

Aula 13

- Barramentos e interfaces de comunicação série
- 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

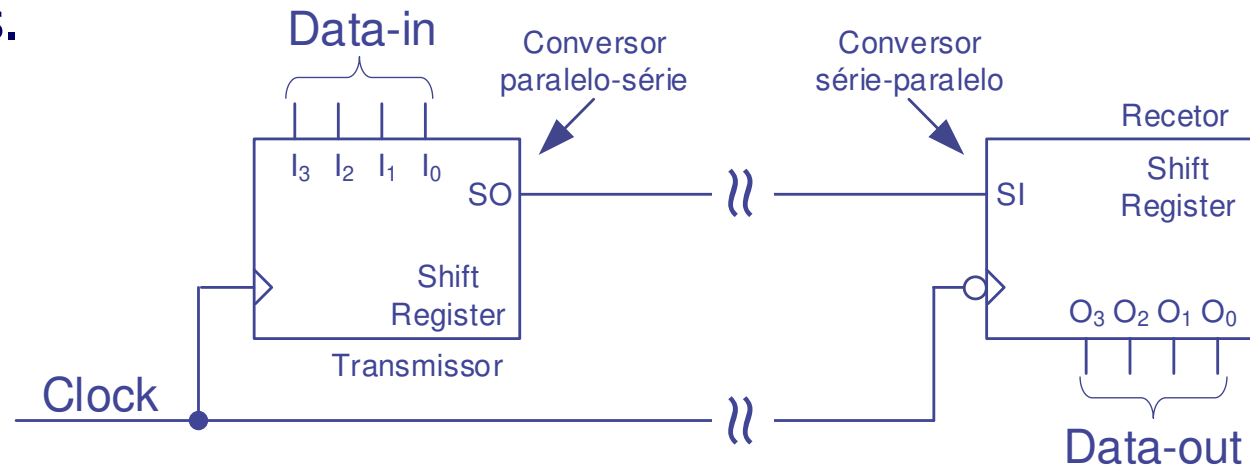
José Luís Azevedo, Arnaldo Oliveira, Tomás Silva, Bernardo Cunha

Introdução

- A transmissão paralela, com relógio comum, a débitos elevados coloca problemas de várias ordens, 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)
- 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

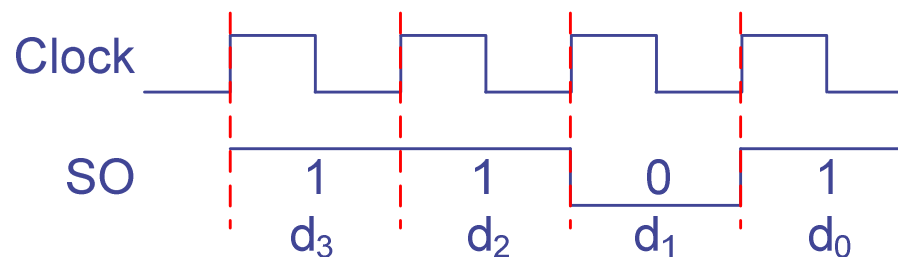
Introdução

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



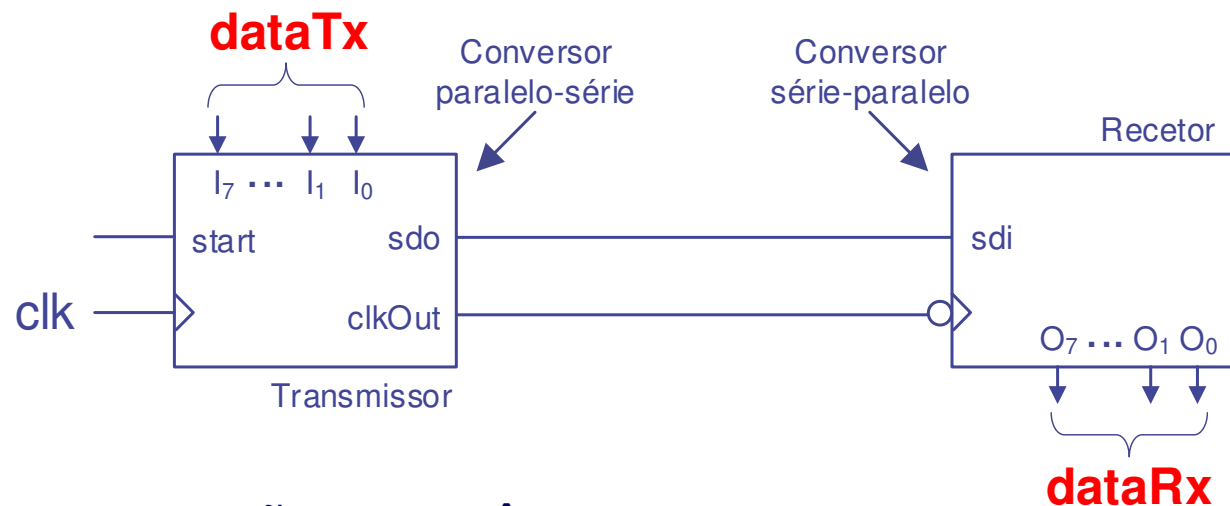
Para transmissão bidirecional podem existir 2 "linhas" separadas, uma para transmissão e outra para receção

