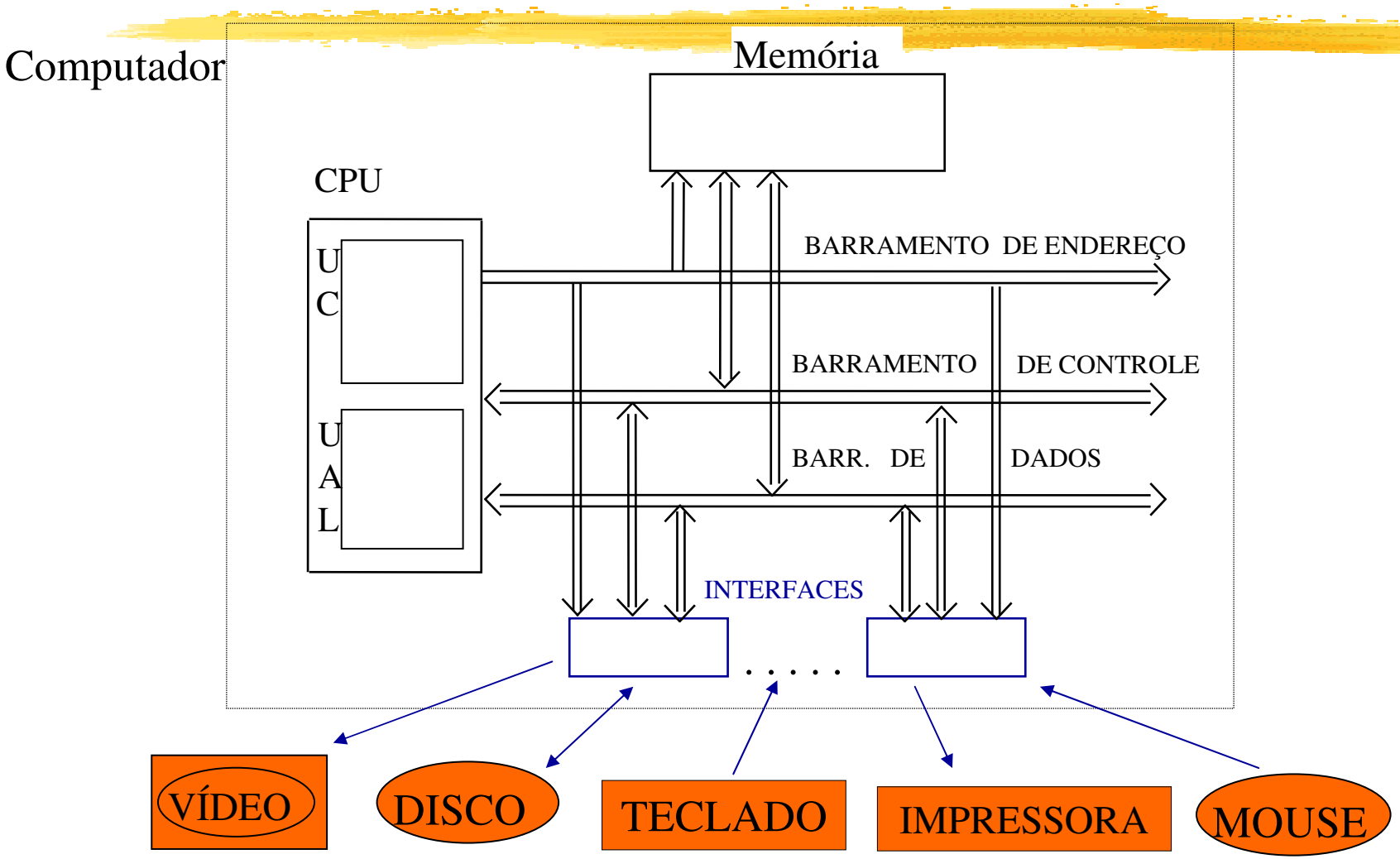
8. Entrada e Saída

(capítulo 7 do livro texto)



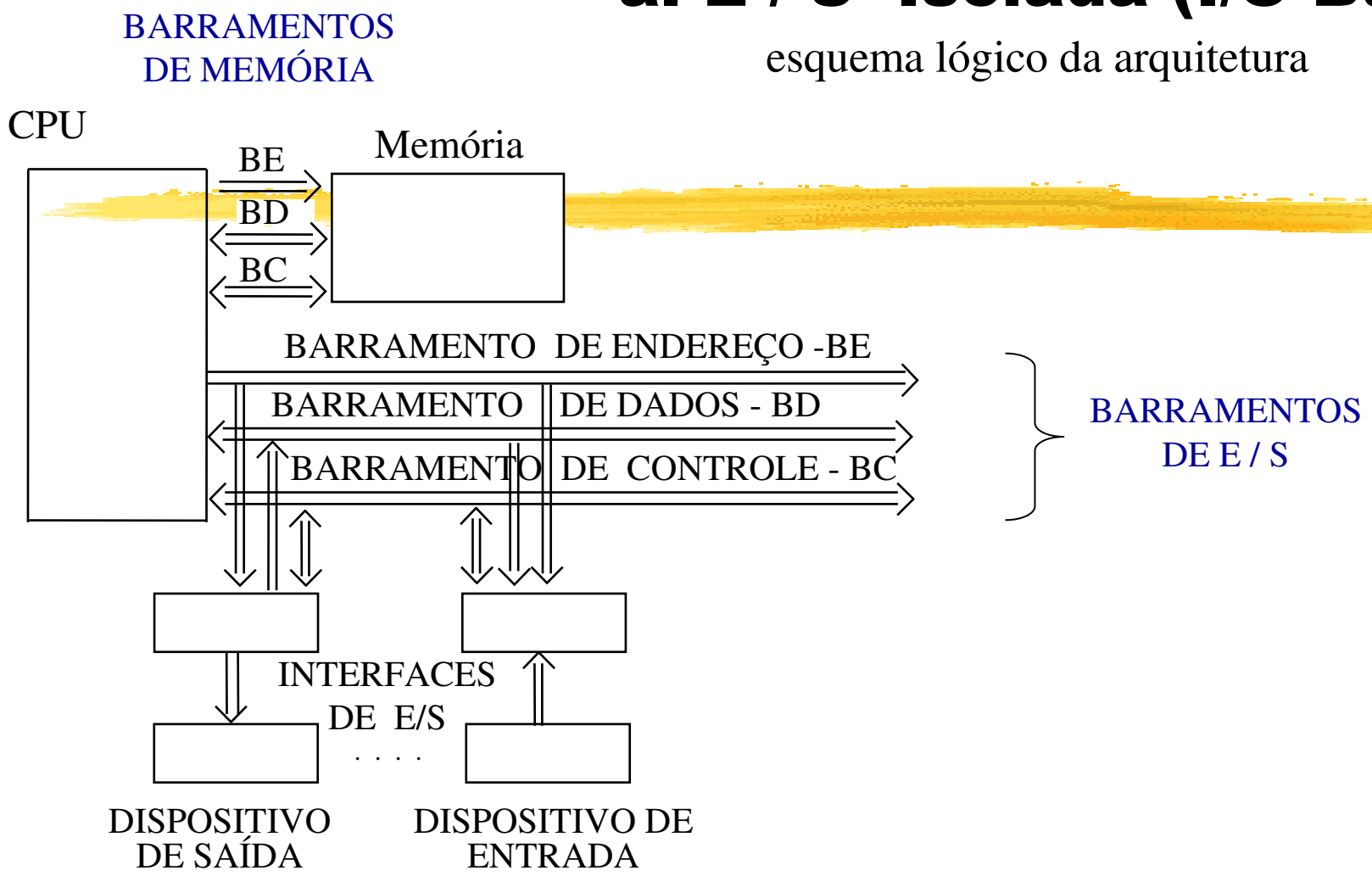
8.1 Arquiteturas de E / S

a. E / S isolada (I/O Bus)

b. E / S mapeada em memória

a. E / S Isolada (I/O Bus)

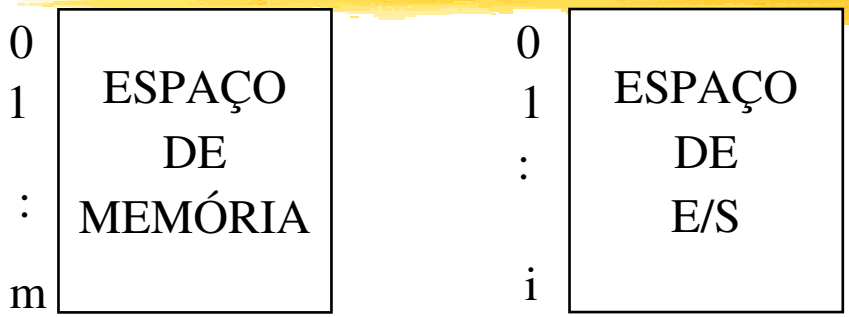
esquema lógico da arquitetura



Neste tipo de arquitetura, os barramentos de controle, endereços e dados para E/S e para acesso à memória são logicamente distintos, de forma que o **acesso** da CPU à Unidade de Memória é **isolado** do seu acesso às portas de E/S.

E / S Isolada (I/O Bus) – cont.

espaço de endereços



o uso de endereços idênticos obriga que arquiteturas do tipo E/S Isolada incluam **instruções específicas** para Entrada de Dados, para Saída de Dados e para Acesso à Memória.

- E/S: IN, OUT
- Memória: LOAD, STORE, MOVE

Exemplos:

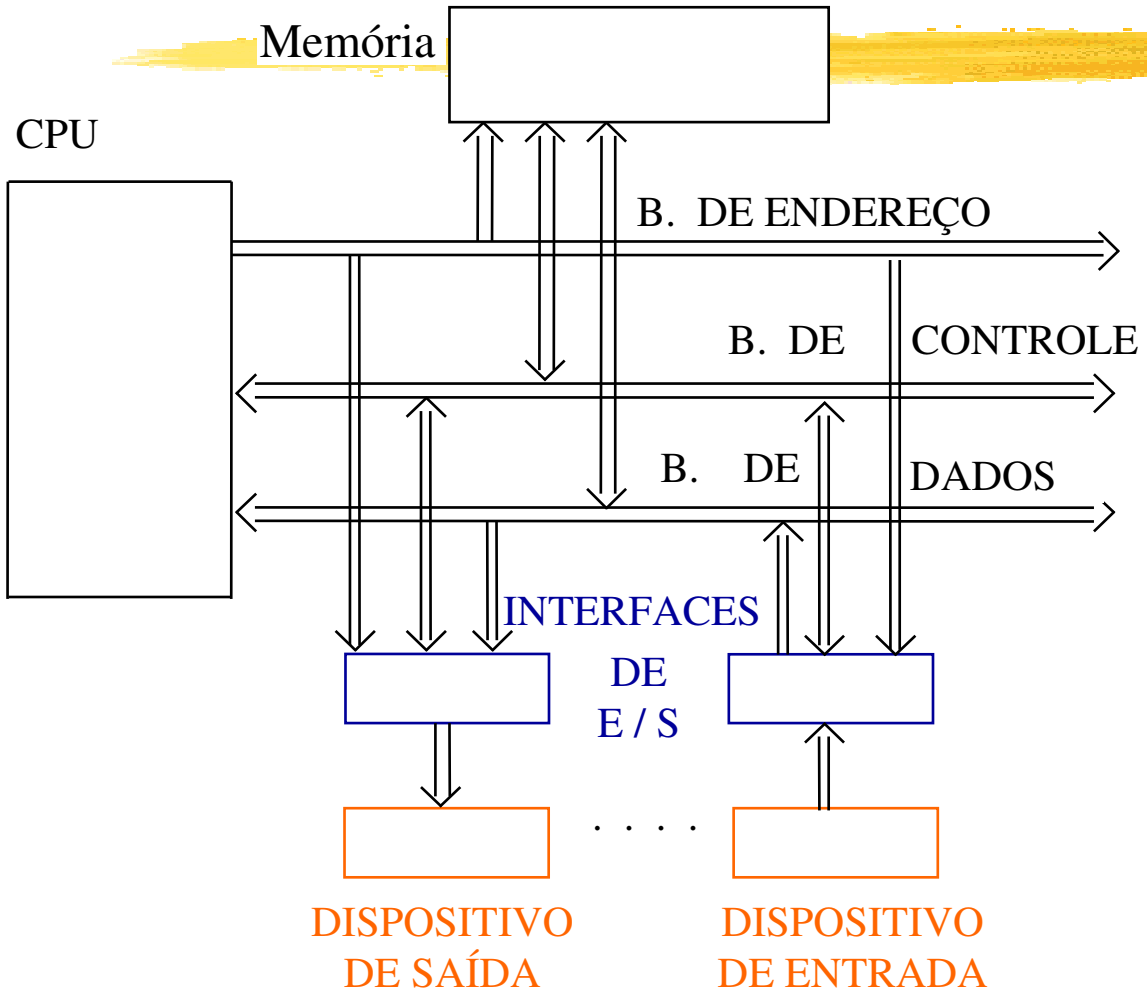
1. transferência de dados entre o acumulador e uma porta de E/S :

IN	port	- entrada de dados para reg. Acc
OUT	port	- saída de dados do reg. Acc

2. transferência de dados entre o acumulador e a posição m da memória:

LOAD	m	- leitura da memória para o reg. Acc
STORE	m	- escrita na memória a partir do reg. Acc

b. E / S Mapeada em memória (UNIBUS)



Neste tipo de arquitetura não há barramentos dedicados para E/S, como também não existem instruções especiais de E/S .

