12/12/23

UNIT :-05

INPUT / OUTPUT

Input Output Interface :- Input Output Interface enables transfer of data b/w internal storage and external input-output device.

Input output interface is a mechanism which provides a communication path b/w processor and peripheral devices.

The purpose of use of input output interface is to resolve the differences b/w the CPU and peripheral devices. These differences are :-

(1) As peripherals are electromagnetic and electromechanical devices, and CPU is an electronic device, signal conversion becomes necessary whenever required.

(2) Due to difference in data transfer rate, synchronization of peripherals with the CPU may be required.

(3) Data codes and formats of each peripheral is different than CPU, a standard conversion is required

(4) The operating modes of all peripherals are different. Thus, each of the peripherals connected to the CPU should be controlled so that the operation of the peripherals continue without any disturbance.

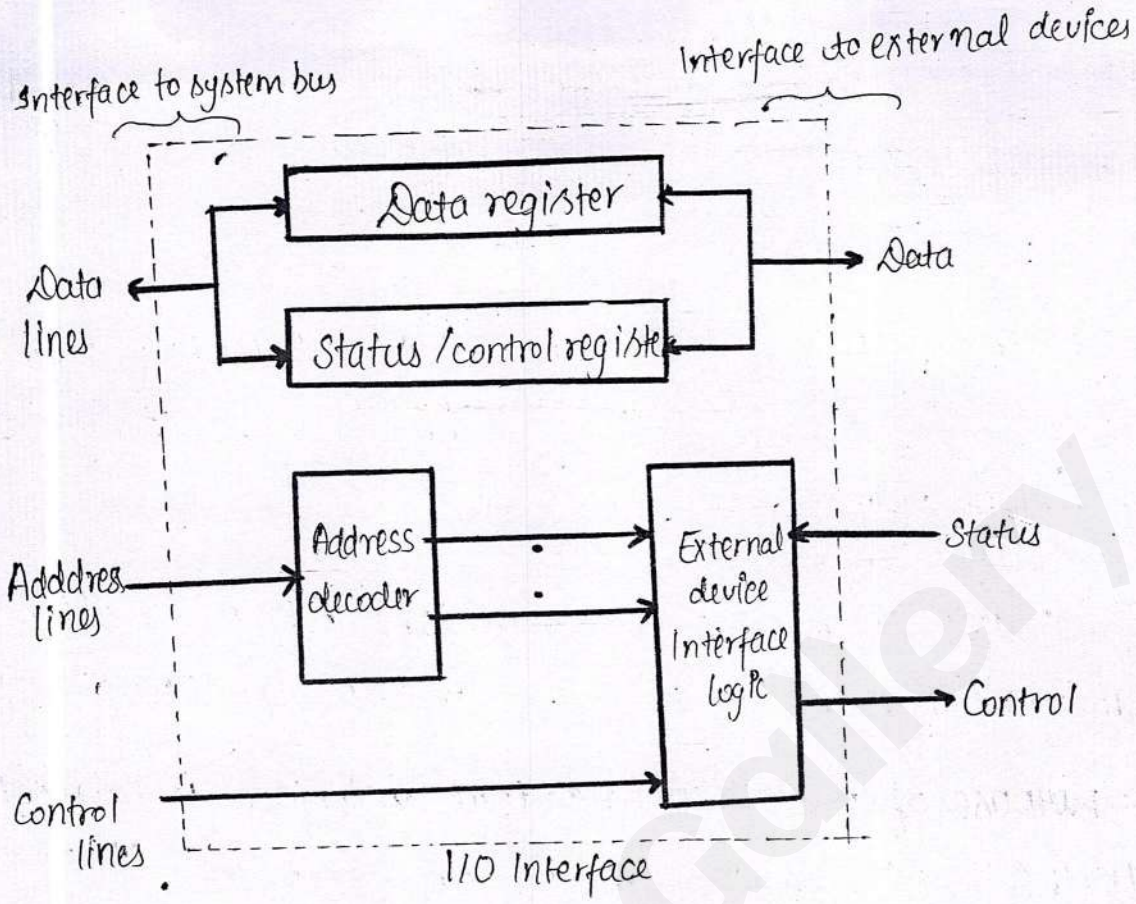
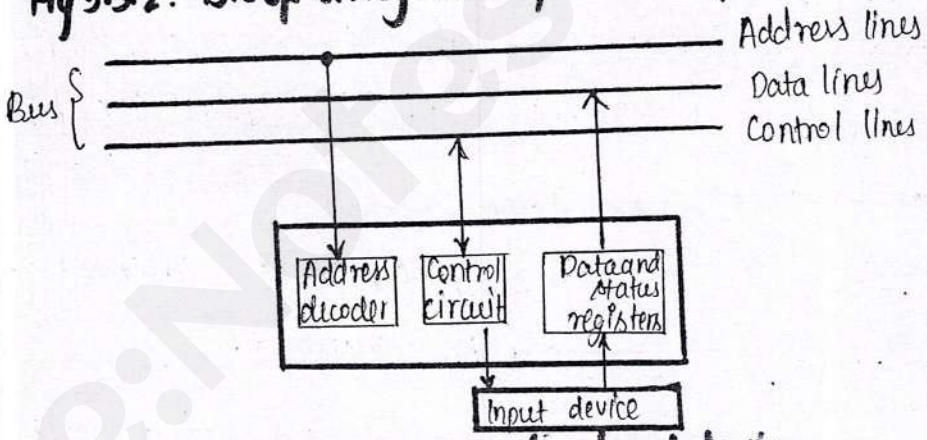
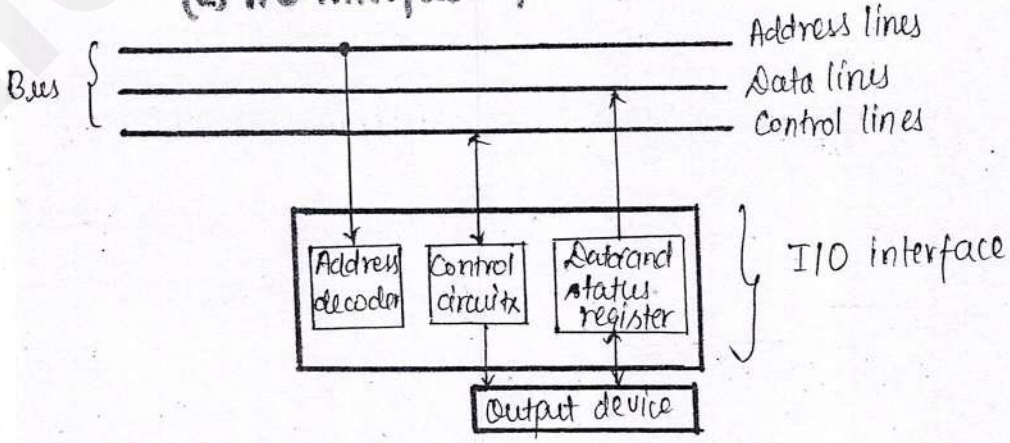


Fig 5.3.2. Block diagram of I/O interface



(a) I/O interface for Input device



(b) I/O Interface for output device

Functions of I/O Interface

1. **Control Timing:-** The input/output interface co-ordinates the flow of data b/w internal and external storage or devices through the control timing signals.

2. **Processor Communication:-** Such communication involves different types of communication signals that are —
   1. Control signals from processor to I/O interface.
   2. Exchange of data b/w processor and I/O interface.
   3. Recognise the particular I/O among various I.O devices

3. **Device Communication:-** The I/O interface must be able to perform device communication which involves commands, status information and data

4. **Data Buffering:-** It is also an essential task due to the difference of speed (data transfer rate) of the peripherals

5. **Error Detection:-** The I/O interface is also responsible for error detection and reporting to the processor.

