

---

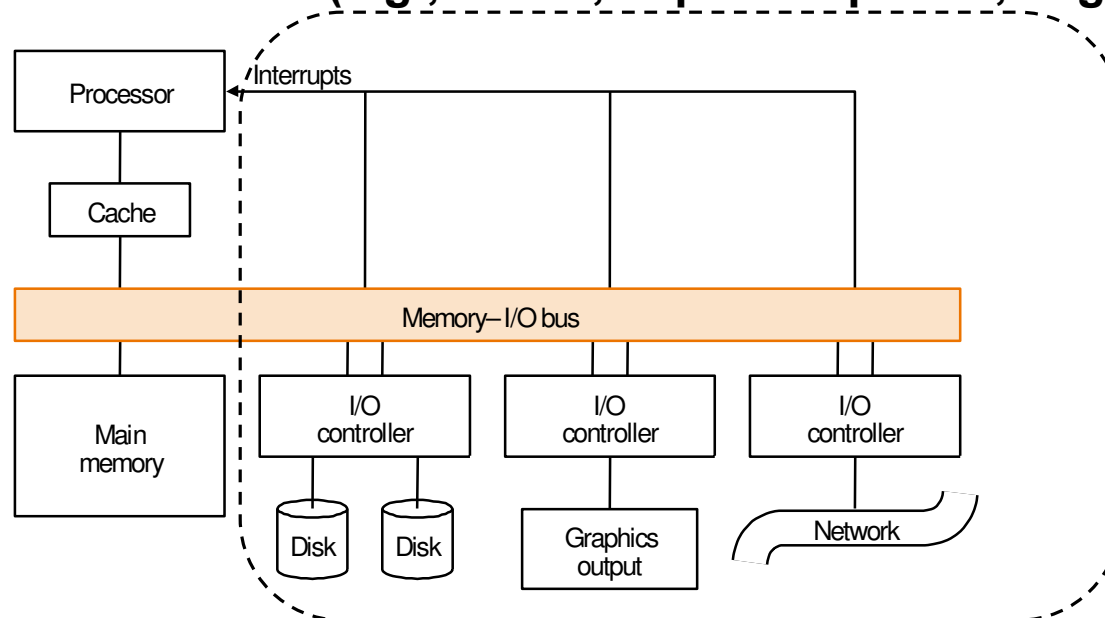
# Chapter 8

# I/O Systems

# Interfacing Processors and Peripherals

---

- I/O Design affected by many factors (expandability, resilience)
- Performance:
  - access latency
  - throughput
  - connection between devices and the system
  - the memory hierarchy
  - the operating system
- A variety of different users (e.g., banks, supercomputers, engineers)



# I/O

---

- **Important but neglected**

**“The difficulties in assessing and designing I/O systems have often relegated I/O to second class status”**

**“courses in every aspect of computing, from programming to computer architecture often ignore I/O or give it scanty coverage”**

**“textbooks leave the subject to near the end, making it easier for students and instructors to skip it!”**

- **GUILTY!**

**— we won’t be looking at I/O in much detail**

**— be sure and read Chapter 8 in its entirety.**

**— you should probably take a networking class!**

# Benchmarks para I/O

---

- **Benchmarks de E/S para Supercomputadores:**
  - acesso a arquivos grandes em batch
  - uma grande operação de leitura seguida de pequenas de escrita (confiabilidade, em caso de crash)
- **Benchmarks para Processamento de Transações:**
  - Deve medir tempo de resposta e throughput
  - Transações são curtas: I/O rate (acessos / s) mais importante do que data rate (bytes / s)
  - I/O rate mais conhecido: TPC-B : simula rede de ATMs
    - medida (throughput para transações dentro de faixa aceitável de tempo de resposta): TPS: transactions per second
- **Benchmarks para Sistemas de Arquivo:**
  - em um ambiente de engenharia: 80% acessos a arq < 10K; 67% de operações de leitura e 27% de escrita (6% R/W)
  - exemplo: benchmark com 70 arquivos, total de 200 KB (mkdir, cp, make, ScanDir, ReadAll)

# I/O Devices

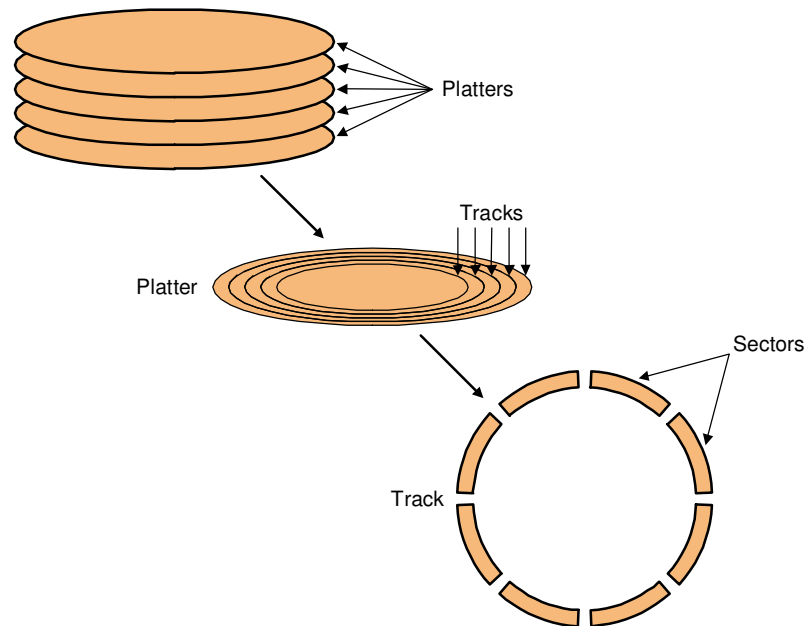
---

- **Very diverse devices**
  - **behavior (i.e., input vs. output)**
  - **partner (who is at the other end?)**
  - **data rate**

