

# Sistemas Operacionais

---

Entrada e saída



2ª edição

Capítulo 5

Revisão: Fev/2003

# Sumário

---

- ⇒ ■ Princípios básicos de hardware
  - Arquitetura de computadores
- Gerência de entrada e saída
  - Software de entrada e saída
- Disco magnético

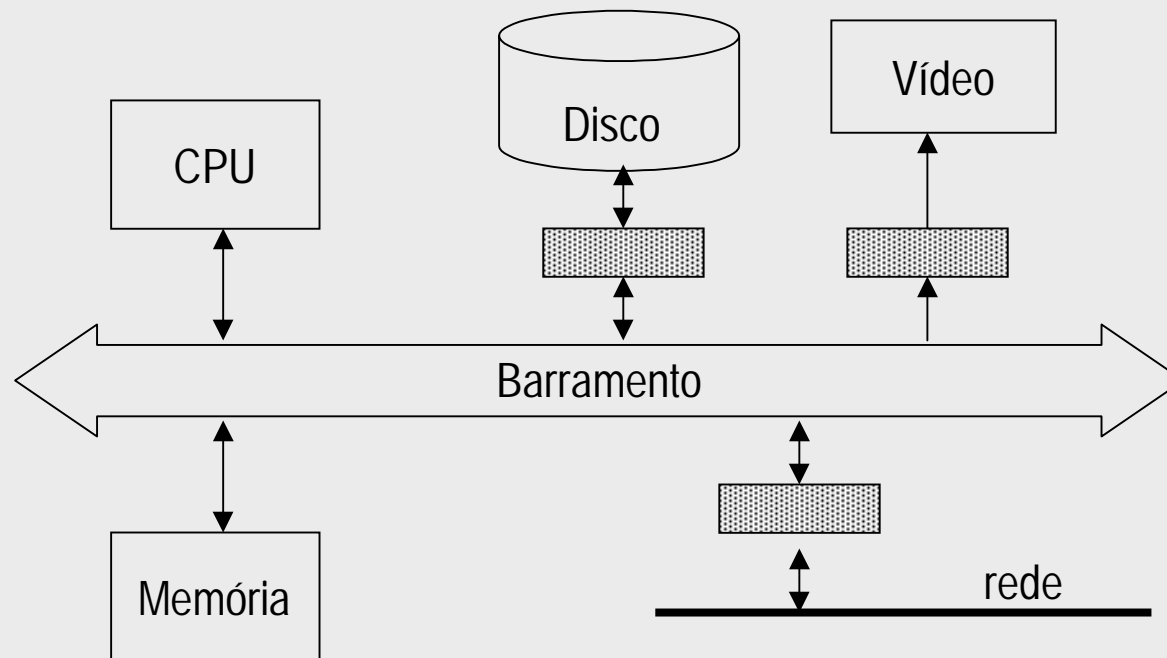
# Princípios básicos de hardware

---

- Periférico é um dispositivo conectado a um computador de forma a possibilitar sua interação com o mundo externo
- Os periféricos são conectados ao computador através de um componente de hardware denominado de interface
- As interfaces são interconectadas aos barramentos internos de um computador
  - Elemento chave na coordenação da transferências de dados
- Interfaces se utilizam de um processador dedicado a realização e controle das operações de entrada e saída
  - Controladoras

# Arquitetura de entrada e saída

- Dispositivo de entrada e saída possui uma parte mecânica e outra eletrônica



# Dispositivos de entrada e saída (1)

---

- Classificados como:
  - Orientado a caractere
    - Unidade de transferência é o caractere
      - e.g.; teclado, interface serial
  - Orientado a bloco
    - Unidade de transferência de dados é um bloco de caracteres (fixo)
      - e.g.; disco, fitas DAT
- Esquema de classificação não é perfeito pois alguns dispositivos não se enquadram nestas situações
  - e.g.; relógio, memória de vídeo mapeada em espaço de E/S

# Dispositivos de entrada e saída (2)

---

- Dispositivos de entrada e saída podem ser classificados de acordo com o tipo de entidade que interagem
  - Com usuário
    - e.g.; vídeo, teclado, mouse, impressora, etc
  - Com dispositivos eletrônicos
    - e.g; discos, fitas, atuadores, controladores, etc
  - Com dispositivos remotos
    - e.g.; modem, interfaces de rede

# Dispositivos de entrada e saída (3)

---

- Apresentam características próprias
  - Taxa de transferência de dados
  - Complexidade de controle
  - Unidade de transferência
    - Caractere, bloco ou stream
  - Representação de dados
    - Esquemas de codificação
  - Tratamento de erros
    - Depende do tipo de dispositivo

# Tipos de conexão e transferência de dados

---

- Em função da interconexão física das interfaces com os periféricos podem ser classificadas em dois tipos:
  - Interface serial
  - Interface paralela
- Interface serial
  - Apenas uma linha para transferência de dados (bit à bit)
- Interface paralela
  - Mais de uma linha para transferência de dados
    - e.g.;  $n \times 8$  bits



# Como controladoras e sistema operacional interagem?

---

- Controladora é programada via registradores de configuração
  - Recebem ordens do processador
  - Fornecem estados de operação
  - Leitura e escrita de dados do periférico
- Registradores são "vistos" como posições de memória
  - E/S mapeada em espaço de E/S
  - E/S mapeada em espaço de memória

# Mapeamento em espaço de memória e em espaço de entrada e saída

---

- Espaço de endereçamento:
  - Conjunto de endereços de memória que o processador consegue endereçar
  - Definido no projeto de processador
    - Pode haver um único espaço de endereçamento
    - Pode haver um espaço de endereçamento dedicado a entrada e saída
- Instruções específicas para acessar um ou outro espaço de endereçamento
  - e.g. mov, in, out

# Maapeamento em espaço de memória

---

- Um único espaço de endereçamento
- No projeto do computador se reserva uma parte de sua área de endereçamento para acesso a periféricos (controladoras)
- Instruções de acesso a memória do tipo *mov end, dado* podem tanto referenciar uma posição real de memória como um registrador associado a um periférico de entrada/saída
- Exemplo:
  - Processadores da família Motorola

# Mapeamento em espaço de entrada e saída

---

- O processador possui duas áreas distintas de endereçamento
  - Espaço de memória: acessado via instruções de acesso de memória (*mov*)
  - Espaço de E/S: acessado via instruções de acesso específica (*in, out*)
- No projeto de um computador (sistema) usando tal processador é possível de utilizar os dois tipos de mapeamento para acesso a periféricos de entrada e saída
- Exemplo:
  - Processadores da família Intel

# Exemplo de acesso a dispositivos

---

- Controladora de impressão onde um registrador fornece o “status” da impressão ( end. 315H) e outro corresponde ao envio do caracter a ser impresso (end. 312H).

## Mapeado em memória

```
Le_status:    mov    AL, 315H
```

```
Print_char:   mov    AL, 65H  
              mov    312H, AL
```

## Mapeado em entrada e saída

```
Le_status:    in     AL, 315H
```

```
Print_char:   mov    AL, 65H  
              out    312H, AL
```

# Técnicas para realização de E/S

---

- E/S programada
- E/S orientada a interrupções (*interrupt driven*)
- Acesso direto a memória

# E/S programada (1)

---

- Toda interação entre o processador e o controlador é de responsabilidade exclusiva do programador
- Ciclo de funcionamento:
  - Envio de comando a controladora
  - Espera pela realização do comando
- Módulo (controladora) de entrada/saída atualiza bits de estado da operação
- Processador espera o término da operação (*busy waiting*)

# Desvantagem E/S programada

---

- Desperdício do tempo do processador para verificar continuamente o estado de uma operação de entrada e saída
  - Diferença de velocidade entre dispositivo de entrada e saída e processador
- Solução é inserir operações entre sucessivas consultas sobre o estado de uma operação de entrada e saída
  - *Polling*
- Problema é determinar a frequência para a realização do *polling*



# E/S orientada a interrupção (*interrupt driven*)

---

- Método utilizado para evitar o desperdício de tempo do método de *polling*
- Processador é interrompido quando o módulo de E/S está pronto
- Enquanto a interrupção não ocorre o processador está liberado para executar outras tarefas
- Processador é responsável por iniciar uma operação de entrada e saída
- Interrupção solicita atenção do processador para executar uma rotina específica ao final da operação de entrada e saída
  - Tratador de interrupção

# Desvantagem E/S orientada a interrupção

---

- Processador atua como um intermediário na transferência, pois cada palavra lida (escrita) passa pelo processador

# Acesso direto a memória

---

- DMA (*Direct Memory Access*)
- Transfere diretamente um bloco de dados entre a memória e o módulo de E/S
- O mecanismo de interrupção é utilizado para sinalizar final de tarefa
- Processador é envolvido com a tarefa de E/S apenas no começo e no final da transferência

# Evolução de arquiteturas de entrada e saída

---

- Processador diretamente controla o periférico
- Controlador ou módulo de E/S é adicionado
  - Processador emprega E/S programada sem interrupções
  - Processador não necessita tratar detalhes dos dispositivos de E/S
- Controlador ou módulo de E/S porém baseado em interrupções
- Transferência de dados em DMA
- Módulo de E/S possui um processador separado
- Processador de E/S
  - computador dedicado ao processamento de E/S
  - Transferência de dados é feita, por exemplo, via rede

# Leituras complementares

---

- R. Oliveira, A. Carissimi, S. Toscani; Sistemas Operacionais. Editora Sagra-Luzzato, 2001.
  - Capítulo 5, seção 5.1
- A. Silberchatz, P. Galvin; Operating System Concepts. 4ª edição. Addison-Wesley.
  - Capítulo 2

# Sumário

---

- Princípios básicos de hardware
  - Arquitetura de computadores
- ⇒ ■ Gerência de entrada e saída
  - Software de entrada e saída
- Disco magnético

