Unit 4
Input and Output System

I/O module

- **I/O module**

is the key element for computer system. I/O modules control the peripheral devices. Thus I/O module is required which has two major functions.

- Interface to the processor and memory via the system bus or *central switch*
- Interface to one or two peripheral devices by tailored data links

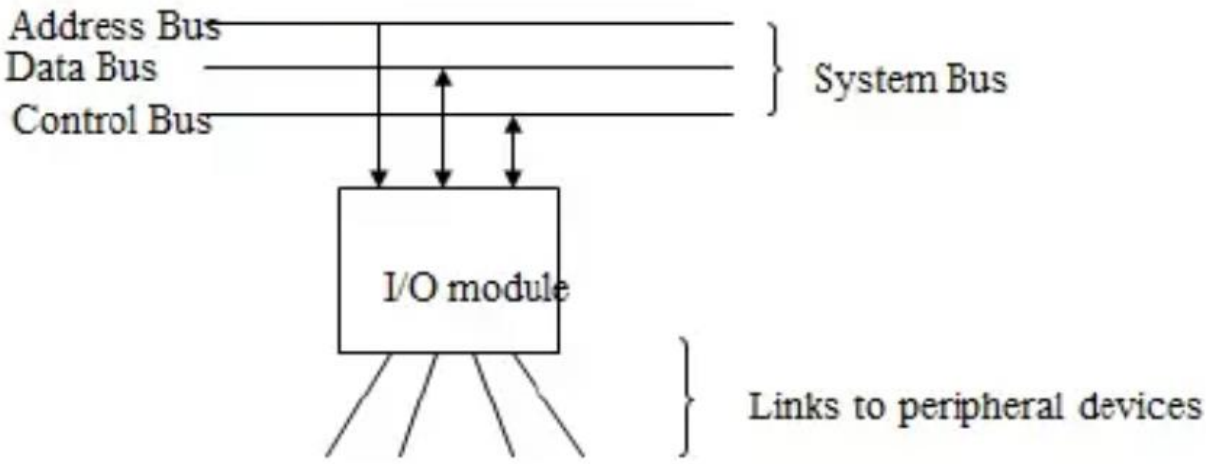


Fig. Generic model of an I/O module

External Devices

It is classified in three categories:

