SERIAL COMMUNICATION

UNIT III

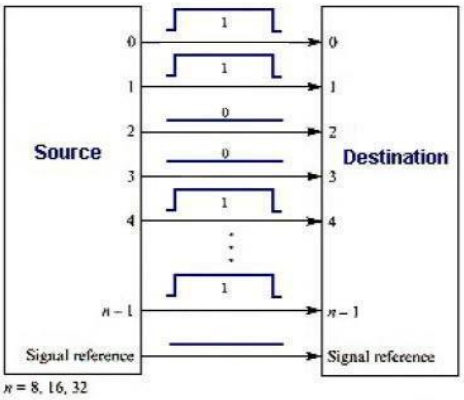
INTRODUCTION

Serial communication is common method of transmitting data between a computer and a peripheral device such as a programmable instrument or even another computer. Serial communication transmits data one bit at a time, sequentially, over a single communication line to a receiver. Serial is also a most popular communication protocol that is used by many devices for instrumentation. This method is used when data transfer rates are very low or the data must be transferred over long distances and also where the cost of cable and synchronization difficulties makes parallel communication impractical. Serial communication is popular because most computers have one or more serial ports, so no extra hardware is needed other than a cable to connect the instrument to the computer or two computers together.

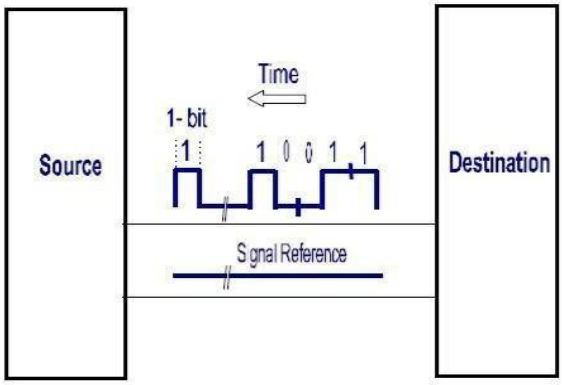
SERIAL AND PARALLEL TRANSMISSION

Let us now try to have a comparative study on parallel and serial communications to understand the differences and advantages & disadvantages of both in detail.

We know that parallel ports are typically used to connect a PC to a printer and are rarely used for other connections. A parallel port sends and receives data eight bits at a time over eight separate wires or lines. This allows data to be transferred very quickly. However, the setup looks more bulky because of the number of individual wires it must contain. But, in the case of a serial communication, as stated earlier, a serial port sends and receives data, one bit at a time over one wire. While it takes eight times as long to transfer each byte of data this way, only a few wires are required. Although this is slower than parallel communication, which allows the transmission of an entire byte at once, it is simpler and can be used over longer distances. So, at first sight it would seem that a serial link must be inferior to a parallel one, because it can transmit less data on each clock tick. However, it is often the case that, in modern technology, serial links can be clocked considerably faster than parallel links, and achieves a higher data rate.



Parallel Transmission



Serial Transmission

SERIAL DATA TRANSMISSION MODES

When data is transmitted between two pieces of equipment, three communication modes of operation can be used.

Simplex: In a simple connection, data is transmitted in one direction only. For example, from a computer to printer that cannot send status signals back to the computer.

Half-duplex: In a half-duplex connection, two-way transfer of data is possible, but only in one direction at a time.

Full duplex: In a full-duplex configuration, both ends can send and receive data simultaneously, which technique is common in our PCs.

SERIAL DATA TRANSFER SCHEMS

Like any data transfer methods, Serial Communication also requires coordination between the sender and receiver. For example, when to start the transmission and when to end it, when one particular bit or byte ends and another begins, when the receiver's capacity has been exceeded, and so on. Here comes the need for synchronization between the sender and the receiver. A protocol defines the specific methods of coordinating transmission between a sender and receiver. For example a serial data signal between two PCs must have individual bits and bytes that the receiving PC can distinguish. If it doesn't, then the receiving PC can't tell where one byte ends and the next one begin or where one bit ends and begins. So the signal must be synchronized in such a way that the receiver can distinguish the bits and bytes as the transmitter intends them to be distinguished.

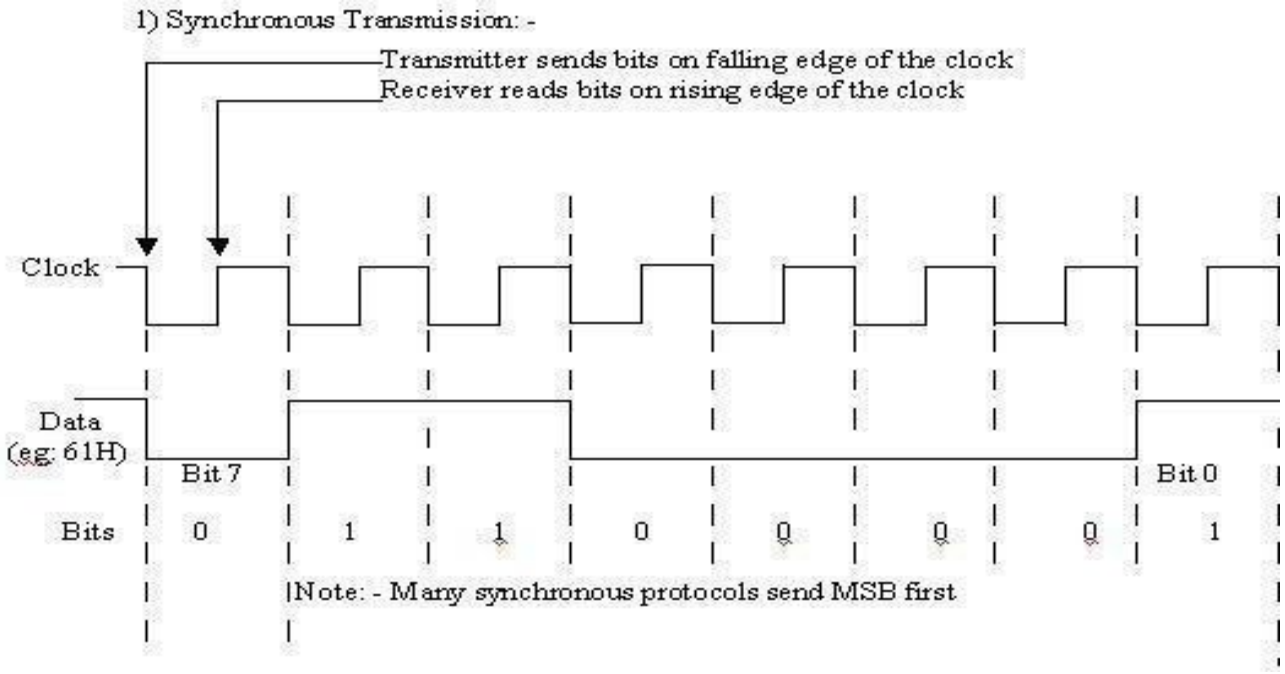
There are two ways to synchronize the two ends of the communication.