<b>Device</b>	<b>Behavior</b>	<b>Partner</b>	<b>Data rate (KB/sec)</b>
Keyboard	input	human	0.01
Mouse	input	human	0.02
Voice input	input	human	0.02
Scanner	input	human	400.00
Voice output	output	human	0.60
Line printer	output	human	1.00
Laser printer	output	human	200.00
Graphics display	output	human	60,000.00
Modem	input or output	machine	2.00-8.00
Network/LAN	input or output	machine	500.00-6000.00
Floppy disk	storage	machine	100.00
Optical disk	storage	machine	1000.00
Magnetic tape	storage	machine	2000.00
Magnetic disk	storage	machine	2000.00-10,000.00

# I/O Example: Disk Drives

---



- **To access data:**
  - **seek:** position head over the proper track (8 to 20 ms. avg.)
  - **rotational latency:** wait for desired sector (.5 / RPM)
  - **transfer:** grab the data (one or more sectors) 2 to 15 MB/sec

# I/O Example: Disk Drives

---

- sector = 512 bytes; 5400 rpm; average seek time = 12 ms  
transfer rate = 5 MB/sec; controller overhead = 2ms
- What is the average time to read or write a sector?

average disk access = average seek  
+ rotational delay  
+ transfer time  
+ controller overhead

$$12 \text{ ms} + 5.6 \text{ ms} + \frac{0.5 \text{ KB}}{5 \text{ MB /sec}} + 2 \text{ ms} = 12 + 5.6 + 0.1 + 2 = 19.7 \text{ ms}$$

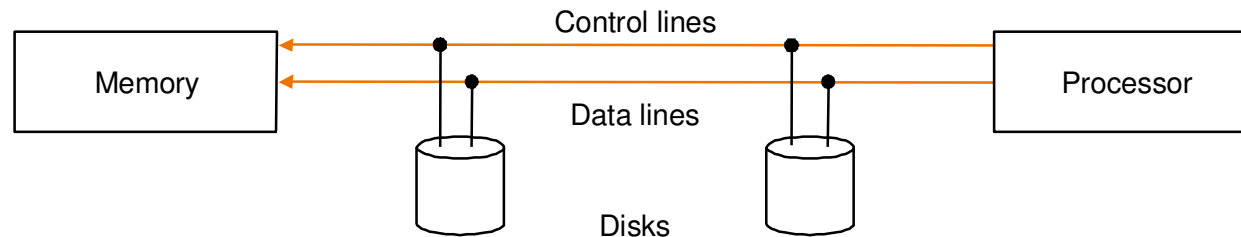
# I/O Example: Buses

---

- **Shared communication link (one or more wires)**
- **Difficult design:**
  - may be bottleneck
  - length of the bus
  - number of devices
  - tradeoffs (buffers for higher bandwidth increases latency)
  - support for many different devices
  - cost
- **Types of buses:**
  - processor-memory (short high speed, custom design)
  - backplane (high speed, often standardized, e.g., PCI)
  - I/O (lengthy, different devices, standardized, e.g., SCSI)
- **Synchronous vs. Asynchronous**
  - use a clock and a synchronous protocol, fast and small but every device must operate at same rate and clock skew requires the bus to be short
  - don't use a clock and instead use handshaking

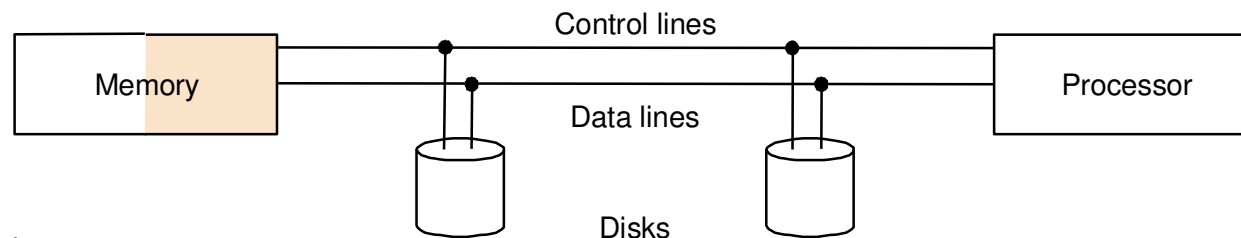


# Output Operation



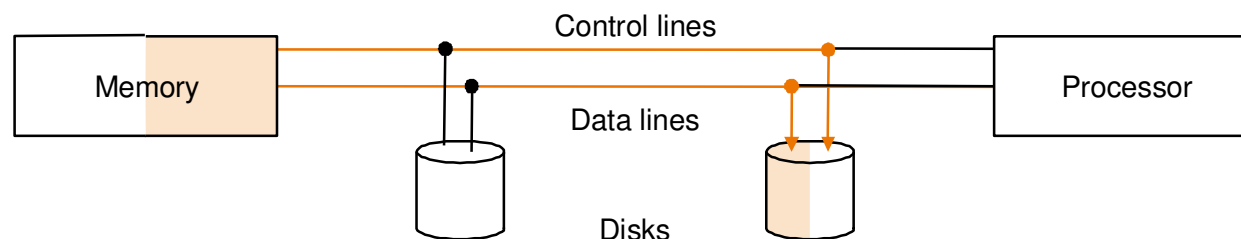
Proc. inicia leitura (RD + endereço)

a.



Memória executa leitura

b.

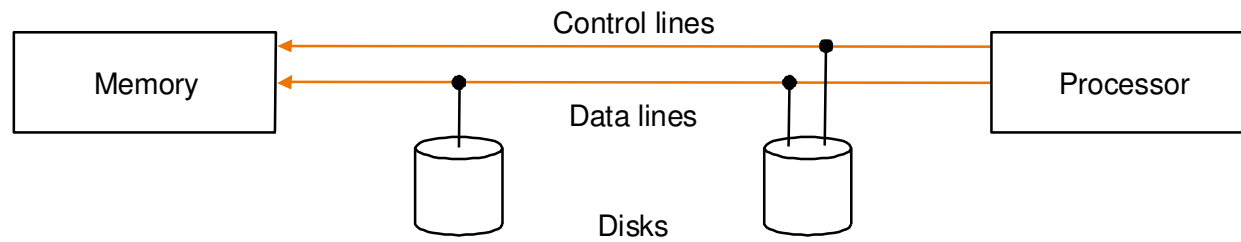


dados da memória para o disco

c.

