Lecture 11: Serial Data Communication & 8251

Prof. Xiangzhong FANG

xzfang@sjtu.edu.cn

The 80x86 IBM PC and Compatible Computers

Chapter 17

Serial Data Communication and the 8251 Chip

Data Communication

- **Data transmission** is the transfer of data from point-to-point often represented as an electromagnetic signal over a physical communication channel
- A **communication channel** refers to the medium used to convey information from a sender (or transmitter) to a receiver.
  - Examples: copper wires, optical fibbers or wireless communication channels.

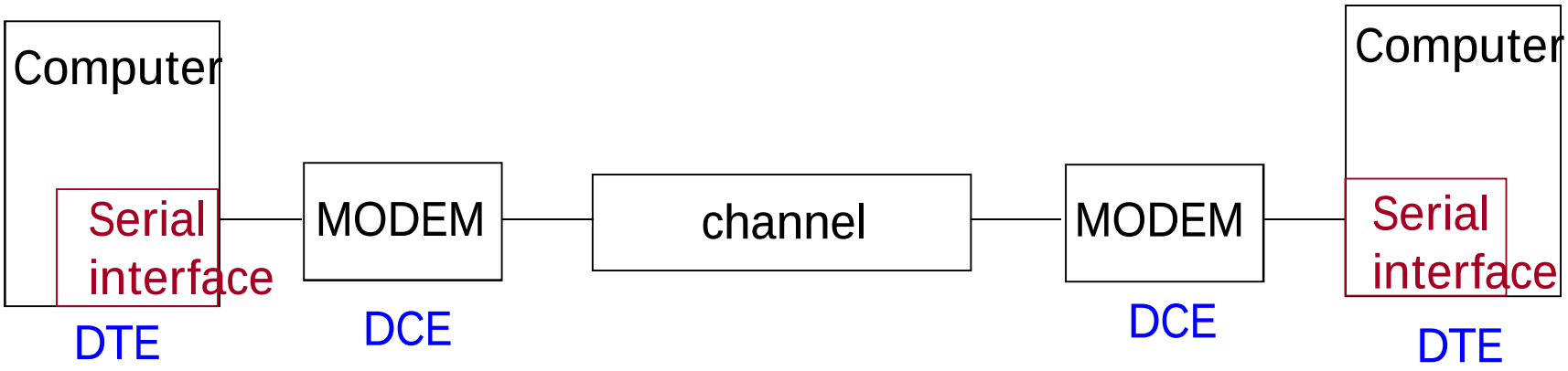
Two Ways: Parallel & Serial

- **Parallel data transfers:**
  - Each bit uses a separate line (wire)
  - Often 8 or more lines are used
  - Control signals in addition
  - Fast & expensive & for short-distance communication
- **Serial data transfers:**
  - One single data line
  - Bits are sent over the line one by one
  - No dedicated lines for control signals
  - Cheap & slow & for long-distance communication

The Whole Picture of Serial Communication

The sender and receiver need a **protocol** to make sense of data:

- e.g., how the data is packed, how many bits constitute a character, when the data begin and end



DTE- Data Terminal Equipment, usually a computer.

DCE- Data Communication Equipment, usually a *modem*.

Serial interface –ICs such as 8251A,16550, and 8250, connecting DTE and DCE.

Serial Communication

- Data transfer rate
- Synchronization methods
- Communication modes
- Error detection
- Modulation and Demodulation

Data Transfer Rate

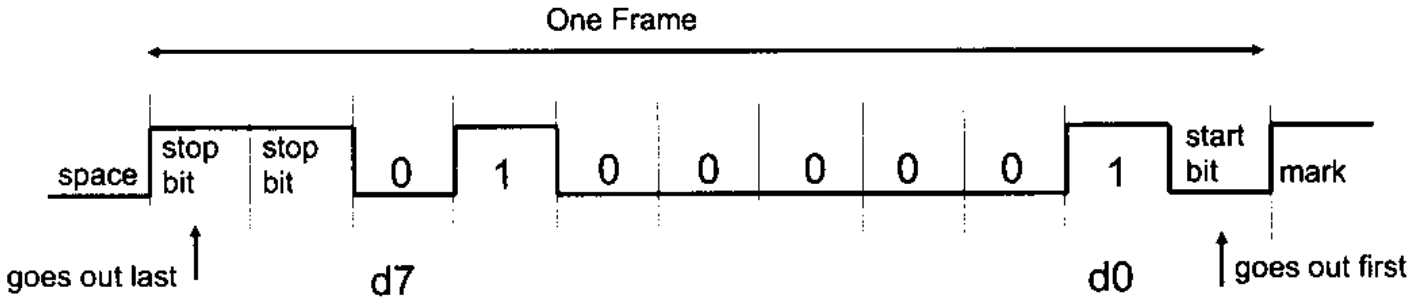
- Symbol rate, a misnomer is *baud rate*
  - The number of distinct symbol or pulse changes (signaling events) made to the transmission medium per second in a digitally modulated signal or a line code, quantified using the baud unit
  - Each symbol can represent one (binary encoded signal) or several bits of data
- Bit rate
  - the number of bits that are conveyed or processed per unit of time, quantified using the bits per second (**bit/s** or **bps**) unit

Synchronization Methods

- Asynchronous serial communication:
  - Transfer one byte at a time
  - The starting of each byte is asynchronous, and therefore each byte needs synchronization between the sender and the receiver using start bit.
- Synchronous serial communication:
  - Transfer a block of data at a time
  - The sender and the receiver are synchronized at the beginning of data transfer using synch characters.