- Exemplo
(Data-in = 1101)

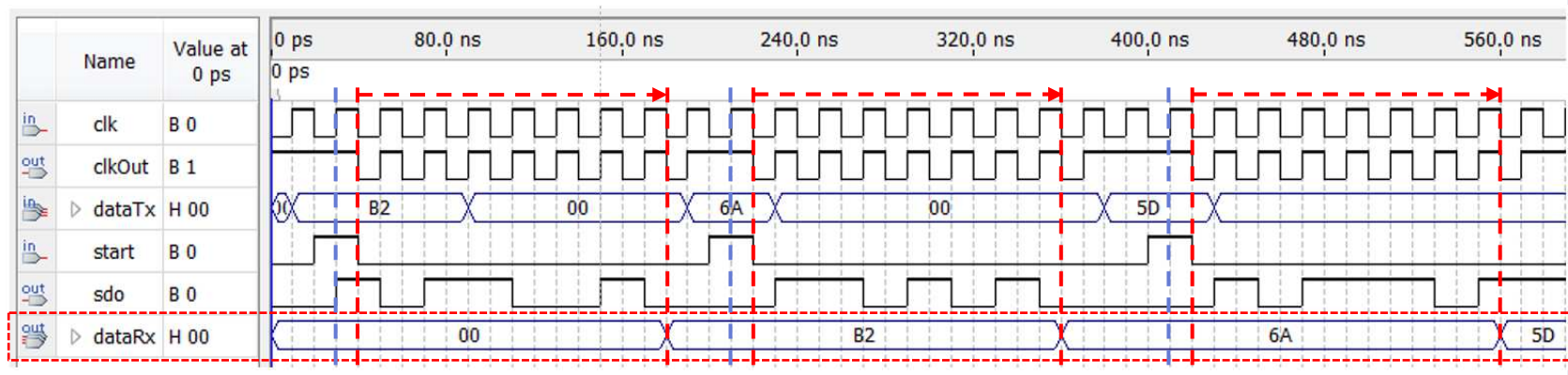


Introdução

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

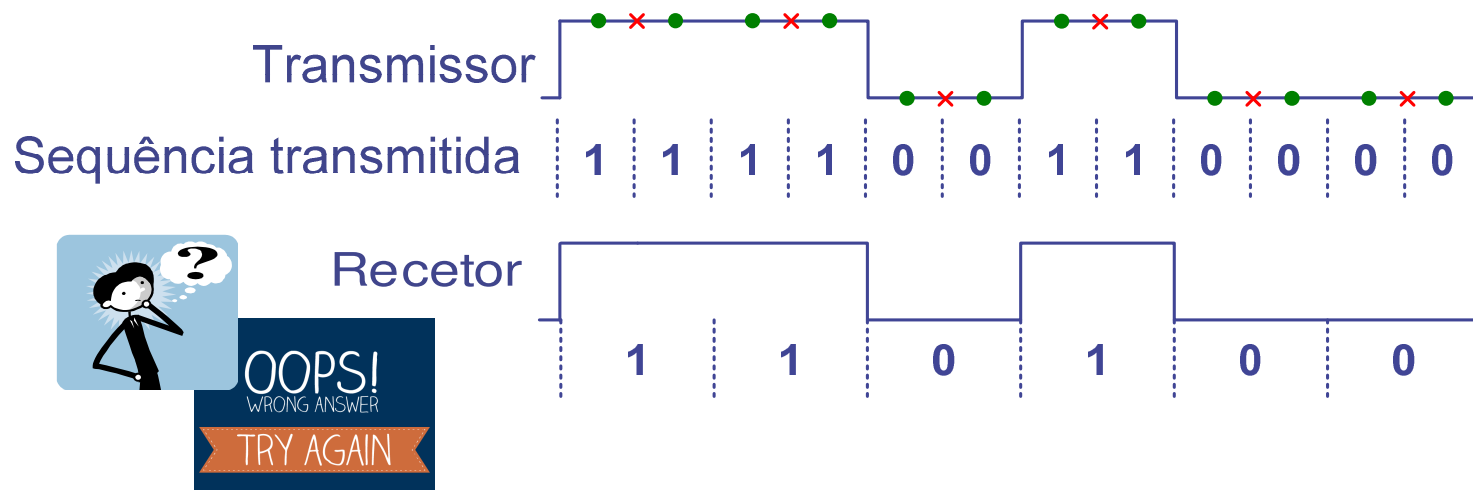


- Exemplo com transmissão da sequência: 0xB2, 0x6A, 0x5D



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**

