UNIT-5

Serial Data Transfer Schemes:-

5.1) Serial Data Communication:-

The data within micro computer is transferred in Parallel as it is a fastest way to transfer. For transferring a data over long distances, however Parallel data transmission requires too many wires. Therefore data can be sent long distance is usually sent on a single wire (or) Pair of wires.

Converted from Parallel form to serial form so that it can be

Serial data received from a distant source is converted to Parallel form ~~a distant source is converted~~ so that it can easily be transferred on micro computer buses.

Classification of Serial Data transfer:-

3 types of Data transmission modes
- 1.) Simplex
- 2.) Half Duplex
- 3). Full-duplex.

(1) Simplex:- The data line can transmit data only in one direction.

Eg:- (1) Transmission from micro Computer to Printer

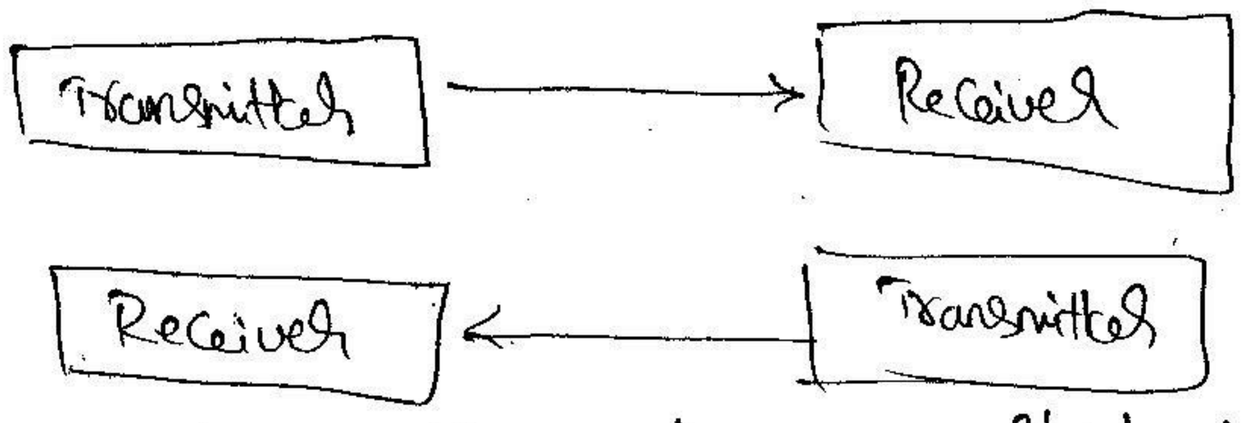
(2) " for Earthquake sensor to radio station.

(2) Half Duplex:- Data can takes Plce either direction between two systems, but can only occur in one direction at a time.

In this mode, each device can both transmit (or) receive but not at the same time.

Eg:- Walkie talkie system.

(3) Full Duplex :- In this mode of transmission, both systems can send and receive data at the same time.



The full duplex mode is like a two way street with traffic flowing in both direction at the same time.

Eg:- Telephone network.

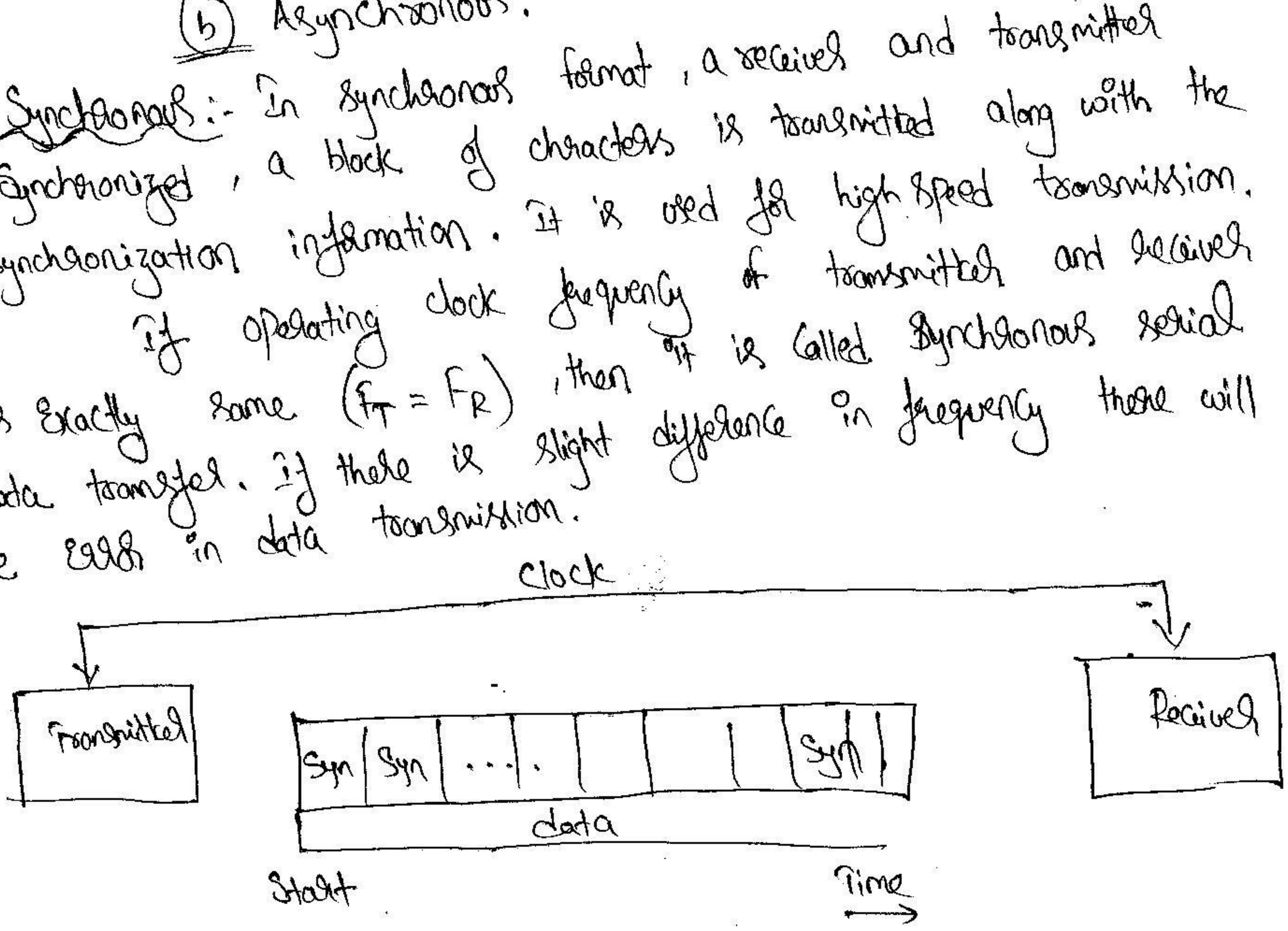
5.2) Serial Data transmission formats:-

→ The transmission formats concerned from issues such as Asynchronization, direction of data flow, speed, errors and medium of transmission.

→ There are 2-methods of serial data transmission.

(a) Synchronous
(b) Asynchronous.

(a) Synchronous :- In synchronous format, a receiver and transmitter synchronized, a block of characters is transmitted along with the synchronization information. It is used for high speed transmission. If operating clock frequency of transmitter and receiver is exactly same ($f_T = F_R$), then it is called synchronous serial data transfer. If there is slight difference in frequency there will be errors in data transmission.

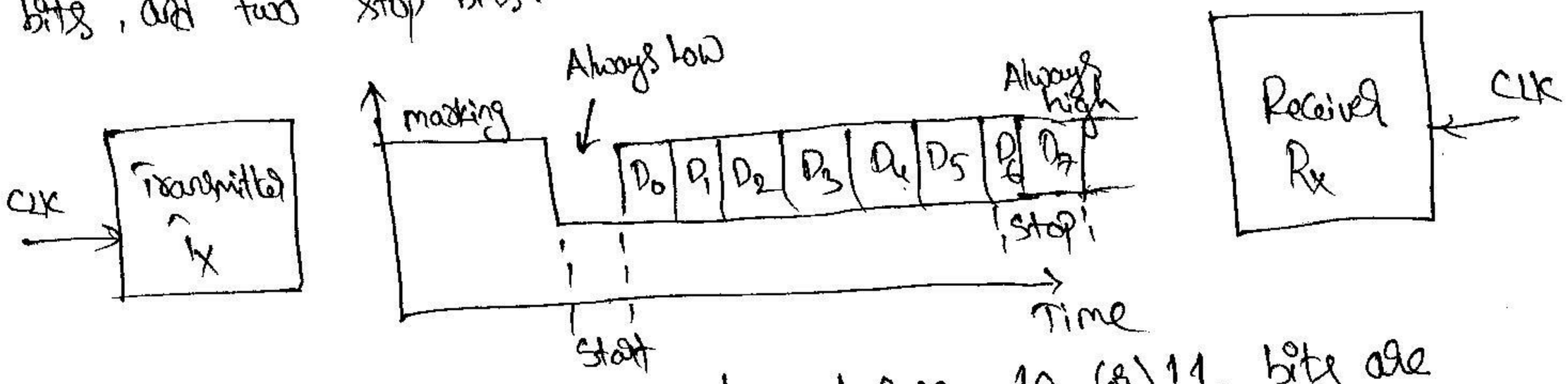


(b) Asynchronous :-

It is character oriented when no:- data is being sent, the signal line is in a constant high (or) marking state.

Transmission begins with one start bit, followed by a character and one (or) two stop bits. This is also known as framing.

The transmission start bit is always at Low (0) and stop bit is always at high (1). The transmission of 11 bits for an ASCII character in asynchronous format i.e. one start bit eight character bits, and two stop bits.



The efficiency of this format is low, because 10, (or) 11 bits are required to transmitt a 7 bit data word such as ASCII character

The Asynchronous format is generally used in low speed transmission.

5.3) 8251 USART (universal Synchronous Asynchronous Receiver & Transmitter):-

The 8251 PIC (Programmable Communication Interface) is a Programmable chip designed for synchronous & Asynchronous serial data communication. Packaged in 28 Pin DIP.

The chip converts the parallel data into a serial stream of bits suitable for serial transmission.

It is able to receive a serial stream of bits and convert it into parallel data types to be read by microprocessor.

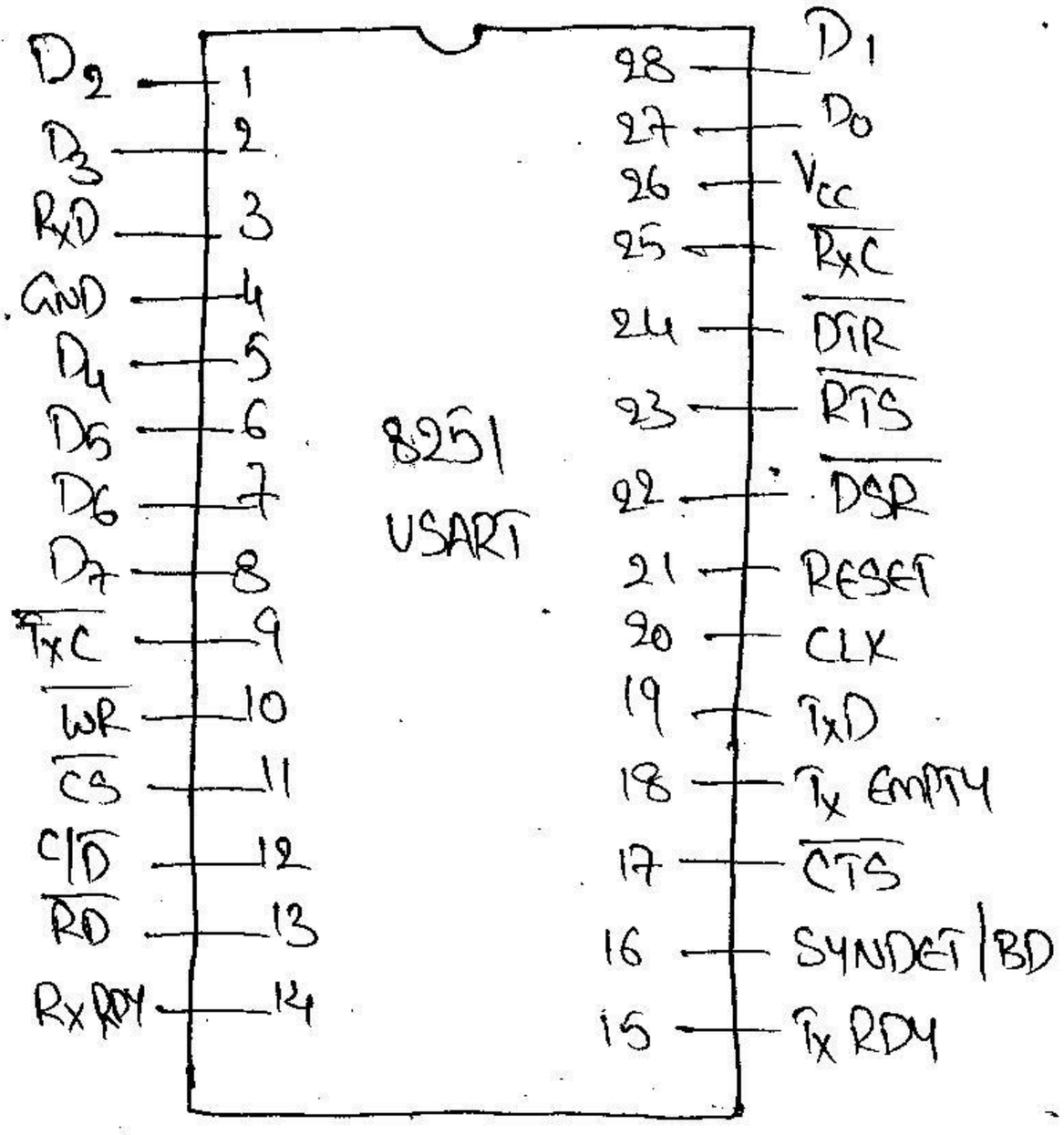
→ It has in built baudrate generator
→ It allows full duplex transmission and reception.
→ It provides double buffering of data both in the Tx section & Rx section

