Universidade Federal de Uberlândia
Faculdade de Computação
Sistemas Operacionais

Gerenciamento de Entrada e Saída

Prof. Dr. Marcelo Zanchetta do Nascimento

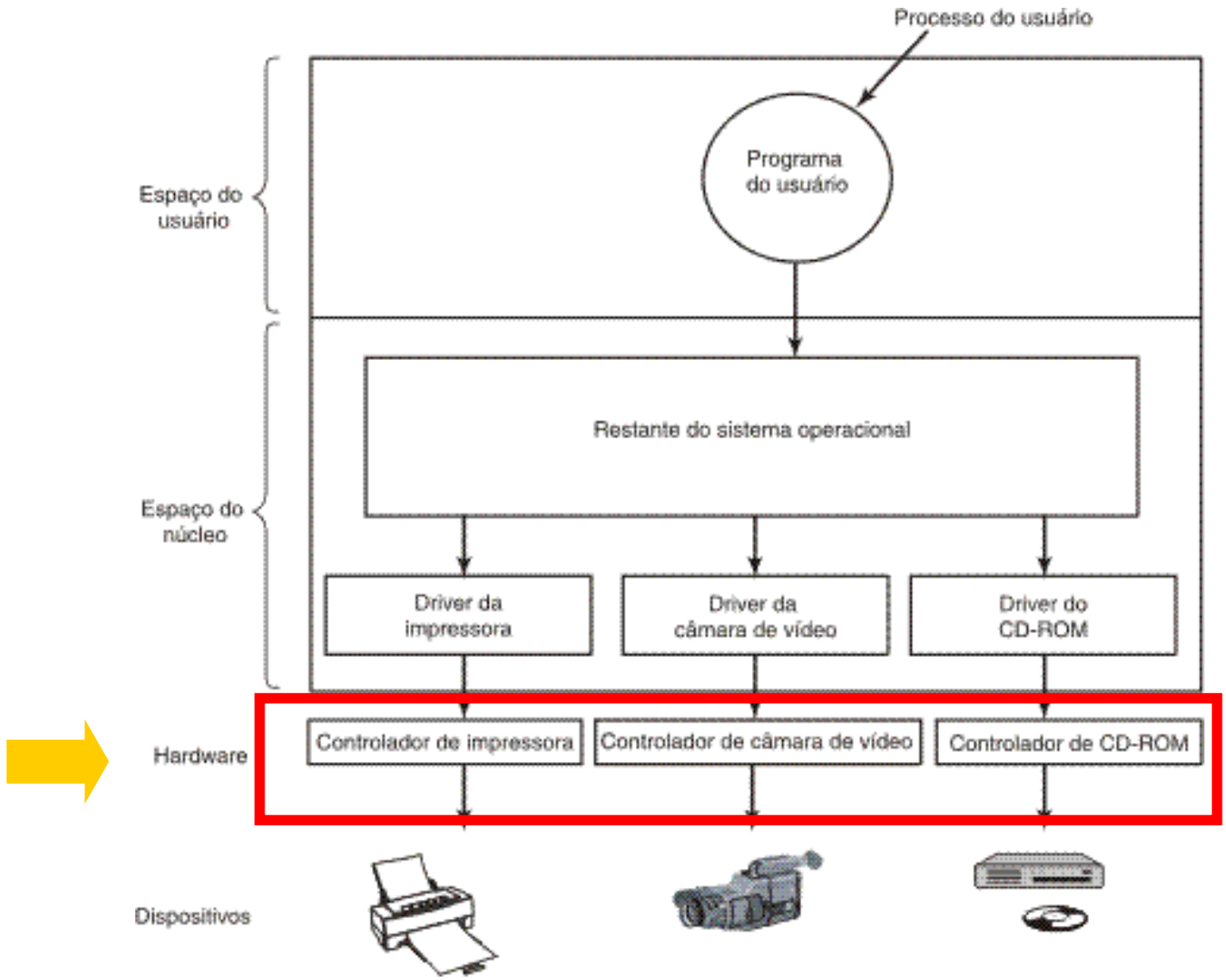
Roteiro

- Gerenciamento de Entrada e Saída
- Princípios básicos de hardware de E/S;
- Operações dos módulos de E/S;
- Princípios básicos de software de E/S;
- Organização dos discos;
- Desempenho;
- Escalonamento de dados;
- RAID;
- Leitura Sugerida

Gerenciamento de Entrada e Saída

- Parte do S.O. responsável por controlar a E/S de dados do sistema computacional;
- Operações realizadas:
  - Tratamento de interrupções;
  - Tratamento erros;
  - “Interface” entre os dispositivos e o resto do sistema;
  - Envio de comandos para os dispositivos;
  - Leitura/escrita entre dispositivo e sistema.

Princípios básicos de hardware de E/S



Princípios Básicos de Hardware

- As formas de E/S de dados em um sistema computacional são multivariadas;
- Para fins de organização e regularidade, dividem-se os dispositivos em duas classes:
  - Dispositivos de blocos;
  - Dispositivos de caracteres;
- Essa organização ocorre devido a forma como a informação é repassada entre o dispositivo de E/S e o restante do sistema;
- Alguns dispositivos não se enquadram em nenhuma das duas categorias:
    - Relógio: Causam interrupções em intervalos definidos;

Princípios Básicos de Hardware

Device	Data rate
Keyboard	10 bytes/sec
Mouse	100 bytes/sec
56K modem	7 KB/sec
Scanner	400 KB/sec
Digital camcorder	3.5 MB/sec
802.11g Wireless	6.75 MB/sec
52x CD-ROM	7.8 MB/sec
Fast Ethernet	12.5 MB/sec
Compact flash card	40 MB/sec
FireWire (IEEE 1394)	50 MB/sec
USB 2.0	60 MB/sec
SONET OC-12 network	78 MB/sec
SCSI Ultra 2 disk	80 MB/sec
Gigabit Ethernet	125 MB/sec
SATA disk drive	300 MB/sec
Ultrium tape	320 MB/sec
PCI bus	528 MB/sec

Tabela 1 - Taxa de dados dos dispositivos

Dispositivos de Caracteres

- É o tipo mais simples de dispositivo;
- Exemplos:
  - Mouse, techado, impressora;
- Stream contínua e assíncrona de dados;
- Não há estrutura nos dados;

Dispositivos de Blocos

- Informação é armazenada em blocos de bytes;
- Tamanho dos blocos varia de 512 a 32.768 bytes;
- Geralmente blocos possuem potências de 2 bytes;
- Bytes nos blocos podem ser acessados aleatoriamente;
- Exemplos:
  - HDD, SSD, CD, pendrive.

Princípios Básicos de Hardware

- Periférico é um dispositivo conectado ao computador de forma a possibilitar sua interação com o mundo externo;

- Não é conectado diretamente aos barramentos (dados, endereços) do computador;

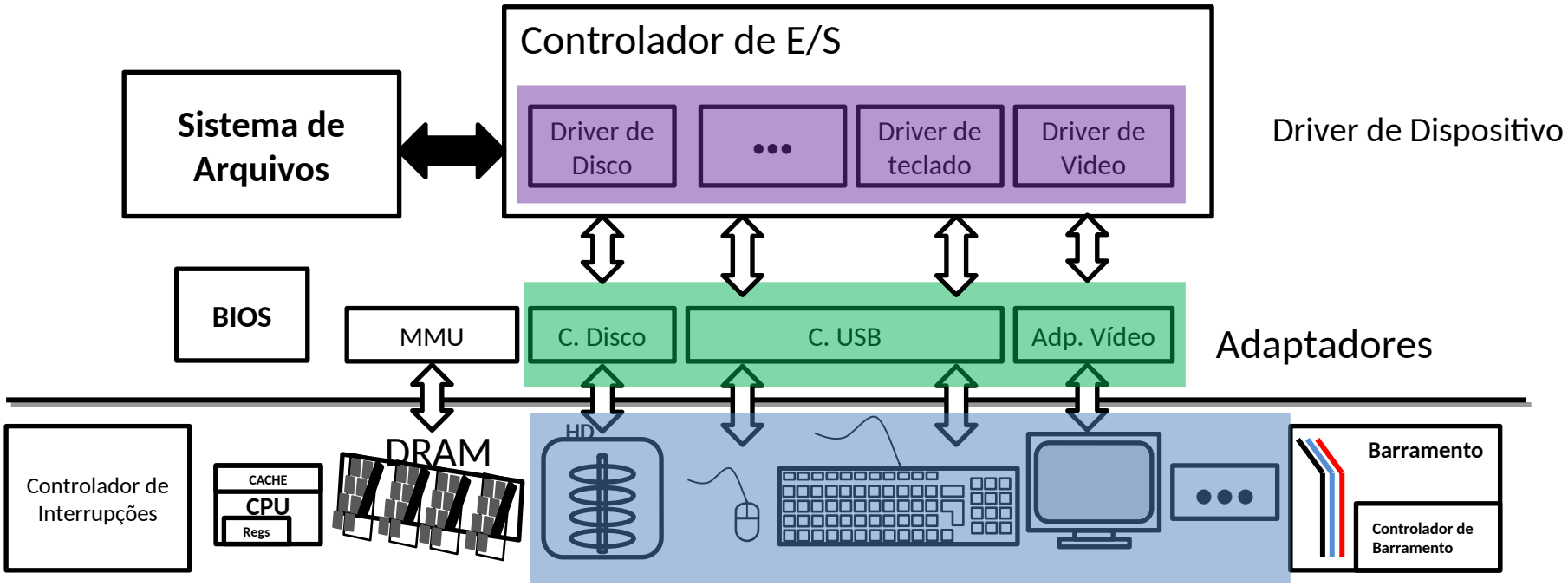
- Usam os elementos básicos de E/S como as controladores:

Elemento chave para transfêrencia de dados entre os dispositivos e as aplicações de um sistema computacional.

Operações como: “ler dados”, “escrever dados”, etc.

Princípios Básicos de Hardware

- Controlador para a transferência de E/S:



Representação Básica das Estruturas de Um Sistema Computacional

Princípios Básicos de Hardware

- Controladora é programado via registradores de dados de configuração (barrramento de dados e de endereços) e sinais:
  - Recebem ordens do processador,
  - Fornece o estado de uma operação,
  - Permite a leitura e escrita de dados do periférico.
- Registradores são “visto” como posições de memória:
  - E/S mapeamento em espaço de entrada e saída;
  - E/S mapeamento para a memória.

