

## Aulas 12 e 13

- Barramentos paralelo vs barramentos série
- Barramentos série
  - Princípio de funcionamento
  - Sincronização de relógio entre transmissor e recetor
  - Modos de transmissão de dados: transmissão orientada ao bit, transmissão orientada ao byte
  - Topologias de ligação
  - Elementos de uma ligação série

José Luís Azevedo, Bernardo Cunha, Tomás Silva

# Introdução

- Barramentos: interligação dos blocos de um sistema de computação
  - CPU, memória, unidades de I/O
- Tipos de dispositivos ligados a um barramento:
  - **Bus Master** – Dispositivo que pode iniciar e controlar uma transferência de dados (exemplos: Processador, Módulo de I/O com DMA)
  - **Bus Slave** – Dispositivo que só responde a pedidos de transferências de dados, i.e., não tem capacidade para iniciar uma transferência (exemplos: Memória, Módulo de I/O sem DMA)
- **Barramento de um só Master:** só há um dispositivo no barramento com capacidade para iniciar e controlar transferências de informação
- **Barramento Multi-Master:** mais que um dispositivo capaz de iniciar e controlar transferências de informação (exemplos: vários CPUs, 1 ou mais controladores de DMA, um ou mais módulos de I/O com DMA)

# Introdução

- **Barramentos paralelo:** os dados são transmitidos em paralelo (através de N Linhas). Incluem:
- **Barramento de dados:**
  - Suporta a transferência de informação entre os blocos
  - O número de linhas (largura do barramento) determina quantos bits podem ser transferidos simultaneamente; a largura do barramento é um fator determinante no desempenho do sistema
- **Barramento de endereços:**
  - Especifica a origem/destino da informação
  - O número de linhas define a dimensão do espaço de endereçamento (determina a capacidade máxima de memória que o sistema pode ter:  $2^N$  palavras, sendo N o número de bits do barramento de endereços)
- **Barramento de controlo:**
  - Conjunto de sinais que especificam operações, sinalizam eventos, efetuam pedidos, ...

# Introdução

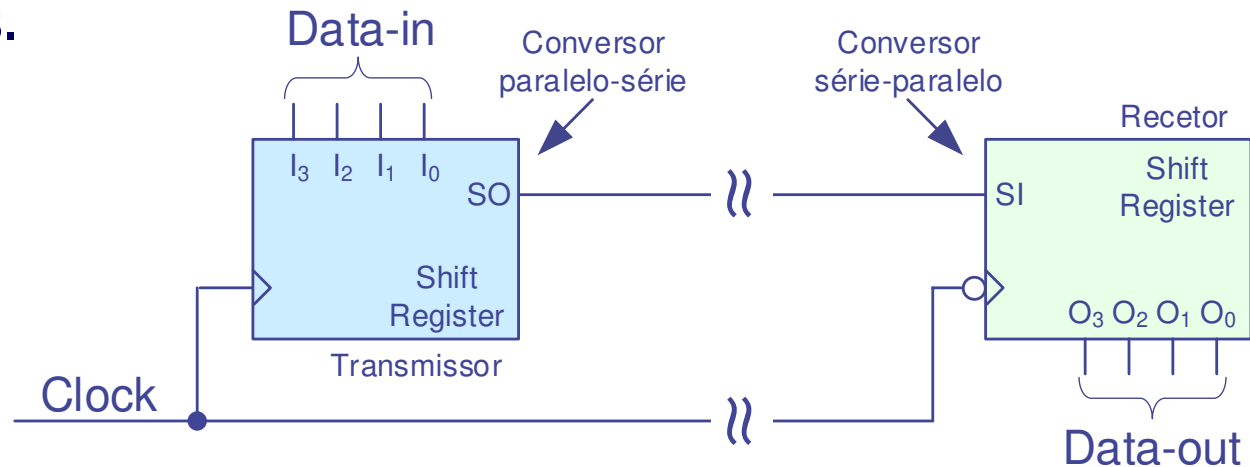
- A transmissão paralela, com relógio comum, a débitos elevados coloca problemas de vária ordem, nomeadamente:
  - Controlo do tempo de "skew" das linhas do barramento
  - Dificuldade em anular a interferência provocada por fontes de ruído externas
  - Interferência mútua, isto é, entre sinais adjacentes ("crosstalk")
  - Elevado número de fios de ligação e custo associado
  - Fichas de ligação volumosas e caras (possivelmente com contactos dourados)
- **Barramentos série:** os dados são serializados no transmissor, ou seja, transmite-se 1 bit de cada vez (tipicamente 1 bit a cada ciclo de relógio)
  - Comunicação série

# Introdução

- Vantagens dos barramentos série (ao nível físico):
  - Simplicidade de implementação
  - Simplicidade de ligação de cablagem
  - Diminuição de custos de interligação
  - Possibilidade de transmissão a distâncias elevadas (em par diferencial)
  - Débito elevado
- Tipos de comunicação série
  - **Simplex**: comunicação apenas num sentido (TX -> RX); usada, por exemplo, em telemetria, para leitura remota de sensores
  - **Half-duplex**: comunicação nos dois sentidos, mas apenas um de cada vez (é usada uma só linha)
  - **Full-duplex**: Comunicação simultânea nos dois sentidos (são usadas duas linhas)

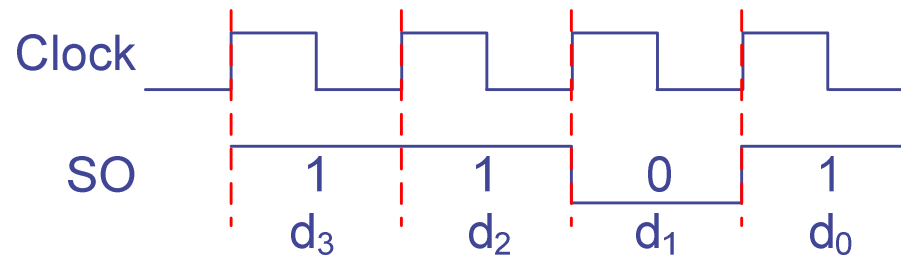
# Introdução

- Diz-se que se está na presença de um barramento ou interface série sempre que exista uma só "linha" (suporte) para transferência de dados.