Input Output Processor

The Input Output Processor (IOP) has an ability to execute I/O instructions and execution of I/O operation. The I/O instructions are stored in main memory. When input output transfer is required, the CPU initiates an I/O transfer by instructing an I/O channel to execute program stored in main memory

An I/O processor can fetch and execute its own instruction. The IOP supports multi processing environment. IOP and CPU can do processing simultaneously.

IOP does all work involved in I/O transfer including device setup, programmed I/O and DMA operation.

* Communication b/w CPU and IOP

The sequence of operations carried out during CPU and IOP communication are-

1. CPU checks the existence of I/O path by sending an instruction.
2. In response to this, IOP puts the status word in the memory showing the condition of IOP (Busy, Ready etc.).
3. CPU checks the status word and if all conditions are OK, it sends the instruction to start I/O transfer along with the memory address where the IOP program is stored.
4. After this CPU continue another program.
5. IOP now conducts the I/O transfer using DMA and prepares status report.
6. On completion of I/O transfer, IOP ~~sets~~ sends an Interrupt request to the CPU.
7. The CPU responds to the Interrupt by issuing an instruction to read the status from the IOP.
   The status indicates whether the transfer has been completed or any error occured during the transfer.

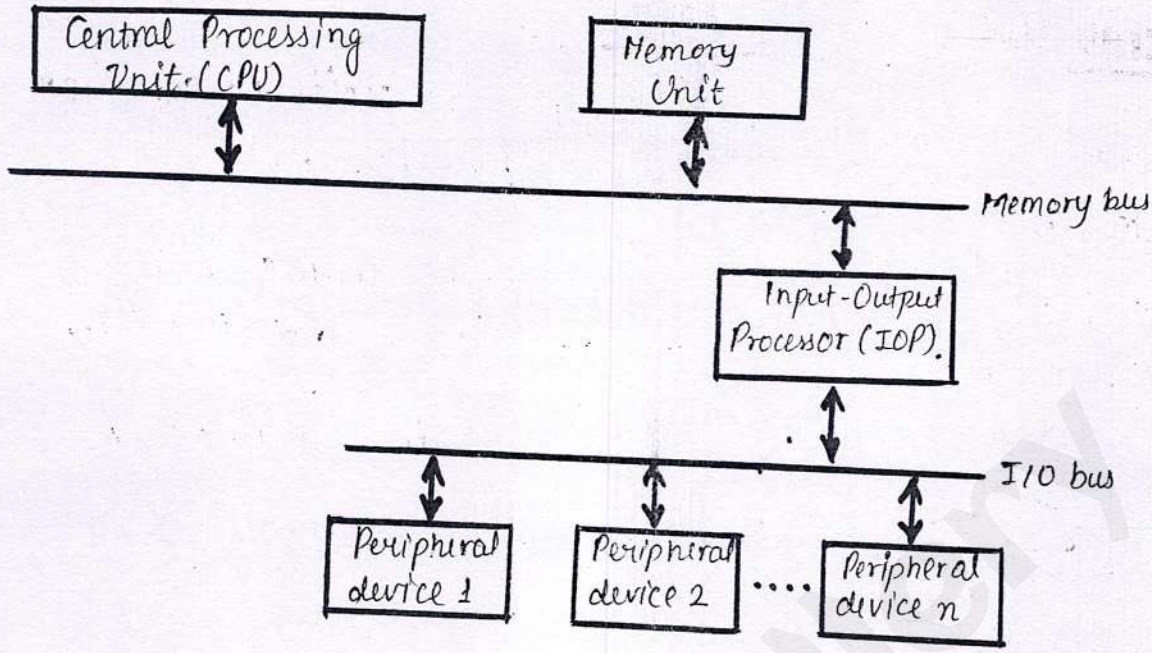


Fig 5.7.1. Block diagram of a computer with I/O processor

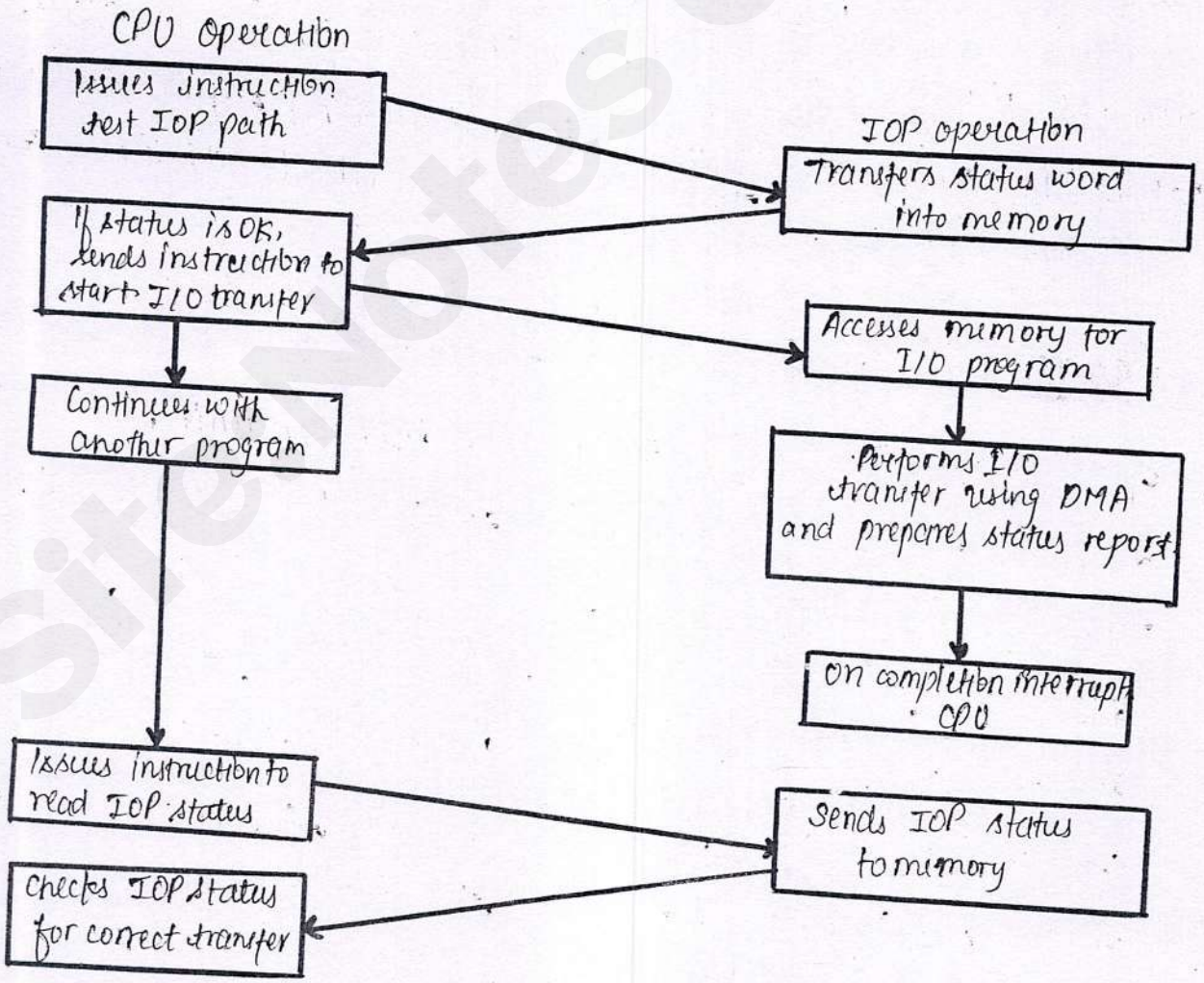


Fig 5.7.2 CPU and IOP communication

Asynchronus Data Transfer

The data transfer blw two independent units by common clock or global clock is known as synchronus data transfer.

When both independent units are using their own private clock for data transfer then it is called asynchronus data transfer.

