UNIT-5

Peripheral Devices :-

Devices that are under the direct control of the computer are said to be connected on line. These devices are designed to read information into or out of the memory unit upon command from the CPU and are also consider to be the part of total computer system. These are called Peripheral Devices.

There are three types of peripheral : input, output and input-output peripherals.

ASCII Alphanumeric Characters:-

Input and output devices that are communicate with people and the computer are usually involve in the transfer of alphanumeric information to and from the device and computer.

ASCII (American Standard Code for Information Interchang) is the standard binary code for alphanumeric characters. It uses seven bit to code 128 characters.

The seven bits of the code are designed by b_1 through b_7, with b_7 being the most significant bit.

The letter A, for example is represented in ASCII as 1000001 (column 100, row 0001).

The ASCII code contains 94 characters that can be printed and 34 non printing character used for various control functions.

The printing characters contains 26 uppercase letter A to Z, the 26 lowercase letter a to z, ten numericals 0 to 9 and 32 special printable character such as %, * and $.

American Standard Code for Information Interchange (ASCII)

	$b_7 b_6 b_5$							
$b_4 b_3 b_2 b_1$	000	001	010	011	100	101	110	111
0000	NULL	DLE	SP	0	@	P	‘	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	B	S	c	s
0100	EOT	DC4	$	4	C	T	d	t
0101	ENQ	NAK	%	5	D	U	e	u
0110	ACK	SYN	&	6	E	V	f	v
0111	BEL	ETB	'	7	F	W	g	w
1000	BS	CAN	(	8	G	X	h	x
1001	HT	EM	)	9	H	Y	i	y
1010	LF	SUB	*	:	I	Z	j	z
1011	VT	ESC	+	;	J	[	k	{
1100	FF	FS	,	<	K	\	l	|
1101	CR	GS	-	=	L	]	m	}
1110	SO	RS	.	>	:	^	n	~
1111	SI	US	/	?	O	-	o	DEL

I/O Inteface :-

I/o Inteface provides a method for transferring information between internal storage and external I/O devices. Peripherals connected to a computer need special communication links for intefacing them with the CPU.

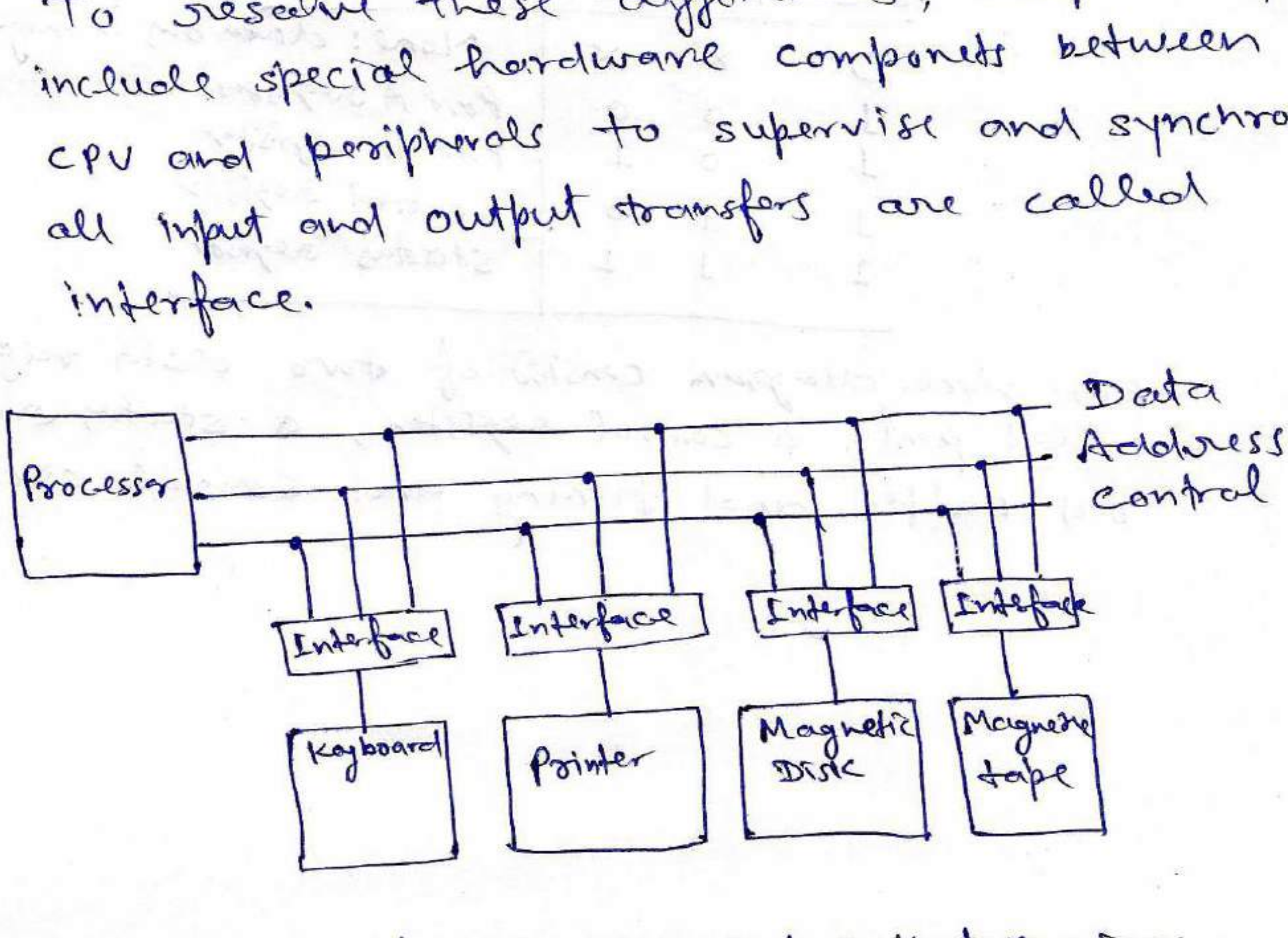
Requirement of I/o interface because of:

(i) Data transfer rate of peripheral are usually slower than the CPU.

(ii) Data codes & formates in peripheral differ from the word formate in CPU and memory.