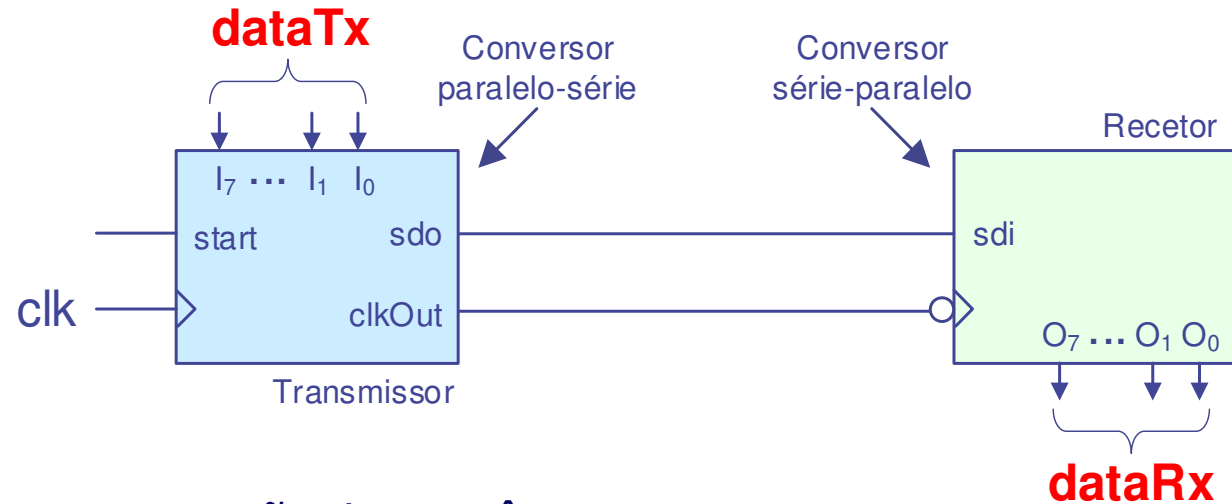
Para transmissão bidirecional podem existir 2 "linhas" separadas, uma para transmissão e outra para receção

- Exemplo  
(Data-in = 1101)

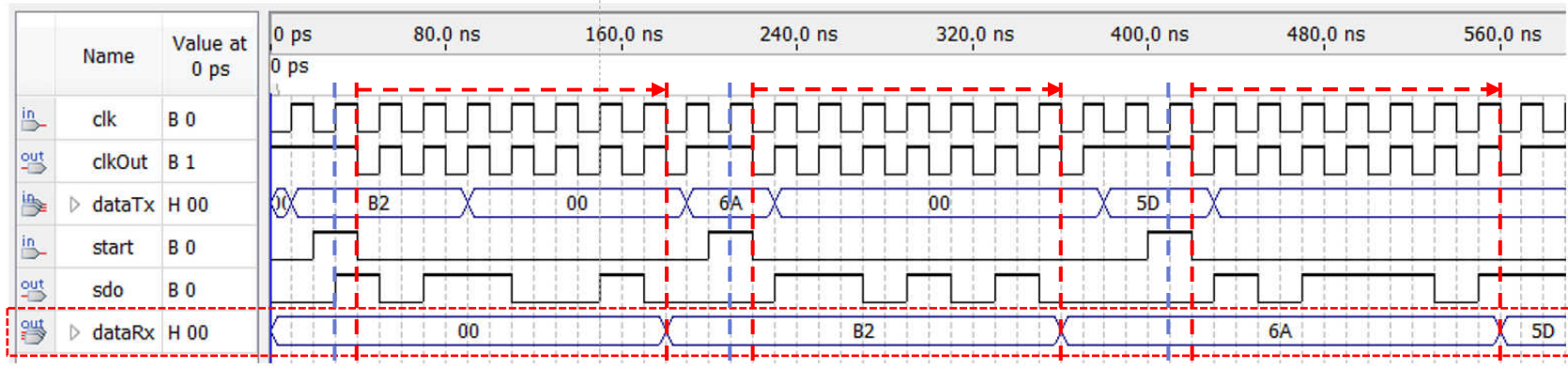


# Introdução

- Exemplo em que o transmissor gera o sinal de relógio (o sinal "start" dá início à transmissão)

