PRIORITY INTERRUPT :-

A priority interrupt is a system that establishes a priority over
the various sources to determine which condition is to be
serviced first when two or more requests arrive simultaneously.
The system may also determine which conditions are permitted
to interrupt the computer while another interrupt is being
serviced. Higher-priority interrupt levels are assigned to request
which, if delayed or interrupted. Devices with high speed
transfers such as magnetic disks are given high priority &
slow devices such as keyboards receive low priority. When
two devices interrupt the computer at the same time, the
computer services the device, with the higher priority first.

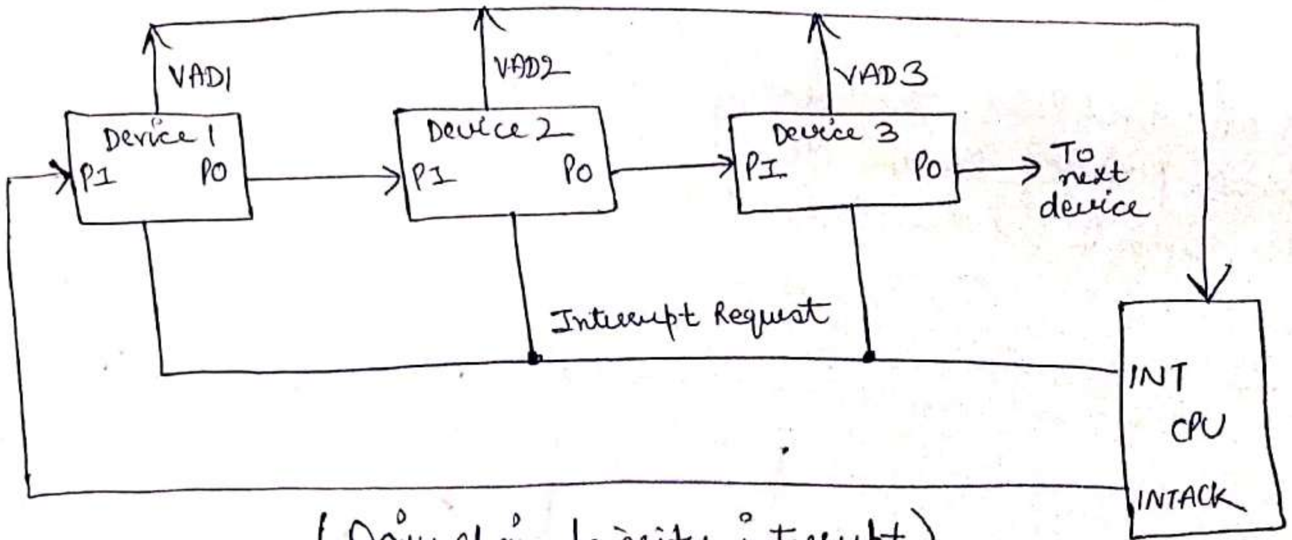
Establishing the priority of simultaneous interrupts
can be done by software or hardware. A polling procedure
is used to identify the highest-priority source by software means.

In this method there is one common branch address for all interrupts. The program that takes care of interrupts begins at the branch address and polls the interrupt sources in sequence. The order in which they are tested determines the priority of each interrupt. The highest-priority source is tested first, and if its interrupt signal is on, control branches to a service routine for this source. Otherwise next-lower priority source is tested & so on. Thus the initial service routine for all interrupts consists of a program that tests the interrupt sources in sequence & branches to one of many possible service routines. The disadvantage of the software method is that if there are many interrupts, the time required to poll them can exceed the time available to service the I/O device. In this situation a hardware priority-interrupt unit can be used to speed up the operation.

A hardware priority-interrupt unit functions as an overall manager in an interrupt system environment. It accepts interrupt requests from many sources, determine which of the incoming requests has the highest priority and issues an interrupt request to the computer based on this determination. To speed up the operation, each interrupt source has its own interrupt vector to access its own service routine directly. Thus no polling is required becz all the decisions are established by the hardware priority-interrupt unit. The hardware priority function can be established by either a serial or a parallel connection of interrupt lines. The serial connection is also known as the daisy chaining method.