E / S Mapeada em memória (UNIBUS) – cont.

espaço de endereços

Leitura:

MOVE K,R1 ; R1 ← (K)

$K < M \Rightarrow$ leitura da memória

$M \leq K \leq N \Rightarrow$ leitura de disp. de entrada

Escrita:

MOVE R1,K ; K ← (R1)

$K < M \Rightarrow$ escrita na memória

$M \leq K \leq N \Rightarrow$ escrita em disp. de saída

Vantagens x Desvantagens

isolada e mapeada

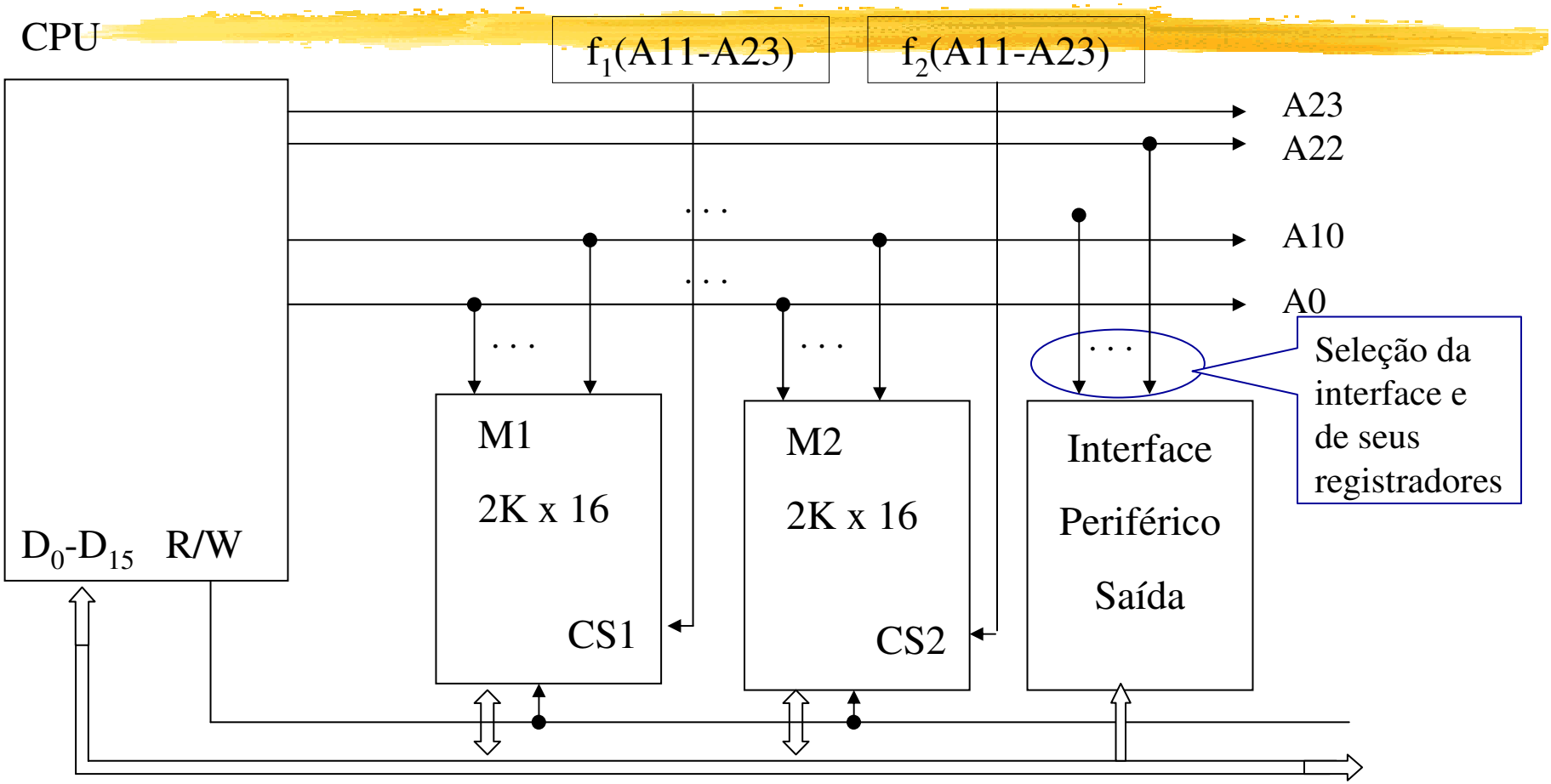
A solução mapeada apresenta as vantagens:

- não são necessários códigos específicos para instruções de E/S;
- todas as instruções que fazem referência à memória também podem ser usadas para manipulação de E/S;
- a quantidade de endereços para E/S é praticamente ilimitada.

Algumas desvantagens:

- parte dos endereços de memória não pode ser utilizada como endereços de memória;
- há necessidade de interfaces mais complexas para reconhecer endereços maiores;
- instruções de referência a endereços de memória (que na verdade realizam a E/S) podem ser mais longas, com maior tempo de execução, do que instruções específicas de E/S.

8.2 Seleção de interfaces e memórias



Seleciona memória com Chip-Select + endereço e Periférico com endereço

Seleção de interfaces e memórias
Exemplo – decodificação de endereço

Seja um processador com a seguinte configuração:

- 10K palavras (blocos de 2K e 8K) de ROM
- 2K palavras de RAM
- 4 palavras para os dois periféricos (2+2)

Assim:

- ROM1 (2K), RAM (2K), ROM2 (8K), PER1 (2), PER2 (2)
- dispositivos têm Chip-Select ativo em 0 (low)
- ROM1 deve estar entre $00.0000 e $00.00FF (hexa)

Seleção de interfaces e memórias
Exemplo – decodificação de endereço (cont)

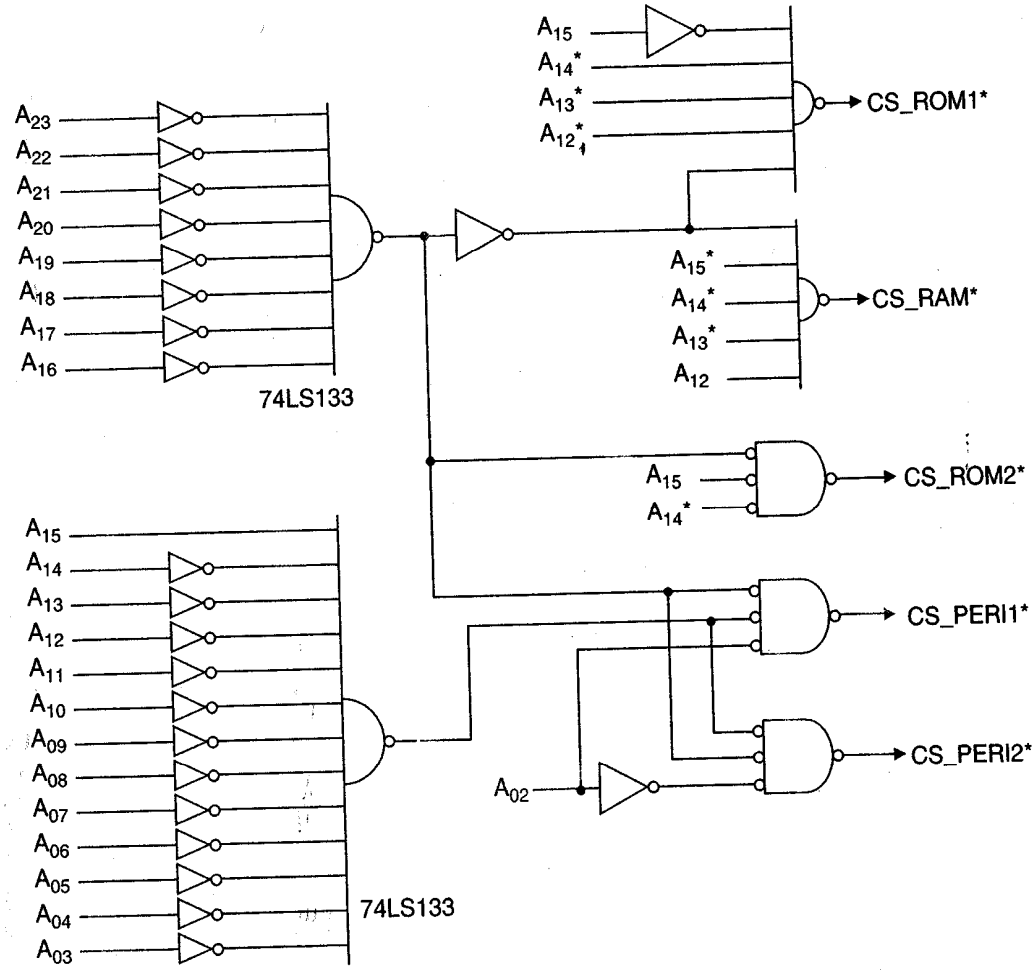
Definiremos o seguinte endereçamento:

2K

		A 23	...	A 15	A 14	A 13	A 12	A 11	A 10	A 9	A 8	A 7	A 6	A 5	A 4	A 3	A 2	A 1	A 0	endereços
2K	ROM 1	0	..	0	0	0	0	X	X	X	X	X	X	X	X	X	X	X	-	000000 0007FF
2K	RAM	0	..	0	0	0	1	X	X	X	X	X	X	X	X	X	X	X	-	000800 000FFF
8K	ROM 2	0	..	0	1	X	X	X	X	X	X	X	X	X	X	X	X	X	-	002000 003FFF
2	PER 1	0	..	1	0	0	0	0	0	0	0	0	0	0	0	0	0	X	-	004000 004001
2	PER 2	0	..	1	0	0	0	0	0	0	0	0	0	0	0	0	1	X	-	004002 004003

Implementação do decodificador de endereço para as memórias (Microprocessor Systems Design, A. Clements, 3. ed., págs. 310-312)

Figure 5.5
Full address decoding network corresponding to Table 5.1



8.3 Interfaces de E / S

Função de uma interface de E/S:
coordenar a transferência de dados entre o processador e um dispositivo externo, comunicando-se com o processador através do barramento e com o dispositivo através de ligações apropriadas.

Assim, uma interface de E/S deve:

- armazenar o estado do dispositivo, para informar à CPU o seu estado quando solicitado, no **Registrador de Estado (RS)** ;
- prover uma área de armazenamento, no caso o **Registrador de Dados (RD)**, para ser usada na transferência de dados;
- reconhecer o endereço dos seus registradores quando este aparecer no barramento de endereços;
- prover sinais apropriados de temporização para possibilitar a transferência de dados quando requeridos; e
- executar a conversão de formatos de dados necessária para a transmissão de dados entre o barramento de dados e os dispositivos periféricos.

Além disso deve prover um mecanismo de definição de seu modo de operação, auxiliado por um **Registrador de Controle (RC)**.

Interfaces de E / S

Registradores

- **Registrador de Estado (RS)**
- **Registrador de Dados (RD)**
- **Registrador de Controle (RC)**

Registrador de Dados - RD

O *Registrador de Dados* é utilizado como um armazenador temporário de dados. Em uma operação de saída, ele é carregado com o dado pela CPU, e lido pelo dispositivo.

Em uma operação de entrada, ele é carregado pelo dispositivo, e lido pela CPU.

Algumas interfaces possuem mais de um *Registrador de Dados*, um buffer para dados, e deste modo, é possível adequar as velocidades da CPU e do dispositivo externo.

Registrador de Estado - RS

O *Registrador de Estado* RS informa o estado da transferência de E/S e cada bit deste registrador tem um significado especial.

Diversas informações relevantes estão associadas aos bits do *Registrador de Estado* tais como: o dispositivo gerou uma interrupção, o dispositivo está desconectado, etc.

Dois bits do *Registrador de Estado, em especial,* são muito importantes para controlar o processo de E/S:

- um deles indica se o dispositivo colocou um dado de entrada no *Registrador de Dados*;
- o outro indica se o dispositivo está pronto para processar um dado de saída, ou seja, se o *Registrador de Dados* está pronto a receber outro dado para saída.

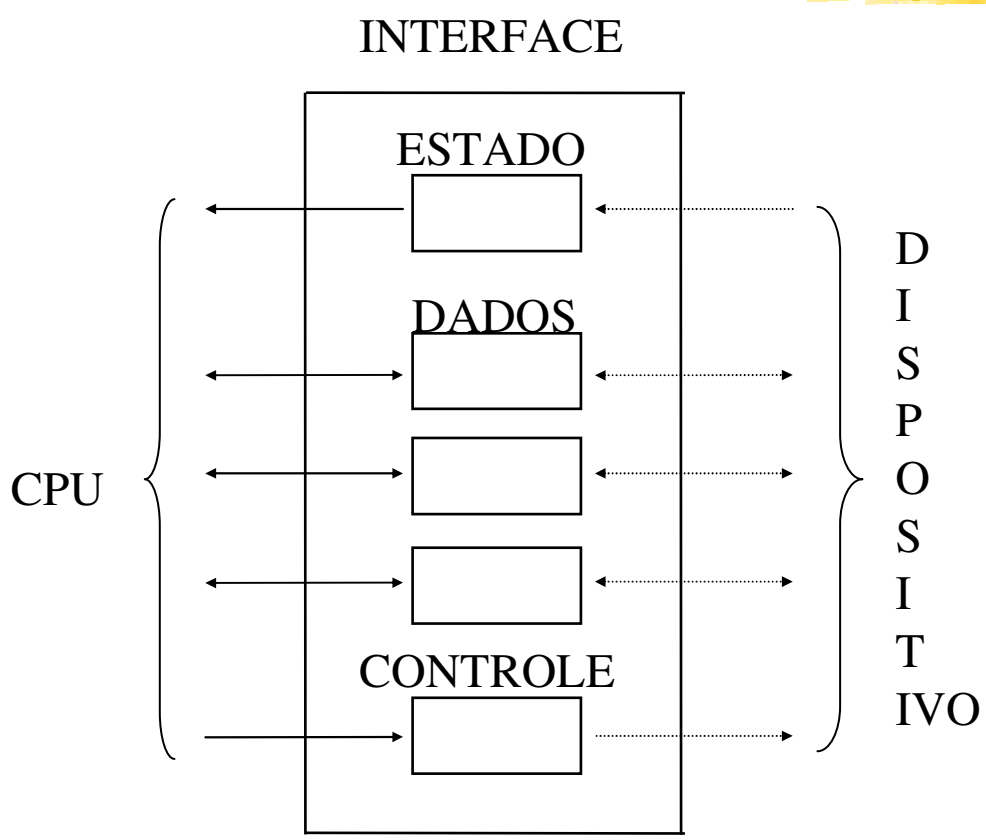
Registrador de Controle - RC

As interfaces possuem ainda um registrador especial - *Registrador de Controle* - dedicado ao estabelecimento de *modos de operação*.

Os *modos de operação* especificam características de operação da interface como *leitura* ou *escrita*, *permissão para interrupção*, etc.

Os *modos de operação* podem ser programados na interface utilizando-se instruções de escrita (saída) que endereçam o registrador de controle.