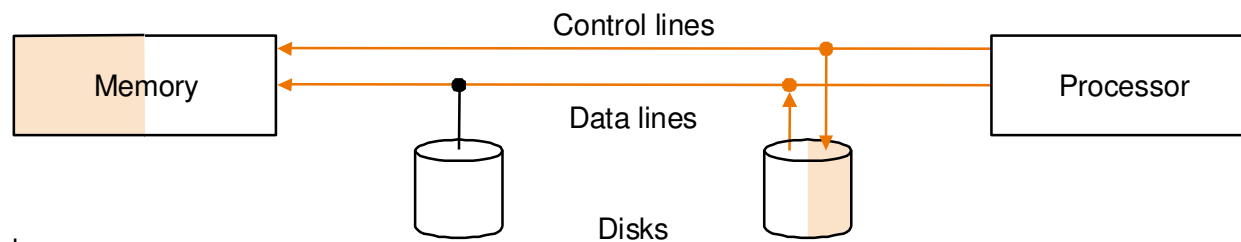
# Input Operation

---



a.

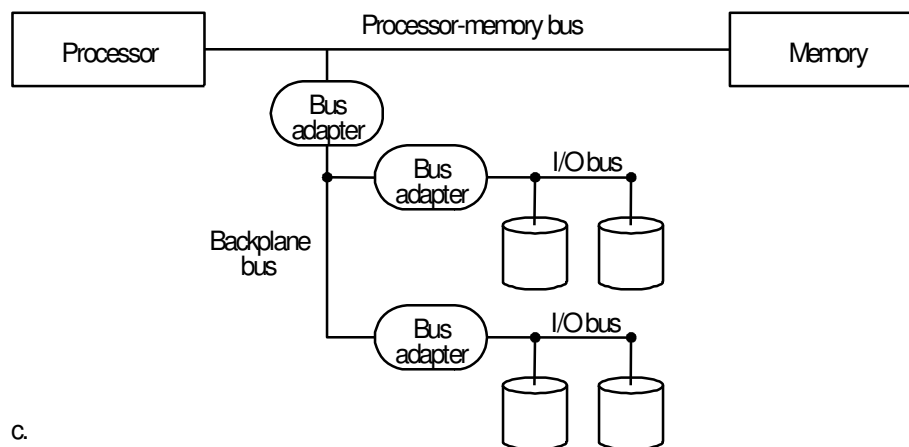
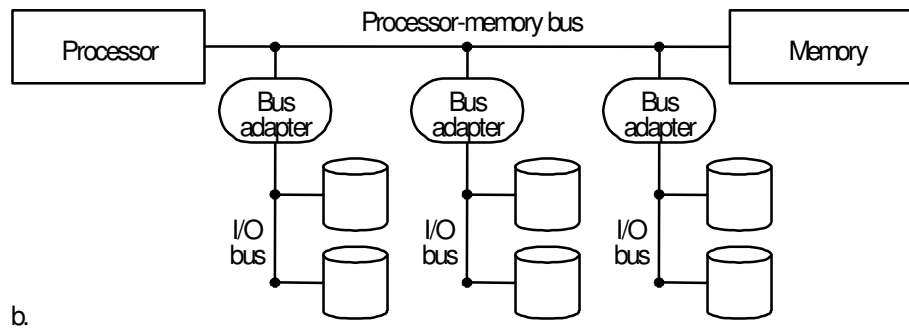
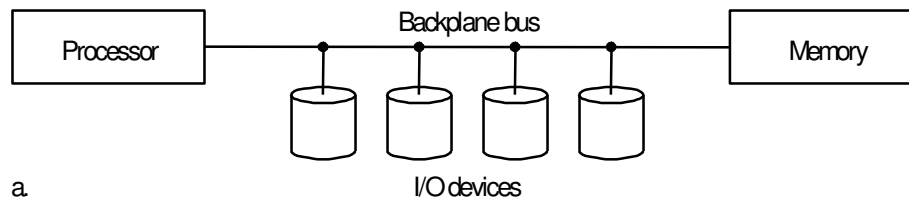
Proc. inicia escrita  
(WR + endereço)



b.

dados do disco  
para a memória

# Barramentos



- aspectos de importância
  - baixo custo
  - flexibilidade
  - gargalos
  - compromisso
    - clock skew

# Alguns tipos de barramentos

---

- **CPU-Memória**
  - geralmente específicos
  - curtos
  - alta velocidade
- **Barramentos de I/O**
  - longo
  - muitos dispositivos
  - não tem interface direta com a memória
  - pode ser padronizado
- **Backplane**
  - objetivo: barramento padrão para permitir a interconexão de vários tipos de dispositivos
  - pode ser padronizado