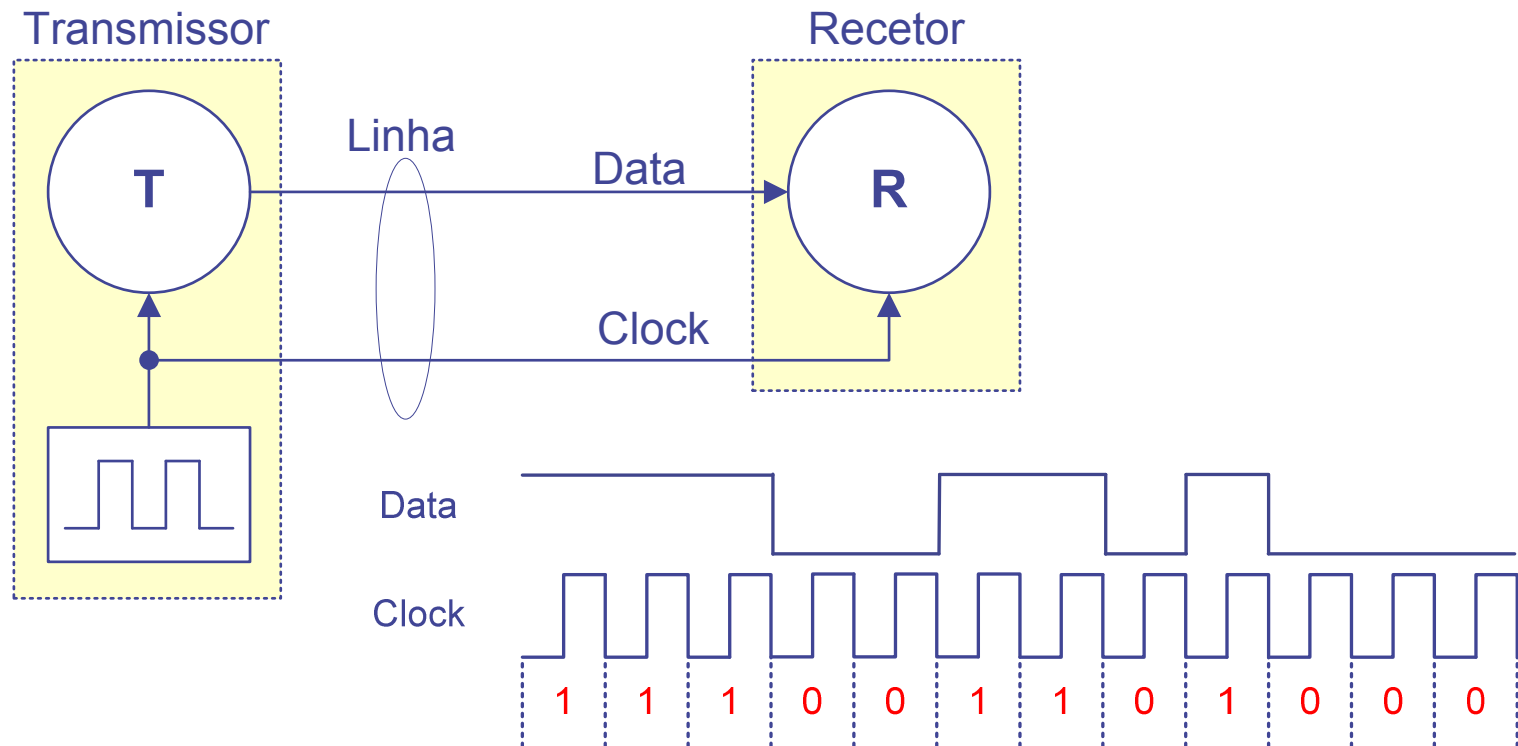
- Não é usado relógio na transmissão, nem há recuperação do relógio na receção
- É necessário acrescentar bits para sinalizar o princípio e o fim da transmissão (e.g. start bit, stop bit), que permitam a sincronização dos relógios do transmissor e do recetor