1. Synchronous data transmission
2. Asynchronous data transmission

Synchronous Data Transmission

The synchronous signaling methods use two different signals. A pulse on one signal line indicates when another bit of information is ready on the other signal line.

In synchronous transmission, the stream of data to be transferred is encoded and sent on one line, and a periodic pulse of voltage which is often called the "clock" is put on another line, that tells the receiver about the beginning and the ending of each bit.



Advantages: The only advantage of synchronous data transfer is the Lower overhead and thus, greater throughput, compared to asynchronous one.

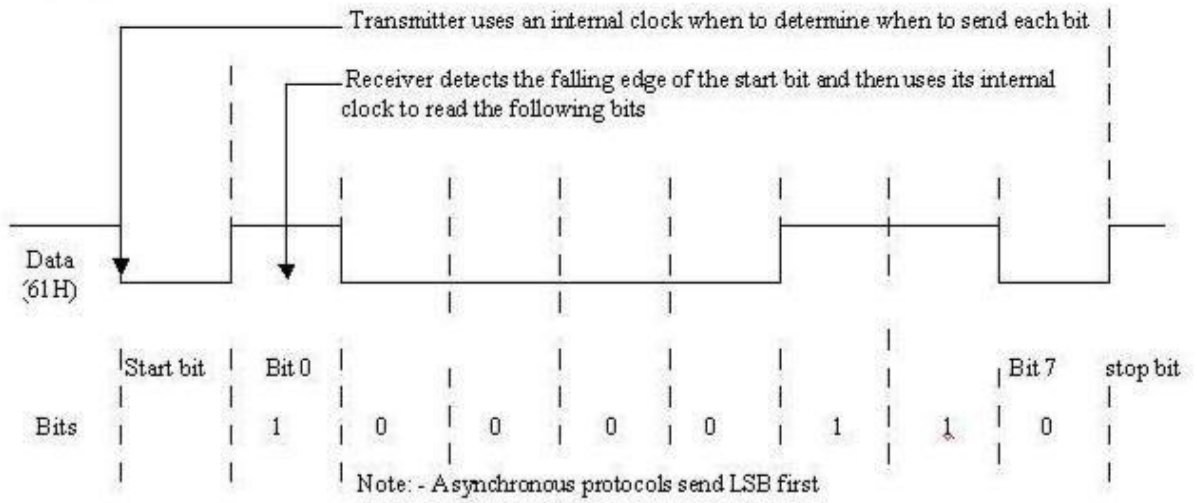
Disadvantages:

- Slightly more complex
- Hardware is more expensive

Asynchronous data transmission

The asynchronous signaling methods use only one signal. The receiver uses transitions on that signal to figure out the transmitter bit rate (known as auto baud) and timing. A pulse from the local clock indicates when another bit is ready. That means synchronous transmissions use an external clock, while asynchronous transmissions are use special signals along the transmission medium. Asynchronous communication is the commonly prevailing communication method in the personal computer industry, due to the reason that it is easier to implement and has the unique advantage that bytes can be sent whenever they are ready, a no need to wait for blocks of data to a c c u m u l a t e .

2) Asynchronous Transmission: -



Advantages:

- Simple and doesn't require much synchronization on both communication sides.
- The timing is not as critical as for synchronous transmission; therefore hardware can be made cheaper.
- Set-up is very fast, so well suited for applications where messages are generated at irregular intervals, for example data entry from the keyboard.

Disadvantages:

One of the main disadvantages of asynchronous technique is the large relative overhead, where a high proportion of the transmitted bits are uniquely for control purposes and thus carry no useful information.

8251-PROGRAMMABLE COMMUNICATION INTERFACE
(USART-Universal Synchronous/Asynchronous Receiver/Transmitter)

INTRODUCTION

A USART is also called a programmable communications interface (PCI). When information is to be sent by 8086 over long distances, it is economical to send it on a single line. The 8086 has to convert parallel data to serial data and then output it. Thus lot of microprocessor time is required for such a conversion.

Similarly, if 8086 receives serial data over long distances, the 8086 has to internally convert this into parallel data before processing it. Again, lot of time is required for such a

conversion. The 8086 can delegate the job of conversion from serial to parallel and vice versa to the 8251A USART used in the system.

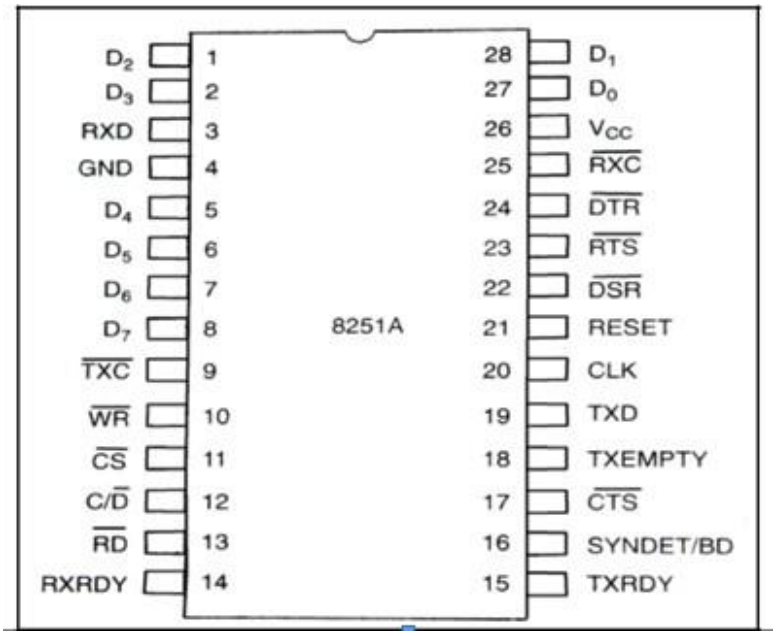
The Intel8251A is the industry standard Universal Synchronous/Asynchronous Receiver/Transmitter (USART), designed for data communications with Intel microprocessor families such as 8080, 85, 86 and
88. The 8251A converts the parallel data received from the processor on the D7-0 data pins into serial data, and transmits it on TxD (transmit data) output pin of 8251A. Similarly, it converts the serial data received on RxD (receive data) input into parallel data, and the processor reads it using the data pins D7-0.