Mapeamento em espaço de E/S

- Espaço de endereçamento
- Conjunto de endereços de memória reservada para que o processador possa endereçar dados
- Pode haver um espaço de endereçamento distinto a entrada e saída:
- Cada registrador é associado a um número de porta de E/S
- Instruções específicas para acessar um ou outro espaço de endereçamento (registradores):
- mov, in, out

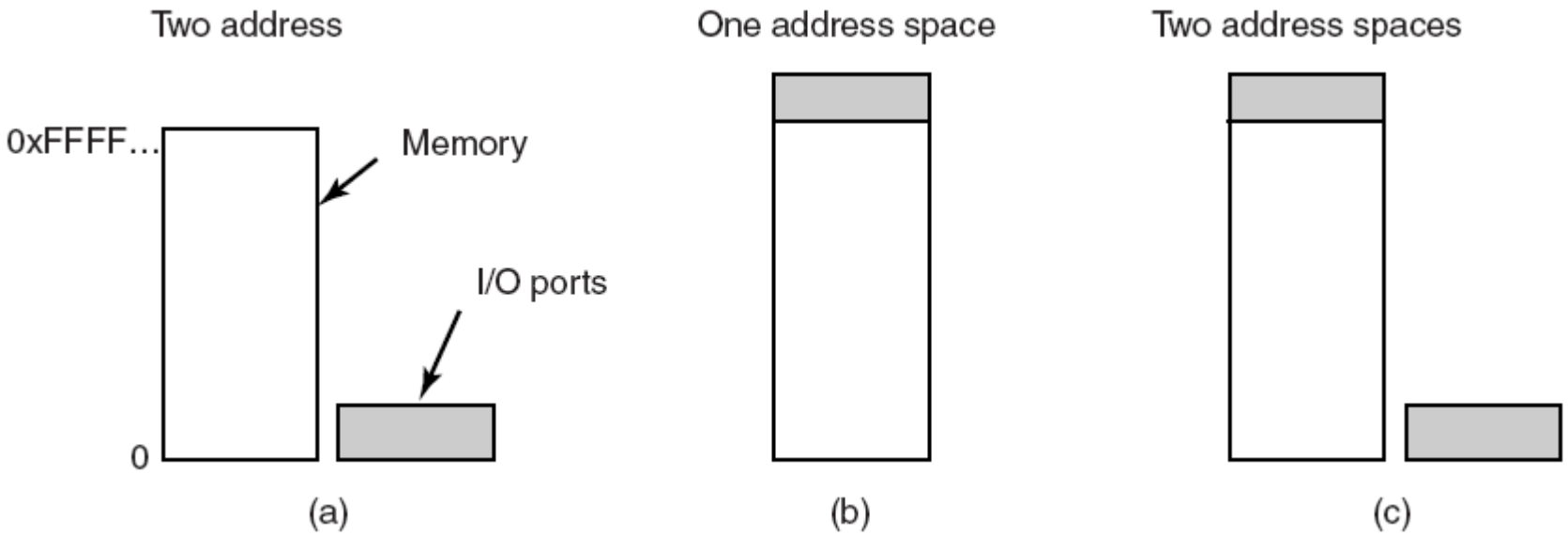
Mapeamento em espaço de memória

- Um único espaço de endereçamento (memória e E/S)
- Projeto do computador reserva uma parte de área de memória de endereçamento para E/S
- Intruções de acesso do tipo *mov end, dados* pode referenciar uma posição real da memória como um registrador associado a um periférico.
- Ex. Processadores da família motorola

Mapeamento em Esquema Híbrido

- Processador possui duas áreas distintas de endereçamento:
  - Acesso a memória: instruções (*mov)*
  - Acesso a E/S: instruções (*in, out*)
- Numericamente o valor do endereço pode ser o mesmo nos dois casos:
  - Diferença depende da instrução empregada (mov ou in).
  - Pode ter uma parte alocada da memória para mapeamento de E/S
  - Ex. Processadores Intel

Mapeamento



(a) Portas de E/S separada e espaço de memória.

(b) Mapeamento na Memória E/S.

(c) Mapeamento híbrido (Pentium).

Figura 2 – Representação de Mapeamento para E/S

Mecanismo de Interrupções

- Quando um dispositivo de E/S termina o trabalho deve gerar uma interrupção;
- Envia um sinal pela linha de barramento à qual está associada;
- O sinal é detectada pelo controlador de interrupção localizado na placa mãe.

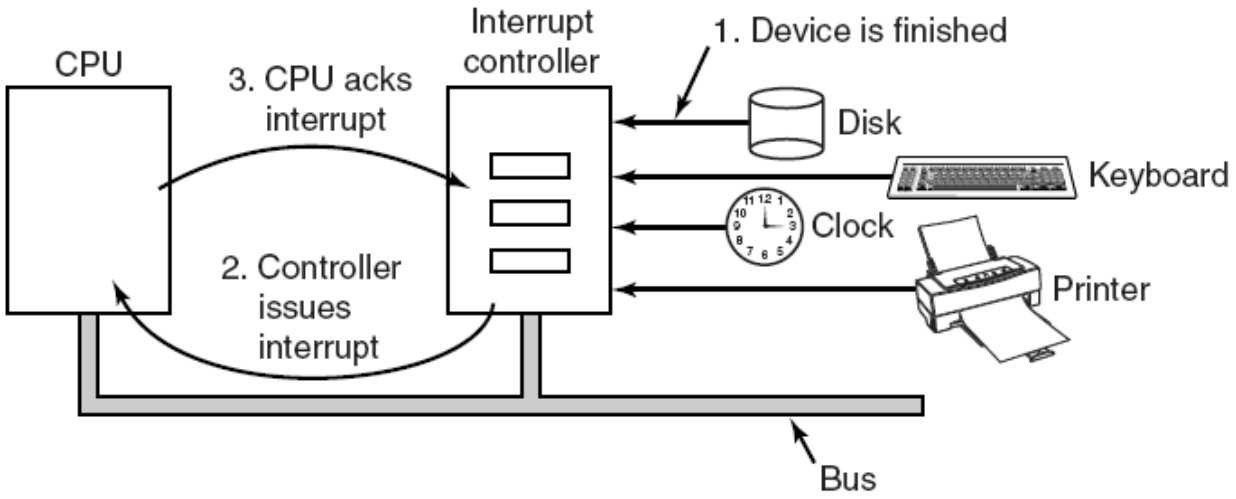


Figura 4 – Ocorrência de Interrupção

Interrupções

- Passos para o tratamento de uma interrupção:

1. Salva **registradores** relevantes na pilha;

2. Estabelece um contexto para a **rotina de tratamento** de interrupção. Isso **pode envolver a configuração** da TLB, MMU e uma tabela de páginas;

3. Estabelece a pilha para a **rotina de tratamento** da interrupção;

4. **Sinaliza o controlador de interrupção**. Senão existe um controlador de interrupção reabilita as interrupções;

5. **Copia os registradores** de onde eles foram salvos (possivelmente de alguma pilha) para a tabela de processos;

Interrupções

6 **Executa a rotina** de tratamento de interrupção. Ela extrairá informações dos registradores do controlador do dispositivo que está interrompendo;

7 **Escolhe o próximo processo** a executar. Se a interrupção deixou pronto algum processo de alta prioridade anteriormente bloqueado, este pode ser escolhido para executar agora;

8 **Estabelece o contexto da MMU para o próximo processo** a executar. Algum ajuste na TLB também pode ser necessário;

9 **Carrega os registradores** do novo processo;

10 **Inicializa a execução** do novo processo.

Interrupções

vector number	description
0	divide error
1	debug exception
2	null interrupt
3	breakpoint
4	INTO-detected overflow
5	bound range exception
6	invalid opcode
7	device not available
8	double fault
9	coprocessor segment overrun (reserved)
10	invalid task state segment
11	segment not present
12	stack fault
13	general protection
14	page fault
15	(Intel reserved, do not use)
16	floating-point error
17	alignment check
18	machine check
19–31	(Intel reserved, do not use)
32–255	maskable interrupts

As interrupções que ocorrem em Linux podem ser vistas:
$ cat /proc/interrupts

Acesso Direto à Memória (DMA)

- Não importa se a CPU tem ou não E/S mapeada na memória: **deve endereçar os controladores** dos dispositivos para trocar dados etre eles;
- A CPU pode **requer dados**, um byte de cada vez, mas essa tarefa **desperdiça tempo** de processamento;
- Com isso, há um hardware especial, **controlador DMA**, para tranferência de dados;
- A técnica que propõe utilizar uma única **interrupção para efetuar a transferência** de um bloco de dados **entre o dispositivo e a memória**, sem o envolvimento da CPU: **reduzindo o número de operações**;

Acesso Direto à Memória (DMA)

- **Eficiente** quando envolve **muitos dados** (leitura de disco);
- **Funcionamento:**
  - CPU inicializa o controlador DMA fornecendo informações sobre quantidade de dados a transferir, origem e destino;
  - CPU dispara a execução do DMA, iniciando a transferência;
  - CPU se dedica a outra tarefa;
  - Ao final da operação o DMA, aciona uma interrupção para sinalizar o término da operação.

Acesso Direto à Memória (DMA)