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

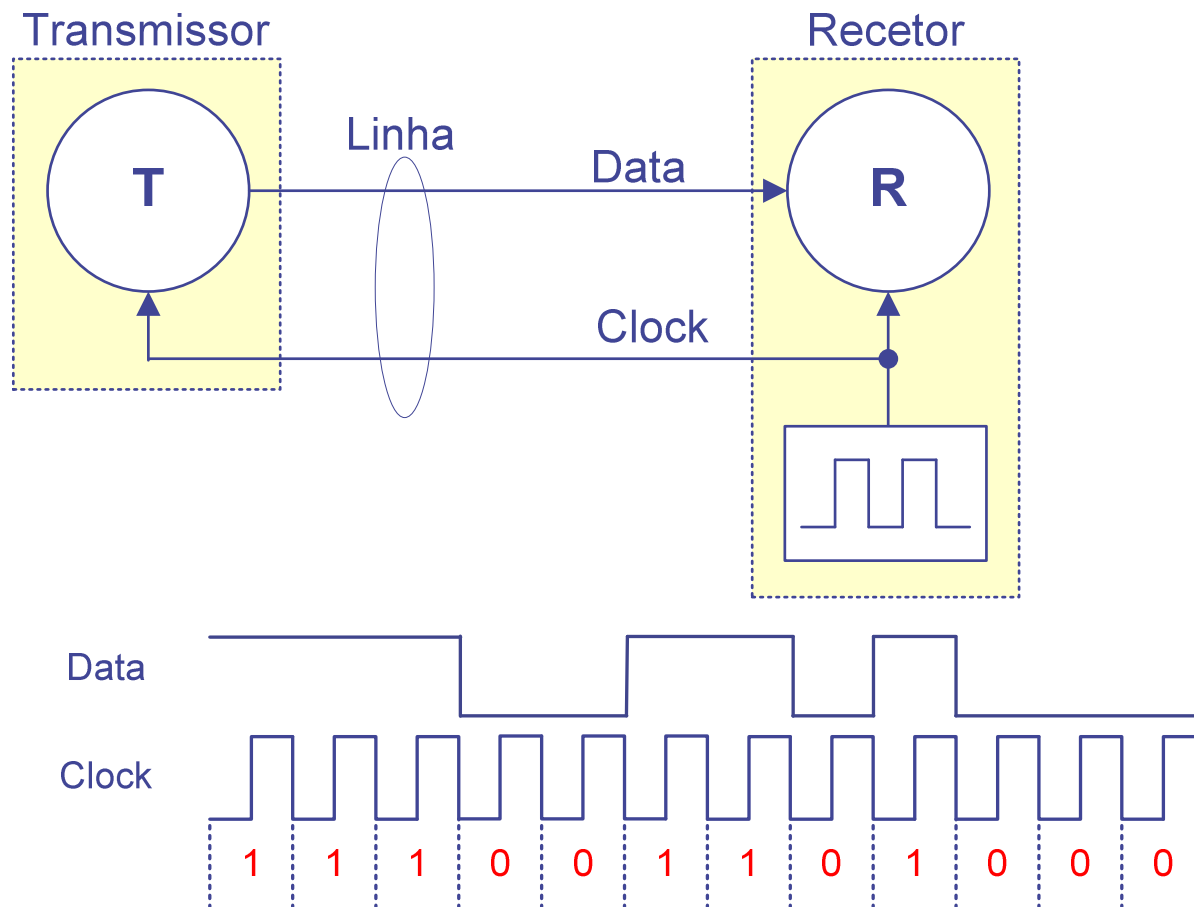
Sincronização de relógio

- **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"