FEATURES

- Compatible with extended range of Intel microprocessors.

- It provides both synchronous and asynchronous data transmission.
- Synchronous 5-8 bit characters.
- Asynchronous 5-8 bit characters.
- It has full duplex, double buffered transmitter and receiver.
- Detects the errors-parity, overrun and framing errors.
- All inputs and outputs are TTL compatible.
- Available in 28-pin DIP package.

PIN DIAGRAM



ARCHITECTURE

The 8251A is a USART (Universal Synchronous Asynchronous Receiver Transmitter) for serial data communication. As a peripheral device of a microcomputer system, the 8251 receives parallel data from the CPU and transmits serial data after conversion. This device

also receives serial data from the outside and transmits parallel data to the CPU after conversion. The internal block diagram of 8251A is shown in fig below.

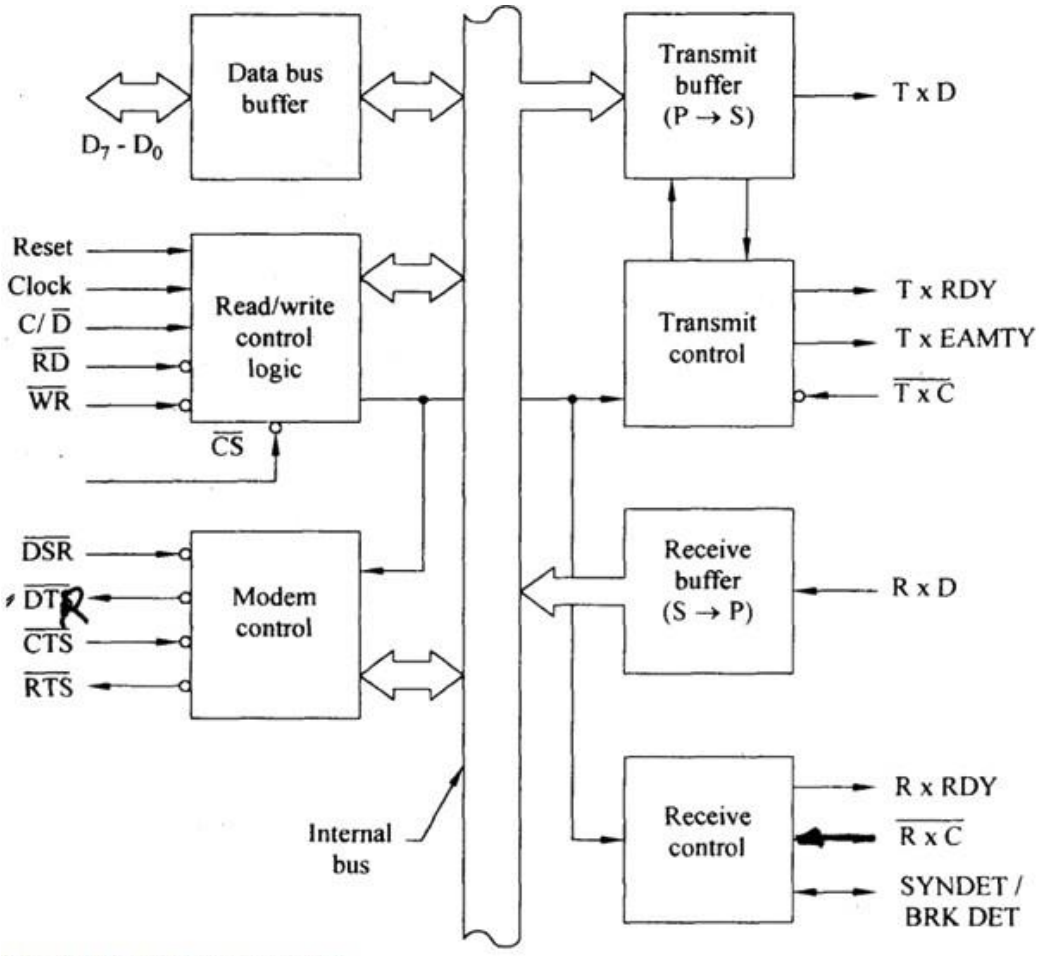


Fig. 5.7 Block diagram of 8251

Fig. 5.7 shows the block diagram of 8251 A. The block diagram shows all the elements of a programmable chip; it includes the interfacing signals, the control register and the status register. The functions of various blocks are described below:

(A) Data bus buffer: This 3-state, bidirectional buffer is used to interface the 8251A to the system data bus. Data is transmitted or received by the buffer upon execution of input and output instruction of the CPU Command words and status information are also transferred through the data bus buffer. The command, status and data in and data out are separate 8-bit registers to provide double buffering.

The functional block accepts inputs form the control bus and generates control signals for overall device operation. It contains the control word register and command word register that store the various control formats for the device functional definition.

(B) Read/Write logic and Registers:

This section includes R/W control logic, six input signals, control logic, and three buffer registers; data register, control register and status register. The input signals to control logic are as follows:

RESET: A high on this input forces the 8251A into an idle mode. The device will remain at idle until a new set of control words is written into the 8251A to program its functional definition.

A command reset operation also puts the device into the idle state.

