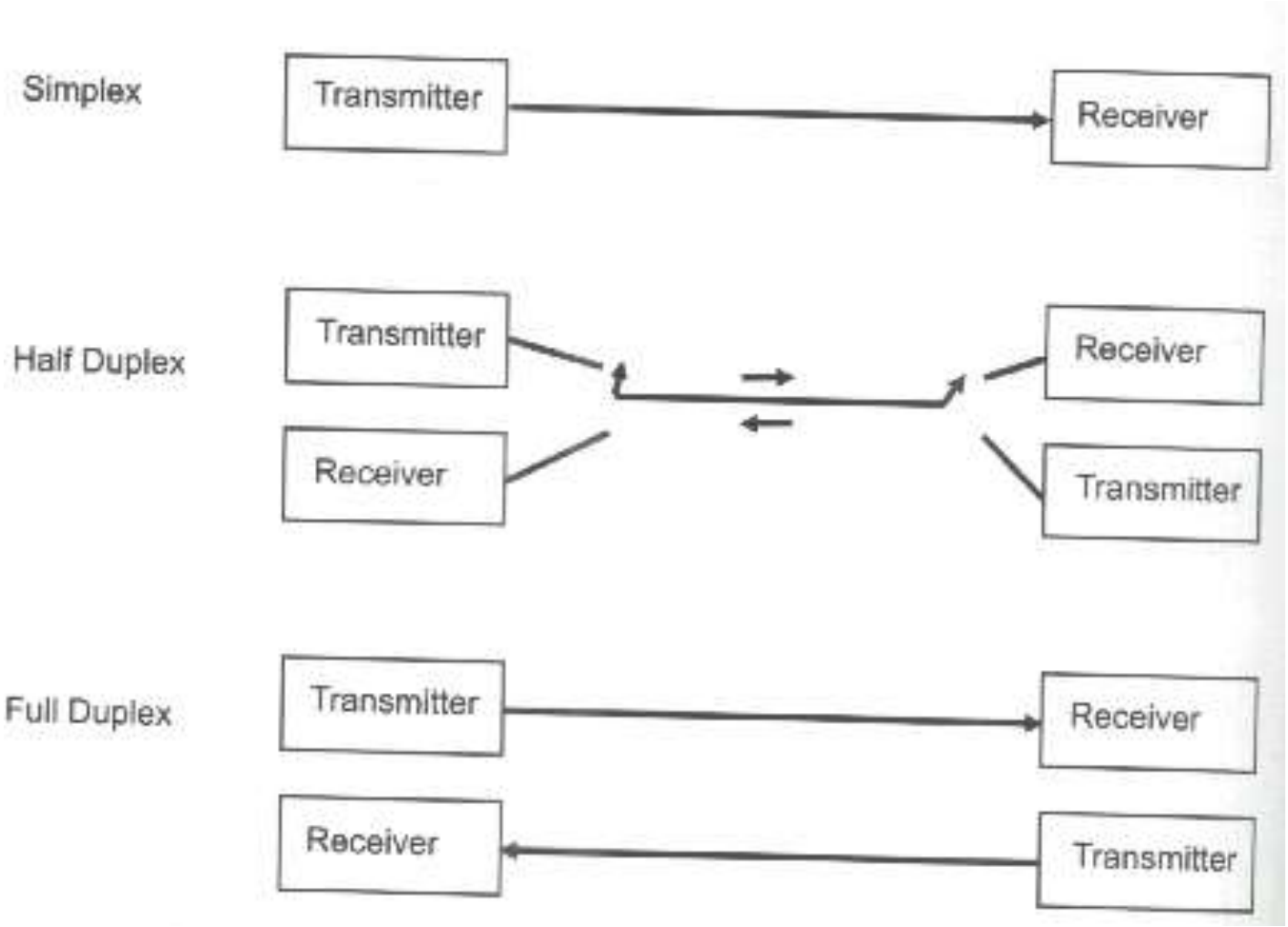
8251 -USART

Serial I/O - Programmable Communication Interface

Data Communications

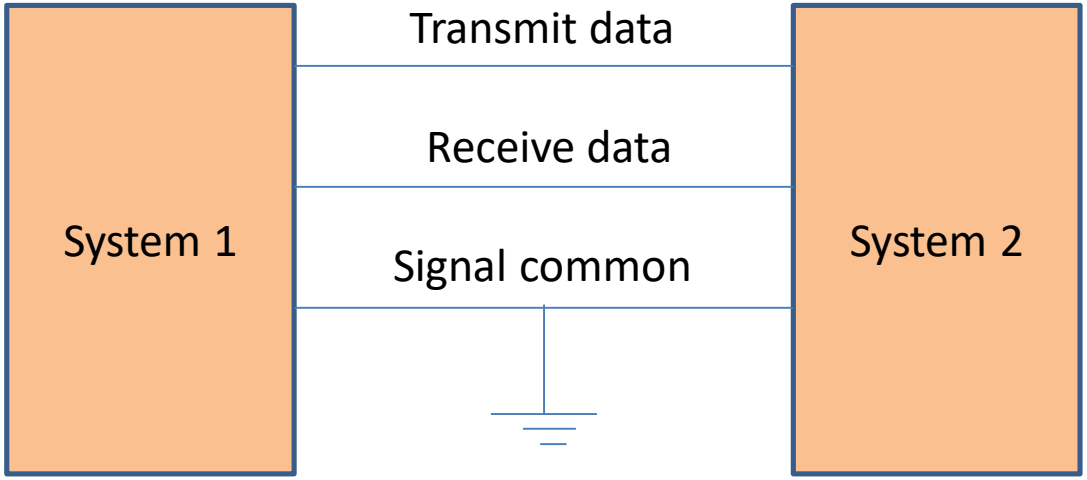
- Data communications refers to the ability of one computer to exchange data with another computer or a peripheral
- Physically, the data comm. path may be a short, 5 to 10 feet ribbon cable connecting a microcomputer and parallel printer; or it might be a high speed telecommunications port connecting two computers thousands of miles apart.
- Standard data communication interfaces and standards are needed
- Centronic's parallel printer interface
- RS-232 defines a serial communications standard
- We focus on serial I/O this week
- 8251 USART (Universal Synchronous/Asynchronous Receiver/Transmitter) is the key component for converting parallel data to serial form and vice versa
- Two types of serial data communications are widely used
  - Asynchronous communications
  - Synchronous communications

