Interrupts of 8085 µp -

- Interrupt is a hardware or software initiated call that interrupts the currently executing program at any point when µp receives an input signal, it suspends the currently executing program and jumps to an Interrupt service Routine (ISR) to respond to the incoming interrupt. It is also called interrupt service provider or Interrupt handler.

* Sources of Interrupt.

(1) Power Interrupt: generated by processor itself in response to an error condition.

(11) Hardware Interrupt: Initiated by external hardware device.

(3) Software Interrupt: errors caused by malfunctioning software.

* Interrupt Processing.

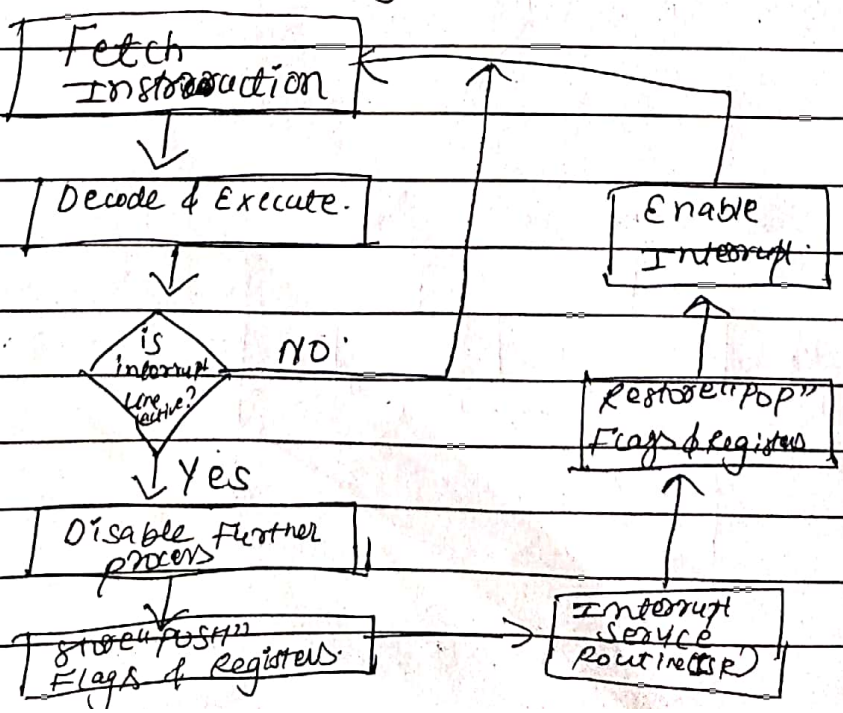


Fig: Interrupt Processing.

Interrupts can be classified as into:-

1) Maskable & Non-Maskable
2) Vectored & Non-vectored.

1) Maskable & Non-maskable:

→ The interrupts that can be blocked or delayed using instructions are called maskable. The RESET interrupts (RST 7.5, RST 6.5, & RST 5.5) are maskable which can be enabled or disabled using 'EI/DI' instructions.

→ The non-maskable interrupts are those interrupts which cannot be blocked using instructions. TRAP is one such interrupts which is non-maskable. It has the highest priority.

2) Vectored & Non-vectored:

→ The interrupts for which the address of ISR is already known to the µp are vectored interrupts
e.g:- RST 7.5, 6.5, 5.5.

→ The interrupts for which the address of ISR needs to the µp are known as non-vectored.
eg:- $\overline{INTR}$ & $\overline{INTA}$.

Multiple interrupts.

- The processor is usually provided with 1 or more interrupts pins. Therefore a special mechanism is necessary to handle multiple interrupt lines. There are mainly the two ways of handling multiple interrupts:-

1) Polled interrupts
2) Unpolled vectored / Daisy chain Interrupts.

1) Polled Interrupt Handling.

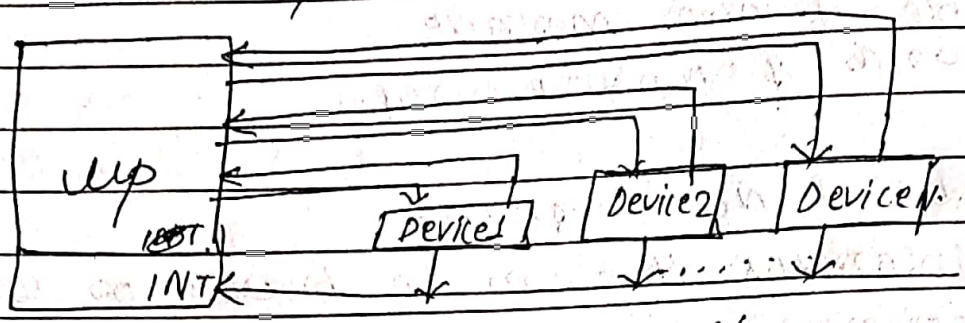


fig: Polled Interrupt.