Two methods are used to transfer data in asynchronus method blw two independent units :-

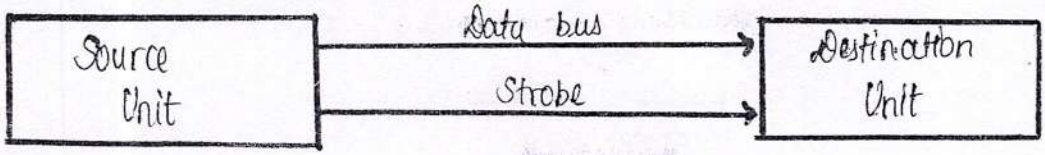
1. Strobe Control
2. Handshaking

1. Strobe Control :- In strobe control asynchronous data transfer, a single control line (strobe pulse) is used to indicate to the other unit when the transfer has to occur. When the strobe signal may be activated by either the source or destination unit. There are two methods of strobe control.

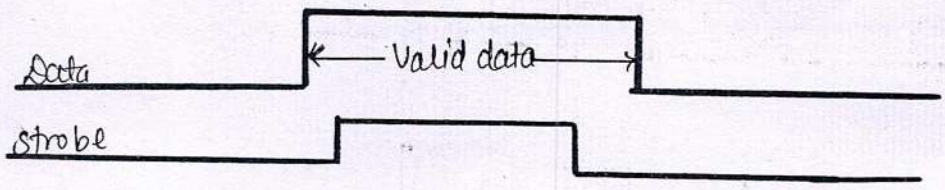
1. Source initiated strobe control data transfer

In this, the strobe signal informs the destination unit that the valid data is available on the data bus ~~on~~ from the source unit.

The source unit first places the data on the data bus. After the data setup the source activates the strobe pulse. The strobe signal and data remain in active state for a sufficient time to allow the destination unit to receive the data. After that strobe is inactivated and source removes the data from the data bus.



(a) Block diagram

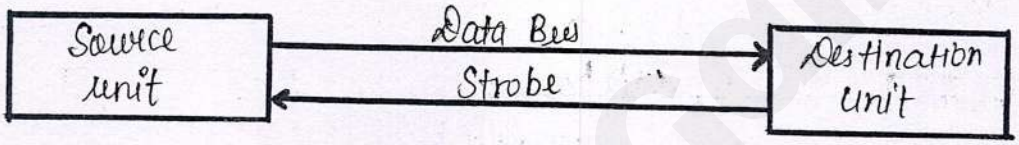


(b) Timing diagram

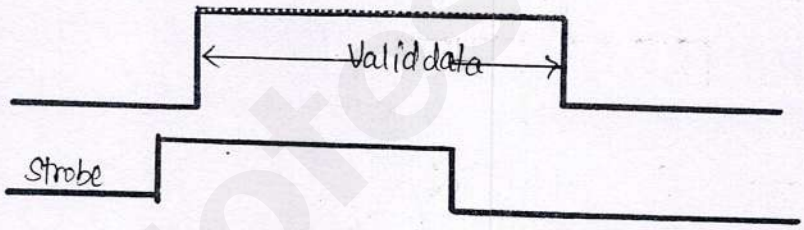
2. Destination Initiated strobe for data transfer

In this case, first the destination unit activate the strobe pulse informing the source to provide the data.

The source unit response by putting valid data on the data bus.



(a) Block diagram



(b) Timing diagram

2. **Handshaking:-** The disadvantage of strobe control method is that the source unit that initiates the transfer has no idea whether the destination unit has actually accepted the data sent by it. Similarly a destination unit that initiates the transfer has no idea whether the source unit has actually placed the data on the bus. The handshake method solves this problem by introducing one ~~or~~ more signal called Acknowledge signal. The acknowledge signal provides a reply to the unit that initiates the data transfer.

Source initiated data transfer using handshaping

In this method, source unit initiates the data transfer by placing the data on the bus and enabling its data valid signal. In response to this, the destination unit accepts the data and it sends data accepted (acknowledgement) signal to the source unit. The source unit then disables its data valid signal and destination unit disables data accepted signal and the system goes into its initial stage.

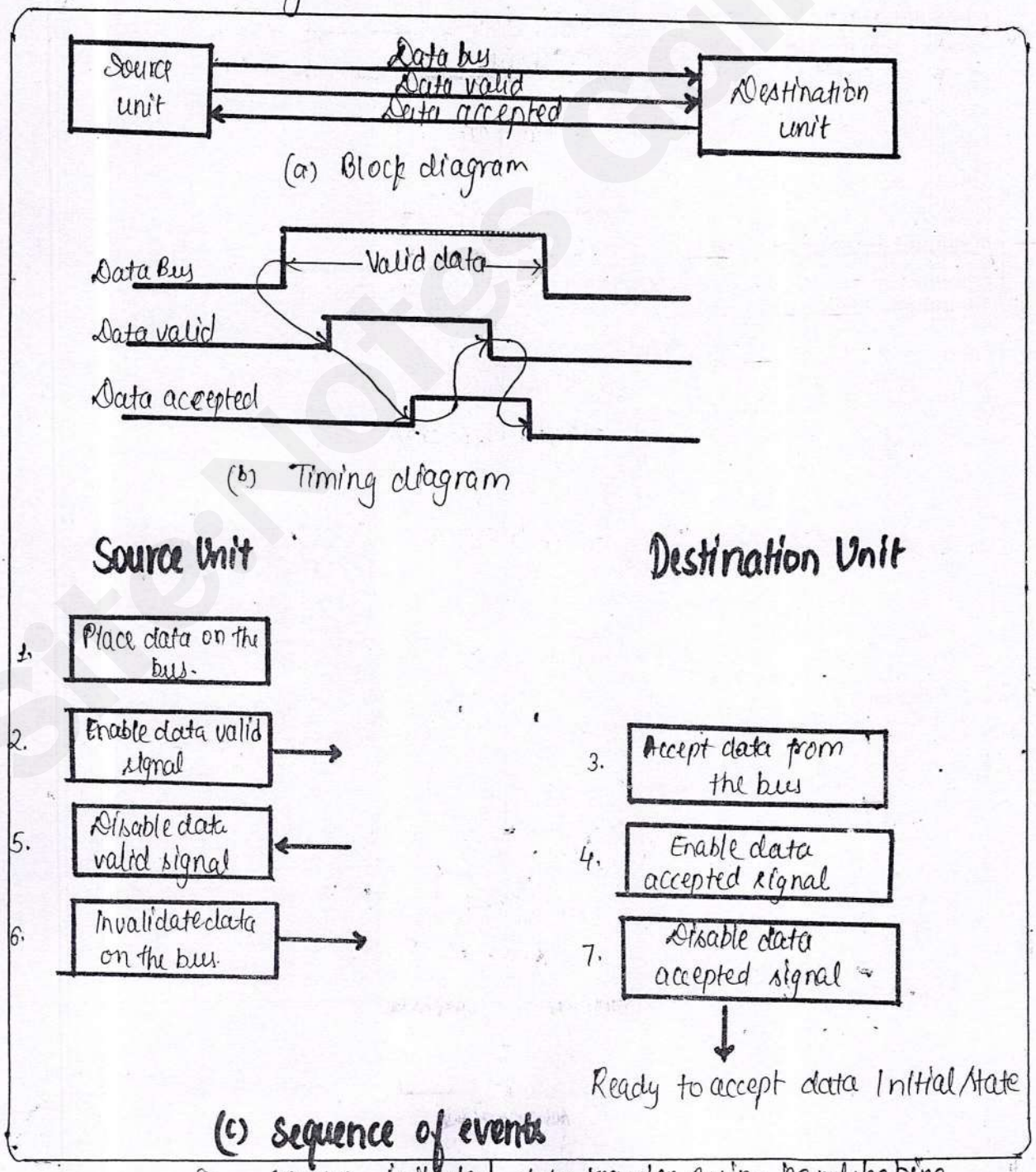
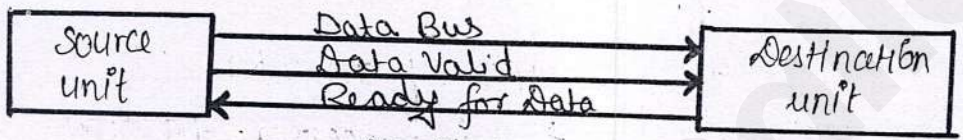