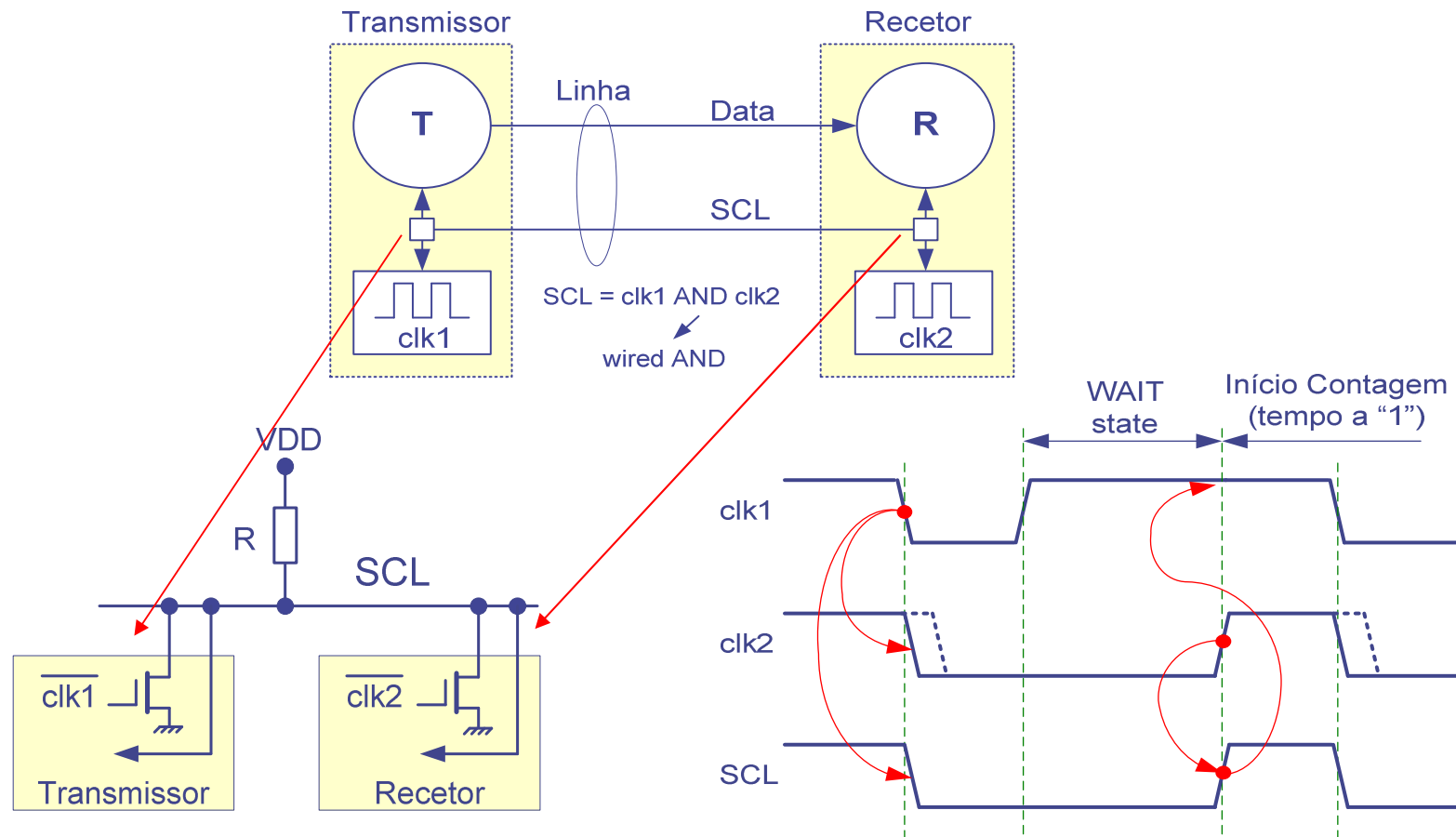
Sincronização de relógio

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



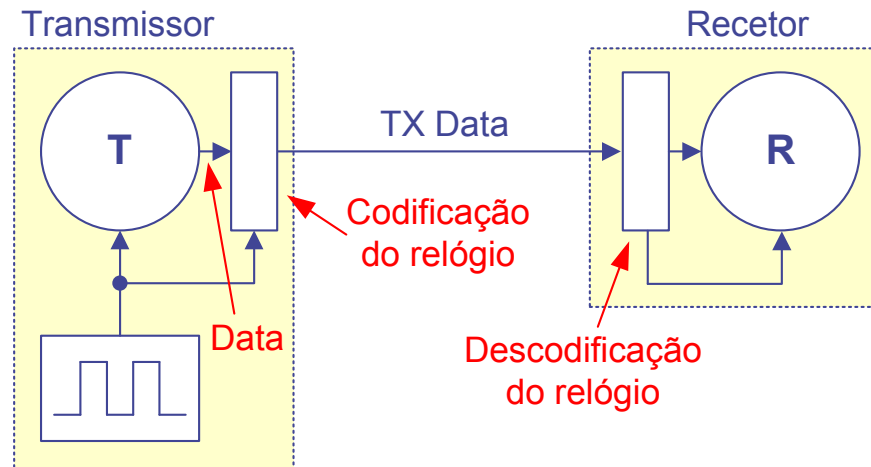
Sincronização de relógio

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

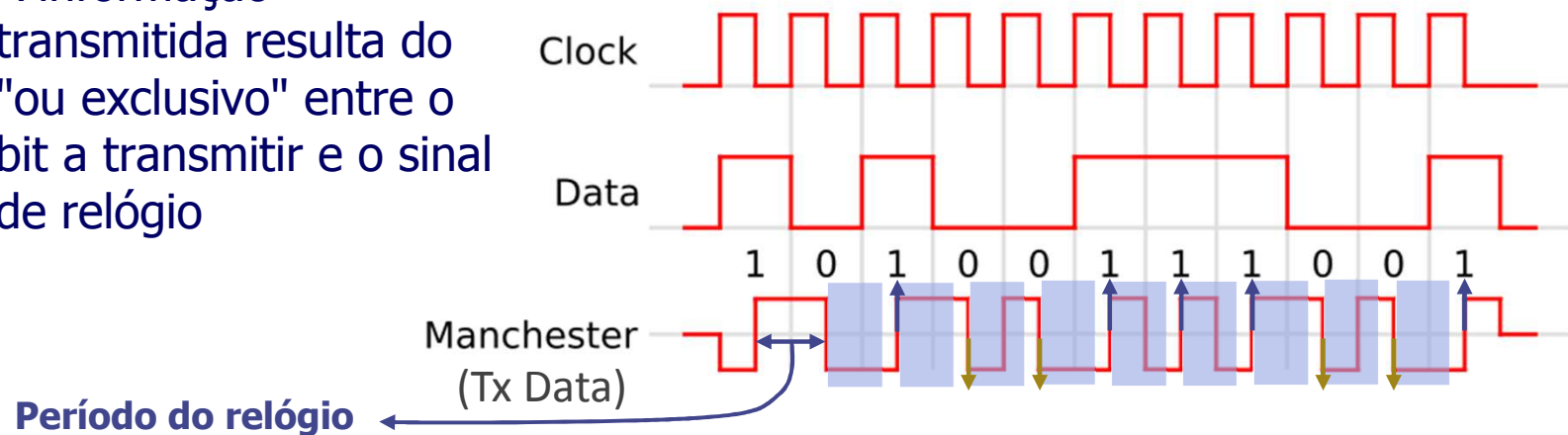


Sincronização de relógio

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



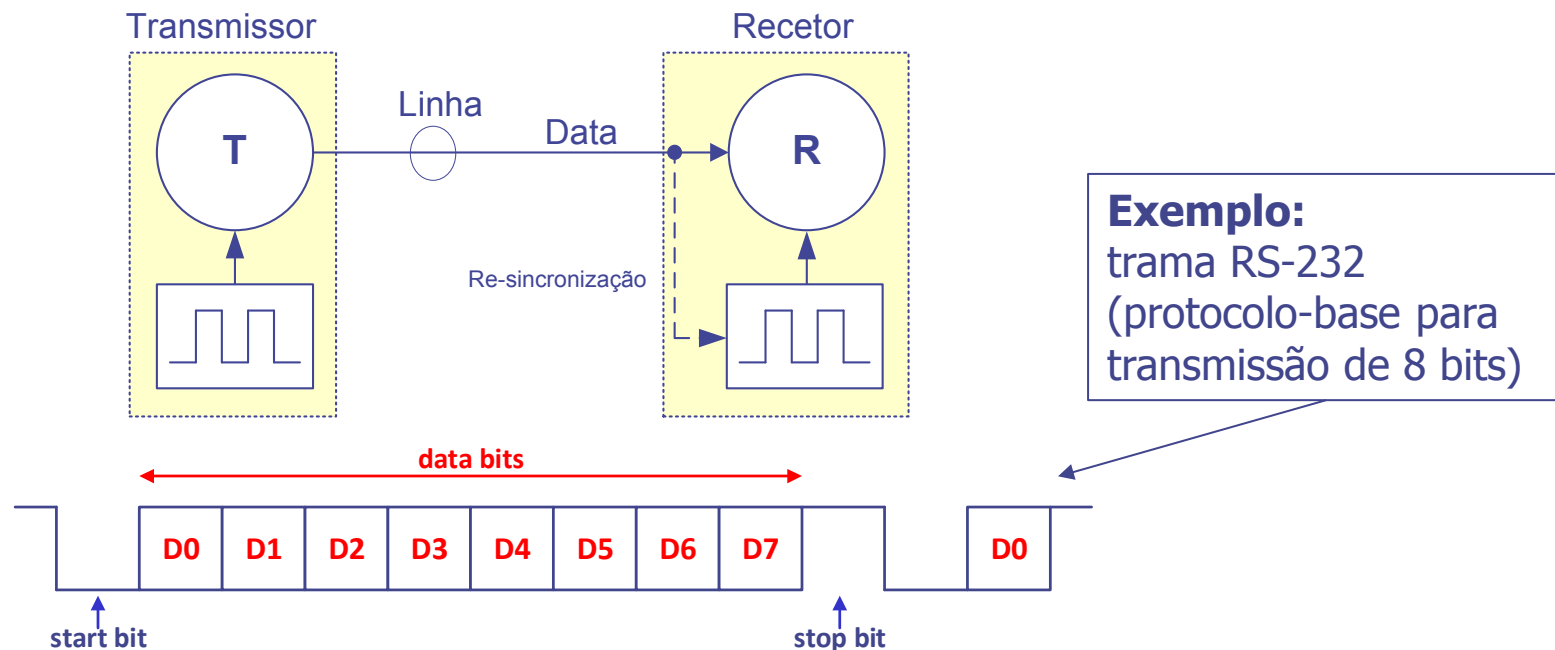
- Exemplo: codificação Manchester
- A informação transmitida resulta do "ou exclusivo" entre o bit a transmitir e o sinal de relógio