Types of Transmission



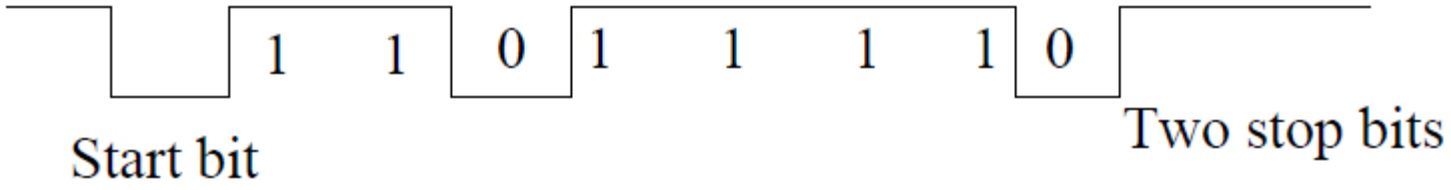
Asynchronous Communications

- Eliminates the need for a clock signal between two microprocessor based systems



Asynchronous Communications

- Data to be transmitted is sent out one character at a time and the receiver end of the communication line synchronization is performed by examining synchronization bits that are included at the beginning and at the end of each character

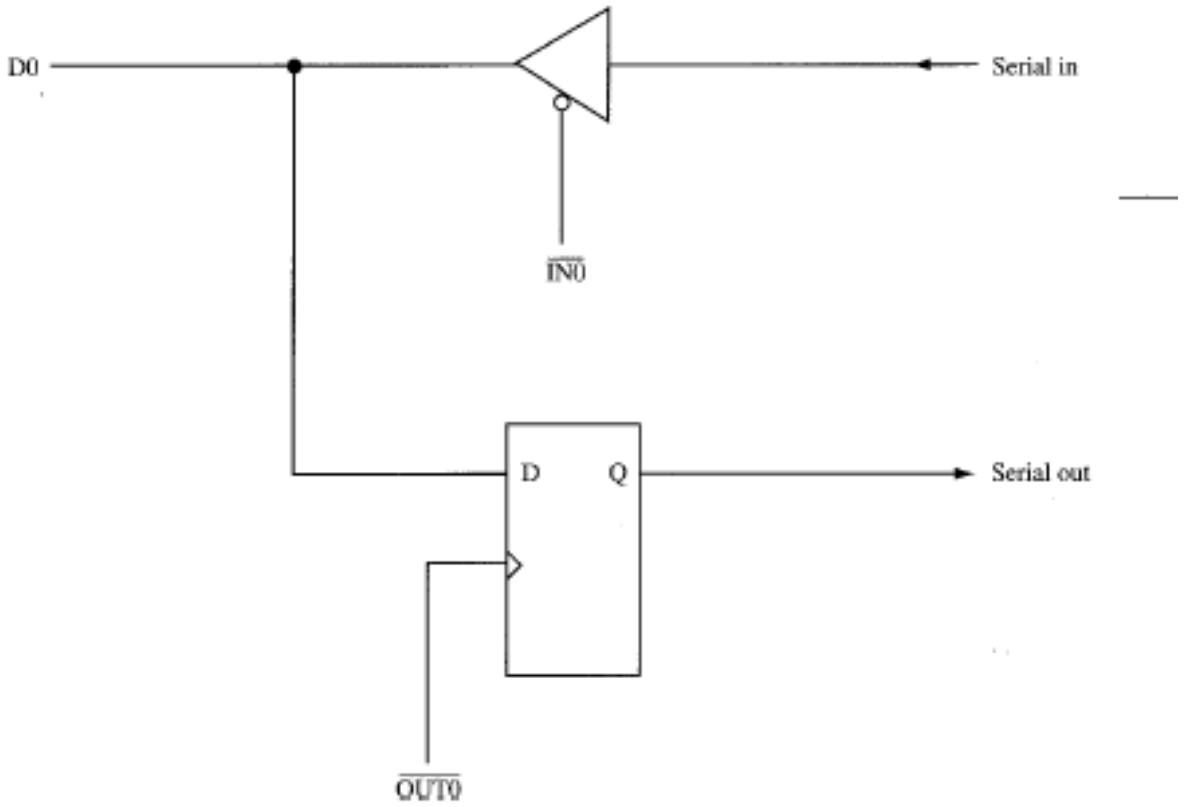


Examples

- What is the data rate in bits/sec and character rate if the bit time is 3.33 ms
  - Bit rate = 1 / 3.33 ms = 300 bits/sec
  - 11 x 3.33 ms = 36.63 ms required to transmit a character so character rate = 1/36.63 ms = 27.3 char/sec
- Modems typically transmit data over the telephone network at 9600, 14400, 28800 or 56K bps.
- Ex: If 1 MByte file is to be transmitted to another computer using a modem calculate the transmission time
  - 9600 bps: 1048576x10/ 9600 bits/sec = 1092 s = 18 minutes and 12 sec
  - 28800 bps: 364 s = 6 minutes and 4 sec