# O problema da gerência de entrada e saída

---

- Entrada e saída é extremamente lenta se comparada com a velocidade de processamento e de acesso a memória principal
- Multiprogramação possibilita que processos executem enquanto outros esperam por operações de entrada e saída
- Procedimento de *swapping* é entrada e saída
- Eficiência no tratamento de entrada e saída é importante

# Objetivos da gerência de entrada e saída

---

- Eficiência
- Generacidade é importante
  - Desejável que dispositivos sejam tratados da forma mais uniforme possível
- Esconder os detalhes do serviço de entrada e saída em camadas de mais baixo nível
- Fornecer ao alto nível abstrações genéricas como *read*, *write*, *open* e *close*
- Envolve aspectos de hardware e de software

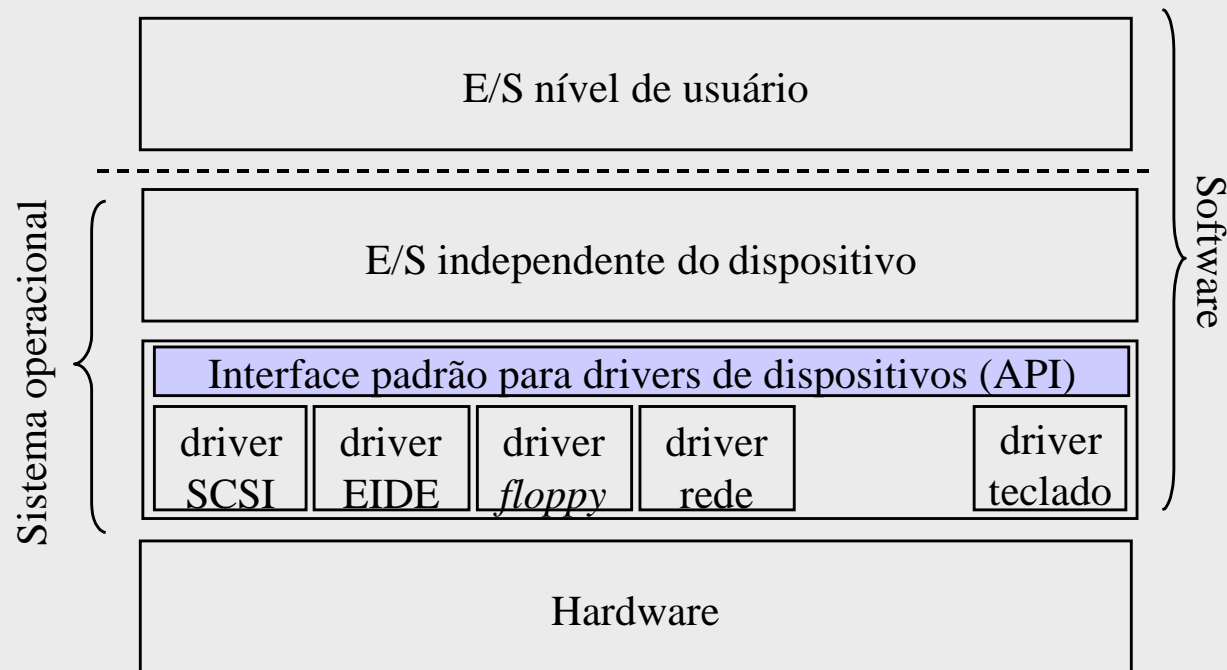


# Princípios básicos de software de entrada e saída

---

- Subsistema de entrada e saída é software bastante complexo devido a diversidade de periféricos
- Objetivo é padronizar as rotinas de acesso aos periféricos de E/S de forma a reduzir o número de rotinas
  - Permite inclusão de novos dispositivos sem alterar “visão” do usuário (interface de utilização)
- Para atingir esse objetivo o subsistema de E/S é organizado em camadas

# Estrutura em camadas do subsistema de E/S



# Organização lógica do software de E/S (1)

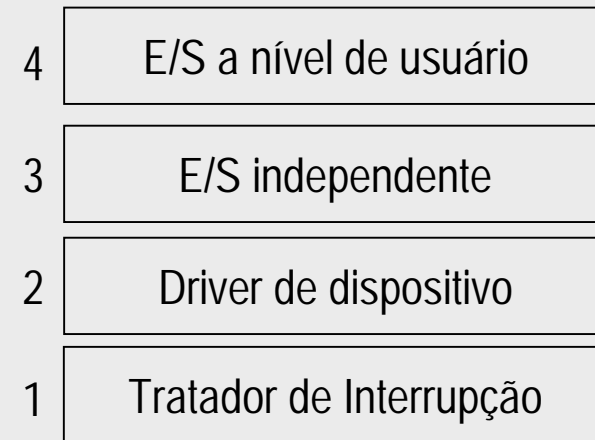
---

- Segue uma filosofia modular (três módulos base)
  - Dispositivo lógico de E/S
    - Trata dispositivo como recurso lógico sem se preocupar com detalhes de controle
    - Oferece uma interface de programação ao processo usuário
  - Dispositivo físico de E/S
    - aceita solicitações abstratas e converte para um tratamento específico
    - Bufferização
  - Escalonamento e controle
    - Tratamento de interrupções e de status
- Organização não é única
  - Depende do tipo do dispositivo e de sua aplicação

# Visão geral do software de E/S

---

- Tratador de interrupção
  - aciona driver ao final da operação de transferência
- Driver de dispositivo
  - Configuração controladora e status
  - Recebimento de requisições
- E/S independente do dispositivo
  - Nomeação e proteção
  - Bufferização
- E/S a nível de usuário
  - Chamadas de E/S
  - Formatação de E/S



# *Device driver*

---

- Conjunto de estruturas de dados e funções que controlam um ou mais dispositivos interagindo com o núcleo via uma interface bem definida
- Fornecido pelo fabricante do periférico
- Vantagens:
  - Isolar código específico a um dispositivo em um módulo a parte
  - Facilidade de adicionar novos drivers
  - Fabricantes não necessitam “mexer” no núcleo
  - Núcleo tem uma visão uniforme de todos os dispositivos através de uma mesma interface

# Funcionamento básico

---

- Núcleo aciona *driver* para:
  - Configurar dispositivo
  - Realizar acessos de leitura e escrita
  - Controlar e obter informações de status
  - Tratamento de interrupções
- *Driver* aciona núcleo para:
  - Efetuar gerenciamento de buffers
  - Escalonamento

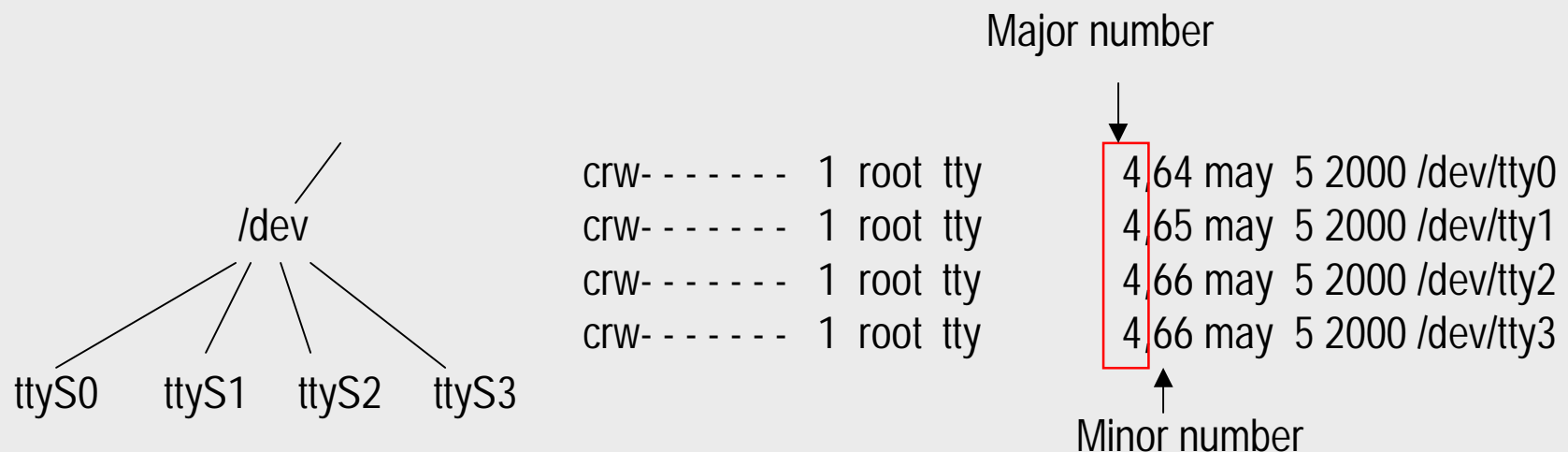
# Subsistema de E/S

---

- Porção do núcleo que controla a parte independente e interage com o *driver* de dispositivo para manipular/tratar a parte dependente
- Responsável por:
  - Distribuição uniforme dos nomes (nomeação, espaço de nomes)
  - Proteção
  - Fornecer uma interface aos processos usuários (aplicativos)
  - Gerenciamento e desempenho do sistema de E/S

# Exemplo: subsistema de E/S do UNIX

- Corresponde a visão lógica do dispositivo
- Atribuição uniforme do nome independente do dispositivo
- O UNIX é um exemplo clássico:
  - Nome do dispositivo é um *string*
  - Discos (dispositivos) fazem parte do sistema de arquivos





# Funcionalidades básicas do subsistema de E/S

---

- Escalonamento de E/S
  - Determinar a melhor ordem para o atendimento de requisições de E/S
  - Dividir de forma justa o acesso a periféricos
- Bufferização
  - Área de armazenamento temporário de dados
- Cache
  - Permitir o acesso rápido aos dados
- *Spooling*
  - Controlar acesso a dispositivos que atendem apenas uma requisição por vez
- Reserva de dispositivos
  - Controlar acesso exclusivo a dispositivos (alocação, desalocação, *deadlock*)

# E/S nível de usuário

---

- Disponibiliza a processos usuário (aplicação) operações de E/S através de bibliotecas ou chamadas de sistema

