no sistema mais rápido do que a taxa com que eles são transferidos para o dispositivo físico, necessitando assim de "buferização" (p. ex., teclado).

... 5.2.3 - Drivers de Dispositivos

❷ em outros casos, entretanto, as operações são rápidas, situação esta em que o *driver* não precisa ser bloqueado (requisições não bloqueantes);

- ... por exemplo, o deslocamento da tela em terminais requer apenas escrita de uns poucos bytes nos registradores do controlador; assim, nenhum movimento mecânico é necessário e a operação toda pode se completar em alguns poucos μs.

★ Neste ponto, após a operação ter sido completada, o *driver* deve verificar a ocorrência de erros. Se tudo estiver correto, ele passa os dados (o bloco lido, por exemplo) para a próxima camada do *software* de entrada/saída.

★ Finalmente, ele retorna alguma informação de *status* de erros. Se alguma requisição está na fila, uma delas pode agora ser selecionada e iniciada. Se nenhuma está na fila, o *driver* fica aguardando a próxima requisição.

5.2.4 - Software Independente do Dispositivo

★ Embora alguns dos *softwares* de entrada/saída sejam específicos do dispositivo, uma grande fração deles é independente do dispositivo.

★ O limite exato entre os *drivers* e o *software* independente dos dispositivos é função do sistema, uma vez que algumas funções que são independentes do dispositivo podem atualmente estarem nos *drivers*, por eficiência e por outras razões;

★ ... as funções listadas abaixo são tipicamente implementadas na parte do *software* que é independente do dispositivo:

- interface uniforme para com os *drivers* de dispositivos;
- identificação simbólica dos dispositivos;
- proteção dos dispositivos;
- manipulação de blocos independente dos dispositivos.
- bufferização e alocação de espaço nos dispositivos;
- alocação e liberação de dispositivos dedicados;
- gerenciamento de erros.

... 5.2.3 - Drivers de Dispositivos

✶ em termos gerais, o trabalho de um *driver* é aceitar requisições abstratas de um *software* de mais alto nível, e providenciar para que o pedido seja atendido;

• ... uma típica requisição é ler um bloco – se o *driver* está desocupado no momento, a requisição é aceita, sendo processada imediatamente, caso contrário, esta normalmente entra numa fila de requisições pendentes.

✶ ... por outro lado, os termos abstratos da requisição precisam ser transcritos para ações concretas, por exemplo, para um *disk driver*, isto significa:

- ... informar onde o bloco se encontra no disco;
- ... verificar se o motor do *driver* está girando;
- ... determinar se o braço está no cilindro apropriado, e assim por diante.

✶ resumidamentes, o *driver* deve decidir quais operações do controlador são requeridas e em que sequência estas devem ser disparadas.

... 5.2.3 - Drivers de Dispositivos

✶ uma vez determinado quais comandos emitir ao controlador, este inicia a emissão escrevendo nos registradores do controlador do dispositivo;

• ... alguns controladores podem manusear somente um comando por vez, outros aceitam uma lista de comandos que são carregadas sem a ajuda do sist. operacional.

✶ após o comando ou comandos terem sido emitidos, podem ocorrer duas situações:

❶ em muitos casos o *device driver* deve esperar até que o controlador execute as operações requisitadas, ou seja, a requisição é bloqueante;

• ... se estas operações forem lentas (envolvendo movimentos mecânicos, por exemplo), o *driver* bloqueia até que as operações se completem.

5.2.2 - Manipulação de Interrupções

★ interrupções são eventos complicados de se tratar e devem ser isoladas de modo que apenas uma pequena parte do sistema operacional as manipule;

- ... um meio para isolá-las é bloquear os processos aguardando operações de entrada/saída até que uma interrupção anuncie que a operação se completou.

★ quando a interrupção acontece, a rotina de tratamento correspondente libera o processo bloqueado, o que é conseguido retirando-o da fila referentes àquele evento;

- ... em alguns sistemas isto é conseguido fazendo-se um UP sobre um semáforo, em outros ele fará um SIGNAL sobre a variável de condição no monitor; e ainda em outros, uma mensagem é enviada ao processo bloqueado;

- ... em todos os casos o efeito da interrupção é que o processo que estava previamente bloqueado deverá agora estar habilitado para execução.

