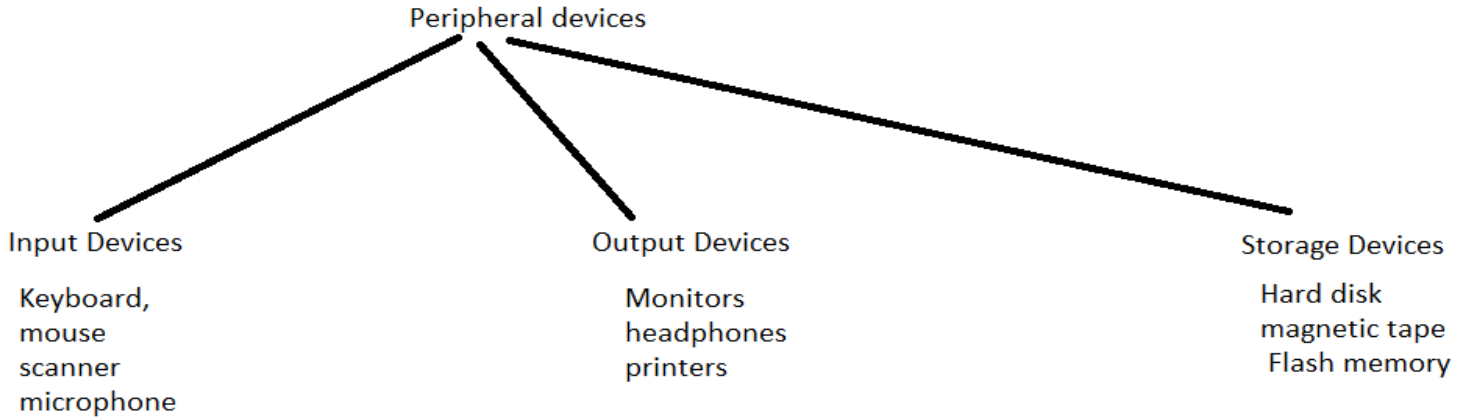
I/P INTERFACE
MODULE-II

Input-Output Organization:

Peripheral Devices: A Peripheral Device is defined as the device which provides input/output functions for a computer.
It is generally classified into 3 basic categories:



- Peripherals that provide auxiliary storage are magnetic tape and magnetic disk.
- Input and output devices communicate alphanumeric information by using ASCII 7bit code.

Advantage of Peripherals Devices:

- It is helpful for taking input very easily.
- It is also provided a specific output.
- It has a storage device for storing information or data
- It also improves the efficiency of the system.

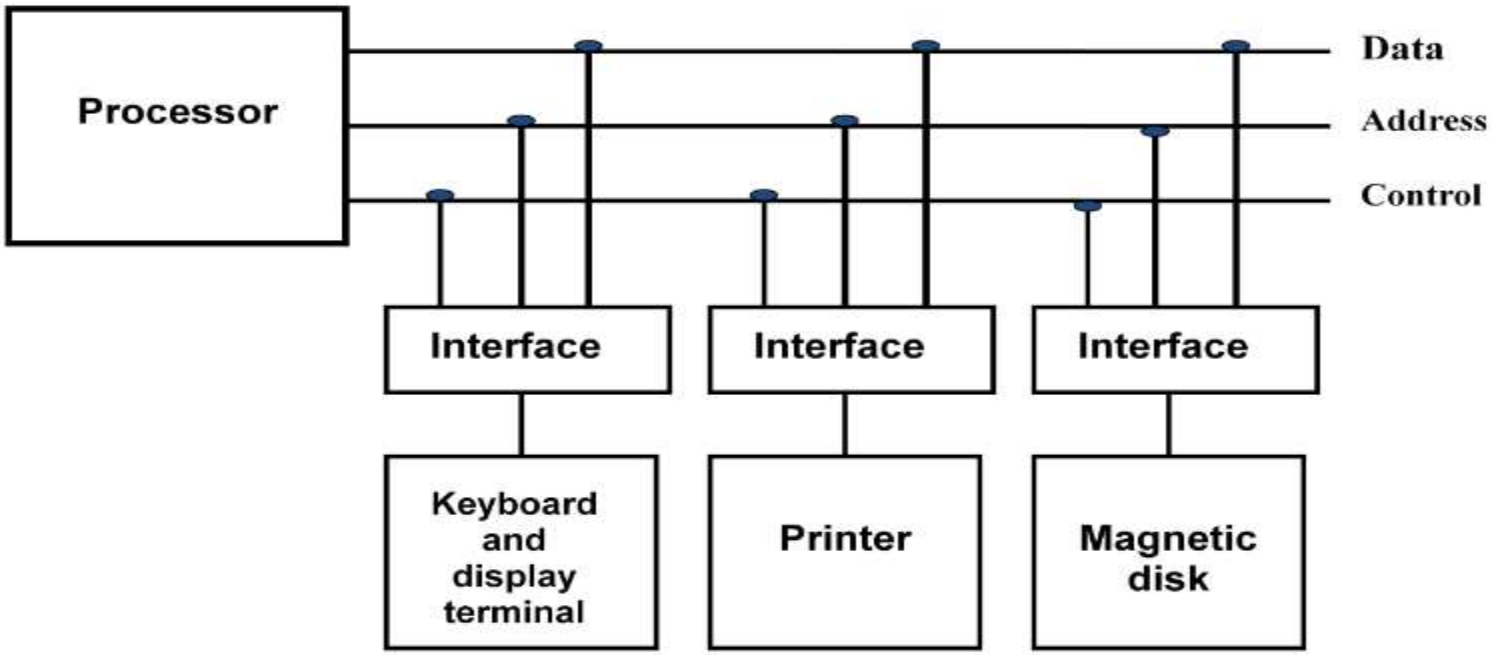
Input-Output Interface:

- ✓ I/O interface provides a method for transferring information between internal storage and external I/O devices.