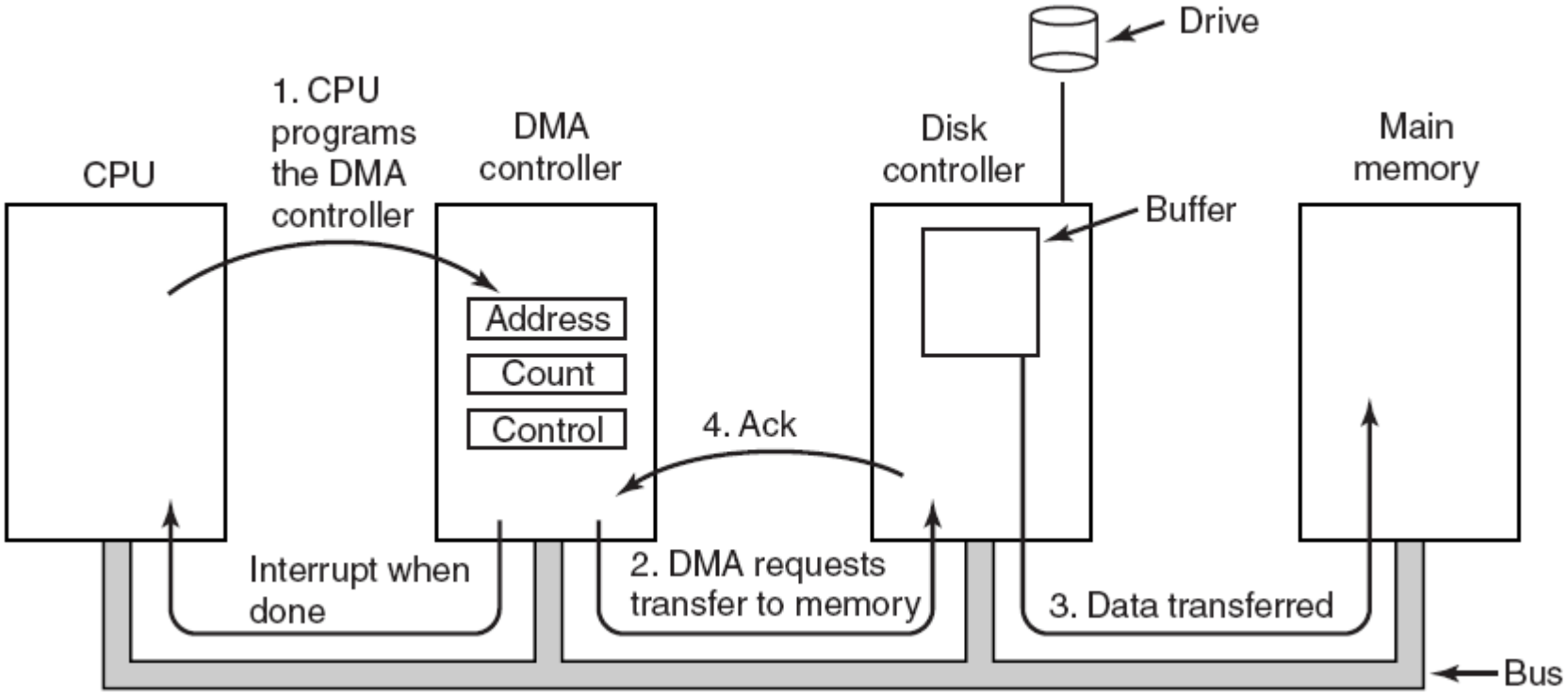


Figura 3 – Operação de Transferência Utilizando DMA

Princípios básicos de software de E/S

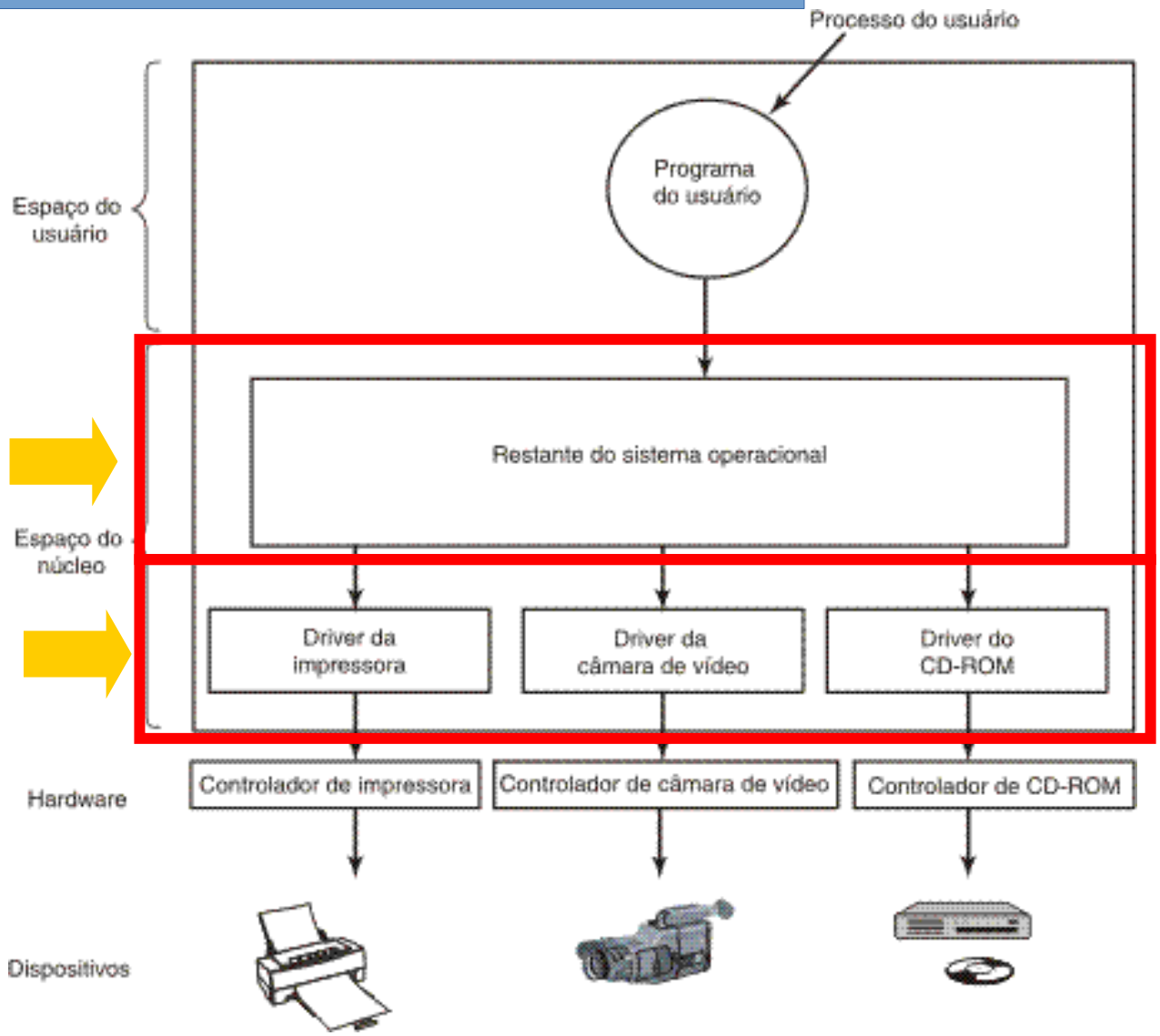
O objetivo é padronizar **as rotinas de acesso aos dispositivos** de forma a reduzir o número de rotinas;

O sistema é organizado em camadas:

- Camadas mais altas apresentam interface para o usuário:
  - Aplicações de Usuário;
  - Chamadas de Sistemas;
- Camadas mais baixas apresentam detalhes de hardware:
  - Software independente de E/S;
  - Drivers.

Princípios básicos de software de E/S

Interface padrão para drivers de dispositivos (API)



Princípios básicos de software de E/S

- **Software de E/S no nível Usuário**:
  - Bibliotecas de E/S são utilizadas pelos programas dos usuários;
  - Chamadas ao sistema (*system calls*);

Princípios básicos de software de E/S

- **Software Independente de E/S**:
  - Realizar as funções comuns a qualquer dispositivo;
  - Fazer o escalonamento de E/S no dispositivo;
  - Atribuir um nome lógico a partir do qual o dispositivo é identificado;
    - Exemplo: UNIX (/dev)
  - Prover *buffering*: ajuste entre a velocidade e a quantidade de dados transferidos;
  - *Cache* de dados: armazenar na memória um conjunto de dados frequentemente acessados.

Princípios básicos de software de E/S

- Reportar erros e proteger os dispositivos contra acessos indevidos:
- Gerenciar alocação, uso e liberação dos dispositivos: acessos concorrentes;
- Transferência de dados:
  - Síncrona (bloqueante): requer bloqueio até que os dados estejam prontos para transferência;
  - Assíncrona (não-bloqueante): transferências acionadas por interrupções; mais comuns.

Princípios básicos de software de E/S

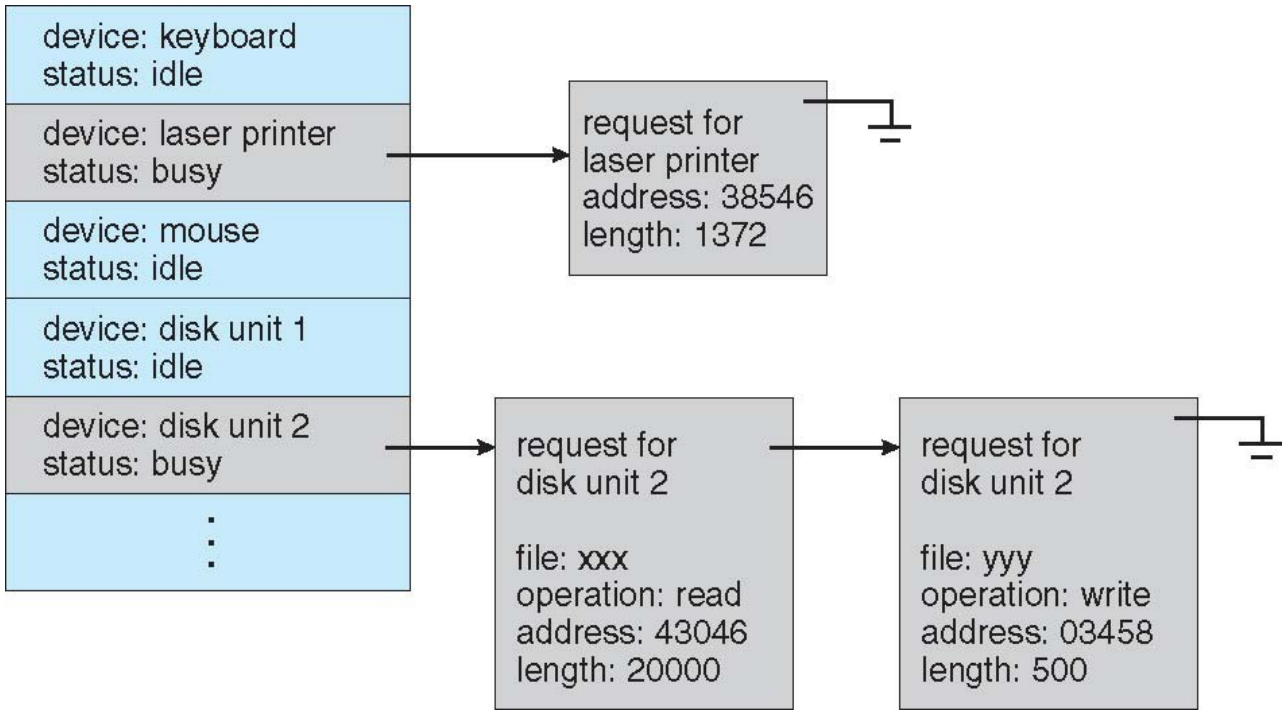
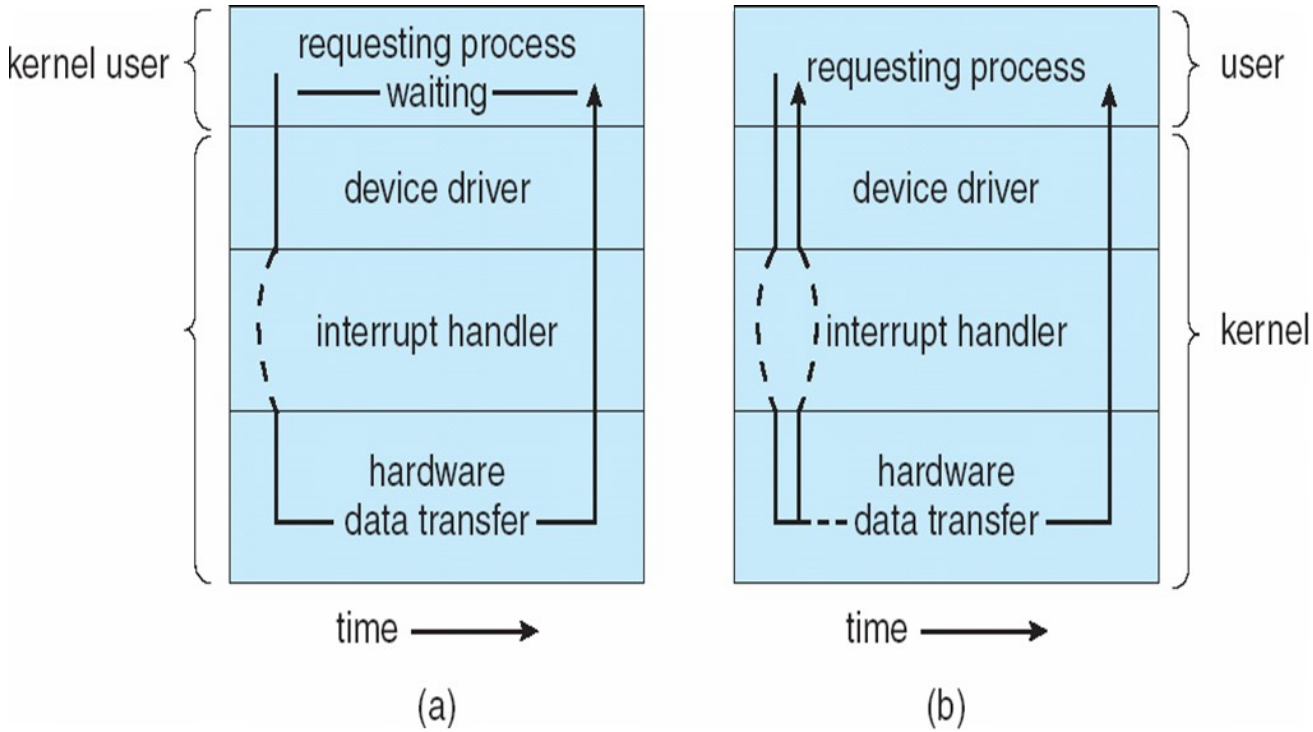