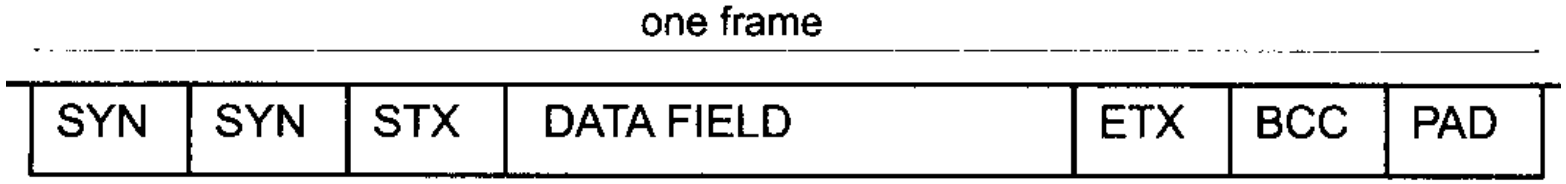
Asynchronous Serial Communication

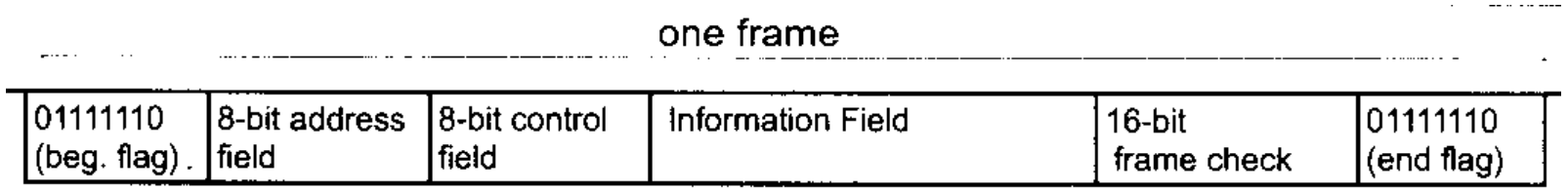
- Framing method:
  - Each character is put in between start and stop bits
  - Start bit: 0 (low)
  - Character: 5, 6, 7, or 8 bits
  - End bit(s): 1 (high)



Synchronous Serial Communication

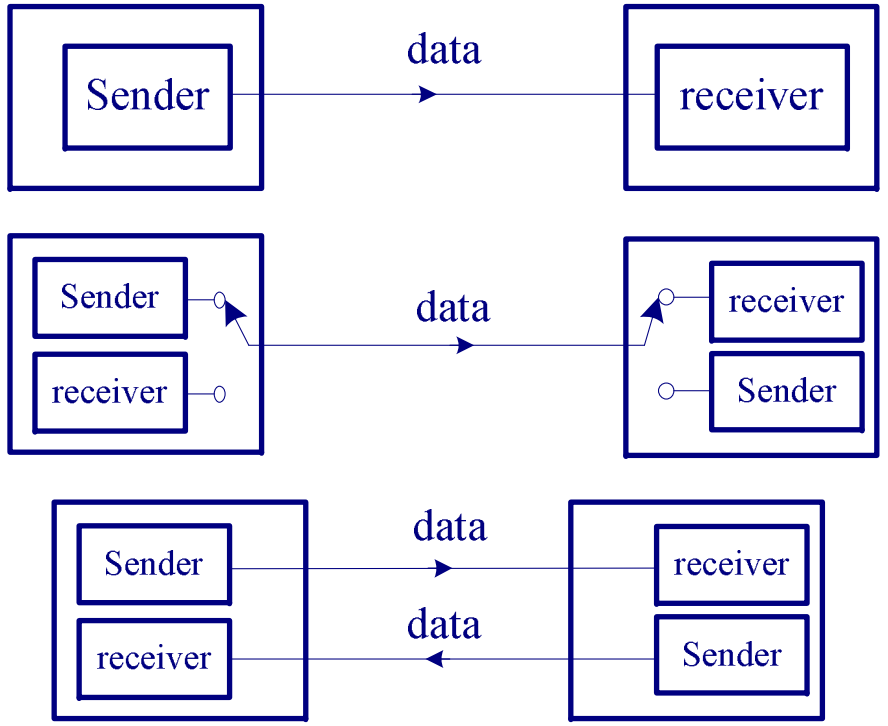
- Framing method, in case of the BISYNC protocol
  - Synch characters
  - STX (start of text) character
  - A block of data
  - ETX (end of text) character





Communication Modes

- Simplex:
  - Only one way
  - E.g., printer
- Half-duplex:
  - Data is transmitted one way at a time
  - E.g., walky-talky
- Full-duplex
  - Data can go both ways at the same time
  - E.g., telephone



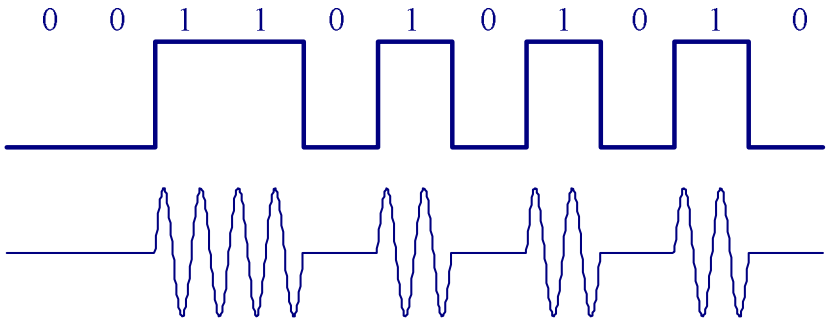
Error Detection

- Parity bit
  - Used in asynchronous serial communication
  - Put an extra parity bit at the end of each character
    - Even-parity: the data and the parity bit has an even number of 1s; odd-parity
- CRC Calculation
  - *k*-bit data, *n*-bit CRC: $\frac{M(X) \times X^n}{G(X)}$
  - Example:
    - Given G(x)= $x^3 + x^2 + 1$ ->1101, (take the coefficients of the polynomial, n=3)
    - If data is 1010110, M(x) * X^n -> 1010110000
    - CRC = 1010110000%1101 (the remainder of binary division, using XOR operation)