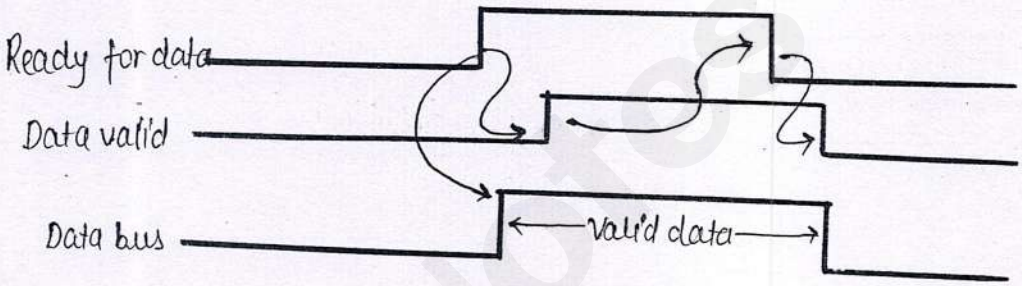
Fig. Source initiated data transfer using handshaping

-:- Destination Initiated data transfer using handshaking -

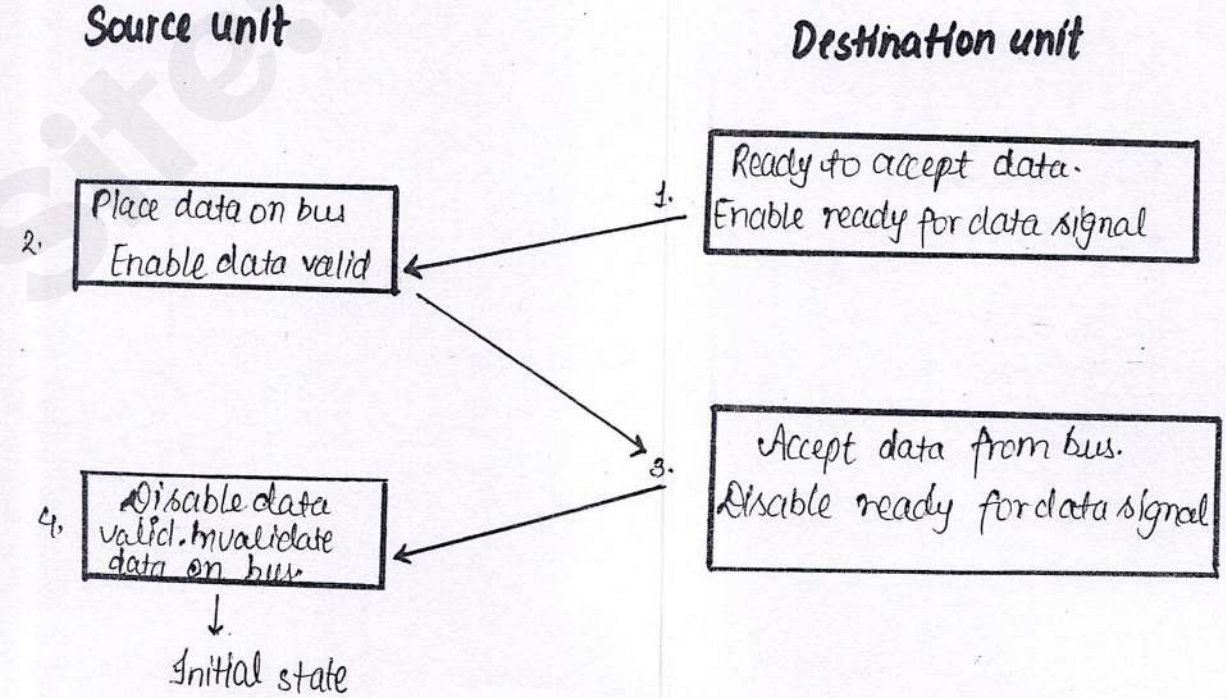
Here, the destination unit activates Ready for data signal when it is ready to accept data from source unit. In response to that, the source unit places the data on the bus and initiates the data valid signal. The destination unit then accepts the data from the data bus and disables the ready for data signal. Then the source unit disables data valid signal, after that system goes to the initial stage.



(a) Block diagram



(b) Timing diagram



(c) Sequence of events

Interrupt :- An interrupt is a process which allows the processor to suspend its current execution and respond to external or internal request.

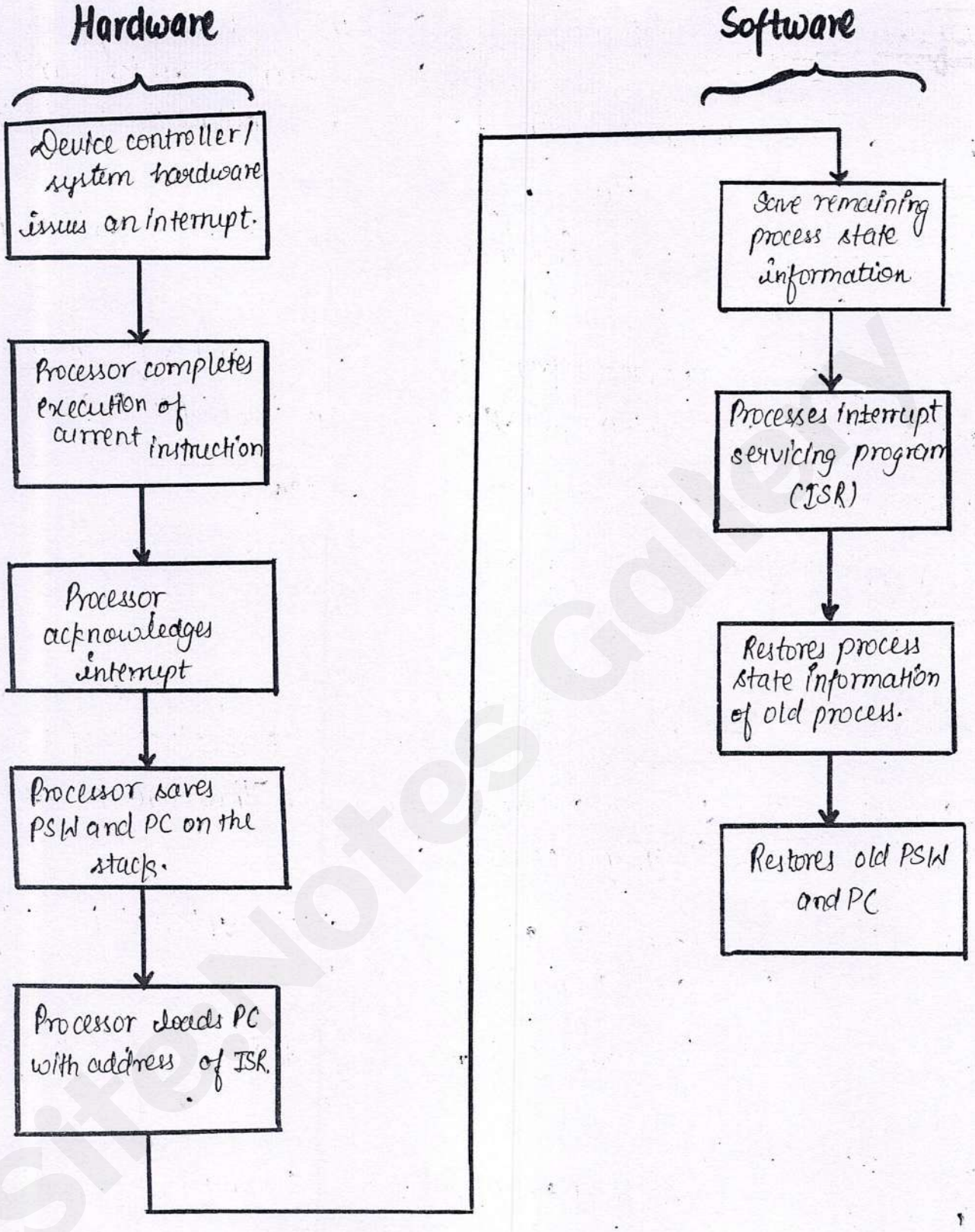
Interrupt is the signal which diverts the concentration of CPU from one program to another for sometime