→ It Provides Error detection logic, which detects Parity, over run and Programming Errors

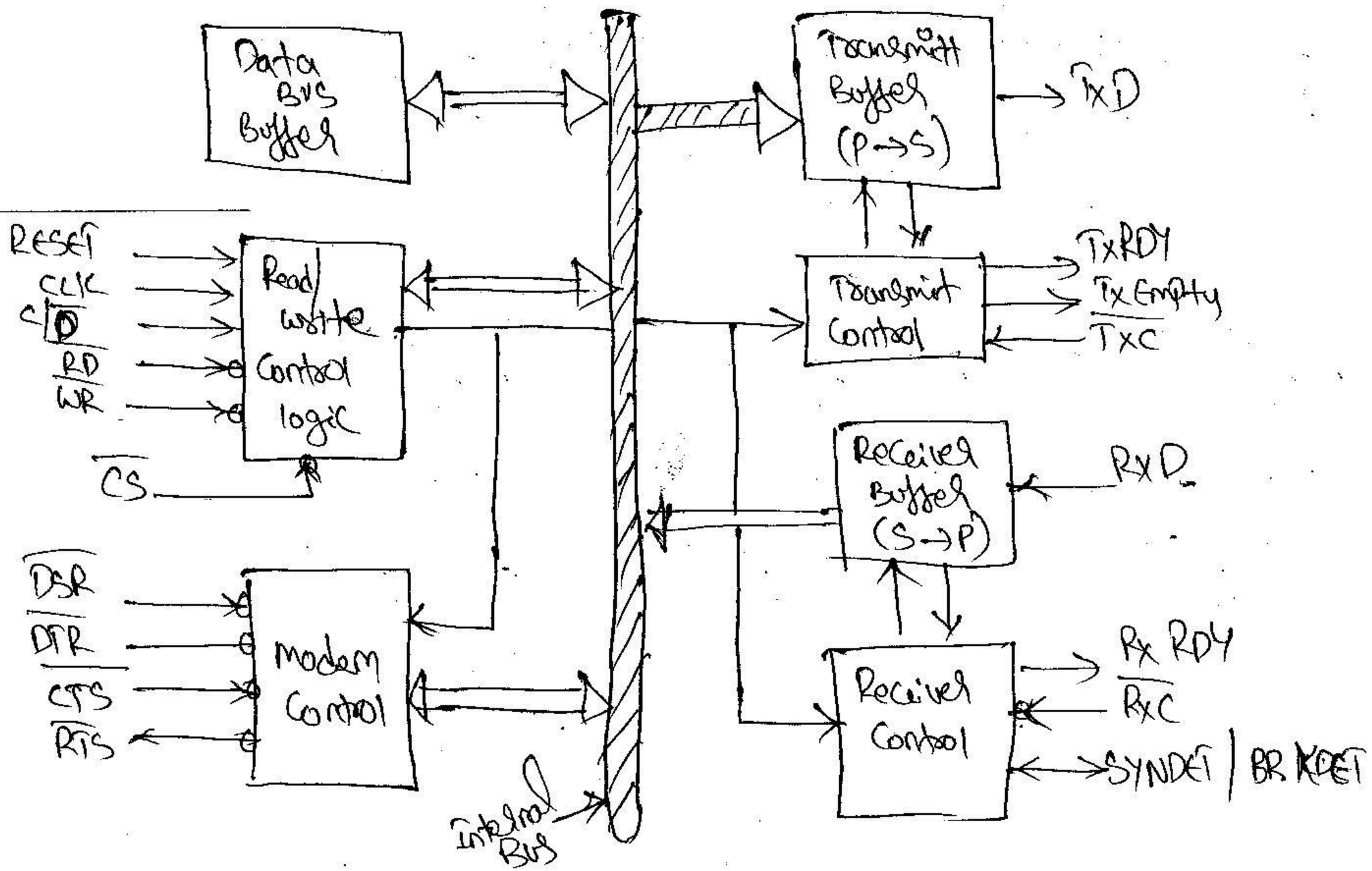
→ It is fabricated in 28 Pin DIP and its all i/p and o/p are TTL Compatible.

→ It has modem Control logic, which supports basic data & Control Signals.

Pin diagram:-



8251 USART Block diagram:-



The block diagram of 8251 includes five sections.

1) Data Bus Buffer
2) Read/write Control logic
3) Modem Control
4) Transmitted section
5) Receiver section.

1) Data Bus Buffer :- It interfaces the internal bus of 8251 with system bus. along with the data, control word, command word, and status information. are also transferred through the data bus buffer.

2) Read/write control logic :- Controls the operation of the peripheral depending upon operations initialised by CPU.

It has 6 - i/p signals, control logic and 3-buffer Registers, data registers, control Register & Status registers.

(i) $\overline{CS}$ → chip select :- When signal goes low, the 8251 is selected by CPU for communication. If it is high, no Read/write operations performed.

(ii) C/$\overline{D}$ Control Data :- When the signal is high, the control reg (or) status reg is addressed, when it is low, data buffer is addressed.

The control reg and status reg are differentiated by $\overline{WR}$ and $\overline{RD}$ signal respectively.

(iii) $\overline{WR}$ - write :- When this signal goes low, the CPU either writes in the control Reg (or) sends output to the data buffer.

(iv) $\overline{RD}$ - Read :- When signal goes low, the CPU either reads a status from status Reg (or) Accept data from the data buffer.

(v) RESET :- A high on this Resets the 8251 and forces into Idle mode

(vi) CLK :- This is the clock i/p, usually connected to system clock.

The i/p frequency should be at least 30 times greater than the receiver (or) transmitter data bit transfer Rate.

The Control Reg is 16-bit Reg for a Control word of two independent bytes, first byte is mode instruction, second " " Command " } Address as o/p Port

*) Status Reg checks the ready status of a Peripheral. It is address as i/p Port.

*) Data Buffers Reg can be addressed as i/p Port and o/p Port.

$\overline{CS}$	C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	function.
0	1	1	0	CPU writes instructions in Control Reg
0	1	0	1	" Reads status from Status Reg
0	0	1	0	" o/p's data to the Data buffer
0	0	0	1	" accepts data from data buffer
1	x	x	x	USART is not selected.

3) Modem Control:- The Modem Control Unit handles the modem handshake signals to Coordinate the Communication b/w the modem and the USART 2 - i/p signals and 2 - o/p signals are associated with modem control section.

(i) $\overline{DSR}$ - Data set Ready:- This i/p used as a general Purpose one bit inverting i/p port this status is checked by CPU by status read operation.

This is used to check if the data set is ready when Communicating with modem.

(ii) $\overline{DTR}$ - Data terminal Ready:- o/p is used as General Purpose one bit inverting o/p port. This is used to indicate that the device is ready to accept data when 8251 is Communicating with modem.

(iii) $\overline{RTS}$ - Request to Send :- It is also used as gretal Purpose one bit inverting o/p Port that can be Programmed to low to indicate the modem that the receiver is ready to receive a data byte from the modem.

(iv) $\overline{CTS}$ - Clear to Send :- A low on this pin enables the 8251 to transmit serial data if TxEN bit in the Command byte is '1'.

4) Transmitter Section :- The transmitter accepts Parallel data from CPU and Converts them in to serial data.

It has 2-Reg A buffer Reg to hold 8-bits and o/p Reg to Convert 8-bits in to stream of serial bits.

The CPU writes a byte in the buffer reg ~~are~~ ~~transferred~~ When ever the o/p reg is Empty, the Contents of buffer reg are transferred to the o/p reg. This section transmitt data on to the TxD Pin with appropriate framing bit.

(i) TxD → Transmit Data :- Serial bits are transmitted on this line with framing bits (start & stop) and Parity bit ... etc.;

(ii) $\overline{TxC}$ → Transmitter clock :- This i/p signal Controls the rate at which bits are transmitted by the USART the clock freq can be 1, 16, (or) 64 times the band.

(iii) TxRDY :- Transmitter Ready :- It is a o/p signal, when it is high it indicates that the buffer reg is empty and the USART is ready to accept a byte. This signal is reset when a data byte is loaded in to the buffer.

(iv) TXE - Transmitter Empty :- It is an o/p signal logic '1' on this line indicates that o/p reg is Empty. This signal is reset when a byte is transfered from the buffer to the o/p register.

5) Received Section :- The received accepts serial data on the RxD line from a peripheral and converts them into parallel data. This section has two registers

(*) The receiver i/p Register
(*) The buffer Register

When the RxD line goes low the control logic assumes it is a START bit, waits for half a bit time, and samples the line again. If the line is still low, the i/p registers accepts the following Bits, from a character and back it into the buffer register.

In Asynchronous mode two i/p signals and one o/p signal are necessary.

(i) $\overline{RxD}$ - Receiver data :- Bits are received serially on this line and converted into a parallel byte in the receiver i/p Reg.

(ii) $\overline{RxC}$ - Receiver clock :- The clock signal that controls rate at which bits are received by the USART. In Asynchronous mode clock can be set 1, 16, (or) 64 times the baud.

(iii) R_xRDY - Received Ready :- It is an o/p signal. It goes high when the USART has a character in the buffer Reg and is Ready to transfer it to the CPU.

(iv) SYNDET/BD (Sync - Detect/Break detect) :- It has 2-uses, when the device is operating in Asynchronous mode, this Pin will go high if the serial data i/p line RxD stays low for more than 2-character times. The signal then indicates an intentional break in data transmission (or) a break in signal line. When programmed for Synchronous data transmission this Pin will go high.

5.4) Initializing an 8251 (8251 Control words & Status words)

To initialize an 8251, must send a Mode word and then a Command word to Control register address for the device.

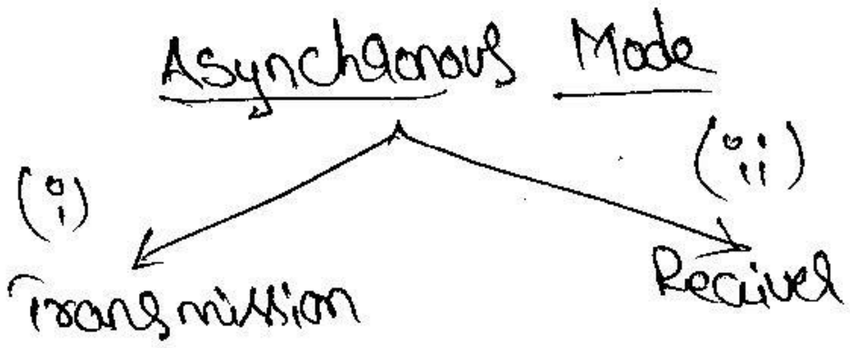
The Status word Provides the information Concerning register status and transmission Errors and this word is read from Control reg address.

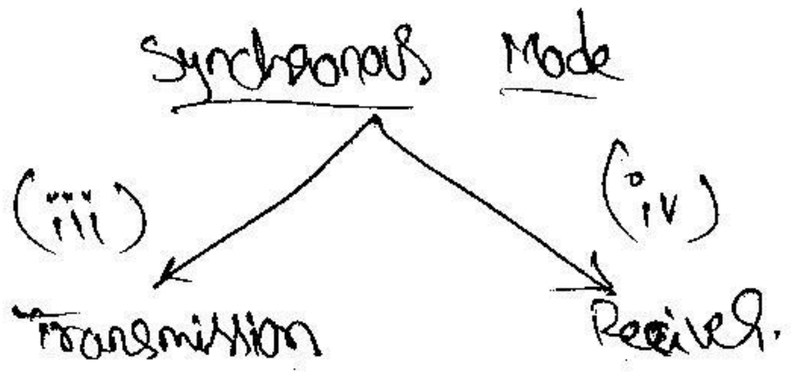
The Control word is divided in 2-formats

(a) mode word
(b) Command word.

(a) **Mode word**:- It is an operational characteristics of 8251A. After internal (reset Command) (&) external (reset input pin) reset, this must be written to Configure the 8251 as Per required operation.