5.2.3 - Drivers de Dispositivos

- **driver do dispositivo** - código dependente do dispositivo responsável pela manipulação do mesmo ou uma classe de dispositivos intimamente relacionados.

- ... desta maneira, o *driver* é a parte do sistema operacional que conhece quantos registradores o dispositivo oferece, como enviar comandos escrevendo nesses registradores e como testá-los no tocante às operações;

- ... por exemplo, é de responsabilidade do *driver* do controlador de disco saber para que serve cada um dos registradores;

- ... adicionalmente, é de sua responsabilidade o reconhecimento dos setores, trilhas, cilindros, cabeças de leitura/escrita, motor, fator de entrelaçamento e todos os mecanismos que fazem um disco trabalhar propriamente.

... 5.2.1 - Objetivos do Software de Entrada/Saída

✯ outra característica chave são as transferências por blocos (síncronas) e as manipuladas por interrupção (assíncronas);

- ... muitos dispositivos de entrada/saída são assíncronos: a CPU inicia a transferência e se ocupa de outras atividades até que chegue uma interrupção;

- ... o sist. oper. realiza as operações de forma assíncrona (programação bem mais simples), mas para o usuário tudo se apresentam como uma transferência de blocos.

✯ dispositivos podem ser dedicados ou compartilhados, permitindo que sejam utilizados por muitos usuários ao mesmo tempo;

- ... outros dispositivos, como impressoras, devem ser dedicados a um único usuário até que a operação requisitada seja finalizada; por outro lado estes dispositivos introduzem uma variedade de problemas, como o *deadlock* .

... 5.2.1 - Objetivos do Software de Entrada/Saída

- ... sistemas operacionais devem manipular ambos os dispositivos compartilhados e dedicados de maneira a evitar os problemas introduzidos por eles;

- ... estes objetivos podem ser organizados de maneira clara e eficiente pela estruturação do *software* em quatro camadas:

❶ manipulação de interrupções;

❷ *drivers* de dispositivos;

❸ *software* do sistema operacional independente do dispositivo;

❹ *software* do nível do usuário.

5.2 - Princípios do Software

- **idéia básica** - organizar o *software* em camadas, com as mais baixas escondendo peculiaridades do *hardware* e as mais altas apresentando-se simples para o usuários;
- **conceito chave no projeto do software de entrada/saída** — independência do dispositivo *(device independence)*;
- ... deve ser possível escrever programas que usem arquivos gravados em dispositivos de armazenamento (p. ex., disquete ou em disco rígido), sem a necessidade de modificar o programa para cada tipo de dispositivo;
- ... de preferência, deve ser possível utilizar o programa sem recompilá-lo.

5.2.1 - Objetivos do Software de Entrada/Saída

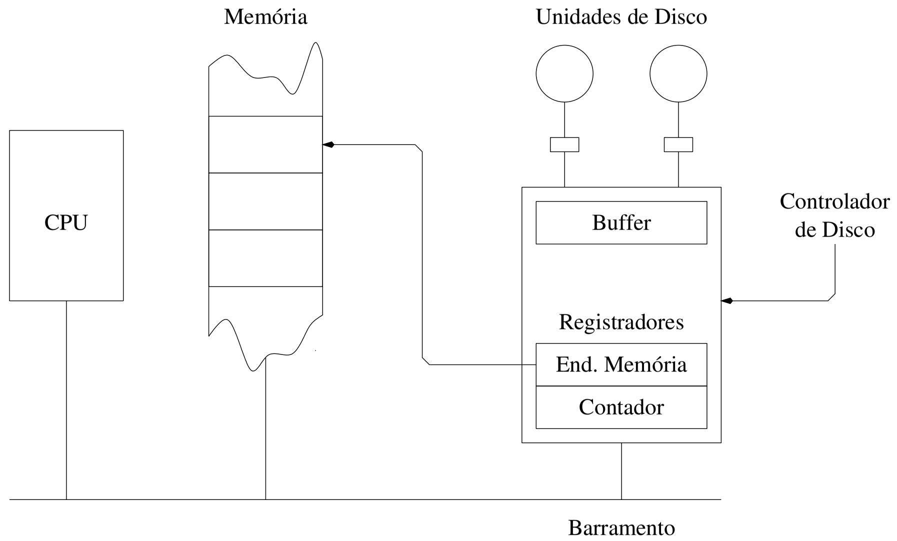
✯ ... relacionado com a indepêndencia do dispositivo está a uniformidade de nome, que deve ser simplesmente uma *string* ou um inteiro que independa do dispositivo;