Sincronização de relógio

- **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 bit:**

- 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
- Trama: sequência de bits intercalada entre duas situações de meio livre (*idle*, i.e. disponível)

- **Transmissão orientada ao byte:**

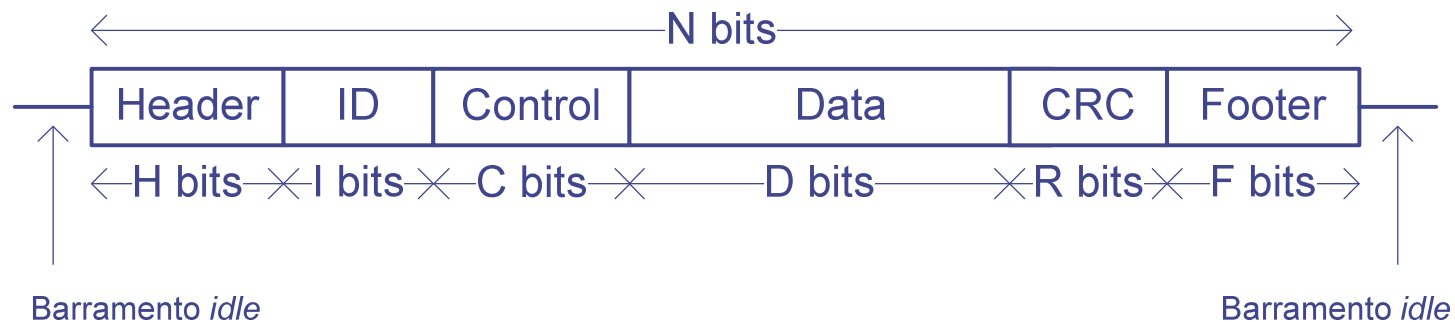
- A transmissão é feita byte a byte
- Existem habitualmente bytes de controlo (reservados) para estruturar e identificar os blocos de informação