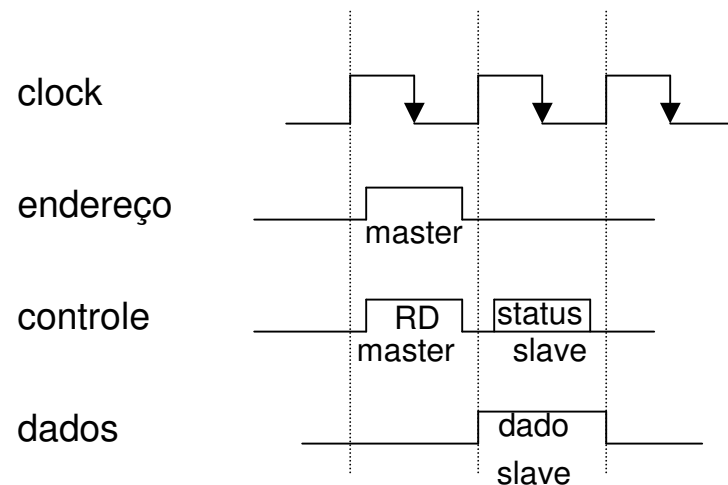
# Comunicação síncrona

---

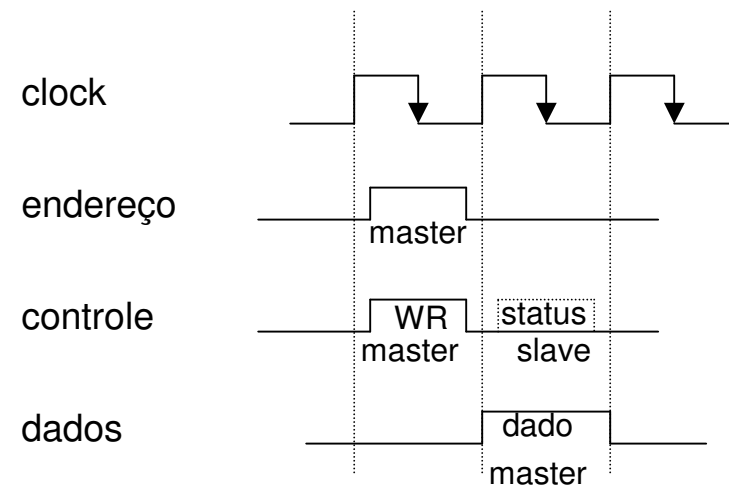
- **Sinais (dados e controle) são sincronizados por um relógio**
- **Controle pode ser feito por uma das unidades (bus master)**
- **Cada item é transferido em time slot predefinido e conhecido pelas partes**
- **Vantagem:**
  - **controle simplificado**
- **Desvantagem:**
  - **pouca flexibilidade**
  - **velocidade limitada pelo dispositivo mais lento**
  - **clock skew**
  - **é normalmente usado na comunicação CPU-MEM**

# Exemplo de comunicação síncrona

Master requisita leitura



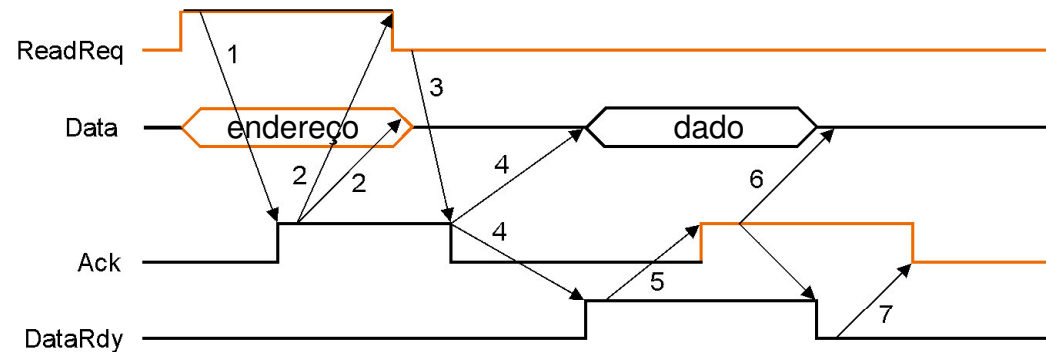
Master requisita escrita



- bordas do clock:
  - delimitam o time slot
  - definem o instante para a amostragem

# Barramento Assíncrono: I/O - Mem

- 1- memória percebe ReadReq, lê endereço no barr. de dados e ativa Ack para indicar que já leu
- 2- I/O percebe Ack ativo e baixa ReadReq e libera o barr. de dados
- 3- memória vê ReadReq baixo e baixa o Ack
- 4- memória coloca o dado lido no barr. de dados e ativa o DataRdy
- 5- I/O percebe DataRdy, lê o dado do barr. de dados e ativa Ack para indicar que já leu
- 6- memória vê Ack ativo, baixa o DataRdy e libera o barr. de dados
- 7- I/O vê DataRdy baixo e baixa o sinal Ack, completando o ciclo



## •Vantagens:

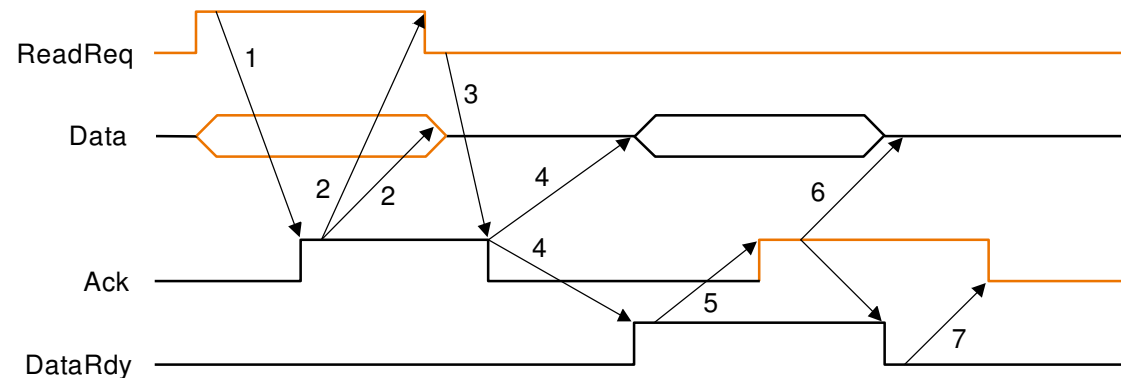
- não tem problemas de clock skew
- flexibilidade
- velocidade