As shown in figure, several external device are connected to the processor where a single interrupt line is connected for interrupt signal. When one or more device activates the INT line is high, the processor saves the contents of PC & registers and branches to the address of interrupt.

For a large no. of device, the time required to poll each device may exceed the time required to service the device.

2) Vectored (Daisy-chained) Interrupt:

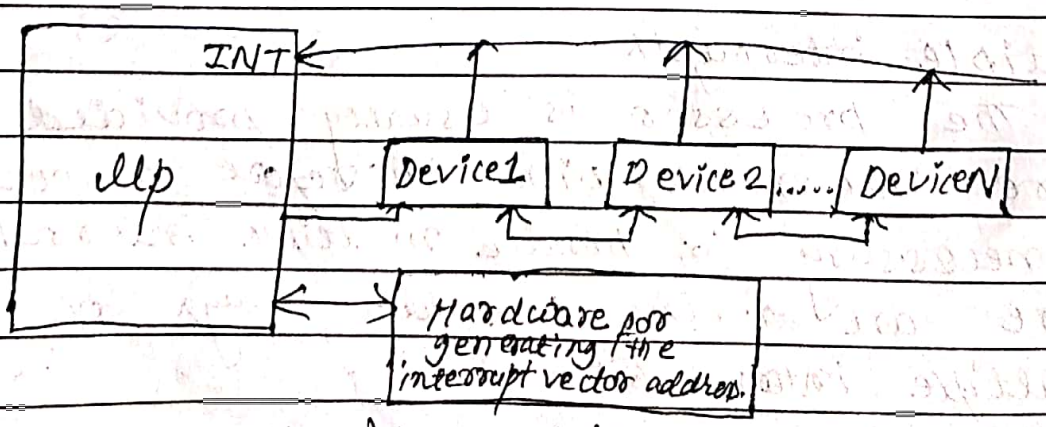


fig: Vectored/Daisy chain Interrupt.

-> In this method, all the devices are connected in a chain fashion.
-> All the device which request the interrupt are connected by matching their priority order where the highest priority device are serviced first and so on.
-> When any device or multiple device signals an interrupt request, the CPU acknowledges the interrupt.
-> The device on the line passes the interrupt acknowledge signal to the next device.

DMA (8237 DMA Controller)

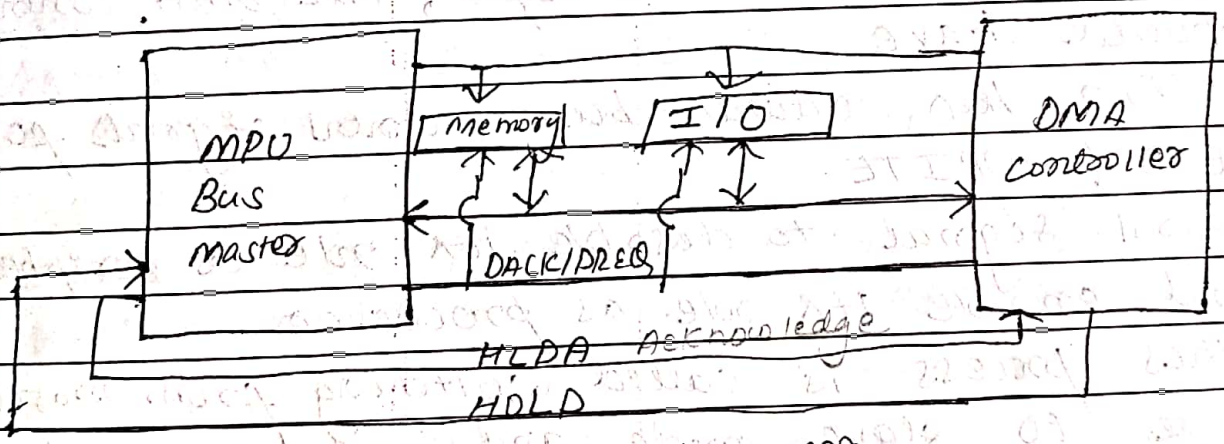


fig: DMA transfer.