An interrupt can be provided to the processor in following two ways:

1. Internally by an instruction of a program
2. Externally by any peripheral devices.

Interrupt Handling:- The occurence of an interrupt causes no. of events both in processor, hardware and software. Following sequence of events occurs:-

1. Device issues an interrupt signal.
2. Processor finishes/completes its current instruction before responding to the interrupt.
3. The processor checks the interrupt
4. Processor transfers the control to the interrupt after saving the information of the current executing program in the register called program status word. (PSW)
5. Processor executes the interrupt.
6. When interrupt processing is complete, the saved information is retrieved from the PSW. Now the normal execution resumes



Types of Interrupts

1. Software & Hardware Interrupt
2. Maskable and non-maskable Interrupt.
3. Vector & non-vectored Interrupt.

Software & Hardware Interrupt:- The interrupt signal generated from external devices and I/O devices are made interrupt to CPU when the instructions are ready is known as hardware interrupt.

Hardware interrupt can be internal (through microprocessor) and external (peripheral devices).

The interrupt signal generated from internal devices and software programs need to access any system call then, software interrupts are present.

Software interrupts can be divided into two types-

① Normal Interrupt

② Exceptions

Maskable & Non-Maskable Interrupt:- Some interrupt request are required immediate response from the processor either serious damage may occur. They are programmed as non-maskable interrupt or enable interrupts.

Some interrupts are referred as maskable or disable interrupt because they may not be responded immediately or processor may reject it.

Vector & Non-Vectored Interrupt:- In a vectored interrupt, the branch address is assigned to the fixed location in the memory.

In a non-vectored interrupt the source supplies the branch information to the processor.

S.No.	Vectored interrupt	Non-vectored interrupt
1.	Vectored interrupt are those interrupt that generates the interrupt request, identifies itself directly to the processor	Non-vectored interrupt are those in which vector address is not pre-defined.
2.	Vectored interrupt has memory address.	A non-vectored interrupt do not have memory address.

3.	Vectored interrupt have fixed memory location for transfer of control for normal execution.	Non-vectored interrupt do not have fixed memory location for transfer of control for normal execution.
4.	The vectored interrupt allows the CPU to be able to know what ISR to carry out in software.	When a non-vectored interrupt received, it jump into the program counter to fixed address in hardware.
5.	Response time is low.	Response time is high.
6.	TRAP is a vectored interrupt.	INTR is non-vectored interrupt.

Modes of Transfer

There are three modes of transfer:-

① Programmed I/O transfer
② Interrupt initiated I/O transfer
③ DMA (Direct memory Access)

Programmed I/O transfer:- In programmed I/O transfer, each data transfer is initiated by an instruction in a computer program.

If any computer system I/O operation are completely controlled by the processor then that system is said to be using programmed I/O.

When such a technique is used, processor executes program that initiate, direct and terminate the I/O operation, including sensing device status (↳direction), sending a read or WRITE command and (checking) tranfering the data.

It is the complete responsibility of the processor so, processor requires constant monitoring of the input output interface and IOP.

No other task will be performed during the data transfer, it is a time consuming process., it keeps processor busy.

Difference b/w Serial and Parallel Communication

Serial Communication	Parallel Communication
• Data transmitted serially, one bit at a time	• Data is transmitted parallely, all bits at a time.
• Low speed.	• High Speed
• It has single transmission line.	• It has multiple transmission lines.
• Serial communication do not have any crosstalk problem	• Parallel Communication may have crosstalk problem.
• Less expensive	• More expensive.
• The bandwidth is higher.	• The bandwidth is lower.
• Serial communication ~~do not have~~ even works at high frequencies.	• Parallel communication may not work properly at high frequencies.
• It is not affected with noise problems.	• It may suffer with noise problems.
• It covers long distance when compared to parallel communication.	• It is used for short distances.
• Example:- Serial communication b/w a computer and modem.	• Example:- Parallel communication b/w a motherboard and hard disk.

Difference blw Programmed I/O & Interrupt I/O Transfer

	Programmed I/O	Interrupt I/O
1.	In programmed I/O, processor has to check each I/O device in sequence and in effect 'ask' each one if it needs communication with the processor. This checking is achieved by continuous polling cycle and hence processor cannot execute other instructions in sequence	External asynchronous input is used to tell the processor that I/O device needs its service and hence processor does not have to check whether I/O device needs it service or not
2.	It is implemented without interrupt hardware support.	It is implemented using interrupt hardware support.
3.	It does not need initialization of stack.	It needs initialization of stack
4.	During polling processor is busy and therefore, have serious and decre-mental effect on system throughput.	In interrupt driven I/O, the processor is allowed to execute its instructions in sequence and only stop to service I/O device when it is told to do so by the device itself. This increa-ses system throughput.
5.	System throughput decreases as no. of I/O devices connected in the system increases.	System throughput does not depend on no. of I/O devices connected in the system.
6.	It does not depend on interrupt status.	Interrupt must be enabled to process interrupt driven I/O.

Interrupt Initiated I/O Transfer

Time consuming limitation of programmed I/O is avoided by interrupt facility. In the mean time, the processor can execute another program, when the interface determines, the device is ready for data transfer, it generates an interrupt request to the computer.

On detecting the external interrupt, the processor stops executing program and serve the I/O transfer.

After this servicing is completed, the processor would resume exactly where it left off.

DMA (Direct Memory Access)

When large amount of data are to be transferred, the most efficient technique is known as DMA.

DMA is the technique for data transfer blw internal & external storage or processor and peripheral without intervention of CPU

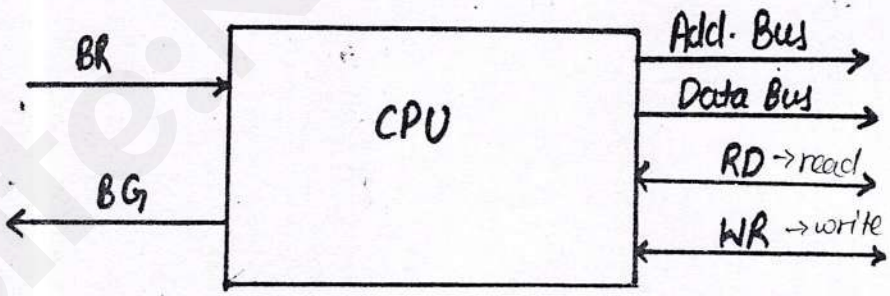


fig: BR and BG signal in DMA

Bus Request (BR) input is used by DMA controller to request the processor to give up the bus control.

Bus Grant (BG) is an output from CPU to inform external DMA controller the status of buses.

In DMA burst transfer, a block sequence consisting of no. of memory words is transferred in a continuous while the DMA controller is the master of mumory.

In cycle stealing method, which allow the DMA controller to transfer one data word at a time after that it must return control of buses to the CPU. Here, DMA controller steals "one mumory cycle from processor."

DMA Controller:- DMA controller requires a circuit of an interface to communicate with the processor and input/output device.