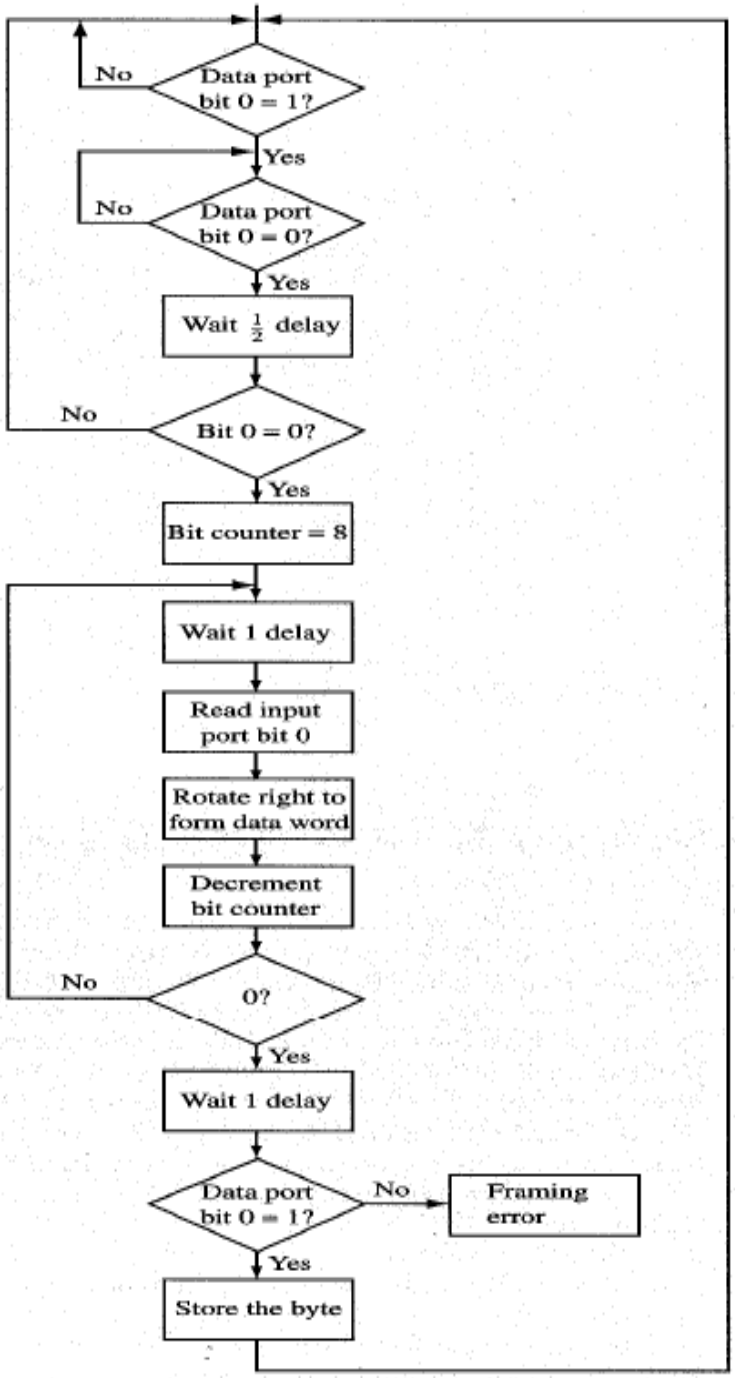
Building a Serial I/O port –Transmitter section