CLK (Clock): The CLK input is used to generate internal device timing and is normally connected to the phase 2 (TTL) output of the Clock Generator. No external inputs or outputs are referenced to CLK but the frequency of CLK must be greater than 30 times the Receiver or Transmitter data bit rates.

$\overline{WR}$ (Write): A "low" on this input informs the 8251A that the CPU is writing data or control words to the 8251A.

$\overline{RD}$ (Read): A "low" on this input informs the 8251A that the CPU is reading data or status information from the 8251A.

C/$\overline{D}$	RD	WR	CS	
0	0	1	0	8251 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
×	1	1	0	Data bus-3 state
×	×	×	1	Data bus 3 state

C/$\overline{D}$ (Control/Data): This input, in conjuction with the $\overline{WR}$ and $\overline{RD}$ inputs informs the 8251A that the word on the Data bus is either a data character, control word or status information.

1 = CONTROL/STATUS; 0 = DATA

$\overline{CS}$ (Chip Select) : A "low" on this input selects the 8251A. No reading or writing will occur unless the device is selected. When $\overline{CS}$ is high, the Data Bus is in the float state and $\overline{RD}$ and $\overline{WR}$ have no effect on the chip.

(C) Modem Control:

The 8251A has a set of control inputs and outputs that can be used to simplify the interface to almost any modem. The modem control signals are general purpose in nature and can be used for functions other than modem control, if necessary.

$\overline{DSR}$ (Data Set Ready) : The $\overline{DSR}$ input signal is a general-purpose, 1-bit inverting input port. Its condition can be tested by the CPU using a Status Read operation. The DSR input is normally used to test modem condition such as Data Set Ready.

$\overline{\text{DTR}}$ (Data Terminal Ready): The $\overline{\text{DTR}}$ output signal is a general-purpose, 1-bit inverting output port. It can be set "low" by programming the appropriate bit in the Command instruction word. The $\overline{\text{DTR}}$ output signal is normally used for modem control such as Data Terminal Ready.

$\overline{\text{RTS}}$ (Request to Send): The $\overline{\text{RTS}}$ output signal is a general-purpose, 1-bit inverting output port. It can be set "low" by programming the appropriate bit in the Command instruction word. The $\overline{\text{RTS}}$ output signal is normally used for modem control such as Req :est to send.

$\overline{\text{CTS}}$ (Clear to Send): A "low" on this input enables the 8251A to transmit serial data if the Tx Enable bit in the Command byte is set to a "one", if either a T× Enable off or $\overline{\text{CTS}}$ off condition occurs while the T× is in operation, the T× will transmit all the data in the USART, written prior to T× Disable command before shutting down.

(D) Transmitter Buffer:

The transmitter Buffer accepts parallel data from the Data Bus Buffer, converts it to a serial bit stream, inserts the appropriate characters or bits (based on the communication technique) and outputs a composite serial stream of data on the TxD output pin on the falling edge of $\overline{\text{T x C}}$. The transmitter will begin transmission upon being enabled, if $\overline{\text{CTS}}$ = 0. The $\overline{\text{T x C}}$ line will be held in the marking state immediately upon a master Reset or when Tx Enable or $\overline{\text{CTS}}$ is off or the transmitter is empty.

(E) Transmitter Control:

The Transmitter Control manages all activities associated with the transmission of serial data. It accepts and issues signals both externally and internally to accomplish this function.

T x RDY (Transmitter Ready): This output signals the CPU that the transmitter is ready to accept a data character. The T x RDY output pin can be used as an interrupt to the system, since it is masked by T x Enable; or, for Polled operation, the CPU can check T x RDY using a Status Read operation. T x RDY is automatically reset by the leading edge of $\overline{\text{WR}}$ when a data character is loaded from the CPU.

T x E (Transmitter Empty): When the 8251A has no character to send, the T x empty will go high. It resets upon receiving a character from CPU if the transmitter is enabled. Tx empty remains high when the transmitter is disabled. T x Empty can be used to indicate the end of transmission node, so that the CPU knows when to turn around in the half-duplex operation mode.

In the synchronous mode, a high on this output indicates that a character has not been loaded and the SYNC character or characters are about to be are being transmitted as filters. T x Empty does not go low when the Sync characters are being shifted out.

T x C (Transmitter Clock): The Transmitter Clock control the rate at which the character is to be transmitted. In the Synchronous transmission mode, the Baud Rate (1x) is equal to the T x C frequency. In Asynchronous transmission mode, the baud rate is a fraction of the actual T x C frequency. A portion of the mode instruction selects this factor it can be 1,1/16 or 1/64 the T x C.

For example
If Baud rate equals 220 Baud

$\overline{TXC}$ equals 220 Hz in the 1x mode.

$\overline{TXC}$ equals 3.52 KHz in the 16x mode.

$\overline{TXC}$ equals 14.08 KHz in the 64x mode.

The falling edge of $\overline{TXC}$ shifts the serial data out of the 8251A.

(F) Receiver Buffer:

The Receiver accepts serial data, converts this serial input to parallel format checks for bits or characters that are unique to the communication technique and sends an "assembled" character to the CPU. Serial data is input to R × D pin, and is clocked in on this rising edge of $\overline{R \times C}$.

(G) Receiver Control:

This functional block shown in Fig. manages all receiver - related activities which consists of the following features.

The R × D initialization circuits prevents the 8251A from mistaking and unused input line for an active low data line in the break condition. Before starting to receive serial characters on the R × D line, a valid '1' must first be detected after a chip master reset. Once this has been determined, a search for a valid low bit (start bit) is enabled. This feature is only active in the asynchronous mode, and is only done once for each master reset.

The false start bit detection circuit prevents false starts due to a transient noise spike by first detecting the falling edge and then strobing the nominal center of the start (R × D = low).

Parity error detection sets the corresponding status bit.

The framing error status bit is set if the stop bit is absent at the end of the data byte (asynchronous mode).

R × RDY (Receiver Ready) : This output indicates the the 8251A contains a character that is ready to be input to the CPU. R × RDY can be connected to the interrupt structure of the CPU or, for polled operation, the CPU can check the condition of R × RDY using a status read operation.