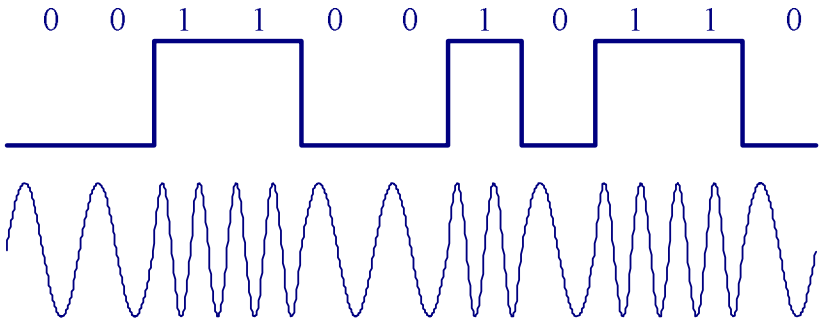
Modulation and Demodulation

- It is not suitable to transmit digital signals directly on a channel for a long distance
  - signal distortion
  - Need to modulate digital signals and get analog signals at the sender, and demodulate the analog signals and get the original digital signals
- Three parameters (Amplitude, frequency, phase) of the carrier can be used for the modulation and demodulation purpose
  - Amplitude-Modulating (AM)
  - Frequency-Shift Keying (FSK)
  - Phase-Shift Keying (PSK)

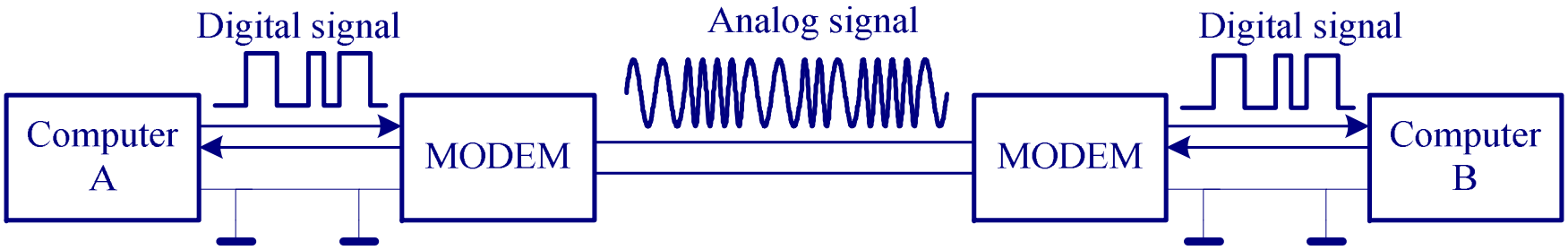
Modulation and Demodulation



1 and 0 are represented using different amplitude



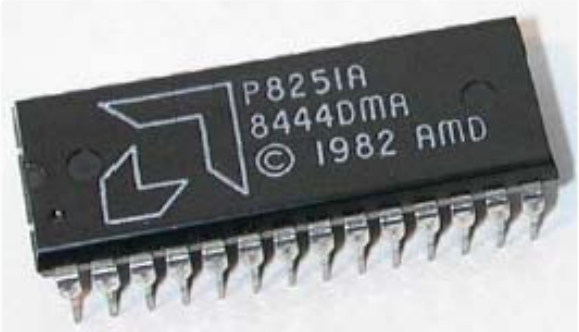
1 and 0 are represented using different frequency