- ✓ The purpose of interfacing are as follows:
  - Peripherals are electromechanical & electromagnetic devices and are interacting with electronics devices (CPU). Therefore, a conversion of signal values may be required.
  - Data transfer rate of peripherals is usually slower than transfer rate of CPU.
  - Data codes and formats in peripherals differ from word format in CPU and memory.
  - Operating modes of peripherals are different from each other and each one must be controlled without disturbing other peripherals connected to the CPU.
- ✓ To resolve these differences, computer system includes Interface units between CPU and peripherals to supervise and synchronize all input and output transfers.

I/O Bus and Interface Modules:

- ✓ Each peripheral has its own controller that operates a particular electromechanical device.
- ✓ To communicate with a particular device, the processor places a device address on the address lines.



Connection of I/O bus to input-output devices

I/O Bus and Interface Modules:

- When the interface detects it own address, it activates path between bus lines and the device.

- At the time address is made available in address lines, the processor provides function code (I/O command) in the control lines.

Types of I/O command:

- Control command: To activate and inform what to do.
- Status command: To test various status conditions.
- Data Output data: It causes the transfer of data from bus into one of its registers.
- Data Input Command: Interface receives data from peripheral and places them on buffer register where it is put into data lines.

I/O versus Memory Bus:

- In addition to communicating to I/O, processor must also communicate with memory unit.

- There are three ways that computer buses can be used to communicate with memory and I/O:

  - Use two separate buses, one for memory and one for I/O. (IOP)
  - Use one common bus for both memory and I/O but have **separate control lines** for each. (Isolated I/O)
  - Use one common bus for memory and I/O with **common control lines.** (Memory Mapped I/O)

Isolated I/O:

- The distinction between memory and I/O transfer is made through separate read and control lines.
- I/O read and I/O write are enabled during I/O transfer and Memory read/write are enabled during memory transfer.
- In the isolated I/O configuration, CPU have distinct input and output instructions where each of it will be associated with address of the interface register.
- When the CPU fetches and decodes the I/O instruction, it places the address associated with the instruction on the common address lines and enables I/O read or I/O write control line.
- When the CPU fetches and decodes the Memory instruction, it places the address associated with the instruction on the common address lines and enables Memory read or Memory write control line.
- The isolated I/O method isolates memory and I/O addresses.

Memory Mapped I/O:

- Here the same address space is used for both memory and I/O. The computer treats interface register as a part of memory system.

- The assigned address cannot be used for storing memory words, which reduces memory address range available.

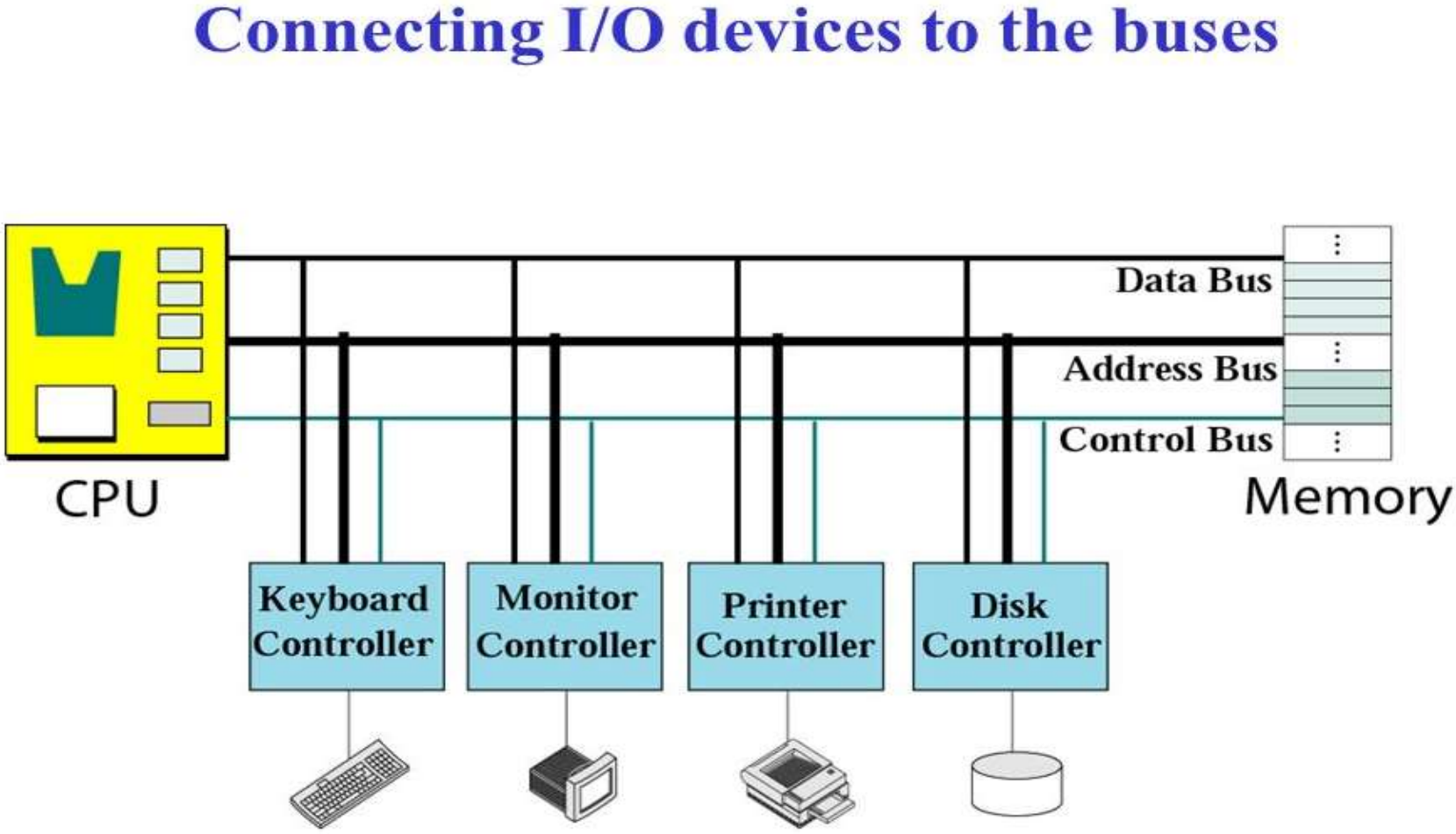
- In a memory-mapped I/O organization, there are no specific, input or output instruction.