1. Human readable:
2. Machine readable:
3. Communication:

Block Diagram of an External Devices

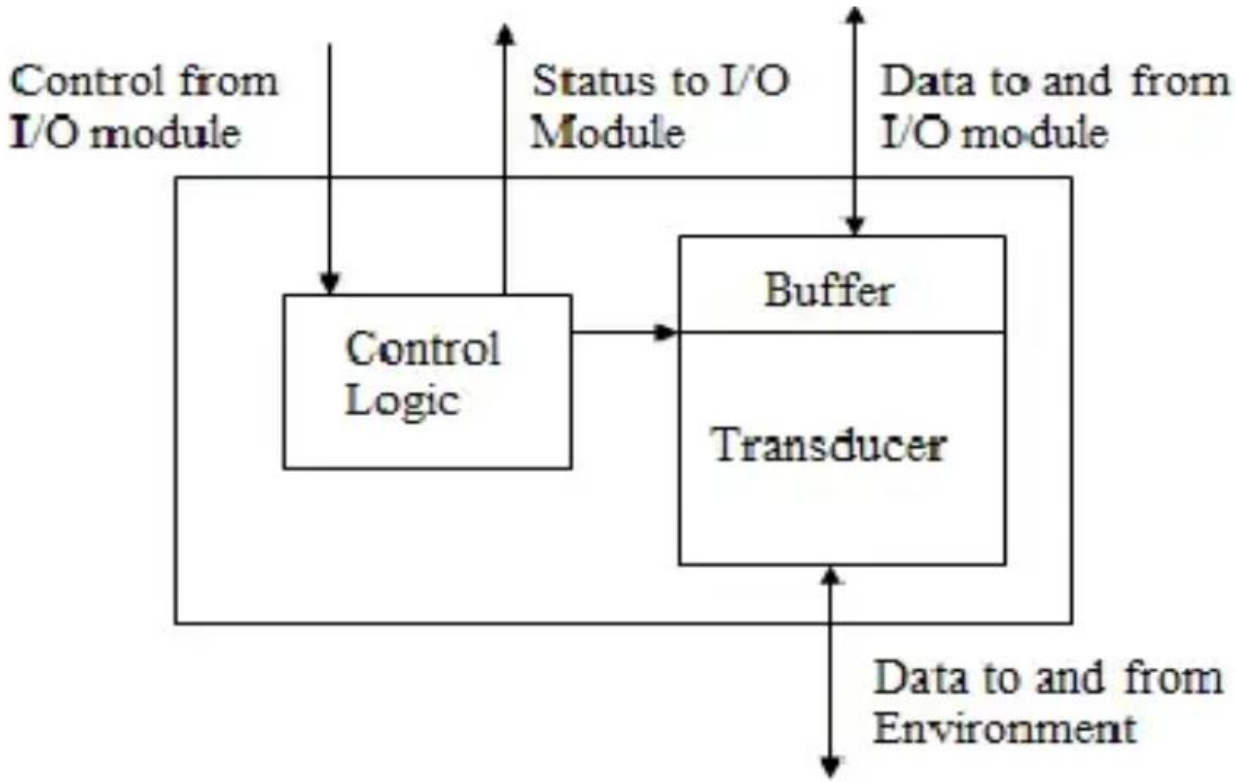
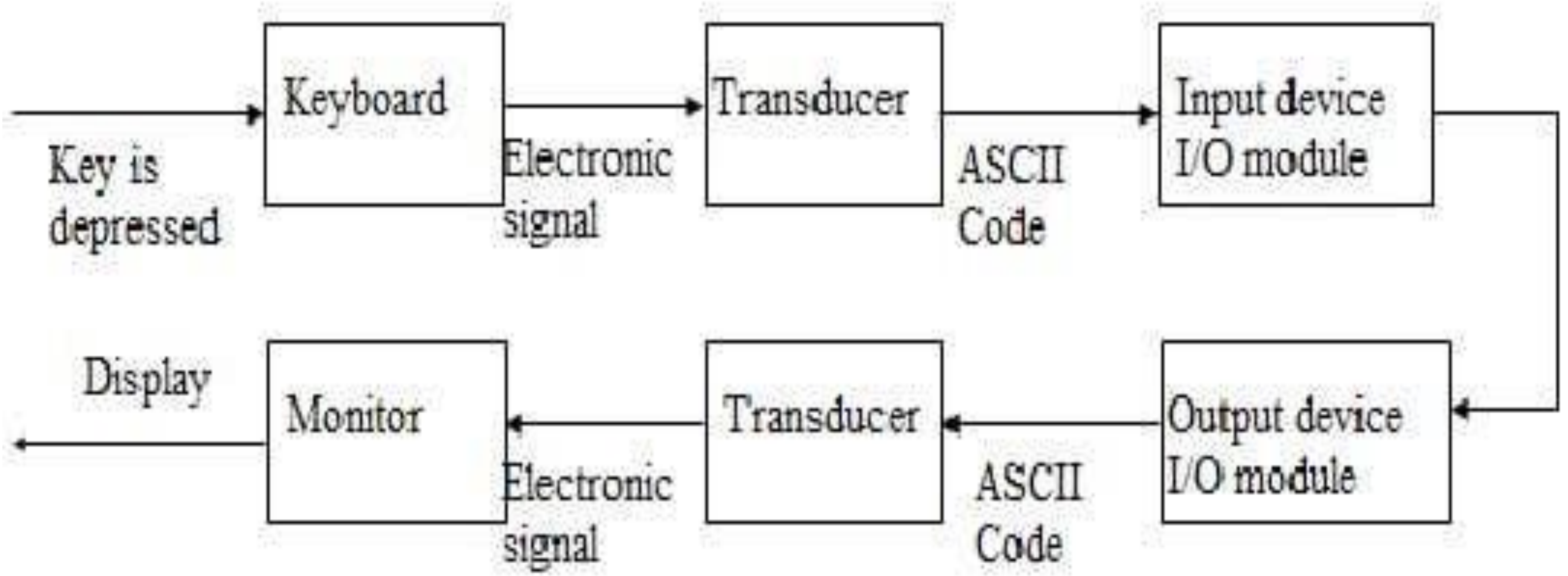