R × Enable, when off, holds R × RDY in the reset condition. For asynchronous mode, to set R × RDY, the receiver must be enabled to sense a start bit and a complete character must be assembled and transferred to the data output register.

Failure to read the received character from the R × Data output register prior to the assembly of the next R × data character will set overrun condition error and the previous character will be written over and lost. If the R × data is being read by the CPU when the internal transfer is occurring, overrun error will be set and the old character will be lost.

$\overline{R \times C}$ (Receiver Clock) : The receiver clock controls the rate at which the character is to be received. In synchronous mode, the baud rate (1×) is equal to the actual frequency of $\overline{R \times C}$. In asynchronous mode, the baud rate is a fraction of the actual $\overline{R \times C}$ frequency. A portion of the mode instruction selects this factor: 1, 1/16 of 1/64 the $\overline{R \times C}$.

For Example:

Baud rate equals 200 baud, if

$\overline{R \times C}$ equals 200 Hz in the 1 × mode.

$\overline{R \times C}$ equals 3200 Hz in the 16 × mode.

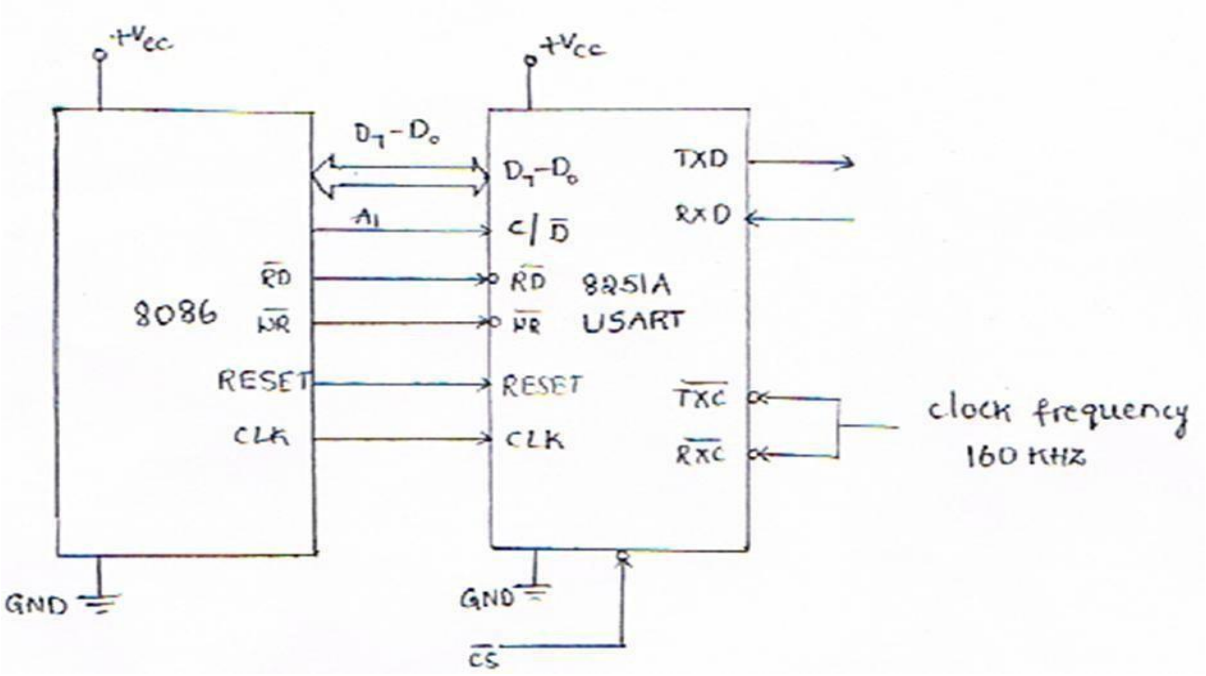
$\overline{R \times C}$ equals 128 KHz in the 64 × mode.

SYNDET (SYNC Detect/BRKDET Break Detect): This is used in Synchronous Mode for SYNDET and may be used as either input or output, programmable through the Control Word. It is reset to output mode low upon RESET. When used as an output (internal Sync mode), the SYNDET pin will go "high" to indicate that the 8251A has located the SYNC character in the Receive mode. If the 8251A is programmed to use double Sync characters (bi-sync), then SYNDET will go "high" in the middle of the last bit of the second Sync character SYNDET is automatically reset upon a status Read operation.

When used as an input (external SYNC detect mode), a positive going signal will cause the 8251A to start assembling data characters on the rising edge of the next $\overline{R \times C}$. Once in SYNC, the "high" input signal can be removed. When External SYNC Detect is programmed, internal SYNC Detect is disabled.

BREAK (Async Mode Only): This output will go high whenever the receiver remains low through two consecutive stop bit sequences (including the start bits, data bits, and parity bits); Break Detect may also be read as a Status bit. It is reset only upon a master chip Reset of R × Data returning to a "one" state.

8251A USART INTERFACING WITH 8086



PROGRAMMING THE 8251A

Prior to starting a data transmission or reception, the 8251A must be loaded with a set of control words generated by the microprocessor. These control signals define the complete functional definition of the 8251A and must immediately follow a reset operation (internal or external). The control words are split into two formats.

1. Mode instruction
2. Command instruction

Mode instruction: Mode instruction is used for setting the function of the 8251A. Mode instruction will be in "wait for write" at either internal reset or external reset. That is, the writing of a control word after resetting will be recognized as a "mode instruction."

Items set by mode instruction are as follows:

- Synchronous/asynchronous mode
- Stop bit length (asynchronous mode)
- Character length
- Parity bit
- Baud rate factor (asynchronous mode)
- Internal/external synchronization (synchronous mode)
- Number of synchronous characters (Synchronous mode)

The bit configuration of mode instruction format is shown in Figures below. In the case of synchronous mode, it is necessary to write one-or two byte sync characters. If sync characters

were written, a function will be set because the writing of sync characters constitutes part of mode instruction.