- This program serializes data through software



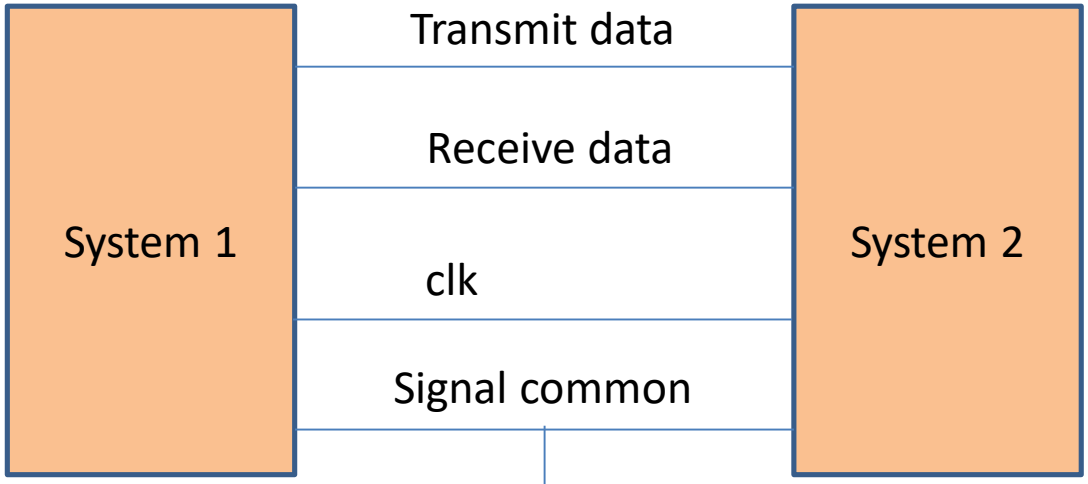
Building a Serial I/O port –Transmitter

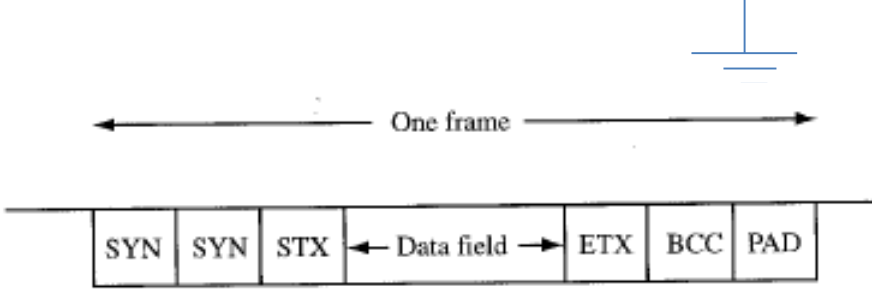
```
;Function:  Serial data transmitter.  DELAY
;          procedure determines data rate.
;Inputs:    Character to be transmitted assumed
;          passed in AL.
;Ouputs:    Serial data on bit 0 of DPORT.

;Destroys:  AL,CX,flags.

           EXTRN     DELAY:NEAR
           DPORT     EQU  00H

CODE       SEGMENT
           ASSUME    CS:CODE

FIG10_3    PROC      NEAR
           MOV       CX,10          ;10 bits/char
           CLC                      ;Start bit
           RCL       AL,1           ;Move to position 0
TRANS:     OUT       DPORT,AL       ;Transmit bit
           CALL      DELAY          ;Wait
           RCR       AL,1           ;Next bit
           STC                      ;Stop bit
           LOOP      TRANS          ;Do 10 times
           RET
FIG10_3    ENDP
           CODE    ENDS
           END
```



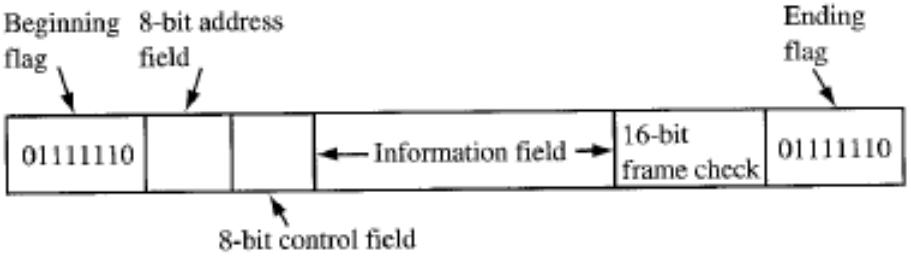
Building a Serial I/O port
Receiver- Flowchart

Synchronous Communications





BISYNC: Each block of data has synch characters. The size of block data can be 100 or more bytes. BCC checks for errors.



Serial Data Link Control: Developed by IBM used for computer networking (Token Ring). After Flag byte the network address is sent. Control Byte stores information about sequence of data etc. Data is thousands of bits. 16 bit field is used for error checking.

USART

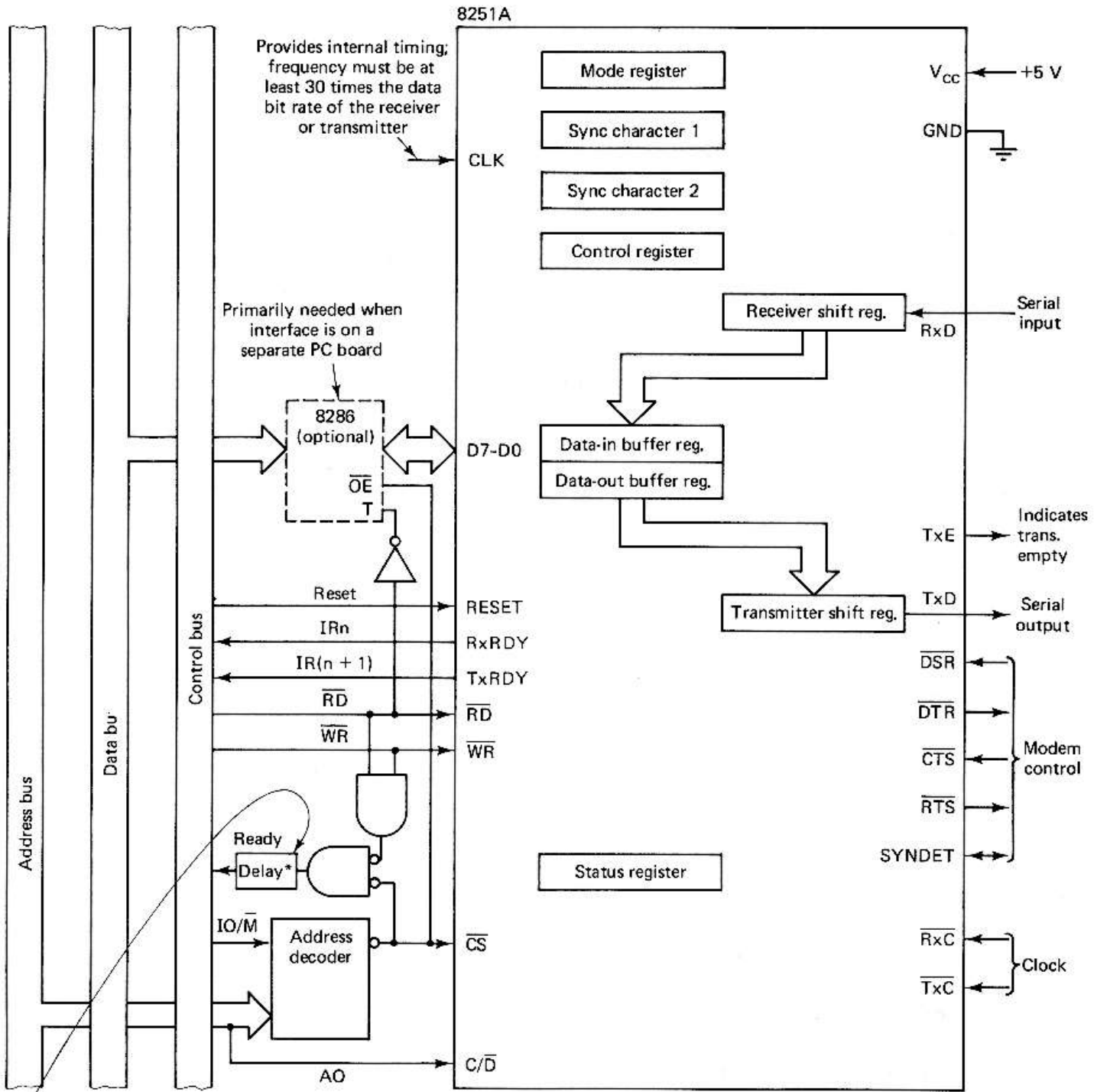
- It is possible to use either of the two methods.
- There are special IC chips for serial data communication
- UART: universal asynchronous receiver transmitter
- USART: universal Synchronous/Asynchronous Receiver/Transmitter
- COM port in the original IBM PC uses 8250 UART
- INTEL has USART 8251
- National Semiconductor's improved version of 8250A is 16450.
- 16450, 16550, 16552 (dual 16550)
- Data Transmission
  - simplex
  - half duplex
  - full duplex