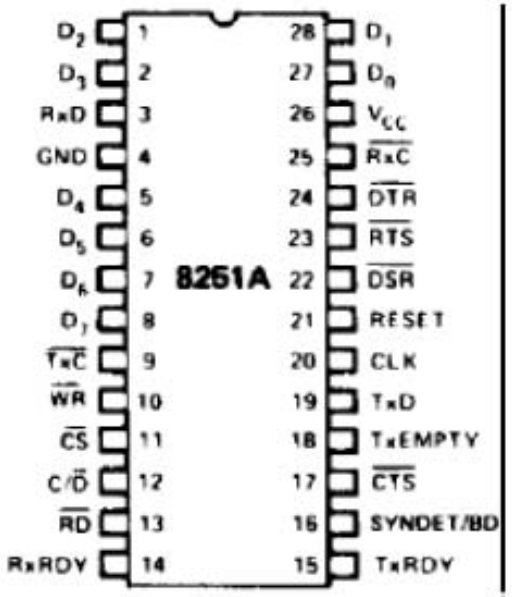


8251 USART Chip

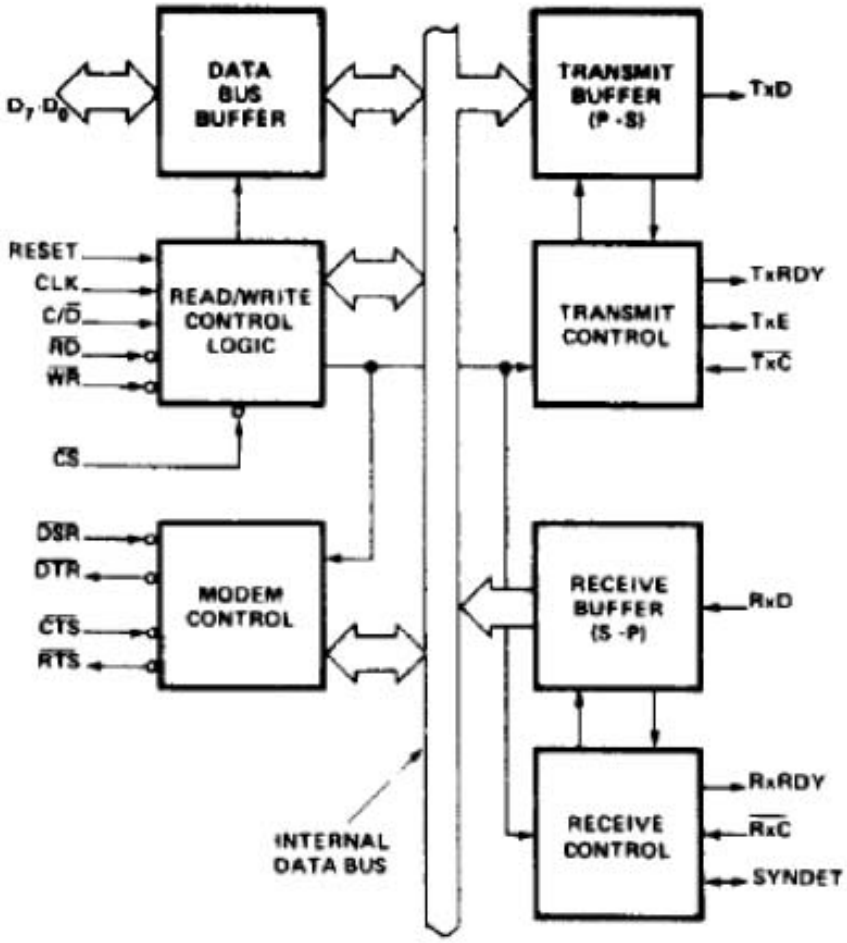
- Capable of doing both asynchronous and synchronous data communication
  - synchronous: baud rate 0-64K, characters can be 5, 6, 7, or 8 bits, automatically detect or insert sync characters
  - Asynchronous: baud rate 0-19.2K, characters can be 5, 6, 7, or 8 bits, automatically insert start, stop and parity bits, TxC and RxC clocks can be 1, 16, or 64 times of the baud rate
- Full duplex, double-buffered
- Error checking circuit