## •Desvantagens:

- complexidade do controle
- Normalmente usado em barr. de I/O

# Some Example Problems

---

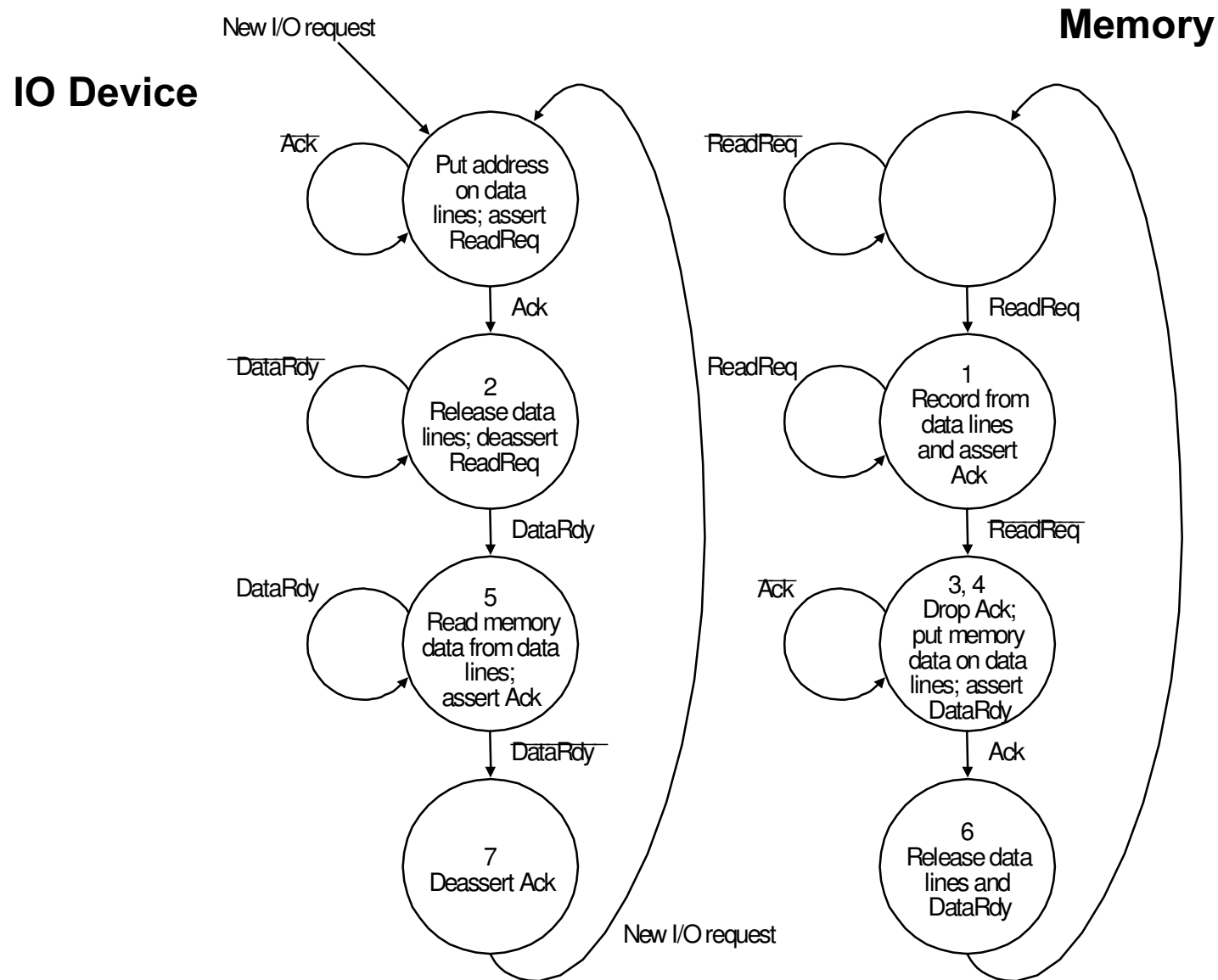


- **asynchronous handshaking protocol - read data in memory and receive it in an I/O device**
- **Let's look at some examples from the text**

**“Performance Analysis of Synchronous vs. Asynchronous”**  
**“Performance Analysis of Two Bus Schemes”**



# FSM - Handshaking protocol



# Arbitragem de Barramentos

---

- **Barramento é compartilhado:**
  - como ganhar acesso?
- **Solução mais simples:**
  - centralizado (bus master)
  - CPU-Mem; Master-Slave
- **Problema: sobrecarga para a CPU**
  - alternativa: vários bus masters
- **Arbitragem: Request - Grant**
- **Objetivo: justiça, sem deadlocks ou starvation**

# Tipos de Arbitragem

---

- **Centralizado:** bus master centralizado recebe múltiplos requests e tem múltiplas linhas de Grant; desvantagens: gargalo e confiabilidade
- **Distribuído:** múltiplas linhas de Request; dispositivos examinam o barramento e julgam a sua própria prioridade (NuBus)
- **Distribuído c/ detecção de colisão (ex: Ethernet; CSMA /CD, carrier sense multiple access / colision detection; meio físico: cabo coaxial, par trançado, fibra óptica, rádio)**