Tabela de status dos dispositivos

Princípios básicos de software de E/S



Síncrono

Assincrono

Camadas do Sistema de E/S: (a) síncrono e (b) assíncrono

Princípios básicos de software de E/S

- Tipos de dispositivos:
  - Compartilháveis: podem ser utilizados por vários usuários ao mesmo tempo;
  - Dedicados: podem ser utilizados por apenas um usuário de cada vez;

Princípios básicos de software de E/S

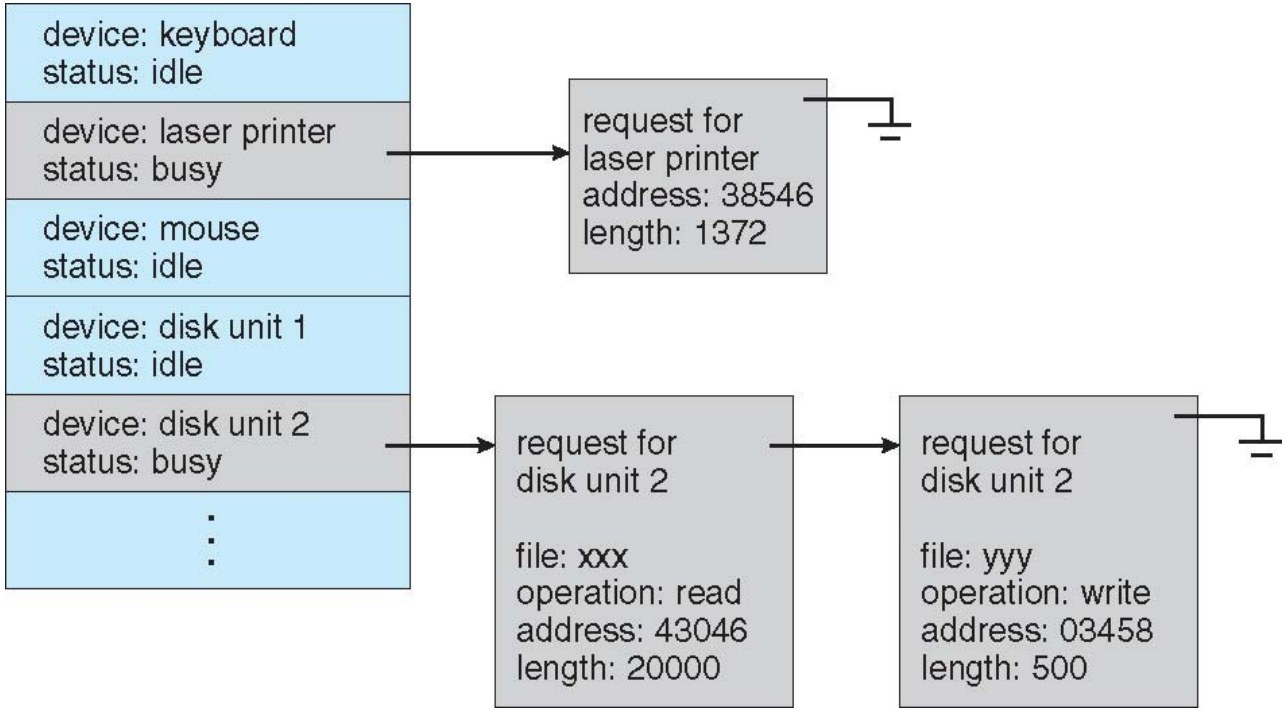


Figura 6 - Tabela de status dos dispositivos

Princípios básicos de software de E/S

Drivers de dispositivos

- Responsável pelo gerenciamento do dispositivo de E/S;
- Código específico a um dispositivo;
- Recebe solicitações da camada de gerenciamento do dispositivo (lógico):
  - Subdivide os periféricos em função da unidade de transferência de dados;
    - Orientado a caracteres;
    - Orientado a blocos;
- Responsável por tratamento de erros;

Drivers de Dispositivo

- No Windows, dispositivos não são tratados como arquivos especiais;
- Há uma chamada do sistema para cada dispositivo distinto;
- Nos idos tempos do DOS, valores eram colocados em registradores específicos e então uma instrução em assembly especial era invocada para que a comunicação ocorrece;
- No Windows o mecanismo ainda é o mesmo, no entanto, o mecanismo fica escondido dentro de chamadas do sistema;

Drivers de Dispositivo

- No UNIX e sistemas baseados (Linux, freeBSD, etc) dispositivos são logicamente mapeados para arquivos (especiais);
- Do ponto de vista do programador, acessar um dispositivo não é mais complicado do que ler ou escrever em um arquivo;
- Toda a complexidade do dispositivo é escondida do usuário (programador) por meio de uma interface de chamadas do sistema comum a todos os dispositivos;
- No caso as mesmas chamadas do sistema usadas para manipular arquivos de dados;

Drivers de Dispositivo

- Driver em modo Kernel são implementados como **MODULOS**;
- No LINUX pelo menos duas funções são necessárias:
  - init_module;
  - cleanup_module;
- Não há função principal (main);

Ex:

```
#define MODULE
#include <linux/module.h>

int init_module(void) { printk("<1>Hello, world\n"); return 0; }
void cleanup_module(void) { printk("<1>Goodbye cruel world\n"); }
```

Dispositivos periféricos tipicos

Existem uma grande variedade de dispositivos:

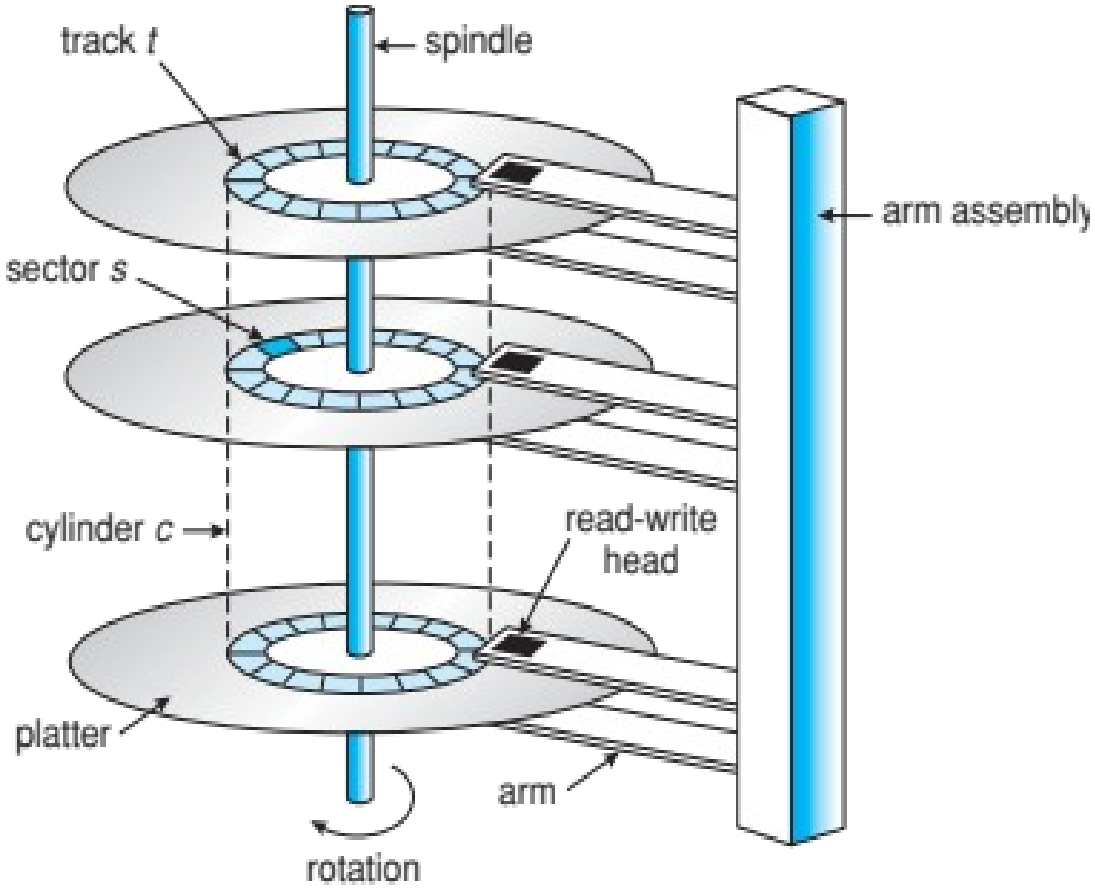
- Dispositivos apropriados para sistemas computacionais de grande porte;
  - Exemplo: terminais de vídeo.
- Dispositivos de armazenamento secundário;
- Dispositivos de comunicação, etc;

Para o sistema operacional: o periférico muito importante é o disco magnético, principalmente **o disco rígido** (Oliveira et al., 2001).