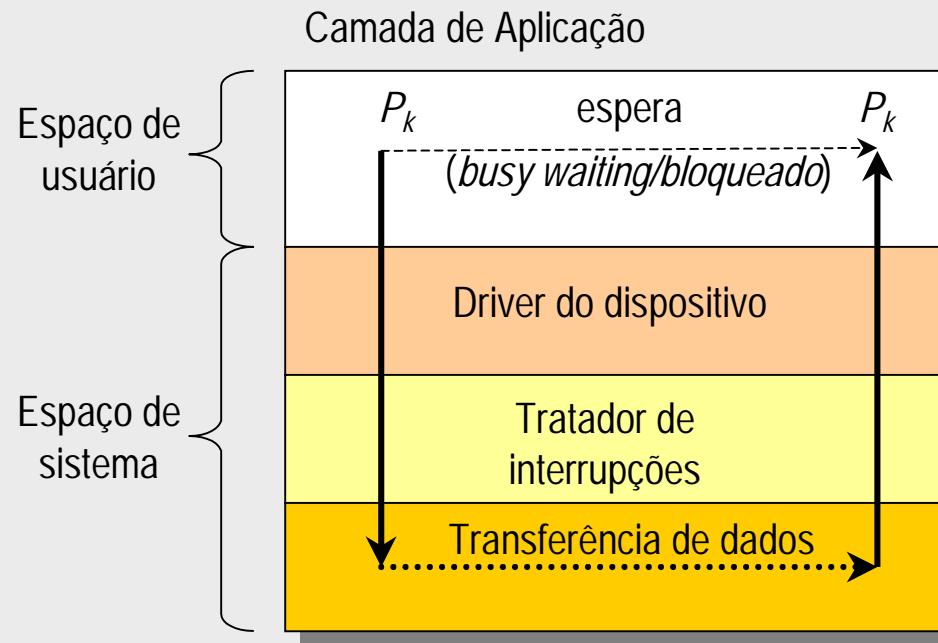
# Interface de programação

---

- Bloqueante
  - Processo é suspenso até a conclusão da operação
    - Fácil programação e compreensão
- Não bloqueante
  - Retorna imediatamente com os dados disponíveis no momento
    - Baseado em buffer
- Assíncrona
  - Processo continua sua execução enquanto a E/S é realizada
    - Difícil programação
    - Necessário determinar o término da operação (*signal* versus *polling*)

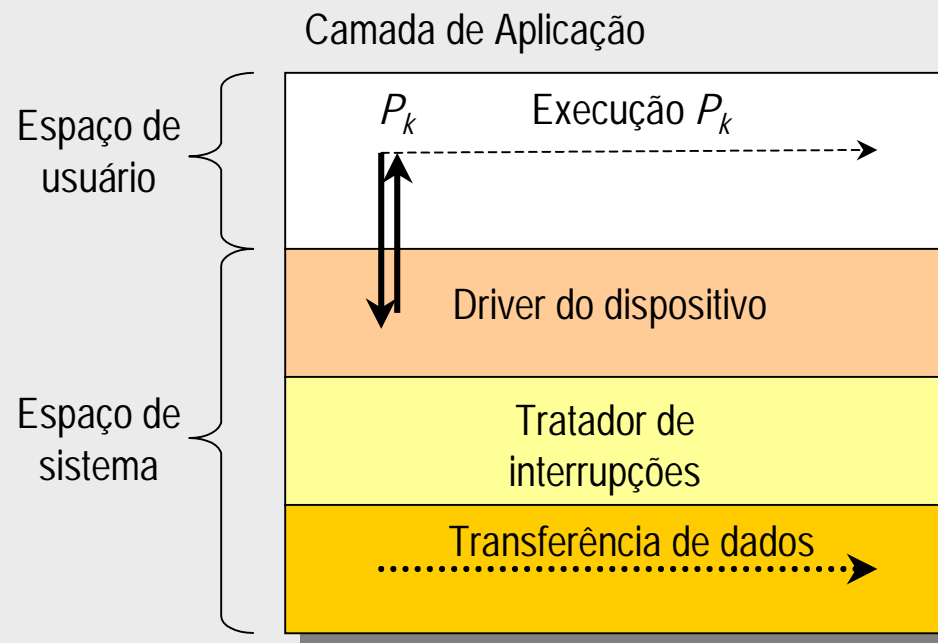
# Bloqueante

- O “controle” retorna a camada de aplicação somente após a conclusão da operação de entrada e saída



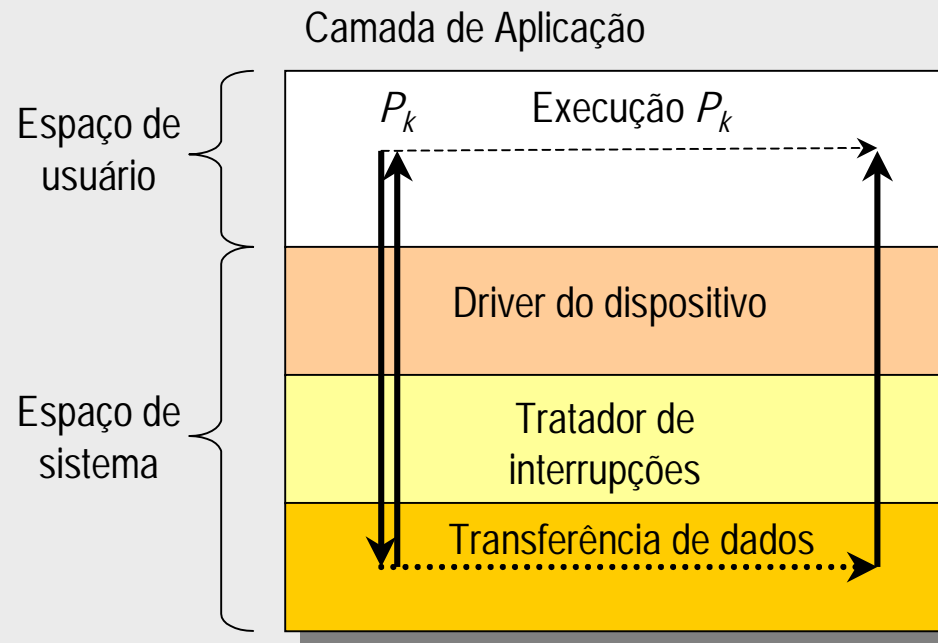
# Não bloqueante

- O “controle” retorna a camada de aplicação com os dados disponíveis no momento



# Assíncrona

- O “controle” retorna a camada de aplicação sem que a operação de entrada e saída tenha sido concluída
  - Programação (agendamento) de uma operação de E/S



# Leituras complementares

---

- R. Oliveira, A. Carissimi, S. Toscani; Sistemas Operacionais. Editora Sagra-Luzzato, 2001.
  - Capítulo 5, seção 5.2
- A. Silberchatz, P. Galvin; Operating System Concepts. 4ª edição. Addison-Wesley.
  - Capítulo 2

# Sumário

---

- Princípios básicos de hardware
  - Arquitetura de computadores
- Gerência de entrada e saída
  - Software de entrada e saída



- Disco magnético



# Dispositivos periféricos típicos

---

- Dispositivos de E/S são fundamentais para que um sistema seja utilizável
- Existe uma grande gama de dispositivos de E/S
  - Impossível de analisar todos
  - Princípio de funcionamento tem uma base comum
- Periférico mais importante é o disco por desempenhar um papel fundamental em diversos aspectos do sistema operacional
  - Armazenamento de dados
  - Suporte a implementação de memória virtual
  - Aspectos relacionados com tolerância a falhas e segurança de dados (RAID)

# Disco magnético

---

- Um disco de plástico, ou metal, recoberto de material magnético
  - Pratos (*platters*)
- Dados são gravados e, posteriormente, recuperados através de um "mola" condutora (cabeçote de leitura e gravação)
  - Escrita: o cabeçote é submetido a uma tensão que gera um campo magnético o qual magnetiza o disco com diferentes padrões de polaridades
  - Leitura: a variação do campo magnético gerado pela rotação do disco induz uma corrente no cabeçote de leitura

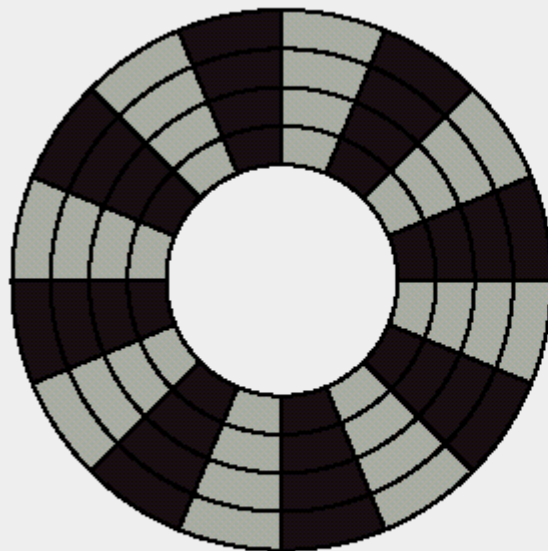
# Organização e formatação (1)

---

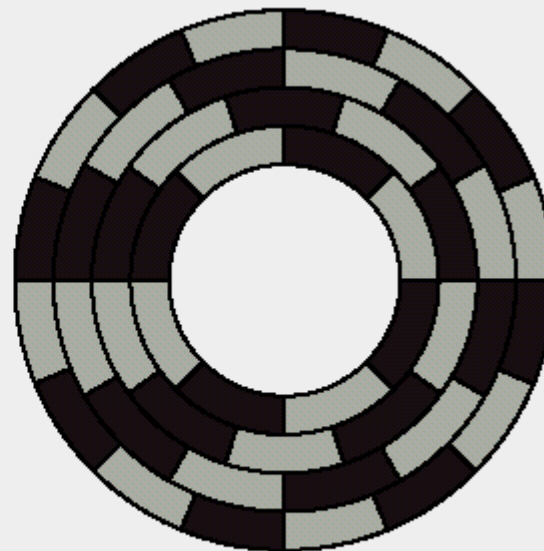
- Disco é organizado em uma seqüência de círculos concêntricos
  - Trilhas
  - Largura da trilha corresponde a largura do cabeçote de leitura/escrita
- Trilhas são separadas por gaps
  - Evitar problemas de alinhamentos
- Duas tecnologias definem o número de bits por trilhas
  - O número de bits por trilha é constante
    - Trilhas mais internas possuem uma densidade maior bits/polegada
    - Discos com tecnologia CAV (*Constant Angular Velocity*)
  - O número de bits por trilha depende se ela é mais interna ou mais externa
    - Discos com tecnologia CLV (*Constant Linear Velocity*)
      - e.g.; CDRom