- CPU manipulate I/O data residing in interface registers with the same instruction used to manipulate memory words.

- Load and store instruction used for reading/writing from memory and can be use for input or output data from I/O instruction.

- With memory-mapped I/O all instructions that refer to memory are also available for I/O.

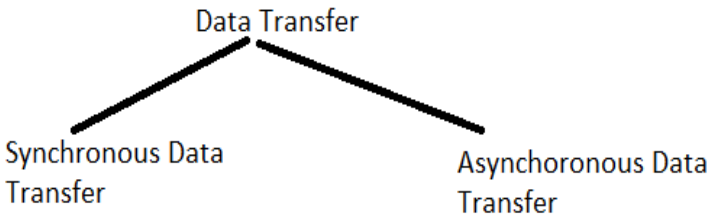
Memory Mapped I/O:

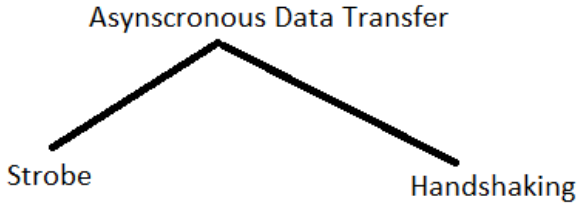


Data Transfer:

- If the registers in the interface share a common clock with the CPU registers, the transfer is **synchronous**.
- **Asynchronous** data transfer requires that control signals be transmitted between communicating units to indicate the time at which data is being transmitted.
- Asynchronous data transfer can be accomplished by **Strobe & Handshaking**.

S. No.	Synchronous transmission	Asynchronous transmission
1.	Transmitter and receivers are synchronized by clock.	Transmitter and receivers are not synchronized by clock.
2.	Data bits are transmitted with synchronization of clock.	Bits of data are transmitted at constant rate.
3.	Data transfer takes place in blocks.	Data transfer is character oriented.



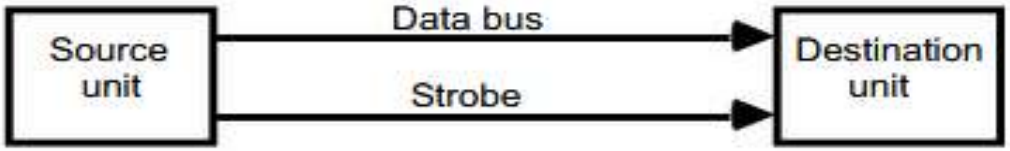


Strobe Control:

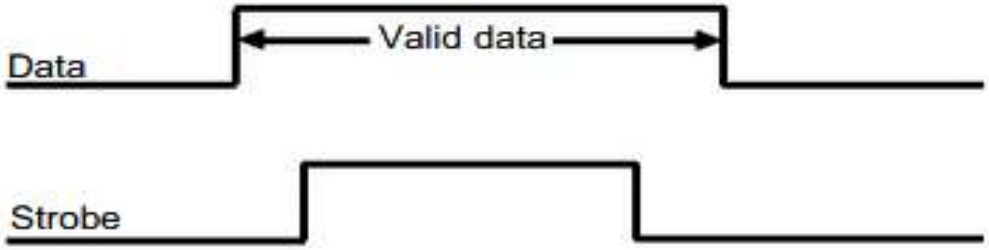
Strobe control:

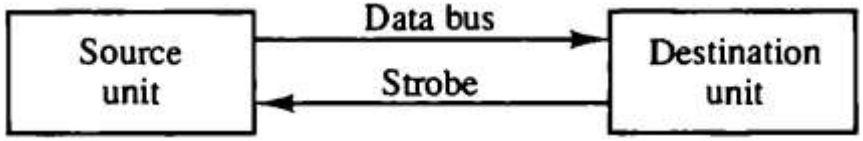
- It employs single control line.
- It can be activated either by source or destination unit.

Block Diagram



Timing Diagram





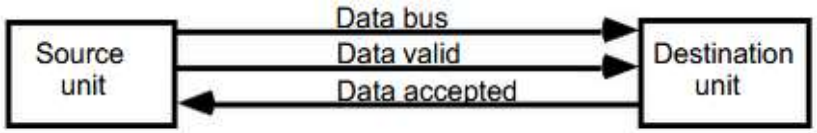
Handshaking:

- ✓ The disadvantage of the strobe method is that the source unit that initiates the transfer has no way of knowing whether the destination unit has actually received the data item that was placed in the bus.
- ✓ The handshake method solves this problem by introducing a second control signal that provides a reply to the unit that initiates the transfer.
- ✓ The basic principle of the two-wire handshaking method of data transfer is as follows.
  - ❖ One control line is in the same direction as the data flow in the bus from the source to the destination. It is used by the source unit to inform the destination unit whether there are valid data in the bus.
  - ❖ The other control line is in the other direction from the destination to the source. It is used by the destination unit to inform the source whether it can accept data.
  - ✓ The sequence of control during the transfer depends on the unit that initiates the transfer.