1- envia sinal

4- espera tempo aleatório

2- mede o meio

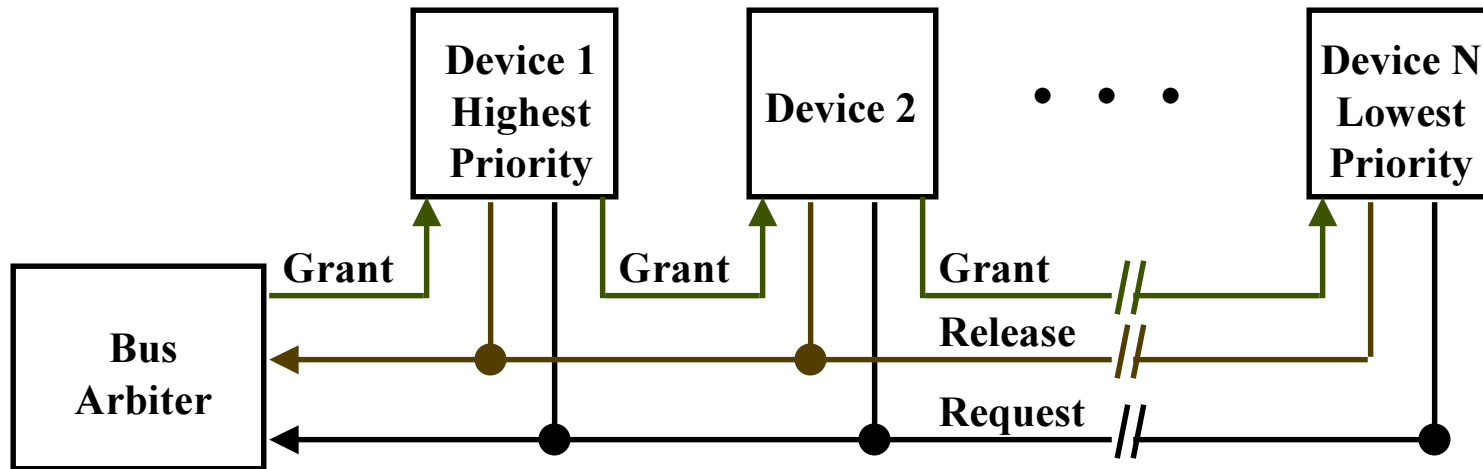
5- volta -> 1

3- se OK -> FIM

6- FIM

# Tipos (cont.): Daisy Chain

---



- **Prioridade:**
  - hardwired pela posição
- **Expansão: simples (online)**
- **Confiabilidade:**
  - ruim

# Comparação entre métodos de arb.

---

- **Centralizado:**
  - **prioridade: programável (off line)**
  - **expansão**
  - **confiabilidade e deadlock: OK se controle OK**
- **Distribuído e ethernet:**
  - **mais flexível, expansível e confiável**

# Escolha de barramentos

---

- **Prioridade, flexibilidade, expansão, confiabilidade, velocidade, tipo de tráfego, distância e “starvation”**

Opção	Alto Desempenho	Baixo Custo
Largura de barr.	endereço e dados separados	endereço e dados multiplexados
Largura de dados	$\geq 32$ bits	8 bits
Bloco de transf.	múltiplas palavras	uma palavra
Bus master	múltiplos masters	um master
Clocking	síncrono	assíncrono

## Dois padrões de barramento (pag 673)

---

Characteristic	PCI	SCSI
Bus type	backplane	I/O
Bus width	32-64	8-32
Addr/data mux'ed	muxed	muxed
Number of bus masters	multiple	multiple
Arbitration	centralized, parallel arbitration	self-selection
Clocking	Synchr. 33-66 MHz	Synchr. 5-10 MHz or Asynchr.
Peak Bandwidth	133-512 MB/s	5-40 MB/s
Estimated Bandwidth	80 MB/s	2.5-40 MB/s (sync) 1.5 MB/s (async)
Max. # devices	1024 (32 / bus segment)	7-31 (bus width - 1)
Max. bus length	0,5 meter	2,5 meter
Standard	PCI	ANSI X3.131

# Interface E/S - Mem - CPU - OS

---

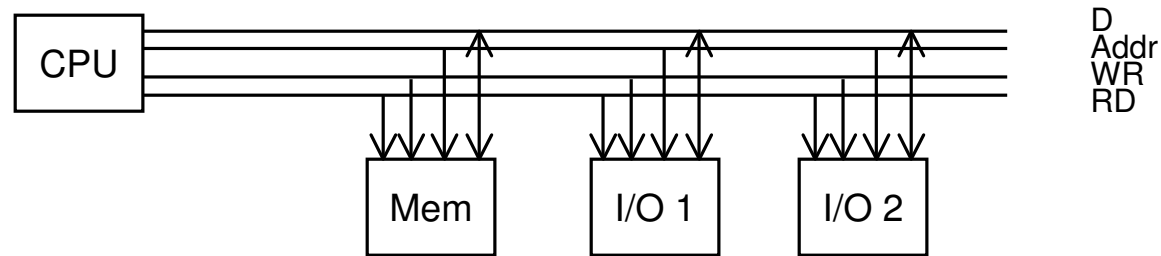
- **Questões:**
  - como o usuário controla a E/S?
  - dados <-> memória ?
  - qual é o papel da CPU?
- **Papel da CPU é grande:**
  - multiprogramação
  - dispositivos de E/S normalmente usam interrupção executada no modo supervisor do OS
  - controle de baixo nível de E/S é complexo



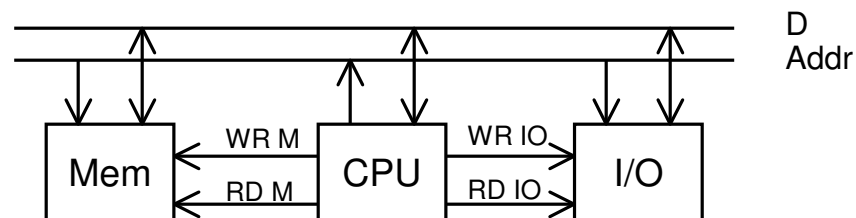
# Comunicação CPU $\Rightarrow$ E/S

---

- **Programmed I/O: CPU inicia, controla, supervisiona e termina a comunicação**
  - **Memory mapped: acesso à memória e I/O é feito com as mesmas instruções (espaço de endereçamento define o dispositivo)**



- **I/O mapped: há instruções e linhas de controle dedicadas à comunicação de I/O**



# Comunicação E/S $\Rightarrow$ CPU

---

- **Alternativa 1: polling**
  - a CPU periodicamente consulta os dispositivos
  - **desvantagens:**
    - desperdício do tempo da CPU
    - dispositivo pode ter que ficar muito tempo aguardando o serviço
- **Alternativa 2: interrupção**

# Comunicação E/S $\Rightarrow$ CPU (cont)

---

- **Interrupção:**
  - **dispositivo de E/S avisa a CPU que precisa de atenção, por meio de sinal assíncrono**
  - **a CPU deve completar a execução da instrução antes de atender a interrupção:**
    - (pipeline: é necessário stall e tratamento de todas as instruções no pipeline)
  - **fornece indicação à CPU sobre o tipo de interrupção e a identidade do dispositivo**
    - solução comum: vetor de interrupção usado para decidir o endereço da rotina de tratamento da interrupção
  - **em caso de múltiplos dispositivos:**
    - priorização para decidir qual pedido será atendido
      - (é possível o uso de um controlador de interrupção)