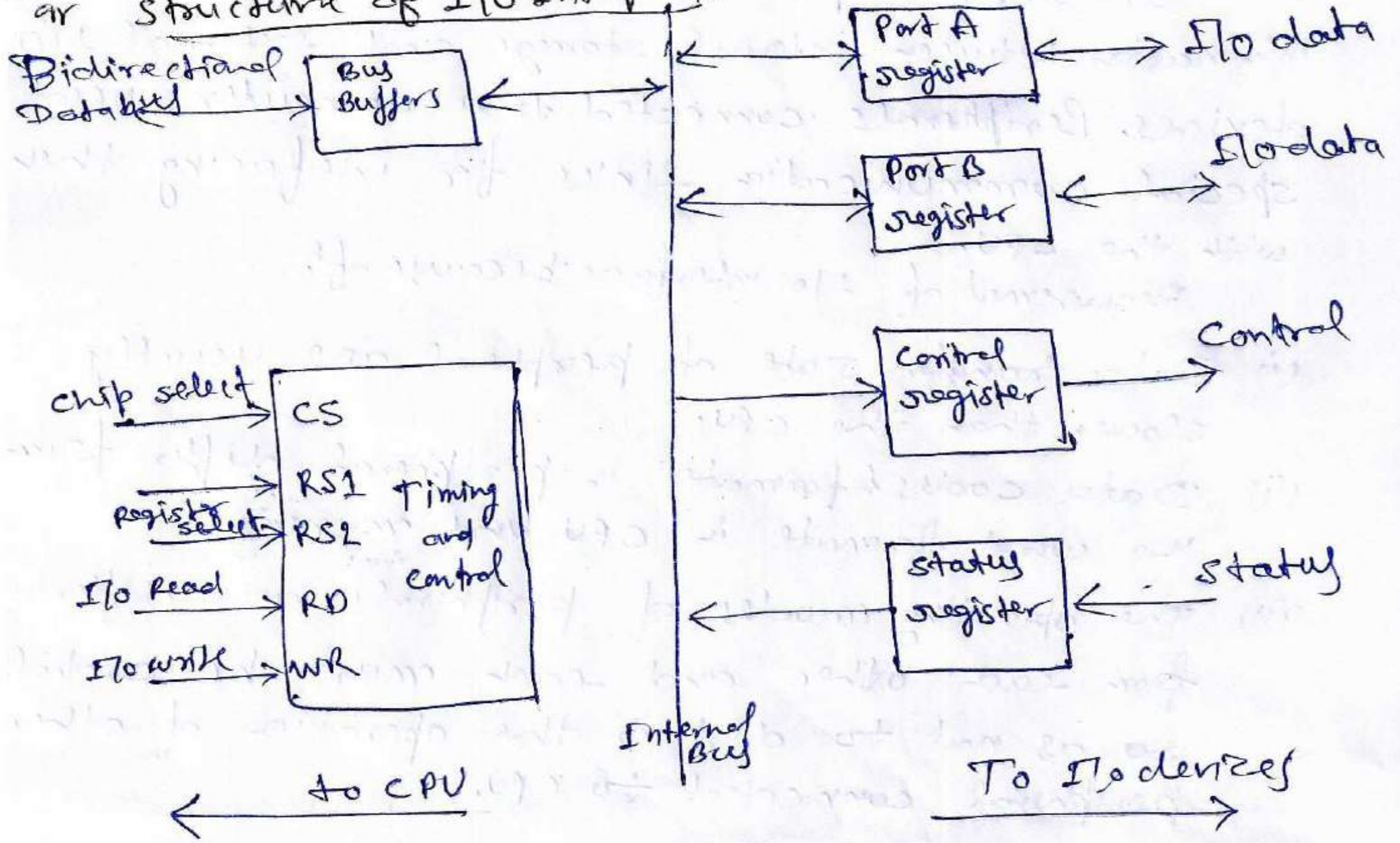
(iii) The operating modes of peripherals are different from each other and each must be controlled so as not to disturb the operation of other peripheral connected to CPU.

To resolve these differences, computer system include special hardware componets between CPU and peripherals to supervise and synchronize all input and output transfers are called interface.



Connection of I/O bus to Input-Output Devices

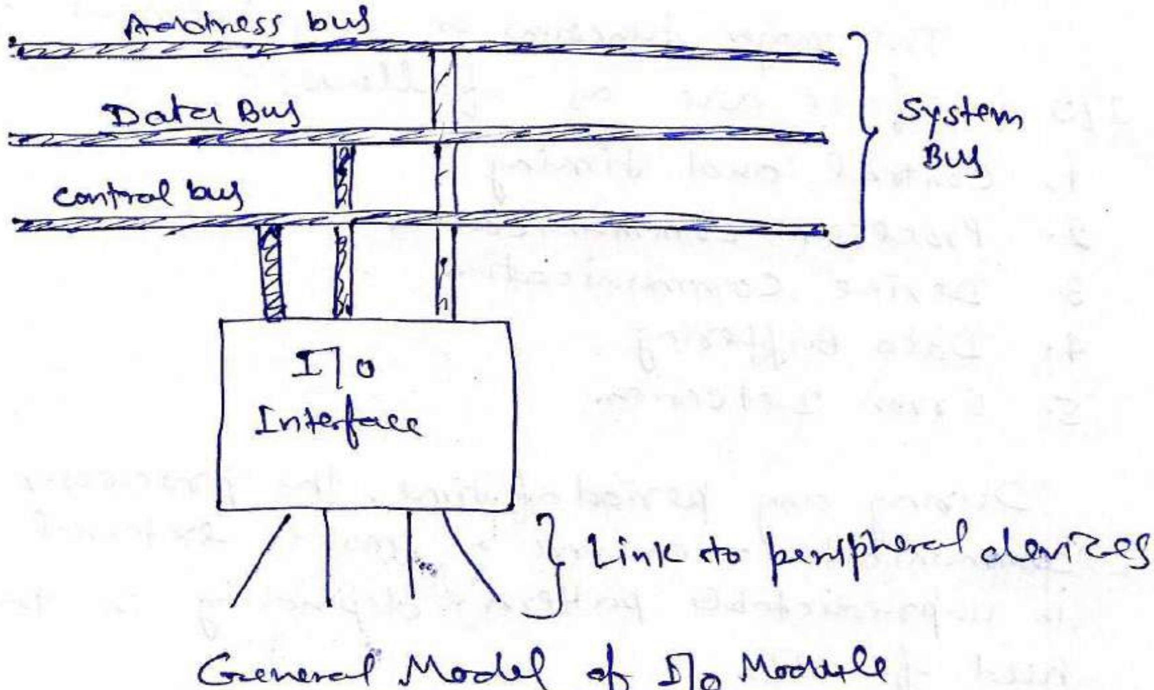
Example of I/O Interface (I/O Port) :-
or Structure of I/O Interface:-



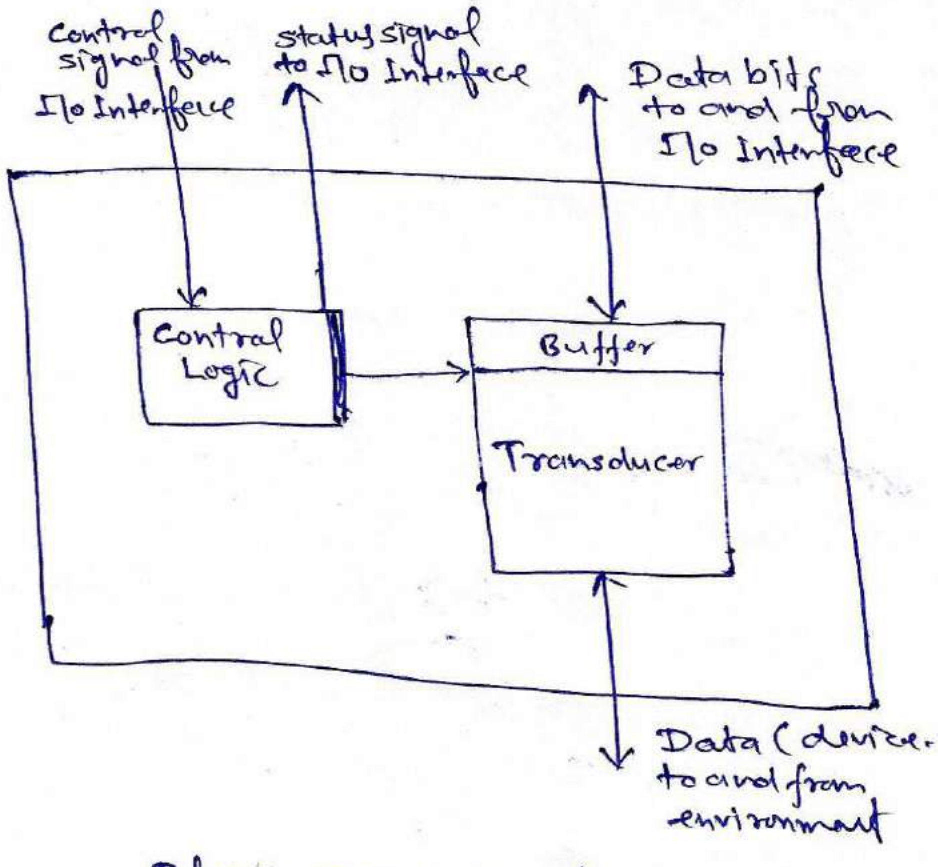
CS	RS1	RS0	Register select
0	x	x	None: data bus in high-impedance
1	0	0	Port A register
1	0	1	Port B register
1	1	0	Control register
1	1	1	Status register

Above block diagram consist of two data registers called port, a control register, a status register, bus buffer and timing and control circuits.