Organização dos Discos - HDD

- Possui um ou vários discos: às vezes chamados de pratos (platters), conectados a uma haste que gira em alta velocidade;
- O conjunto de discos é dividido em circunferências concêntricas denominadas cilindros (cylinders);
- A cabeça de leitura (head) define as trilhas (track) => que é dividida radialmente em setores (sectors);
- Exemplo:
  - Taxa de transferência (Teoria) – 6 Gb/sec
  - Taxa de transferência efetiva (Real) – 1 Gb/sec
  - Tempo de busca de 3 ms a 12 ms

Organização dos Discos



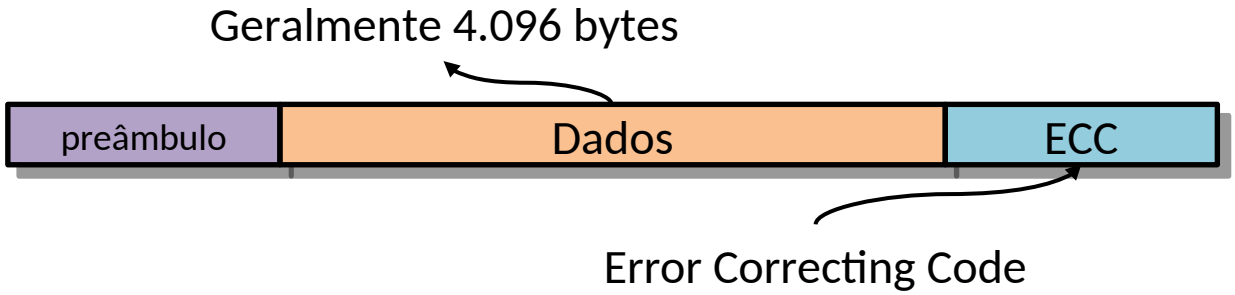
Mecanismo de movimentação do cabecote do disco

Organização dos Discos

- **Múltiplos pratos (disk pack)**
  - Vários pratos empilhados e concentrados;
  - Cada prato tem um cabeçote de leitura/escrita (braço móvel);
- **Mecanismo do cabeçote de leitura e escrita**
  - Contato físico entre a superfície magnética e o cabeçote;
  - Distância física (air gap) da superfície magnética (menor do que uma partícula de fumaça);

Organização dos Discos

- **Setores *(sectors)***
  - Armazenamento de informações;
  - Informações de controle:
  - Exemplo: início, final do setor e ECC (Error Correcting Code);
  - Entre as trilhas existe um espaço livre *(inter-track gap)* da mesma forma entre os setores *(inter-track gap)*;
  - Todo o espaço livre existente entre trilhas e setores não é utilizado por este dispositivo.

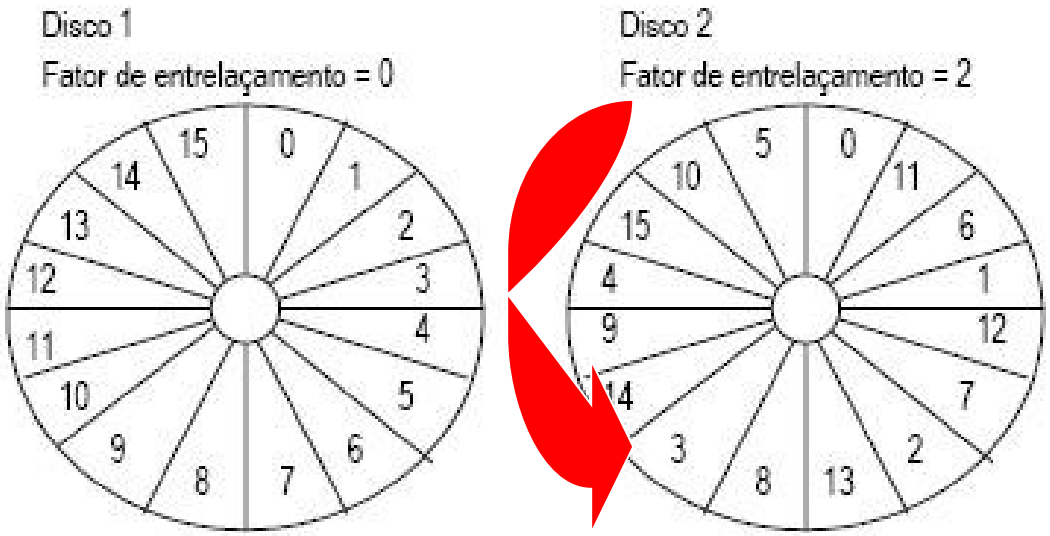


Desempenho em Disco

- Tempo de acesso aos dados:
  - O tempo necessário para mover até o cilindro correto: o tempo de acesso à trilha ou tempo de pesquisa da trilha (***seek time***);
  - O tempo necessário para a cabeça ser posicionada no início do setor desejado, chamado de atraso rotacional (***rotational delay***);
  - O tempo de transferência dos dados: a leitura ou escrita dos dados (***transfer time* ou *transmission time***).

Entrelaçamento

- Técnica:
  - Forma de aumentar o desempenho no acesso ao disco;
  - Evitar latência rotacional em setores adjacentes;
  - Consiste em numerar os setores não mais de forma contígua, mas com espaço entre eles.



Exemplo:

- leitura dos setores 4 e 5.
- existem 2 setores.

Escalonamento dos Dados

- Um processo realiza uma operação de E/S em disco através de uma chamada de sistema (*System Call*);
- Reflete em posições diferentes de trilhas e setores do disco;
- Menor tempo de posicionamento pode melhorar substancialmente o desempenho do sistema;
- Objetivo: Minimizar os movimentos da cabeça de leitura e maximizar o número de bytes transferidos para atender a um maior número de requisições;
- Algoritmos de escalonamento para realizar a movimentação da cabeça de leitura do disco.

Escalonamento de Dados

- Exemplo:
  - Disco organizado em 200 trilhas (**0-199**);
  - Posição inicial do cabeçote: trilha 53;
  - Atender a seguinte lista de requisições:
    - **96, 183, 37, 122, 14, 124, 65 e 67.**

Escalonamento de Dados

FCFS (First Come First Served)

- Acesso na ordem em que as requisições são solicitadas;
- Obtém-se um deslocamento equivalente a 640 trilhas;

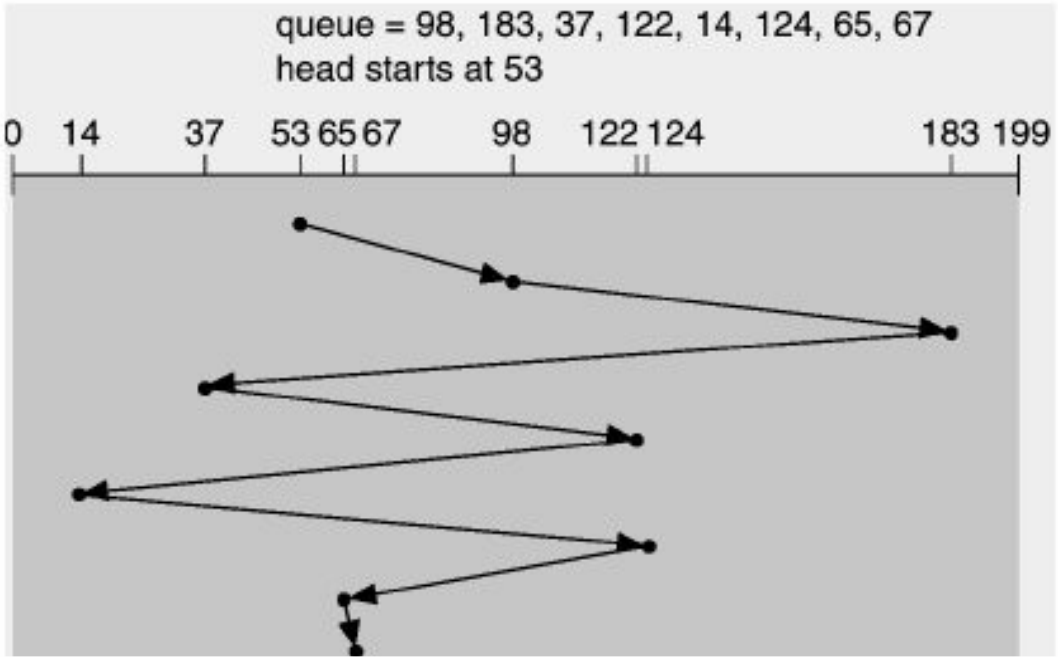


Figura 9 – Escalonamento usando o algoritmo FCFS

Escalonamento de Dados

SSTF (Shortest Seek Time First)

- Seleciona a requisição com menor tempo de Seek em relação a posição atual do cabeçote de leitura e escrita;
- Análogo ao algoritmo SJF (Shortest Job First);
  - Pode provocar um atraso em uma requisição

Escalonamento de Dados

SSTF (Shortest Seek Time First)

- Deslocamento equivalente a 236 trilhas:

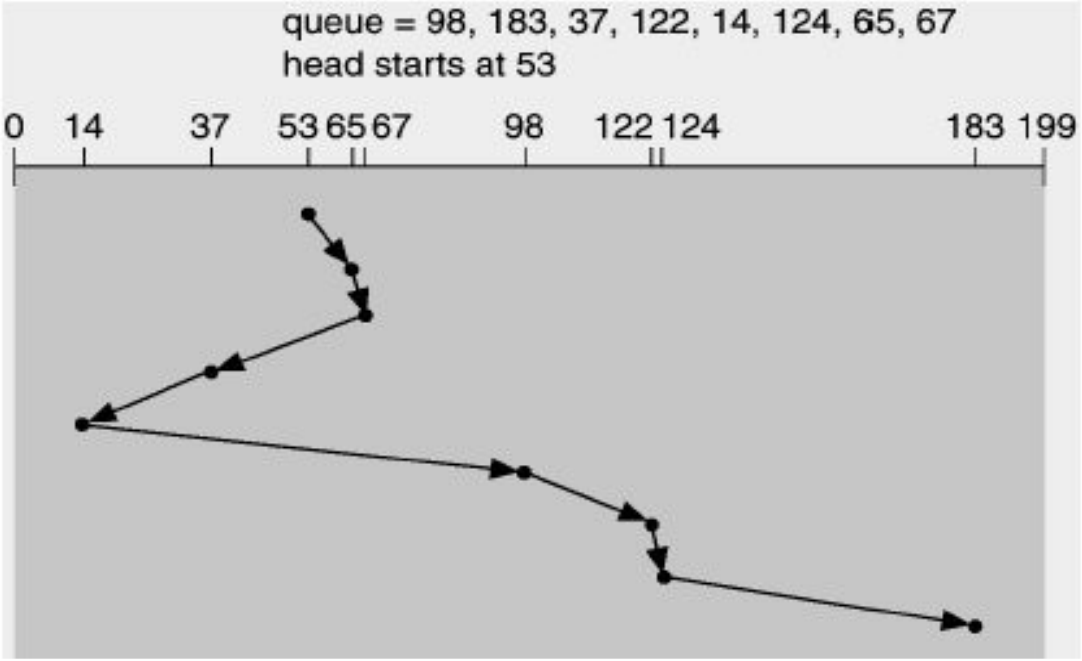


Figura 10 – Escalonamento usando o algoritmo SSTF

Escalonamento de Dados

SCAN

- O movimento do cabeçote inicia em uma extremidade do disco e se movimenta em direção a outra extremidade;
  - Executa as requisições na ordem desta varredura;
  - Ao chegar ao outro extremo, inverte o sentido e repete o procedimento;
- Conhecido como algoritmo do elevador;

Escalonamento de Dados

SCAN

- Deslocamento equivalente a 208 trilhas (não conta a extremidade).