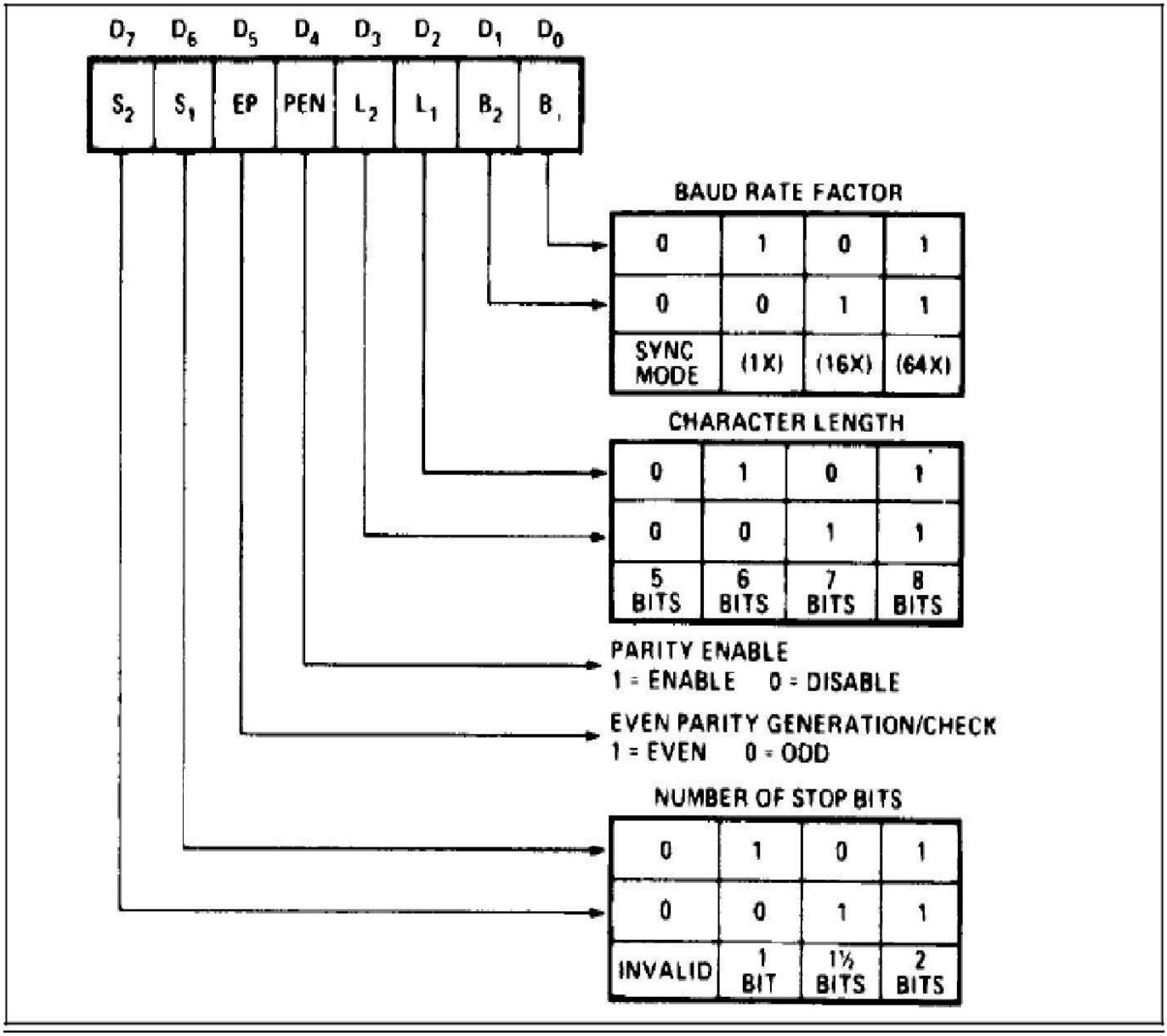
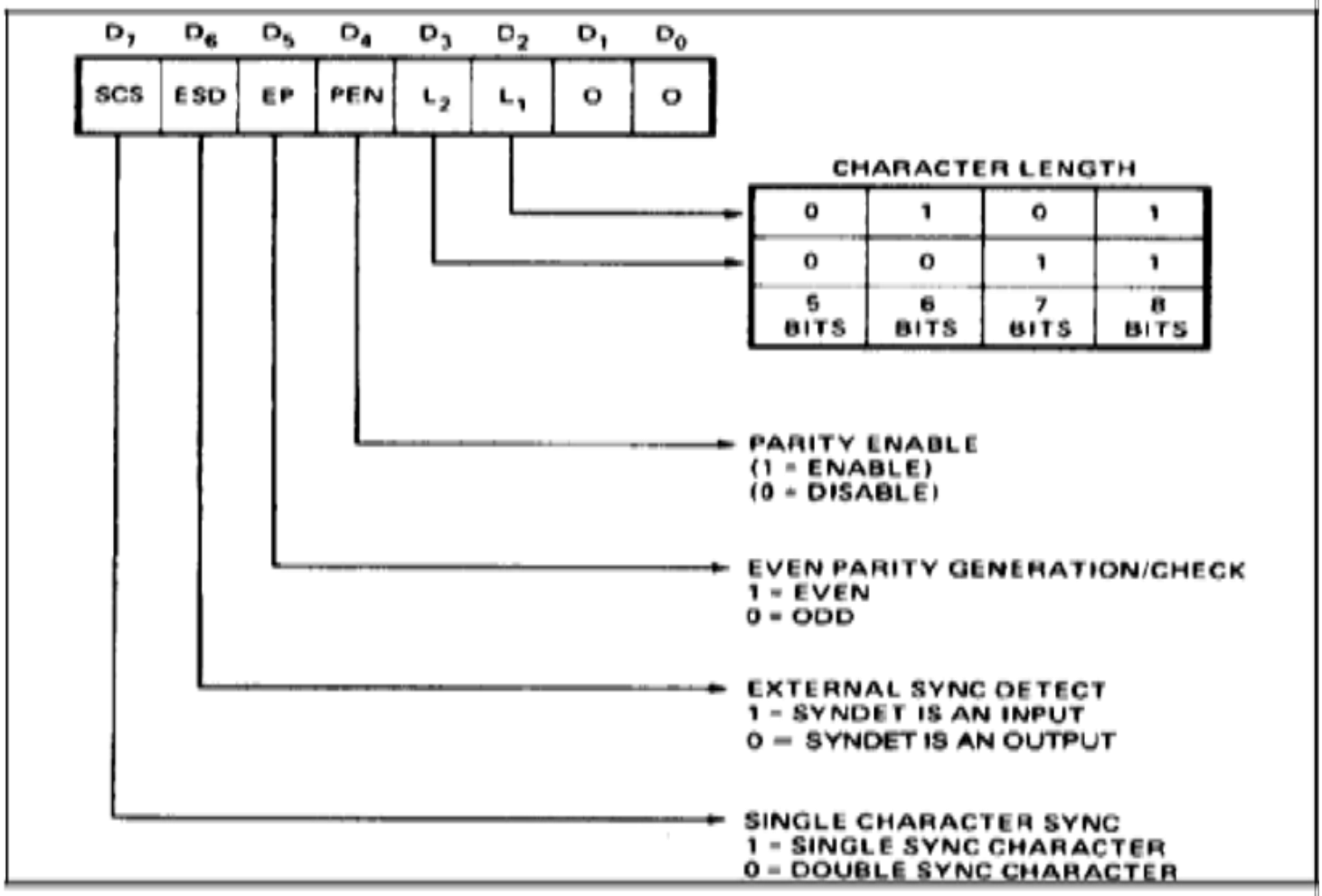