=> DMA is a controller commonly used for high speed data transfer.
-> Data transfer is relatively slow from the microprocessor bus system, hence DMA is introduced.
-> The controller manages the data transfer between memory and peripheral device by passing the MPU.
-> It uses two signal for the use of up bus:
(a) HOLD: It is an active high signal to pass up for the case of buses (address & data). After receiving the

HOLD signal the MPU releases the bus in the next machine cycle. The HLDA signal is sent out and finally the control of bus is released after the HOLD signal gets low.

(6) Hold AC HLDA (Hold Acknowledge).

→ It is a signal that indicates that the cpu has released the control of the bus.

→ The DMA controller is capable of only copying data from one location to another at higher speed

DMA controller (8237)

To perform DMA transfer, the DMA controller should have

- a data bus, address bus & control signals for READ & WRITE.
- control signal to disable its role as peripheral and enable its role as processor.

This process is called switching from master mode to slave mode and vice versa.

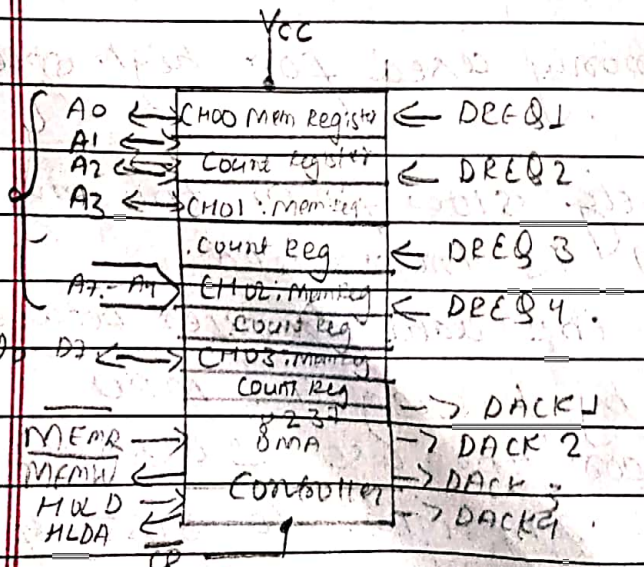


fig: DMA controller.

* DREQ1 – DREQ4 are the DMA Request pins through which the devices request for DMA.
* DACK1 – DACK4 – are the DMA acknowledge pins through which the requesting devices gets acknowledge by the controller.
* $\overline{MEMR}$ & $\overline{MEMW}$ are READ & WRITE pins to get access to memory.
* A0 – A3 are bidirectional address bus
* A4 – A7 are unidirectional address bus.
* D0 – D7 are 8-byte data bus.

DMA execution: The execution of DMA occurs in two modes of operation.
(a) Slave mode (b) Master Mode.

(a) Slave mode: In slave mode the DMA controller is treated as a peripheral device and stays idle until the operation of high speed data transfer is required.

(b) Master mode: When peripheral device are ready, it will sends high signal to DREQ and the HOLD is set high, the MPU releases the control of the bus by releasing HLDA signal after receiving the HLDA the address bus and databus are used to transfer the data.

When HOLD signal gets low the bus is sent back to MPU. The controller uses and $\overline{EOP}$ (End of process) to denote the end of operation.

Difference between I/O mapped & memory mapped.

I/O Mapped	Memory mapped
1) The I/O mapped uses the instructions such as IN & OUT.	1) The memory mapping uses the instructions such as STA, LDA, MVI, MOV.
2) The total capacity of I/O mapped process is 256 [00-FF]	2) The total capacity of memory mapped process is 64K [0000-FFFF]
3) It is an 8-bit operation cycle.	3) It is a 16-bit operation cycle.
4) signals such as $\overline{IOWR}$ & $\overline{IORD}$ are used to indicate the I/O read & write process.	4) signals such as $\overline{MEMR}$ & $\overline{MEMW}$ are used to indicate the memory read and write process.
5) Data transfer is between I/O device and accumulator.	5) The data transfer is between accumulator, register & memory