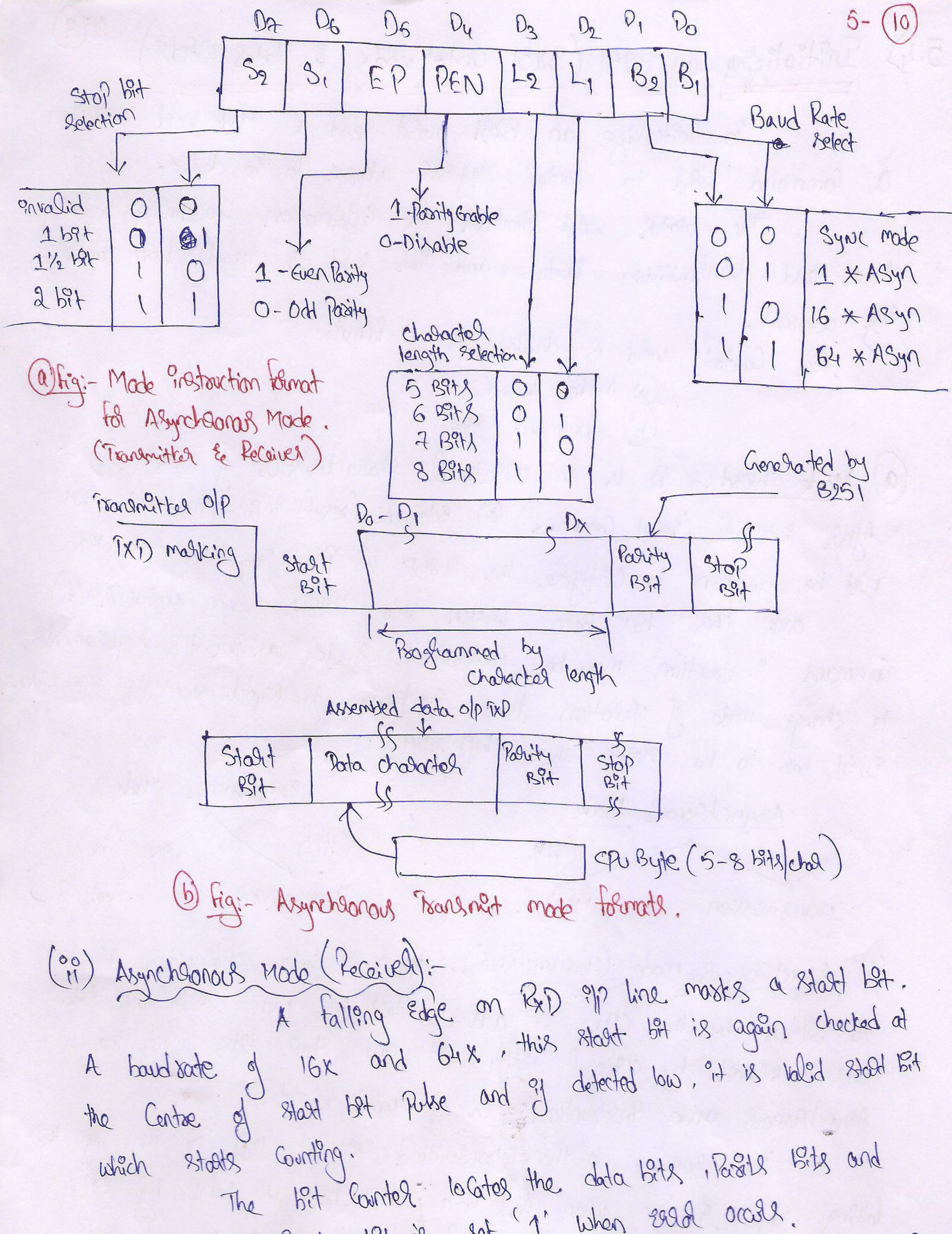
once this has been written into 8251A Sync character (&) Command instruction may be Programed. further as Per the requirements. to change mode of operation from Synchronous to Asynchronous. (&) vice-versa. 8251 has to be reset using chip select.





(i) **Asynchronous mode (transmission)**:- When a data character is sent to 8251A by the CPU, it adds start bits, Pich to Serial data bits. followed by optional Parity bit and stop bits using the Asynchronous mode instruction Control word format.

This sequence is then transmitted using TXD o/p pin on the falling Edge of $\overline{TXC}$. when no data character is sent by CPU to 8251A. The TXD o/p remains 'high' if a break has not been detected.



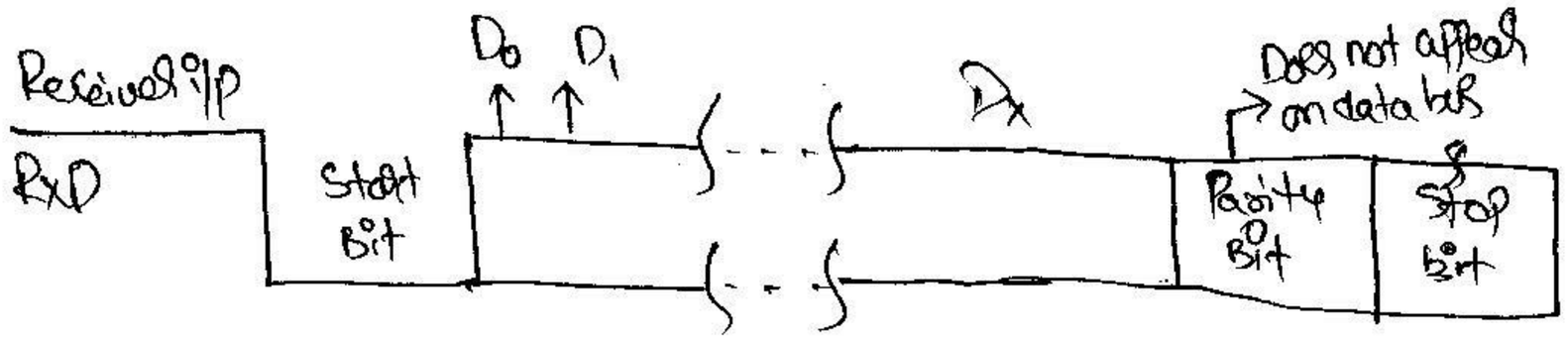
(a) Fig:- Mode instruction format for Asynchronous Mode. (Transmitter & Receiver)

(b) Fig:- Asynchronous Transmit mode formats.

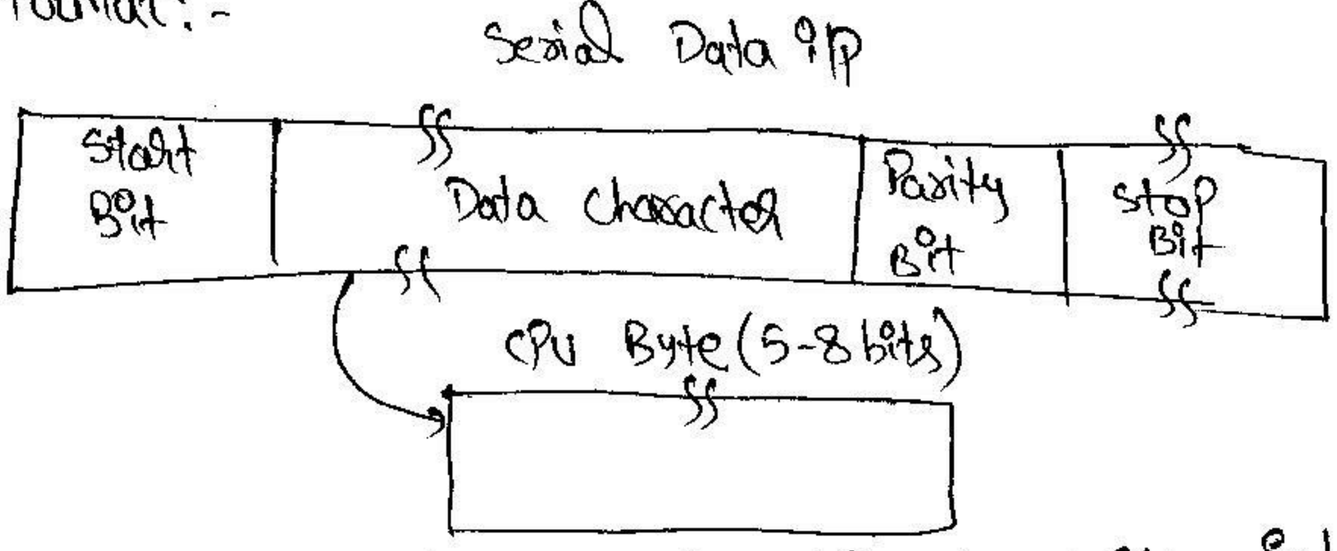
(ii) Asynchronous mode (Receiver):-

A falling edge on RxD i/p line marks a Start bit. A baud rate of 16x and 64x, this start bit is again checked at the Centre of start bit Pulse and if detected low, it is valid start bit which starts Counting.

The bit Counter locates the data bits, Parity bits and stop bit. If Parity bit is set '1' when error occurs. If a low level is detected as the stop bit, the framing error flag is set.



Receiver format:-



The receiver requires only one stop bit to mark end of data bit string. regardless of stop bit programmed at transmitting end. This 8-bit character is then loaded into parallel I/o buffer of 8251A. RxRDY Pins is then raised high to indicate to the CPU that a character is ready for it. If the previous character has not been read by the CPU, the new character replaces it, and overrun flag is set indicating that previous character is lost.

This error flags can be cleared using Reset instruction.

(iii) Synchronous Mode (transmission):- The TxD o/p is high until the CPU sends a character to 8251A which usually is a SYNC character. When $\overline{CS}$ line goes low, the first character is serially transmitted out. All the characters are shifted out on falling edges of $\overline{TxC}$, Data is shifted out at the same rate as $\overline{TxC}$, over TxD output line. If the CPU buffer becomes empty, the SYNC character (s) characters are inserted in data stream over TxD output.

The TXEMPTY Pin is raised high to indicate that the 8251A is empty (ie no byte to transmit) and transmitting SYNC characters. The TXEMPTY Pin is reset, automatically when data character is written to 8251A by the CPU.

(iv) Synchronous Mode (Received):- In this mode, the character synchronization can be achieved internally (or) externally. If it is programmed then "ENTER HUNT" command should be included in the first command instruction word written in to the 8251A. The data on RxD Pin is sampled on rising edge of the RxC.

The 8251A is programmed for two sync characters, the subsequent received character is also checked. when both characters matches the hunting stops.

The SYNDET pin is set high and is reset automatically by a status read operation. If a Parity bit is programmed the SYNDET signal will not go as high as the middle of Parity bit.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
SCS	ESD	EP	PEN	L_2	L_1	0	0

SCS: Single char. Sync — 1 - Single char Sync; 0 - Double char Sync

ESD: External Sync Detect — 1 - SYNDET is i/p; 0 - SYNDET is o/p

EP: 1 - Even Parity generate and check; 0 - Odd Parity

PEN: 1 - Parity Enable; 0 - Disable

L_2, L_1: character length

L_2	L_1	
0	0	5 Bits
0	1	6 Bits
1	0	7 Bits
1	1	8 Bits

fig:- Synchronous Mode instruction format.

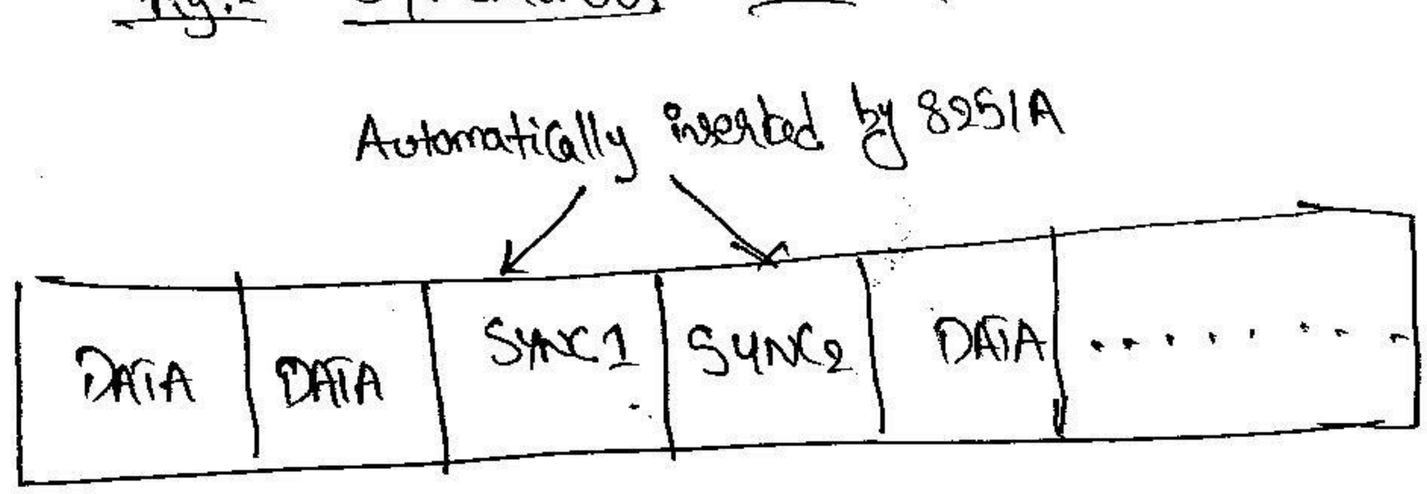


fig:- TXEMPTY Signal and SYNC Characters.