✯ outra característica importante é a manipulação de erros, que em geral devem ser manipulados o mais próximo possível do *hardware*;

- ... se o controlador encontra um erro, ele deve tentar corrigí-lo se possível, caso contrário, o *driver* do dispositivo deve fazê-lo, talvez apenas tentando ler novamente;
- ... muitos erros são transientes e desaparecem se a operação for repetida, por conseguinte, somente se as camadas mais baixas não conseguirem resolver o problema é que este deve ser apresentado às camadas superiores.

... 5.1.3 - Acesso Direto à Memória

- ... quando utilizado, a CPU fornece duas informações ao controlador (além do endereço do bloco a ser lido): o endereço de memória para onde o bloco deve ser copiado e o número de bytes a serem transferidos.



... 5.1.3 - Acesso Direto à Memória

- após o controlador ter lido o bloco, efetuado o *checksum* e não ter constatado erros de leitura, o próprio controlador copia (via barramento) o primeiro byte (ou palavra) para o endereço de memória suprido pela CPU;

- ... a partir daí, a cópia prossegue incrementado-se o endereço da memória e decrementado-se o contador do DMA (bytes transferidos), até que este se torne zero, quando então uma interrupção é gerada pelo controlador;

- ... quando o sistema operacional reassume a CPU, nenhuma transferência necessita ser efetuada pois o buffer requisitado já se encontra na memória.

... 5.1.2 - Controladores de Dispositivos

- como exemplo, vejamos com o sistema operacional realiza entrada/saída escrevendo comandos nos registradores dos controladores de disquete do IBM PC:

- ... o controlador de disquete do IBM PC aceita 15 diferentes comandos, tais como **read**, **write**, **seek**, **format**, e **recalibrate** (alguns possuem parâmetros);

- ... quando um comando é aceito, o processador pode abandonar o controlador e atender a outra requisição, p. ex. do sistema operacional;

- ... quando completado, o controlador causa uma interrupção com o objetivo de permitir que o sistema operacional tome o controle do processador e teste o resultado da operação realizada;

- ... o processador obtém o resultado e o *status* do dispositivo pela leitura de um ou mais bytes de informação nos registradores do controlador.

5.1.3 - Acesso Direto à Memória

✯ muitos controladores, especialmente os que operam em blocos, suportam acesso direto à memória — DMA *Direct Memory Access*:

❶ ... o controlador lê serialmente o bloco (um ou mais setores) do dispositivo, bit a bit, até que este seja transferido para o buffer interno do controlador;

❷ ... depois, o controlador executa a operação de *checksum* para atestar que o bloco está livre de erros, após o qual uma interrupção é gerada.

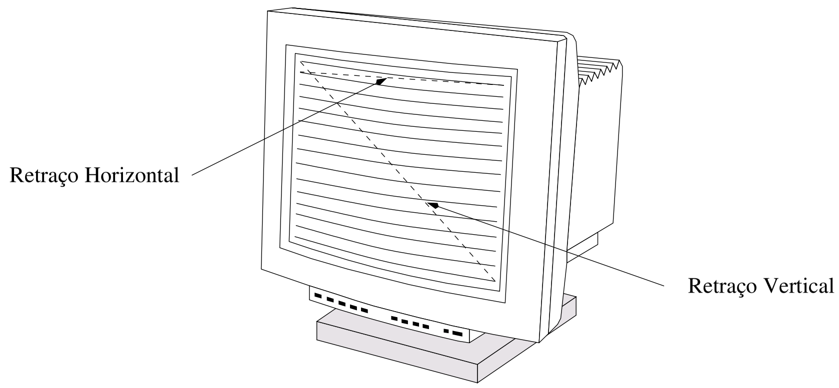
- quando o sistema operacional reassume a CPU, ele pode ler o bloco do buffer do controlador (byte a byte ou palavra a palavra) numa operação cíclica, onde em cada ciclo um byte ou palavra é transferido do controlador para a memória;