Fig: Block Diagram of an External Device

Keyboard/Monitor:

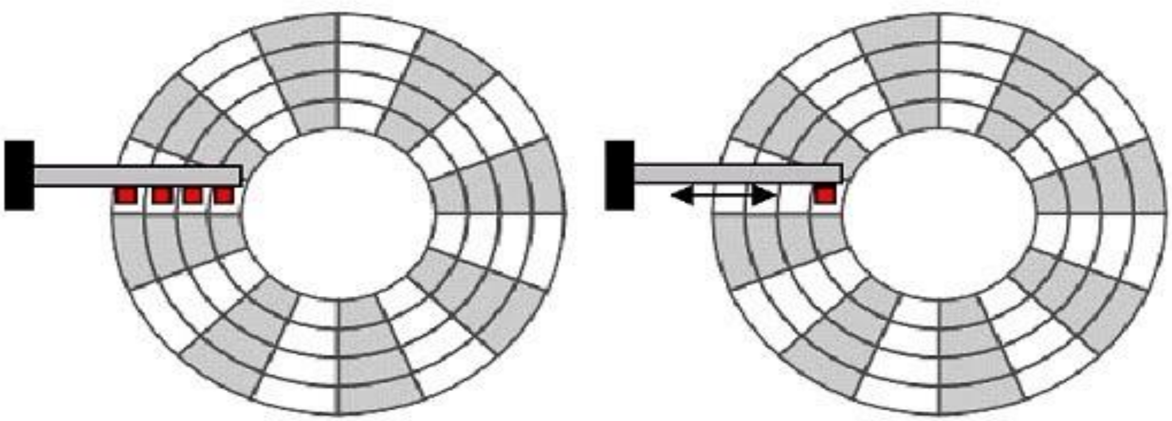


Disk Drives:

Two types of Disk

1. Fixed head: Fixed-head disk s were magnetic disks which had a separate head, fixed in position, for each track.

2. Movable head: A type of disk drive in which read/write heads are moved over the surface of the disk, toward and away from the center, so that they are correctly positioned to read or write the desired information.



(a) Fixed head (b) Movable head

Functions of I/O Module

The detailed functions of I/O modules are:

1. **Control & Timing**
2. **Processor Communication**
3. **Device Communication**
4. **Data Buffering**
5. **Error Detection**

Control & Timing:

I/O module includes control and timing to coordinate the flow of traffic between internal resources and external devices. The control of the transfer of data from external devices to processor consists following steps:

- The processor interrogates the I/O module to check status of the attached device.
- The I/O module returns the device status.
- If the device is operational and ready to transmit, the processor requests the transfer of data by means of a command to I/O module.
- The I/O module obtains the unit of data from the external device.
- The data are transferred from the I/O module to the processor.

Processor Communication:

I/O module communicates with the processor which involves:

- Command decoding: I/O module accepts commands from the processor.
- Data: Data are exchanged between the processor and I/O module over the bus.
- Status reporting: Peripherals are too slow and it is important to know the status of I/O module.
- Address recognition: I/O module must recognize one unique address for each peripheral it controls.