Fluxo de Informação nos registradores de interfaces

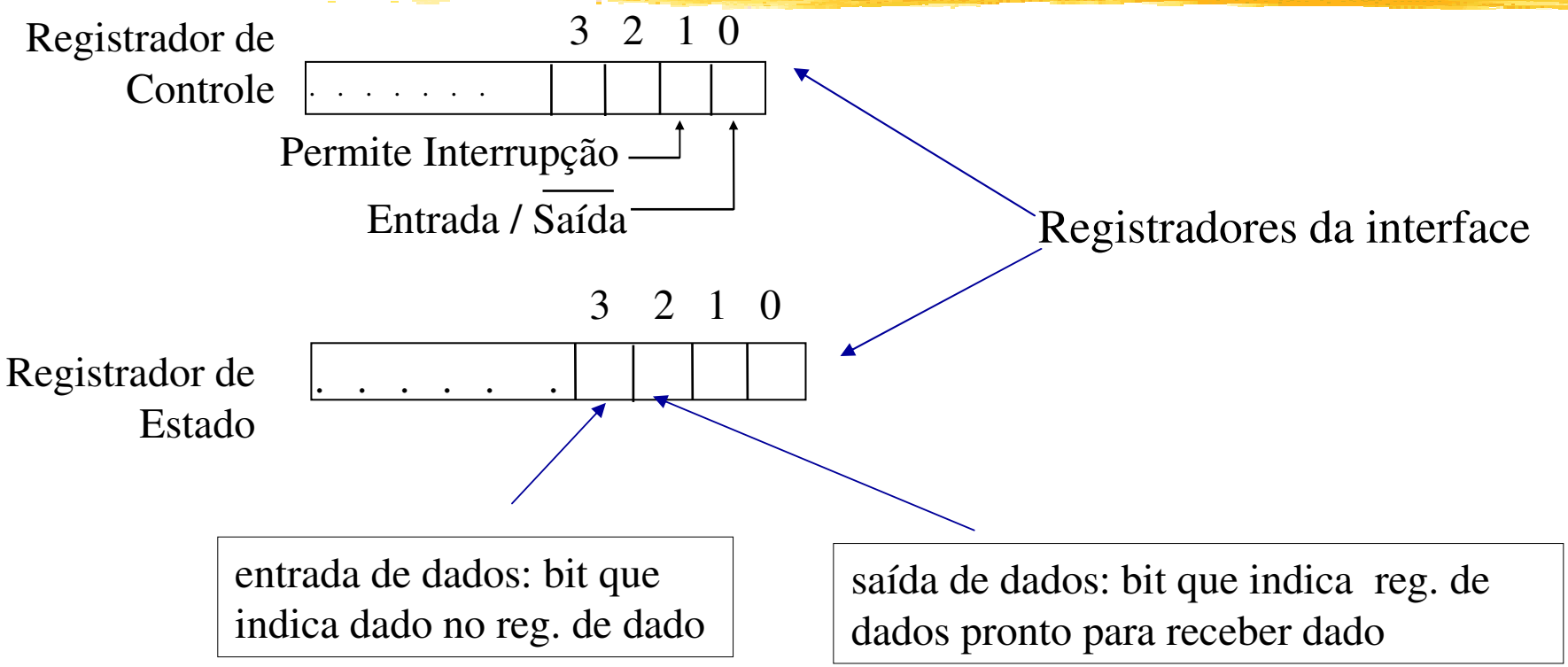


Do ponto de vista da CPU, no *Registrador de Estado* é permitida somente a leitura e o *Registrador de Controle* é apenas de escrita.
Em algumas interfaces a leitura do Registrador de Controle é também permitida.

O *Registrador de Dados* possui fluxo de dados bidirecional, permitindo tanto a leitura como a escrita.

Interfaces podem conter mais de um *Registrador de Estado*, *de Dados* ou *de Controle*. O esquema mostra a interação da CPU com a interface, realizada através de instruções da CPU.

Exemplo: entrada / saída de dados



Exemplo: entrada / saída de dados – cont.

Saída (com arquitetura E/S isolada)

```
        LDA     #0      ; sem interrupção e modo saída
        OUT     RC      ; escreve (programa) na (a) interface
LOOP:   IN      RS      ; leitura do estado
        TSTBIT 2        ; testa bit 2 (estado de saída)
        JNZ     LOOP    ; se bit=0 continua, se não LOOP
        LDA     dado    ; carrega dado no reg. Acumulador
        OUT     RD      ; carrega dado em RD para a
                        ;  interface realizar a saída do dado
```

O bit de RS só é zerado após a saída completar-se.

Exemplo: entrada / saída de dados – cont.

Entrada (com arquitetura E/S mapeada)

```
        MOVE   #01,RC  ; sem interrupção e modo entrada (modo de op.)
LOOP:   MOVE   RS, Ra  ; leitura do estado (no endereço RS)
        TSTBIT 3,Ra    ; testa bit 3 de Ra (estado de entrada)
        JZ     LOOP    ; se bit=0 não há dado ainda
        MOVE   RD, Rb  ; carrega  RD no registrador da CPU.
```

A execução de " MOVE RD, Rb" faz zerar a posição de RS que indica a presença de dado gerado pelo dispositivo.

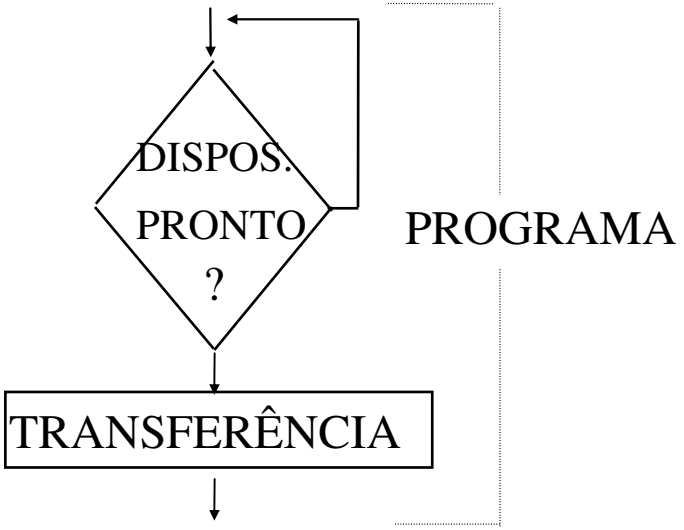
8.4 Controle de transferência de dados

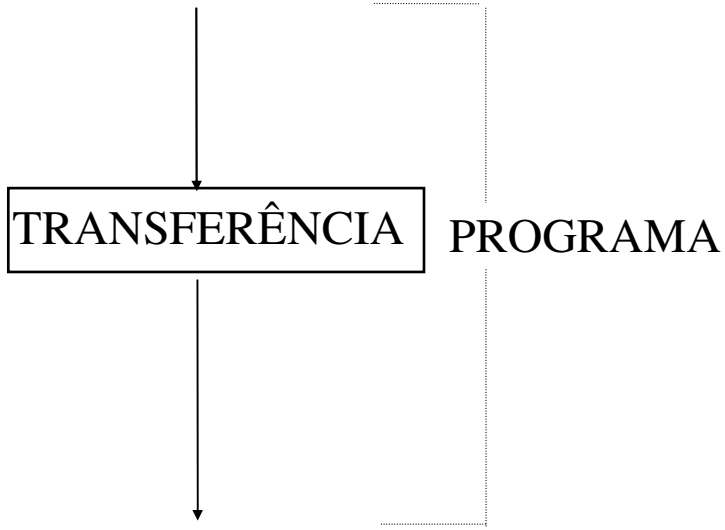
Existem três formas básicas de controlar a transferência de dados:

- E/S controlada por programa (controlada pela CPU);
- E/S controlada por interrupção (controlada pela CPU) e
- E/S controlada por hardware específico - DMA ("Direct Memory Access") ou Acesso Direto à Memória.

Controle de transferência de dados

E/S controlada por programa



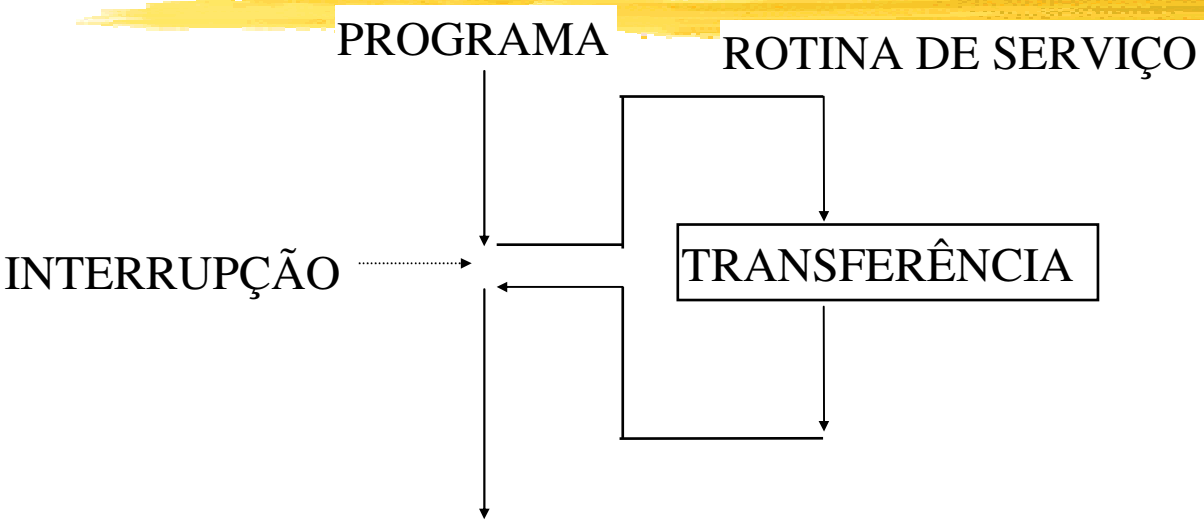


A transferência só é efetuada quando o dispositivo está pronto para a mesma. Utilizada em transferências do tipo *Requisição* ("Request"), onde o dispositivo é requisitado a executar a transferência.

O programa efetua a transferência independentemente do estado do dispositivo. É utilizada em transferência do tipo *Amostragem* ("Sample"), para ler/escrever o valor *corrente*.

Controle de transferência de dados

E/S controlada por interrupção



A E/S controlada por interrupção é utilizada em aplicações onde o tipo de transferência é por Evento ("Event"), o que implica a não existência de um ponto determinado do programa onde é feita a E/S de dados.

A interface gera um sinal de interrupção quando está pronta para a transferência. O programa em execução é interrompido e a rotina de serviço de interrupção executa a transferência desejada.

A transferência é executada *assincronamente* com relação ao programa.

Controle de transferência de dados

E/S controlada por programa
E/S controlada por interrupção

Tanto na **E/S controlada por programa**, como na **E/S controlada por interrupção**, a transferência é controlada pela CPU:

O dado é transferido entre a interface e registradores da CPU pelas instruções de E/S :

```
IN  port        ; Acc ← (port), para entrada
OUT port        ; port ← (Acc), para saída
```

quando a arquitetura é E/S Isolada;

ou por instruções comuns como :

```
MOVE  port,R1   ; R1 ← (port), para entrada
MOVE  R1,port   ; port ← (R1), para saída
```

quando a arquitetura é Mapeada em Memória.

Controle de transferência de dados
E/S controlada por hardware (Direct Memory Access - DMA)

As operações de E/S controladas pela CPU são inadequadas para a transferência de dados entre a Memória e dispositivos de "alta velocidade", como as unidades de disco magnético.

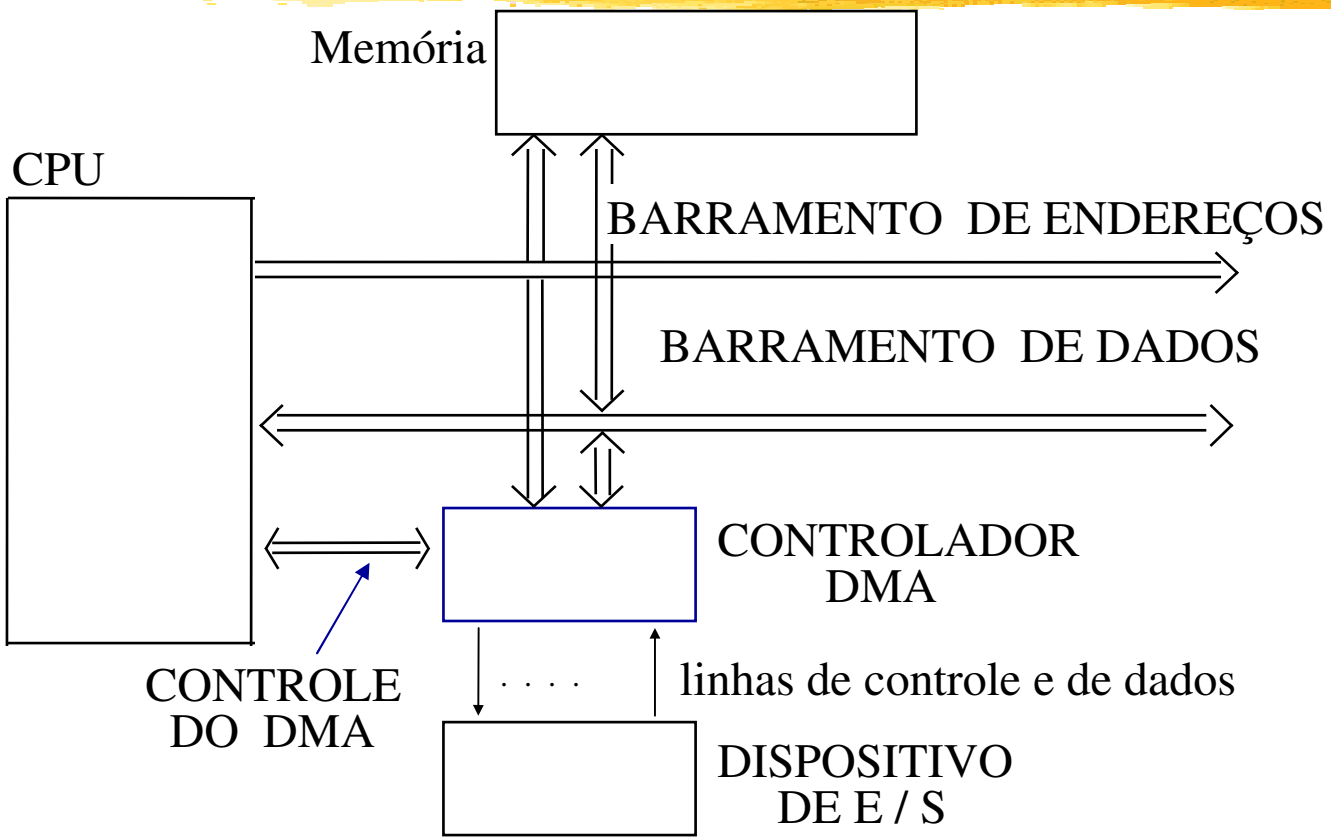
Nestes casos são transferidos grandes blocos de dados e a execução repetitiva das instruções de E/S torna a transferência ineficiente.