8251



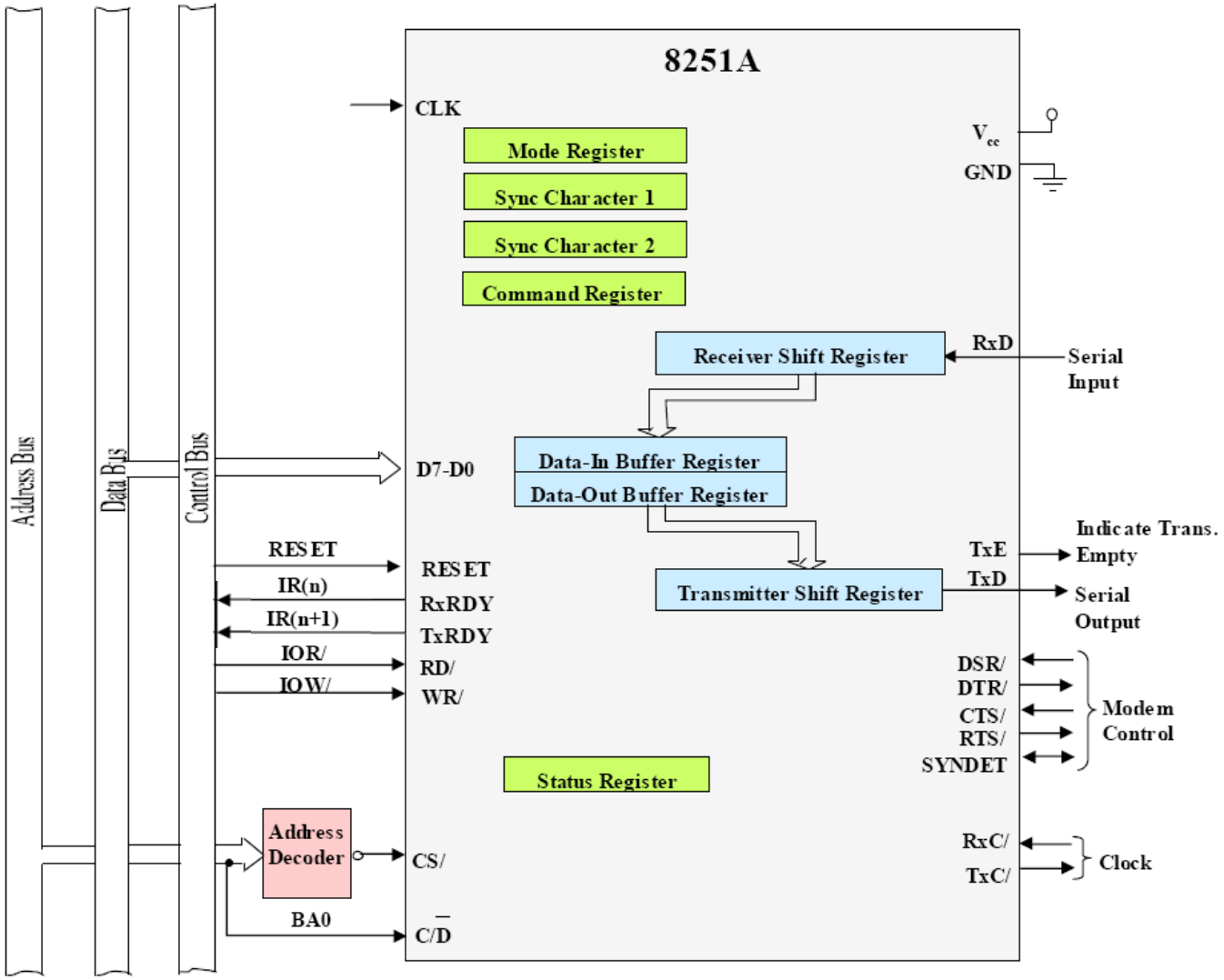


Pin Configuration

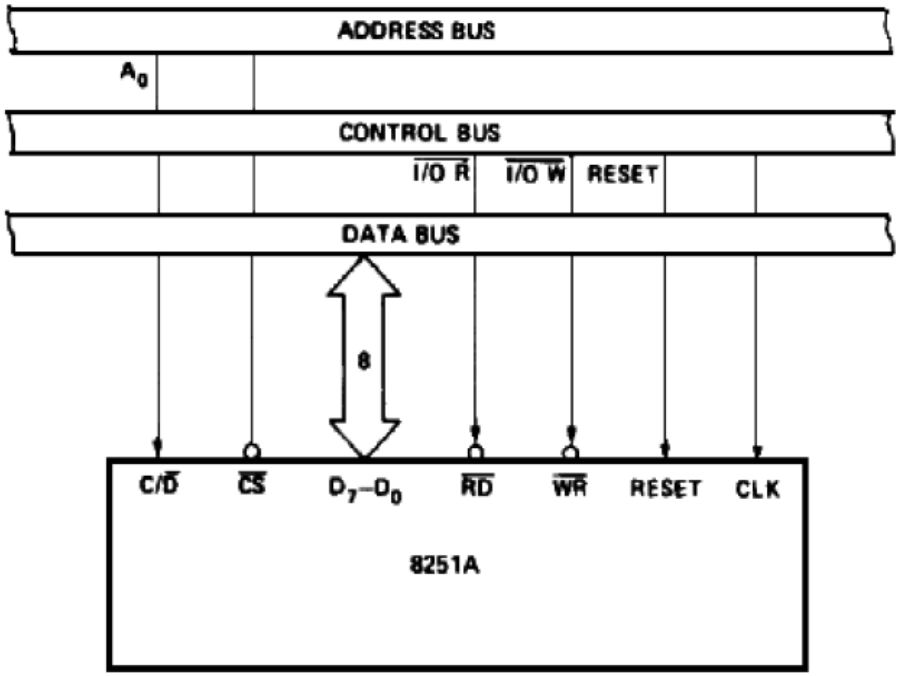


Block Diagram

8251 Communication Interface



Interfacing to 8088



C/$\overline{D}$	RD/	WR/	CS/	Function
0	0	1	0	8251A DATA → DATA BUS
0	1	0	0	DATA BUS → 8251A DATA
1	0	1	0	STATUS → DATA BUS
1	1	0	0	DATA BUS → Control
X	1	1	0	DATA BUS → 3-STATE
X	X	X	1	DATA BUS → 3-STATE

8251 Signals

The 8251A is **doubled-buffered**. This means that one character can be loaded into a **data-out buffer register** while another character is being shifted out of the actual **transmit shift register**.

The **TxRDY** output of the 8251A will go high when:

- The **data-out buffer register** is Empty for another character from the CPU.
- The **CTS/** input has been asserted low.
- The **transmit-enable** (**TxEN**) bit of the 8251A's command word is set.

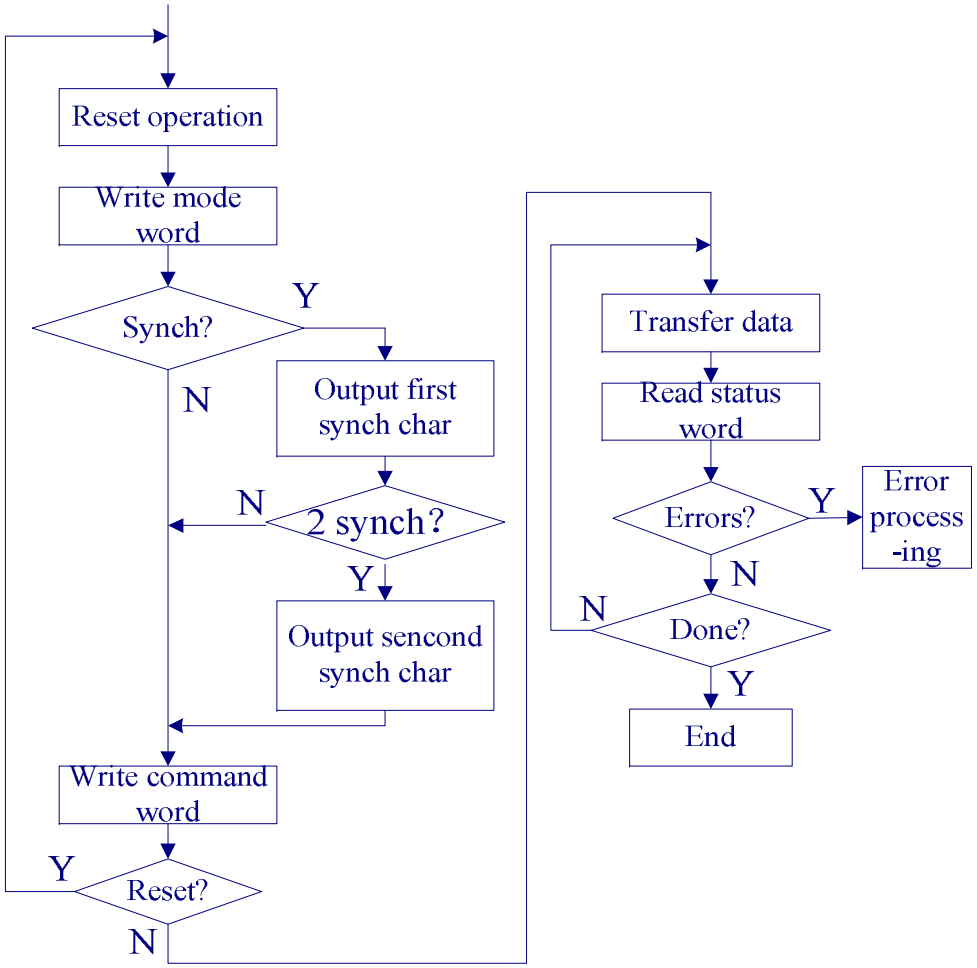
The **TxEMPTY** output of the 8251A will go high when both the **data-out buffer register** and the **transmit shift register** are empty.