8259 PIC (Programmable Interrupt controller)

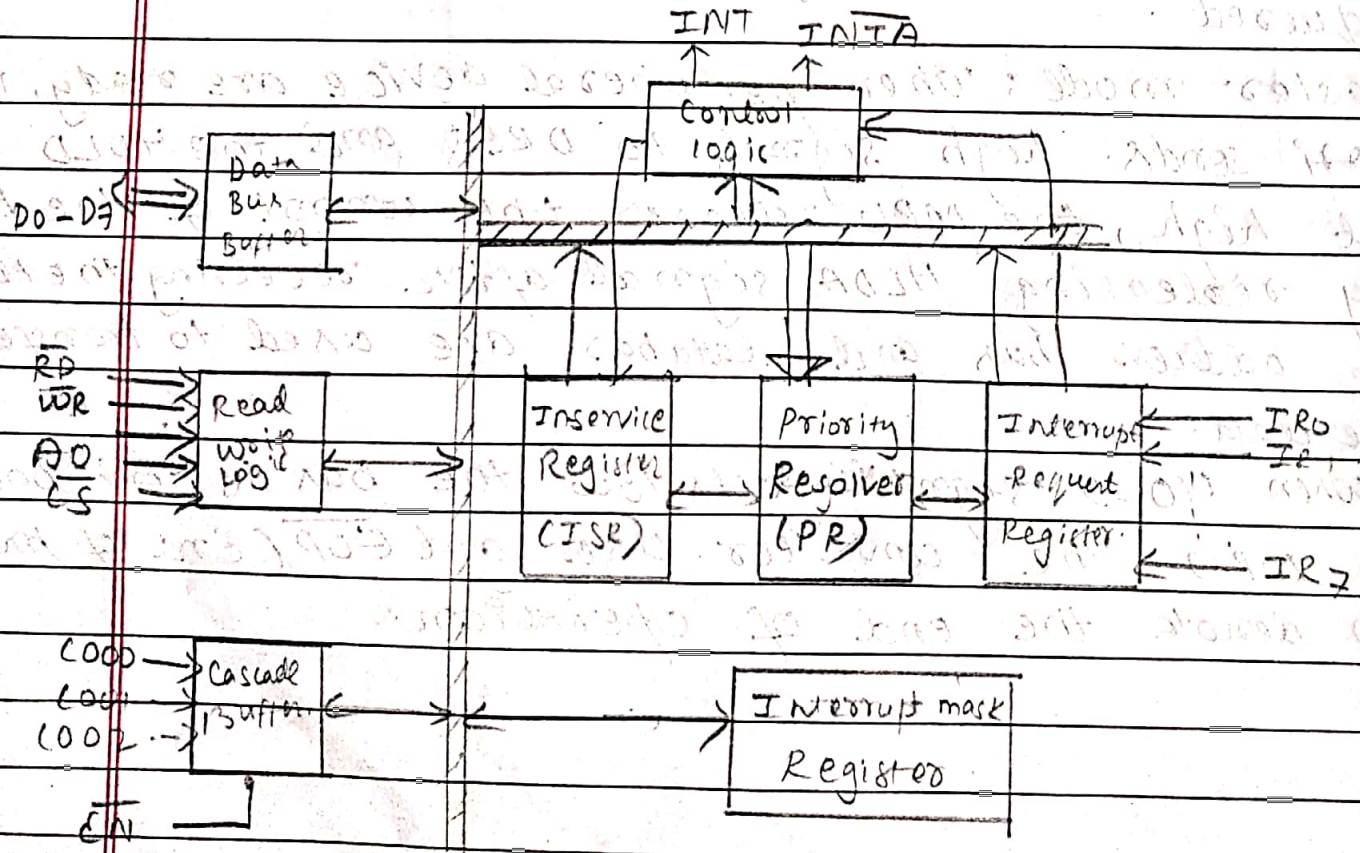


fig: 8259 PIC controller.

8259 (PIC)

→ It is designed to work with intel 8085, 8086 & 8088 µPs.

→ It can be programmed to take various types of interrupts from simple I/O interrupt to multiple interrupts.

→ It can manage 8 - interrupts request at one time.

→ The vectored interrupts can request anywhere in the memory unit without additional hardware.

→ Can solve eight interrupts on the basis of their priority.

The major components of PIC include.

1) Read/write logic: When the address line (A_0) is at logic '0' the controller is selected to write command or read command. The ($\overline{CS}$) chipselect pin determines the port address of the controller to the microprocessor.

2) Control logic: The INT pin is used for O/P and the $\overline{INTA}$ pin is used for I/P.

3) Interrupt Request Register & Priority Resolver:

→ The IRR has 8 lines for interrupt. The requests are stored in registers. The ISR stores all the bits of the interrupt line. The priority Resolver solves the order of priority.

4) Cascade Buffer: This block is used to expand the number of interrupt lines by cascading more PIC's.

Remaining...