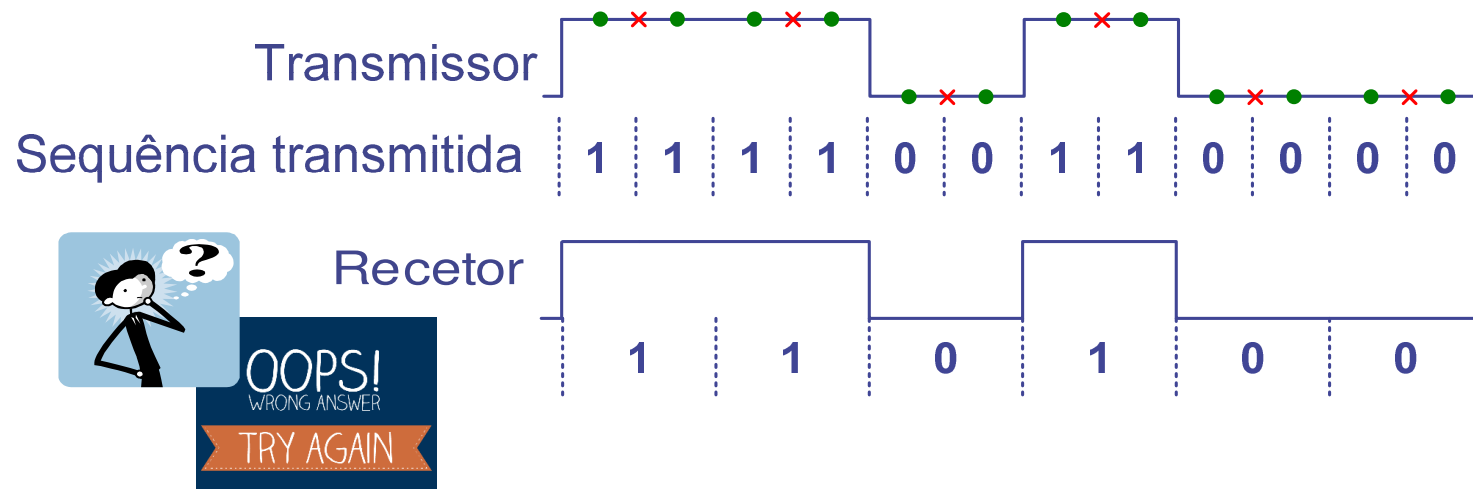


- Exemplo com transmissão da sequência: 0xB2, 0x6A, 0x5D



# Sincronização entre transmissor e recetor

- O sincronismo é obtido através da utilização do mesmo relógio no transmissor e no recetor, ou de relógios independentes que terão que estar sincronizados durante a transmissão



- Caso sejam distintos, estes relógios têm de estar sincronizados para que a amostragem do sinal seja realizada nos instantes corretos



# Sincronização entre transmissor e recetor

- **Transmissão Síncrona**

- O sinal de relógio é transmitido de forma explícita através de um sinal adicional, ou de forma implícita na codificação dos dados
- Os relógios do transmissor e do recetor têm de se manter sincronizados
- Quando o relógio não é explicitamente transmitido, o relógio do recetor é recuperado a partir das transições de nível lógico na linha de dados

- **Transmissão Assíncrona**

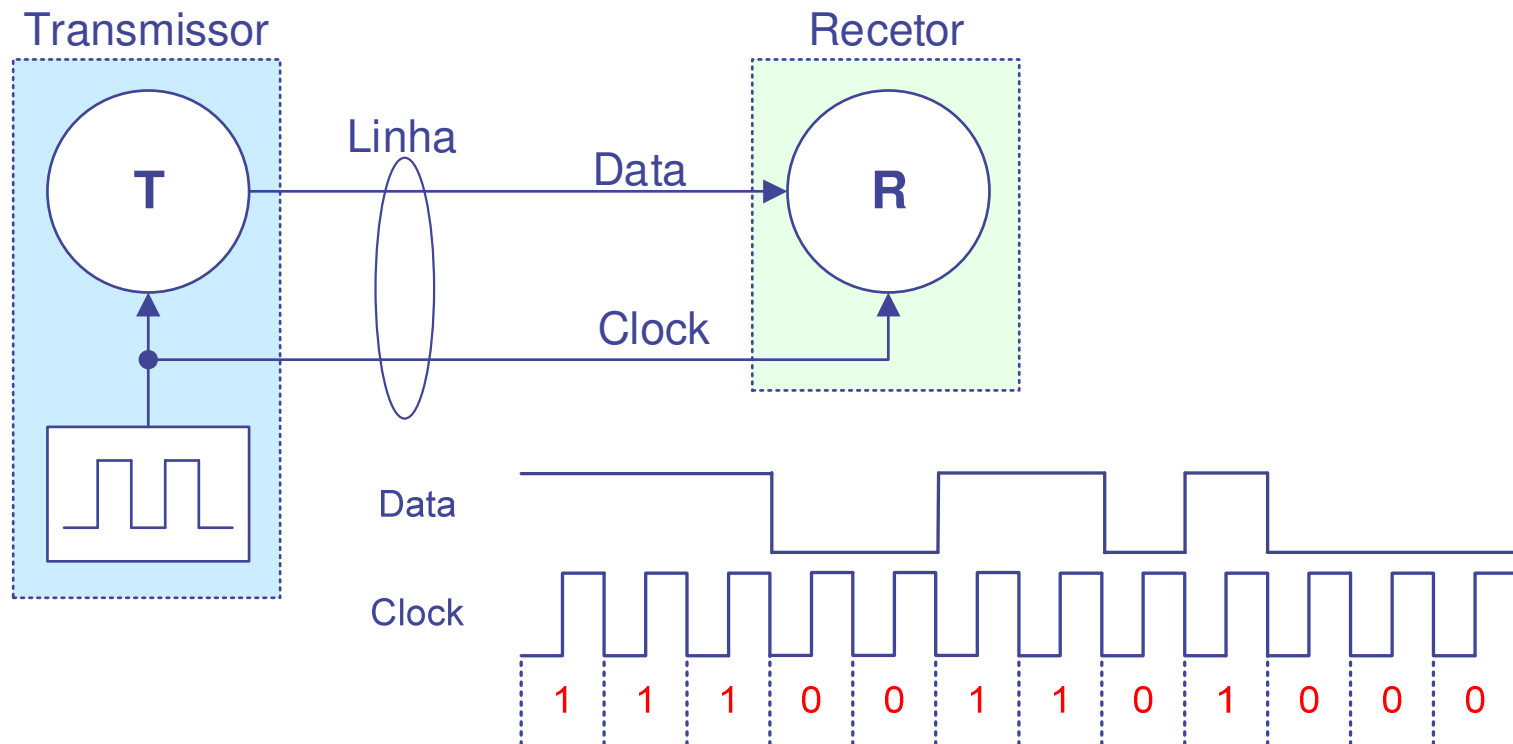
- Não é usado relógio na transmissão, nem há recuperação do relógio na receção
- É necessário acrescentar bits para sinalizar o princípio e o fim da transmissão (e.g. start bit, stop bit), que permitam a sincronização dos relógios do transmissor e do recetor