- ... o acesso direto à memória foi criado para livrar a CPU de transferir os bytes dos controladores para a memória, o que na verdade consome tempo;

... 5.1.2 - Controladores de Dispositivos

- **controlador para o terminal CRT** - trabalha como um dispositivo serial de bits e em baixo nível, ou seja, lê da memória o byte contendo o caracter a ser exibido, e gera os sinais usados na modulação do feixe do CRT para causar a escrita na tela;

- ... o controlador também gera os sinais para o feixe CRT fazer o *retrace* horizontal após ele ter terminado de esquadrinhar a linha, como também, sinais para fazer o *retrace* vertical após a tela toda ter sido esquadrinhada;



... 5.1.2 - Controladores de Dispositivos

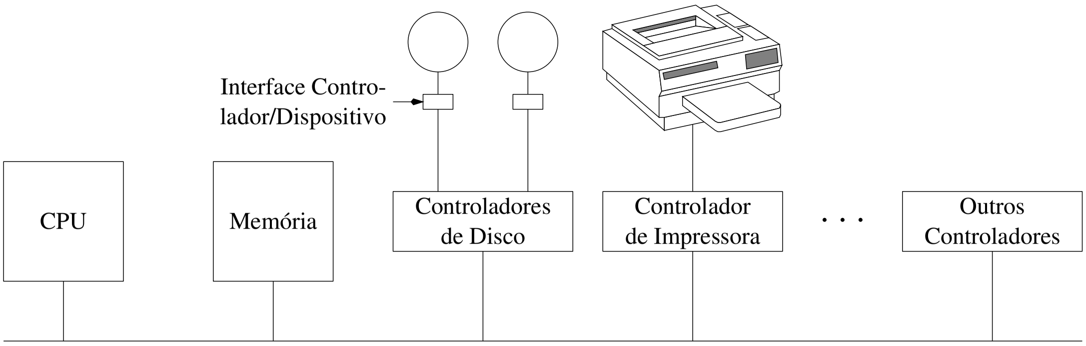
- para comunicação com o processador, cada controlador tem alguns poucos registradores que em alguns computadores são parte do espaço de endereçamento regular;

- ... na seqüência mostra-se os endereços de entrada/saída e os vetores de interrupção alocados para alguns dos controladores do IBM PC.

Dispositivo	Endereço de Entrada/Saída	Vetor de Interrupção
Relógio	040 - 043	8
Teclado	060 - 063	9
Porta Serial Secundária	2F8 - 2FF	11
Disco Rígido	320 - 32F	13
Impressora	378 - 37F	15
Vídeo Monocromático	380 - 3BF	-
Vídeo Colorido	3D0 - 3DF	-
Disco Flexível	3F0 - 3F7	14
Porta Serial Paralela	3F8 - 3FF	12

... 5.1.2 - Controladores de Dispositivos

- normalmente, minicomputadores e microcomputadores usam um barramento único para comunicação entre CPU e os controladores;
- ... *mainframes* frequentemente usam um modelo diferente, no qual múltiplos barramentos e computadores especializados de entrada/saída, chamados canais de entrada/saída, aliviam parte da carga da CPU.



... 5.1.2 - Controladores de Dispositivos

- **interface controlador – dispositivo** - interface de baixo nível;
- ... por exemplo, um disco pode ser formatado com 8 setores de 512 bytes por trilha, mas o que realmente sai do *driver* é uma lista serial de bits;
- ... partindo com um preâmbulo, depois os 4096 bits no setor, e finalmente o *checksum* ou o código de correção de erro;
- ... o preâmbulo é escrito quando o disco é formatado, e contém o número de cilindros e de setores, o tamanho do setor, e outros dados.
- **tarefa do controlador** - converter a lista serial de bits em um bloco de bytes e realizar alguma correção de erro necessária;
- ... o bloco de bytes é tipicamente montado, bit a bit, em um *buffer* mantido no controlador e após verificação do *checksum* e ausência de erros no bloco, ele pode então ser copiado para a memória principal.