8259 PIC, Modes of operation.

(1) Fully Nested Mode: In this mode the 8-interrupts are arranged from IR_0 - IR_7 and the priority is set likewise. After IR_0 gets serviced the immediate next interrupt gets serviced.

i.e. IR_0, IR_1 IR_2 IR_3 IR_4 IR_5 IR_6 IR_7

↑ ↑

Highest priority - - - - - - - - lowest priority

(2) Rotation mode: It includes two modes i.e.

(a) Automatic Rotation mode.

(b) Specific Rotation mode.

(a) Automatic Rotation mode: In this mode of operation the highest priority interrupt after being serviced gets the lowest priority and the next interrupt after that will get the highest priority.

i.e. IR_0 IR_1 IR_2 IR_3 IR_4 IR_5 IR_6 IR_7.

↓

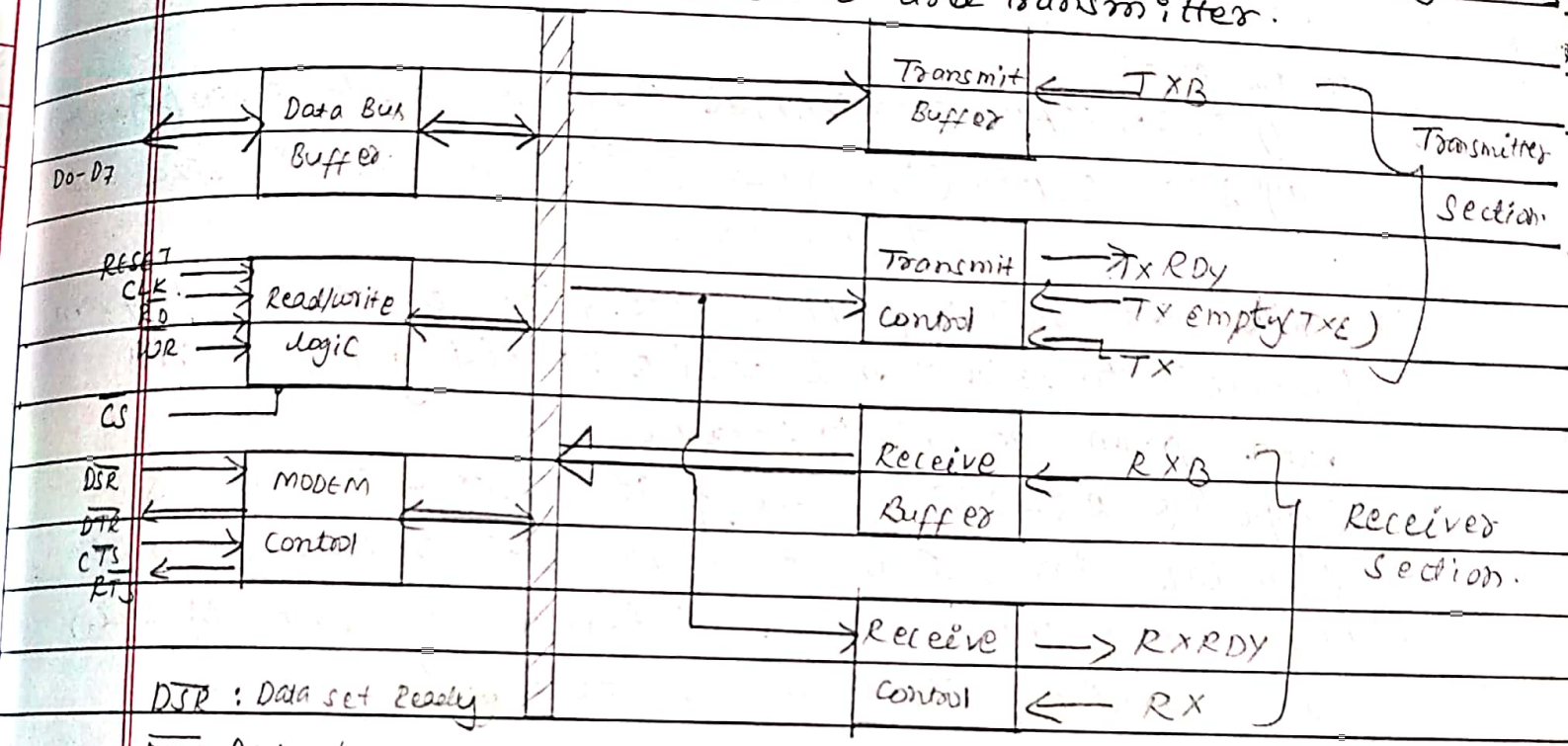
highest priority.

After IR_2 is serviced the priority is passed into $(IR_2+1) = IR_3$.

(b) Specific Rotation mode.