# Técnicas de sincronização do relógio

- Transmissão síncrona
  - **Relógio explícito do transmissor**
    - Exemplo: SPI
  - **Relógio explícito do recetor**
  - **Relógio explícito mutuamente-sincronizado**
    - Exemplo: I2C
  - **Relógio codificado ("self-clocking")**
    - Exemplo: USB, Ethernet
- Transmissão assíncrona
  - **Relógio implícito**
    - Exemplo: RS232

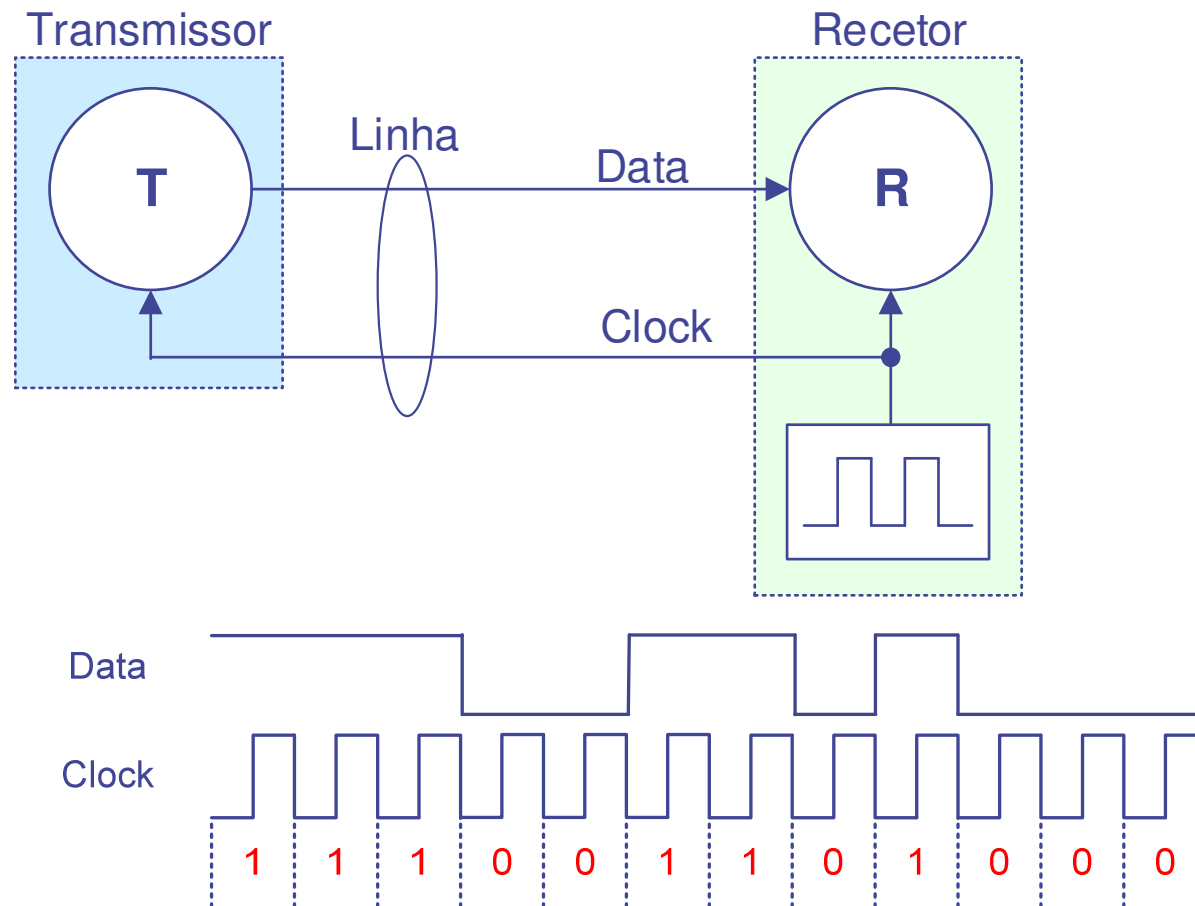
# Sincronização de relógio

- **Relógio explícito do transmissor**
- O transmissor envia os dados e informação de relógio em linhas separadas
- A linha de relógio pode ser vista como um sinal "Valid"