Daisy chaining Priority

The daisy-chaining method of establishing priority consists of a serial connection of all devices that request an interrupt. The device with the highest priority is placed in the first position, followed by lower priority devices up to the device with the lowest priority which is placed last in the chain. This method of connection between three devices & the CPU is shown.



(Daisy chain priority interrupt)

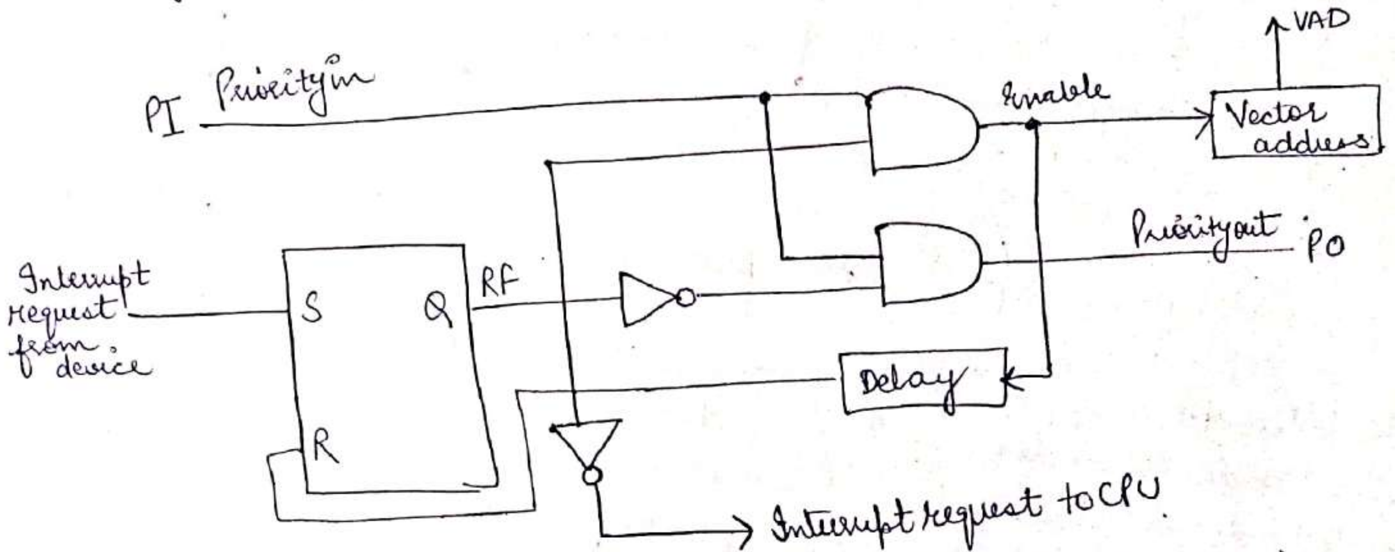
The interrupt request line is common to all devices and forms a wired logic connection. If any device has its interrupt signal in low level state, the interrupt line goes to the low level state and enables the interrupt input in the CPU. When no interrupts are pending, the interrupt line stays in the high level state & no interrupt are ~~pending~~ recognized by the CPU. The CPU responds to an interrupt request by enabling the interrupt acknowledge line. This signal is received by device 1 at its PI (priority in) input. The ack. signal passes on to the next device through PO (priority out) output only if device 1 is not requesting an interrupt. If device 1 has a pending interrupt, it blocks the acknowledge signal from the next device by placing 0 in the PO output. It then proceeds to insert its own interrupt vector address (VAD) into the data bus for the CPU to use during the interrupt cycle.

A device with a 0 in its PI input generates a 0 in its PO output to inform the next-lower priority device that the acknowledge signal has been blocked.

A device that is requesting an interrupt has a 1 in its PI input will intercept the ack. signal by placing 0 in its PO output.

If the device doesnot have pending interrupts, it transmits the ack. signal to the next device by placing 1 in its PO output.