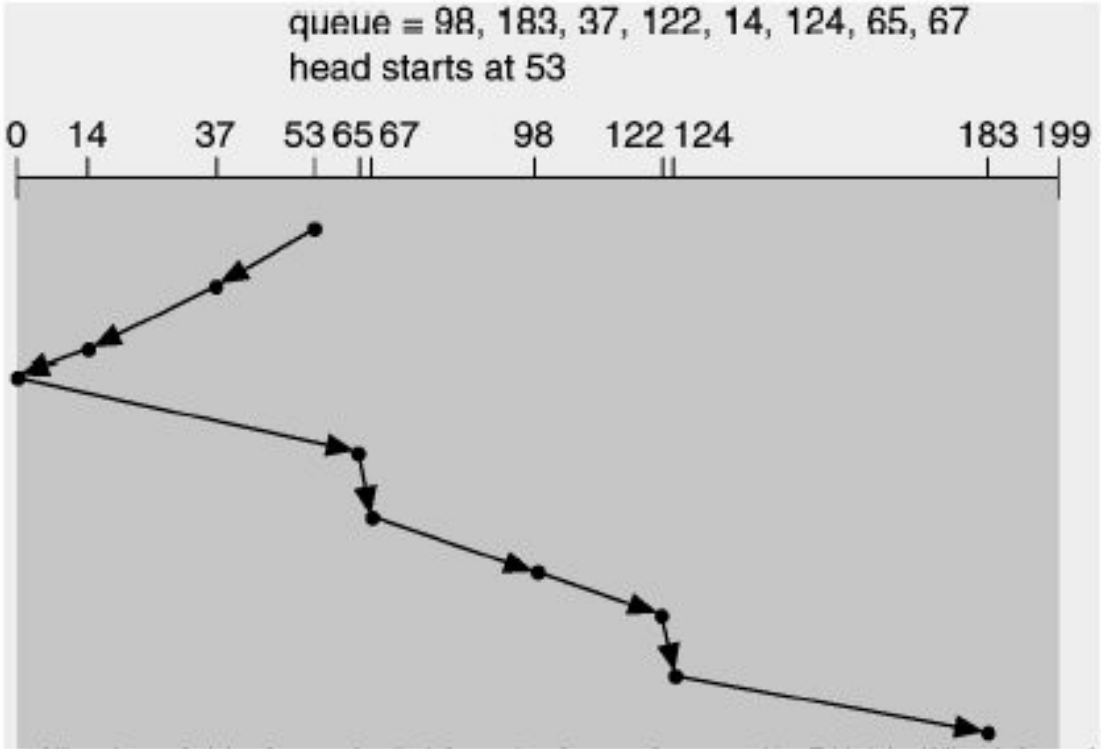


Figura 11 – Escalonamento usando o algoritmo SCAN

Escalonamento de Dados

C-SCAN

- Variação do algoritmo de SCAN;
- Procedimento é idêntico ao do algoritmo SCAN, porém, as requisições são atendidas apenas em um sentido da varredura;
  - Ao final da varredura o cabeçote é posicionado no início do disco;
- Fornece uma visão lógica onde o disco é tratado como uma fila circular;
  - Oferece um tempo médio de acesso mais uniforme que o SCAN;

Escalonamento de Dados

C-SCAN

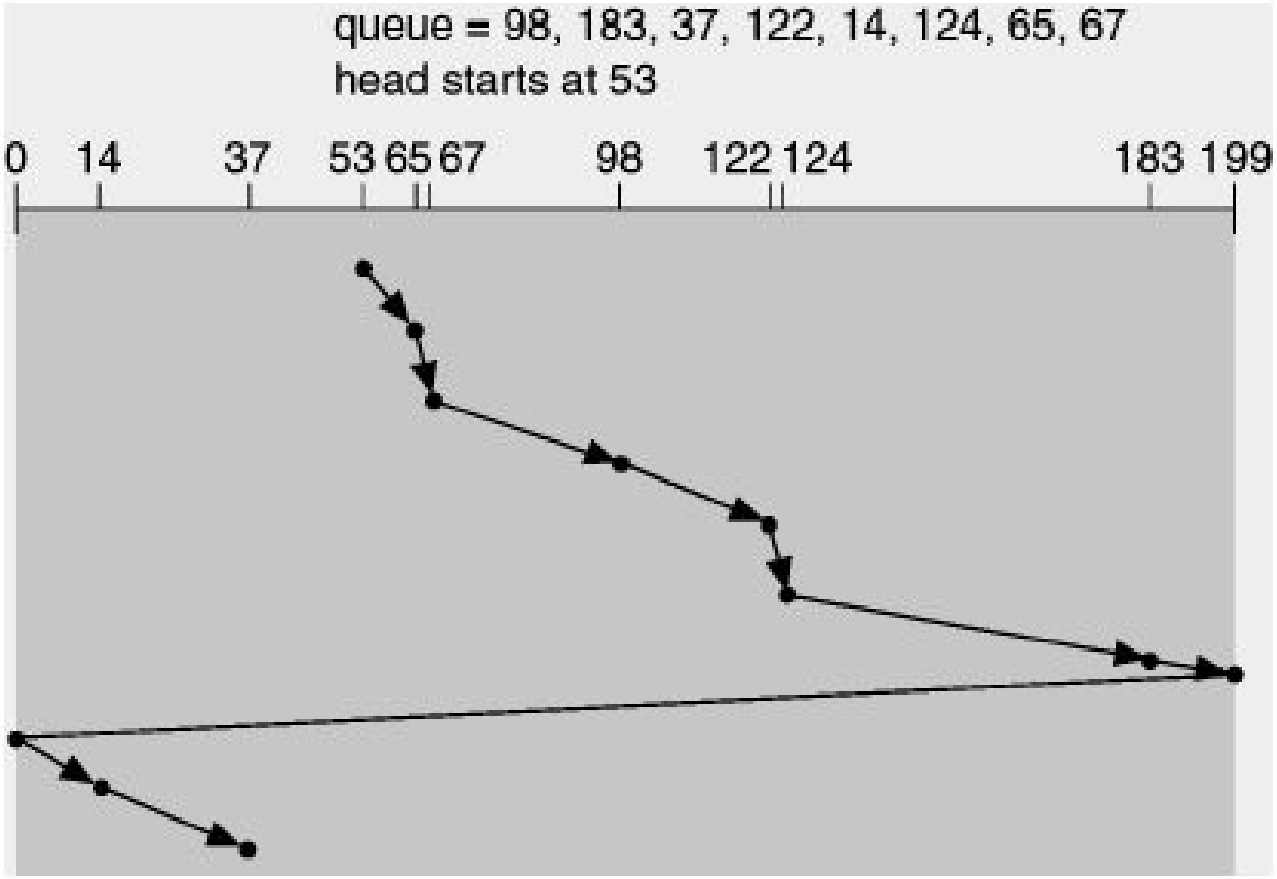


Figura 12 – Escalonamento usando o algoritmo C-SCAN

RAID

- O ritmo de melhora na tecnologia de armazenamento secundário tem sido menor quando comparado a memória principal e processadores;

- Se um componente só pode ser avançado até certo ponto, ganhos adicionais podem ser obtidos usando-se múltiplos componentes paralelos;

- RAID (*Redundant Array of Independent Disks)* foi um acordo para o desenvolvimento de bancos de dados de múltiplos discos, dividido em sete níveis (0 - 6).

RAID

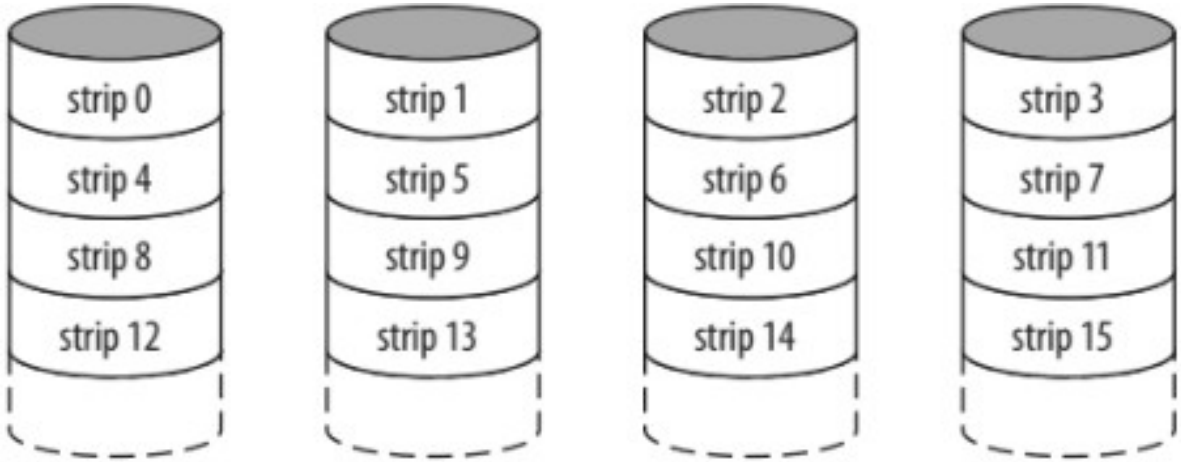
- Os níveis não possuem relacionamento hierárquico entre si, mas têm as seguintes características:
  - RAID é um conjunto de unidades de discos físicos, vistos pelo SO como uma única unidade lógica;
  - Os dados são distribuídos pelos discos em um esquema conhecido como intercalação de dados **(striping)**;
  - A capacidade de disco redundante é usada para armazenar informações de paridade, o que garante a facilidade de recuperação dos dados no caso de uma falha de disco.