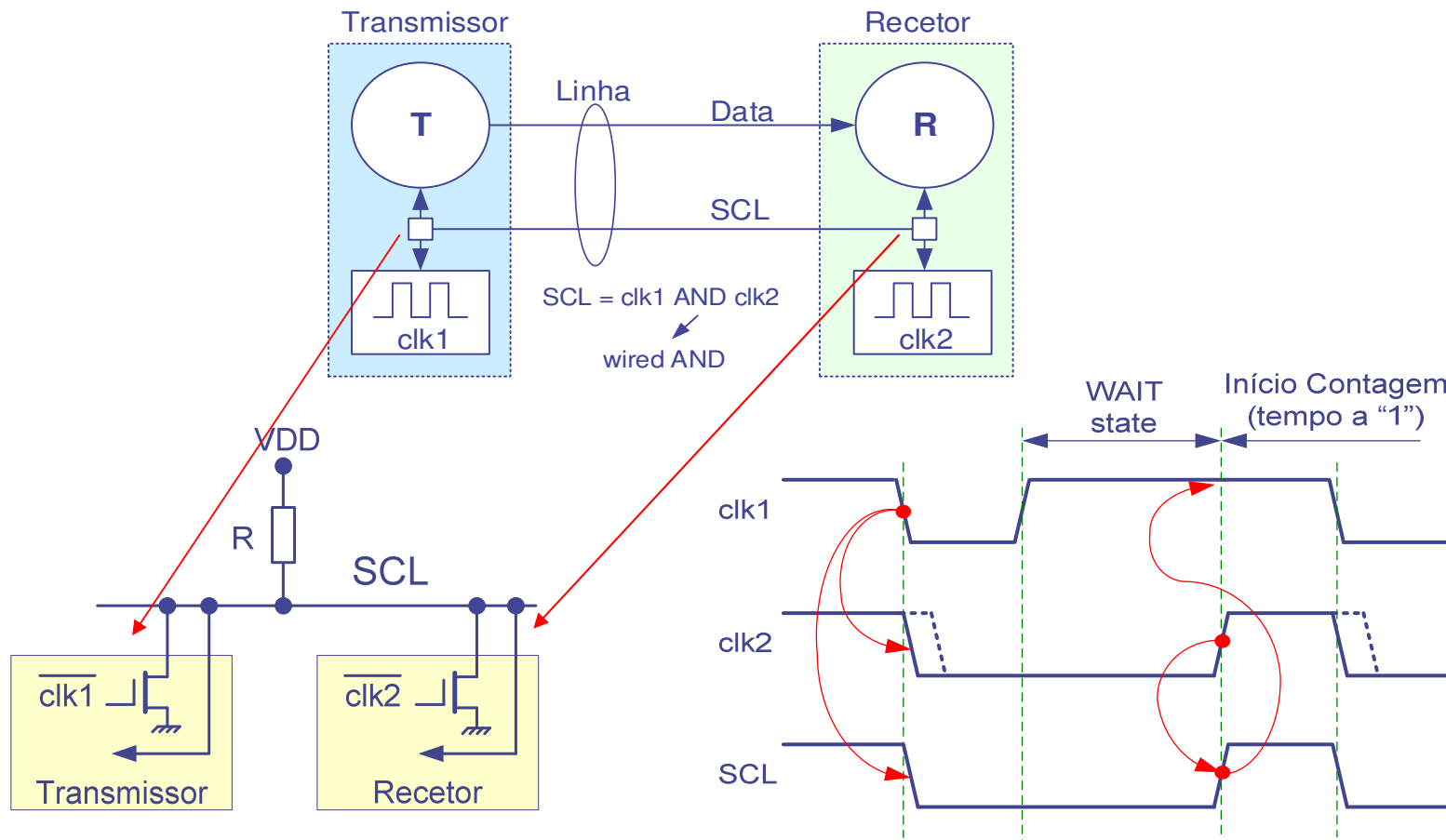
# Sincronização de relógio

- **Relógio explícito do recetor**
- Semelhante ao anterior, sendo o recetor a gerar o sinal de relógio



# Sincronização de relógio

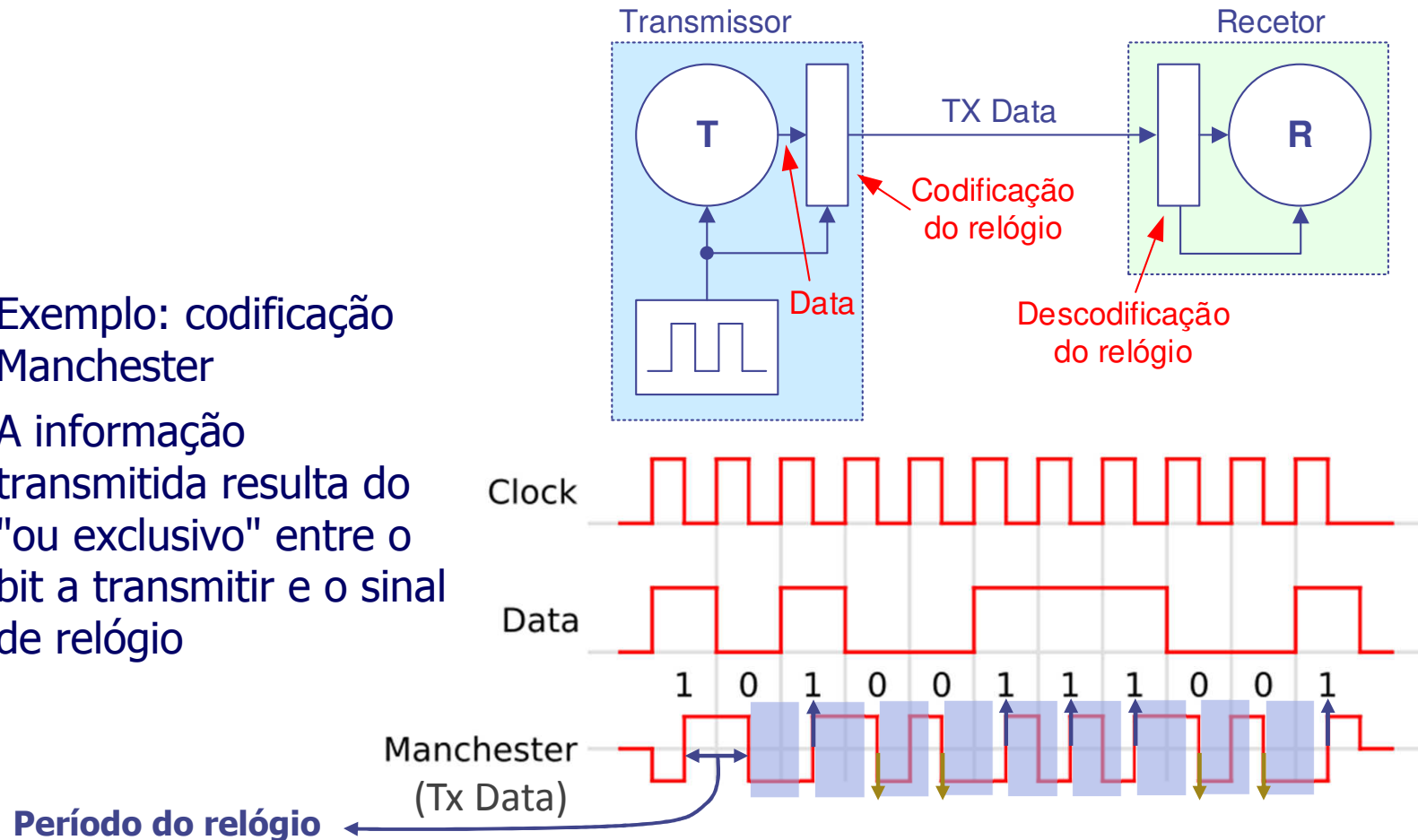
- **Relógio explícito mutuamente-sincronizado ("clock stretching")**
- Transmissor e Recetor sincronizam-se mutuamente