The **RxRDY** output of the 8251A will go high when:

- The **data-in buffer register** is full and is ready to be read by the CPU.
- The **receive-enable** (**RxE**) bit of the 8251A's command word is set.

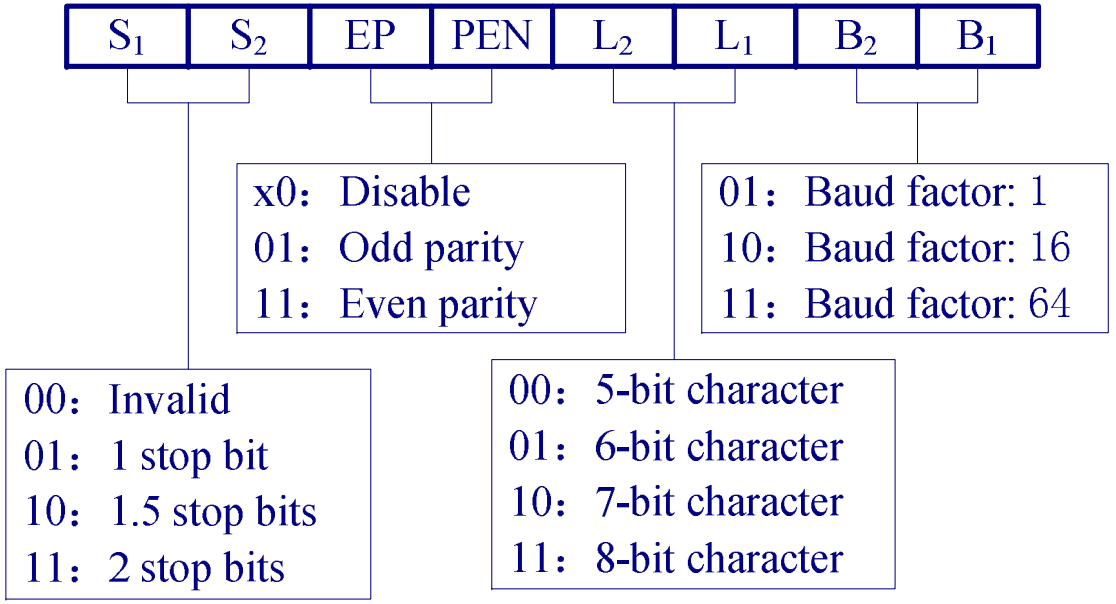
If the CPU does not read a character from the **data-in buffer register** before another character is shifted in, the first character will be overwritten and lost.

Using 8251

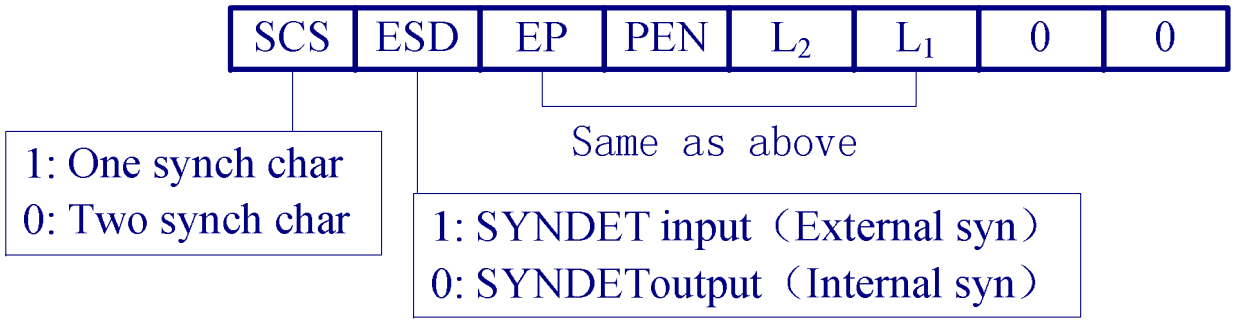

Reset operation
Write mode word
Synch?
Y
N
Output first synch char
2 synch?
N
Y
Output sencond synch char
Write command word
Reset?
Y
N
Transfer data
Read status word
Errors?
Y
Error process -ing
N
Done?
N
Y
End