8251 receiver

- The receiver section: whenever RxD line goes low, control assumes it is a start bit, waits for half bit time and samples again.
  - responsible for reading the serial bit stream of data atRxD(receive data) input and converting it into parallel form.
  - RxRDY(receive ready) output switched to logic 1 level to tell the microprocessor **that a char. is available** and is sitting inside the USART and should be read from the receive –buffer register.
  - RxC Receiver Clock. Controls the rate which bits are received by the USART. In Asynchronous Mode, the clock can be set to 1,16 or 64 times the baud.

8251 transmitter

- Transmitter section receives parallel data from the microprocessor over the data bus. The character is then automatically framed with the start bit, parity bit, correct number of stop bits, and put into the transmit data buffer register.
  - Finally, it is shifted out of this register to produce a bit serial output on the TxD line.
  - TxRDY is switched to logic 1 when the **transmit buffer register** is empty.
  - TxE transmitter Empty –This is an output signal. Logic 1 on this line indicates the **output register** is empty. Reset when a byte is transferred from the buffer to output registers.
  - TxC Transmitter Clock. Controls the rate which bits are transmitted by the USART. The clock can be set to 1,16 or 64 times the baud (using Mode Word –next slide)



8251A serial com. inter-face

Initializing the 8251

- To implement serial communication the MPU must inform the 8251 about the mode, baud, stop bits, parity etc. A set of control words must be loaded.
  - Mode Words
    - Specifies general characteristics of the operation.
  - Command Words
    - Enables the data transmission and/or reception
  - Status Word provides the information concerning register status and transmission errors.
- Any control word written into the control register after a mode word is interpreted as a command word; that means a command word can be changed anytime, however 8251 should be reset prior to writing a Mode word.
- 8251 can be reset internally by using the Internal Reset Bit D6.