Transmissão orientada ao bit

- 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 orientada ao bit

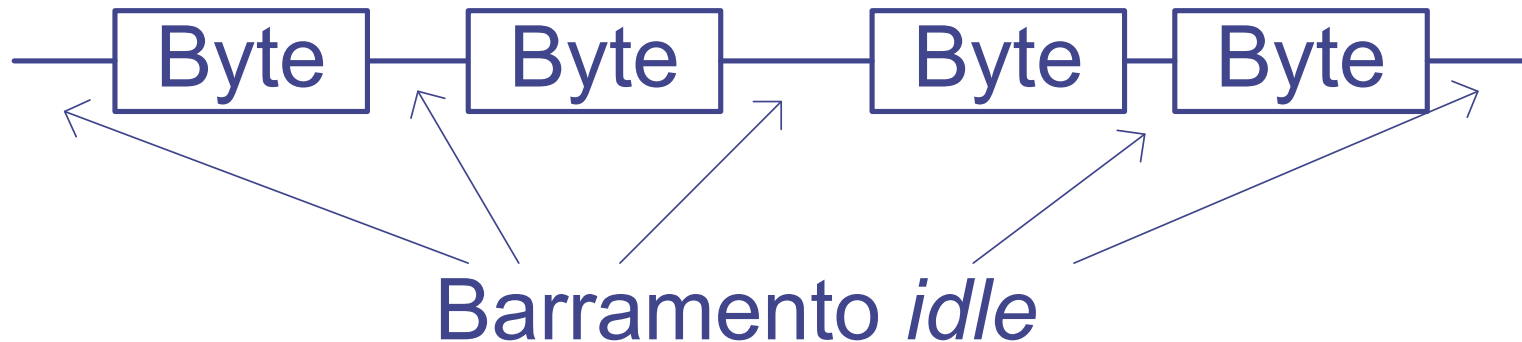
- Exemplo de estrutura de uma trama



- "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
- Exemplo de transmissão orientada ao bit: barramento CAN ("Controller Area Network")

Transmissão orientada ao Byte

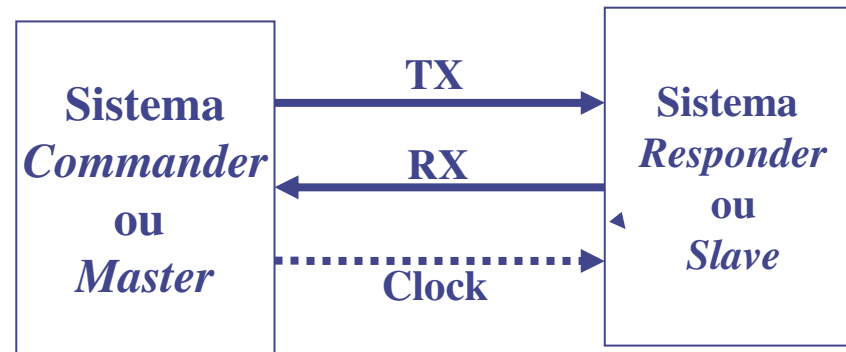
- O envio de um byte é a operação atômica (indivisível) do barramento
- Alguns bytes podem estar reservados para estruturar a informação



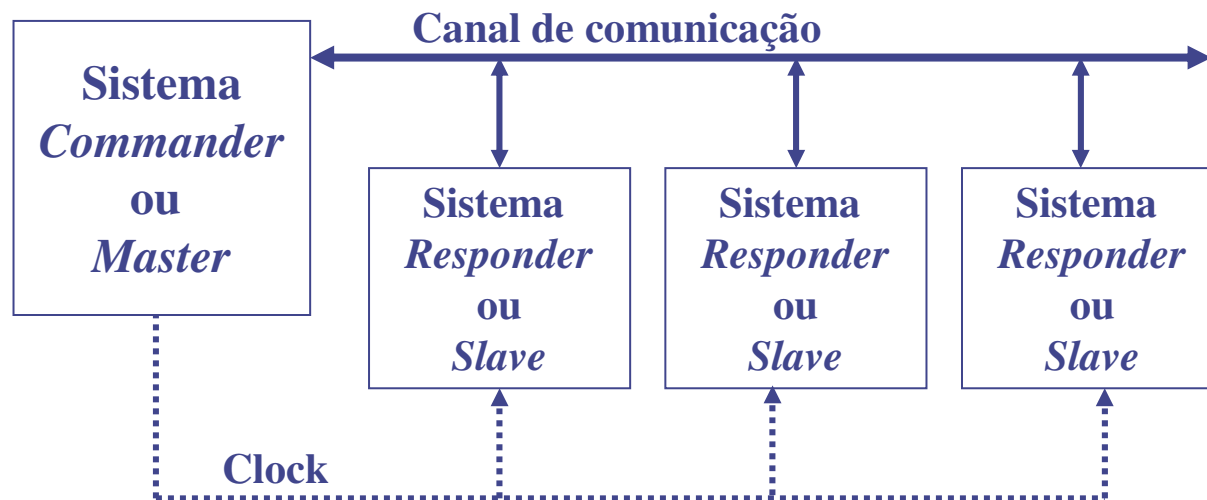
- Exemplo de transmissão orientada ao byte: RS232