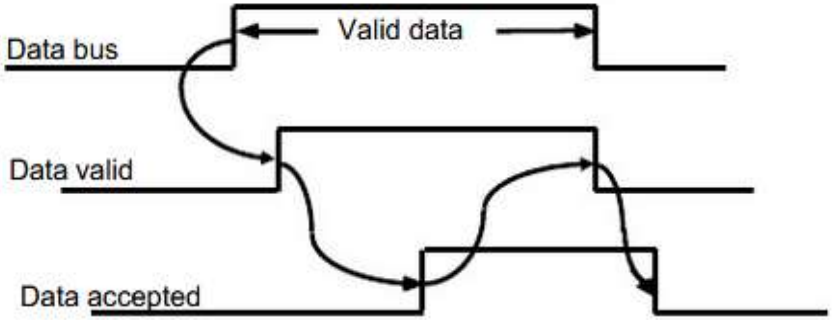
Handshaking:

The two handshaking lines are data valid, which is generated by the source unit, and data accepted, generated by the destination unit.

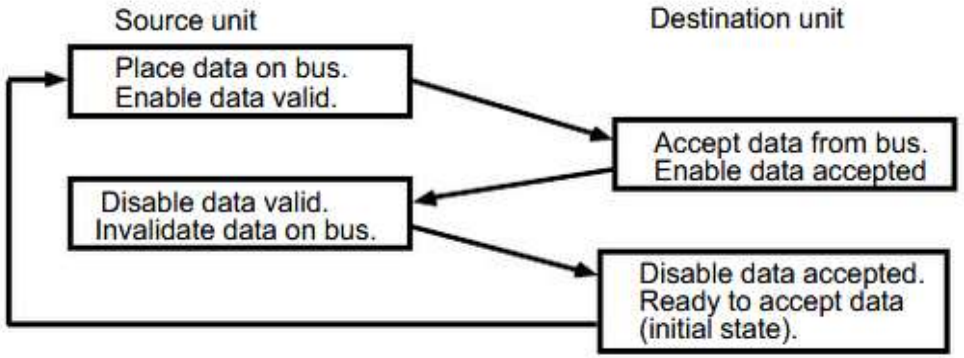
Block Diagram



Timing Diagram

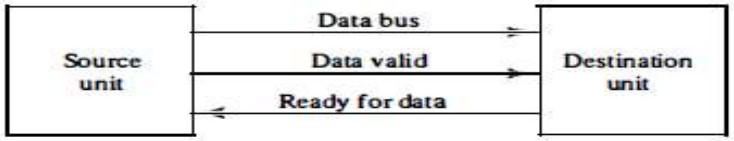


Sequence of Events

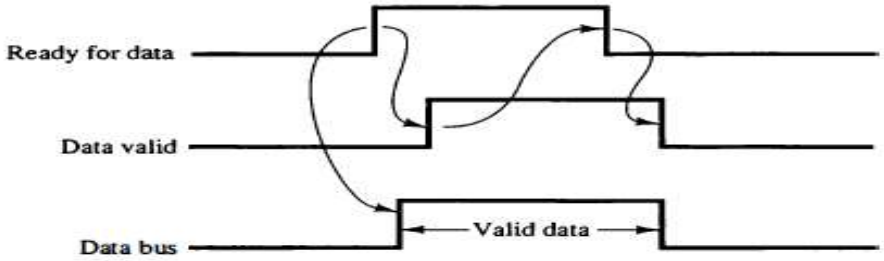


Handshaking:

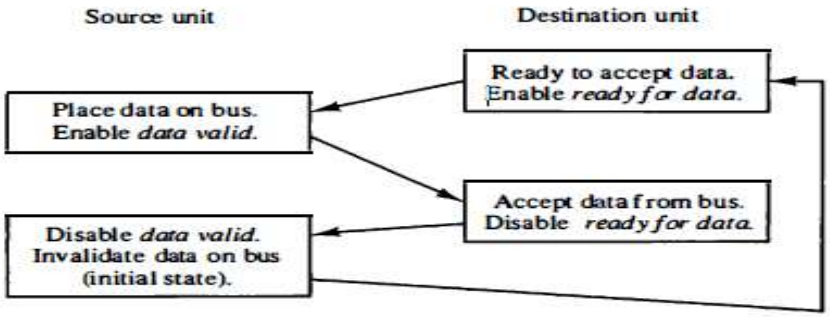
- In the destination-initiated transfer the name of the signal generated by the destination unit has been changed to ready for data to reflect its new meaning.
- The source unit in this case does not place data on the bus until after it receives the ready for data signal from the destination unit.



(a) Block diagram

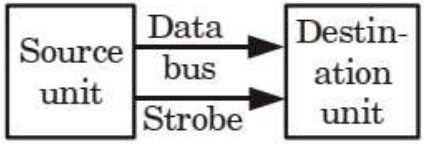
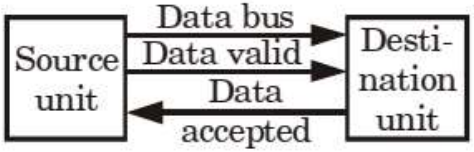
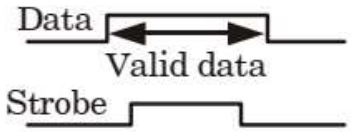
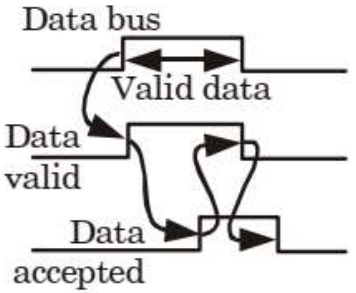


(b) Timing diagram



(c) Sequence of events

Difference Between Strobe & Handshaking Control:

S. No.	Parameter	Strobe control	Handshaking
1.	Control line	It employs a single control line to time each transfer.	It employs more than single control line to time each transfer.
2.	Acknowledgement	Reply message is not present.	Reply message is present.
3.	Block diagram		
4.	Timing diagram		

Modes of Transfer:

Data transfer between the CPU and the I/O devices may be handled in variety of modes. Some modes use the CPU as an intermediate path and others transfer the data directly to and from the memory unit.