- **Device Communication**: It involves commands, status information and data.
- **Data Buffering**: I/O module must be able to operate at both device and memory speeds. If the I/O device operates at a rate higher than the memory access rate, then the I/O module performs data buffering. If I/O devices rate slower than memory, it buffers data so as not to tie up the memory in slower transfer operation.
- **Error Detection**: I/O module is responsible for error detection such as mechanical and electrical malfunction reported by device e.g. paper jam, bad ink track & unintentional changes to the bit pattern and transmission error.

I/O Module Structure

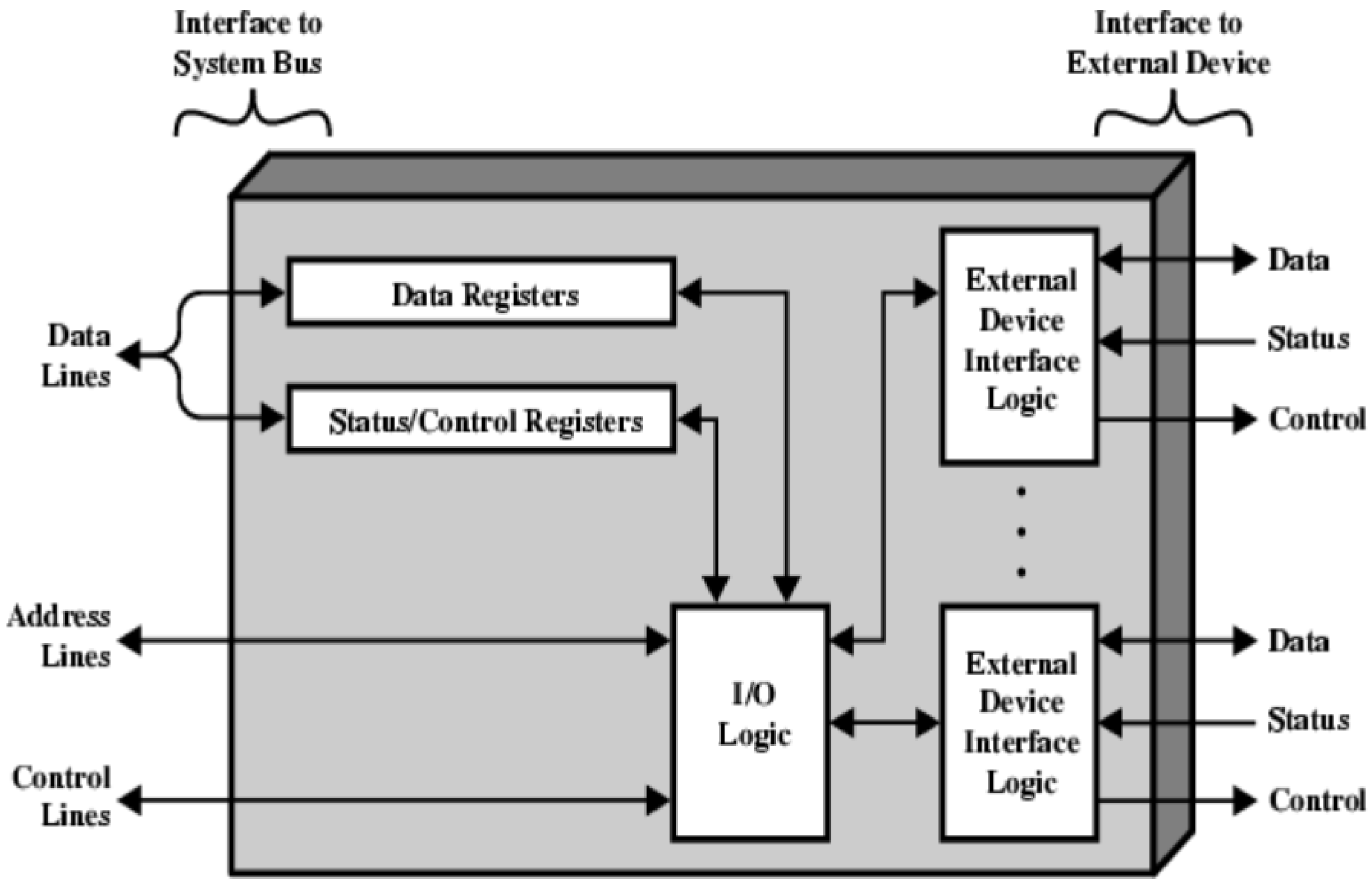
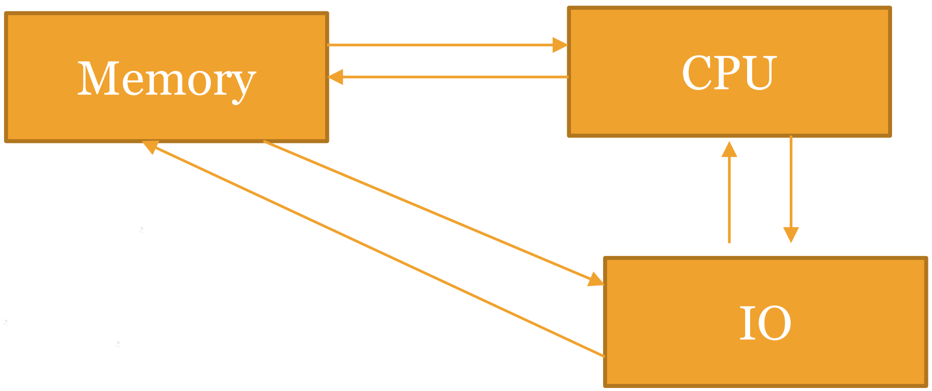