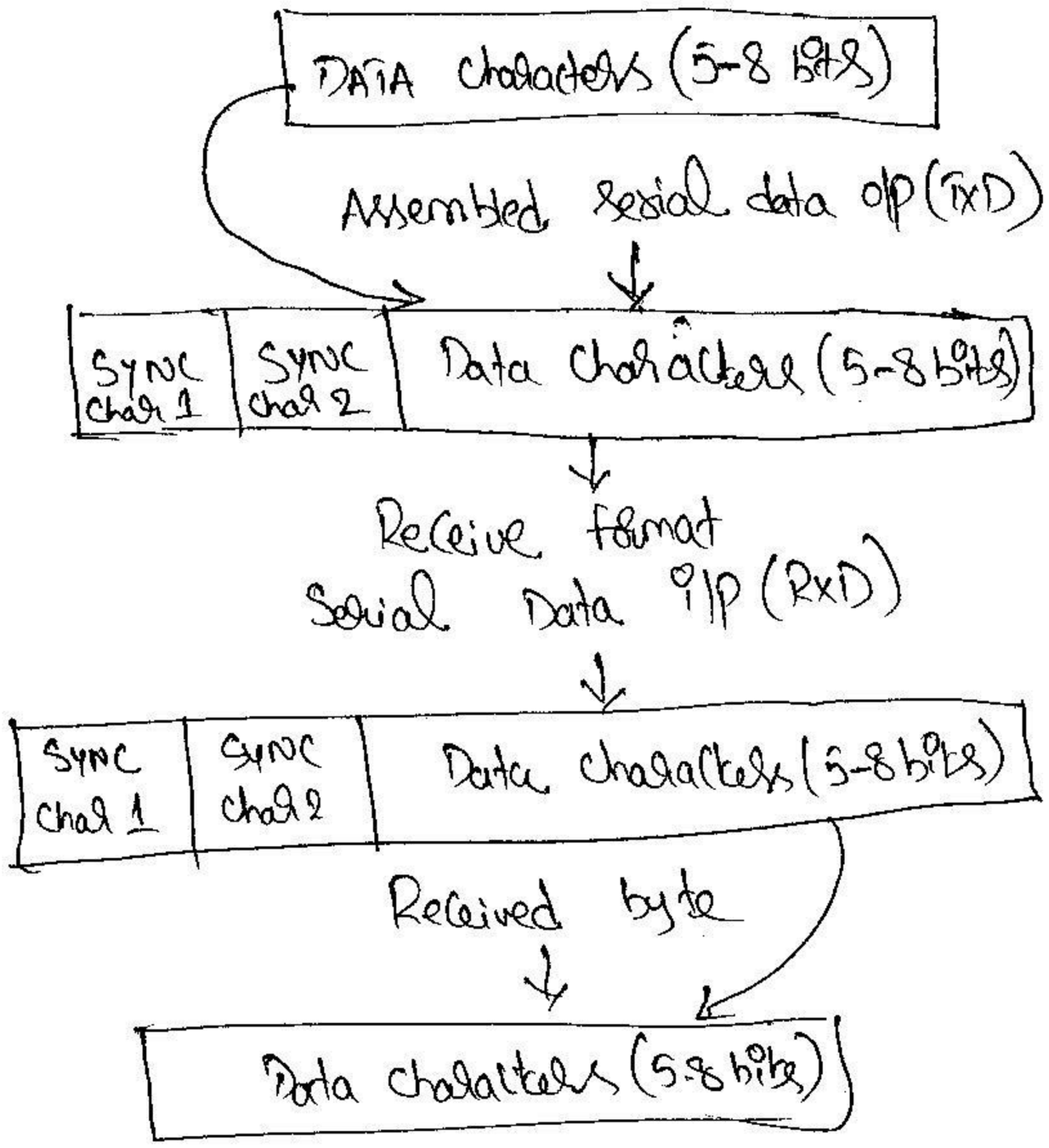
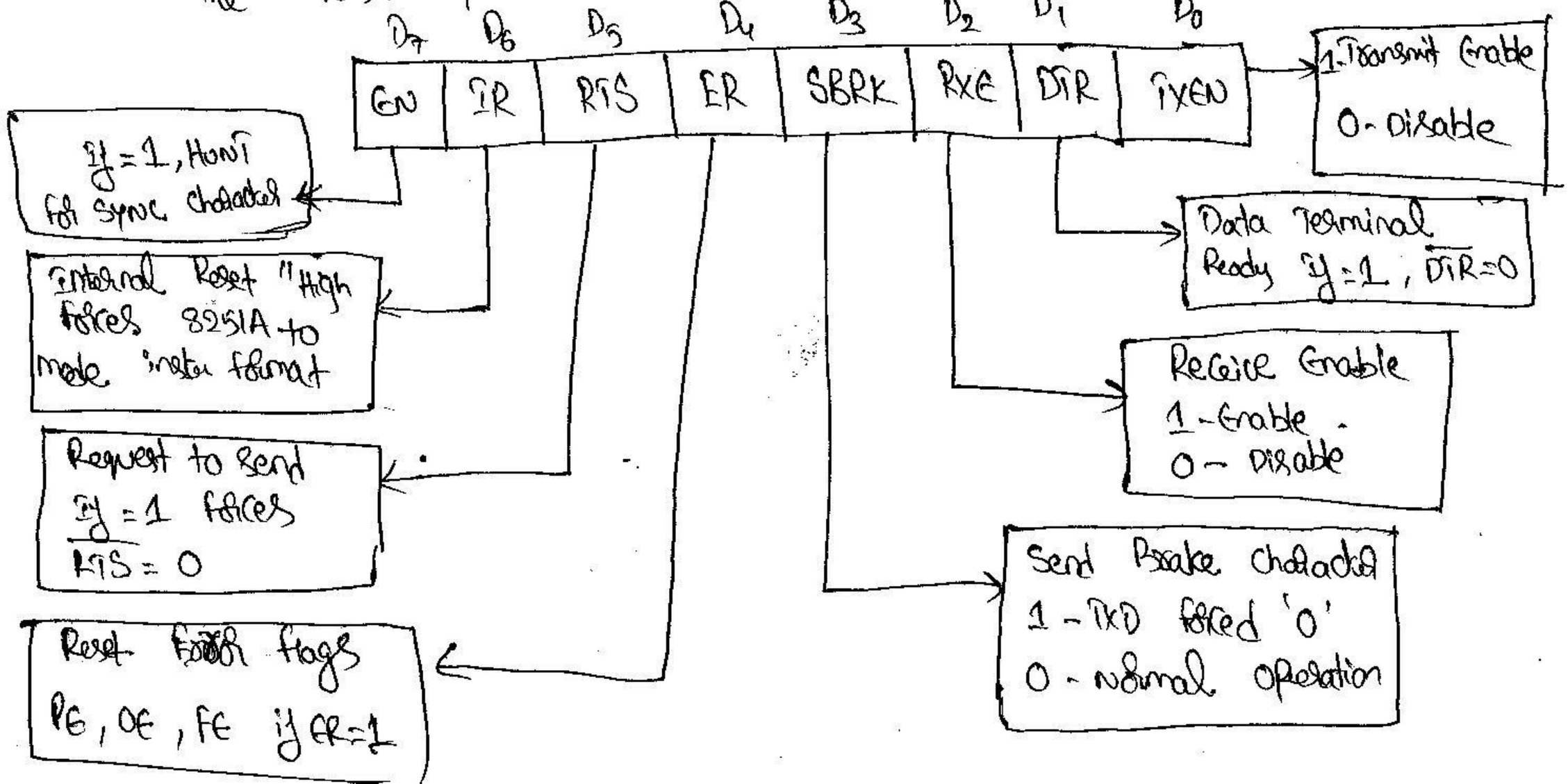


fig:- Data formats of Synchronous Mode.

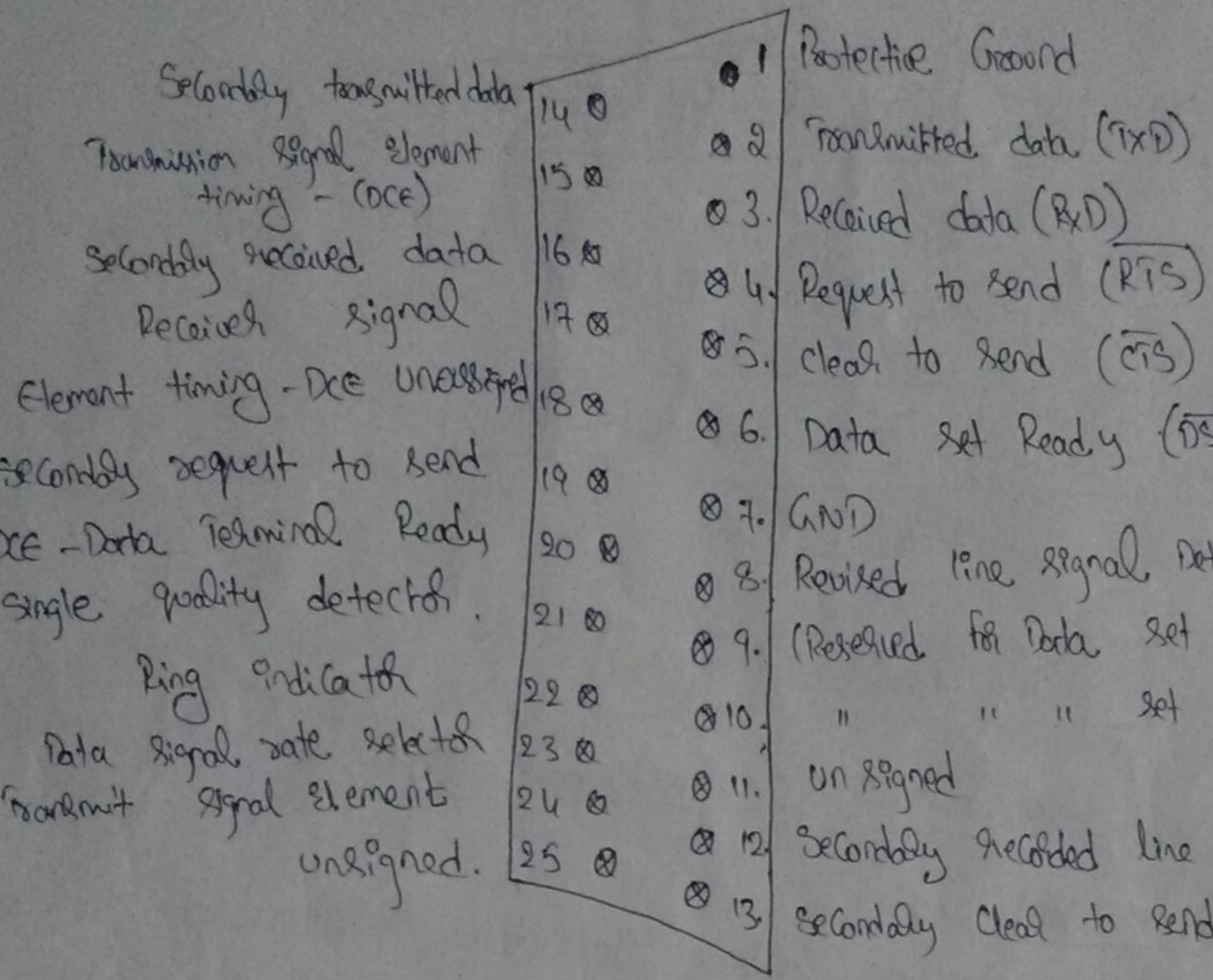
(b) Command Word :-

This instruction controls the actual operation of the selected format like Enable transmit / receive, error reset and modem control. once the mode instruction written into 8251A and synchronous character inserted internally by 8251A, all further control words written with $C/\overline{D} = 1$ will load Command instruction.

The reset operation returns 8251A back to mode instruction format.



5.5) Pin Structure of RS-232C :-



Pins 2, 3, 7 are used for minimum interface b/w Computer & Peripheral.

the values of logic 1 & logic '0' of RS-232 device is

logic 1 = -3V to -15V
logic 0 = +3V to +15V

Voltage & Current specifications of RS-232C

logical '0' = +3V
logical '1' = -3V
maximum length = 50ft
maximum speed = 20k band
Driver o/p voltage for open circuit = 25V max
Drive o/p " range = 5→15V
" " off resistance = 300Ω
Received i/p threshold = -3 to +3V
" " Resistance for (5-15V) = 3k-7kΩ

RS 232C to TTL interfacing:-

It is divided in to 4-grops.