RAID

- A estratégia RAID emprega várias unidades de disco e distribui os dados de modo que permita o acesso simultâneo aos dados a partir das várias unidades:
    - Melhora o desempenho de E/S e permitindo aumentos na capacidade de modo mais fácil;
- Embora permita que várias cabeças e atuadores operem simultaneamente e gere taxas de E/S e transferência mais altas, o uso de múltiplos dispositivos aumenta a probabilidade de falha;

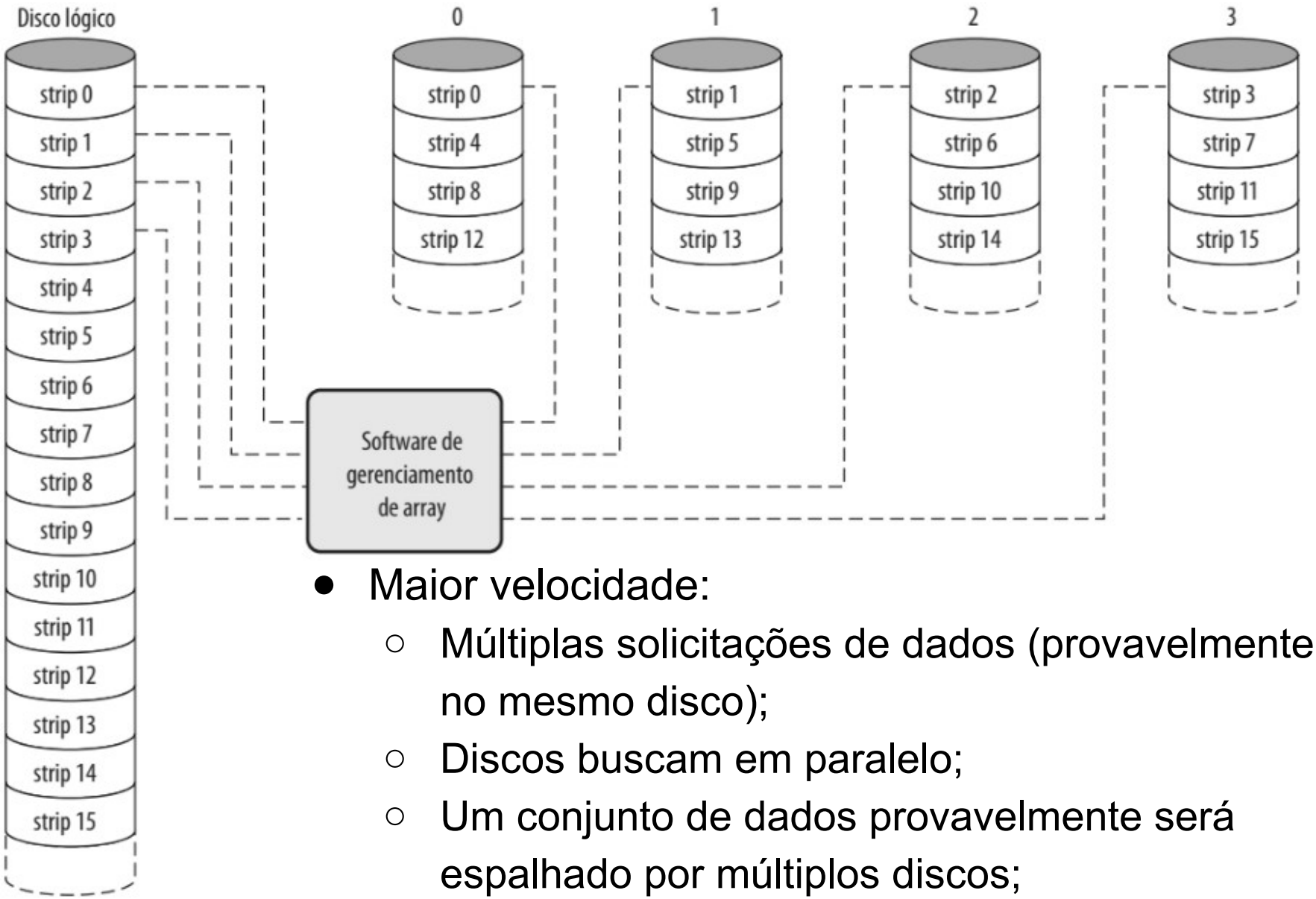
RAID 0

- Não redundante;
- Dados espalhados por todos os discos;
- Mapeamento *Round Robin*;



RAID 0 (não redundante)

RAID 0



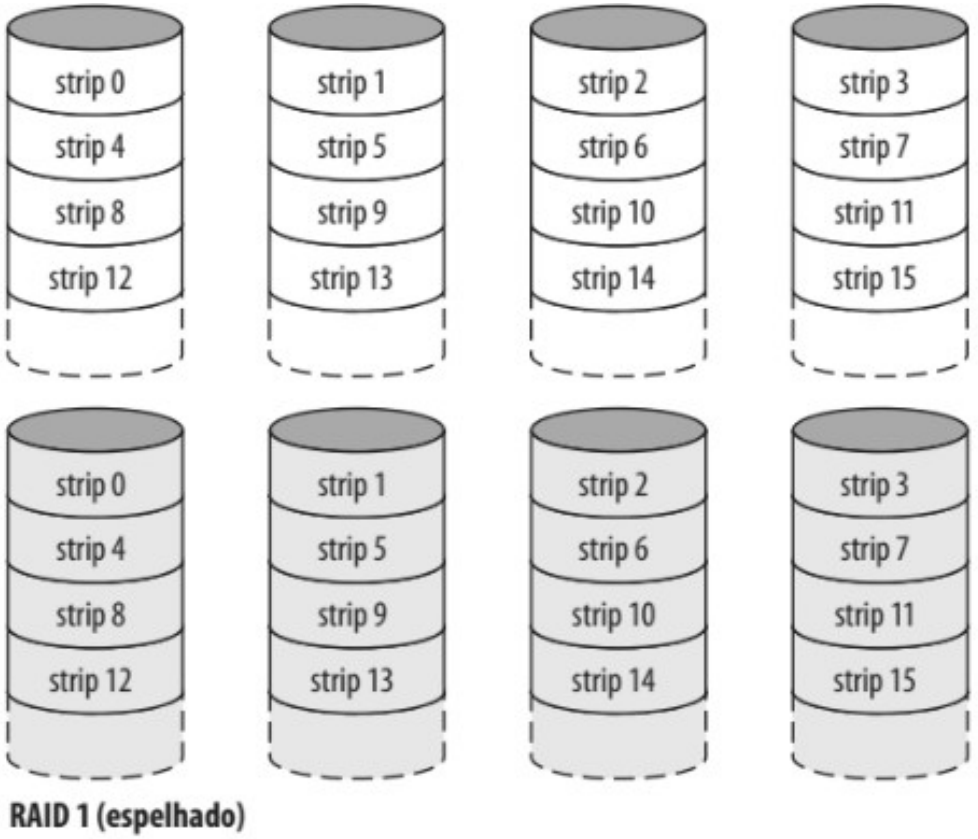
- Maior velocidade:
  - Múltiplas solicitações de dados (provavelmente não no mesmo disco);
  - Discos buscam em paralelo;
  - Um conjunto de dados provavelmente será espalhado por múltiplos discos;

RAID 0

- Não é um membro “verdadeiro” da família RAID pois não inclui redundância;
- Aplicações em que o desempenho e a capacidade são preocupações principais;
- Baixo custo é mais importante que confiabilidade;
- Supercomputadores.

RAID 1

- Discos espelhados;
- Dados espalhados pelos discos;
- 2 cópias de cada ***stripe*** em discos separados;
- Leitura e gravação em ambos;
- Mecanismo caro de ser implementado.



RAID 1 (espelhado)

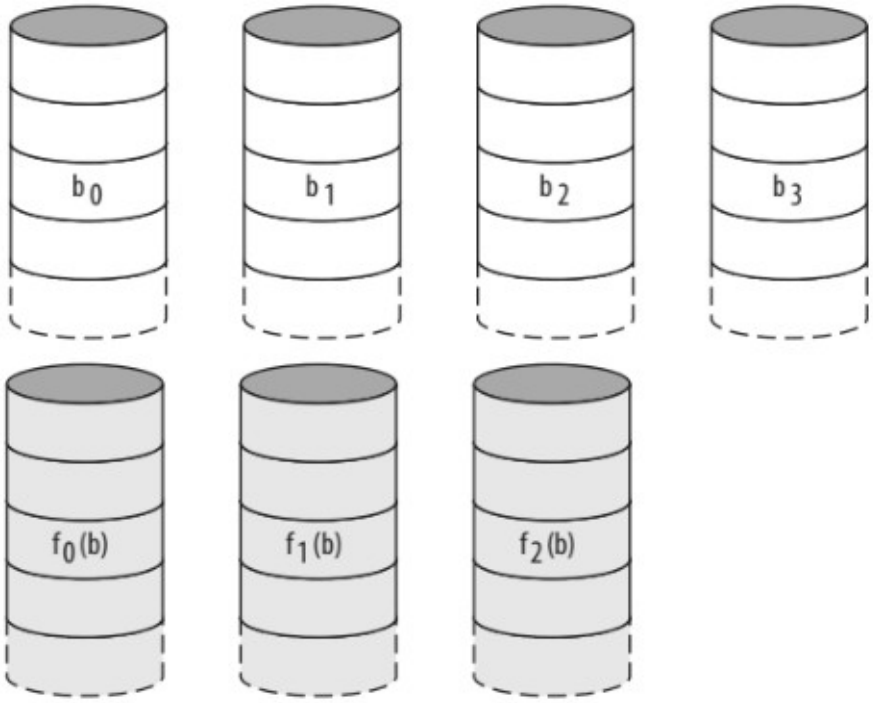
RAID 1

- Uma solicitação de leitura pode ser atendida por qualquer um dos dois discos (geralmente, o que envolver o mínimo de tempo de busca + atraso rotacional);

- Uma solicitação de gravação requer que os dois *strips* correspondentes sejam atualizados, mas isso pode ser feito em paralelo;

RAID 2

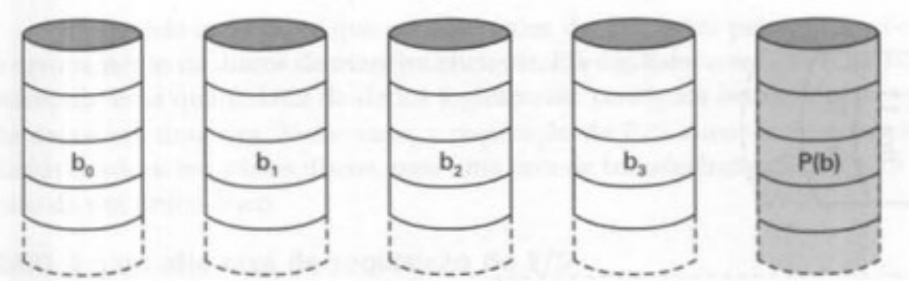
- Discos são sincronizados;
- Correção de erro calculada pelos *bits* correspondentes nos discos;
- Múltiplos discos de paridade armazenam correção de erro via **código de Hamming** em posições correspondentes;
- Muita redundância: caro e pouco utilizado.