General Model of I/O Interface:-



General Model of I/O Module


Control signal from I/O Interface
status signal to I/O Interface
Data bits to and from I/O Interface
Control Logic
Buffer
Transducer
Data (device-unique) to and from environment

Block Diagram of an External Device

Functions of I/O Interface:-

The major functions or requirement for an I/O interface are as follows:

1. Control and timing
2. Processor communication
3. Device communication
4. Data Buffering
5. Error Detection.

During any period of time, the processor may communicate with one or more external devices in unpredictable pattern, depending on the program's need for I/O.

Modes of Data Transfer:-

Binary information received from an external device is usually stored in memory for later processing. The CPU executes the I/o instructructia and may accept the data temporarily, the ultimate destination of the data is memory unit.

Data transfer between central computer and I/o devices may be handle in a variety of modes.

1. Programmed I/o
2. Interrupt-initiated I/O
3. Direct memory access (DMA)

1- Programmed I/O :-

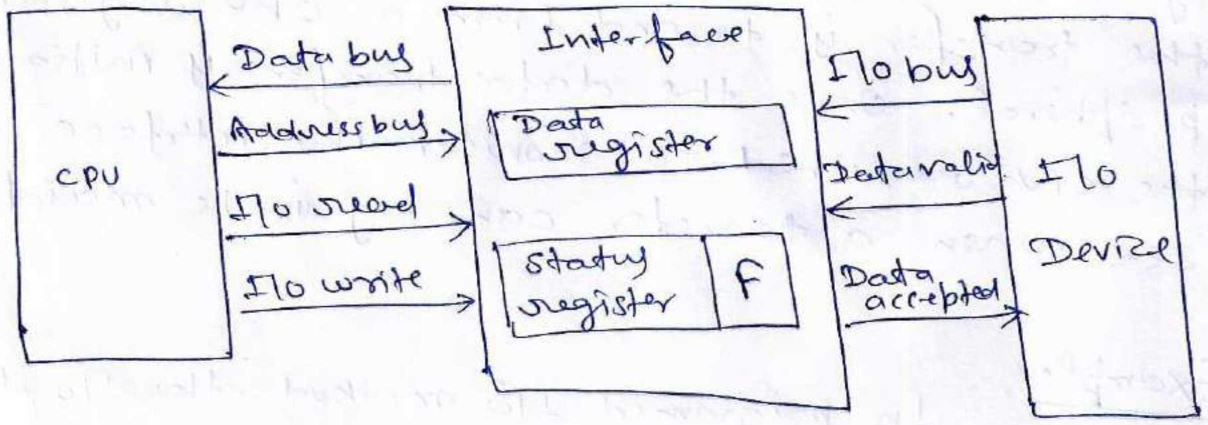
Programmed I/o operations are the result of I/o instructions written in the computer program. Each data item transfer is initiated by an instruction in the program. Here usually the transfer is to and from a CPU register and peripheral. Once the data transfer is initiated, the CPU required to monitor the interface to see when a transfer can again be maid.

Example:- In programed I/o method, the I/o device does not have direct access to memory. A transfer from an I/o device to memory requires the execution of several instructions by the CPU, including an input instruction to transfer the data from device to CPU and a store instruction to transfer data from CPU to memory.

Other instructions may be needed to verify that the data are available from the device and to count the numbers of words transfered.

A program is written for the computer to check the flag in the status register to determine if a byte has been placed in the data register by the I/O device. This is done by reading the status register into a CPU register and checking the value of the flag bit.

If the flag bit 1, the CPU read the data from data register. The flag bit is then cleared to 0 by either the CPU or the interface.