Thus a device with PI =1 and PO=0 is the one with the highest priority that is requesting an interrupt, and this device places its VAD on data bus. The daisy chain arrangement gives the highest priority to the device that receives the interrupt ack. signal from the CPU.



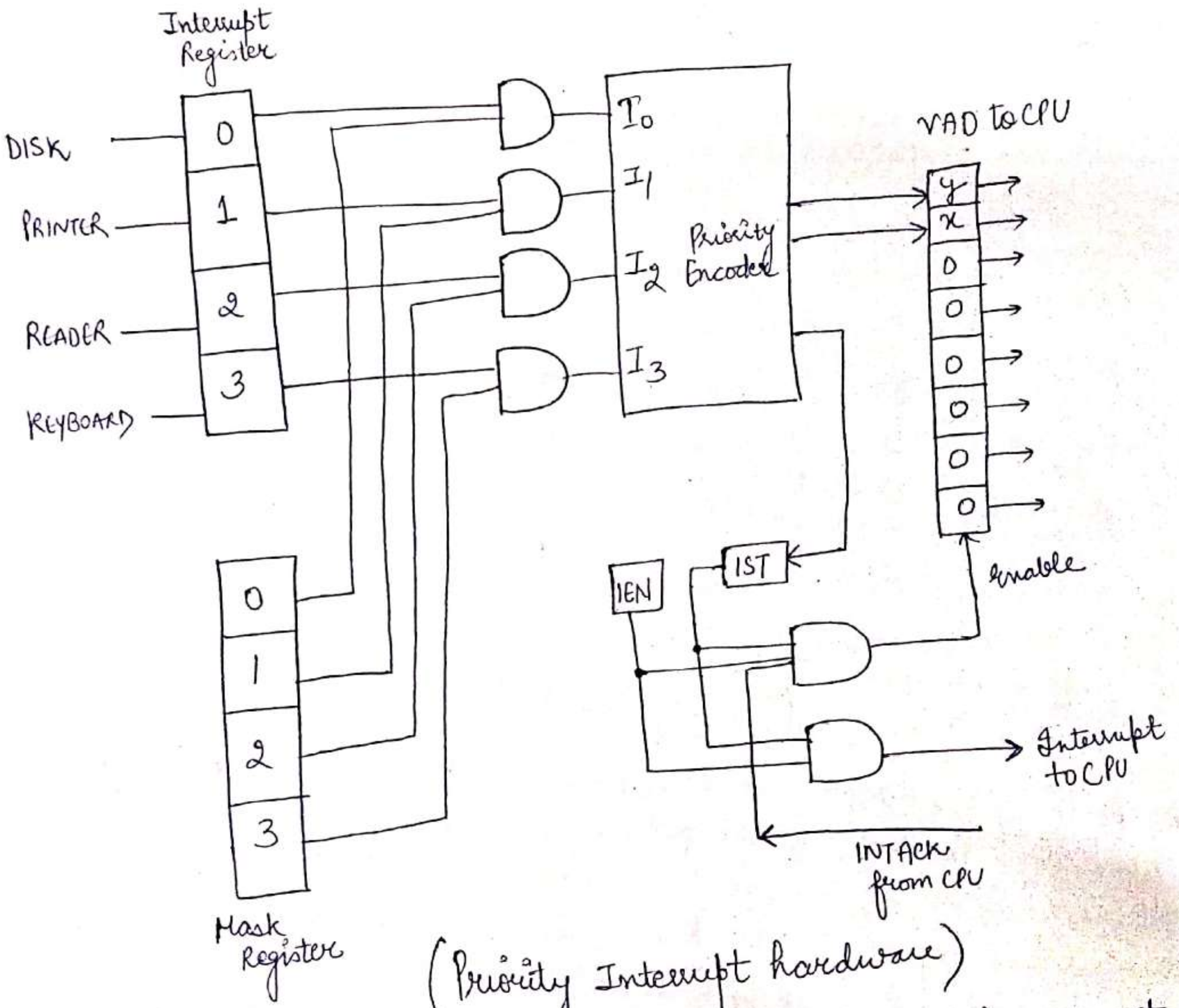
(One stage of daisy chain priority arrangement)