# Sincronização de relógio

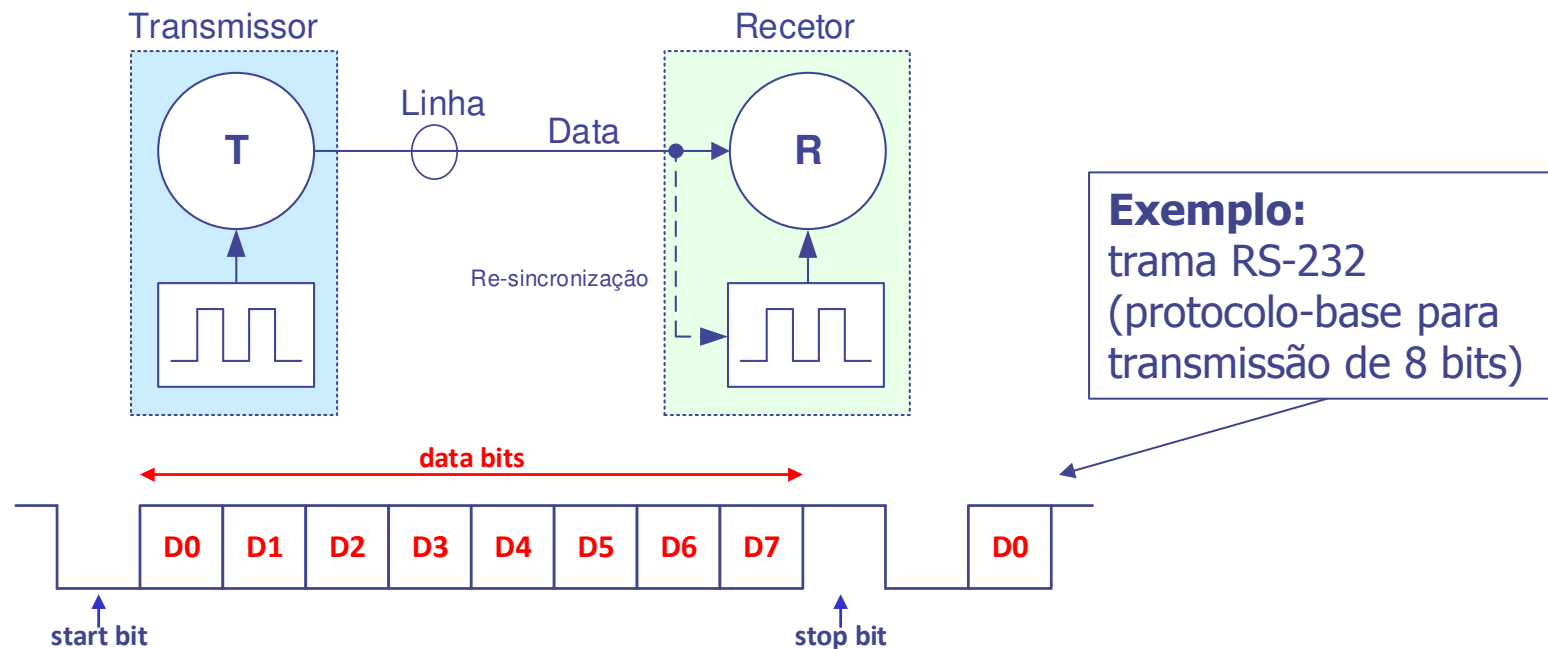
- **Relógio codificado**
- Relógio enviado, em forma codificada, conjuntamente com os dados

- Exemplo: codificação Manchester
- A informação transmitida resulta do "ou exclusivo" entre o bit a transmitir e o sinal de relógio



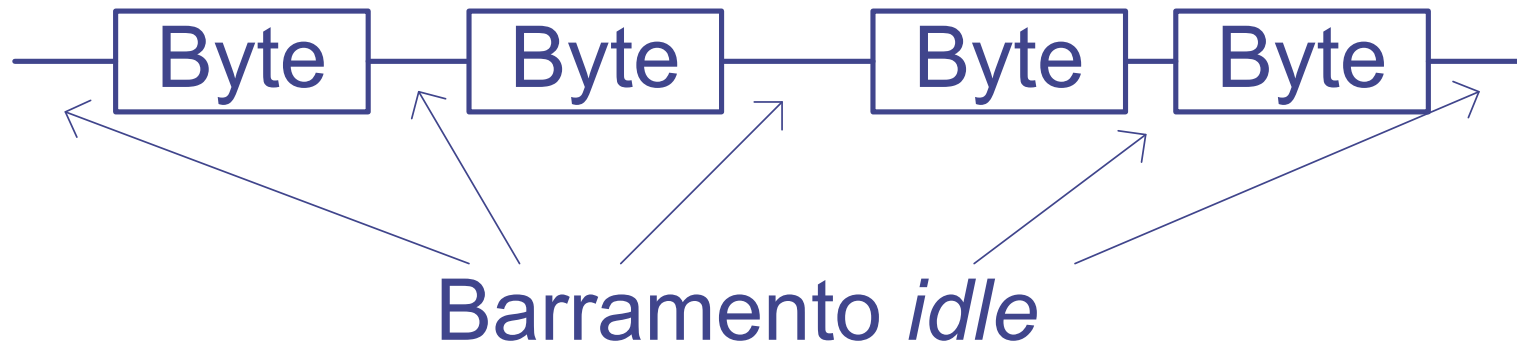
# Sincronização de relógio (transmissão assíncrona)