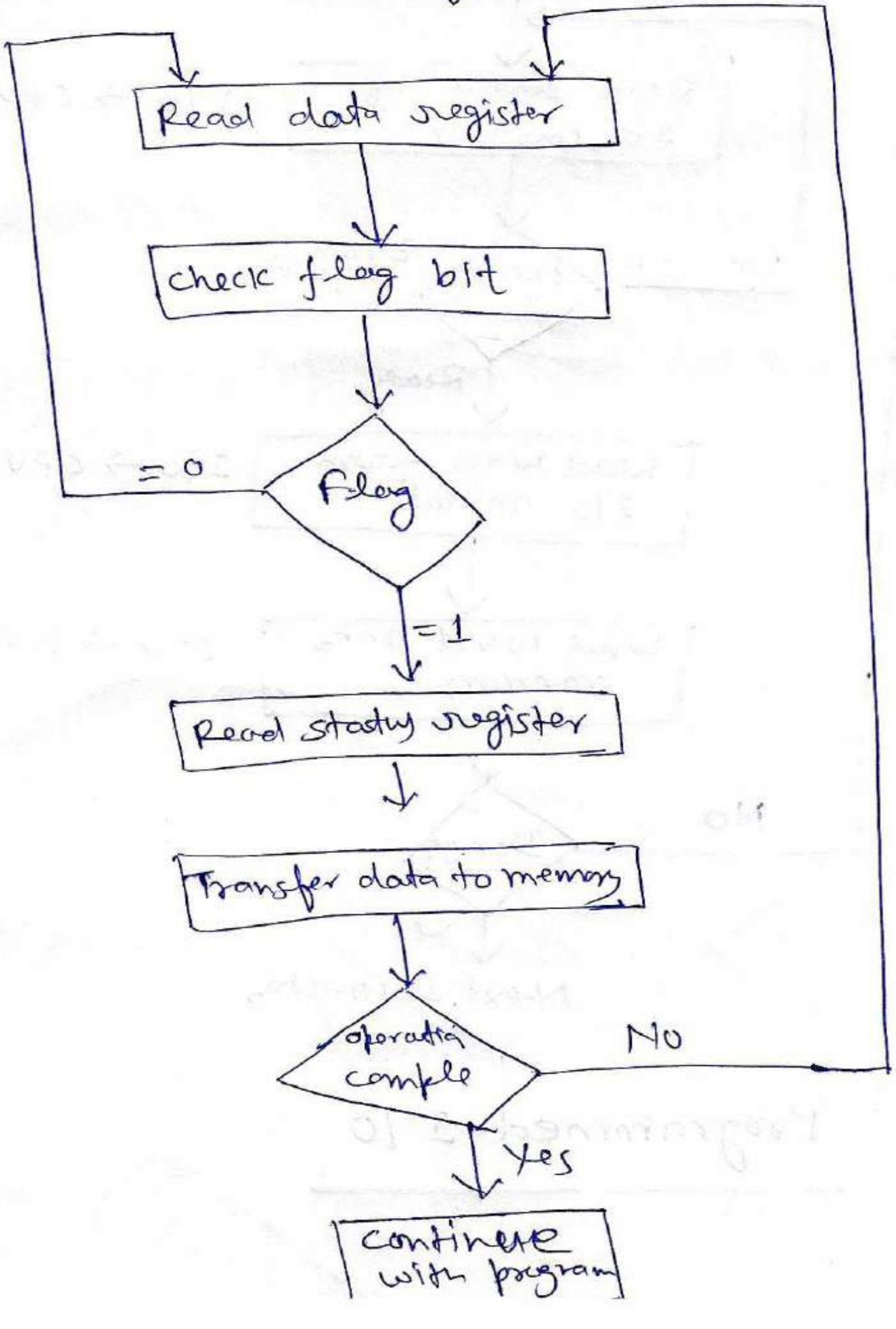


F = Flag bit

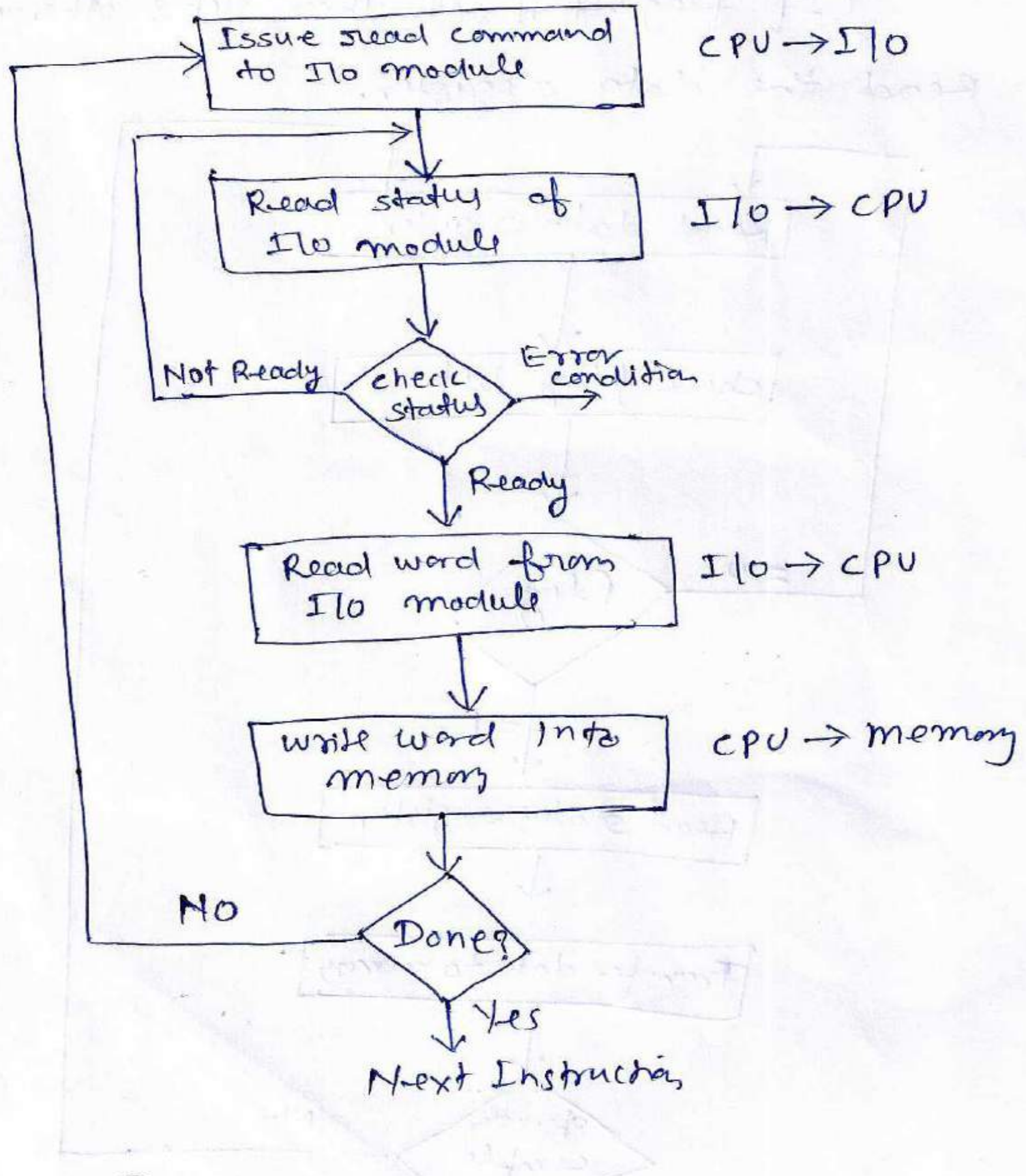
Data Transfer from I/O device to CPU

A flowchart of the program that must be written for the CPU is as follows. It assumed that the device is sending a sequence of bytes that must be stored in memory. The transfer of each bytes requires three instructions:

1. Read the status register
2. Check the status of flag bit
   (If flag bit is set then step 3 otherwise step 1)
3. Read the data register.



The programmed I/O method is particularly useful in small low speed computers or in systems that are dedicated to monitor a device continuously. The difference in information transfer rate b/w the CPU and the I/O device makes this type of transfer inefficient.

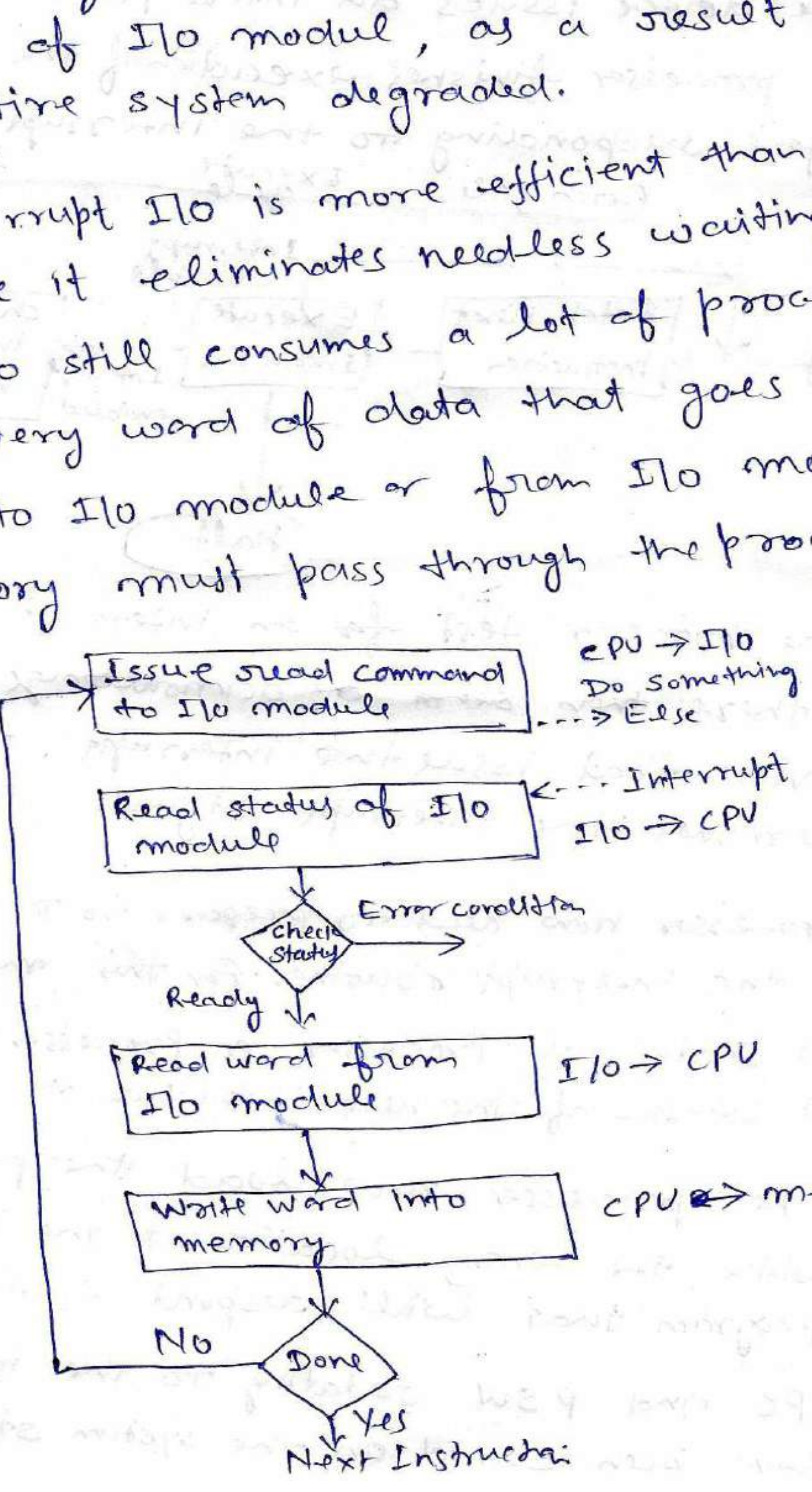


Programmed I/O

2. Interrupt-Driven I/O :-

The problem with programmed I/O is that the processor has to wait a long time for the I/o module of concern to be ready for either reception or transmission of data. The processor while waiting must repeatedly interrogate the status of I/O modul, as a result performance of the entire system degraded.

Interrupt I/O is more efficient than programmed I/o because it eliminates needless waiting. However interrupt I/o still consumes a lot of processor time, because every word of data that goes from memory to I/O module or from I/O module to memory must pass through the processor.