Fig: Block diagram of I/O Module

Modes of Data Transfer

Data transfer from computer system to input output device may be handled by three modes



1. Programmed IO
2. Interrupt Initiated IO
3. Direct Memory Access (DMA)

Programmed I/O

Data are exchange between the processor and I/O module

- Processor execute program controlling I/O operation
- Processor must wait until the I/O module complete the operation.
- If the processor is faster than I/O module
  - CPU time waste

I/O Commands

For I/O related instruction processor issues an address specifying I/O module and an I/O command.

Four types of I/O command is received:

- Control: It activates peripherals to perform required action. eg. Rewind, forward, pause in an magnetic disk etc.
- Test: Checks the status conditions associated with an I/O module and its peripherals. Processor checks for the available peripherals. It also checks the most recent operation is completed or not..
- Read: I/O module reads data from peripherals and place it into an internal buffer. The processor then obtains data by requesting I/O module to place it on data bus.
- Write: I/O module takes data from data bus and transmits the data to peripheral

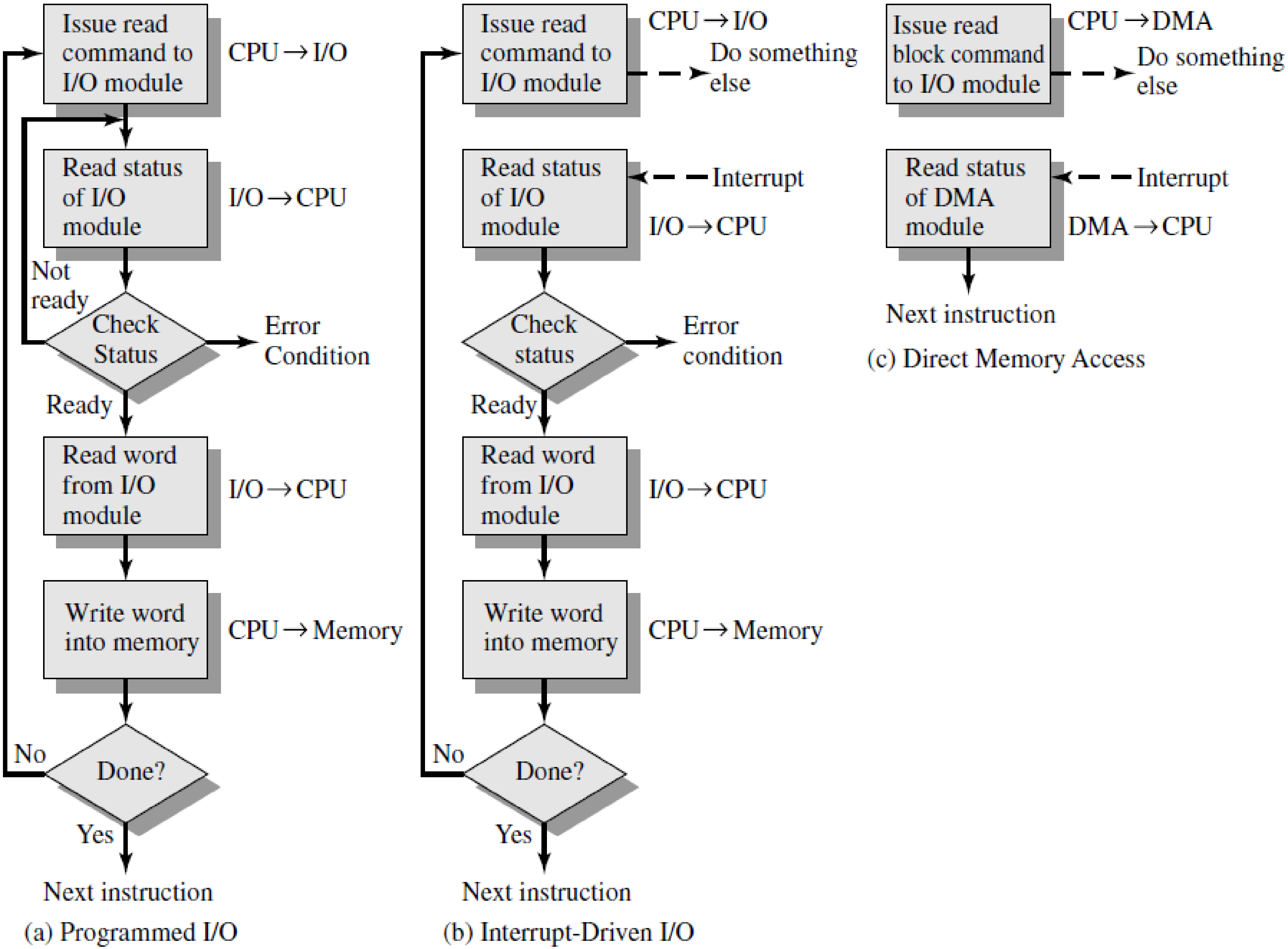