# Exemplo (pag 677)

---

- **Impacto do polling para 3 dispositivos (fck = 500 MHz; ciclos p/polling = 400)**

## Mouse

taxa de polling = 30 / s

30 polling/s = 30 \* 400  
ciclos/s = 12000 c/s

% ciclos da CPU =  
 $12000 / 500 * 10^6 =$   
0,002 %

ACEITÁVEL

## Floppy Disk

blocos de 16 bits;  
taxa 50 KB/s

pollings / s =  
 $50 \text{ KB / s} / (2 \text{ B / polling}) =$   
25 k pollings / s

400 ciclos /polling ->  
 $25 \text{ k} * 400 =$   
 $100 * 10^5 \text{ ciclos de polling / s}$

% de ciclos da CPU =  
 $10^7 / 500 * 10^6 = 2\%$

ALTO MAS +- OK para  
sistemas de baixo custo

## Disco Rígido

blocos de  $4 * 32$  bits; 4 MB / s

pollings / s =  
 $4 \text{ MB / s} / (16 \text{ B / polling}) =$   
 $0.25 * 10^6 \text{ pollings / s}$

400 ciclos /polling ->  
 $100 * 10^6 \text{ ciclos de polling / s}$

% de ciclos da CPU =  
 $100 * 10^6 / 500 * 10^6 = 20\%$

Tempo inaceitável de  
ocupação da CPU para  
polling

## Exemplo (cont)

---

- **Impacto da interrupção sobre o Disco Rígido do exemplo anterior (com overhead da transferência de 500 ciclos, incluindo a interrupção, assumindo que o HD está ativo apenas 5% do tempo):**

Taxa de acessos é o mesmo de polling = 250 K / s

# ciclos / s = 250 k \* 500 = 125 \* 10<sup>6</sup> ciclos / s

% da CPU = 125 \* 10<sup>6</sup> / 500 \* 10<sup>6</sup> = 25%

perto do Polling, MAS, levando em consideração que o HD não está ativo apenas 5% do tempo:

impacto 5% \* 25% = 1,25 %           (bastante razoável)

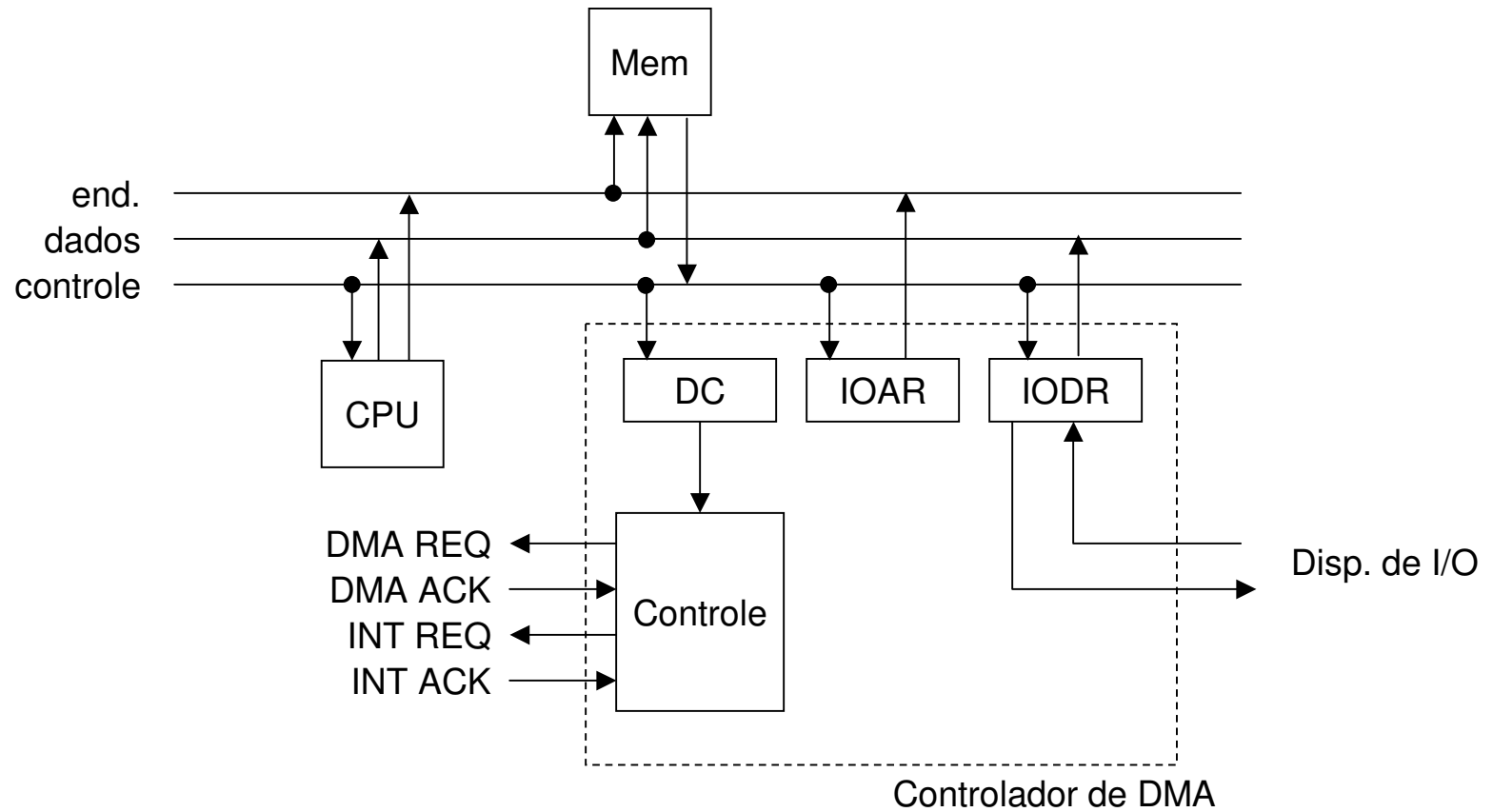
Interrupção só envolve a CPU se houver atividade