Solução: usar um hardware específico.

Com o *Controlador DMA*, a CPU inicia a transferência de um bloco de dados pelo carregamento de registradores especiais desta interface; após a inicialização o Controlador DMA realizará a transferência de dados e a CPU ficará livre para continuar a execução do programa (durante a transferência).

Controle de transferência de dados

E/S controlada por hardware (DMA) – cont.



DMA
detalhamento de sua operação

É basicamente utilizado para transferência de *blocos de dados* entre disco e memória principal sem a intervenção da CPU.

A CPU pode efetuar um outro processamento enquanto o controlador DMA processa a transferência.

A transferência é controlada através de registradores do controlador.

Exemplos de registradores existentes em controladores DMAs:

- reg. para armazenar o endereço inicial, na memória, do bloco de dados;
- reg. para armazenar o tamanho do bloco de dados;
- reg. (ou buffer) fonte e destino dos dados; etc.

Controle do DMA

- **Transferência controlada por programa:**

O programa inicia a transferência de um bloco, isto é, "programa" o controlador DMA e a CPU prossegue executando o programa.

Quando o programa está pronto para utilizar os dados do bloco, ou quando é necessário processar a transferência de outro bloco, o programa entra em um *laço de espera*, consultando o registrador de status do controlador, aguardando que o controlador DMA complete a transferência.

- **Transferência controlada por interrupção**

O programa também inicia a transferência de um bloco, isto é, "programa" o controlador DMA e a CPU prossegue executando o programa.

Ao terminar a transferência do bloco, o controlador DMA gera um sinal de interrupção que, aceito pela CPU, interrompe a execução do programa e inicia a execução de uma rotina de serviço para, por exemplo, iniciar a transferência do bloco seguinte, retornando em seguida o controle ao programa.

Modos de transferência de dados no DMA

O controlador DMA e a CPU compartilham os mesmos barramentos para transmitir dados e sinais de controle. Portanto, alguma estratégia deve ser introduzida para coordenar o uso destes barramentos.

O uso de barramentos pelo controlador DMA pode ser feito de três formas:

- *rajada* ("burst"),
- *furto de ciclo* ("cycle stealing"), ou
- *DMA transparente*.

Modos de transferência de dados no DMA

Rajada

O controlador requisita os barramentos quando está pronto para a transferência e, uma vez de posse dos mesmos, transfere todo o bloco de palavras enquanto a CPU aguarda o término da transferência.

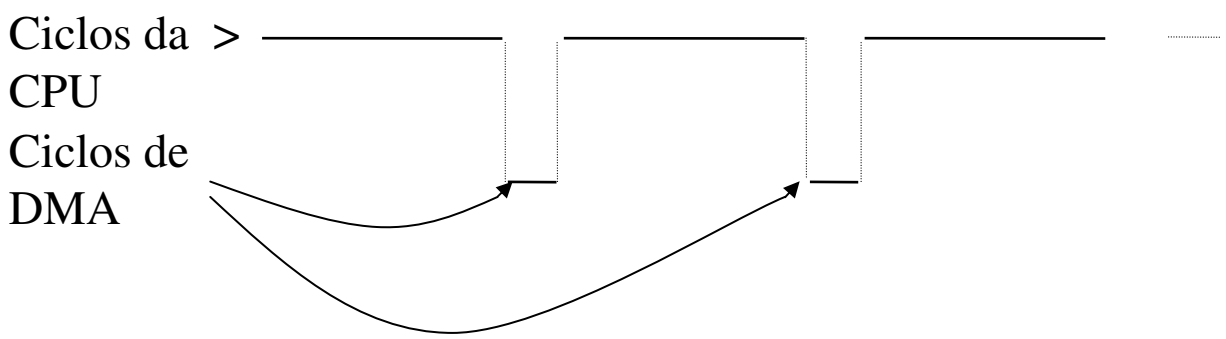
Embora não haja paralelismo entre E/S e CPU, a transmissão é mais rápida do que a E/S controlada pela CPU, no caso de dispositivos rápidos.

A transferência dos dados é feita diretamente da interface para a memória, não passando pela CPU.

Modos de transferência de dados no DMA

Roubo de ciclo (ciclo stealing)

O controlador assume o controle dos barramentos durante o tempo necessário para transmitir uma palavra e após retorna o controle dos barramentos à CPU. Durante este tempo a CPU não pode executar operações que façam uso dos barramentos.

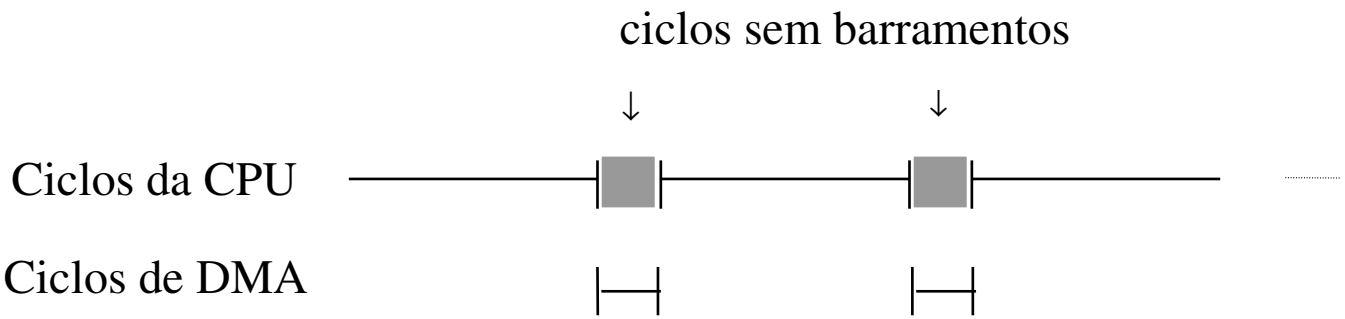


A execução de programas pela CPU fica um pouco mais lenta pois ela espera pela liberação dos barramentos; contudo, nem sempre a CPU precisa esperar pois podem coincidir ciclos de DMA com ciclos de CPU com operações internas que não façam uso do barramento.

Modos de transferência de dados no DMA

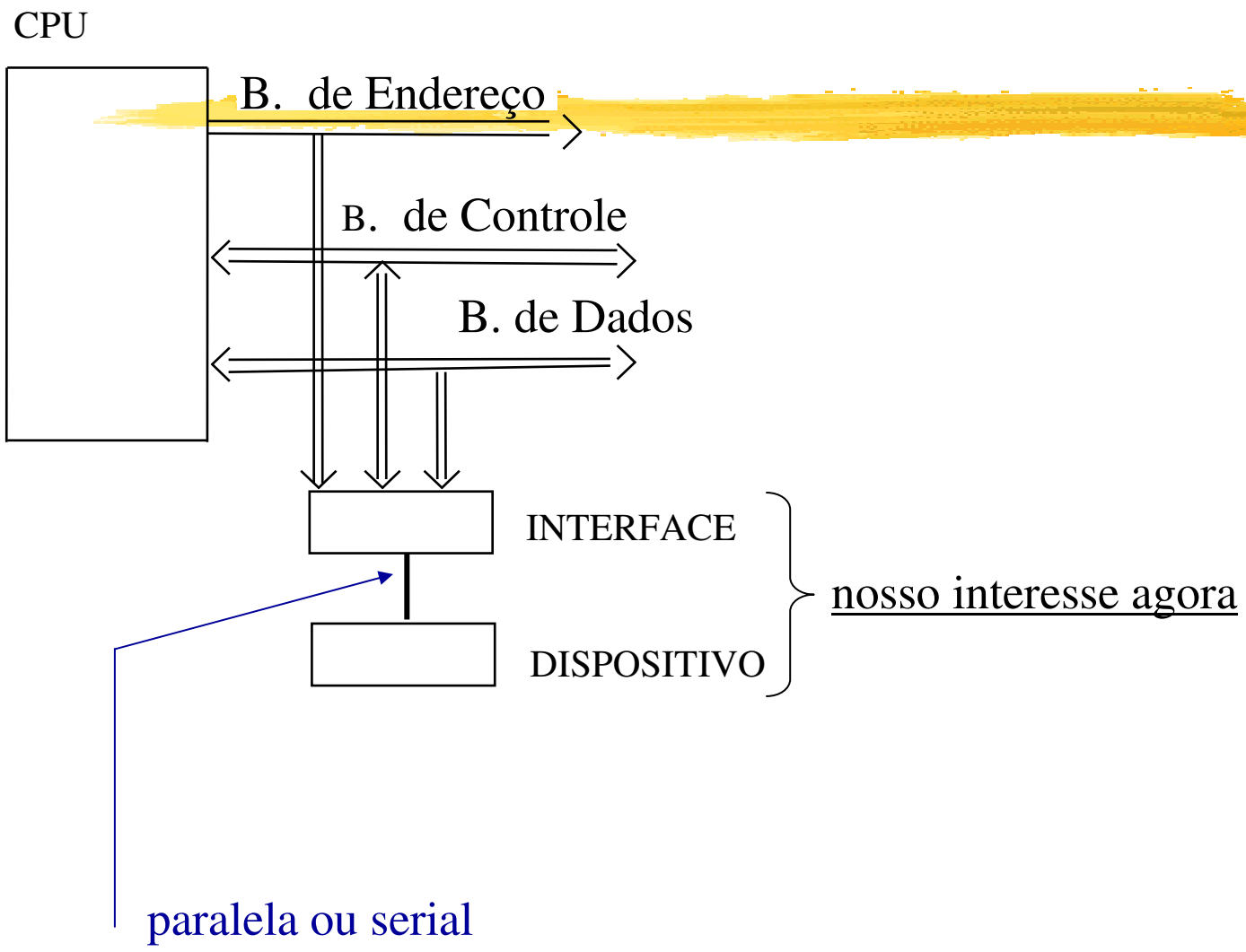
DMA transparente

O controlador DMA utiliza, para transferência de dados, ciclos de máquina da CPU que não façam uso dos barramentos. Deste modo, não há interferência do DMA na execução de instruções pela CPU. Para isto, é necessário que o controlador DMA identifique o início de um ciclo de máquina da CPU que não irá utilizar os barramentos.

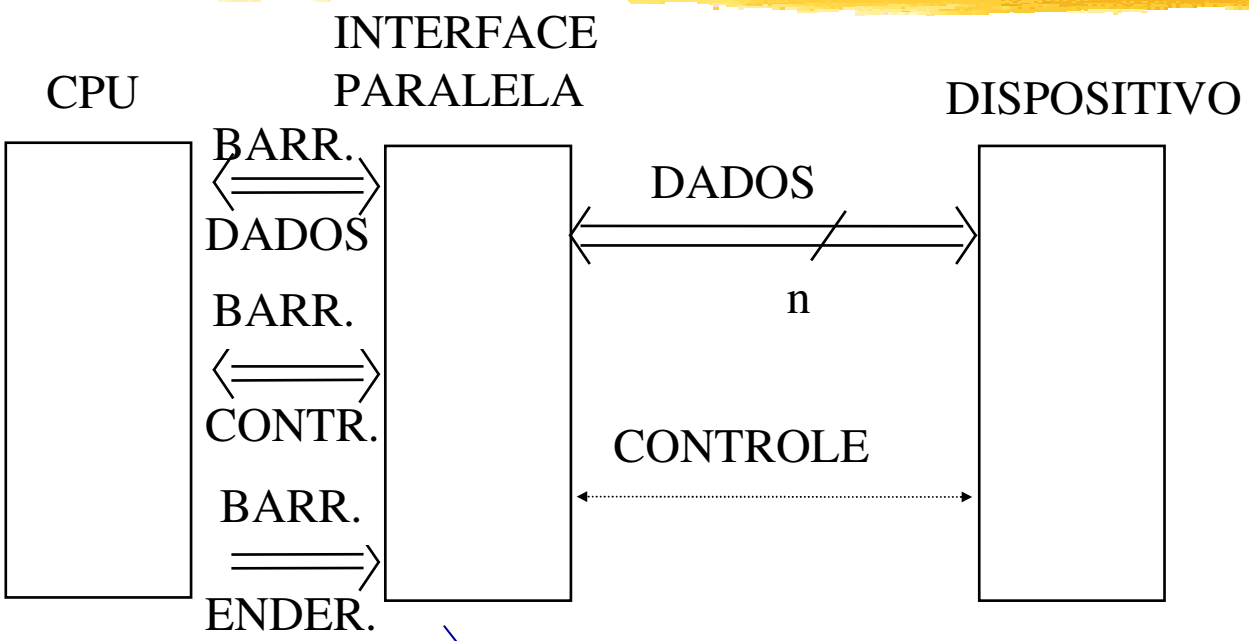


Tipicamente, o controlador DMA libera totalmente a CPU durante a transferência de dados, aumentando o paralelismo do conjunto (CPU, memória, E/S), mas não necessariamente aumentando a taxa de transferência de dados entre o dispositivo e a memória.