Figure 7.4 Three Techniques for Input of a Block of Data

I/O Instructions

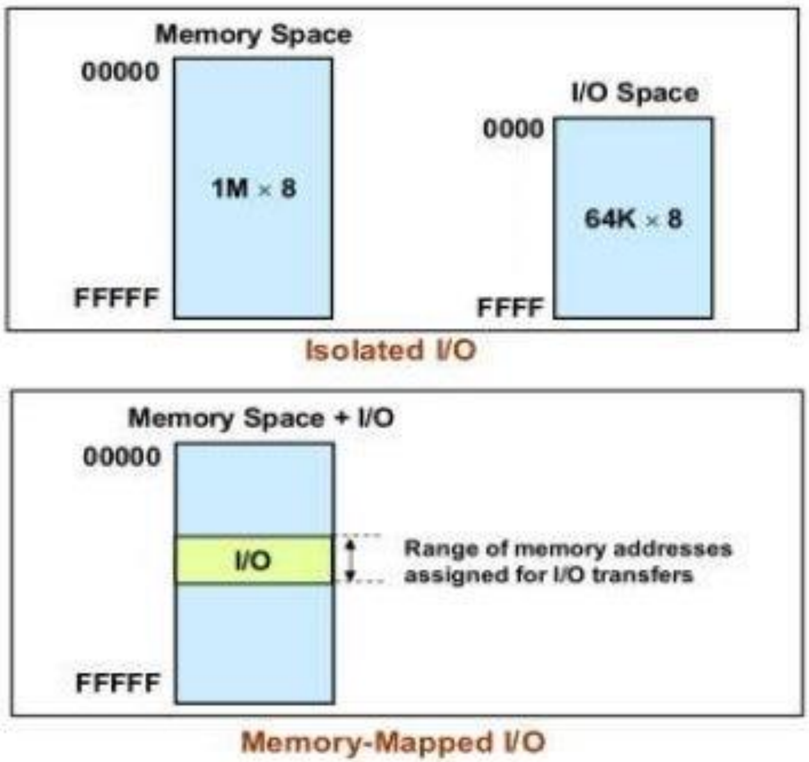
- In programmed I/O there is a relation between the I/O related instructions that processor fetches from memory and I/O commands that processor issues to an I/O module to execute the instructions. Instructions are mapped into I/O commands that are they have one to one relationship.
- There are many devices connected through I/O module to system. Each device has its own unique address. When processor issues command, the command contains the address of desired device. I/O module must interpret the address. When processor, memory and I/O share a common bus, two modes of addressing are possible:

1.Memory mapped

2. Isolated.

[redacted]

Isolated vs. Memory Mapped I/O



- **Memory Mapped I/O:**

Memory mapped I/O uses the same address bus to connect both primary memory and memory of hardware devices (registers). Memory and registers of I/O devices gets assigned values, thus when CPU try to access an address value, it can either from memory or from registers of I/O devices. Memory mapped I/O thus helps in utilizing the same instruction for accessing or addressing both primary memory and I/O device memory locations.