(a) Data signals, (b) Control signals (c) timing signals (d) ground

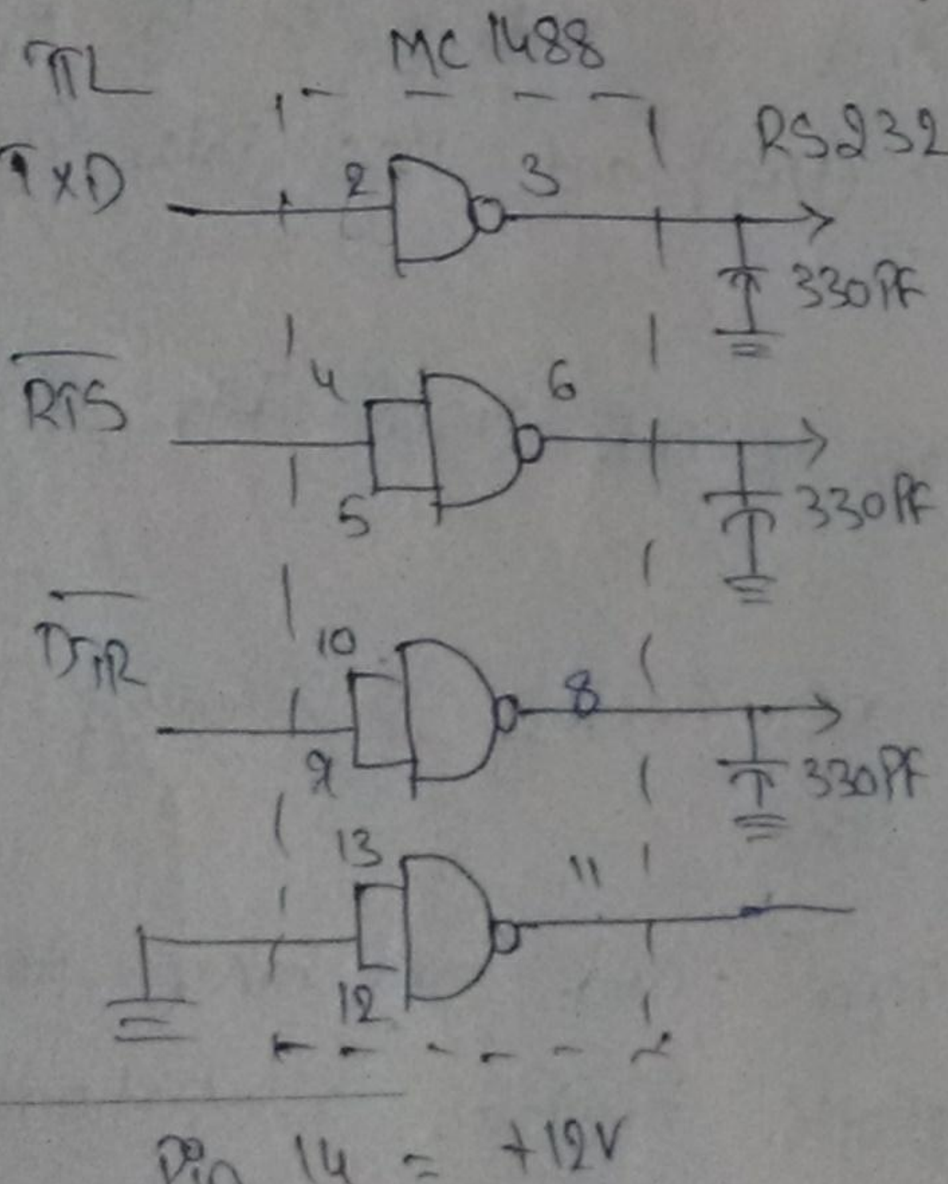
(*) The 8251 USART has TTL signals on its o/p.

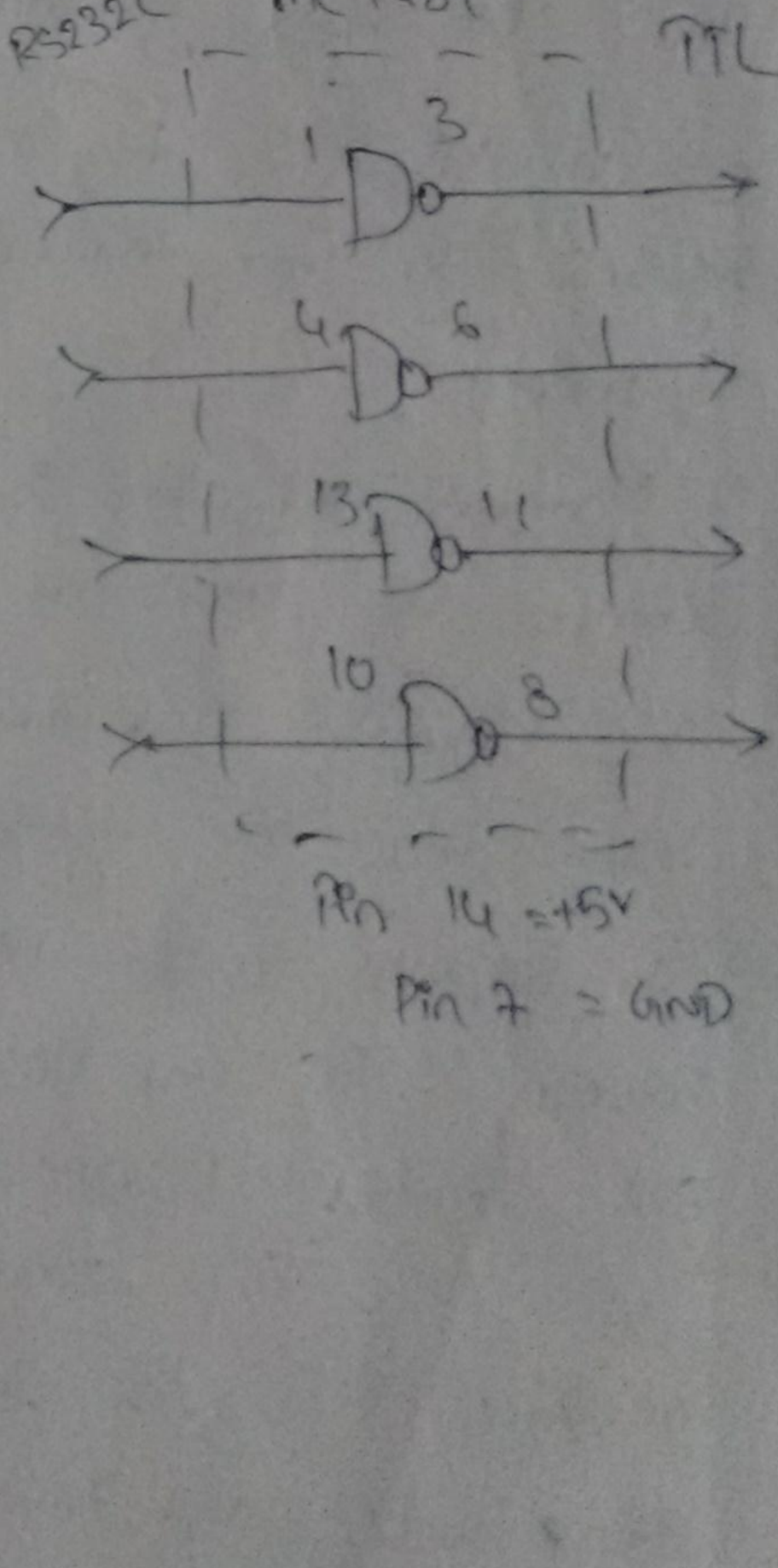
(*) The TTL signals of 8251 are not compatable with data signals of RS 232C,

(*) The standard way to interface RS232 and TTL levels with MC 1488 Quad TTL to RS232 drivers and MC 1489 quad RS232 to TTL receivers.

(*) The MC 1488 require + & - supplies,

(*) The MC 1499 " only +5V



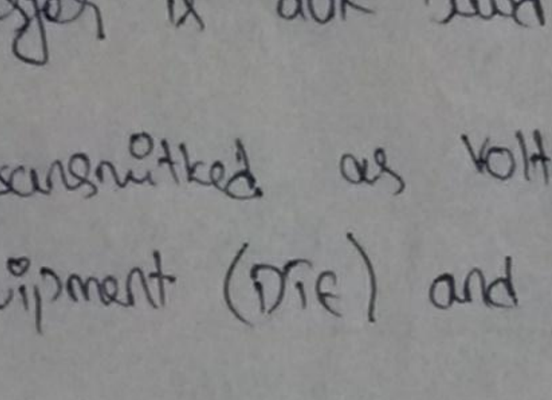


Pin 14 = +12V
Pin 1 = -12V
Pin 7 = GND.

Capacitors are used to reduce the noise. &

Connector:- 9-Pin DIN Connector

1 Pin - $\overline{DCD}$ (Data Carrier Detect)
2 Pin - RxD
3 Pin - TxD
4. Pin - $\overline{DTR}$
5. Pin - GND
6. Pin - $\overline{DSR}$
7. Pin - $\overline{RTS}$
8 Pin - $\overline{CTS}$
9 Pin - Ri (Ring indicator).



→ Data Terminal Equipment
→ Data Communication Equipment

length of Cable in between DTE & DCE can be maximum 50 feet and max speed of data transfer is 20K Baud."

*) In RS 232 data are transmitted as voltage. It is defined in reference to Data terminal Equipment (DTE) and DCE (Data Communication Equipment)

*) Modems and other Equipments used to send serial data over long distance are known as DCE

*) The terminals and computers that are sending or receiving the serial data are referred to as DTE.

*) In response to the need for signal and hand shake standards between DTE and DCE, the electronic industries association developed EIA Standard RS232C.

This standard describes function of 25 signals and handshake pins for serial data transfer. It also describes the voltage levels and impedance levels, rise and falling time, Maximum bit rate, and max-Capacitance for these signals lines

*) In this 25 pins are not need, a 9-Pin DIN Connector is used. for systems.

1) connector:-

1 pin — $\overline{DCD}$ (data carrier detect)

2 pin — RXD

3 pin — TXD

4 pin — $\overline{DTR}$

5 pin — GND

6 pin — $\overline{DSR}$

7 pin — $\overline{RTS}$

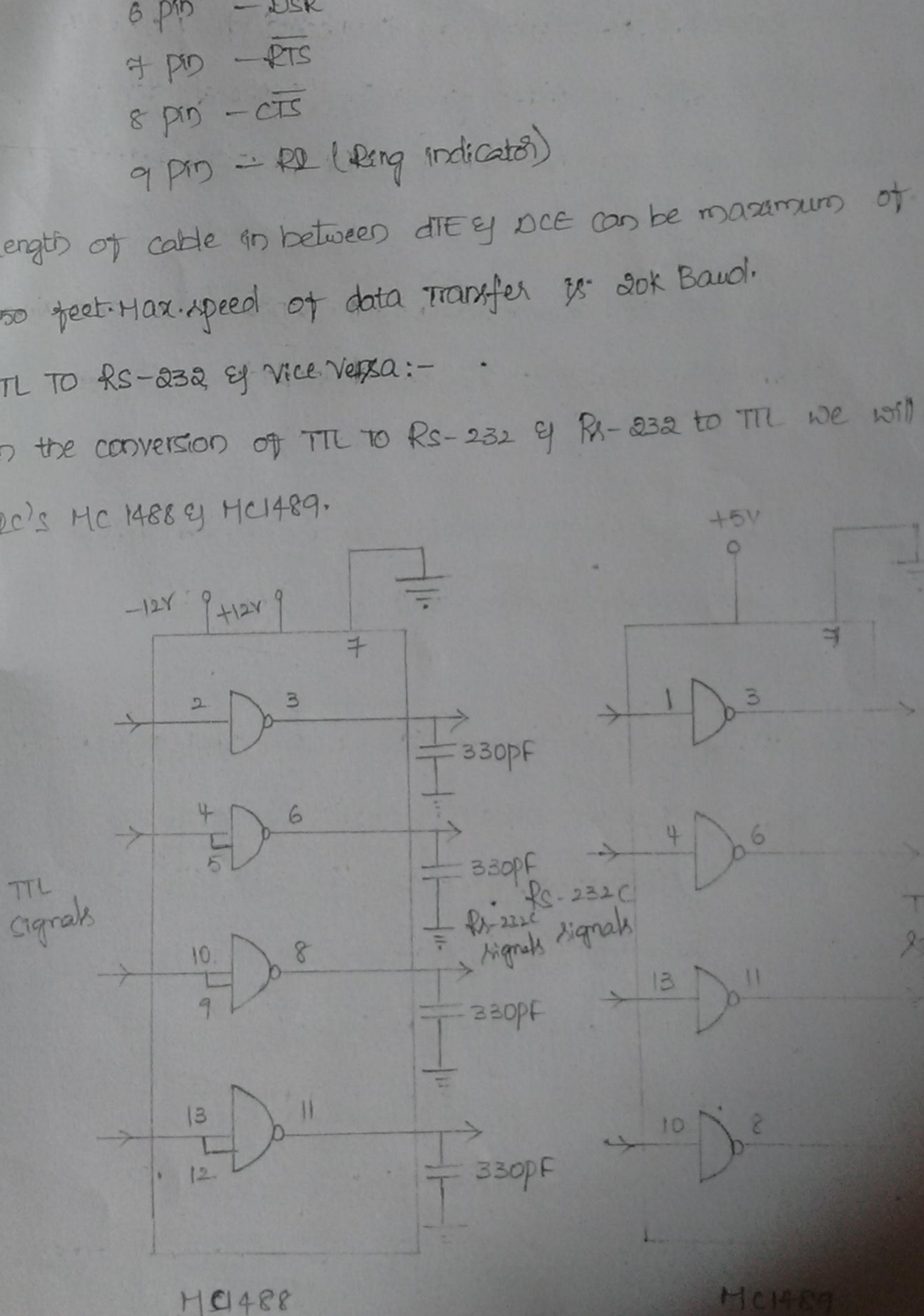
8 pin — $\overline{CTS}$

9 pin — RI (Ring indicator)

length of cable in between DTE & DCE can be maximum of 50 feet. Max. speed of data transfer is 20k Baud.