RAID 2 (redundância por código de Hamming)

RAID 3

- REDUNDÂNCIA: calculada por meio da operação XOR:
  - Suponha um array de 4 discos X0 a X4, sendo que X4 é o disco de paridade;
  - X4 = X0 ⊕ X1 ⊕ X2 ⊕ X3;
  - Se uma unidade Xn falhar, seus dados podem ser recuperados da seguinte forma:
  - X3 = X0 ⊕ X1 ⊕ X2 ⊕ X4 (exemplo para n=3);
  - Isso é aplicado nos RAIDs de níveis 3 a 6;

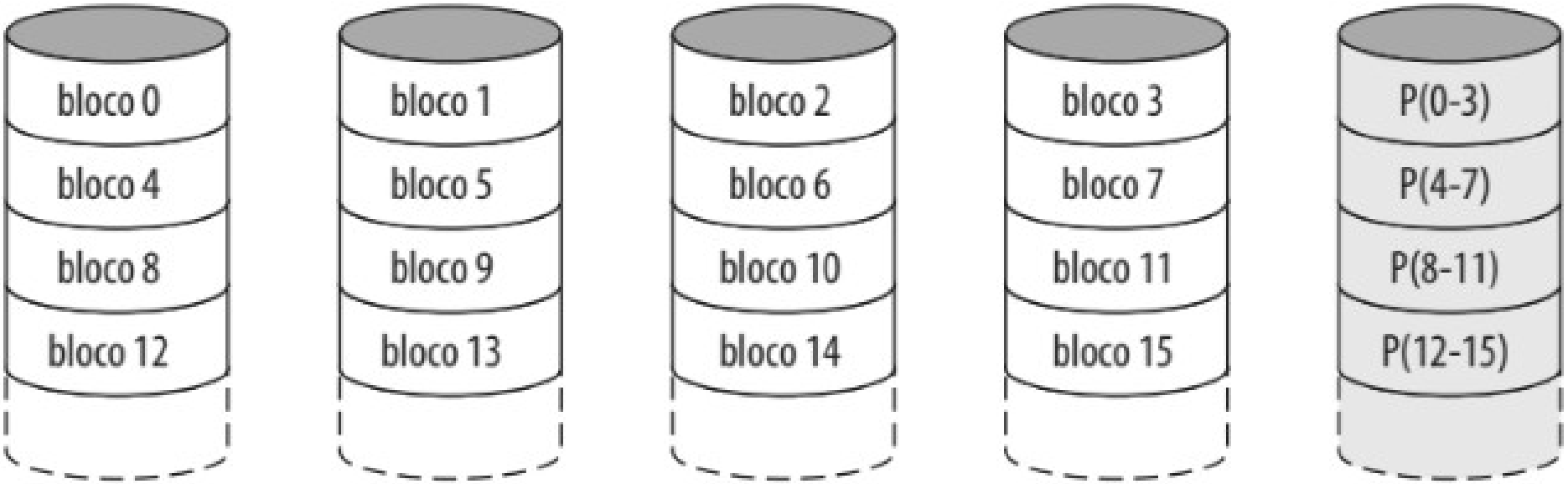


(d) RAID 3 (bit de paridade intercalado)

- DESEMPENHO:
  - Dados são distribuídos em *stripes* pequenos (1 *byte* ou palavra);
  - Taxas de transferência de dados altas;

RAID 4

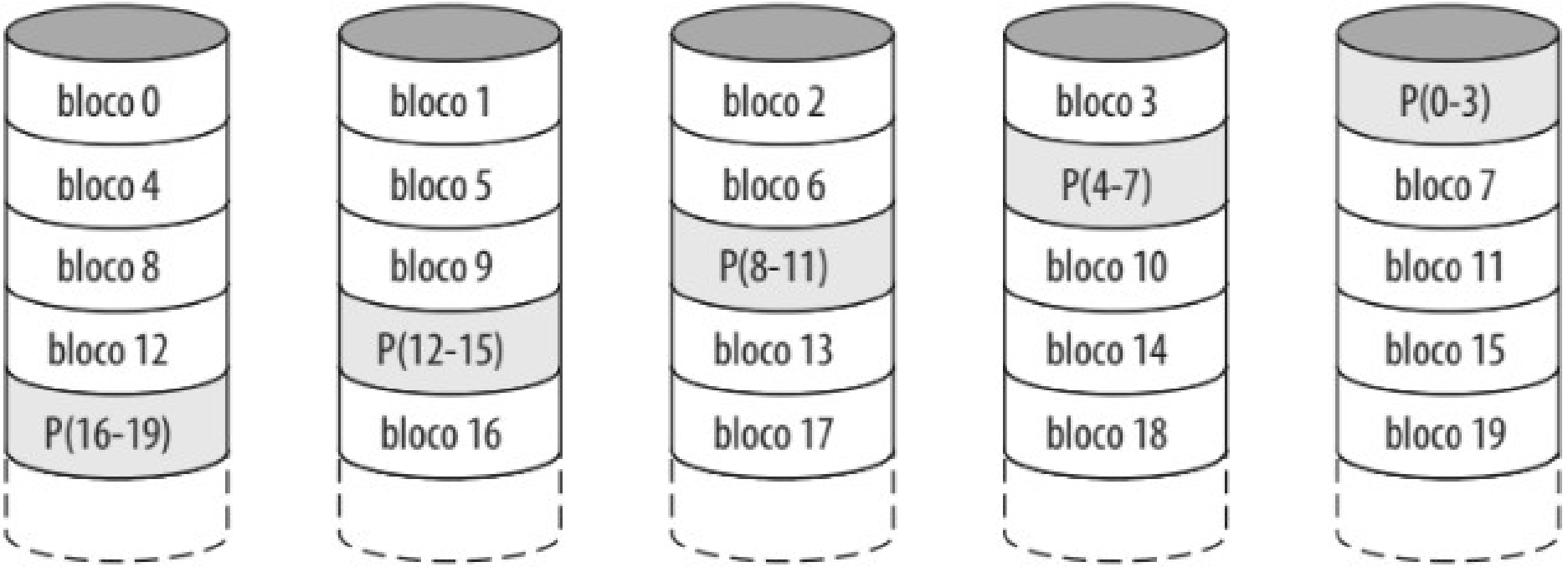
- Cada disco opera independentemente;
- Bom para taxa de solicitação de E/S alta;
- Paridade *bit* a *bit* calculada por *stripes* em cada disco;
- Paridade armazenada no disco de paridade



RAID 4 (paridade em nível de bloco)

RAID 5

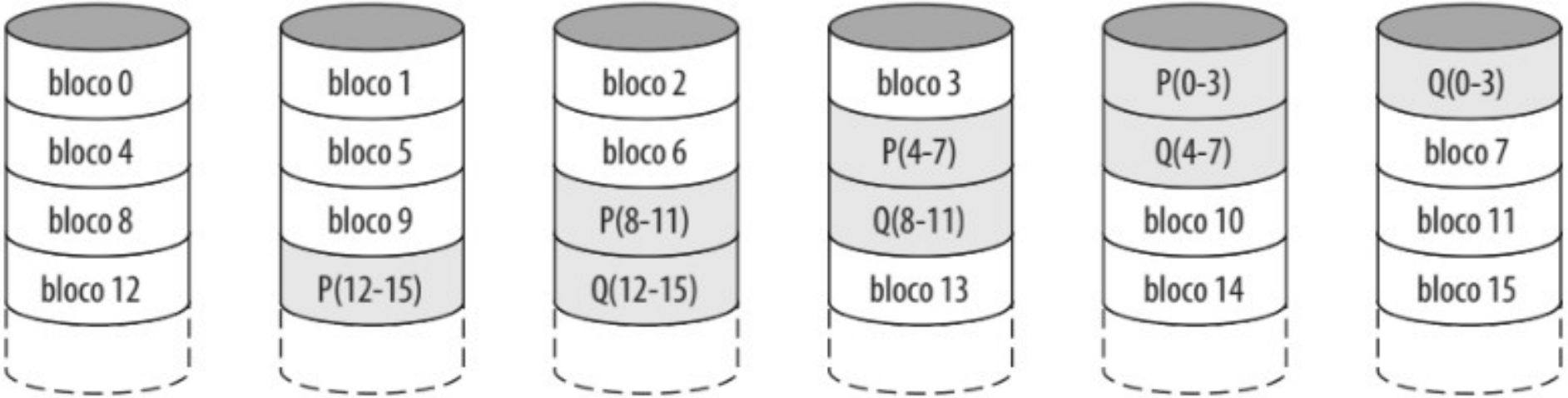
- Paridade espalhada por todos os discos;
- Alocação *round-robin* para *stripe* de paridade;
- Evita gargalo do RAID 4 no disco de paridade.



RAID 5 (paridade em nível de bloco distribuída)

RAID 6

- Dois cálculos de paridade;
- Armazenado em blocos separados em discos diferentes;
- Alta disponibilidade de dados:
  - Três discos precisam falhar para haver perda de dados;



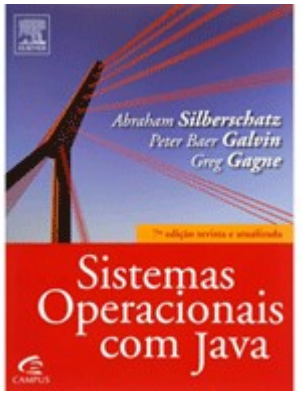
RAID 6 (redundância dual)

RAID

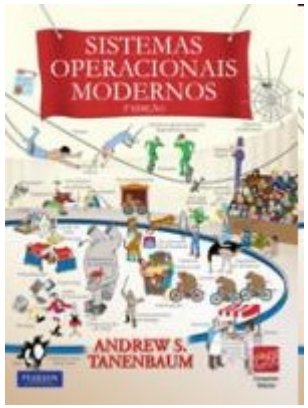
Categoria	Nível	Descrição	Discos exigidos	Disponibilidade dos dados	Capacidade para grande transferência de dados de E/S	Taxa para pequena solicitação de E/S
Striping	0	Não redundante	*N*	Menor que disco único	Muito alta	Muito alta para leitura e gravação
Espelhamento	1	Espelhado	2*N*	Maior que RAID 2, 3, 4 ou 5; menor que RAID 6	Maior que único disco para leitura; semelhante a único disco para gravação	Até o dobro de um único disco para leitura; semelhante a único disco para gravação
Acesso paralelo	2	Redundante via código de Hamming	$N + m$	Muito mais alta que único disco; comparável a RAID 3, 4 ou 5	Mais alta de todas as alternativas listadas	Aproximadamente o dobro de um único disco
	3	Paridade de bit intercalada	$N + 1$	Muito mais alta que único disco; comparável a RAID 2, 4 ou 5	Mais alta de todas as alternativas listadas	Aproximadamente o dobro de um único disco
Acesso independente	4	Paridade de bloco intercalada	$N + 1$	Muito mais alta que único disco; comparável a RAID 2, 3 ou 5	Semelhante a RAID 0 para leitura; muito menor que único disco para gravação	Semelhante a RAID 0 para leitura; muito menor que único disco para gravação
	5	Paridade de bloco distribuída e intercalada	$N + 1$	Muito mais alta que único disco; comparável a RAID 2, 3 ou 4	Semelhante a RAID 0 para leitura/ menor que único disco para gravação	Semelhante a RAID 0 para leitura; geralmente, menor que único disco para gravação
	6	Paridade de bloco dual distribuída e intercalada	$N + 2$	Mais alta de todas as alternativas listadas	Semelhante a RAID 0 para leitura; menor que RAID 5 para gravação	Semelhante a RAID 0 para leitura; muito menor que RAID 5 para gravação

Leituras Sugeridas

- Silberschatz, A., Galvin, P. B. Gagne, G. Sistemas Operacionais com Java. 7º edição. Editora Campus, 2008.
  - Capítulo 12.



- TANENBAUM, A. Sistemas Operacionais Modernos. Rio de Janeiro: Pearson, 3 ed. 2010
  - Capítulo 5.



Exercício

- Aplicar os algoritmos de escalonamento no disco, FCFS, SSTF e *SCAN* para a seguinte sequência de requisições:
  - Sequência: 10, 22, 20, 2, 30 e 6;
  - Posição inicial – cilindro 20;
  - Disco com 36 cilindros;
  - Tempo de cada movimentação: 6 mseg;
  - Fazer:
    - Fazer um gráfico para cada algoritmo;
    - Calcular o número de movimentos do braço e o tempo total para o atendimento de todas as requisições;
  - No *SCAN* considerar movimento inicial para a direita*;*