It consists of an address register and word count register. The address register and address lines are used for direct communication with memory

A word count register specifies the no. of words that must be transferred. It is decremented by 1 after each word is transferred

The address register contains an address of desired location in memory. The address bits go through bus buffer into the address buses.

Control register specifies the mode of transfer.

The registers in DMA are selected by the processor through address bus by enabling DS (DMA select) and RS (Register select) inputs.

The RD (Read) and WR (Write) lines are bidirectional.

When BG=0, the processor can communicate with the DMA register through the data bus to read from or write to the DMA registers.

when BG=1, the processor gives up the control over the buses and DMA can communicate directly with the memory.

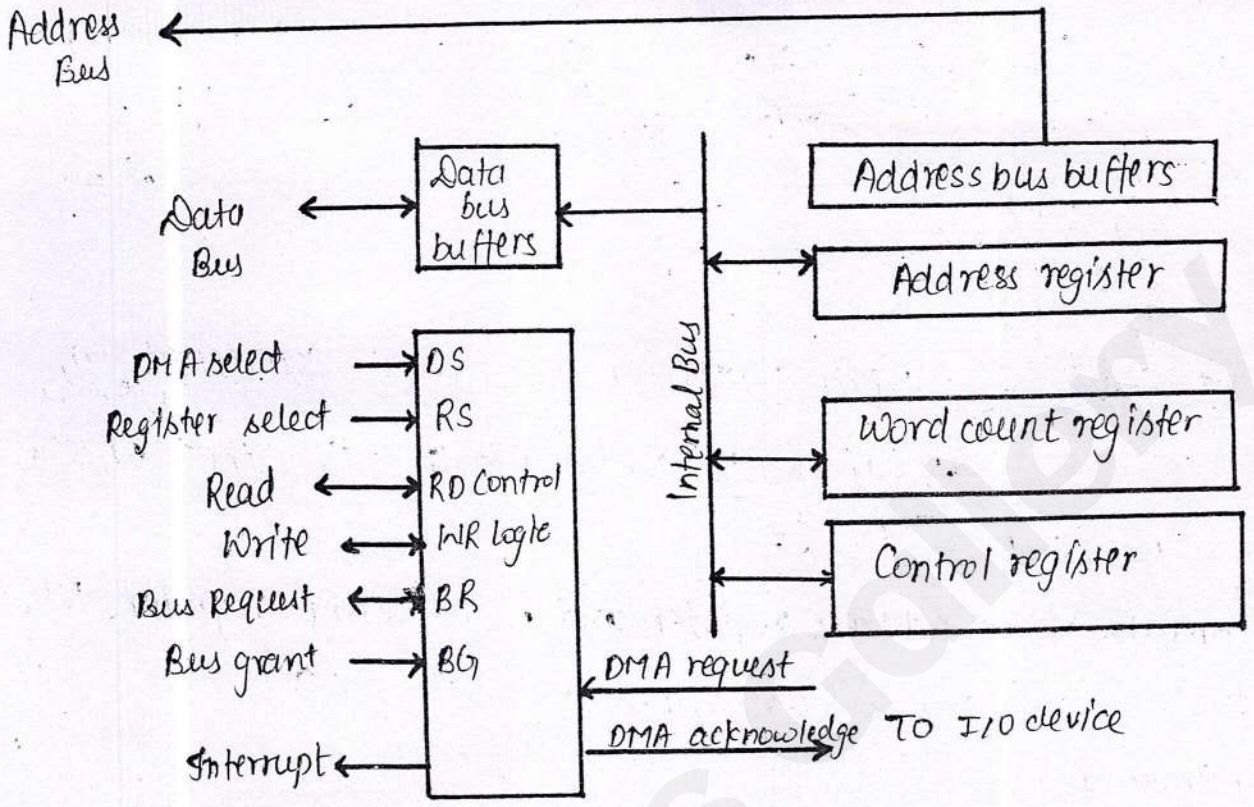
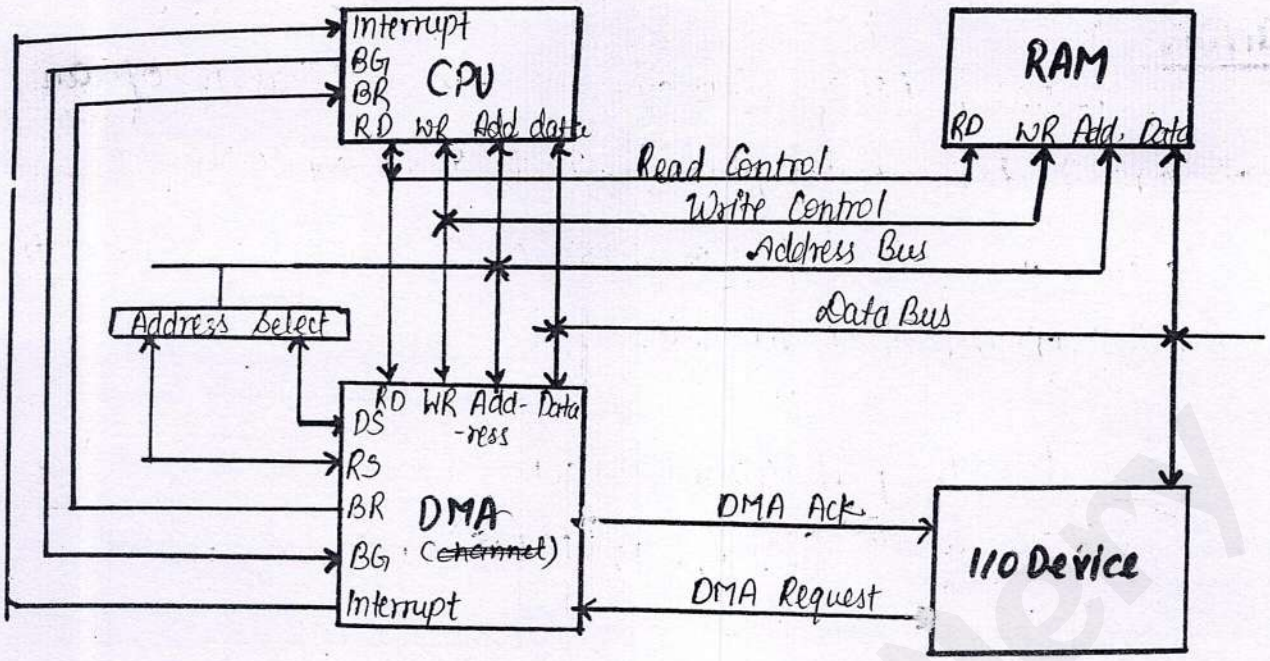


Fig. Block diagram of DMA controller

DMA Transfer :- When the peripheral devices send a DMA request to DMA controller, DMA controller activates BR line and the processor responds with the BG line informing that the buses are disabled or activated then, DMA puts the current value of its address register into the address bus, initiates RD or WR lines and sends a DMA acknowledgement to the peripheral devices. On receiving a DMA acknowledgement, the I/O device puts a word in the data bus (for Write) or receives a word from data bus (for Read). Thus, the peripheral unit can communicate with the memory without processor's intervention.



Input/Output Port :- An input/output interface consists of circuit required to connect an I/O device to the buses.