- In this mode of operation the user can select the order of priority according to the demand. The user has full control over it.

* 8251 USART : Universal Synchronous Asynchronous Receiver and transmitter.



DSR : Data set Ready

DTR Data transmit Ready

CTS : clear to send.

RTS : Request to send

Serial IIO (8250 and 8251)

→ There are two basic approaches for data transmission Namely a) Software controlled Data Transmission.
b) Hardware controlled Data Transmission.

a) Software Controlled ; The two pins SOD (serial Out Data) and SID (serial In Data) of µp are used for software controlled data transmission. They transmit data by the use of RIM & SIM instructions.

(b) Hardware Controlled Data Transmissions :-

→ Two basic chips are used for hardware controlled.

(i) 8250 UART (Universal Asynchronous Receiver & Transmitter)

(ii) 8251 USART (Universal Synchronous & Asynchronous Receiver and Transmitter).

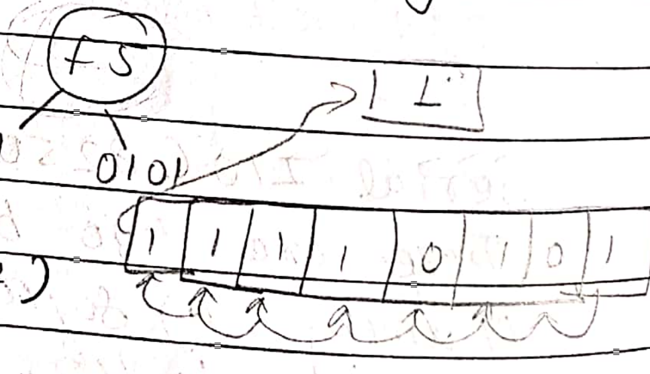
→ While transmitting the data, it converts data from parallel to serial and while receiving it converts data from serial to parallel.

→ Supports RS-232 cabling standard which is popular in serial communication bus standard.

→ Commonly used in PC related equipments such as printers, FAX device etc.

). WAP to count the number of 1's from a given data byte.

```
        MVI A, F5 H.
        MVI B, 00 H.
        MVI C, 08 H.
loop 2  RAL (Rotate ACC. left)
        JC / JNC loopL
        ↓
NO of zero's   INR B; [B]=01, → 02 //02 //03 //04 //05 // 06.
loopL   DCR C; [C] = 07 .... 06     06 //05 //04 //03 //02
        JNZ loop 2                  01 // 00
        MOV A, B ; [A]=06
        STA C050 ; [C050]=06
        RST 5.
```



* 8251 USART [continued....]
-The main components of 8251 USART includes:

1) Read/Write logic
2) Transmitter section
3) Receiver section
4) Data bus buffer.
5) Modem control.

1) Read/Write logic: It determines the function of the chip according to the data byte written in the control word.
- The RESET pin will allow the USART to stay idle.
- When $\overline{RD}$ and $\overline{WR}$ are low the µp is ready to read or write.
- The $\overline{CS}$ (Chipselect) pin will enable the controller.
- The CLK pin is provided to sync with the clock of the processor.

2) Transmitter section.
-> Accepts parallel data from the µp & converts them into serial data stream.
-> It has two registers, a buffer register to hold the 8-bit and output register to convert those 8-bits into serial stream.
TXB - Transmit data.
TXC - Transmitter Clock [rate of transfer].
TxRDy - Transmitter Ready [high -> USART is Ready]
TxE - Transmitter empty [high -> output register is empty].

3) Receiver Section : Accepts serial data on the RxB from a peripheral and converts them into parallel data format.

4) Data Bus Buffer: This is a 8-bit buffer used to interface 8251 to the system data bus.

5) Modem Control : Interface with modem with the help of input & output controls:
- DSR (Data set Ready)
- DTR (Data Transmit Ready)
- CTS (Clear to send)
- RTS (Request to send)

8255-A PPI (Programmable Peripheral Interface)