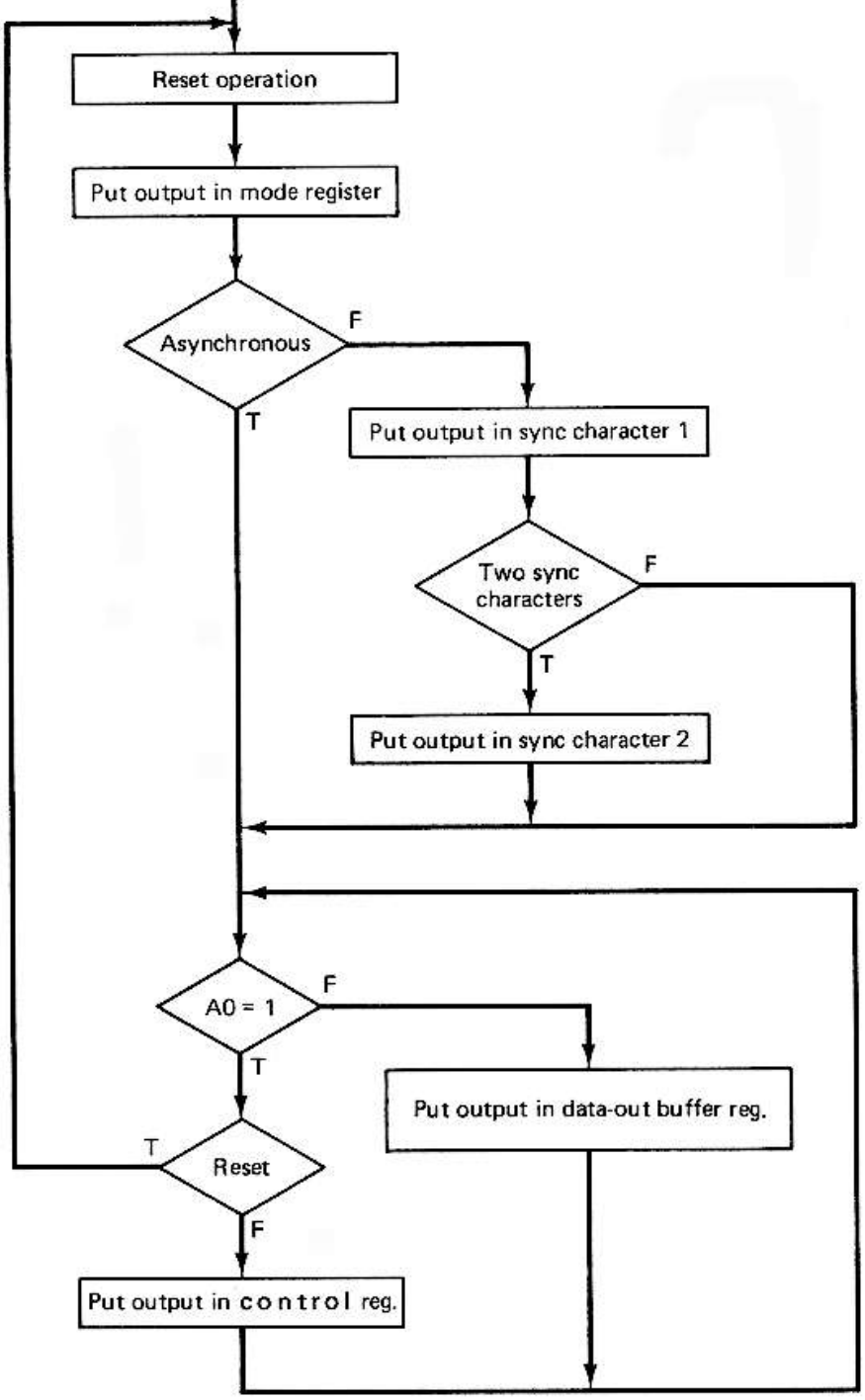
8251 A Serial Communication Interface

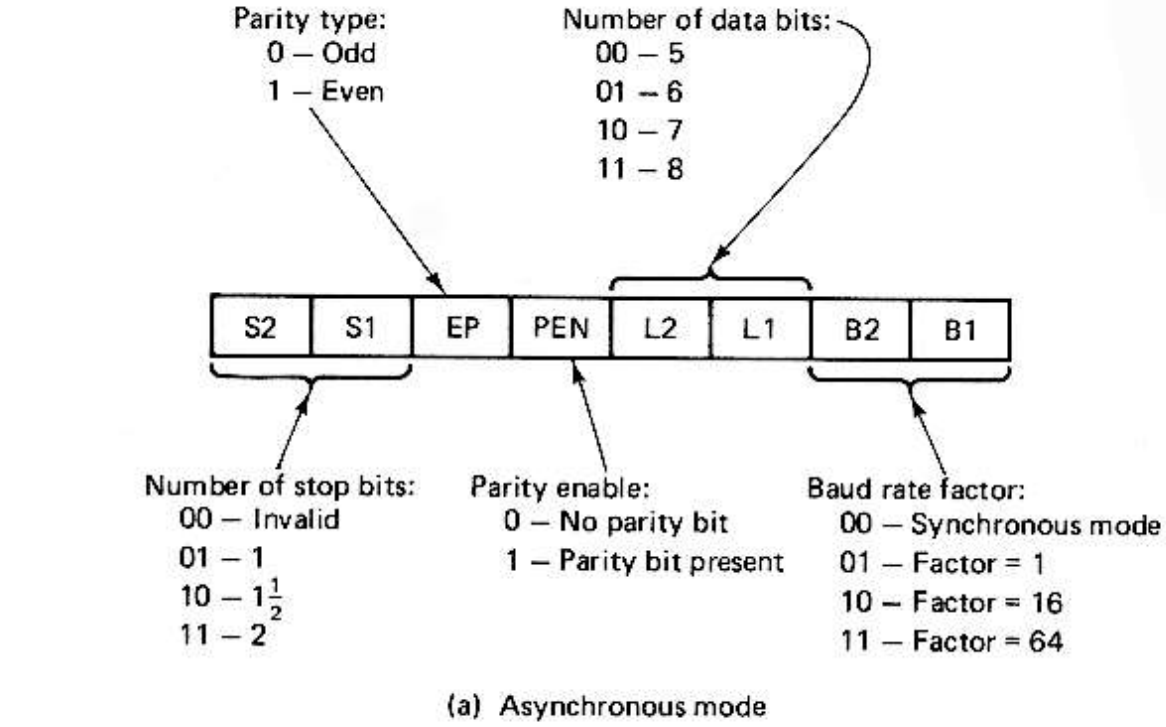
- The 8251A internally interprets the C/D,RD and WR signals as follow:

$C/\overline{D}$ (=Ao)	$\overline{RD}$	$\overline{WR}$	
0	0	1	Data input from the data-in buffer
0	1	0	Data output to the data-out buffer
1	0	1	Status register is put on data bus
1	1	0	Data bus is put in mode, control or sync character register

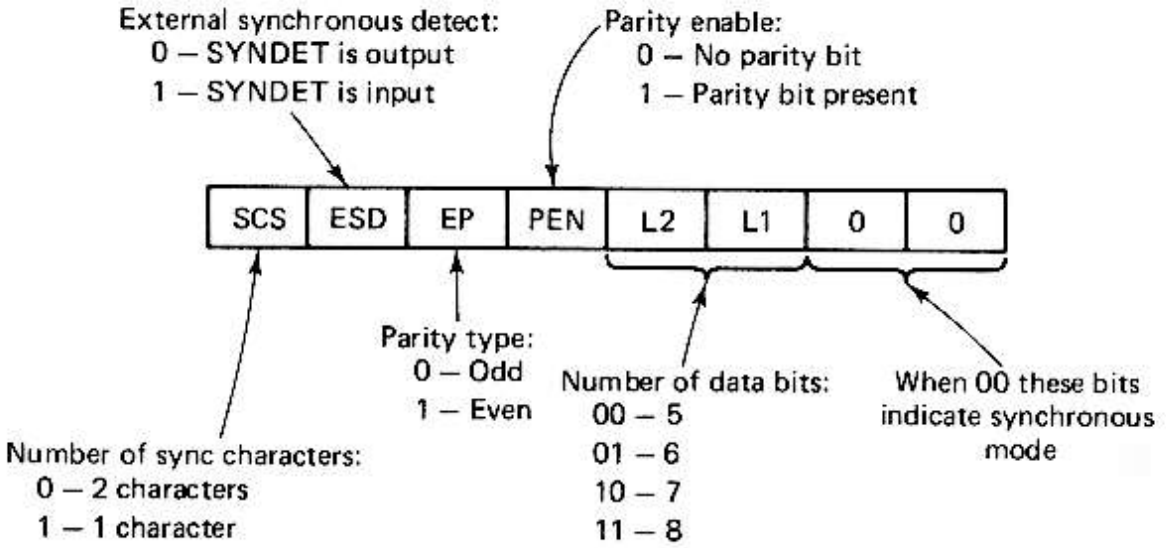
- Whether the mode, control or sync character register is selected depends on the accessing sequence.