Data transfer to and from peripherals may be handled in three ways:

- Programmed I/O
- Interrupt-initiated I/O
- Direct Memory Access (DMA)

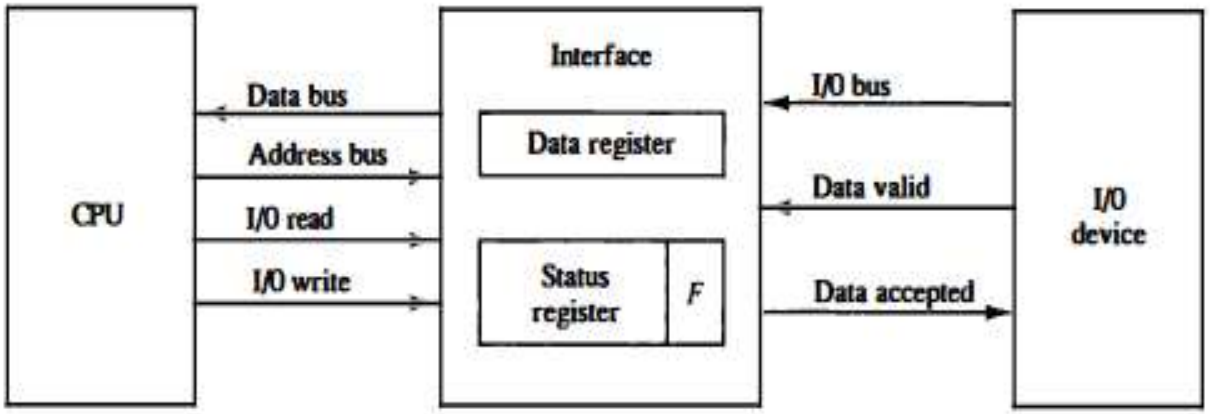
Programmed I/O:

- Programmed I/O operations are the result of I/O instructions. Each data item transfer is initiate by an instruction in the program.

- Usually the transfer is to and from a CPU register and peripheral. Other instructions are needed to transfer data between memory and CPU.

- Once a data transfer is initiated, the CPU is required to monitor the interface to see when a transfer can again be made.

Figure 11-10 Data transfer from I/O device to CPU.



F = Flag bit

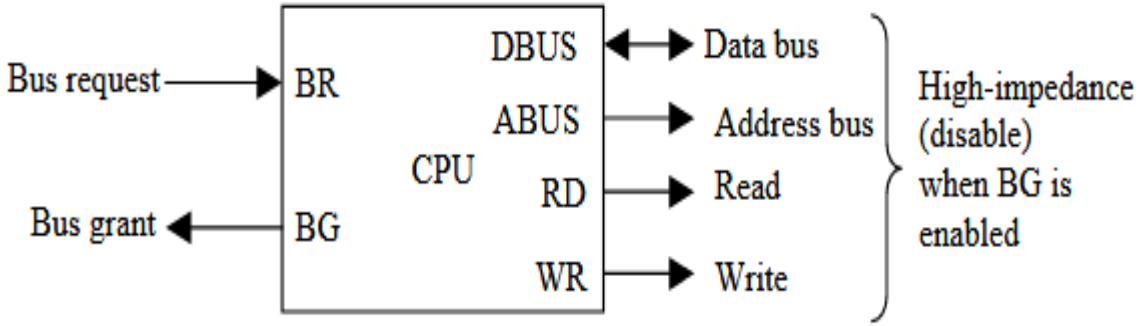
Interrupt Initiated I/O:

- In the programmed I/O CPU stays in program loop until the I/O indicates that it is ready for data transfer.
- This is time consuming process since it makes CPU busy needlessly.
- This can be avoided by using an interrupt facility.
- When the interface determines that device is ready for data transfer, it generates an interrupt request.
- Upon detecting external interrupt signal, the CPU momentarily stops the task it is processing.
- It then branches to fulfill the I/O request and return to the original task.
- Based on the concept of Priority Interrupt.

Direct Memory Access (DMA):

- The transfer of data between a fast storage device such as magnetic disk and memory often limited to the speed of CPU.
- Removing the CPU and letting the peripheral device manage the memory bus directly improve speed of transfer.
- Such transfer technique is called Direct Memory Access (DMA).
- A DMA controller takes over the buses to manage the transfer directly between I/O device and memory.
- During DMA transfer, the CPU is idle and has no control over memory buses.
- By using Bus Request (BR) and Bus Grant (BG) the buses are released to DMA controller.