8251 Mode Word

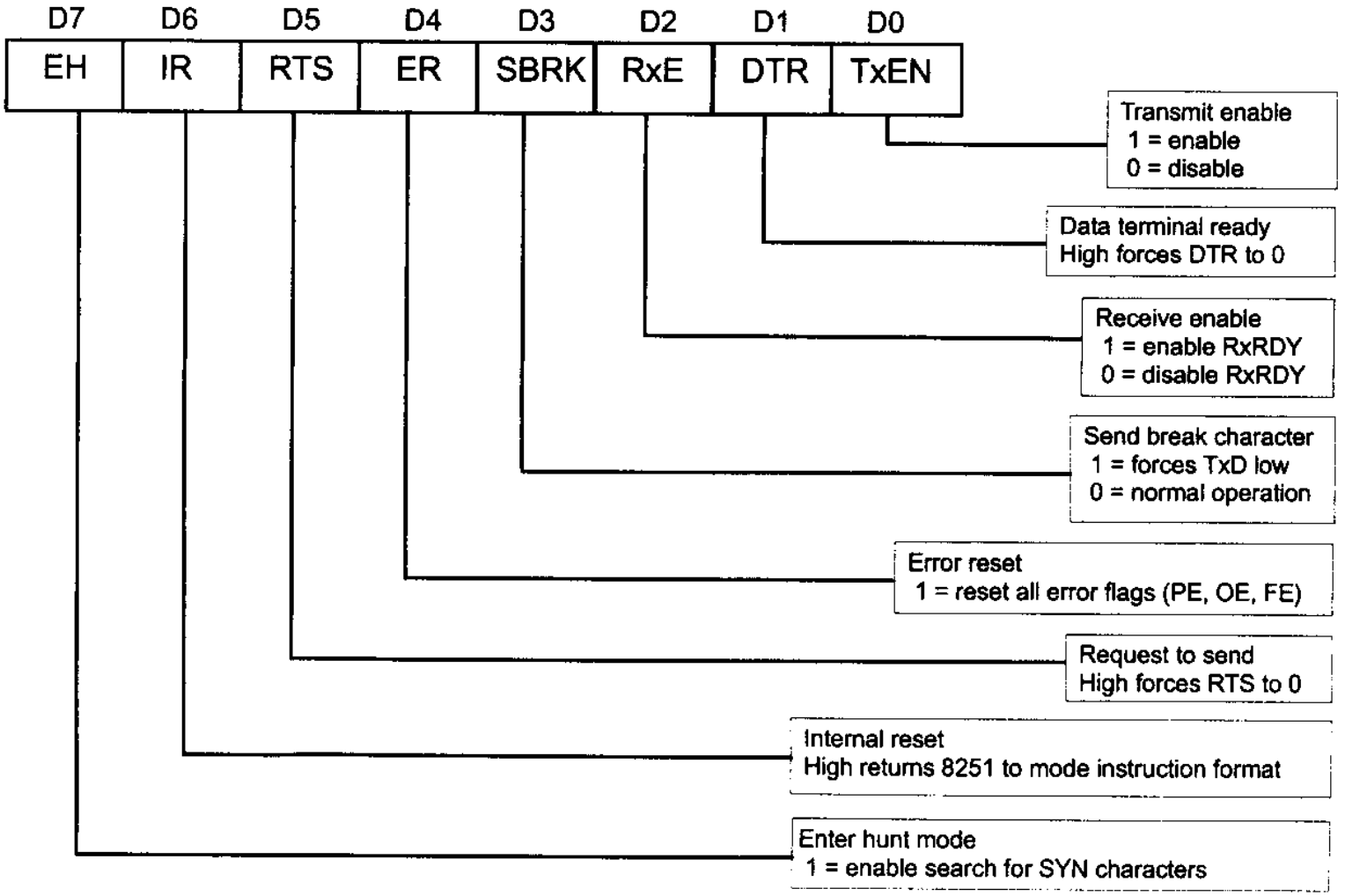
Asynchronous



Synchronous



8251 Command Word



8251 Command Word

Initializing the **TxEN** bit to 1 will enable the transmitter section of the 8251A and the **TxRDY** output.

Initializing **DTR/** bit to 1 will cause the DTR/ output of the 8251A to be asserted low. This signal is used to tell a modem that a PC or terminal is operational.

Initializing **RxE** bit to 1 will enable the RxRDY output of the 8251A.

Initializing **SBRK** bit to 1 will cause the 8251A to output characters of 0's including start bits, data bits, and parity bits (**break character**). A break character is used to indicate the end of block of transmitted data.