# Organização e formatação (2)

---



(a) Constant angular velocity



(b) Constant linear velocity

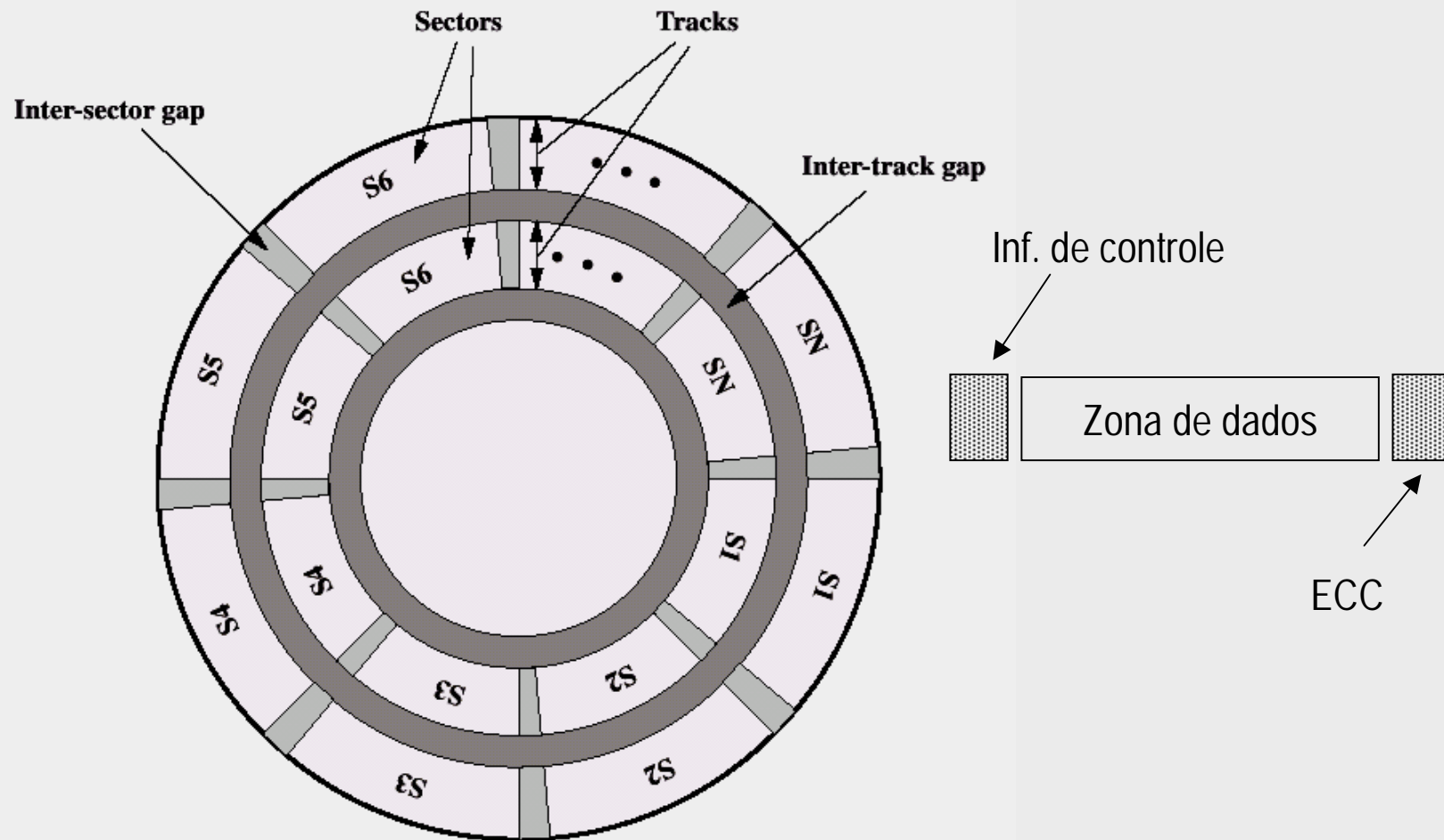
W. Stallings. *Operating Systems (4th Ed.)*, Prentice Hall, 2001.

# Organização e formatação (3)

---

- Transferência de dados para o disco é feito em blocos
  - Tipicamente, bloco é menor que a capacidade de uma trilha
- Trilha é subdividida em unidades de tamanho fixo (setores)
- Setor:
  - Armazenamento de informações
  - Informações de controle
    - e.g.; início e final do setor, ECC (*Error Correcting Code*)
- A definição de trilhas e setores é feita pela formatação física
  - Feita na fábrica

# Organização e formatação (4)



# Características físicas (1)

---

- Cabeçote de leitura/escrita são montados sobre um braço
  - Fixo: um cabeçote por trilha
  - Móvel: cabeçote se desloca sobre as trilhas
- Removível versus não removível
  - Meio magnético (pratos) são montados fisicamente no braço ou não
    - e.g.; disquete
- Densidade dupla versus densidade simples
  - Filme magnético é posto, ou não, nas duas superfícies do prato

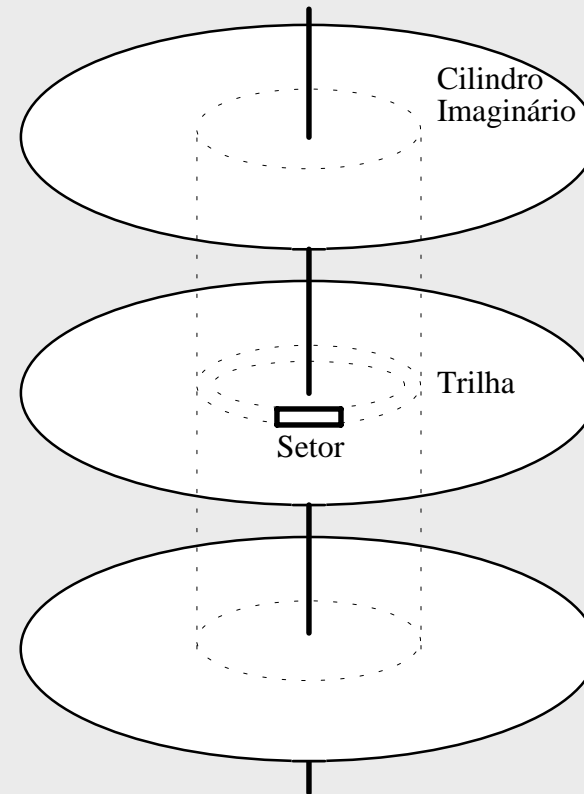
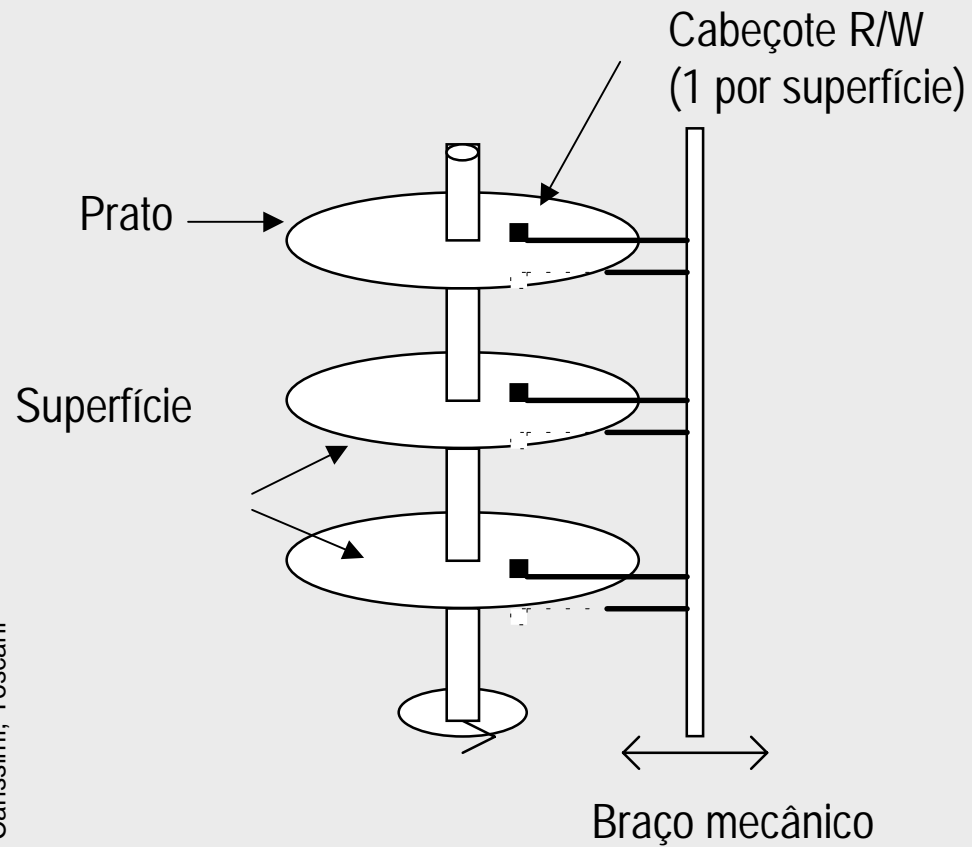
# Características físicas (2)