Topologias

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

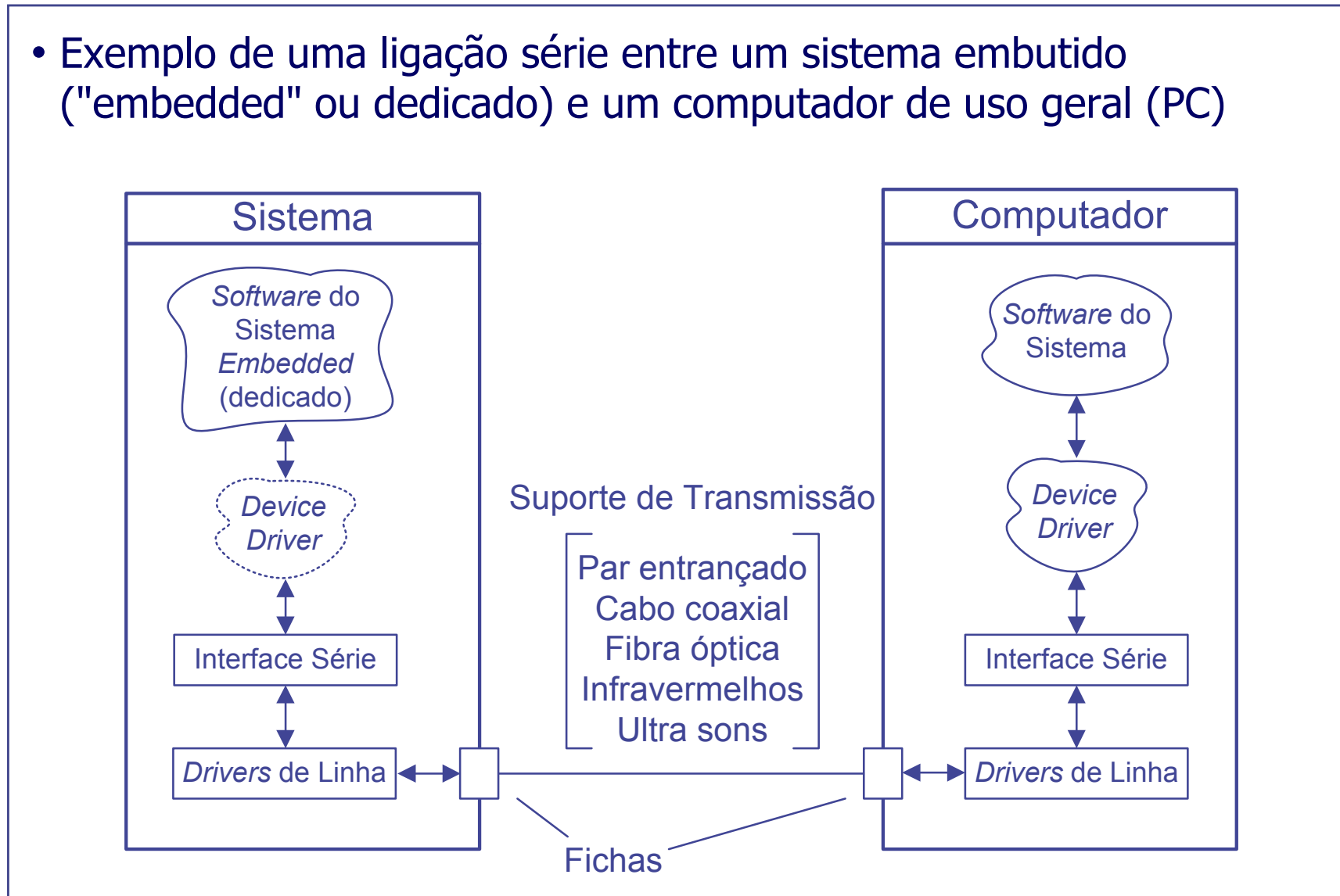


- Comunicação multiponto ("half-duplex"):



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 emissor série (slides 3 e 4)

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;
```