TTL TO RS-232 & Vice Versa:-

In the conversion of TTL to RS-232 & RS-232 to TTL we will use IC's MC1488 & MC1489.



MC 1488 uses postive & negative supply. 330pF capacitors are normally connected by user at the output to reduce noise. Cable length is upto 50 feet.

USB [Universal Serial Bous developed by INTEL]

Many of the peripherals such as scanners, digital cameras, video devices, printers, pen drives are connected to pc. with RS-232C we can able to interface these peripheral devices but connected to Host (PC) pc. Each of these peripherals had complicate installation procedures.

we need to have ADD-ON-Cards too. The speed was also slow.

To overcome these problems USB was developed.

features of USB:-

* USB has a common interface to all types of peripherals.
* Using USB, upto 127 peripheral devices can be connected to Host (PC)
* Individual USB cables can be as long as 5 metres. USB Hubs, devices can be upto 3 meters from host.
* A USB cable has two wires of power (+5V & GND) and a twisted pair of wires to carry the data.

↳ USB 1.1 supports two high speed protocols.

1> Asynchr. PROTOCOL operates at 1.5 Mbps.

2> Synchro. (Isochronous) protocol operates at 12 Mbps.

↳ USB 2.0 supports 40 times the speed of USB 1.1 ie. 480 Mbps.

it actually supports 3 speeds (1.5, 12 & 480 Mbps)

USB 2.0 interfaces the productivity by allowing multiple pc 9 applications simultaneously.

↳ It supports of variety of operating system.

USB connector:-

It is a 4 pin connector. 4 signals are +5V, GND, -data, +data
(1) (4) (2) (3)

USB Hub:- It will have four (or) more USB ports. By using it, user can able to interface many devices to computer.

USB protocols:-

USB is a serial interface. The information sent is organised in the form of packets & frames. This is done to provide error checking, flow control & to synchronise the devices.

Each USB transaction consists of the following.

1↳ Token packet

2↳ data packet

3↳ Handshake packet

4↳ start of frame packet.

Token packet:- It is generated by host to describe what is to follow & whether the data transaction will be read (or) written

↳ data packets carries the data bits.

↳ Handshake packet reports if data (or) token was received successfully. (or) if the end point is stalled (or) not available to accept data.

↓

(indicates that device is unable to receive (or) transmit data & it requires host intervention)

H
&

2 pin → RXD
3 pin → TXD
4 pin → $\overline{DTR}$
5 pin → GND
6 pin → $\overline{DSR}$
7 pin → $\overline{RTS}$
8 pin → $\overline{CTS}$
9 pin → RI [Ring Indicator]

Length of cable in between dte & DCE can be max of 50 Feet Max speed of Data T/F is 20K Baud.

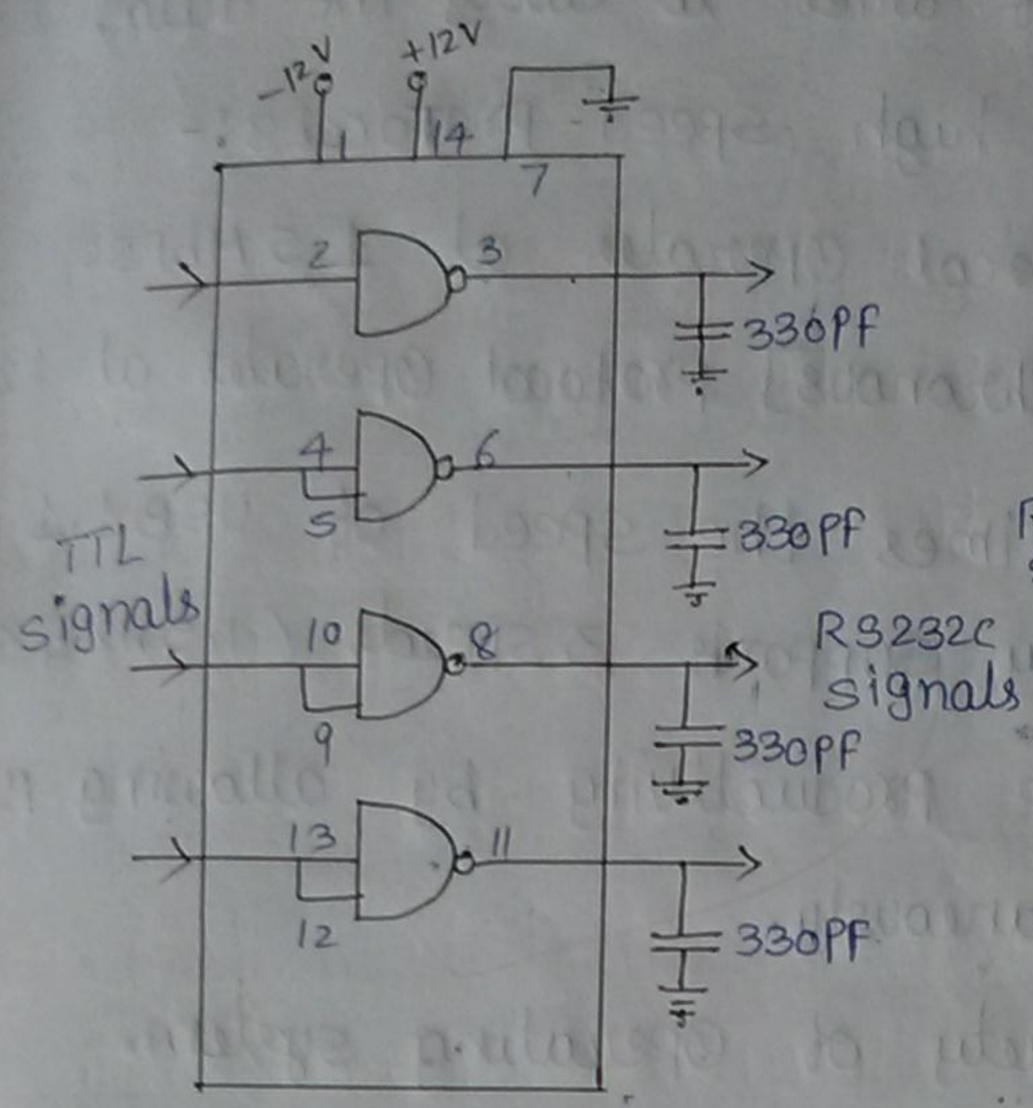
8251 INTERFACING :-

TTL to RS 232C & Vice Versa :-

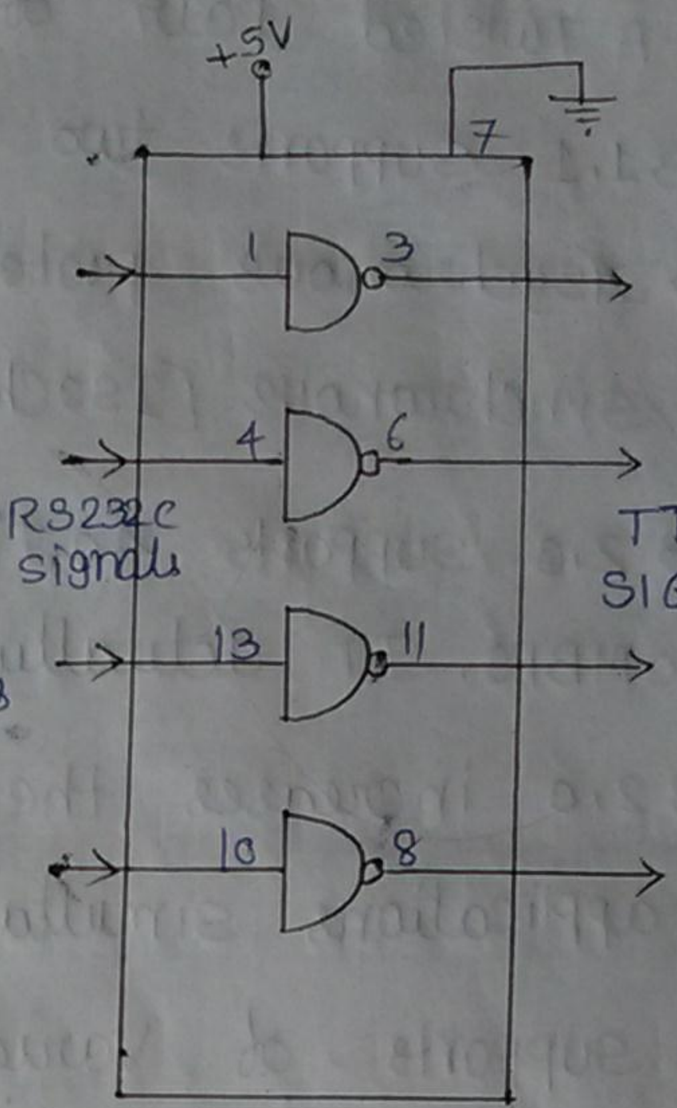
In the Conversion of TTL to RS 232C & RS232C to TTL we will use Ic's MC 1488 & MC1489.

MC 1488 uses +Ve & -Ve supply. 330Pf capacitors are normally connected by user at the output to reduce

noise cable length is upto 50 Feet.



MC1488
[TTL - RS232C]



MC1489
[RS232C - TTL]

USB :- Universal serial BUS Developed by intel.

Many of the peripherals such as scanners, digital cameras, video devices, printers, pen drives are connected to PC. with RS-232C we can able to interface these Peripheral devices to PC. each of these peripherals had complicated installation procedures. we need to have ADD-ON cards too. The speed was also slow. To over come these problems USB was developed.

Features of USB :-

* USB has a common interface to all types of peripherals.
* Using USB, upto 127 Peripheral devices can be connected to Host → [PC]
* Individual USB cables can be as Long as 5 meters. USB HUBs, devices can be upto 30 Meters from Host.

* A USB cable has two wires of power [+5V & GND] & A Twisted pair of wires to carry the data.
* USB 1.1 supports two high speed protocols:-
  1) Asychronous protocol operates at 1.5 Mbps.
  2) synchronous [Isochronous] protocol operates at 12 Mbps.
* USB 2.0 supports 40 Times the speed of USB 1.1 i.e 480 MBPS. It actually supports 3 speeds (1.5, 12 & 480 Mbps).
* USB 2.0 increases the productivity by allowing Multiple Pc applications simultaneously.
* It supports of variety of Operating system.

USB connector:- It is a 4-pin connector. 4 signals are +5V, GND, -data, +Data.

USB HUB :- It will have 4 or more USB ports. By using It, user can able to interface many devices to computer.

USB Protocols :-

USB is a serial interface. The information sent is organised in the form of Packets & Frames. This is done to provide Error checking, flow control & to synchronize the devices.

Each USB Transaction consists of the following:

1) Token Packet.
2) DATA Packet.
3) Handshake Packet.
4) Start of Frame Packet.

1] Token Packet :- It is Generated by Host to describe what is to follow & whether data transaction will

be Read (or) written.

2] DATA Packet :- It carries the data bits.

3] Handshake Packet :- It Reports if data or Token was received successfully (or) If the End Point is stalled (or) not available to accept data.

↳ Indicates that device is unable to receive or Transmit data & it requires Host intervention.

4] SOF Packet :- There is one More packet sent by Host every 1ms ± 500 ns on a Full speed bus or every 125 µs ± 0.0625 µ on A high speed bus.

USB Process :-

The Host assigns the address to each device connected to it. This process is called "Enumeration". It also checks what types of data Transfer is Required.

Interrupt Mode ⟶ It is used by device which sends very little data like Mouse (or) Keyboard.

Bulk Mode ⟶ It is used by device which requires data in one big Packet like printer.

Isochronous Mode ⟶ used by streaming devices like speaker. Here data streams b/w Host & device in Real time.

UNIT - 6

INTERNAL ARCHITECTURE OF 8259

8259 consists of 8 functional blocks

↳ Bidirectional data buffer
↳ R/W Logic
↳ control Logic
↳ Interrupt request register
↳ Priority resover
↳ In-service register
↳ Interrupt mask register
↳ Cascade buffer/comparator

Bidirectional data buffer ($D_7 - D_0$)

These 8 bi-directional data bus is connected to system bus of 8086. It is used for writing control words & reading status words of 8259.

8259 sends interrupt type to CPU on these lines.