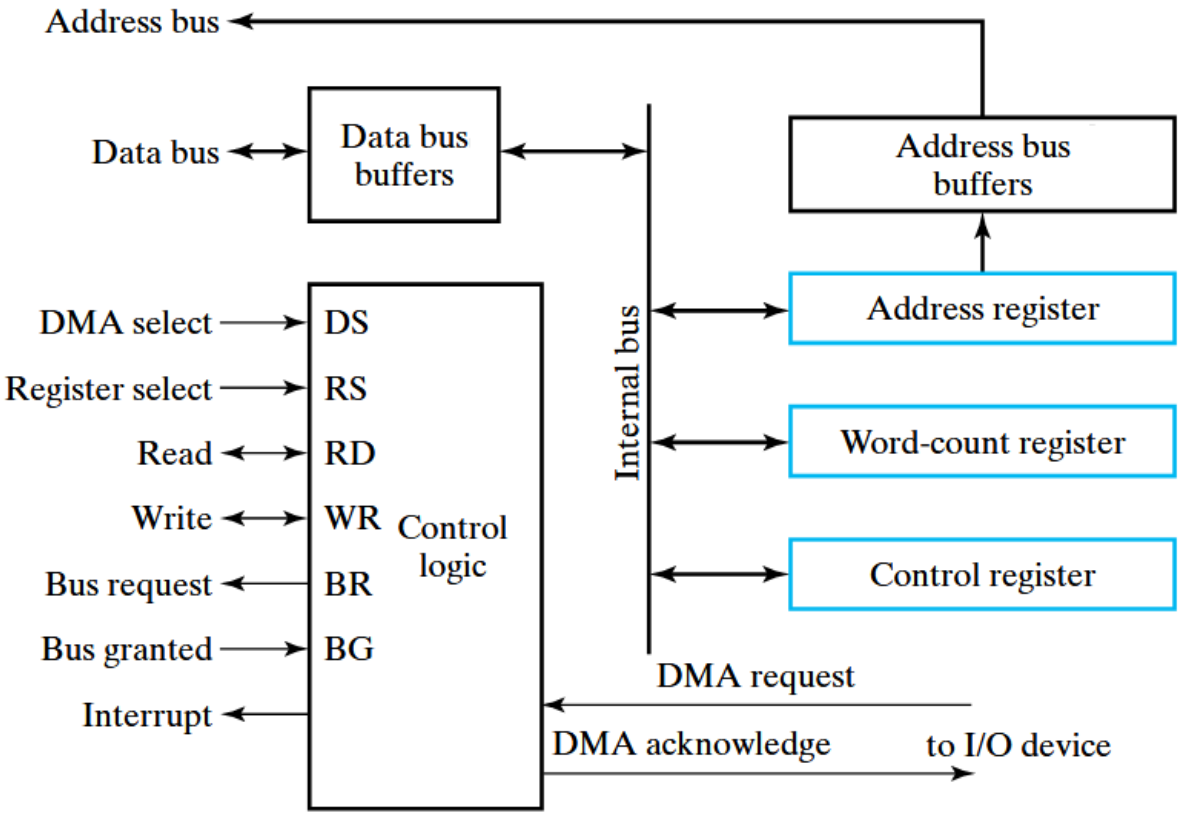
Figure 6.13 CPU bus signals for DMA transfer.



Direct Memory Access (DMA):

Data transfer ways:

- **Burst Transfer:** Here number of words are transferred in a block. Example: Magnetic disk.
- **Cycle stealing:** Allows the DMA controller to transfer one data word at a time after it must return the control of buses to CPU.

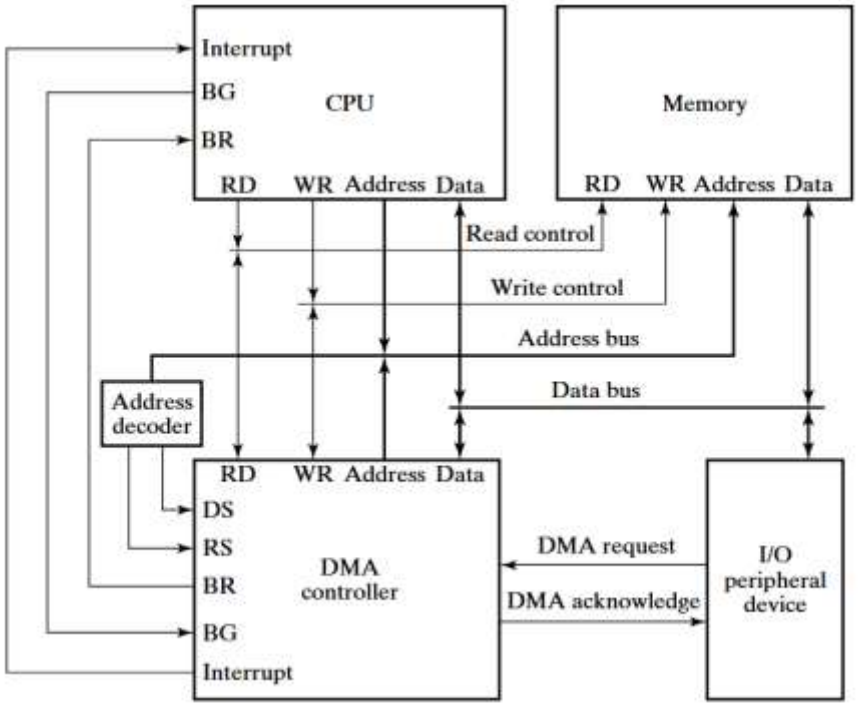


Initialization of DMA:

The CPU initializes the DMA by sending the following information through the data bus.

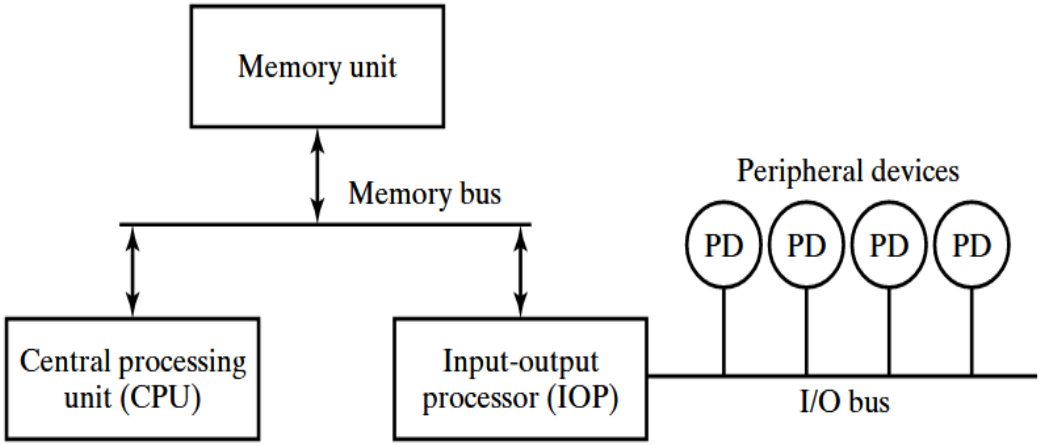
- The starting address of the memory block where data are available (for read) or where data are to be stored (for write).
- The word count, which is the number of words in the memory block.
- Control to specify the mode of transfer such as read or write.
- A control to start the DMA transfer.