On the one side of Interface, the bus signals for address, data and control are required. On the other side, there is a data path with its associated controls to transfer data b/w interface and the I/O device. This is called a port

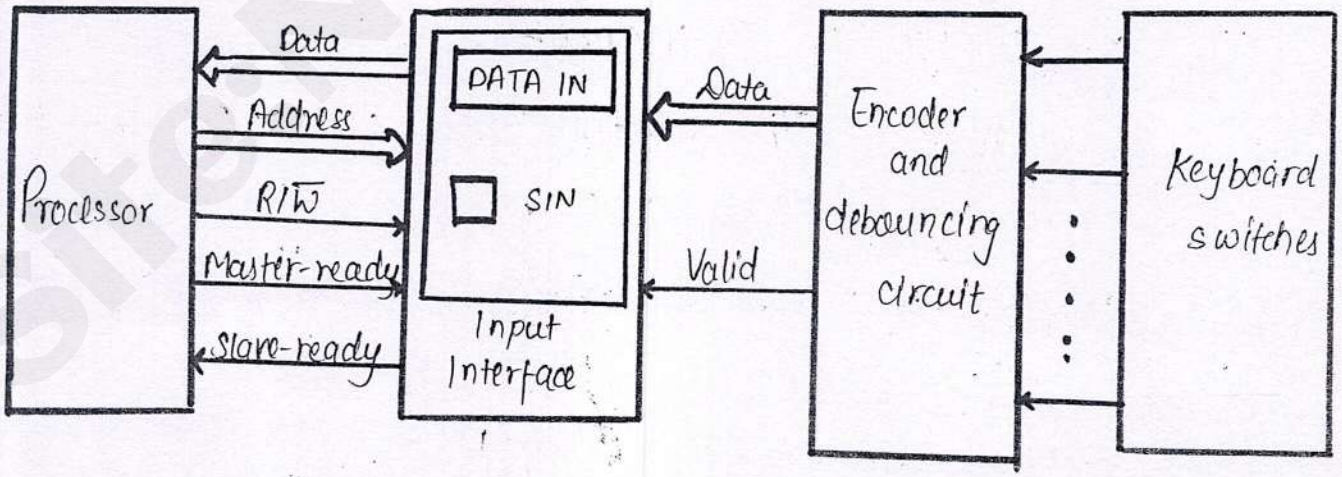


Fig (a) Hardware components for connecting a keyboard to a processor

The port can be classified as :- serial port and parallel port.

Serial Port:- A serial port is used to transmit / receive data serially i.e. one bit at a time. A key feature of an interface circuit for a serial port is that it is capable of communicating in a bit serial manner on the device side and in a bit parallel on the processor side.

Parallel Port:- Parallel port is used to send or receive data having group of bits (eg. 8 bit or 16 bit) at a time.

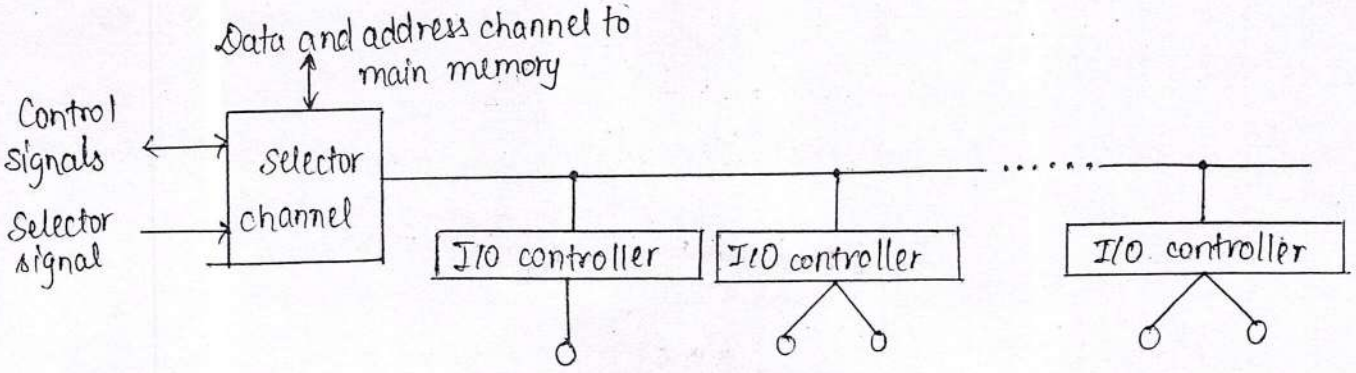
:- Input / Output Channels

An I/O channel has a special purpose processor. This processor has an ability to execute I/O instruction and it can have complete control over the I/O operation.

The I/O instructions are stored in the main memory. When I/O transfer is required, the CPU initiates an I/O transfer by instructing I/O channel to execute I/O program stored in the main memory.

There are two types of I/O channel :-

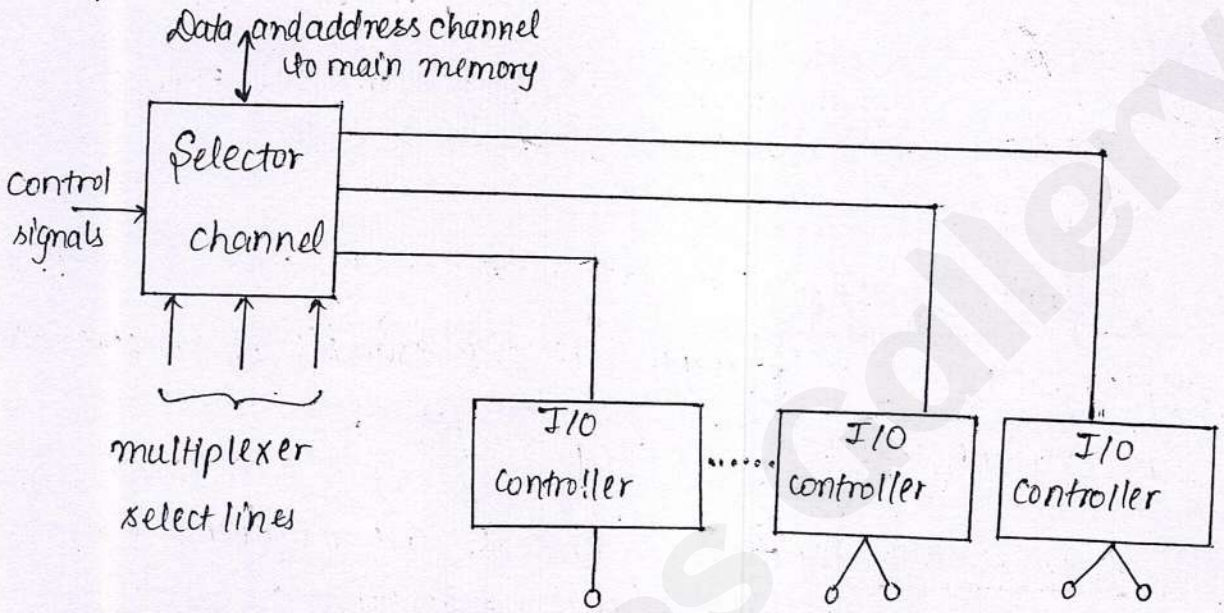
1. **Selector Channel:-** The selector channel controls multiple high speed devices. It selects only one device at a time and does data transfer. Each device is handled by a controller or I/O module.



2. **Multiplexer channel**:- A multiplexer channel can handle with I/O multiple devices at the same time.

A multiplexer channel does the time multiplexing and communicates with I/O controller in an alloted time slot.

In multiplexer channel, different devices can have different speeds. However, this is the best technique for low speed devices.



Difference b/w Asynchronous & Synchronous Serial Communication

S.No.	Asynchronous serial comm.n	Synchronous serial commn
1.	Transmitters and receivers are not synchronized by clock.	Transmitter and receivers are synchronized by clock.
2.	Bits of data are transmitted at constant rate.	Data bits are transmitted with synchronization of clock.
3.	Character may arrive at any rate at receiver.	Character is received at constant rate
4.	Data transfer is character oriented.	Data transfer takes place in blocks.
5.	Used in low-speed transmissions at about speed less than 20kbits/sec	Used in high-speed transmissions.

start and stop bits are required to establish communication of each character.	start and stop bits are not required to establish communication of each character; however, synchronization bits are required to transfer the data block.

Exceptions:- An interrupt is an except that causes the execution of one program to be suspended and the execution of another program is started.