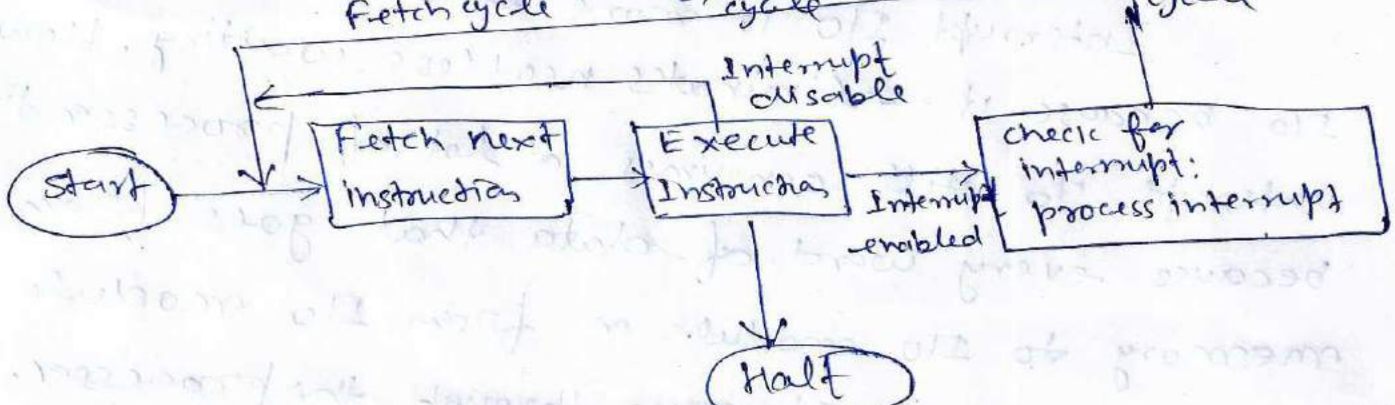


Interrupt Processing:

The occurrence of an interrupt triggers a number of events both in processor hardware and in software. When an I/o device complete an I/o operation, the fallowing sequence of events occurs:

1- The device issues an interrupt to the processor.

2. The processor finishes execution of the current instruction before responding to the interrupt.



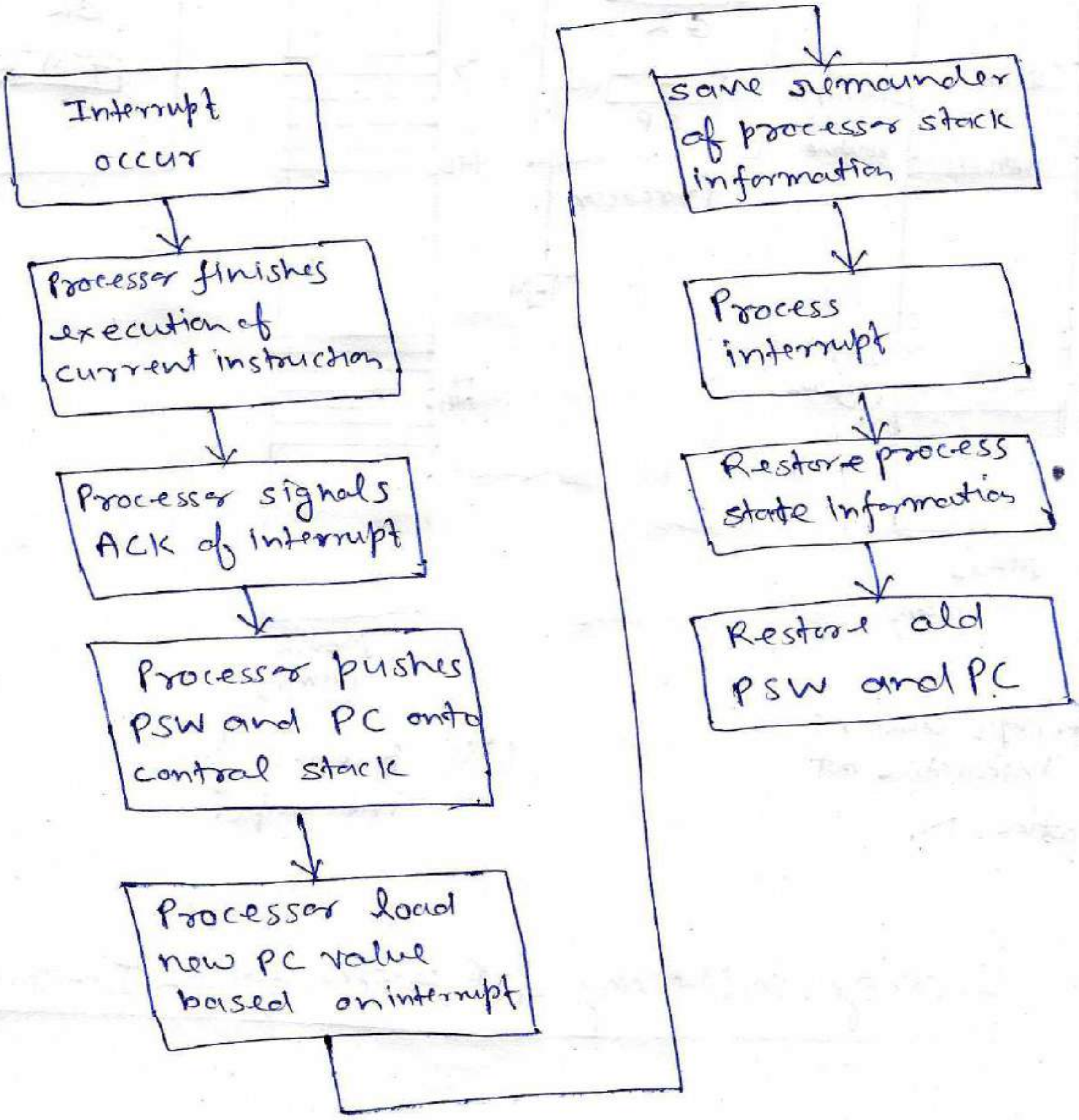
3. The processor test for an interrupt, if interrupt is there then send a acknowledge signal to the device that issue the interrupt. Then device removes the interrupt signal:

4. Processor now need to prepare to transfer control to the interrupt routine. for this minimum requirment
(a) Status of Processor or Processor Status Word (PSW)
(b) Location of the next instruction to be executed.

5. The processor now load the program counter with the entry location of the interrupt-handling program that will respond to this interrupt.

6. PC and PSW relating to the interrupted program have been saved on the system stack.

7. Interrupt handler next process the interrupt.
8. When interrupt processing is complete, the saved register values are retrieved from the stack and restored to the registers.
9. The final act is to restore the PSW and program counter values from the stack.



Interrupt occur
Processor finishes execution of current instruction
Processor signals ACK of interrupt
Processor pushes PSW and PC onto control stack
Processor load new PC value based on interrupt
save remainder of processor stack information
Process interrupt
Restore process state information
Restore old PSW and PC

Simple Interrupt Processing

Types of Interrupt :-

Basically interrupts are of two types. One is called vector interrupt and the other non-vector interrupt.

In non-vector interrupt, the branch address is assigned to a fixed location in memory.

In a vectored interrupt, the source that interrupts, supplies the branch information to the computer. this information is called the interrupt vector.