DMA Transfer in a computer system:



Input-Output Transfer (IOP):

- An IOP takes care of input and output tasks.
- IOP is similar to CPU except IOP can fetch and execute an I/O instruction.
- CPU is usually assigned task of initiating the I/O program and thus IOP.
- The CPU is assigned the task of initiating all operations, but I/O instructions are executed in IOP.
- When an I/O operation is required, the CPU informs the IOP where to find the I/O program and then leaves the transfer details to the IOP.
- CPU instructions also test I/O status conditions needed for making decisions on various I/O activities.
- IOP is also responsible of taking care of data synchronization, formats etc between CPU and I/O devices.
- In most computers CPU is master and IOP is slave.
- The IOP typically asks for CPU attention by means of Interrupt.



Types of Interrupt:

- An **interrupt** is a signal to the processor emitted by hardware or software indicating an event that needs immediate attention.
- It alerts the processor to a high priority process requiring interruption of the current working process.
- In I/O devices one of the bus control lines is dedicated for this purpose and is called the ***Interrupt Service Routine*** *(ISR)*.

TYPES OF INTERRUPTS

There are many type of interrupts but basic type of interrupts are –

1. Hardware and software interrupts
2. Vectored and Non- vectored Interrupts
3. Mask able and non-maskable interrupts

1. Hardware and software interrupts

Hardware Interrupts: If the signal for the processor is generated from external device or hardware is called hardware interrupts. There are 5 Hardware Interrupts in processor. They are

1. INTR (Lowest Priority)
2. RST 7.5
3. RST 6.5
4. RST 5.5
5. TRAP (Highest Priority)

Software Interrupts are those which are inserted in between the program which means these are mnemonics of microprocessor.

There are 8 software interrupts: RST 0, RST 1, RST 2, RST 3, RST 4, RST 5, RST 6, RST 7.

Types of Interrupt:

2. Vectored and Non- vectored Interrupts

Vectored Interrupts are those which have fixed vector address (starting address of sub-routine) and after executing these, program control is transferred to that address. When an interrupt is occurred and program control automatically branches the program execution to a specific address.

Non-Vectored Interrupts (Scalar Interrupt) are those in which vector address is not predefined. Interrupts that have a variable address. When an interrupt device have to provide an address from where the execution of program will begin. **INTR** is the only non-vectored interrupt

3. Mask able and Non-maskable interrupts

Maskable Interrupt: The hardware interrupt that can be ignored or delayed for some time if the processor is executing a program with higher priority are termed as maskable interrupts**.** These interrupts are either edge-triggered or level-triggered, so they can be disabled. INTR, RST 7.5, RST 6.5, RST 5.5 are maskable interrupts.

Non Maskable Interrupt: The hardware interrupts that can neither be ignored nor delayed and must immediately be serviced by the processor are termed as non-maskable interrupts**.** It consists of both level as well as edge triggering and is used in critical power failure conditions. **TRAP** is a non-maskable interrupt.

Types of Interrupt:

Major types of Interrupt are:

- **External Interrupt:** It comes from I/O devices, from timing device, or from any other external source.

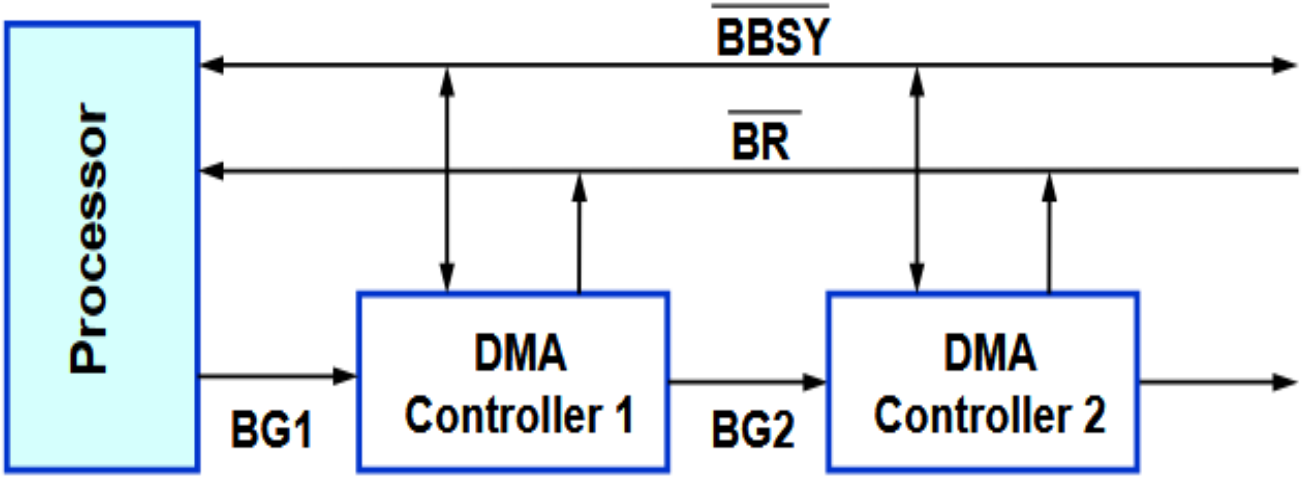
- **Internal Interrupt:** It includes register overflow, invalid operation code, stack overflow etc.

- **Software Interrupt or Hardware Interrupt**: External and Internal interrupts are initiated from signals that occur in the hardware of the CPU. A software interrupt is initiated by executing an instruction.

- **Priority Interrupt**: In case several sources will request service simultaneously, in this case system must decide which device to service first. For ex: Polling, Daisy Chain.