8259 : Programmable interrupt controller (PIC)

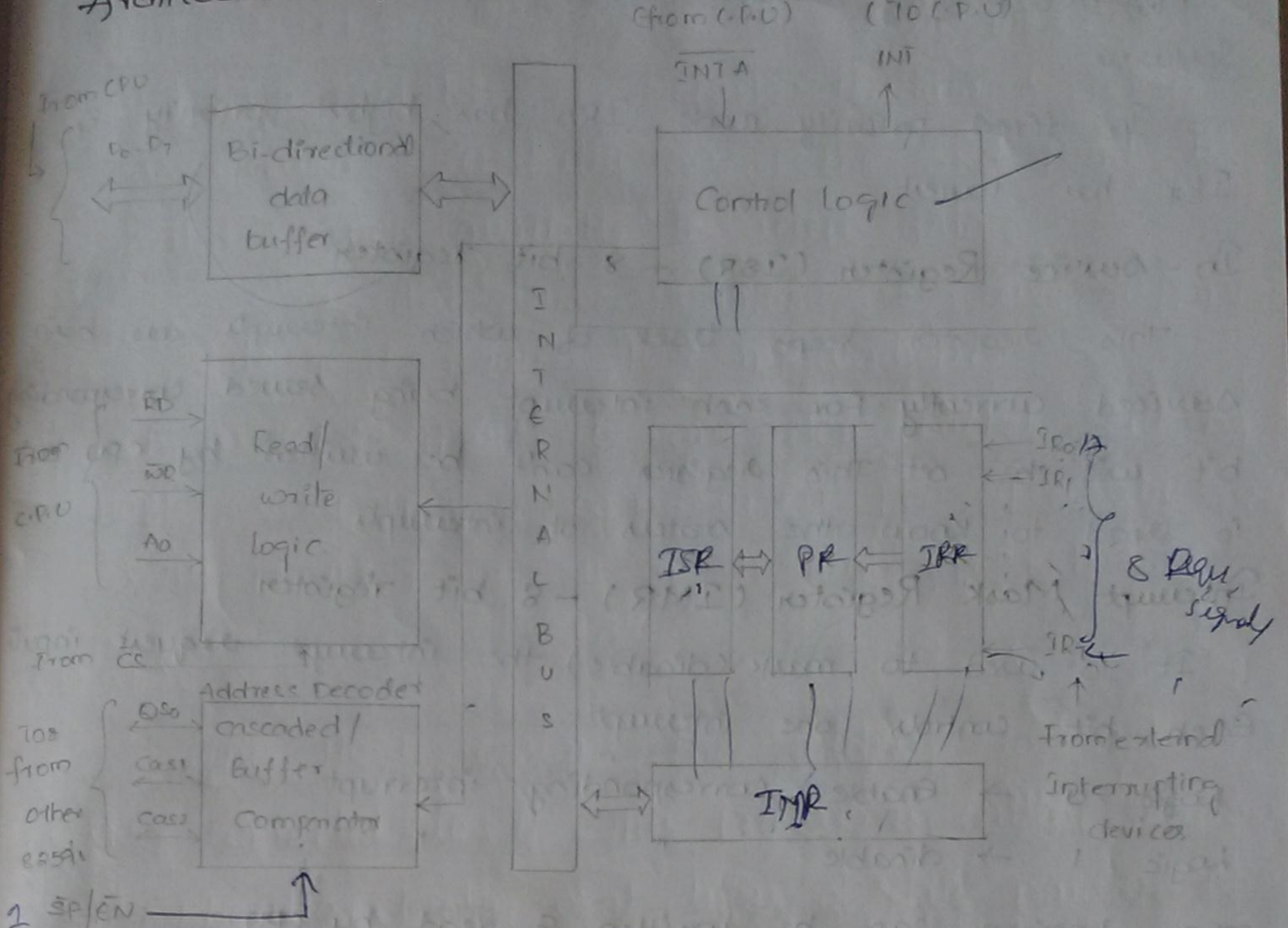
① In small systems number of devices interrupting the CPU will be less. Hence two interrupt lines NMI, INTR is sufficient

If no. of interrupting devices are more then it is not sufficient to run those interrupt devices by using two interrupt lines.

By using 8259 user can able to connect many no. of interrupting devices to 8086 MP & run those interrupts. It adds 8 priority encoded interrupts to 8086.

The 8 interrupts will be input to 8259 & o/p of 8259 will be connected to INTR pin of 8086. These interrupts can be expanded upto 64 interrupts.

Architecture of 8259 :



Read / write logic : (A_0, $\overline{RD}$, $\overline{WR}$ & $\overline{CS}$)

It controls the data transfers through buffer when device is selected through $\overline{CS}$ line. A_0 is used to select different command words of 8259.

Control logic : (INT, $\overline{INTA}$)

It controls the timing & activities of other blocks. Interrupt related signals are also controlled by this block

Interrupt Request Register: (IRR)

This block has got 8 inputs IR_0, IR_7 external devices will send the interrupt request on these lines. These requests will be registered by IRR. At any given point IRR indicates how many interrupt lines are active. Each bit corresponds to one interrupt

Priority Resover : (PR)

It determines the priorities of active interrupts and devices which interrupt should be serviced & when these are

different priority schemes which can be selected by Software.

In fixed priority mode IR0 has highest priority & IR7 has lowest

In-Service Register (ISR) - 8 bit register

This register keeps track of which interrupts are being serviced currently. For each interrupt being served corresponding bit will be set. This register can't be written by CPU but is read to know the states of interrupts.

Interrupt Mask Register (IMR) - 8 bit register

It is used to mask (disable) the interrupt request input. Each bit controls one interrupt.

Logic 0 → Enable corresponding interrupt

Logic 1 → disable " "

This register can be written & read by CPU.

Cascade Buffer / Comparator (CAS_{0-2}, $\overline{SP}$ / $\overline{EN}$)

It provides the interface between master 8259 & slave 8259's. It is used to expand the interrupts beyond 8. CAS0's lines are used for slave identification.

$\overline{SP}$/$\overline{EN}$ Signal is used to identify 8259 master (or) slave. If it is high, then 8259 is a master or else slave.

Sequence of operation:

The interrupts through 8259 are connected to 8086 using INTR line.

If interrupt flag is set & INTR becomes active high, 8086 will do the following:

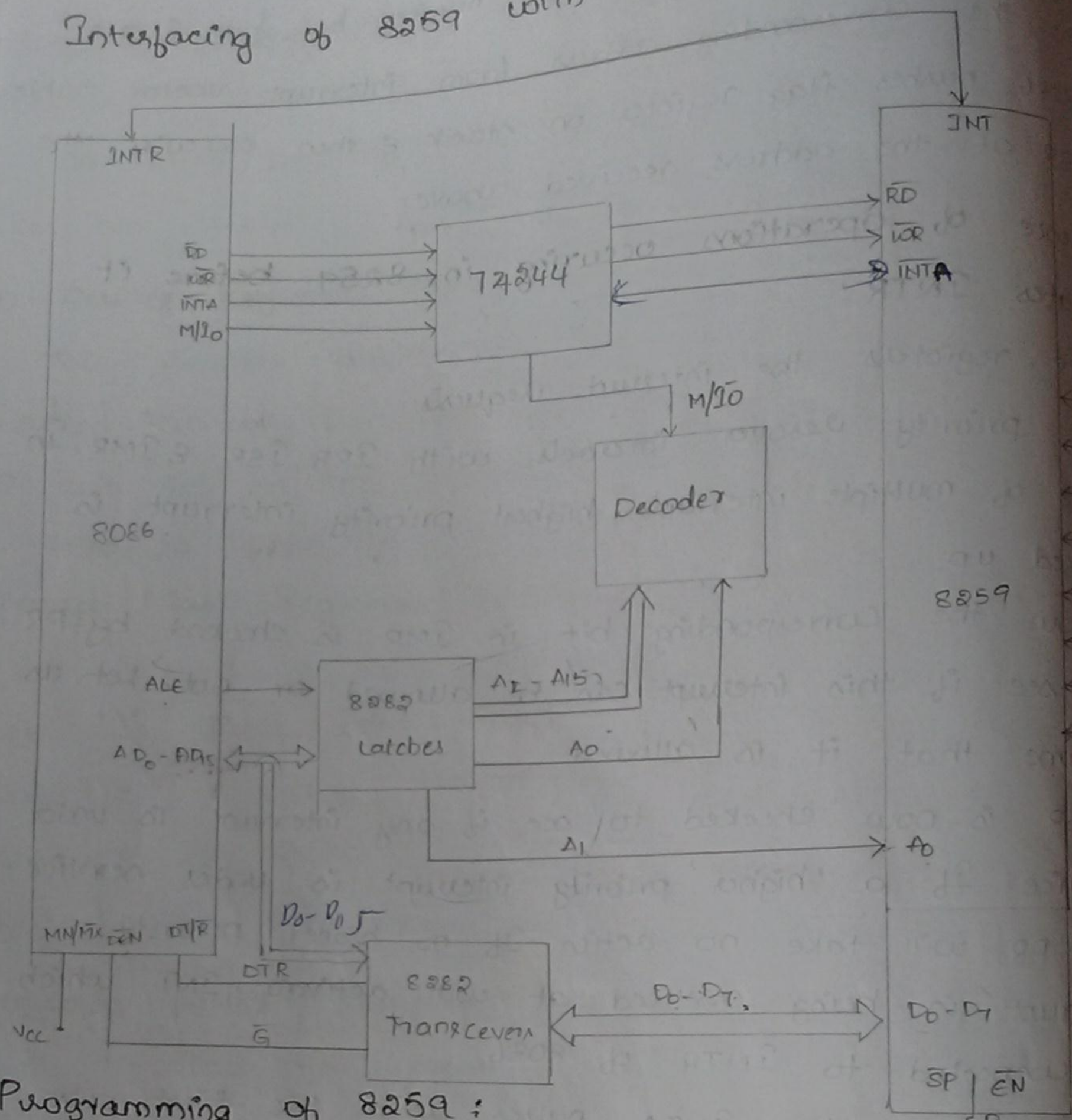
1. 8086 will pulse $\overline{INTA}$ signal twice indicating to 8259 that interrupt has been accepted.
2. 8259 will send the type number of highest priority interrupt line to 8086 on D_0-D_7 lines during second $\overline{INTA}$ pulse

3. 8086 will multiply this type numbes by four & get the the corresponding address from interrupt vector table
4. 8086 pushes flag registes on stack & then executes the ISR at the address received above.

Sequence of operations occuring in 8259 before it activates INTR:

1. IRR registes the interrupt requests
2. The priority resover interacts with IRR, ISR & IMR. In case of multiple interrupts, highest priority interrupt is picked up.
3. Now the corresponding bit in IMR is checked by PR to see if this interrupt can be allowed or not. Let us assume that it is allowed.
4. ISR is now checked to see if any interrupt is under service. If a higher priority interrupt is under service the PR will take no action. If no higher priority interrupt is being serviced at will activate INT which is connected to INTR of 8086
5. 8086 sends two INTA pulses using the first pulse 8259 will reset the corresponding IR bit of IRR to indicate the request is accepted & set the corresponding bit of ISR to indicate which IR level is under service.
6. During the second INTA pulse 8259 will send the type numbes of interrupt.
7. ISR bit will be reset at the end of interrupt service routine.

Interfacing of 8259 with 8086:



Programming of 8259:

8259 is programmed by two sets of command words. they are Initialization command word (ICW's) & operation command word (OCW's)

* ICW's are programmed to begin with & decide the basic operations of 8259
* OCW's are programmed during the normal course of operation

ICW's:

There are four ICW's are used in 8259 programming first two ICW's i.e., ICW_1 & ICW_2 are essential & remaining two are optional based on modes used

ICW1:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
A_7	A_6	A_5	1	LTIM	ADI	SNGL	IC_4

↳ D_0 → 0 ⇒ ICW4 is not needed
→ 1 ⇒ " " needed

↳ D_1 → 0 → Cascade mode is used
→ 1 → Single mode is used

↳ D_2 is labelled as ADI (address interval). It is used by 8085 only

↳ D_3 is labelled as LTIM (Level trigger interrupt mode)

D_3 → 0 → interrupt request i/p is edge triggered
→ 1 → " " " " level triggered

Ext. device Info.

↳ $D_5 - D_7$ ⇒ $A_5 - A_7$ of interrupt vector address.
These are used by 8085 only.

ICW2 :

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
T_7/A_{15}	T_6/A_{14}	T_5/A_{13}	T_4/A_{12}	T_3/A_{11}	A_{10}	A_9	A_8

↳ $D_0 - D_7$ are labelled as $A_8 - A_{15}$ (interrupt vector address used by 8085)

↳ $D_0 - D_2$ ⇒ not used by 8086

↳ $D_3 - D_7$ ⇒ $T_3 - T_7$ (interrupt vector type)
they are used by 8086 & will be 5 MSB's of 8 bit interrupt type number

ICW3 :

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
S_7	S_6	S_5	S_4	S_3	S_2	S_1	S_0	Master
X	X	X	X	X	ID_2	ID_1	ID_0	slave

ICW3 is required only when there is more than one 8259 & cascading is used.

ICW3 master :

$D_0 - D_7$ → $S_0 - S_7$ (slave 0 - slave 7)

$D_0 - D_2 \Rightarrow ID_0 - ID_2$ · These 3 bits identify the slave.