8.5 Comunicação Interface - Dispositivo



Comunicação Paralela



Uma interface paralela:

- recebe / envia dados (entre CPU e dispositivo)
- comunicação através n linhas (em paralelo) de dados

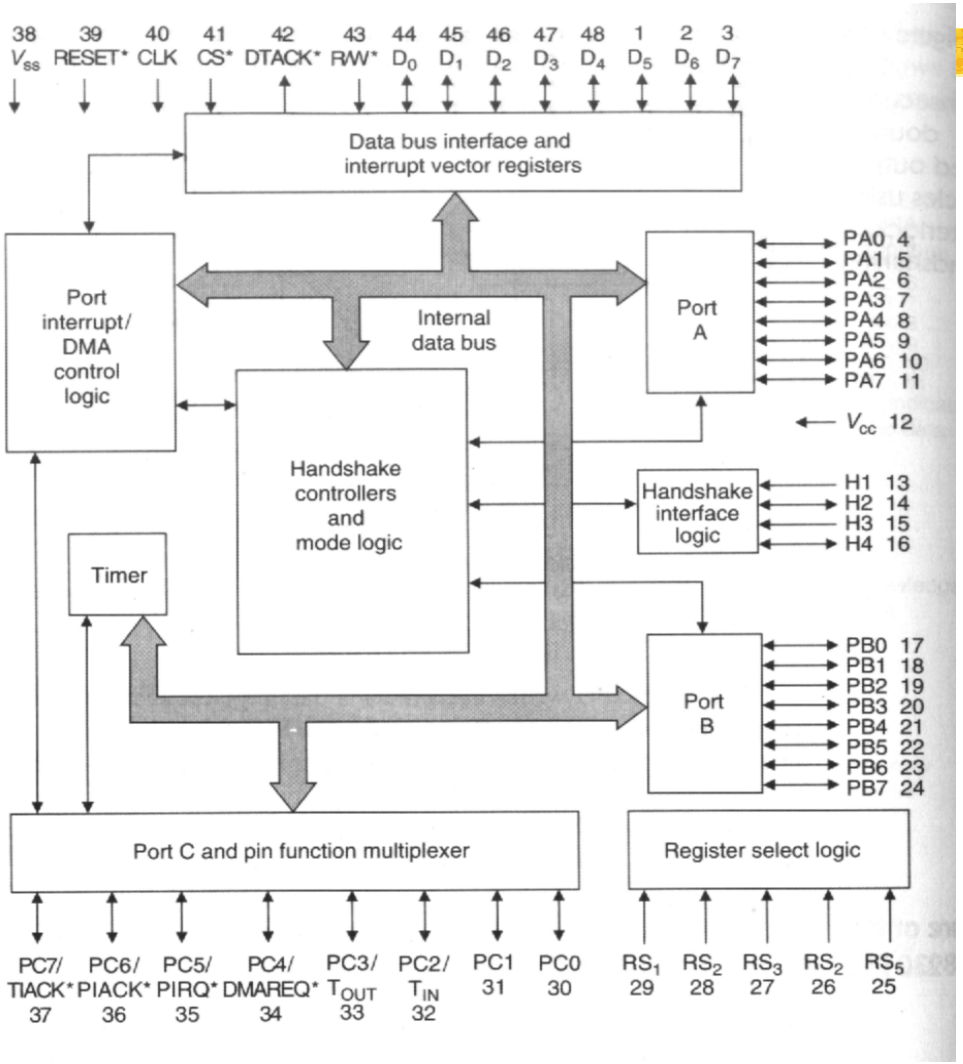
Exemplo: Motorola 68230 PI/T Parallel Interface/Timer

- 8 / 16 bits paralelos entre a interface e o dispositivo externo
- E / S com buffering e handshaking

permite sobreposição na transferência de dados entre CPU- Interface e Interface-Dispositivo

transferência de dados pode ser "interlocked" com um dispositivo externo, ou seja, próxima ação aguarda término da atual.

M 68230 (fonte: A. Clements, Microprocessor Systems Design, PWS Pub., 3. edição, 1997 – cap.8)



Porta A: A0 – A7

Porta B: B0 – B7

Porta C: C0 – C7

também usado para interrup.

B.Dados: D0 – D7

B.Endereço: RS1 – RS5

CS: Chip Select

R/W: read ou write

M 68230 (fonte: A. Clements, Microprocessor Systems Design, PWS Pub., 3. edição, 1997 – cap.8) – cont.

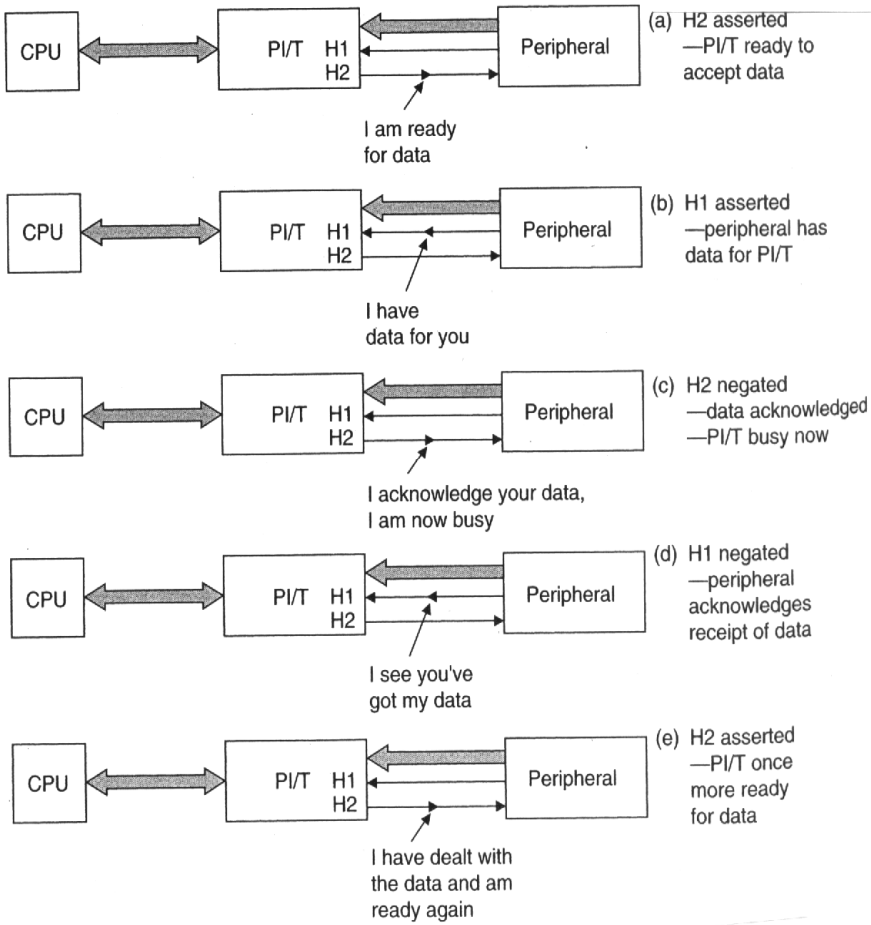


Figure 8.19 Closed-loop data transfer and the input handshake

Entrada com handshaking

Linhas de controle são H1 e H2,

H1-entrada sensível à nível

H2 – saída

(a) H2 informa: pronto para dado

(b) H1 informa: há dado para PI/T

(c) H2 é negado: ACK do dado e PI/T ocupado

(d) H1 é negado: periférico ACK do dado

(e) H2 informa: pronto para dado ...

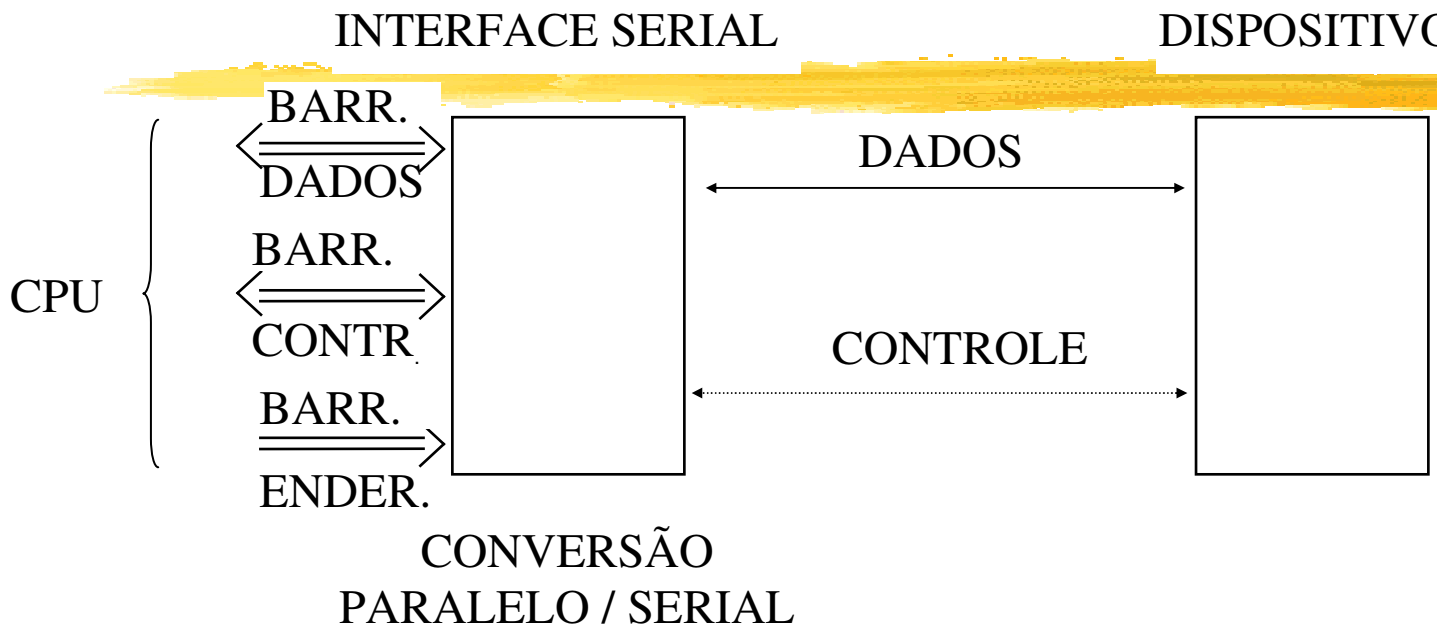
Comunicação Serial

A comunicação serial requer um número muito menor de linhas de comunicação (fios) entre interface e dispositivo, tipicamente de 2 a 4 linhas.

É apropriada quando:

- o dispositivo de saída tem uma operação inerentemente serial, e/ou
- a distância entre a CPU e o dispositivo é grande, o que influi no custo, por exemplo, do cabo de transmissão.

Comunicação Serial – cont.

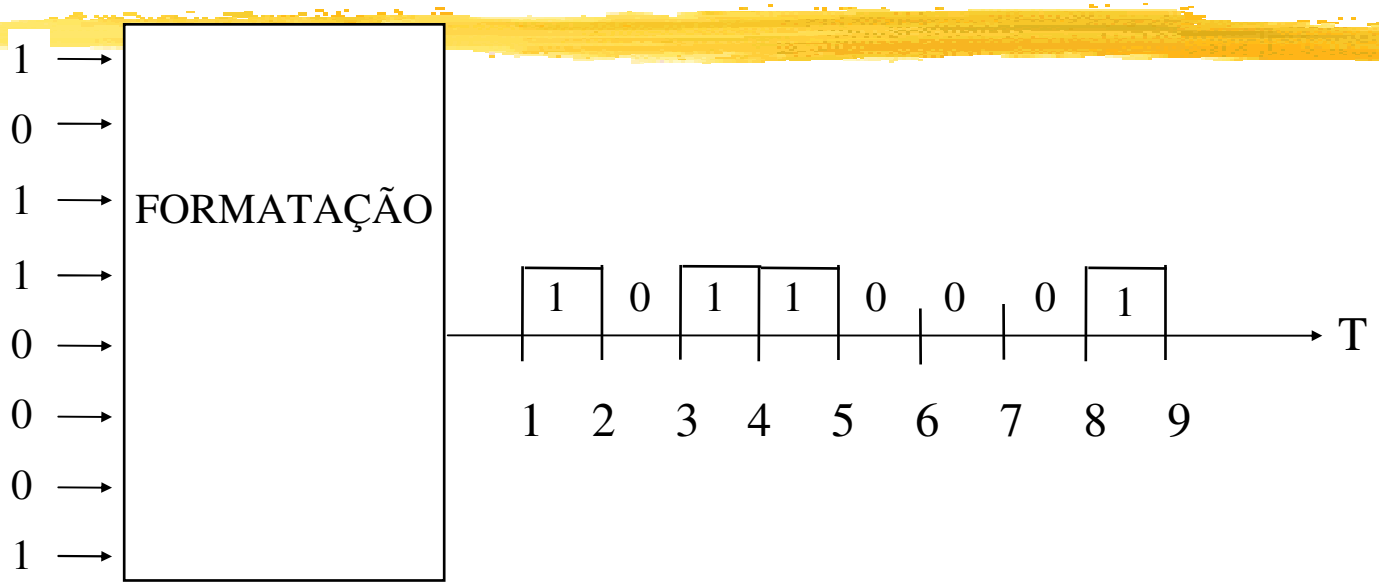


A interface serial deve prover duas funções:

- *formatação lógica dos dados*, incluindo a conversão paralelo/serial e serial/paralelo,
- *compatibilização*, a nível elétrico, para transmissão e recepção de dados.

Comunicação Serial – cont.

Formatação: Transformação paralela - serial



A formatação executada pela interface deve estar de acordo com a formatação que o dispositivo "entende", ou seja, com o tempo de amostragem do dispositivo.

Para que isto aconteça, existem taxas de transmissão padronizadas, denominadas "*Baud Rate*". No caso de sinais binários, *Baud Rate* corresponde ao número de bits por segundo.

Exemplos de "*Baud Rate*" padronizados vão de valores de 110 a 28800 bits/segundo, com valores discretos dentro desta faixa.

Comunicação Serial – cont.

Conexão Interface - Dispositivo

Tipicamente, existem 3 configurações de transferência serial:

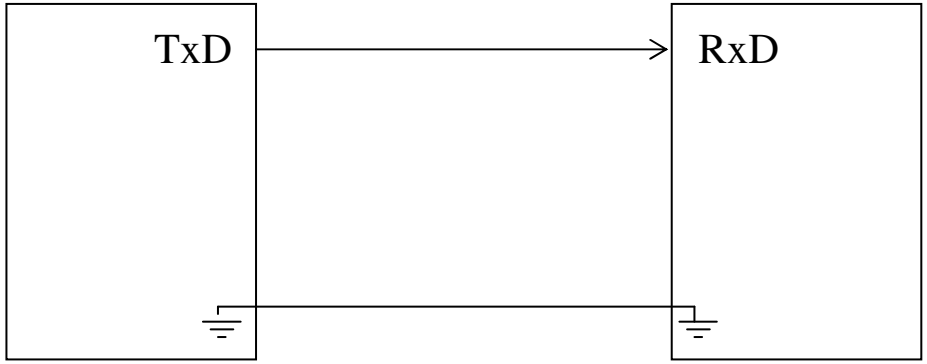
- **Simplex** - um único sentido de transmissão ;
- **Half Duplex** - ambos os sentidos de transmissão, um de cada vez ; e
- **Full Duplex** - ambos os sentidos de transmissão, simultaneamente.