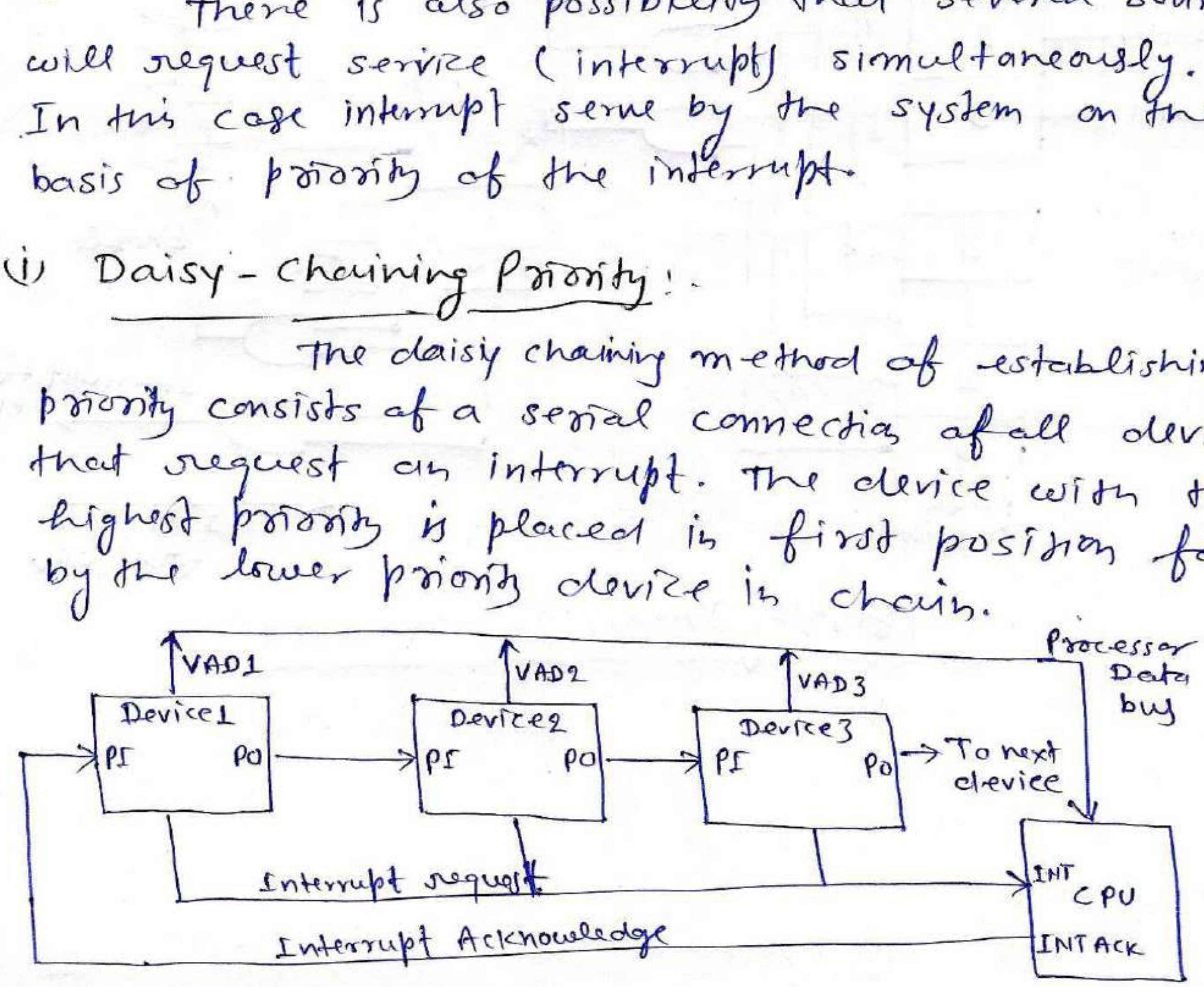
Priority Interrupt :-

When ever interrupt originate in system, the first task of interrupt system is to identify the source of the interrupt.

There is also possiblity that several sources will request service (interrupts) simultaneously. In this case interrupt serve by the system on the basis of priority of the interrupt.

(i) 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 first position followed by the lower priority device in chain.



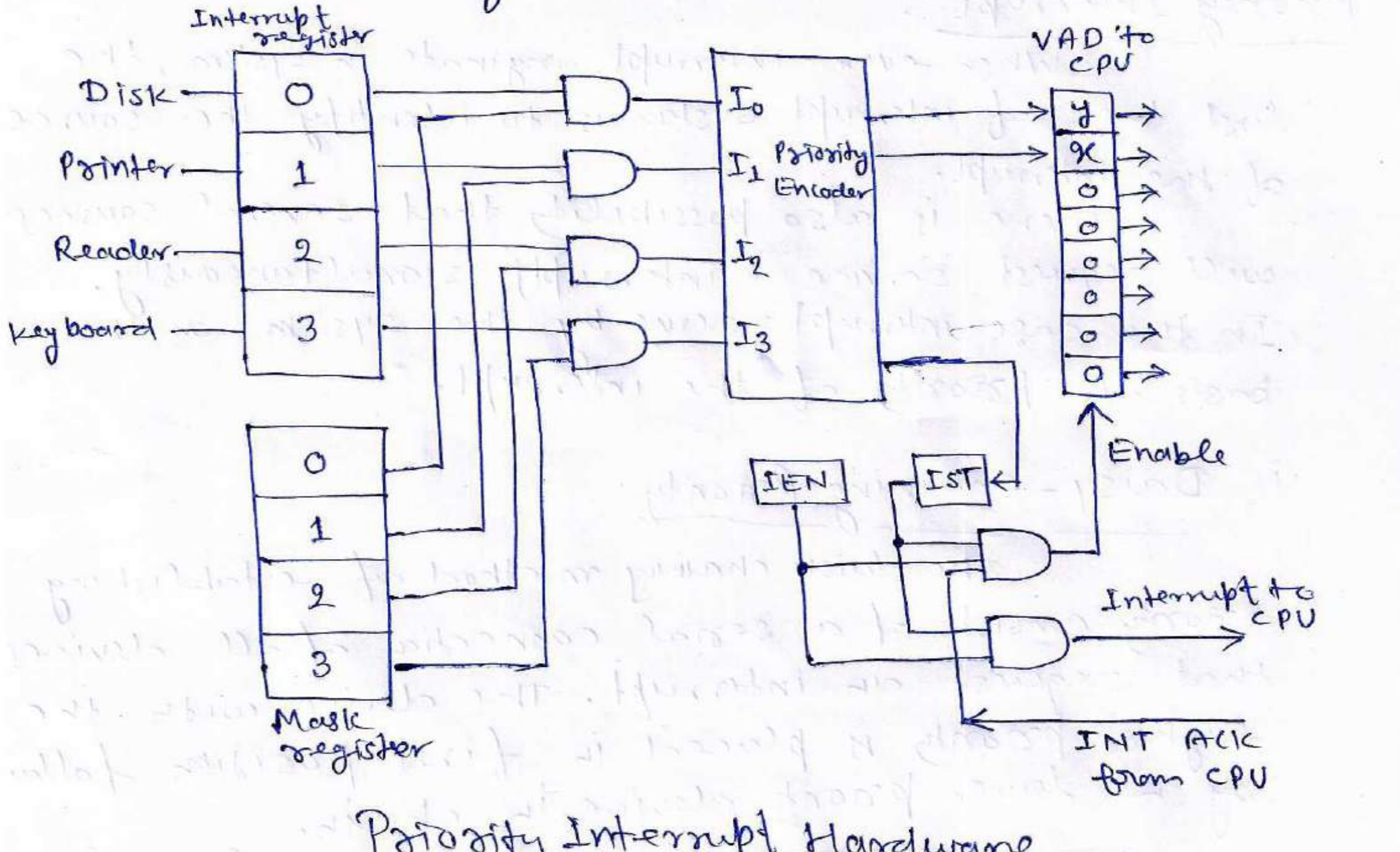
Daisy - chain Priority Interrupt

*PO → Priority Out
PI → Priority In
VAD → Vector Address

(ii) Parallel Priority Interrupt :-

The parrallel priority method uses a register whose bit are set separately by the interrupt signal from each device. Priority is established according to the position of the bits in the register.

There is a mask register whose purpose is to control the status of each interrupt request. The mask register can be programmed to disable lower priority interrupt while a higher priority device is being served.