ICW4:

D7							D0
✗	✗	✗	SFNM	BUF	M/S	AEOI	MPM

D_0 → 0 – 8085 MP → 8 bit
→ 1 – 8086 mp → 16-bit

D_1 → 0 – for normal end of interrupt
→ 1 – for automatic " " "

D_2 → 0 – 8259 to be slave
→ 1 – " " " master

D_3 → 0 – for non buffer mode
→ 1 – for buffer mode

$\overline{SP}/\overline{EN}$ → input
→ output

D_4 → 0 – not specially fully nested mode
→ 1 – Specially fully nested mode

$D_5 - D_7$ – not used

OCW8:

There are three OCW's once 8259 is programmed using ICW's, it is ready to accept interrupts However during its operation various mode can be selected through OCW's.

OCW1:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
M_7	M_6	M_5	M_4	M_3	M_2	M_1	M_0

It is used to set and reset the mask bits in the interrupt mask register

$D_0 - D_7$ are labelled as $M_0 - M_7$ (mask 0 to mask 7)

D_0 = 1 – IR_0 is masked (or) disabled
= 0 – " unmasked (or) enabled

OCW2 :

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
R	SL	EOI	0	0	L_2	L_1	L_0

It decides how to respond to an interrupt.

$D_0 - D_2 \Rightarrow L_0 - L_2$ (level 0 - level 2)

These bits are used only in specific rotation mode & Specific EOI command

D_5 - EOI (End of interrupt)

D_6 - SL (Specific Level).

D_7 - R (rotation)

OCW3 :

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	ESMM	SMM	0	1	P	RR	RIS

D_0 - RIS (Read Interrupt Service Register)

D_0 → 0 - Selects IRR

D_0 → 1 - Selects ISR

$D_1 \rightarrow$ RR (read register)

D_1 → 0 - no read operation

D_1 → 1 - read either IRR (or) ISR

$D_2 \rightarrow$ P (poll command)

D_2 → 0 - Reset poll command

D_2 → 1 - Set poll command

D_5 - SMM (Special mask mode) → PR

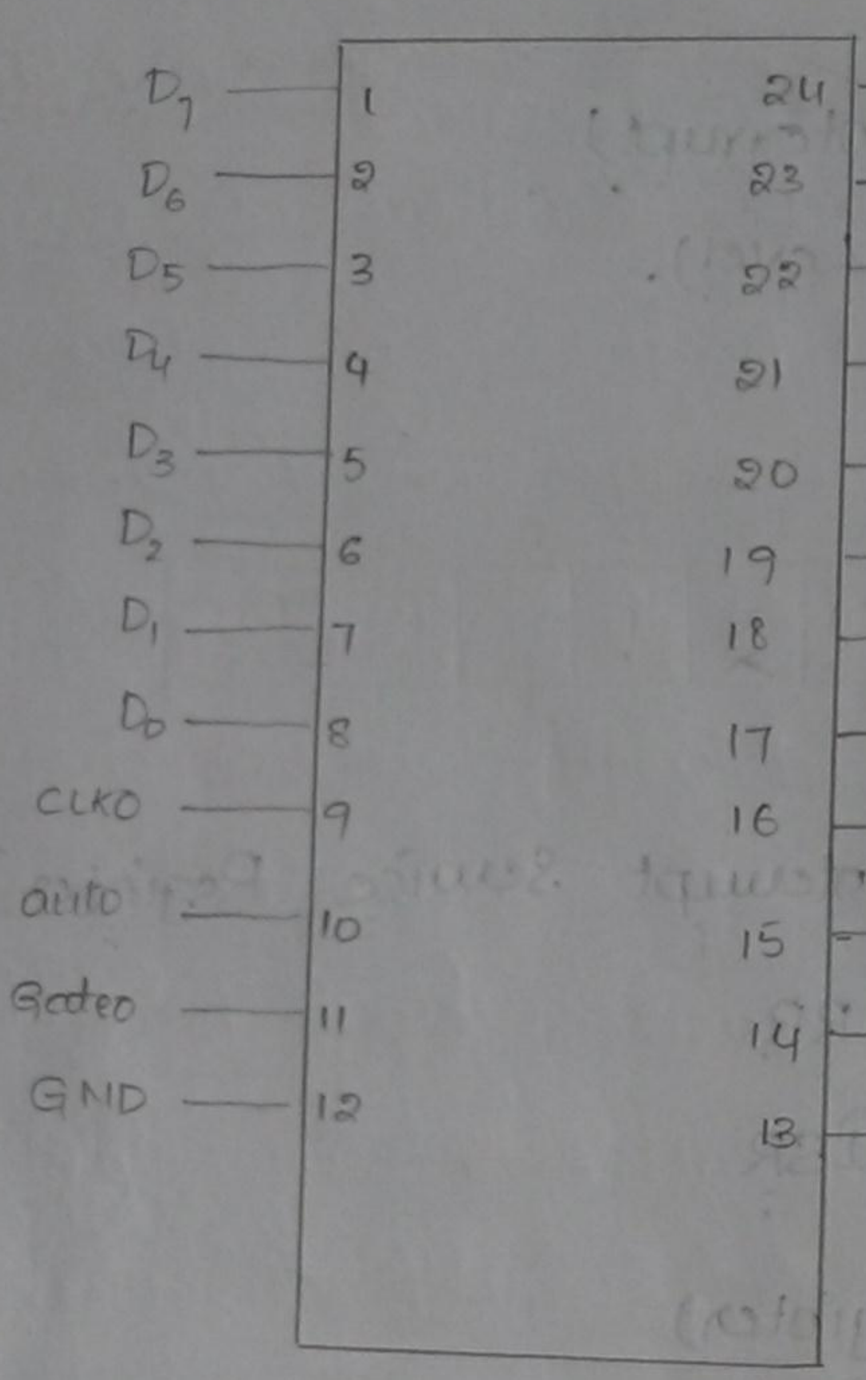
D_6 - ESMM (Enable special mask mode) } IMR

ESMM	SMM	Function
0	X	NO action
1	0	Reset SMM
1	1	Set SMM

5(a)

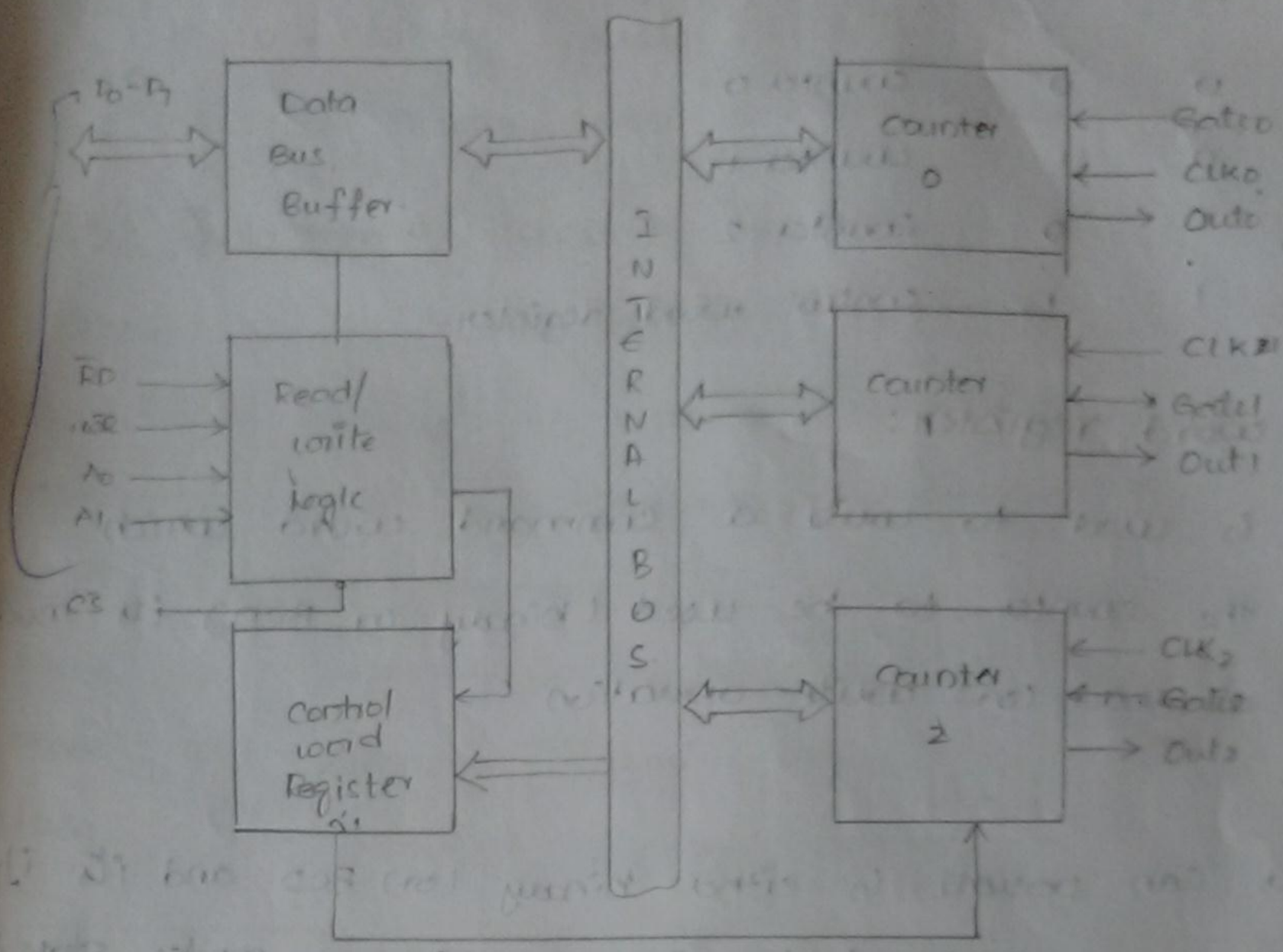
8253 programmable interval timer:

PIT is a counter which triggers an interrupts, when they reach their programmed count. It is a 24 pin IC these PIT consists of 3 counters R/w logic, data bus buffer, control word register



Features of 8253:

→ 3 independent 16-bit down counters
→ It can operate upto 2.6 MHz
→ Counters can be programmed in six different modes
→ Each counter has two i/p signals clk, gate & the o/p signal OUT



Data bus buffer:

This tri-state, bi-direction, 8 bit- buffer is used to interface the 8253 to the system data bus.

It has 3 functions.

- ↳ programming the modes of 8253
- ↳ Loading the count register
- ↳ Reading the count values

R/W Logic:

It has 5 signals $\overline{RD}$, $\overline{WR}$, $\overline{CS}$ & address lines A_0 & A_1 in peripheral I/O device (peripheral mode)

$\overline{RD}$ & $\overline{WR}$ are connected to $\overline{IOR}$ & $\overline{IOW}$

In memory map mode $\overline{RD}$ & $\overline{WR}$ is connected to $\overline{MEMR}$ & $\overline{MEMW}$. A_0 & A_1 of 8253 is connected to A_0 & A_1 of CPU.

The control word register & counters are selected according to the signals on lines A_0 & A_1

	A_1	A_0	Selection
	0	0	counter 0
	0	1	counter 1
	1	0	counter 2
	1	1	control word register

Control word register:

It is used to write a command word which specifies the counter to be used (binary or BCD) its & either a read (or) write operation

Counter:

Counter can operate in either binary (or) BCD and its gate & o/p are configured by the selection of modes in the control word register, counters are fully independent

The programmer can read the contents of any of the three counters without disturbing the actual count in process

8253 control word format:

SC_1	SC_0	RW_1	RW_0	M_2	M_1	M_0	BCD

SC - Select counter

SC_1	SC_0	
0	0	→ Select counter 0
0	1	→ " " 1
1	0	→ " " 2
1	1	→ Illegal for 8253 used in 8254

RW_1	RW_0	
0	0	→ Counter latch command
0	1	→ R/W least significant byte only
1	0	→ R/W most " "
1	1	→ R/W least " " first

M_2	M_1	M_0	
0	0	0	mode 0
0	0	1	mode 1
x	1	0	mode 2
x	1	1	mode 3
1	0	0	mode 4
1	0	1	mode 5

Then most significant byte

BCD → 0 - binary counter 16-bits
BCD → 1 - binary coded decimal counter.

8253 programming modes

There are 5 modes present in PIT

Mode 0 - interrupt on terminal count

Mode 1 - hardware retriggerable one-shot

Mode 2 - Rate Generator

Mode 3 - Square wave rate generator

Mode 4 - Software triggered strobe

Mode 5 - Hardware triggered strobe

Signal Status modes	Low (or) going low	Rising	High
Mode 0	Disables counting	—	Enables counting
Mode 1	—	1. Initiates counting 2. Resets output after next clock	—
Mode 2	1. disables counting 2. Sets the o/p immediately high	1. Reloads counter 2. initiates counting	enables counting
Mode 3	1. disables counting 2. Sets the o/p immed-ately high	1. Reloads counter 2. initiates counting	Enables counting
Mode 4	Disables counting	—	Enables counting
Mode 5	—	Initiates counting	—