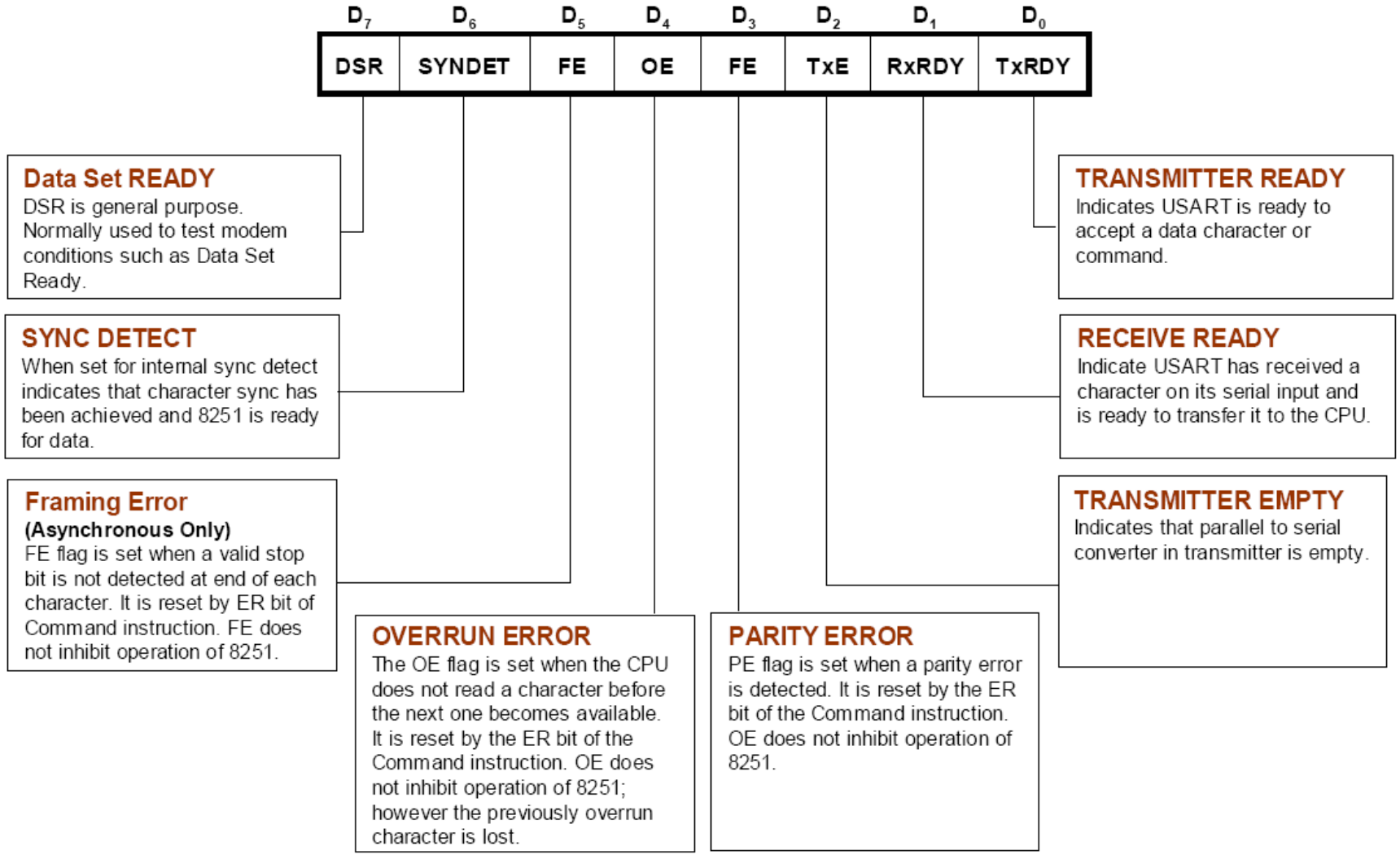
Initializing **ER** bit to 1 will cause the 8251A to reset the **parity**, **overrun**, and **framing** error flags in the 8251A status register.

Initializing **RTS** bit to 1 will cause the 8251A to assert its **request-to-send** (**RTS/**) output low. This signal is sent to a modem to ask whether a modem and the receiving system are ready for a data character to be sent.

Initializing **IR** bit to 1 will cause 8251A to be internally reset. After the software- reset command, a new mode word must be sent.

Initializing **EH** bit to 1 will cause 8251A to enter hunt mode (search for **SYN** characters, and is used only in synchronous mode.

8251 Status Word



Initializing 8251

When power is first applied, the 8251A may come up in the mode, SYN character or command format.

It is safest to execute the worst-case initialization sequence (**SYNC** mode with two **SYN** characters). Loading three 00H consecutively into the device with C/#D = 1.

An **internal reset command** (40H) may then be issued to return the device to **mode word**.

The **mode word** must then be issued, and followed by the **command word**.

8251 Programming Example

- Use 8251 to transfer 256 characters in asynchronous mode, assuming that the port addresses are 208H and 209H, the baud factor is 16, and 1 stop bit, 1 start bit, no parity bit, and 8-bit character are used.

Solution: Sender side: data is stored in Buf1

```
LEA DI, Buf1
MOV DX,209H
MOV AL,00H          ;worse-case init.
OUT DX , AL
CALL DELAY
MOV AL,00H          ;
OUT DX , AL
CALL DELAY
MOV AL,00H          ;
OUT DX , AL
CALL DELAY
MOV AL,40H          ;reset command
OUT DX , AL
```

```
MOV AL , 01001110B      ; mode word
MOV DX , AL
MOV AL , 00110111B      ; command word
OUT DX , AL
MOV CX , 256            ; to send 256 char.
NEXT :  MOV DX, 209H
IN AL , DX              ; status word
AND AL , 01H            ; TxRDY?
JZ NEXT
MOV AL , [DI]
MOV DX , 208H           ; data register 208H
OUT DX , AL             ; send the char.
INC DI
LOOP NEXT
```

8251 Programming Example

Receiver side: data will be stored in Buf2

```
Data segment
buf2 DB 256 dup(?)
Data ends
        ¦

        MOV DX,209H
        MOV AL,00H
        OUT DX , AL
        CALL DELAY
        MOV AL,00H
        OUT DX , AL
        CALL DELAY
        MOV AL,00H
        OUT DX , AL
        CALL DELAY
        MOV AL,40H          ; reset
        OUT DX , AL
        MOV AL , 01001110B    ; mode word
        OUT DX , AL
        MOV AL , 00110111B    ; command word
        OUT DX , AL
        MOV CX , 256          ; to receive 256 char.
        MOV SI , 0
NEXT:   MOV DX , 209H
        IN AL , DX                  ; status word
        AND AL , 02H                ; RXRDY ?
        JZ NEXT
        MOV DX , 208H
        IN AL , DX                  ; receive a char
        MOV buf2[SI] , AL
        INC SI
        LOOP NEXT
```