- **Relógio implícito**
- Os relógios são locais (i.e. não há comunicação do relógio)
- O relógio do recetor é sincronizado ocasionalmente com o do transmissor por meio da receção de símbolos específicos
- Entre instantes de sincronização o desvio dos relógios depende da estabilidade/precisão dos relógios do transmissor e do recetor



## Transmissão de dados – transmissão orientada ao Byte

- O envio de um byte é a operação atômica (indivisível) do barramento
- Cada byte é encarado como independente dos restantes
- Não há restrições temporais para a transmissão em sequência de 2 bytes



- Alguns bytes podem estar reservados para estruturar a informação
- Exemplo de transmissão orientada ao byte: RS232

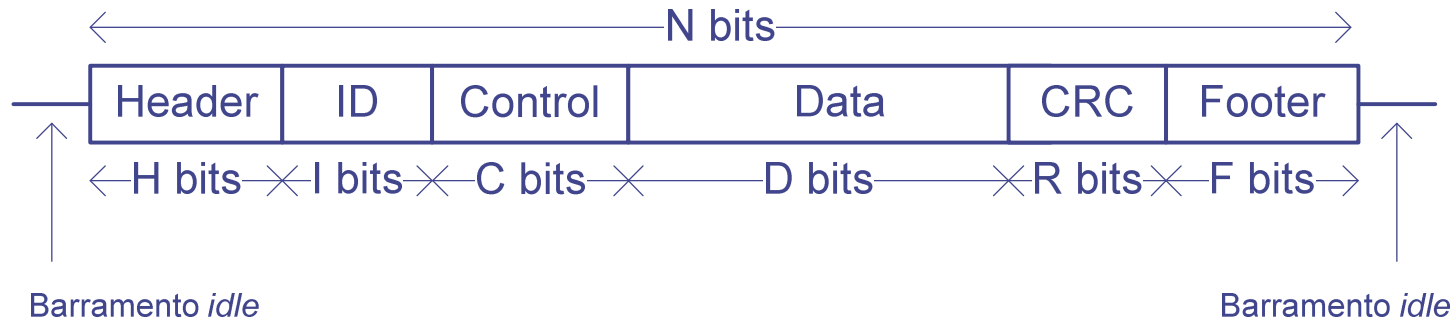


# Transmissão de dados – transmissão orientada ao bit

- A informação é organizada em tramas (sequência de bits intercalada entre duas situações de meio livre)
- As tramas são constituídas por um símbolo de sincronização (delimitador, constituído por 1 ou mais bits) seguido por uma sequência de bits de comprimento arbitrário
- As tramas podem conter campos com diferentes funções:
  - Sincronização: sinalização de início e de fim da trama
  - Arbitragem de acesso ao meio (em barramentos multi-master)
  - Identificação. Diversas formas possíveis:
    - Quem produz
    - Qual o destino
    - Identificação da informação que circula na trama
    - ...
  - Quantidade de informação transmitida
  - Dados
  - Detecção de erros de transmissão

# Transmissão de dados – transmissão orientada ao bit

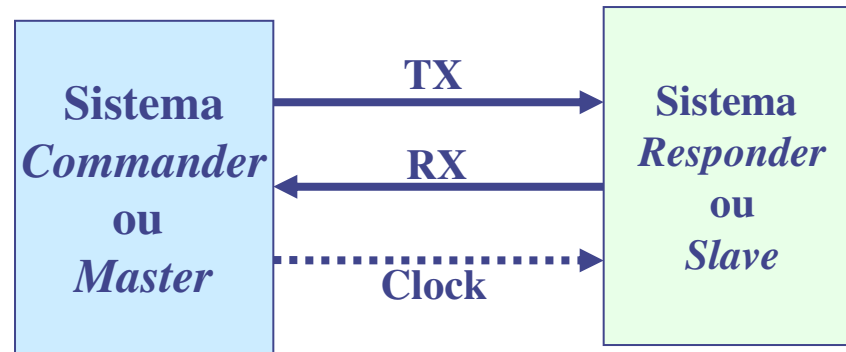
- Exemplo de estrutura de uma trama



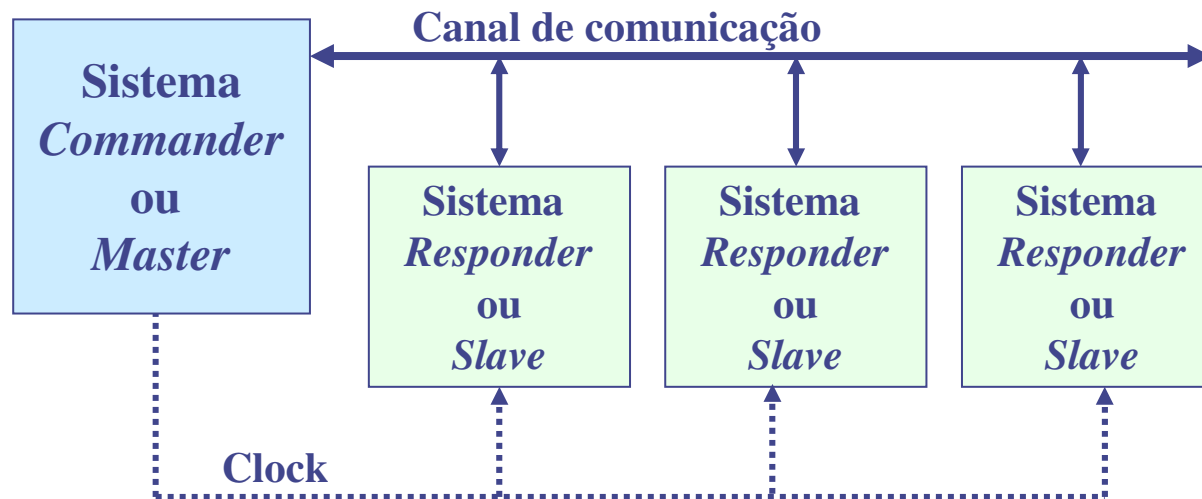
- Exemplo de transmissão orientada ao bit: barramento CAN ("Controller Area Network")
- "Header" e "footer": delimitadores de início e fim de trama
- Data: campo de dados
- CRC ("cyclic redundancy check"): código usado para detetar, no recetor, erros na comunicação
  - Uma forma simples de CRC consiste em somar todos os bytes transmitidos (soma truncada com R bits) – *checksum*
  - O recetor soma todos os bytes recebidos e compara com a soma recebida