# DMA: dispositivo $\Leftrightarrow$ memória

---

- **Usando polling ou interrupção a CPU é envolvida diretamente na transferência**
  - **interrupção pode ser OK para pequenos blocos; CPU se ocupa de outra coisa enquanto o dispositivo está trabalhando mas a escrita/leitura envolve diretamente a CPU**
- **Solução: DMA (Direct Memory Access)**
  - **não há polling e a CPU não participa da transferência**
  - **Passos:**
    - 1- CPU informa: identidade do dispositivo, qual é a operação, endereço inicial da memória, N<sup>o</sup> de bytes da transferência
    - 2- Controlador de DMA inicia a operação. Solicita e ganha controle do barramento. Faz a transferência entre o dispositivo (endereço, RD/WR, incrementa endereço, etc) e a memória.
    - 3- Controlador de DMA interrompe a CPU e avisa que a transferência está completa

# Típico Controlador de DMA



# Tipos de Operação de DMA

---

- **Block Transfer:** seqüência de palavras; usado por discos, fitas (vel. do disp. é importante); a CPU não usa o barramento :
  - CPU executa instrução de IO; escreve em IOAR e DC
  - Controlador faz DMA REQ; CPU -> DMA ACK
  - Controlador faz transferência com Inc IOAR; Decr DC
  - Se o dispositivo for lento, retorna o controle para a CPU com DMA REQ = 0; estando pronto DMA REQ = 1
  - Fim se DC =0
- **Cycle Stealing:** uma (ou mais palavras transmitidas) e o controle devolvido à CPU
- **DMA transparente:** cycle stealing sincronizado de modo a usar barramento somente nos ciclos em que a CPU não usa
  - contenção de memória deixa de ser problema sério com o uso de cache



# DMA, Cache e Memória Virtual

---

- **Perguntas:**
  - **DMA deve trabalhar com endereço físico ou virtual?**
  - **como transferir se os dados não estão em uma única página?**
- **Soluções:**
  - **se os dados estão em uma única página, trabalhar com endereço físico**
  - **caso contrário, trabalhar com o endereço virtual; o próprio controlador deve fazer o mapeamento (pequena page table interna)**
- **Outros problemas:**
  - **O sistema operacional não deve mover páginas afetadas enquanto o DMA está em operação**
  - **pode acontecer incoerência de cache-memória devido a atualização da memória pelo DMA**

# Exemplo; impacto em Hard Disk

---

- **Mesmo exemplo anterior:**
  - **bloco 16B; taxa de transferência de 4 MB / s; clock de 500 MHz**
  - **setup do DMA: overhead de 1000 ciclos**
  - **interrupção do DMA 500 ciclos**
  - **tamanho médio de transferência: 8 KB**
  - **atividade do HD: 100% do tempo (ignorar contenção de barramento)**

tempo por transferência:  $8 \text{ KB} / 4 \text{ MB/s} = 2 \text{ ms}$

Para atividade constante do HD:

$$\frac{(1000 + 500) \text{ ciclos / transf.}}{2 \text{ ms / trans.}} = 750 * 10^3 \text{ ciclos / s}$$

$$(750 * 10^3 \text{ ciclos/s}) / (500 * 10^6 \text{ ciclos/s}) = 1,5 * 10^{-3} = 0,15\%$$

CPU é só envolvida no início e no final

como atividade < 100 % este percentual é ainda menor

# Exemplo: I/O System Design (1)

---

- Dados:
  - CPU com 300 MIPS e 50K instruções (no OS) por operação de I/O
  - Memory backplane: taxa de transmissão de 100 MB/s
  - Controladores SCSI-2: 20 MB/s e capacidade para 7 discos
  - HD com taxa de 5 MB/s e atraso seek+rotational = 10 ms
  - workload de IO: leituras de 64 KB (100 K instruções de usuário / Operação de I/O)
- Encontrar: máxima taxa de I/O e o número de (HD+controladores) necessário
- Dois componentes (CPU e memory bus), qual é o gargalo?
  - cada op de I/O precisa de 50 K + 100K instruções

$$\text{IO rate da CPU} = (\text{Instr rate}) / (\text{Instr.} / \text{IO}) = 300 \cdot 10^6 / (100+50) \cdot 10^3 = 2000 \text{ IO/s}$$

$$\text{IO rate do bus} = (\text{bus bandwidth}) / (\text{bytes} / \text{IO}) = 100 \cdot 10^6 / 64 \cdot 10^3 = 1562 \text{ IO/s}$$

gargalo é o barramento

agora, configurar o resto do sistema para acomodar 1562 IO/s -> quantos HDs?

## Exemplo: I/O System Design (2)

---

- Meta: 1562 IO/s
- Tempo por IO no disco:  $10 \text{ ms} + 64 \text{ KB} / (5 \text{ MB/s}) = 10\text{ms} + 12.8\text{ms} = 22.8\text{ms}$   
cada disco:  $1/22.8\text{ms} = 43.9 \text{ IO/s}$   
para ocupar totalmente o barramento  
 $1562 / 43.9 = 36 \text{ discos}$
- Número de controladores SCSI  
Taxa de transf. / disco =  $64 \text{ KB} / 22.8\text{ms} = 2.74 \text{ MB /s}$   
possível colocar 7 discos por controlador ( $7*2.74 < 20 \text{ MB/s}$ )  
Número de barramentos de IO e controladores SCSI =  
 $36 \text{ discos} / (7 \text{ discos} / \text{barramento}) = 6 \text{ (barramentos + controladores)}$
- Grande número de hipóteses simplificadoras: melhor precisão somente com simulação