RS-232C, definido em 1969, foi o primeiro padrão (americano) e largamente empregado em comunicação serial. Usamos a notação deste padrão aqui.

Comunicação Serial – cont.

Conexão Interface – Dispositivo: Simplex

Simplex - um único sentido de transmissão

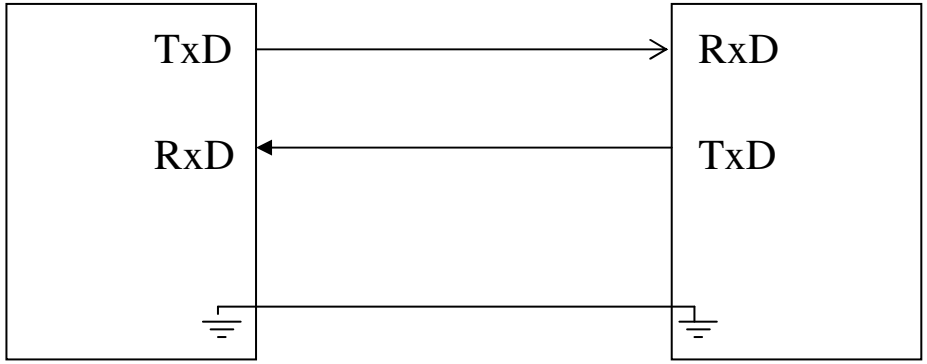


Neste caso não temos nenhum sinal de controle e comunicação unidirecional.

Comunicação Serial – cont.

Conexão Interface – Dispositivo: Duplex

Duplex – dois sentidos independentes de transmissão

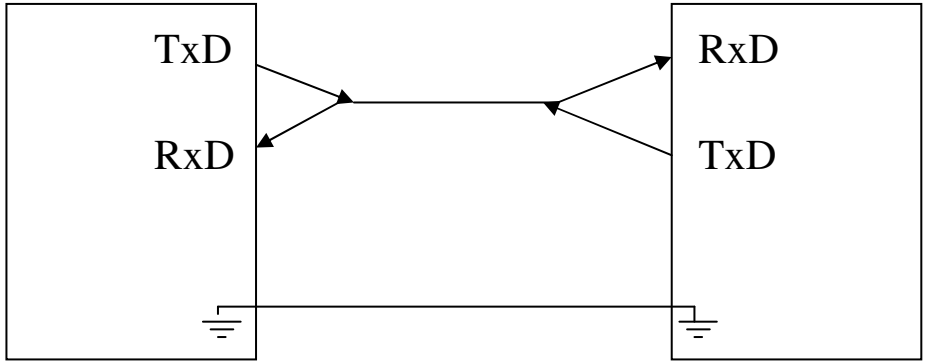


Neste caso não temos nenhum sinal de controle e comunicação bidirecional.

Comunicação Serial – cont.

Conexão Interface – Dispositivo: Half-duplex

Half-duplex – dois sentidos interdependentes de transmissão



Neste caso não temos nenhum sinal de controle e comunicação bidirecional, porém um sentido de cada vez.

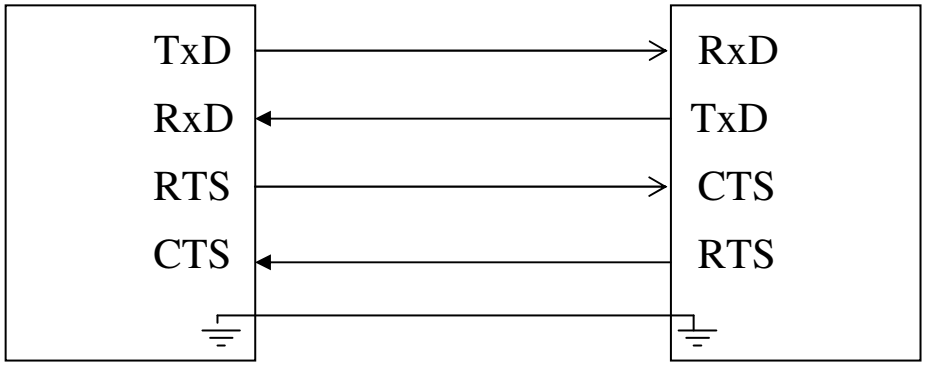
Comunicação Serial – cont.

Conexão Interface – Dispositivo: linhas de controle

Normalmente a comunicação dá-se com controle de um ou dos dois lados.

Linhas de controle usuais:

- RTS - request to send: informa ao destino que fonte deseja transmitir;
- CTS – clear to send: informa à fonte que destino pode receber;



Comunicação Serial – cont.

Conexão Interface – Dispositivo: Modos de Transferência

MODOS DE TRANSFERÊNCIA SERIAL
(caracter a caracter, sem modulação)

A transferência serial de dados pode ser ***síncrona*** ou ***assíncrona***.

Transferência Serial Assíncrona: é utilizada em transferências de baixa velocidade.

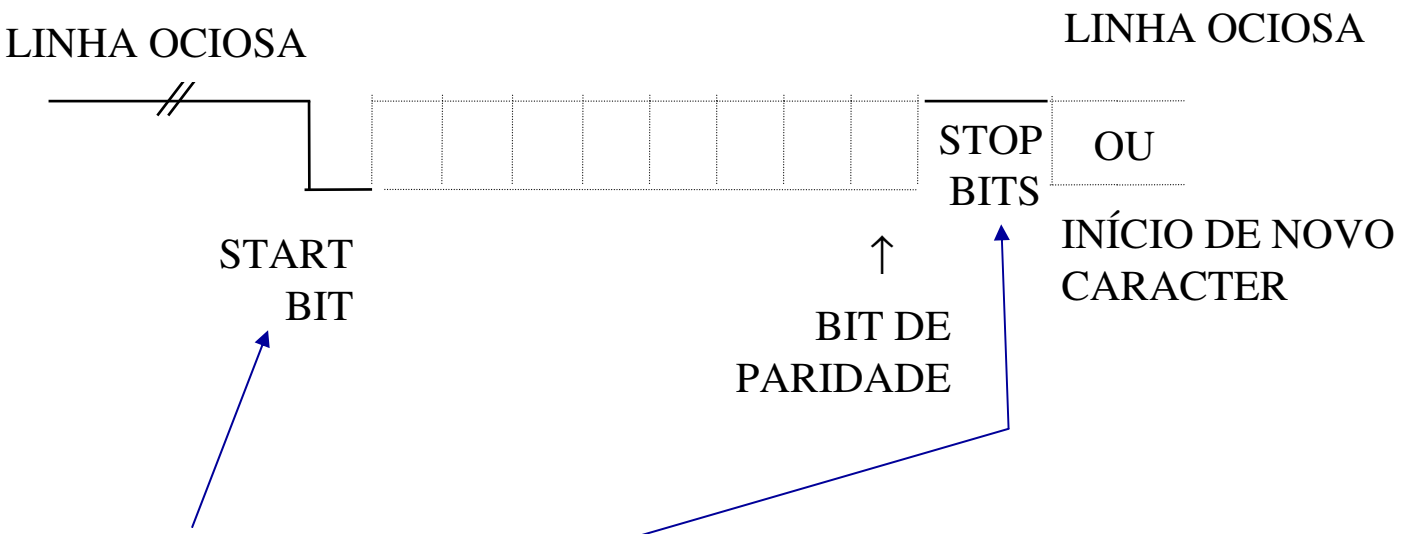
Os caracteres são transferidos um a um, na medida em que haja caracteres a serem transferidos.

Desta forma, a linha está ociosa ou transmitindo um caracter.

Para que o receptor detete que um carater está sendo transmitido, a transferência do carater é precedida por um sinal de início de transferência - "start bit" – e, ao final, seguida de um sinal de fim de transferência - "stop bit".

Comunicação Serial – cont.

Transferência Serial Assíncrona



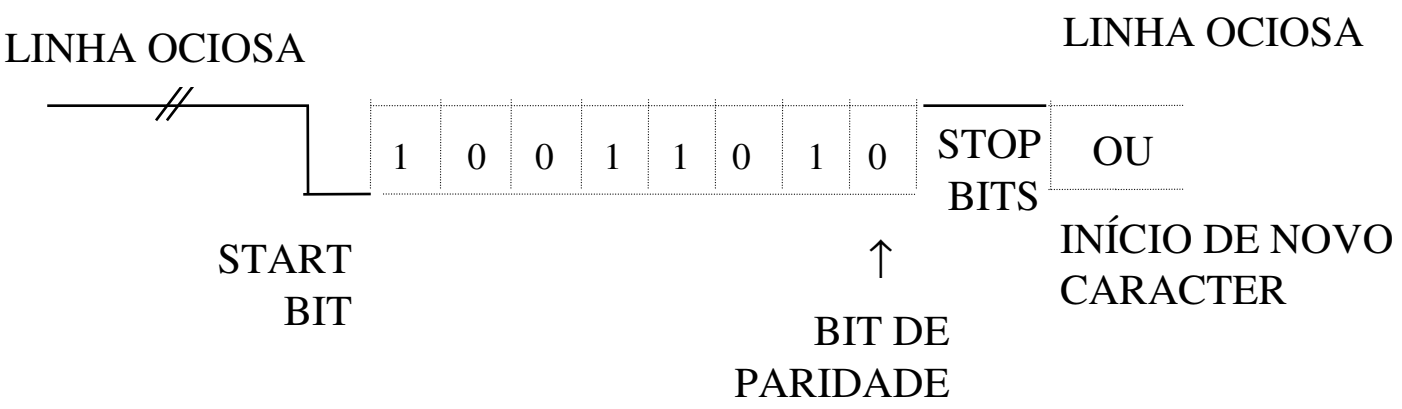
START BIT - igual a zero, indica o início de transmissão sincronizando transmissor e receptor

STOP BITS - dois bits iguais a 1, indicando o fim de transmissão

Comunicação Serial – cont.

Transferência Serial Assíncrona

Exemplo: Transmissão do carater M (em ASCII - 1001101_2 ou $4D_{16}$)



a transição de cada start-bit 1 → 0 inicializa o clock do receptor para o início da recepção da cadeia de bits.

Comunicação Serial – cont.

Transferência Serial Assíncrona – efeito de atrasos entre relógios

Um aspecto crítico de transmissão assíncrona é a precisão do relógio de recepção (com relação à geração do caráter).

1. Suponhamos que a transição 1 → 0 do Start-bit aconteça e que o relógio do receptor espere T/2 segundos e a partir daí amostre a cada T segundos até amostrar o Stop-bit.
2. Vamos agora assumir que o relógio do receptor esteja operando mais lentamente a T+dt.
3. O 1. bit de dado será amostrado em T+dt após o Start-bit assim: (T+dt)/2 + (T+dt)
4. O Stop-bit será amostrado em: (T+dt)/2 + N.(T+dt), sendo N o número de bits

   Assim o erro total acumulado ao amostrar o Stop-bit é:

   (T+dt)/2 + N.(T+dt) – (T/2 + NT) ou, (2N+1)dt/2 segundos.
5. Para um comportamento correto o Stop-bit deve ser amostrado dentro de T/2 segundos:

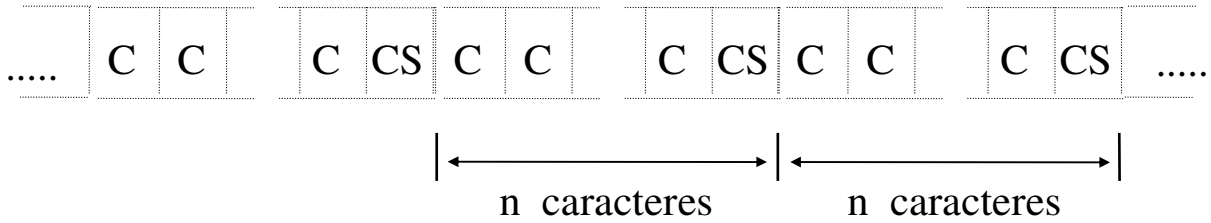
   $T/2 > (2N+1)dt/2$ ou $dt/T < 1/(2N+1)$

   Admitindo N=9 (ASCII de 7 bits + 1 paridade + 1 Stop-bit): 1/19 ou 5 por cento.
6. Ou seja a discrepância entre os relógios do gerador e do receptor não podem diferir em mais do que 5%. O que é um valor plenamente satisfeito atualmente.

Comunicação Serial – cont.

Transferência Serial Síncrona

Permite transmissão mais rápida.



C - caracter

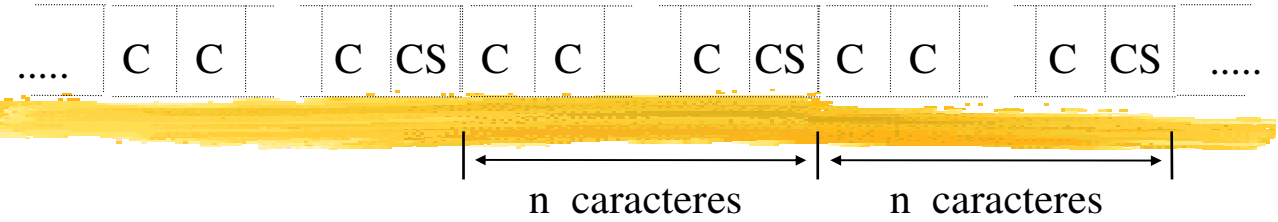
CS - caracter de sincronismo

Os caracteres são transmitidos ininterruptamente, eliminando a necessidade dos bits de sincronismo como o "start bit" ou "stop bit".

Os dados transmitidos ou recebidos constituem uma *sequência contínua de bits* sem indicação de fronteiras entre caracteres adjacentes.

Comunicação Serial – cont.