# Topologias

- Comunicação ponto a ponto ("half-duplex" ou "full-duplex"):

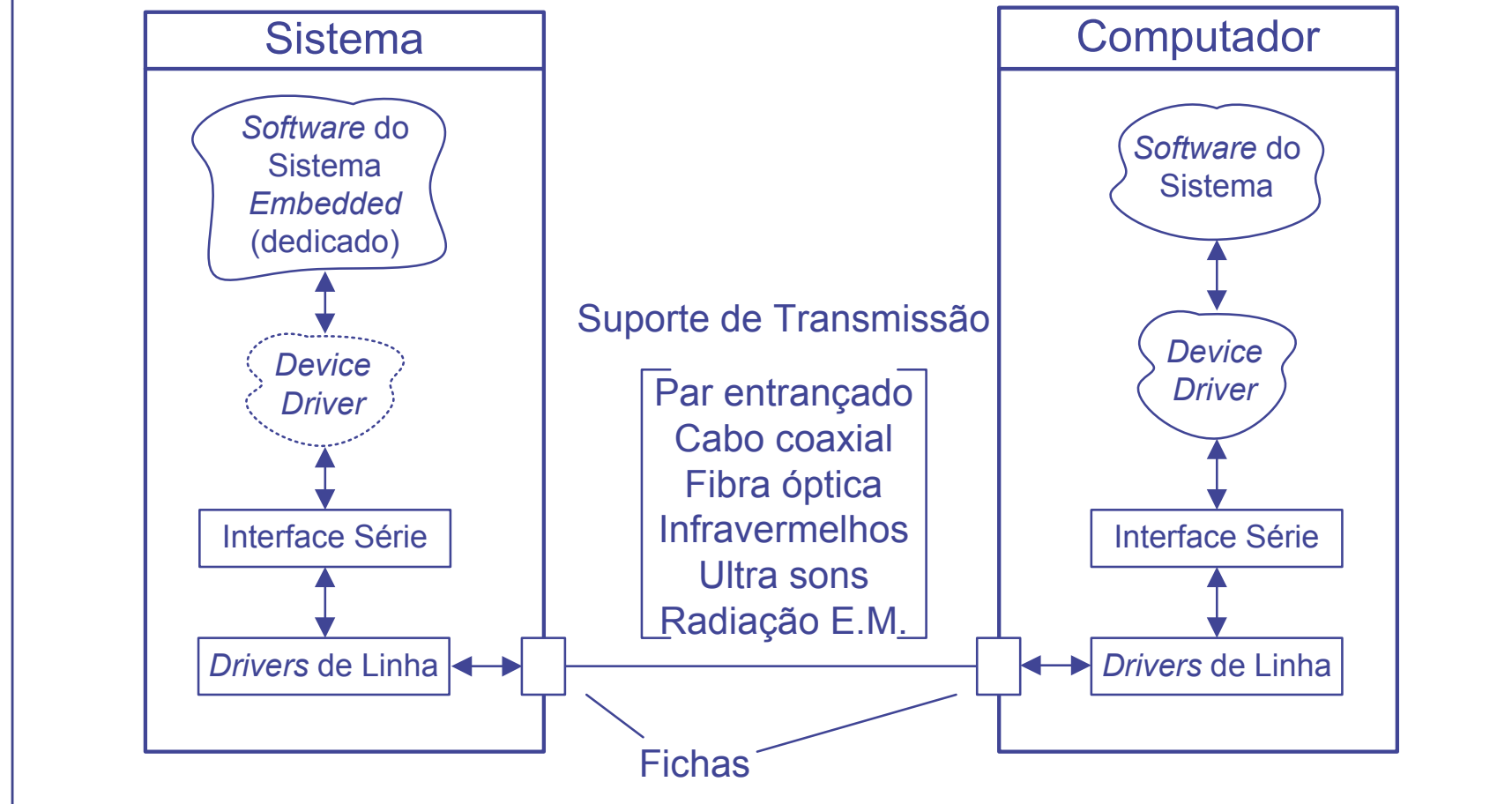


- Comunicação multiponto ("half-duplex"):



# Elementos de uma ligação série

- Exemplo de uma ligação série entre um sistema embutido ("embedded" ou dedicado) e um computador de uso geral (PC)



# Anexos

- Exemplo de modelação em VHDL de:
  - Um transmissor série (slide 7)

# Modelação VHDL do transmissor

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity SerialTx is
    port ( clk      : in  std_logic;
          start     : in  std_logic;
          dataTx    : in  std_logic_vector(7 downto 0);
          sdo       : out std_logic;
          clkOut    : out std_logic);
end SerialTx;

architecture behav of SerialTx is
    signal s_counter : unsigned(3 downto 0) := "0000";
    signal s_dataTx   : std_logic_vector(7 downto 0);
begin

    -- continua
```

# Modelação VHDL do transmissor

```
process(clk)
begin
    if(rising_edge(clk)) then
        if(s_counter = "0000") then
            if(start = '1') then
                sdo <= dataTx(7);
                s_dataTx <= dataTx(6 downto 0) & '0';
                s_counter <= s_counter + 1;
            end if;
        elsif(s_counter /= "1000") then
            sdo <= s_dataTx(7);
            s_dataTx <= s_dataTx(6 downto 0) & '0';
            s_counter <= s_counter + 1;
        else
            s_counter <= (others => '0');
        end if;
    end if;
end process;
clkOut <= clk when s_counter /= "0000" else '1';
end behav;
```