Priority Interrupt Hardware

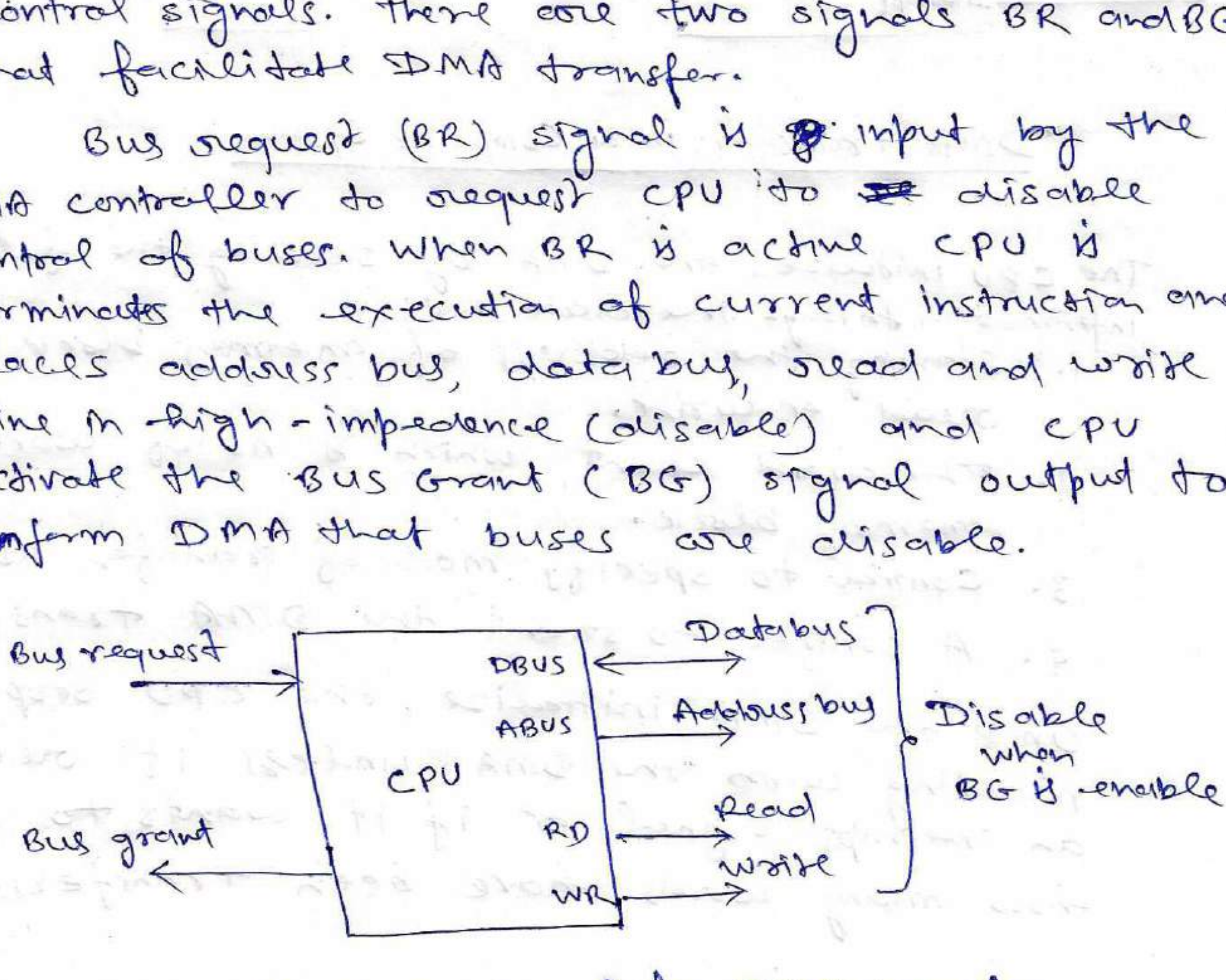
3- Direct Memory Access (DMA):-

The transfer of data between a fast storage device such as magnetic disc and memory is limited by the speed of the CPU. Removing the CPU from the path the peripheral devices manage the memory buses directly would improve the speed of transfer. This transfer technique is called direct memory access (DMA).

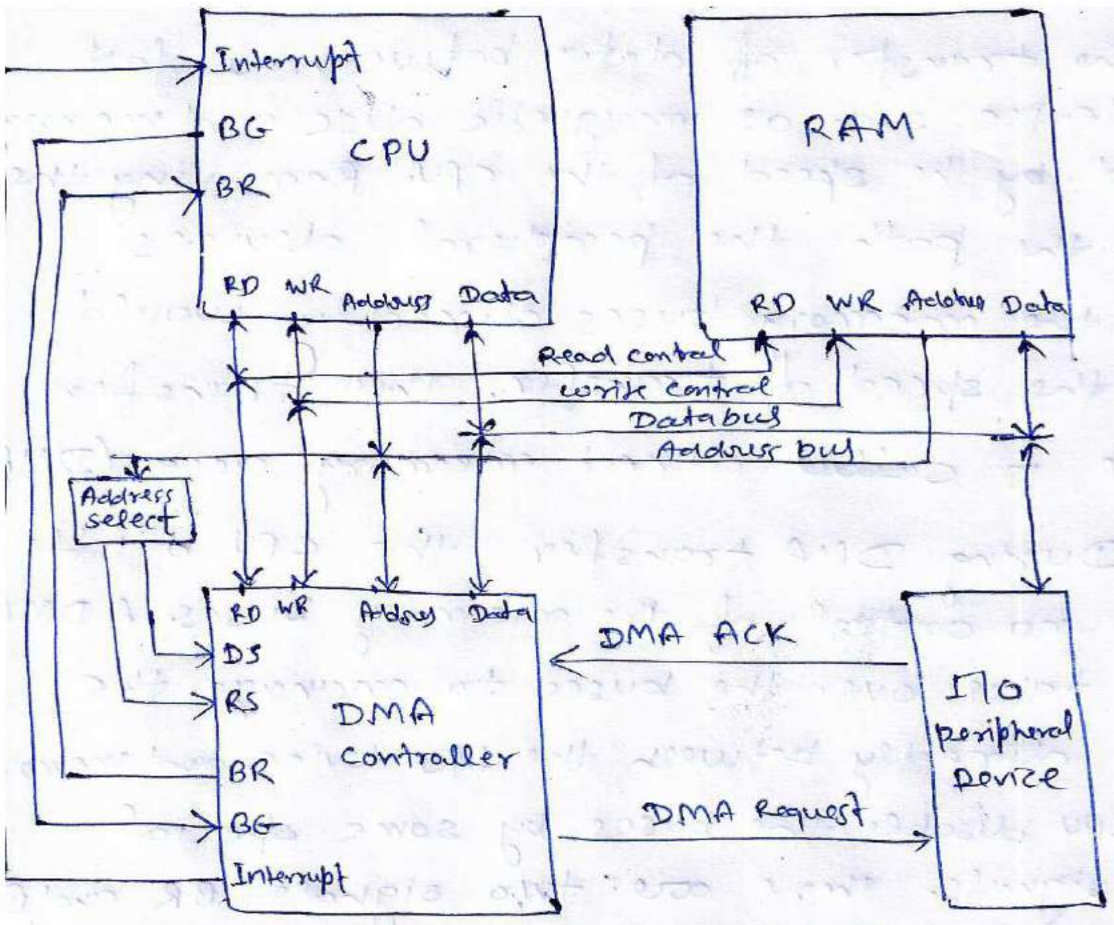
During DMA transfer, the CPU is idle and has no control of the memory buses. A DMA controller takes over the buses to manage the transfer directly between the I/o device and memory.

CPU disable the buses by some special control signals. There are two signals BR and BG that facilitate DMA transfer.

Bus request (BR) signal is input by the DMA controller to request CPU to ~~de~~ disable control of buses. When BR is active CPU is terminates the execution of current instruction and places address bus, data bus, read and write line in high-impedance (disable) and CPU activate the Bus Grant (BG) signal output to inform DMA that buses are disable.



CPU bus signal for DMA transfer



DMA transfer in a Computer System

The CPU initilizes the DMA by sending the following information through the data bus:

1. Starting the address of memory block for read or write.
2. The word count, which is no of word in memory block.
3. Control to specify mode of transfer RD or WR.
4. A control to start the DMA transfer.

Once the DMA initialize, the CPU stop communicating with the DMA unless it receives an interrupt signal or if it wants to check how many words have been transferred.

DMA Transfer :-

(i) When peripheral device sends a DMA request, the DMA controller activates the BR line, informing the CPU to disable the buses.