PI	RF	PO	Enable
0	0	0	0
0	1	0	0
1	0	1	0
1	1	0	1

The device sets its RF flip flop when it wants to interrupt CPU.
The output of the RF flip flop goes through an open-collector inverter, a ckt that provides the wired logic for the common interrupt line.
If PI = 0, both PO and enable line to VAD are 0.
If PI = 1 & RF = 0 then PO = 1 & VAD is disabled. This condition passes the acknowledge signal to next device through PO.
If PI = 1 & RF = 1 then PO = 0 and VAD is enabled.

Parallel Priority Interrupt

The parallel priority interrupt method uses a register whose bits are set separately by the interrupt signal from each device. Priority is established according to the position of the bits in the register. In addition to the interrupt register, the circuit may include a mask register whose purpose is to control the status of each interrupt request. The mask register can be programmed to disable lower priority interrupts while a higher-priority device is being serviced. It can also provide a facility that allows a high-priority device to interrupt the CPU while a lower-priority device is being serviced.



(Priority Interrupt hardware)

It consists of an interrupt register whose individual bits are set by external conditions and cleared by program instructions. The magnetic disk, being a high speed device, is given highest priority. The printer has the next priority, followed by a character reader and a keyboard. The mask register has the same number of bits as interrupt register. Each interrupt bit and its corresponding mask bit are applied to an AND gate to produce the four inputs to priority encoder. The priority encoder generates two bits of the vector address, which is transferred to the CPU.

Another O/P from the encoder sets an interrupt status flip flop IST when an interrupt that is not masked occurs. The interrupt enable flip flop IEN can be set or cleared by the program to provide an overall control over the interrupt system. The O/P of IST ANDed with IEN provide a common interrupt signal for the CPU. The interrupt ack. INTACK signal from the CPU enables the bus buffers in the o/p register and a VAD is placed into the data bus.

INPUTS				OUTPUTS			Boolean function
I_0	I_1	I_2	I_3	x	y	IST	
1	x	x	x	0	0	1	
0	1	x	x	0	1	1	$x = I_0' I_1'$
0	0	1	x	1	0	1	$y = I_0' I_1 + I_0' I_2'$
0	0	0	1	1	1	1	$(IST) = I_0 + I_1 + I_2 + I_3$
0	0	0	0	x	x	0	

(Priority encoder truth table)

The priority encoder is a circuit that implements the priority function. The logic of the priority encoder is such that if two or more inputs arrive at same time, the input having the highest priority will take precedence.
The x's in the table designate don't care conditions. Input I_0 has the highest priority; when this input is 1, the o/p generates an output xy = 00. I_1 has the next priority level. The output is 01 if $I = 1$ provided that $I_0 = 0$, regardless of values of the other two lower - priority inputs.
The o/p for I_2 is generated only if higher inputs are 0 & so on
The interrupt status IST is set only when one or more inputs are equal to 1. If all the inputs are 0, IST is cleared to 0 and the other o/p of encoder are not used, so they are marked with donot-care condition.
The flipflop IEN can be set or cleared by program instructions. when IEN is cleared, the interrupt request coming from IST is neglected by the CPU. An instruction to set IEN indicates that the interrupt facility will be used while the current program is running. Most computers include internal hardware that clears IEN to 0 everytime an interrupt is acknowledged by the processor

At the end of each instruction cycle the CPU checks IEN and the interrupt signal from IST. If either is equal to 0, control continues with next instruction. If both IEN and IST are equal to 1, CPU goes to interrupt cycle. During interrupt cycle the CPU performs following sequence of microoperations :

$SP \leftarrow SP - 1$	Decrement stack Pointer
$M[SP] \leftarrow PC$	Push PC into stack
$INTACK \leftarrow 1$	Enable interrupt acknowledge
$PC \leftarrow VAD$	Transfer vector address to PC
$IEN \leftarrow 0$	Disable further interrupts

Go to fetch next instruction