Transferência Serial Síncrona



C - caracter

CS - caracter de sincronismo

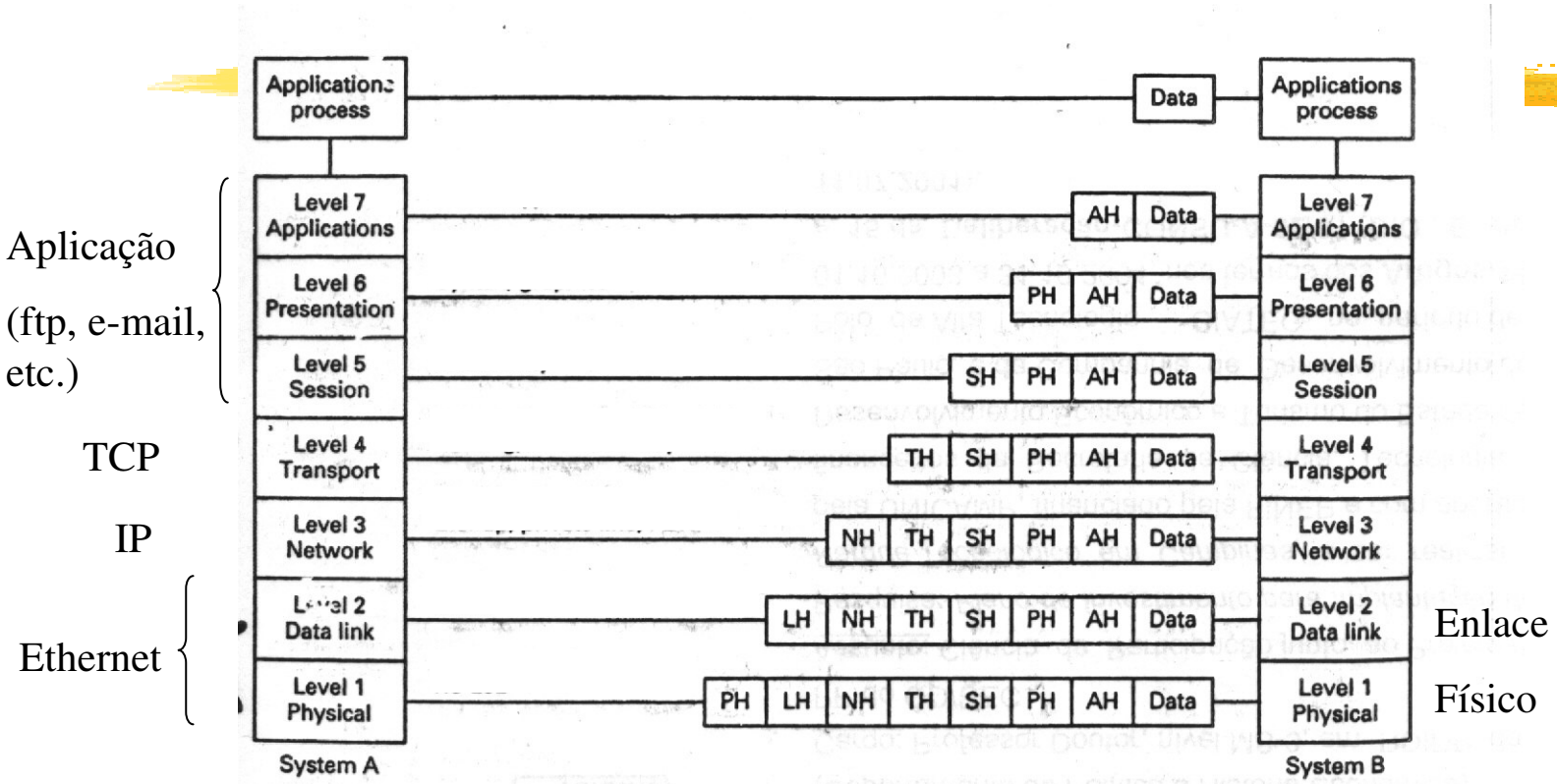
Os relógios no receptor e no transmissor operam exatamente na mesma frequência e devem ser muito estáveis para manter o sincronismo por um longo período de tempo.

O sincronismo é mantido através do chamado *caracter de sincronismo* (CS).
A cada n caracteres de informação, o transmissor gera um caracter de sincronismo e o receptor deve detetar este caráter. Ver sincronismo por codificação Manchester ou codificação de fase (Tr. A-8)

Início da transmissão: receptor opera em um modo de busca ("hunt mode"), fazendo uma comparação bit a bit da sequência de entrada, com o caráter de sincronismo. Quando este caráter é encontrado, o receptor trata cada grupo subsequente de (n-1) caráteres como caráteres de informação.

O n-ésino caráter deve ser o caráter de sincronismo.

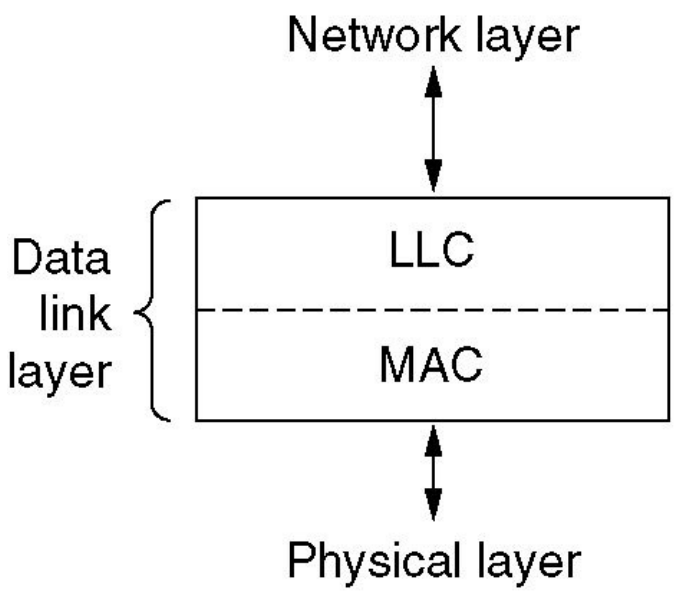
8.6 Redes – Comunicação intercomputadores



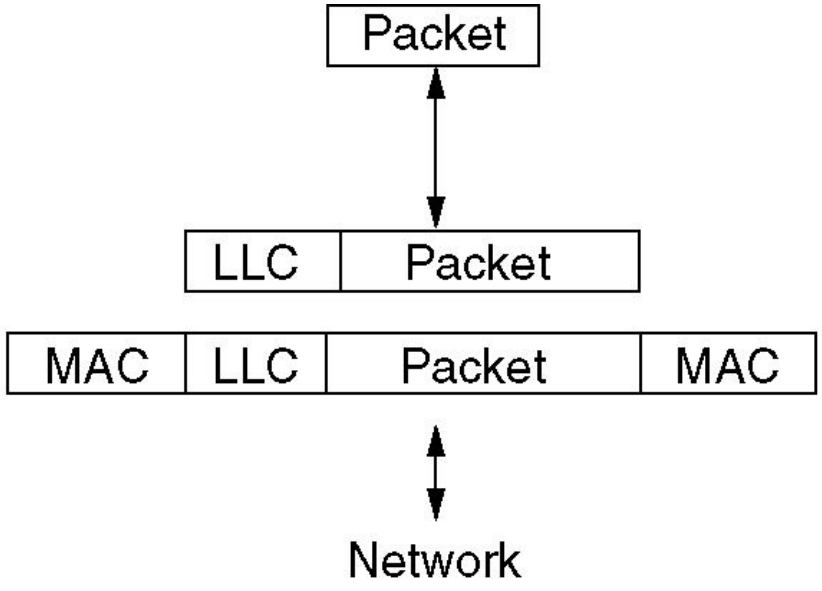
Hierarquia de protocolos – padrão ISO (International Standard Organization)

AH-applications header; PH-presentation header; SH-session header; TH-transport header; NH-network header; LH-link header; PH-physical header.

Camada Física (nível 1) & Camada de Enlace (nível 2)



(a)

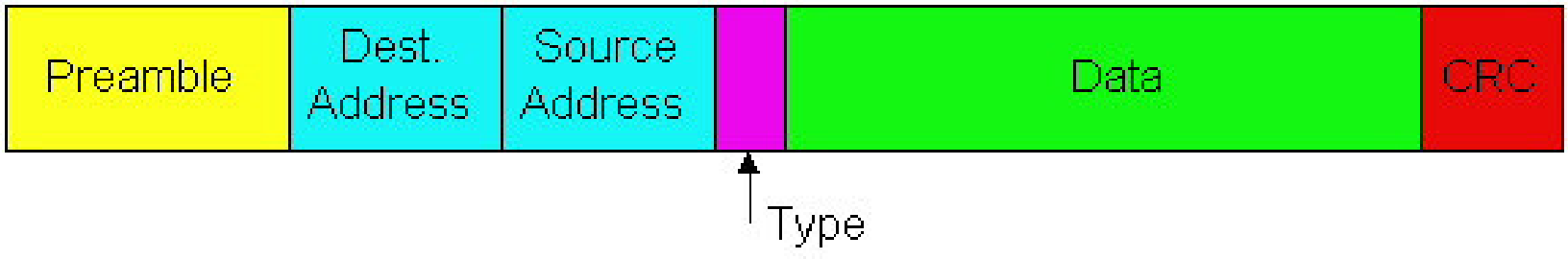


(b)

LLC: link layer control
MAC: medium access control

Estrutura do Quadro Ethernet

Adaptador do transmissor encapsula o datagrama IP (ou outro pacote de protocolo da camada de rede) num quadro Ethernet

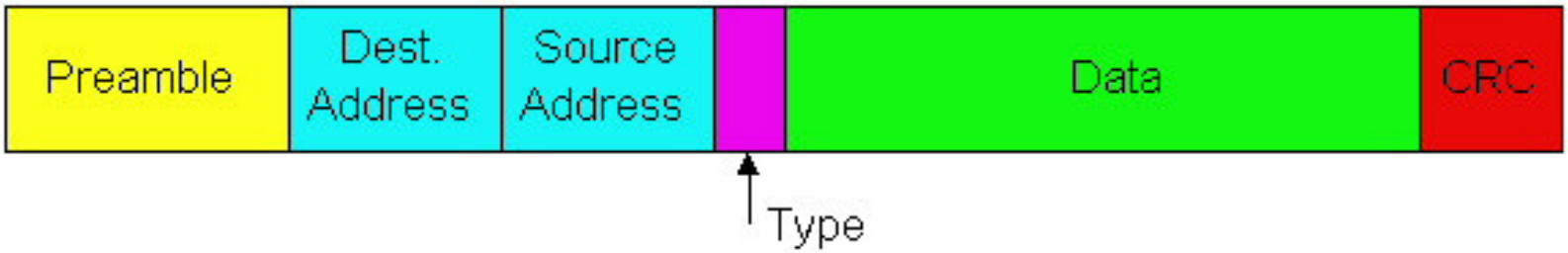


Preâmbulo:

- 7 bytes com padrão 10101010 seguido por um byte com padrão 10101011
- usado para sincronizar as taxas de relógio do transmissor e do receptor

Estrutura do Quadro Ethernet (cont.)

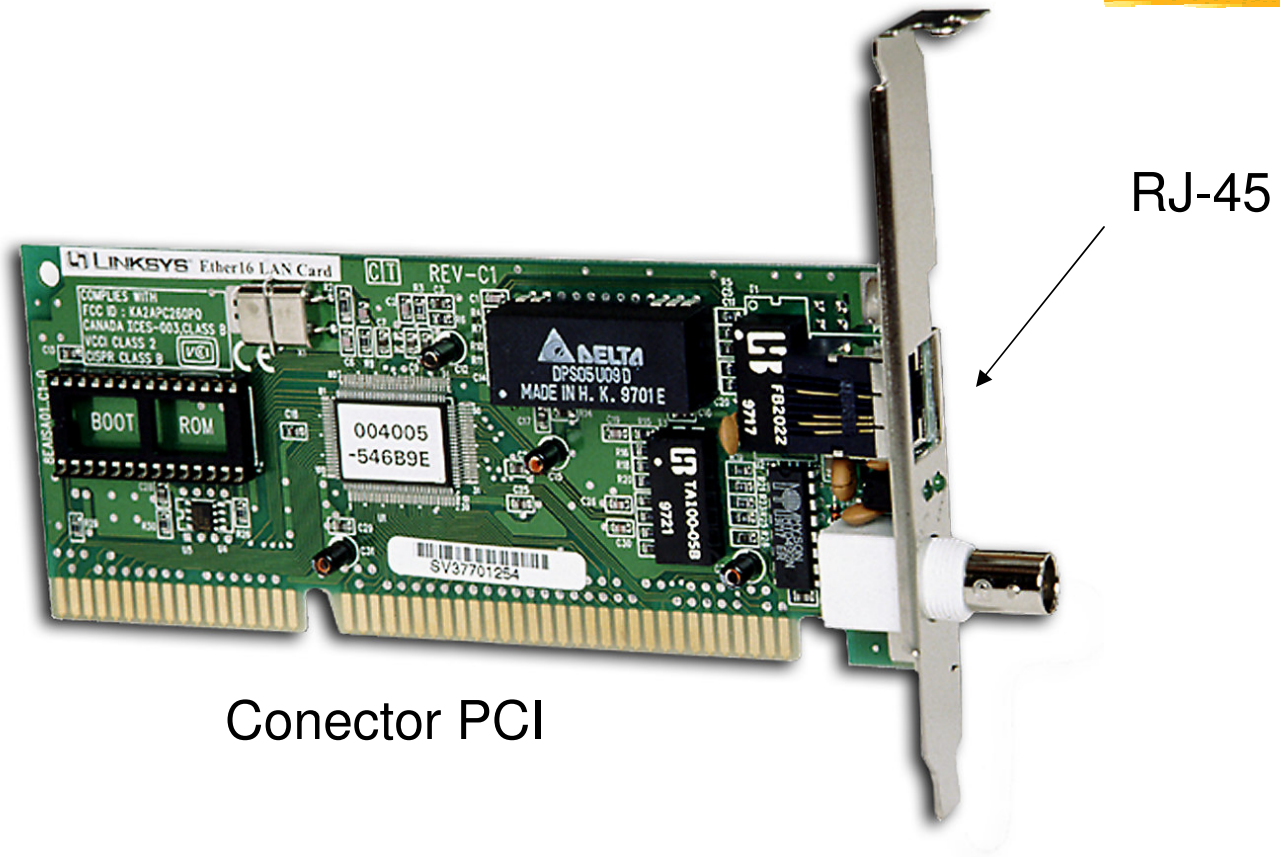
- Endereços: 6 bytes, quadro é recebido por todos os adaptadores e descartado se o endereço do quadro não coincide com o endereço do adaptador
- Tipo: indica o protocolo da camada superior, geralmente é o protocolo IP mas outros podem ser suportados tais como Novell IPX e AppleTalk
- CRC: verificado no receptor, se um erro é detectado, o quadro é simplesmente descartado.