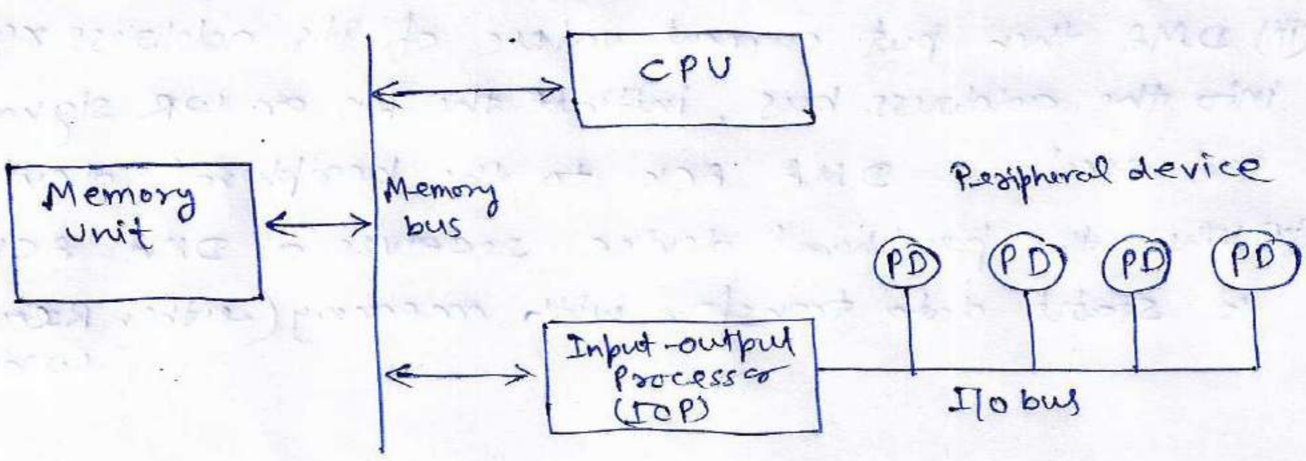
(ii) The CPU responds with BG line, informing the the DMA that its buses are disable.

(iii) DMA then put current valeve of its address register into the address bus, initiate the RD or WR signal and sends a DMA ACK to the peripheral device.

(iv) When the peripheral device receives a DMA ACK, it stadt data transfer with memory (either Read or write).

Input-Output Processor (IOP):-

An Input-output Processor (IOP) may be classified as a processor with direct memory access capability that communicate with I/o devices. In this configuration the computer system can be devided into a memory unit, a number of processors comprised of the CPU, and one or more IOPs.



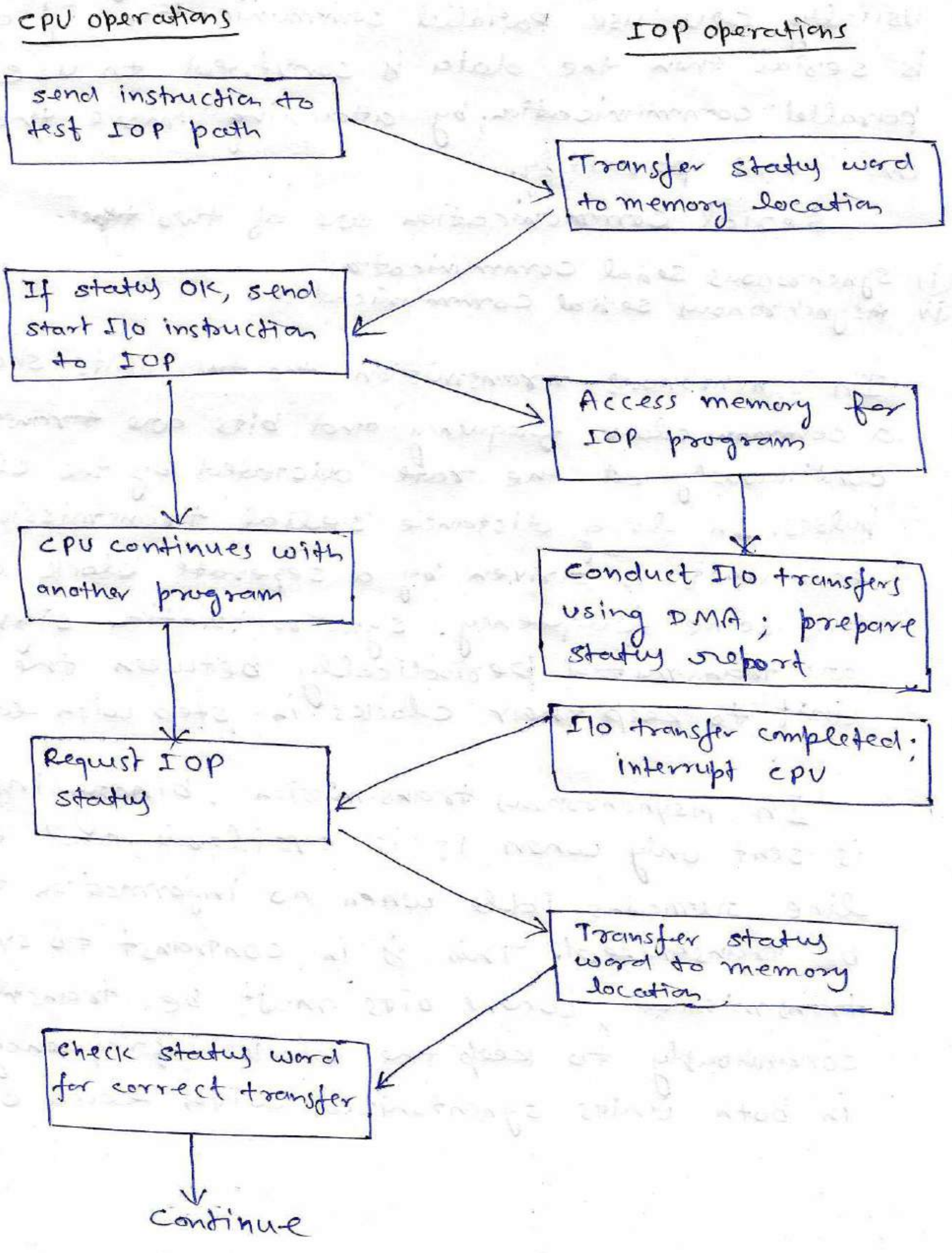
Block Diagram of a computer with I/o processor

IOP is similar to a CPU except that it is designed to handle the details of I/o processing. Unlike the DMA controller that must be set up entirely by the CPU, the IOP can fetch and execute its own instructions.

"Instead of having each interface communicate with the CPU, a computer may incorporate one or more external processor and assign them the task of communicating directly with all I/o devices."

CPU - IOP Communication :-

Memory unit act as a message center where each processor leaves information for the other.


CPU operations
IOP operations
send instruction to test IOP path
Transfer status word to memory location
If status OK, send start I/O instruction to IOP
Access memory for IOP program
CPU continues with another program
Conduct I/O transfers using DMA ; prepare status report
I/O transfer completed ; interrupt CPU
Request IOP status
Transfer status word to memory location
check status word for correct transfer
Continue

CPU - IOP Communication

Serial Communication :-

serial communication refer to device that cannot handle more than one bit of data at any giventime. It requires only one wire and it is very slow. Usually CPU use Parallel communication, if device is serial then the data is converted to use parallel communication, by attaching more than one wire parallely.

Serial communication are of two types-

(i) Synchronous serial communication
(ii) Asynchronous serial communication.

In synchronous transmission, the two units share a common clock frequency and bits are transmitted continuously at the rate dictated by the clock pulses. In long distance selial transmission, each unit is driven by a seperate clock of the same frequency. synchronization signals are transmitted periodically between the two unit to keep their clocks in step with each other.

In Asynchronous transmission, binary information is sent only when it is available and the line remains idle when no information to be transmitted. This is in contranst to synchrous transmission where bits must be transmitted continuously to keep the clock frequency in both units synchronized with each other.