---

- Múltiplos pratos (*disk pack*)
  - Vários pratos empilhados e centrados
  - Cada prato um cabeçote de leitura/escrita (braço móvel)
  - Cilindro: conjunto de trilhas de mesmo número em pratos diferentes
- Mecanismo do cabeçote leitura/escrita
  - Contato físico entre a superfície magnética e o cabeçote (floppy)
  - Distância fixa (*air gap*) da superfície magnética
  - Distância fixa (*air gap*) da superfície magnética quando o disco entra em rotação (disco rígido)



# Características físicas (3)



# Exemplo de especificações de disco

---

- Disco 4.1 Gigabytes
  - 255 cabeças
  - 63 setores de 512 bytes
  - 525 cilindros
  - Capacidade:  $255 \times 63 \times 512 \times 525$
- Sexagésimo quarto setor mantém um mapa de setores na trilha
- Tecnologia atual permite até 8 pratos com 16 cabeçotes
  - Diferença no número de cabeçotes é lógico

# Acesso a dados

---

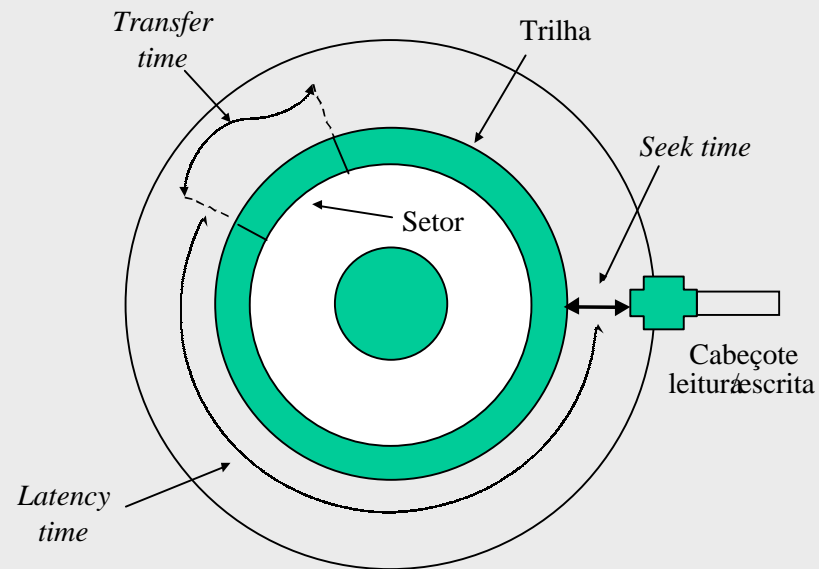
- Menor unidade de transferência é um bloco lógico
  - Composto por um ou mais setores físicos
- Acessar dado implica em localizar trilha, superfície e setor
- Dois métodos:
  - Método CHS (*Cylinder, Head, Sector*)
  - Método LBA (*Linear Block Addressage*)
    - Tradução do L-CHS (*Logical*) para P-CHS (*Physical*)
- Discos modernos endereçam blocos lógicos sequencialmente
  - Conversão de um bloco lógico para sua localização física
    - Não é um mapeamento direto por haver setores fisicamente defeituosos e pelo número de setores por trilha não ser constante
      - Cilindros que possuem mesmo número de setores (zonas)

# Desempenho do disco

---

- Para ler/escrever dados é necessário que o cabeçote de leitura e escrita esteja posicionada na trilha e no início do setor desejados
- Três tempos envolvidos:
  - Tempo de posicionamento (*seek time*)
    - Tempo necessário para posicionar o cabeçote de leitura/escrita na trilha desejada
  - Tempo de latência rotacional
    - Tempo necessário para atingir o início do setor a ser lido/escrito
  - Tempo de transferência
    - Tempo para escrita/leitura efetiva dos dados

# Temporização de acesso ao disco



$$t_{\text{acesso}} = t_{\text{seek}} + t_{\text{latência}} + t_{\text{trasnf.}}$$

# Tempo de posicionamento (*seek*)

---

- Possui duas componentes:
  - Tempo de acionamento e aceleração do braço do cabeçote
  - Tempo de deslocamento até a trilha desejada
- Não é linear em função do número do trilhas
  - Tempo de identificação da trilha (confirmação)
- Tempo médio de *seek*
  - Dado fornecido pelo fabricante
    - e.g.; 5 a 10 ms (tecnologia ano 2000)

# Tempo de latência rotacional

---

- Definido pela velocidade de rotação do motor
  - e.g. (ano 2000):
    - discos rígidos (5400 rpm a 10000 rpm)
    - unidades de *floppy* (300 rpm a 600 rpm)
- Considera-se o tempo médio
  - Não se sabe a posição relativa do cabeçote com a do setor a ser lido
  - Metade do tempo de uma rotação
    - e.g.; 3 ms para um disco de 10000 rpm ( 6 ms uma rotação )

# Tempo de transferência

---

- Tempo de transferência de dados de/para disco depende da velocidade de rotação

$$T = \frac{b}{rN}$$

T = tempo de transferência

b = número de bytes a serem transferidos

N = número de bytes em uma trilha

r = velocidade de rotação, nro de rotações por segundo

- Tempo médio de acesso é dado por:

$$T_{acesso} = t_{seek\_médio} + \frac{1}{2r} + \frac{b}{rN}$$



# Exemplo

---

- Acessar um arquivo de dados de 1.3 Mbytes armazenado em disco com as seguintes características:

$T_{\text{seek\_médio}} = 10 \text{ ms}$ , 10000 rpms, 512 bytes por setor, 320 setores por trilha

Caso I: Acesso seqüencial (2560 setores = 8 trilhas x 320 setores)

$$T_{\text{acesso}} = 10 \text{ ms} + 8 \times (3 + 6) \text{ ms} = 0.082 \text{ s}$$

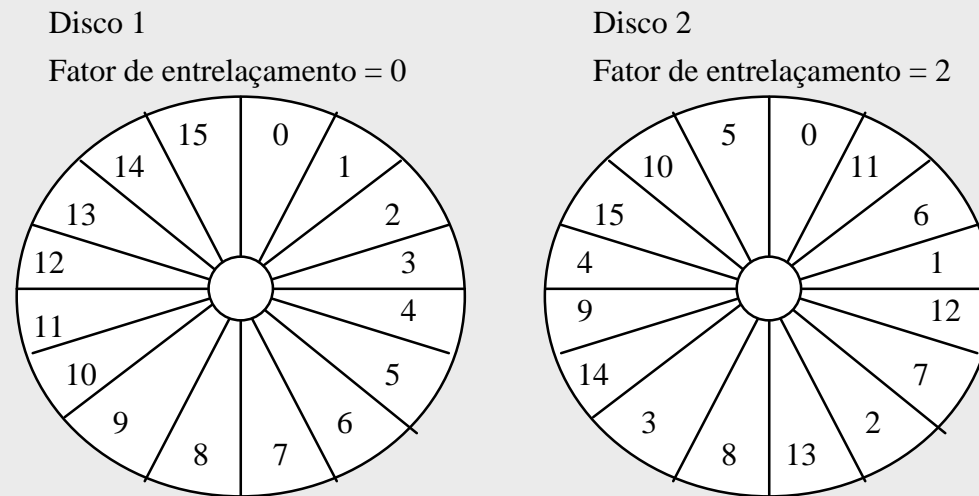
(Obs: supondo trilhas vizinhas, despreza-se o tempo de *seek*)

Caso II: Acesso randômico (leitura na base um setor por vez)

$$T_{\text{acesso}} = 2560 \times (10 \text{ ms} + 3 \text{ ms} + 0.01875 \text{ ms}) = 33.328 \text{ s}$$

# Entrelaçamento (*interleaving*)

- Forma de aumentar o desempenho no acesso ao disco
- Objetivo é evitar a latência rotacional em setores adjacentes
- Técnica consiste em numerar os setores não mais de forma contígua mas sim com um espaço entre eles



# Escalonamento do disco (1)

---

- Tratar E/S em disco de forma eficiente se traduz em obter um tempo de acesso rápido e explorar ao máximo a largura de banda do disco
  - Se traduz em minimizar o tempo de posicionamento (seek)
- Largura de banda do disco é definida como sendo o número total de bytes transferidos, divididos pelo tempo decorrido entre o primeiro acesso e a conclusão da transferência.

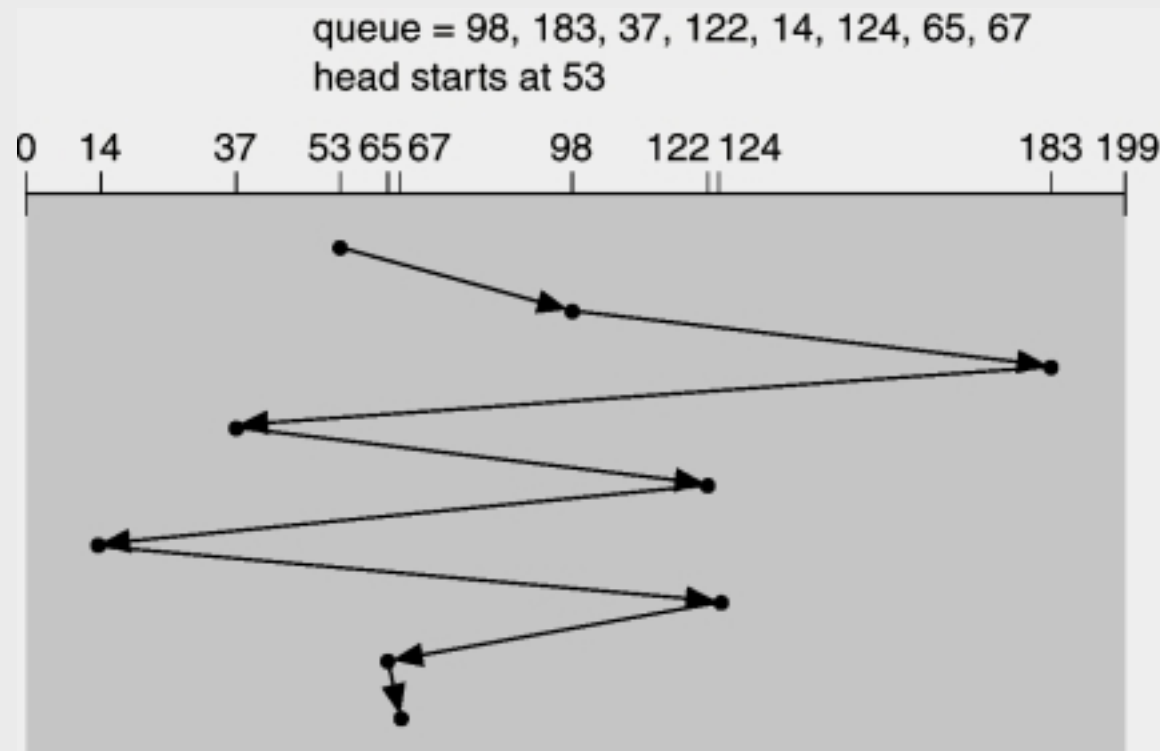
# Escalonamento do disco (2)