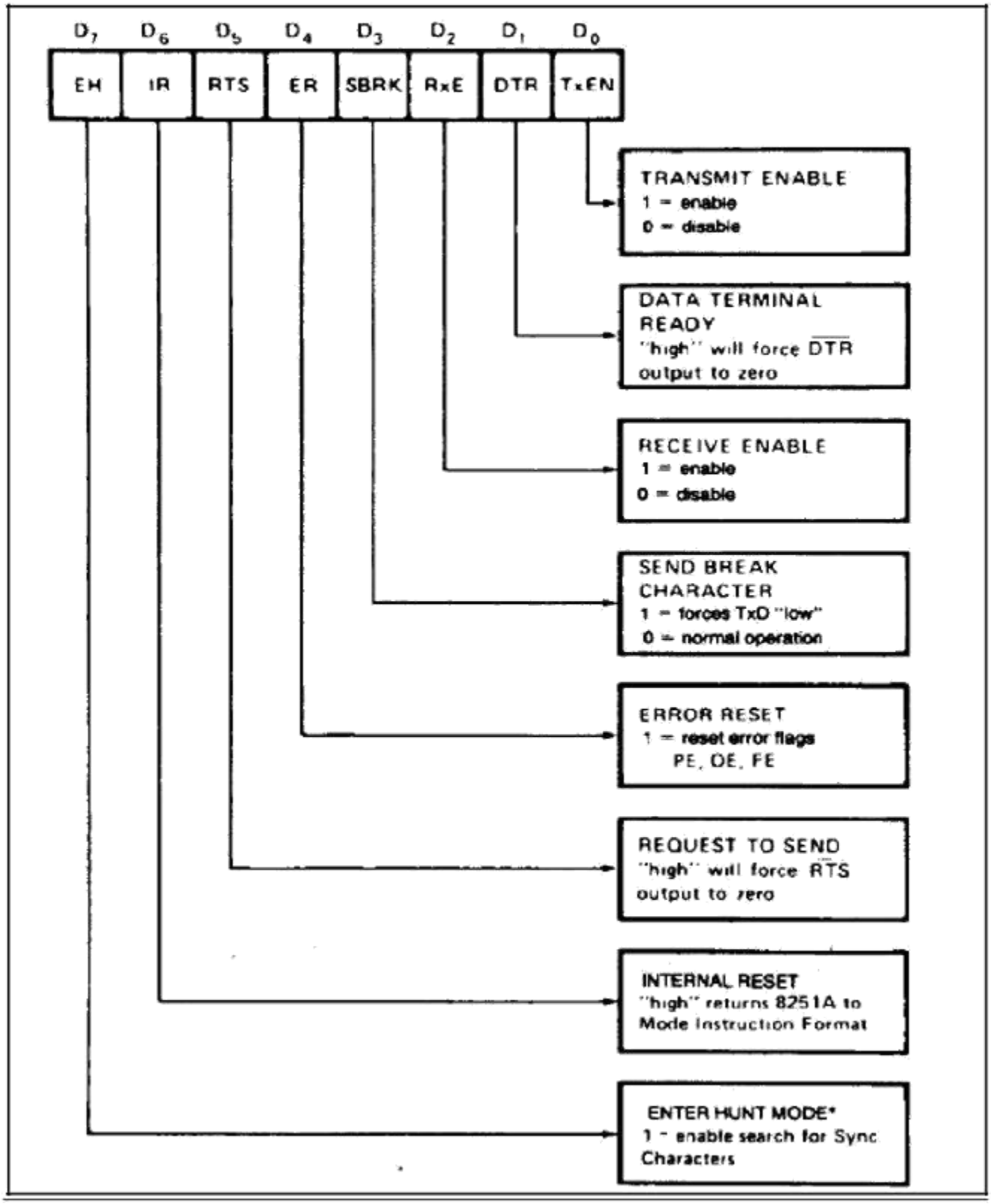
Fig. Mode instruction format, Asynchronous mode



Command Instruction: Command is used for setting the operation of the 8251. It is possible to write a command whenever necessary after writing a mode instruction and sync characters.

Items to be set by command are as follows:

- Transmit Enable/Disable
- Receive Enable/Disable
- DTR, RTS Output of data.
- Resetting of error flag.
- Sending to break characters
- Internal resetting
- Hunt mode (synchronous mode)


D7 D6 D5 D4 D3 D2 D1 D0
EH
IR
RTS
ER
SBRK
RxE
DTR
TxEN
TRANSMIT ENABLE
1 = enable
0 = disable
DATA TERMINAL READY
"high" will force DTR output to zero
RECEIVE ENABLE
1 = enable
0 = disable
SEND BREAK CHARACTER
1 = forces TxD "low"
0 = normal operation
ERROR RESET
1 = reset error flags PE, OE, FE
REQUEST TO SEND
"high" will force RTS output to zero
INTERNAL RESET
"high" returns 8251A to Mode Instruction Format
ENTER HUNT MODE*
1 = enable search for Sync Characters

Status Word: It is possible to see the internal status of the 8251 by reading a status word. The format of status word is shown below.