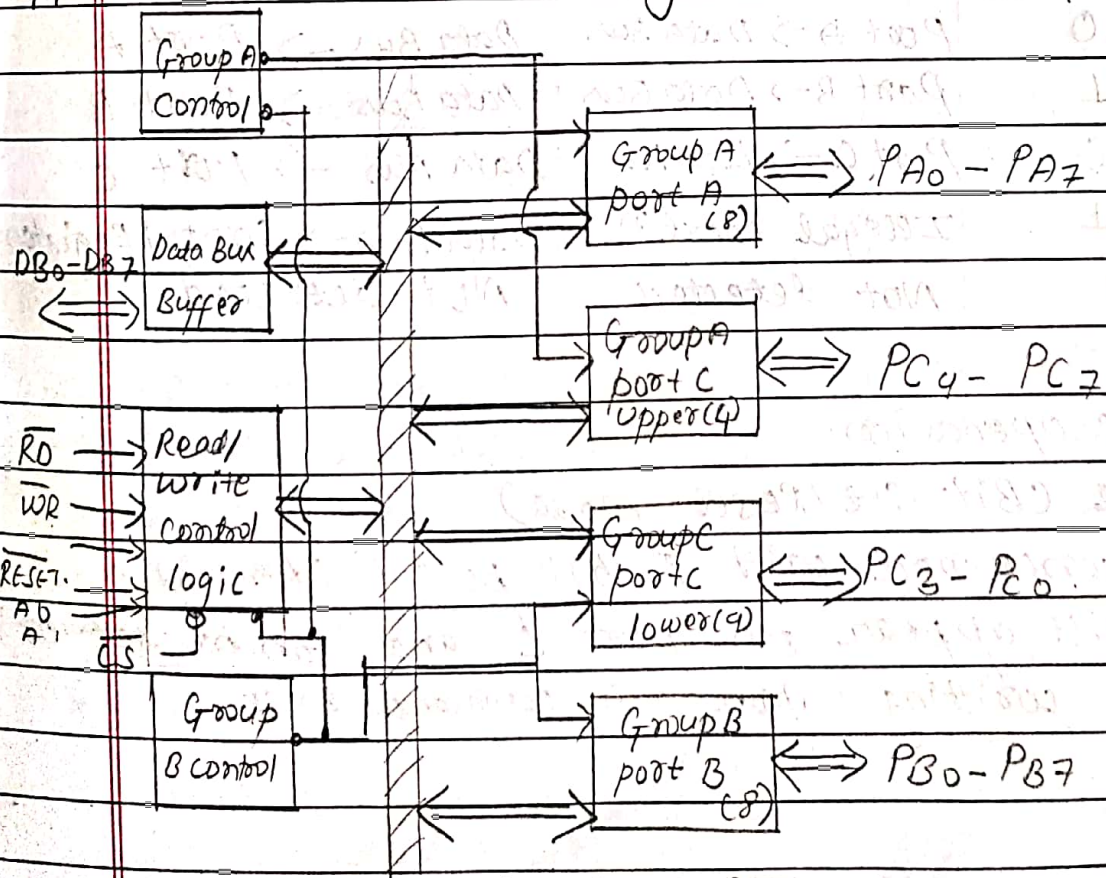


fig: 8255-A PPI

- It is a general purpose parallel I/O interfacing device. 8255-A can be set up to perform specific function by writting an instruction in its internal register called control word/control Register. The function can be changed anytime during the execution of the program by writting new instructions in its CR.

-> 8255-A is a 40 pin IC having 24 I/O ports
-> They are divided into 3, 8 bit I/O ports i.e. Ports A, port B & Port C.
-> Port C are further divided into 4 bits i.e. C Upper & C lower.

Select signals				
$\overline{CS}$	A1	A0	$\overline{RD}$ = 0 (I/p operation)	$\overline{WR}$ = 0 (O/p Operation)
0	0	0	Port A -> Data Bus	Data Bus -> Port A
0	0	1	Port B -> Data Bus	Data Bus -> Port B
0	1	0	Port C -> Data Bus	Data Bus -> Port C
0	1	1	Illegal condn	Data Bus -> Control Register
1	x	x	Not detected.	Not detected.

* Modes of operation.

(1) BSR Mode (Bit set/Reset mode)
-> It is concerned with 8-bit port C Operation.
-> Output will appear on port C and can be set or Reset by writting appropriate command on CR.

2) I/O Mode : The I/O mode are further divided into 3 modes.
i.e. (a) Mode 0 (b) Mode 1 (c) Mode 2.

3) (a) Mode 0 -> simple I/O mode
-> port A, port B & port C are used.
-> Unidirectional data transfer.
-> No handshaking signal are required.

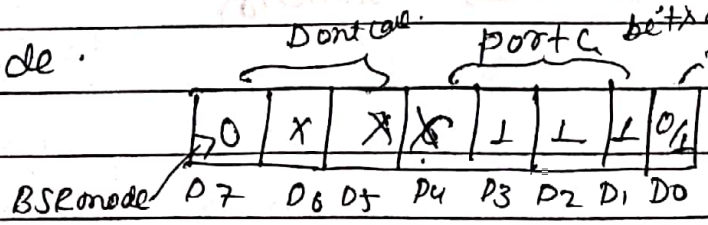
(b) Mode 1 : -> port A and/OR port B are used.
-> Unidirectional data transfer.
-> Handshake signal are used.