- A flowchart of the sequencing is given in Fig.


Reset operation
Put output in mode register
Asynchronous
F
T
Put output in sync character 1
Two sync characters
F
T
Put output in sync character 2
A0 = 1
F
T
Put output in data-out buffer reg.
T
Reset
F
Put output in control reg.



(a) Asynchronous mode



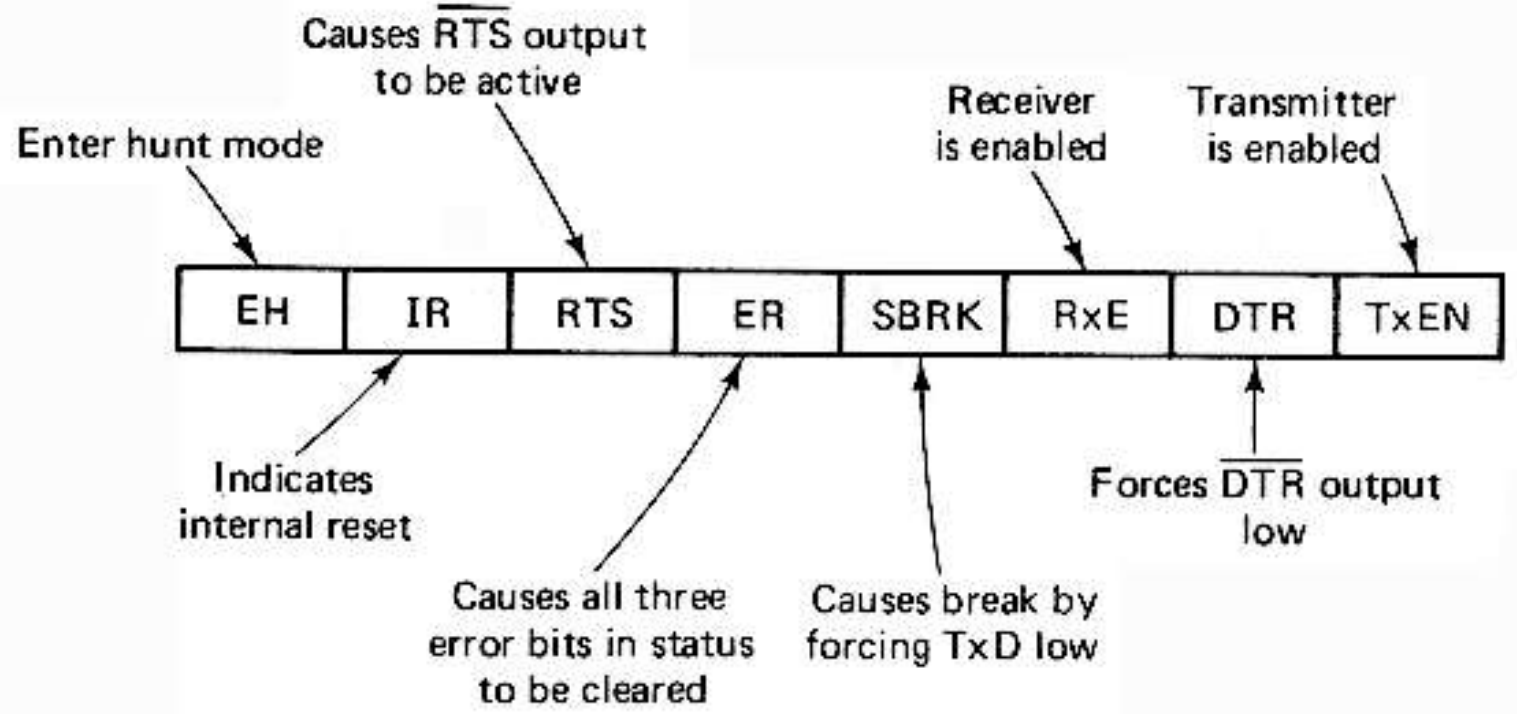
(b) Synchronous mode

Format of the mode register

What value must be written into the mode control register with baud rate divided by 16, char. Size 16 bits, odd parity, one stop bit ?

01011110 b = 5Eh

Format of the control register



Note: In all cases action is taken when the bit is set to 1.

Example 1

- A program sequence which initializes the mode register and gives a command to enable the transmitter and begin an asynchronous transmission of 7-bit characters followed by an even-parity bit and 2 stop bits is:

```
MOV AL,11111010B
OUT 51H,AL
MOV AL,00110011B
OUT 51H,AL
```