---

- Algoritmos para reduzir o tempo de *seek*
  - Algoritmos de escalonamento do disco
    - Forma de organizar o atendimento a requisições
      - FCFS, SSTF, SCAN, C-SCAN, etc
- Exemplo para análise
  - Disco organizado em 200 trilhas (0-199)
  - Posição inicial do cabeçote: trilha 53
  - Atender a seguinte fila de requisições:  
98, 183, 37, 122, 14, 124, 65, 67

# FCFS - *First Come First Served* (1)

- Acessa na ordem em que as requisições são solicitadas
- Para análise em questão obtem-se um deslocamento equivalente a 640 trilhas



Silberchatz, Galvin, Gagne. Applied Operating System Concepts (1st Ed.) John Wiley & Sons, 2000.

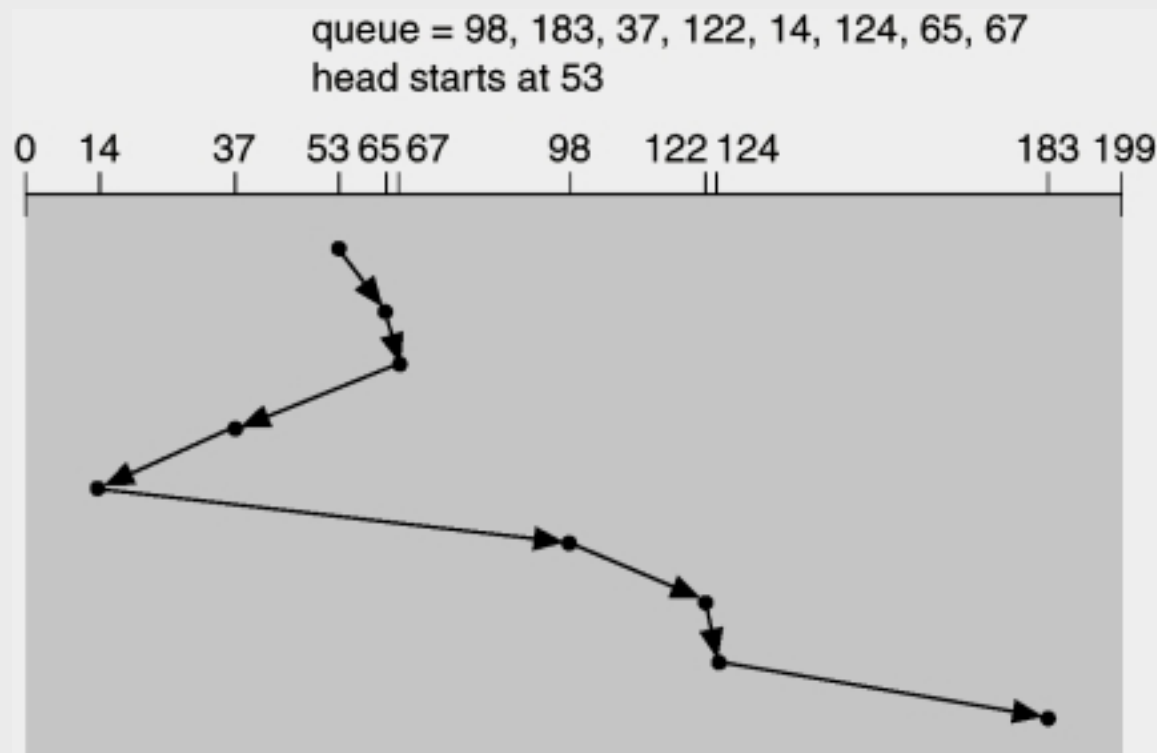
# SSTF - *Shortest Seek Time First* (1)

---

- Seleciona a requisição com o menor tempo de seek em relação a posição atual do cabeçote de leitura/escrita
- Análogo ao escalonamento SJF (*Shortest Job First*)
  - Pode provocar postergação de uma requisição

# SSTF - *Shortest Seek Time First* (2)

- Deslocamento equivalente a 236 trilhas



Silberchatz, Galvin, Gagne. Applied Operating System Concepts (1st Ed.) John Wiley & Sons, 2000.

# SCAN (1)

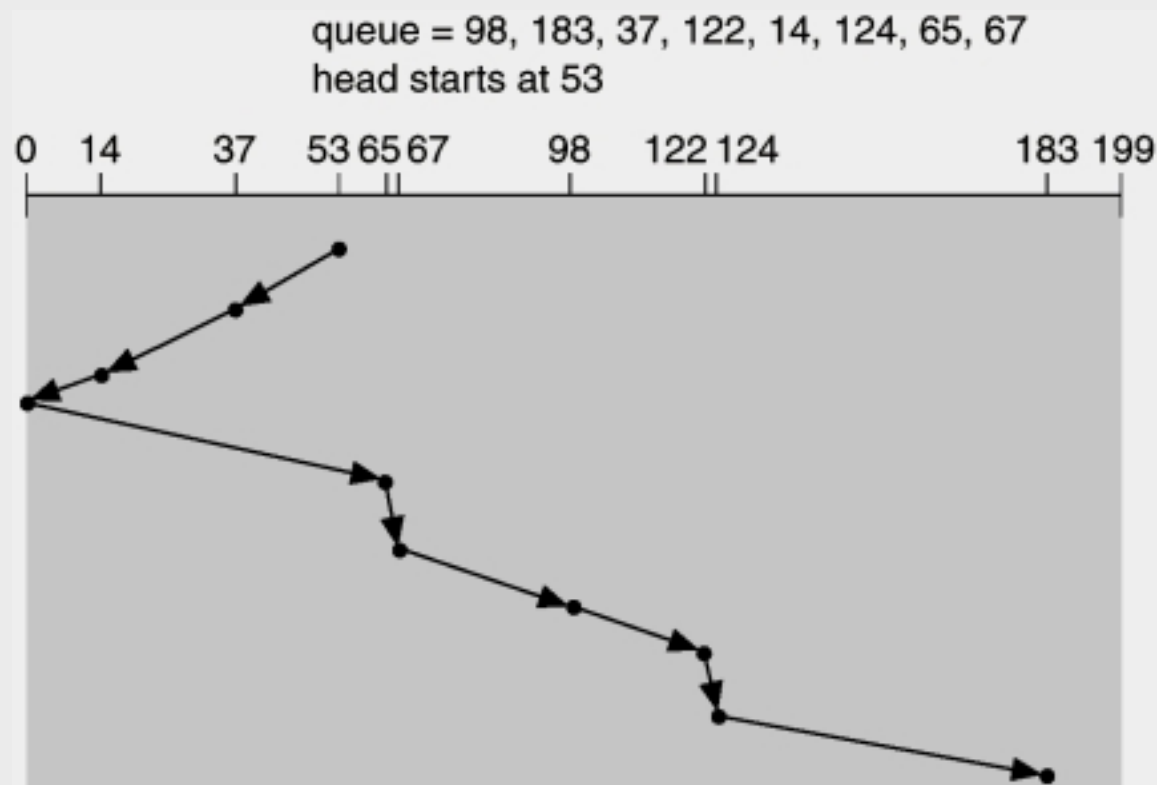
---

- 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 no outro extremo, inverte o sentido e repete o procedimento
- Conhecido como *algoritmo do elevador*



# SCAN (2)

- Deslocamento equivalente a 208 trilhas



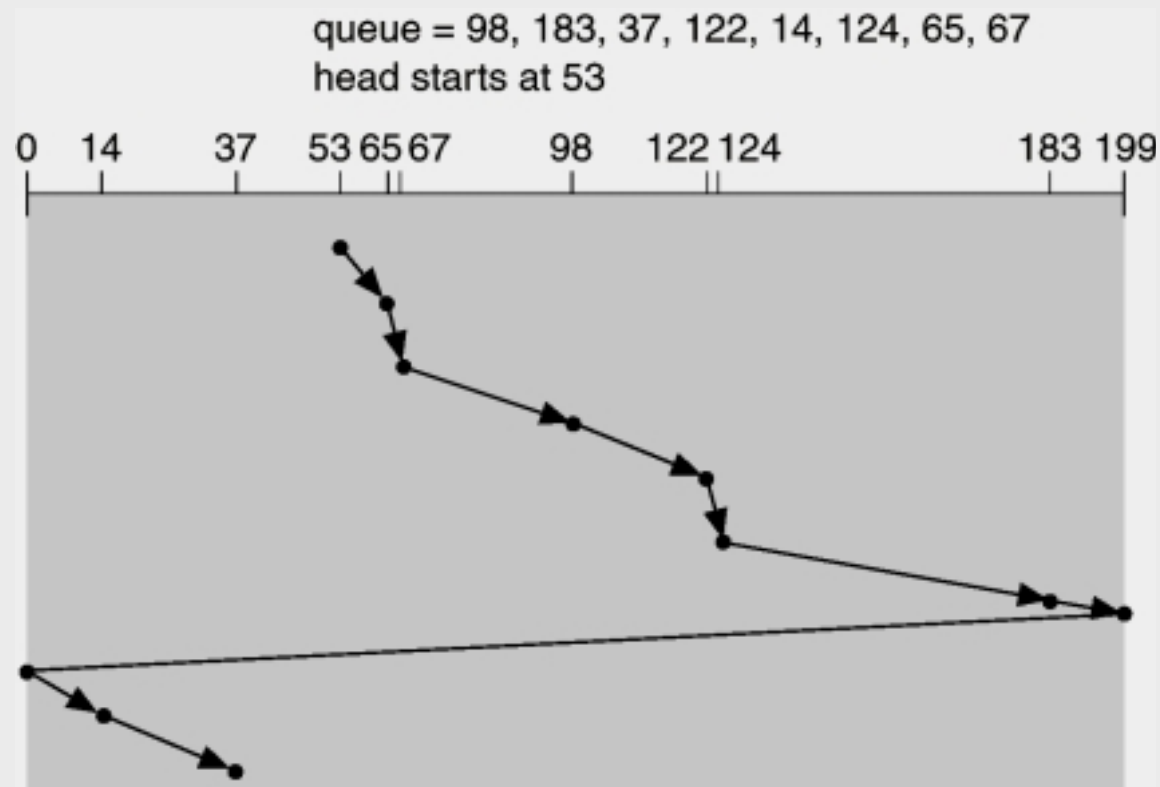
Silberchatz, Galvin, Gagne. Applied Operating System Concepts (1st Ed.) John Wiley & Sons, 2000.