(c) Mode 2 : Bidirectional data transfer.
-> only port A is used.
-> For handshaking, port C is used.

Therefore, port A is programmed by mode 0, 1 and port B is programmed by mode 0 & 1 and port C is divided into two (upper & lower)

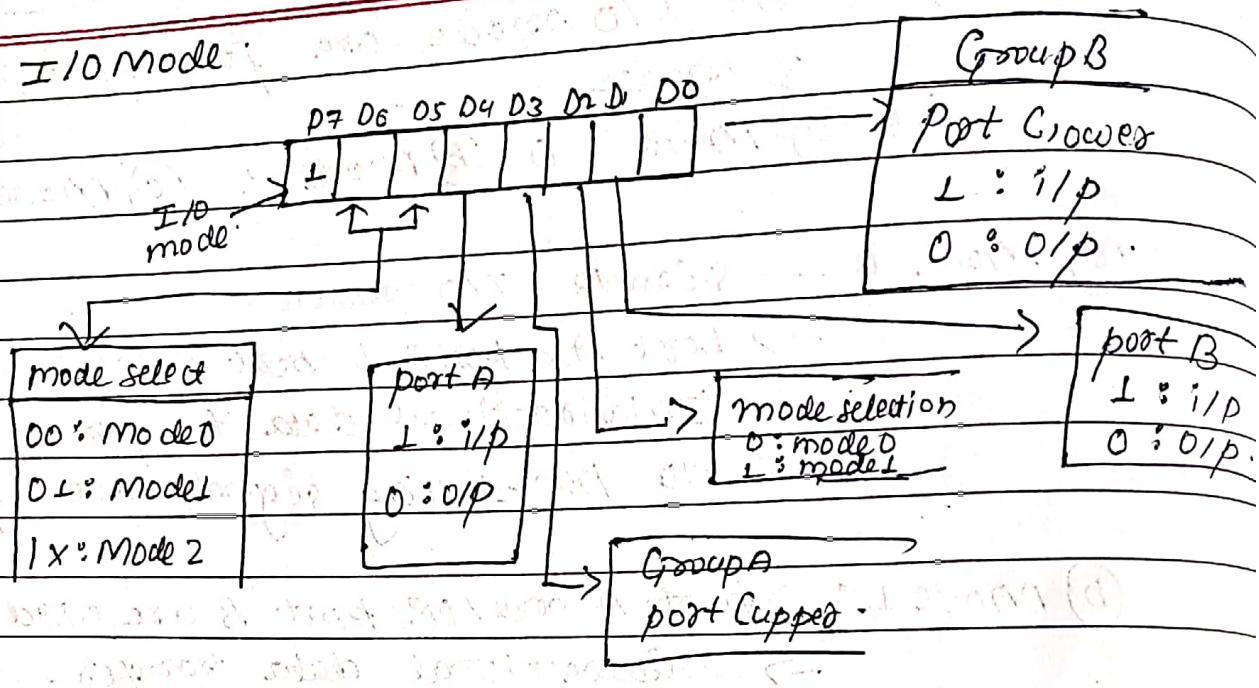
* Control word / control Register (8-bit).
-> It is basically an 8-bit code which needs to be written into the control registers, which helps to determine the which ports are to be used.

* BSR mode.



1 : Set
0 : Reset.

* I/O Mode.



Stack operation in 8085 µp.

-> Stack is used as a memory during the execution of programs that are used to hold the 8-bit hex data in memory address.

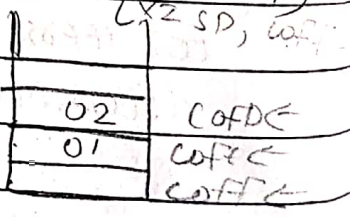
-> It follows the 'LIFO' order for storing [Last In First out].

Instructions of Rp Stack Operation.

1) PUSH Rp (PUSH B, PUSH D)

-> It is initiated by writing the command LXI SP, 16-bit address.

eg: PUSH B: B | 01 | 02 | C



-> It is a post-decremental instruction.

2) POP Rp.

-> It is used to extract the data from the stack memory location.

eg: POPH, POPB, POPD.

—> POP H.

H | 01 | 02 | L

-> It is a post decremental instruction.