Interface de Rede (NIC: Network Interface Card)

- Cada estação deve possuir uma interface de rede (Ex: IEEE 802.3);
- A placa de rede implementa os protocolos de rede (físico e enlace)
  - Nível físico
    - interface física & codificação do sinal;
  - Nível lógico
    - enquadramento da informação;
    - deteção de erro, controle de fluxo;
    - protocolo de acesso ao meio (MAC – Medium Access Control)
      - Ethernet (CSMA/CD); Token-Bus; Token-Ring, Sem-Fio (CSMA-CA), etc.

Placa de Rede (NIC)

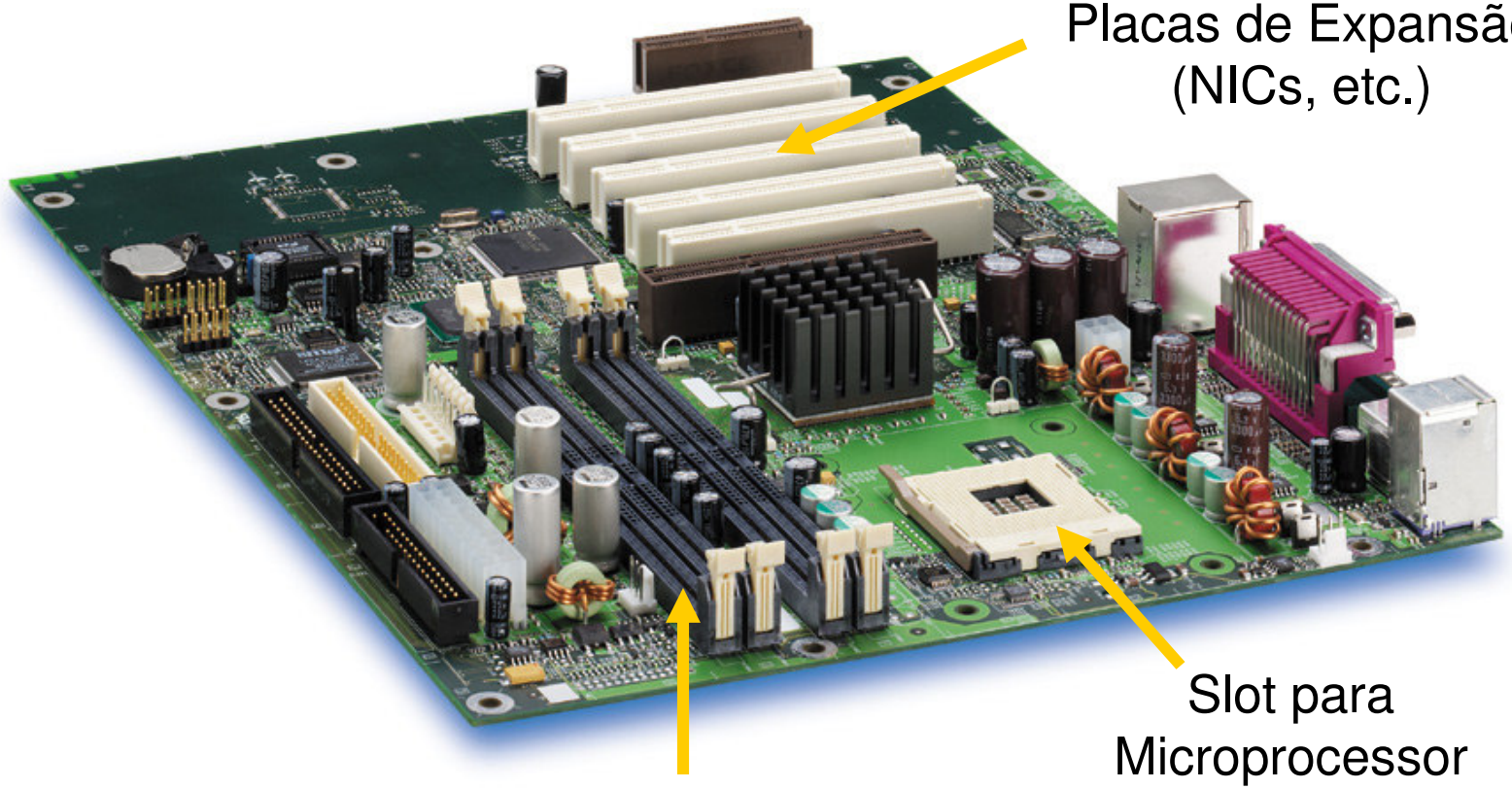


RJ-45

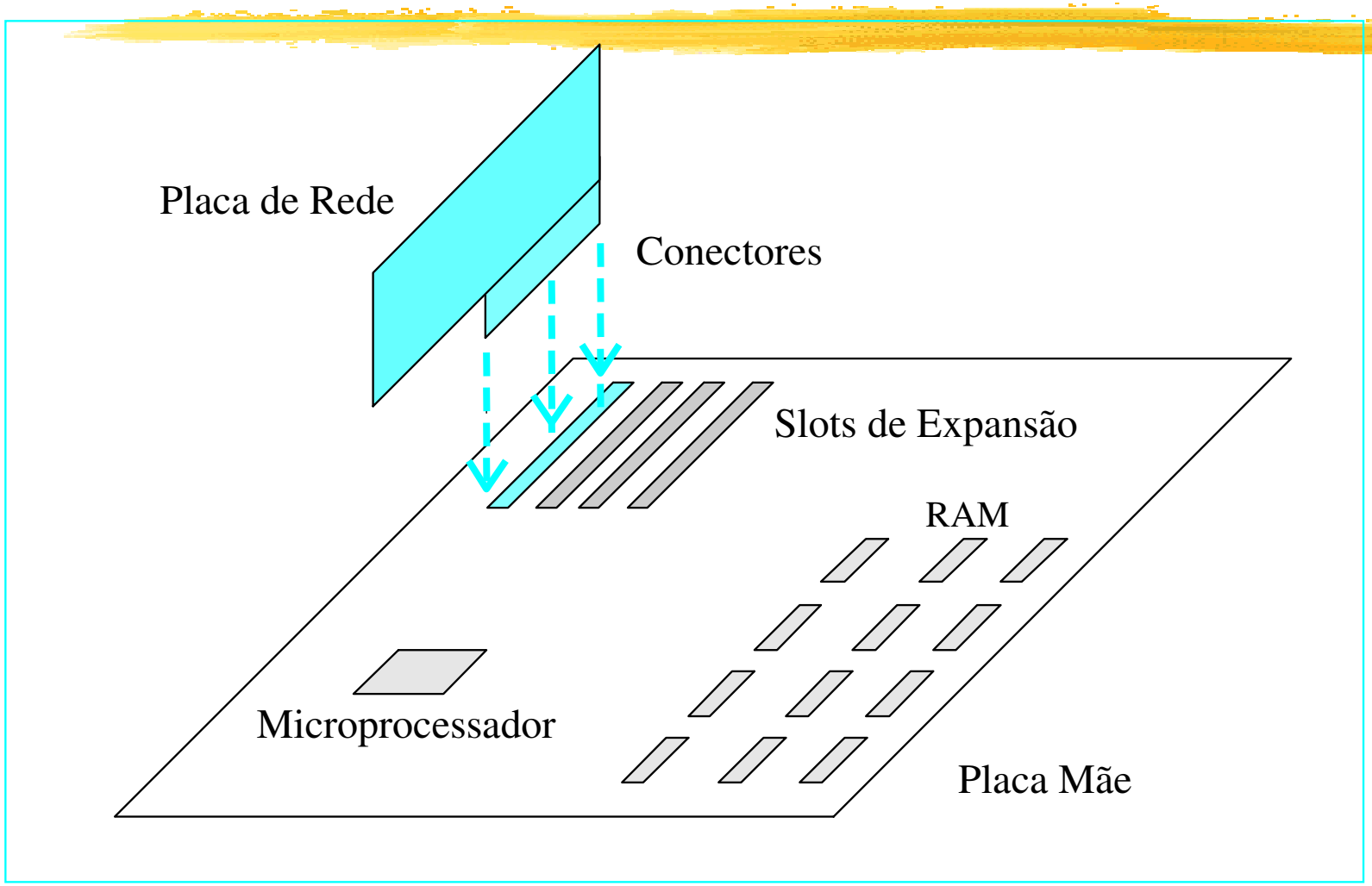
Conector PCI

Placa Mãe (Motherboard)

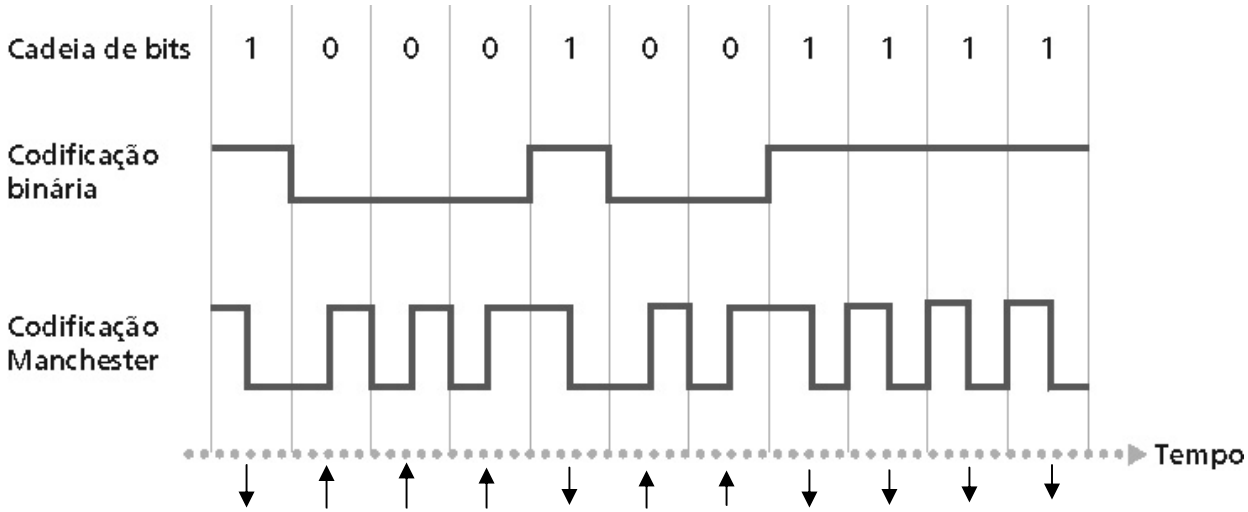
- Intel D850MV Desktop Board



NIC & Placa Mãe



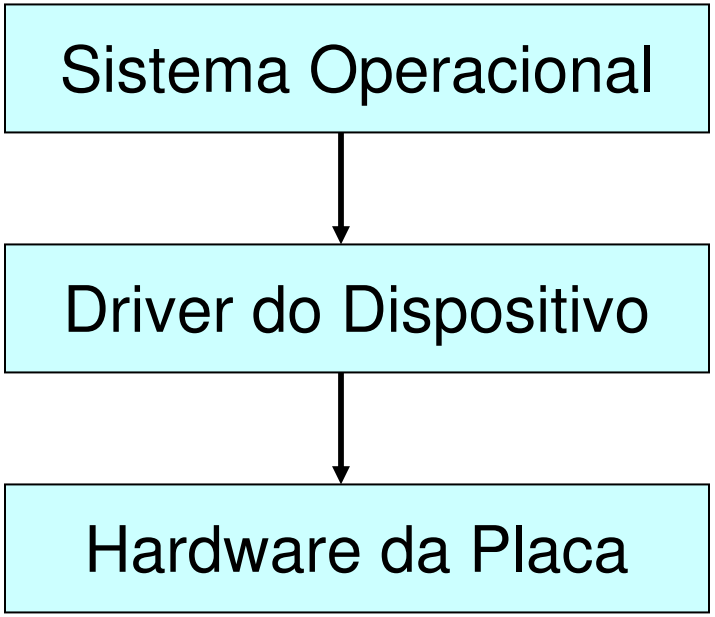
Codificação Manchester ou codificada em fase (camada física)



- Usada em 10BaseT
- Cada bit possui uma transição (1->0 = 1 ; 0 ->1 = 0)
- Permite que os relógios nos nós de transmissão e de recepção possam sincronizar um com o outro
  - Não é necessário relógio global centralizado entre os nós!

Instalação do Software

- A placa de rede necessita de um software especial: driver de dispositivo;
- Permite que o Sistema Operacional converse com o hardware da placa de rede;



Padronização (LAN)

<table>
<tr><td colspan="2">Internet</td><td colspan="2">TCP/IP</td><td colspan="2">Outros padrões</td></tr>
<tr><td rowspan="2">Camada de Enlace</td><td>Sub-camada de Enlace Lógico</td><td colspan="4">802.2</td></tr>
<tr><td>Sub-camada de Acesso ao Meio</td><td colspan="3">Ethernet 802.3</td><td>Outros Padrões MAC (802.5, etc.)</td></tr>
<tr><td colspan="2">Camada Física</td><td>10 Base-T</td><td>100 Base-TX</td><td>1000 Base-TX</td><td>Outros Padrões de Camada Física (802.5, etc.)</td></tr>
</table>

8.7 Exercícios

1. Forneça instruções típicas para a execução de E/S em arquiteturas isoladas e mapeadas em memória.
2. Resolva o exercício (13) do livro texto.
3. Explique o esquema da “handshaking” da interface 68230.
4. Discuta o funcionamento de uma interface serial. Comente os termos: simplex, halfduplex, fullduplex, síncrona, assíncrona.
5. Comente a transparência 398 (linhas de controle).