# C-SCAN (1)

---

- 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 reposicionado 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 algoritmo SCAN

# C-SCAN (2)



Silberchatz, Galvin, Gagne. Applied Operating System Concepts (1st Ed.) John Wiley & Sons, 2000.

# C-LOOK

---

- Versão do C-SCAN
- O cabeçote de leitura/escrita não necessita atingir o extremo do disco para voltar ao início do disco

# Outras variações de SCAN

---

## ■ N-step-SCAN

- Divide a fila de requisições em um certo número de filas de tamanho N
- Cada fila é atendida separadamente utilizando SCAN
- Novas requisições são inseridas em filas diferentes da que está sendo atendida no momento da chegada destas requisições

## ■ FSCAN

- Baseada em duas filas
- Um fila recebe todas as novas requisições enquanto a outra é empregada para atender as requisições já feitas

# Qual algoritmo de escalonamento é melhor?

---

- SSTF é o método comumente empregado
- SCAN e C-SCAN apresentam um melhor desempenho em discos que possuem um grande número de acesso
- Fatores importantes
  - Quantidade e tipo de requisições
  - Organização de arquivos e diretórios no disco (sistema de arquivos)
- O algoritmo de escalonamento deve ser escrito como um módulo separado do sistema operacional para permitir sua substituição de forma fácil

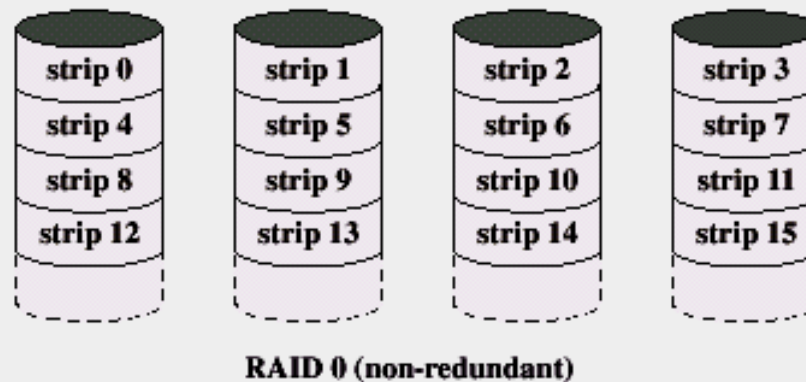
# RAID: *Redundant Array of Inexpensive Disks*

---

- Conjunto de discos rígidos visto pelo sistema operacional como um único disco lógico
- Dados são distribuídos entre os diferentes discos físicos
  - Permite o acesso paralelo a dados aumentando o desempenho
- Possibilita o armazenamento de informações de paridade para permitir a recuperação de dados em caso de problemas no disco
  - Aumento de possibilidade de falhas já que existem mais dispositivos envolvidos
- Diferentes níveis

# RAID nível 0

- Dados são distribuídos nos diferentes discos do *array*
  - Requisições a blocos de dados armazenados em discos distintos podem ser efetuados em paralelo
- O disco lógico é dividido em unidade de distribuição (*strips*)
  - *Strip* pode ser blocos físicos, setores, etc...
  - *Strip* são mapeados de forma round-robin em  $n$  discos (*stripes*)
  - Técnica conhecida como *stripping*



W. Stallings. *Operating Systems (4th Ed.)*, Prentice Hall, 2001.



# RAID nível 1

---

- Objetivo de RAID é fornecer uma redundância de dados para fornecer um certo grau de tolerância a falhas
- RAID 1 a redundância é obtida através da replicação dos dados
  - Strips são armazenados em 2 conjuntos distintos de discos físicos
  - Denominado de espelhamento (*mirroring*)

# RAID nível 1: vantagens e desvantagens

---

## ■ Vantagens:

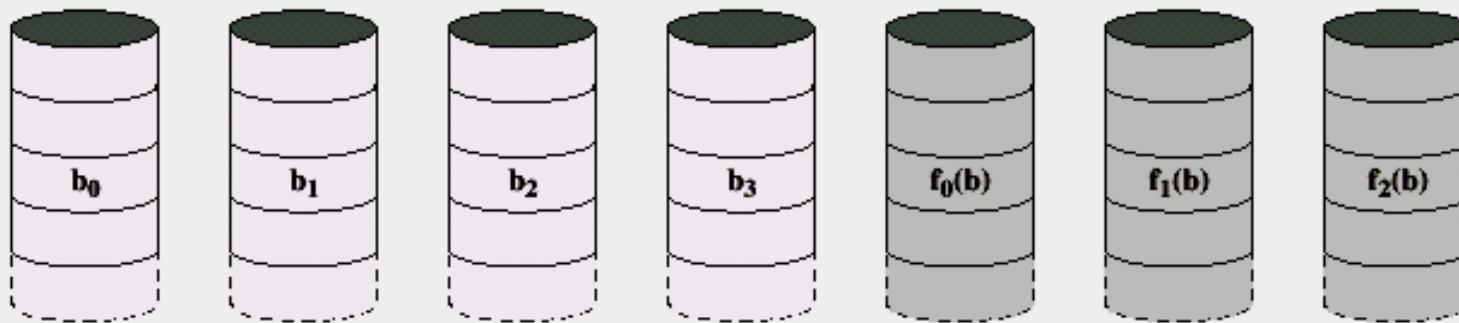
- Leitura de um dado pode ser feito privilegiando-se o disco que oferece o menor tempo de seek e atraso rotacional
- Recuperação em caso de erro é simples
  - Acesso dados do disco não danificado

## ■ Desvantagem:

- Custo: necessita o dobro do espaço do disco lógico em discos físicos

# RAID nível 2

- O conjunto de discos é sincronizado, isto é, todos os cabeçotes estão posicionados no mesmo ponto (trilha e setor)
- Todos discos são acessados na realização de uma requisição E/S
- A unidade de *stripping* é byte ou palavra
- Executa o cálculo de código de correção de erros considerando um certo número de bits e armazena o resultado em discos separados



**RAID 2 (redundancy through Hamming code)**

W. Stallings. *Operating Systems (4th Ed.)*, Prentice Hall, 2001.

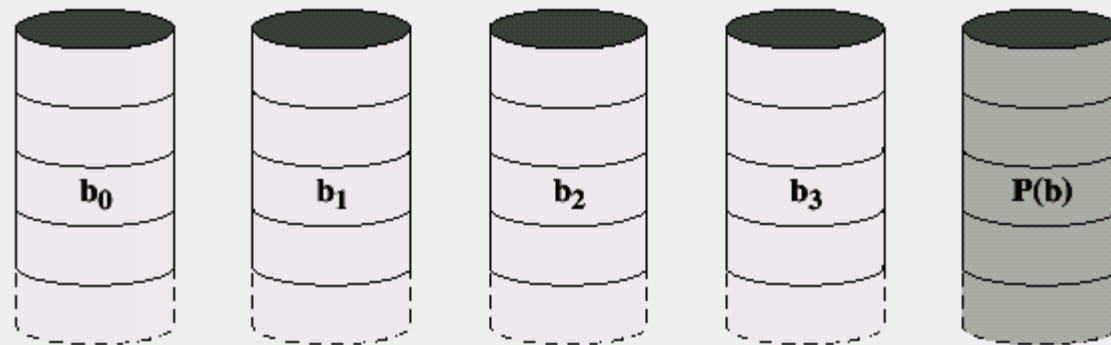
# RAID nível 2

---

- O número de discos redundantes é proporcional ao logaritmo da quantidade de dados armazenados no disco
- Requisição de E/S
  - Leitura: código de correção é calculado para os dados lidos e comparado com o código de correção lido
  - Escrita: cálculo e gravação do código de correção

# RAID nível 3

- Similar ao RAID 2
- Diferença é que existe apenas um disco de redundância independente do número de discos para armazenamento de dados
- Cálculo de um código de detecção de erro (paridade)
  - Possível reconstruir dados de um disco falho a partir desta informação



**RAID 3 (bit-interleaved parity)**

W. Stallings. *Operating Systems (4th Ed.)*, Prentice Hall, 2001.

# RAID nível 3: reconstrução de dados

---

## ■ Exemplo:

- Array composto por 5 discos físicos, onde discos 0 a 3 servem ao armazenamento de dados e o disco 4 a paridade (redundância)

Paridade para o bit  $i$  de cada disco é calculado da seguinte forma:

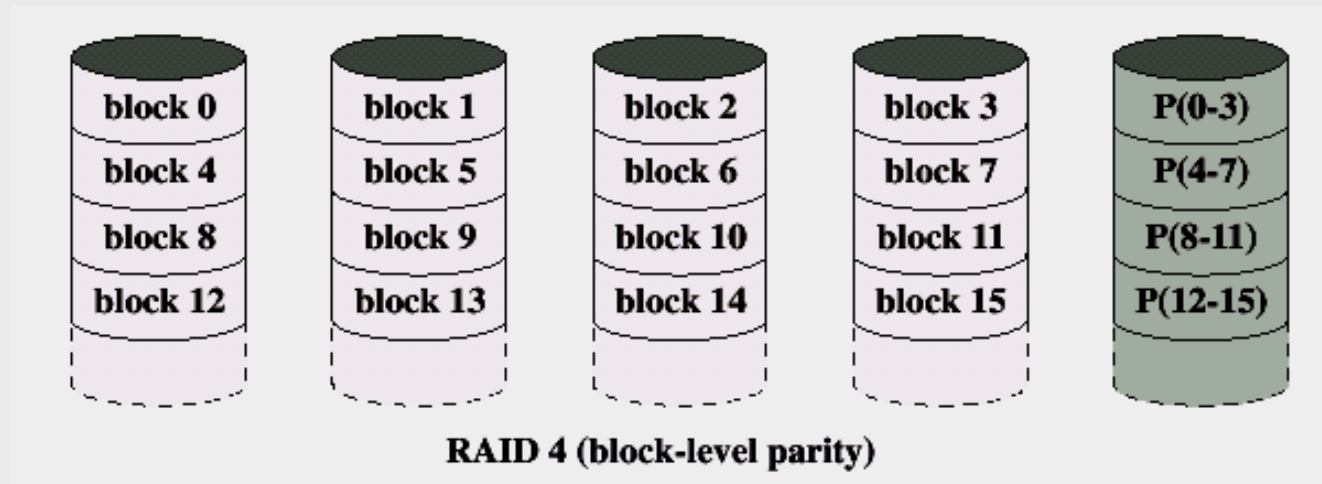
$$x4(i) = x3(i) \oplus x2(i) \oplus x1(i) \oplus x0(i)$$

Em caso de erro do disco 1, se pode reconstruir seus dados (bit a bit) realizando o seguinte cálculo:

$$x1(i) = x4(i) \oplus x3(i) \oplus x2(i) \oplus x0(i)$$

# RAID nível 4

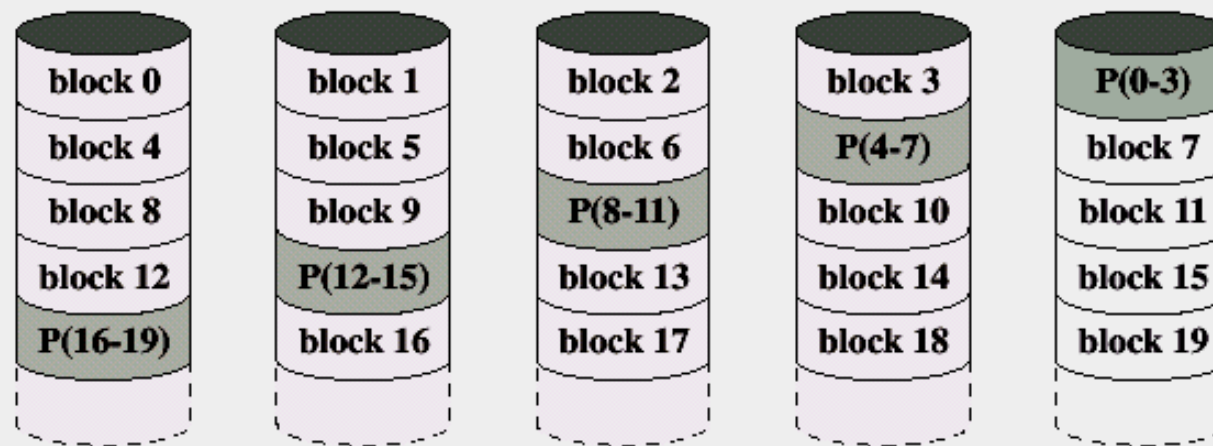
- Diferença em relação aos níveis 2 e 3 é que os discos são independentes podendo então satisfazer requisições em paralelo
- Redundância é obtida calculando-se a paridade bit a bit de cada strip e armazenando o resultado em disco de paridade



W. Stallings. *Operating Systems (4th Ed.)*, Prentice Hall, 2001.

# RAID nível 5

- Organização é similar ao RAID 4
- A informação de paridade é distribuída em todos os discos do array de forma *round-robin*

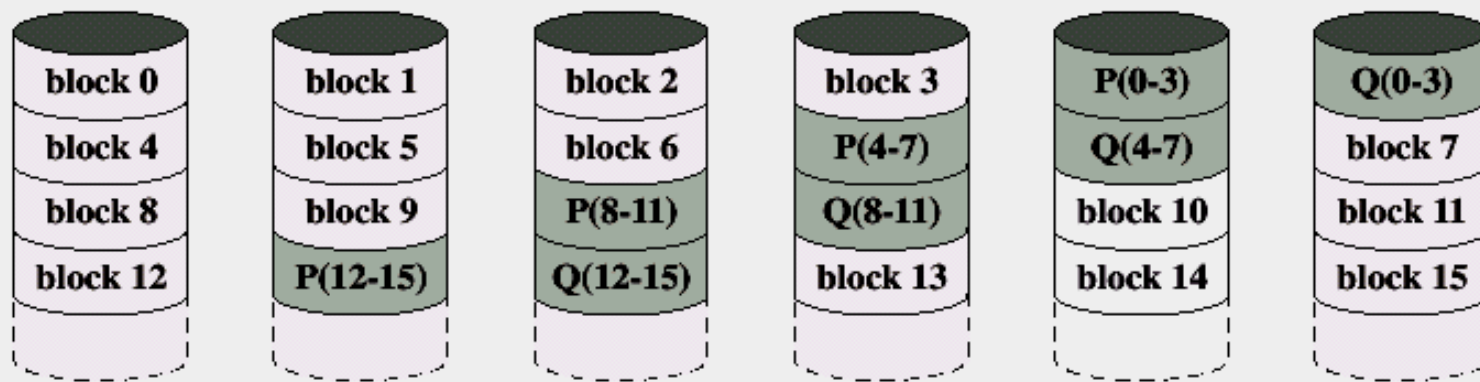


**RAID 5 (block-level distributed parity)**



# RAID nível 6

- Introduz um segundo cálculo de paridade para o mesmo conjunto de dados:
  - Paridade é calculada utilizando o esquema de OU exclusivo de RAID 4 e 5
  - Algoritmo adicional de verificação de dados
- Vantagem sobre demais esquemas de RAID é que dois discos podem falhar simultaneamente

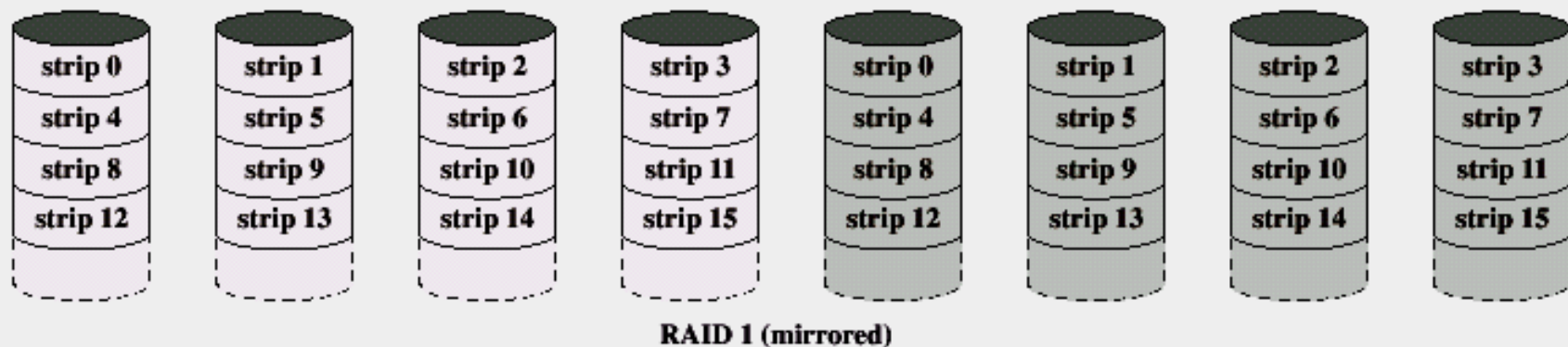


**RAID 6 (dual redundancy)**

W. Stallings. *Operating Systems (4th Ed.)*, Prentice Hall, 2001.

# RAID nível 10

- Combinação dos níveis RAID 0 e RAID 1
- Os dados são divididos em strips (RAID 0) e após os discos de armazenamento de dados são espelhados (RAID 1)



W. Stallings. *Operating Systems (4th Ed.)*, Prentice Hall, 2001.

# Software de RAID

---

- Configurações de RAID podem ser implementadas em hardware (firmware) ou em software
- RAID em hardware:
  - Diferentes discos físicos são organizados de forma a compor um disco lógico
  - Organização é configurada e gerenciada pelo controlador de disco
  - Controlador realiza a geração das informações de redundância (firmware)
- RAID em software:
  - Diferentes partições (discos lógicos) compõem um único disco
  - RAID é feito pelo driver de disco (software)
  - Normalmente implementa RAID nível 1 e RAID nível 5

# Leituras complementares

---

- R. Oliveira, A. Carissimi, S. Toscani; Sistemas Operacionais. Editora Sagra-Luzzato, 2001.
  - Capítulo 5, seção 5.3
- A. Silberchatz, P. Galvin, G. Gagne; Applied Operating System Concepts. Addison-Wesley, 2000.
  - Capítulo 12 e 13
- W. Stallings; Operating Systems. (4<sup>th</sup> edition). Prentice Hall, 2001.
  - Capítulo 11