... 5.1.1 - Dispositivos de Entrada/Saída

✘ este esquema de classificação não é perfeito e, assim, alguns dispositivos não são enquadrados nele, como por exemplo os relógios que não são endereçáveis por bloco e nem geram ou aceitam filas de caracteres;

- ... tudo o que fazem é gerar interrupções em intervalos regulares;

- contudo, este modelo é geral o suficiente para ser usado como base na construção de um sistema operacional com bom nível de independência dos dispositivos de E/S.

- O sistema de arquivo, por exemplo, negocia apenas com dispositivos de blocos abstratos, e deixa a parte dependente do dispositivo para o *software* de mais baixo nível, chamado *acionadores de dispositivos (device drivers).*

5.1.2 - Controladores de Dispositivos

✯ unidades de entrada/saída consistem tipicamente de componentes mecânicos e eletrônicos:

❶ **componente eletrônico** - chamado de controlador do dispositivo (*device controller* ou *adapter*), normalmente assume a forma de um circuito impresso que pode ser inserido no computador (mini e microcomputadores);

❷ **componente mecânico** - dispositivo propriamente dito;

- ... o cartão do controlador normalmente tem um conector, no qual um cabo condutor do próprio dispositivo pode ser conectado e que em geral é capaz de manusear dois ou mais dispositivos do mesmo tipo.

- ... a distinção entre dispositivo e controlador deve ser ressaltada, já que o sistema operacional vê o controlador, não com o dispositivo.

5.1 - Princípios do Hardware

- importante função do sistema operacional responsável pelo controle de todos os dispositivos de entrada/saída do computador, emissão de comandos para os dispositivos, atendimento as interrupções e manipulação de erros;
- ... também responsável pelo oferecimento de uma interface entre os dispositivos e o resto do sistema, que adicionalmente seja simples e fácil de usar;

✭ Diferentes indivíduos vêem o *hardware* de diferentes maneiras:

- ... engenheiros eletrônicos vêem em termos de *chips*, fios, fontes de potência, motores e todos os outros componentes físicos;
- ... programadores vêem a interface apresentada pelo sistema, os comandos que o *hardware* aceita, as funções que ele suporta e os erros que são reportados;

✭ ... neste tópico nosso interesse se restringe em como o *hardware* é programado.

5.1.1 - Dispositivos de Entrada/Saída

- dispositivos de entrada/saída podem ser grosseiramente divididos em duas categorias: *dispositivos de bloco* e *dispositivos de caracter*.

❶ **dispositivo de bloco** - armazena informações em blocos de tamanho fixo (p. ex., 128 a 1024 bytes), cada um com seu próprio endereço, o que permite leitura ou escrita em cada bloco de maneira independente dos outros;

- ... exemplo típico dessa classe é o disco magnético.

❷ **dispositivo a caracter** - libera ou aceita uma fila de caracteres sem definir uma estrutura de bloco, ou seja, não é endereçável e não aceita operações de busca;

- ... pertencem a esta classe de dispositivos os terminais, impressoras, leitoras ótica e outros dispositivos que não trabalham como os discos.

Cap. 5 - Sistema de Entrada/Saída

5.1 - Princípios de Hardware

- dispositivos de entrada/saída;
- controladores de dispositivos;
- acesso direto à memória.

5.2 - Princípios de Software

- objetivos do software de entrada/saída; manipulação de interrupções;
- drivers de dispositivos; software de entrada/saída independente;
- software no nível do usuário.

5.3 - Discos de Acesso Aleatório - RAM

5.4 - Discos Rotativos - Memória Secundária

- hardware do disco e software do disco.

Cap. 5 - Sistema de Arquivo - Referências Bibliográficas

✭ Eleri CARDOZO; Maurício MAGALHÃES; Luís F. FAINA; "Sistemas Operacionais", Dep. de Eng. de Computação e Automação Industrial, Fac. de Engenharia Elétrica e de Computação, UNICAMP, 1996.

- Cap. 5 - Sistema de Entrada/Saída

✭ A.S. TANEMBAUM; "Modern Operating Systems", Prentice-Hall International Editions, 1984, ISBN: 0-13-637331-3

- Cap. 5 - Input/Output System