The term 'exception' often used to refer any event that causes an interruption. Hence, I/O interrupts are one example of an execution

Following are the different kinds of exceptions:-

1. **Recovery from errors:-** Various techniques are available which ensures the proper working of computer hardware. For eg:- An error checking code, is included by many computes in the main memory which allows detecting errors in stored data. In case of any error control hardware detects it and informs the processor.

   The processor may also interrupt a program if it detects an error or an unusual condition while executing any instruction.

   Eg:- An attempt to divide by 0.

2. **Debugging:-** * Another important exception is used as a support in debbing programs. System software usually includes a program called "debugger" which helps the programmer to find errors in a program.

   * The debugger uses exceptions to provide two important facilities called trace and break points

3. Privilege exceptions:- To protect the operating system from being corrupted by the user program, certain instructions can be executed only while the processor is in the supervisor mode. These instructions are called privilege instructions.

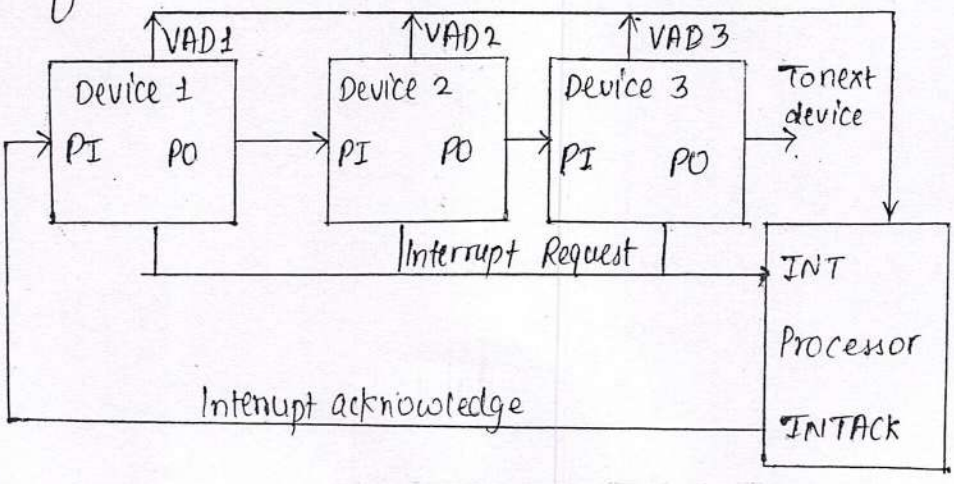
Priority Interrupt

A priority interrupt is a system that establishes a priority over the various sources to determine which condition is to be served first when two or more requests arrive at the same time.
A priority system is a combination of hardware and software techniques.
Priority interrupt can be resolved by Daisy Chaining Approach.

Daisy Chaining approach:- This is a serial connection method. The device having highest priority is placed first and followed by lowest priority. When more than one interrupt request are generated at a time, the processor responds by interrupt acknowledgement line.

This signal is received by device 1 at its PI (Priority In). The acknowledgement signal is passed to the next device through PO (priority out) only if device 1 is not requesting for interrupt.

If device 1 has a pending interrupt, it blocks the acknowledgement signal from the next device by placing 0 in PO. (PO=0)



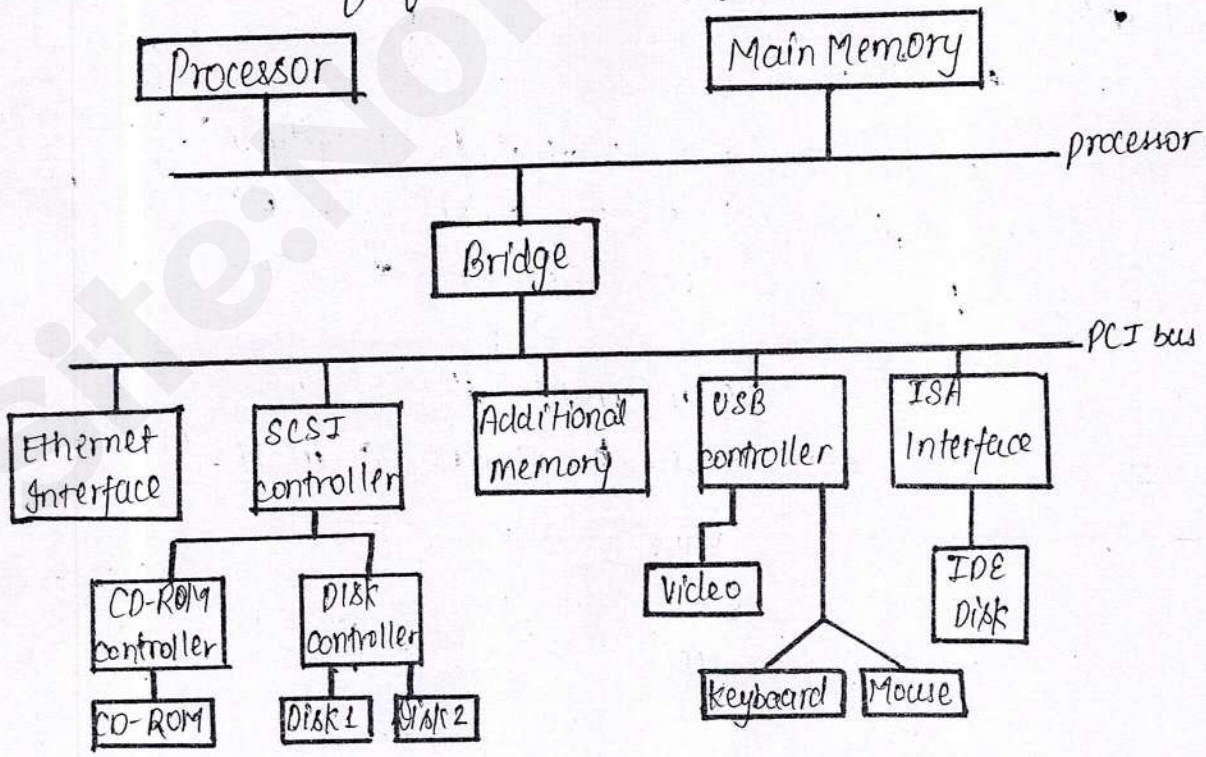
Standard Communication Interface:- There are many alternatives for bus design. A single bus is not suitable for every communication in a computer system.

In personal computer processor is mounted on the motherboard. Processor require high speed connection. The motherboard provides another bus that support various devices.

The two buses are connected by a circuit called bridge

There are various number of standards developed for bus expansion. Ex- PCI (Peripheral Component Interconnect), SCSI (Small computer system interface) and USB (Universal serial Bus).

PCI standard is used for expansion bus on the motherboard. SCSI bus is high speed parallel bus intended for devices which need the large amount of data transfer. USB is used for serial transmission to fulfill the needs of device like keyboard, mouse etc.



(Standard Communication Interface)

Difference blw I/O mapped and Memory mapped I/O of 8086

Sr. No.	I/O Mapped I/O	Memory Mapped I/O
1.	I/O device is treated as an I/O device and hence given an I/O address.	I/O device is treated like a memory device and hence given a memory address.
2.	I/O device has an 8 or 16-bit I/O address.	I/O device has a 20 bit memory address.
3.	I/O device is given IOR# and IOW# control signals.	I/O device in given MEMR# and MEMW# control signals.
4.	Decoding is easier due to lesser address lines.	Decoding is more complex due to more address lines.
5.	Decoding is cheaper.	Decoding is more expensive.
6.	Works faster due to less delays.	More gates add more delays hence, slower.
7.	Allows max $2^{16} = 65536$ I/O devices.	Allows many more I/O devices as I/O addresses are now 20 bits
8.	I/O devices can only be accessed by IN and OUT instruction.	I/O devices can now be accessed using any memory instruction
9.	ONLY AL/AH/AX registers can be used to transfer data with the I/O device.	Any register can be used to transfer data with the I/O device.