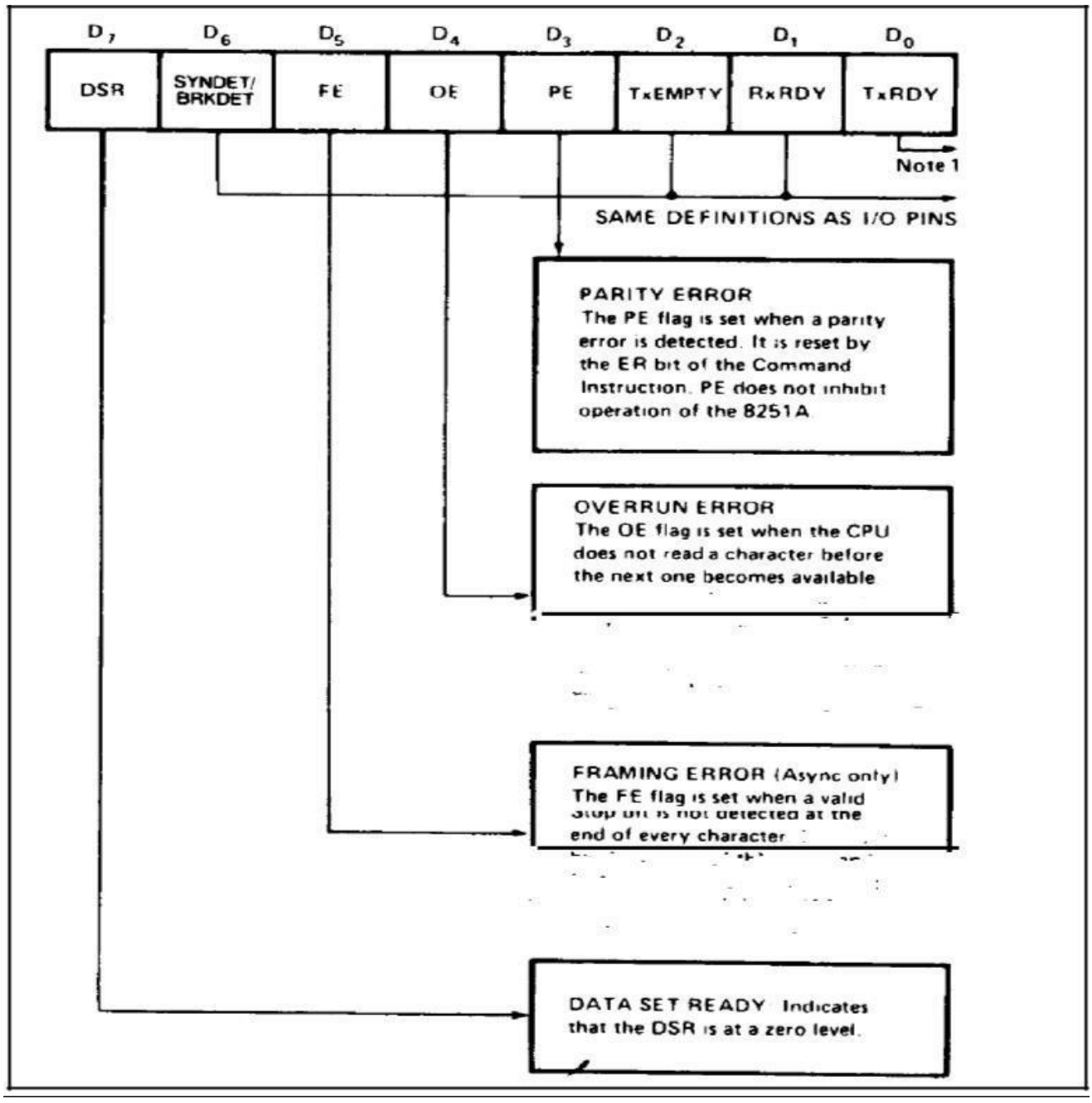


Fig. Status word

RECOMMENDED STANDARD -232C (RS-232C)

RS-232 (Recommended standard-232) is a standard interface approved by the Electronic Industries Association (EIA) for connecting serial devices. In other words, RS-232 is a long-established standard that describes the physical interface and protocol for relatively low-speed serial data communication between computers and related devices. RS-232 is the interface that your computer uses to talk to and exchange data with your modem and other serial devices. The serial ports on most computers use a subset of the RS-232C standard.

RS-232C is defined as the "Interface between data terminal equipment and data communications equipment using serial binary data exchange." This definition defines data terminal equipment (DTE) as the computer, while data communications equipment (DCE) is the modem. A modem cable has pin-to-pin connections, and is designed to connect a DTE device to a DCE device. In addition to communications between computer equipment over telephone lines, RS-232C is now widely used for direct connections between data acquisition devices and computer systems. RS-232C cables are commonly available with 4, 9 or 25-pin wiring. The 25-pin cable connects every pin; the 9-pin cables do not include many of the uncommonly used connections; 4-pin cables provide the bare minimum connections, and have jumpers to provide "handshaking" for those devices that require it.

In RS-232, user data is sent as a time-series of bits. Both synchronous and asynchronous transmissions are supported by the standard. In addition to the data circuits, the standard defines a number of control circuits used to manage the connection between the DTE and DCE. Each data or control circuit only operates in one direction, which is, signaling from a DTE to the attached DCE or the reverse. Since transmit data and receive data are separate circuits, the interface can operate in a full duplex manner, supporting concurrent data flow in both directions.

The RS-232 standard defines the voltage levels that correspond to logical one and logical zero levels for the data transmission and the control signal lines. Valid signals are either in the range of +3 to +15 volts for logic 0 or the range -3 to -15 volts for logic 1, the range between - 3 to +3 volts is not a valid RS-232 level. For data transmission lines (TxD, RxD and their secondary channel equivalents) logic one is defined as a negative voltage, the signal condition is called "mark." Logic zero is positive and the signal condition is termed "space." The 9-pin RS-232C standard is shown in figure below.