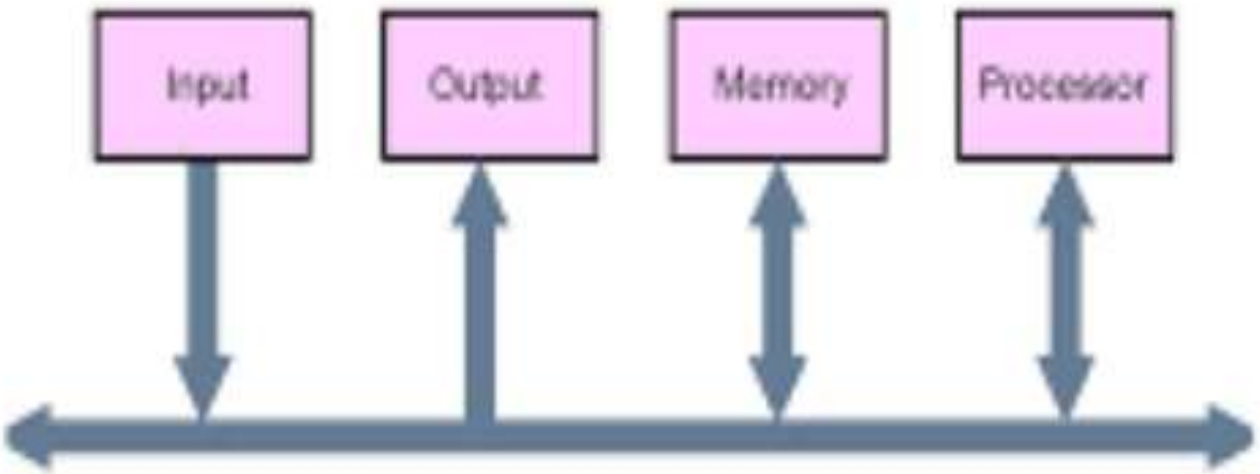
Bus Arbitration:

- The device that is allowed to initiate data transfers on the bus at any given time is called Bus master.
- Bus arbitration is the process by which the next device becomes Bus master and will do the data transfer.
- Two approaches: Centralized Arbitration and Distributed Arbitration.



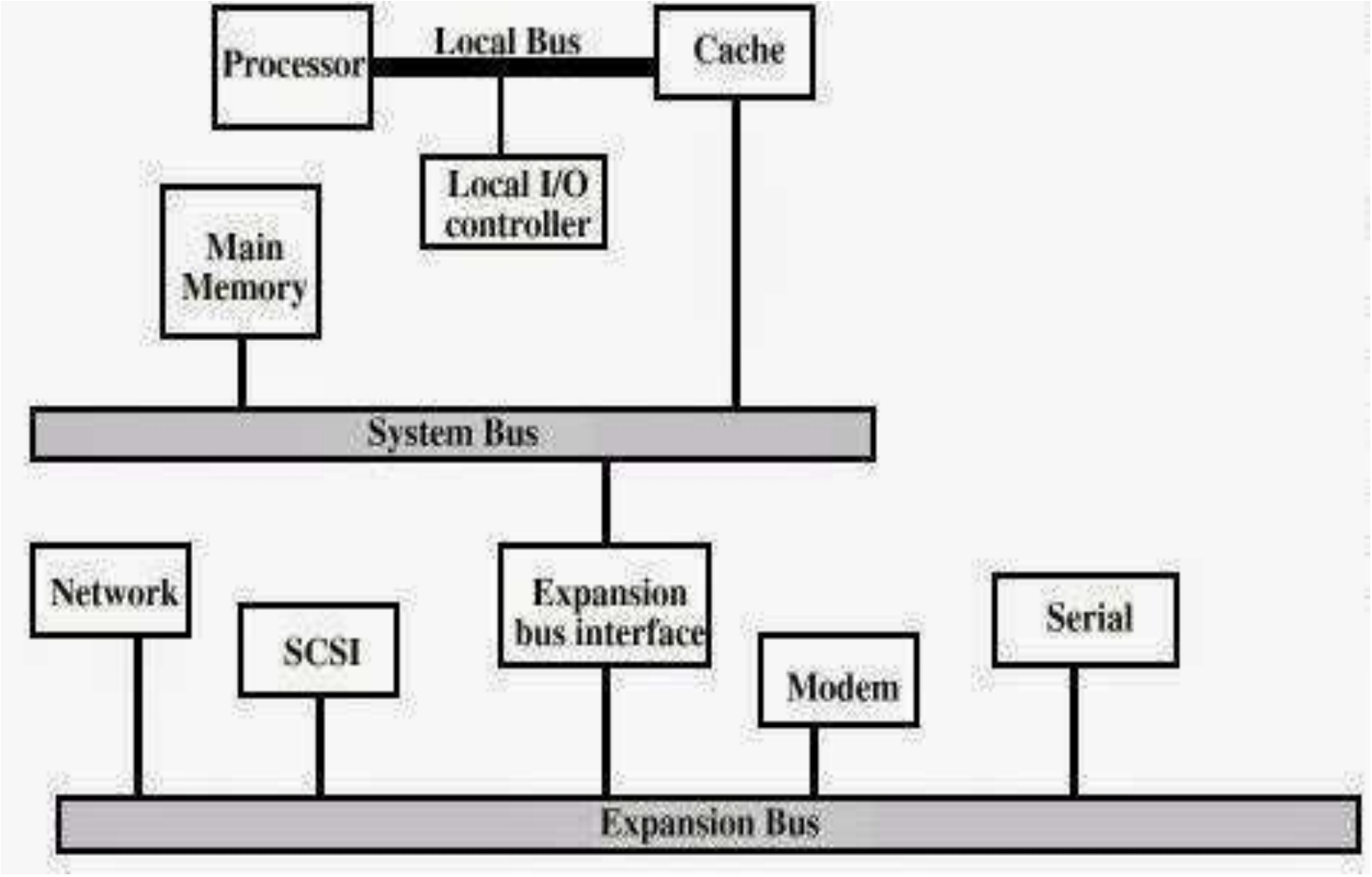
Buses:

- **Data Bus:** Bi-directional and transfers data.
- **Address Bus:** Uni-directional and sends the address.
- **Control Bus:** R/W, BR,BG etc.
- **Bus Structure**: Single bus



Buses:

- **Bus Structure:** Multi bus



THANK YOU