Example 2

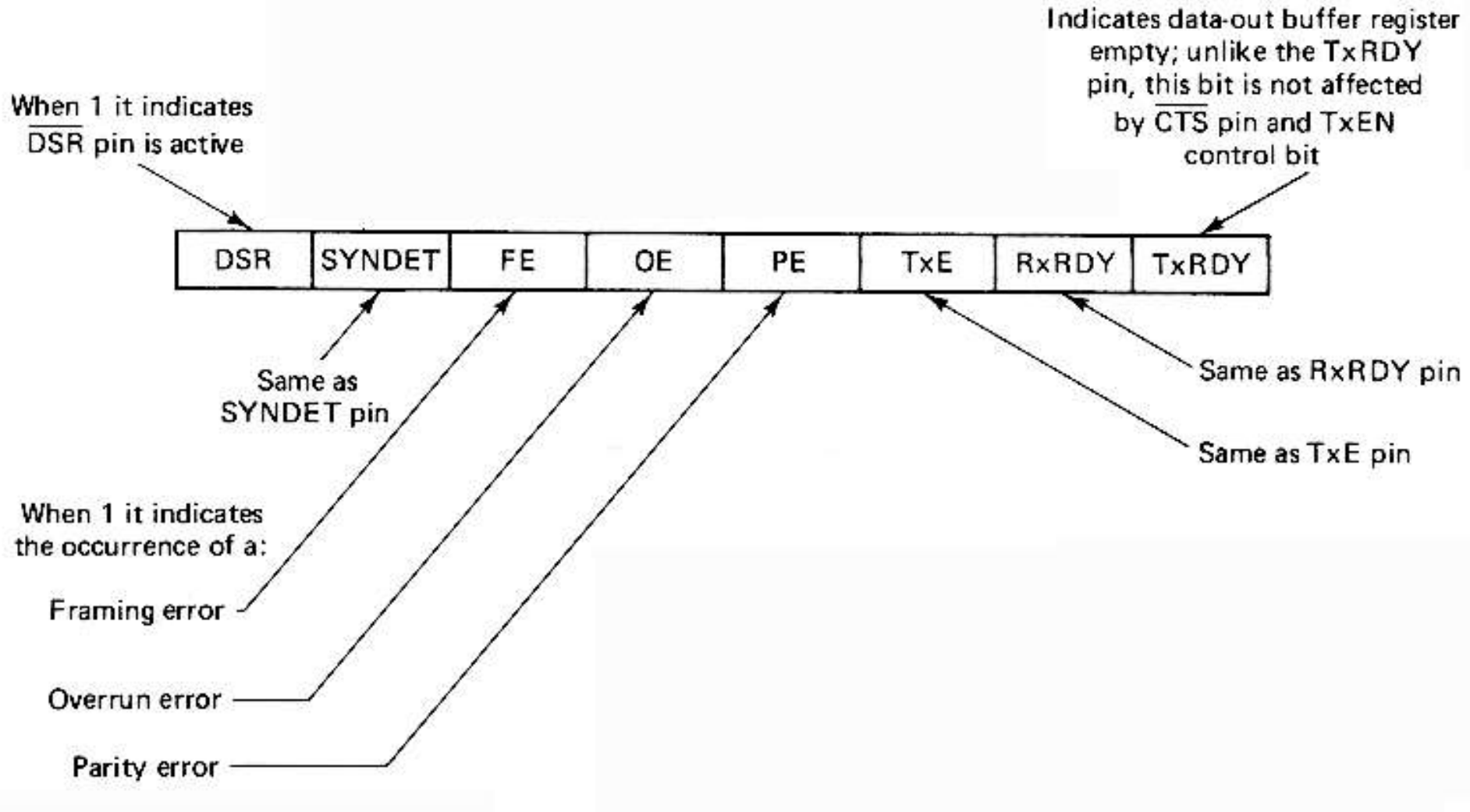
- This sequence assumes that the mode and control registers are at address 51H and the clock frequencies are to be 16 times the corresponding baud rates.

The sequence:

```
MOV AL,00111000B
OUT 51H,AL
MOV AL,16H
OUT 51H,AL
OUT 51H,AL
MOV AL,10010100B
OUT 51H,AL
```

would cause the same 8251A to be put in synchronous mode and to begin searching for two successive ASCII sync characters

Format of the status register



Example 3

- A typical program sequence which uses programmed I/O to input 80 characters from the 8251A, whose data buffer register's address is 0050, and put them in the memory buffer beginning at LINE.

```
        MOV     AL,00110101B            ;ENABLE TRANSMITTER AND RECEIVER
        OUT     51H,AL                  ;AND CLEAR ERROR BITS
        MOV     DI,0                    ;INITIALIZE INDEX
        MOV     CX,80                   ;PUT COUNT IN CX
BEGIN:  IN      AL,51H
        TEST    AL,02H
        JZ      BEGIN
        IN      AL,50H                  ;INPUT CHARACTER AND
        MOV     LINE[DI],AL             ;PUT IN LINE BUFFER
        INC     DI
        IN      AL,51H                  ;CHECK ERROR
        TEST    AL,00111000B            ;BITS AND
        JNZ     ERROR                   ;IF NO ERROR IS FOUND
        LOOP    BEGIN                   ;CONTINUE INPUTTING
        JMP     SHORT EXIT
ERROR:  CALL    NEAR PTR ER_ROUT        ;ELSE CALL ERR_ROUT
EXIT:    .
         .
         .
```

Example 4

- 1000 000 0 : data register address: xx80h
- 1000 000 1: control or status register address: xx81h
- Mode word:
  - 2 stop bits. no parity, 8 bit characters. Baud rate factor of 16 (1200 Kbps)
  - 1110 1110 =EEh
- Command Word:
- 0001 0101 = 15h ; enable TxRDY and RxRDY and reset all flags first

```
INIT8251:MOV AL,0EEh
OUT 81h, AL
MOV AL, 15h
OUT 81h, AL
```

Initialize the Mode Word andCommand Word

```
CHKRX:IN AL,81h
ROR AL,1
ROR AL,1
JNC CHKRX
```

Receive Ready?

```
IN AL,80h
NOT AL
MOV BL,AL
```

If Ready get data

```
CHKTX:IN AL,81h
ROR AL,1
JNC CHKTX
OUT 80h,AL
JMP CHKRX
```

Send data if the T buffer register is available