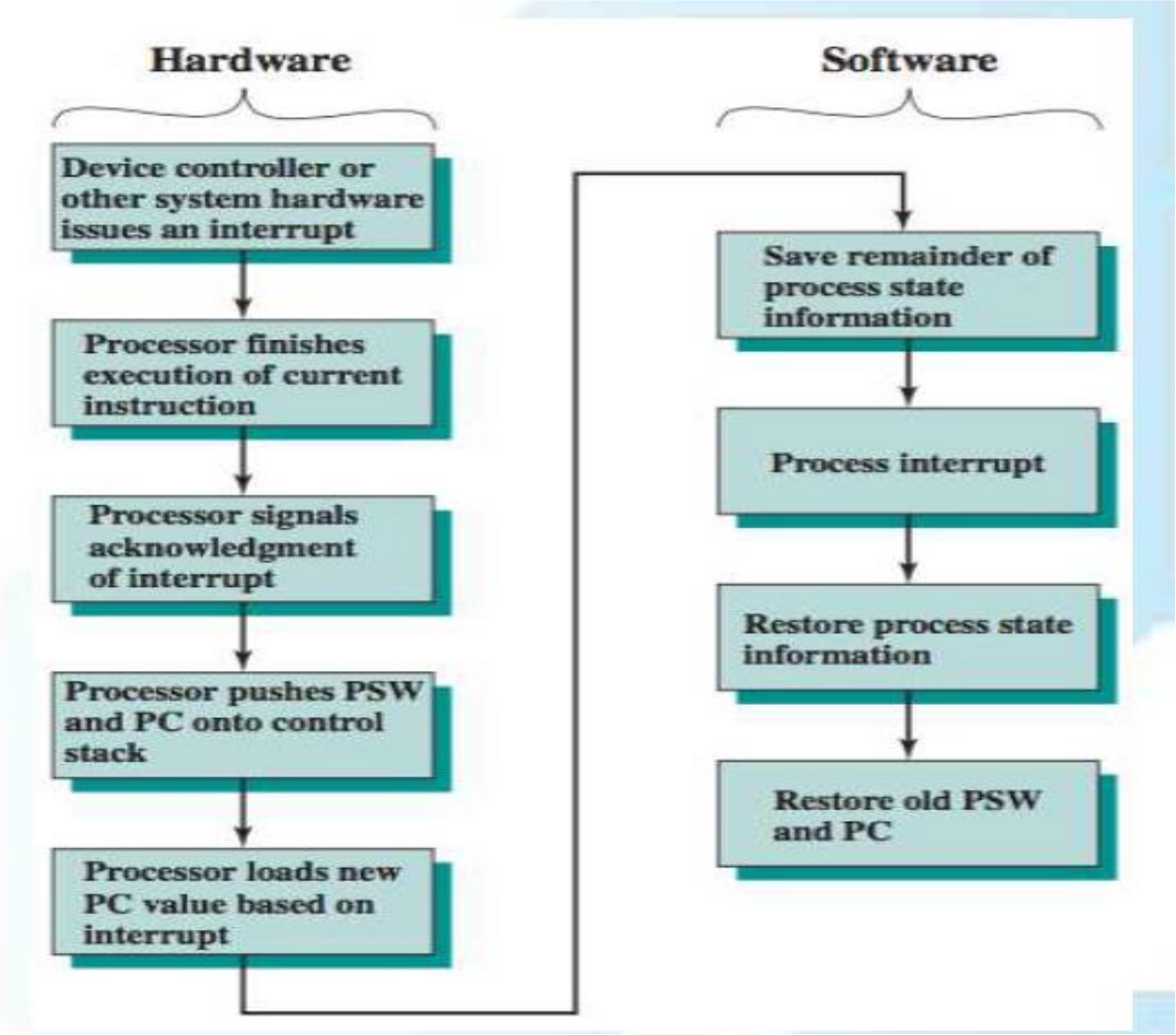
- **Isolated I/O:**

Isolated I/O uses separate instruction classes to access primary memory and device memory. In this case, I/O devices have separate address space either by separate I/O pin on CPU or by entire separate bus. As it separates general memory addresses with I/O devices, it is called isolated I/O. As the peripheral devices are slower than the memory devices, I/O operations can be slow. Isolated I/O accelerates I/O operations by using separate buses.

Interrupt- Driven I/O

- Interrupt driven I/O is an alternative scheme dealing with I/O.
- Interrupt I/O is a way of controlling input/output activity whereby a peripheral or terminal that needs to make or receive a data transfer sends a signal.
- This will cause a program interrupt to be set.
- At a time appropriate to the priority level of the I/O interrupt.
- Relative to the total interrupt system, the processors enter an interrupt service routine.
- The function of the routine will depend upon the system of interrupt levels and priorities that is implemented in the processor.
- The interrupt technique requires more complex hardware and software, but makes far more efficient use of the computer's time and capacities.

simple Interrupt Processing



Design Issues

There are 2 main problems for interrupt I/O, which are:

1. There are multiple I/O modules, how should the processor determine the device that issued the interrupt signal?
2. How does the processor decide which module to process when multiple interrupts have occurred?

There are 4 main ways to counter these problems, which are:

- ***Multiple Interrupt Lines*****:** The processor picks the interrupt line with highest priority . (One of the reasons is that there might be more than one I/O module attached to a single line).

- ***Software Poll*****:** Whenever an interrupt is detected by the processor, it branches to an interrupt service routine which will poll each and every I/O module to determine the exact interrupting module. The processor raises a poll which could be in the form of a command line. Consequently, the address of the respective I/O module which is interacted by the poll will be placed on the address line. The module will respond positively if it is responsible for setting the interrupt. The downside to this techniques is that it is time consuming.

- ***Daisy Chain (Hardware Poll, Vectored)*****:** The priority is determined by the order in which the modules are polled.

- ***Bus Arbitration (Vectored)*****:** This method involves the I/O module gaining control over the bus before requesting for the interrupt.

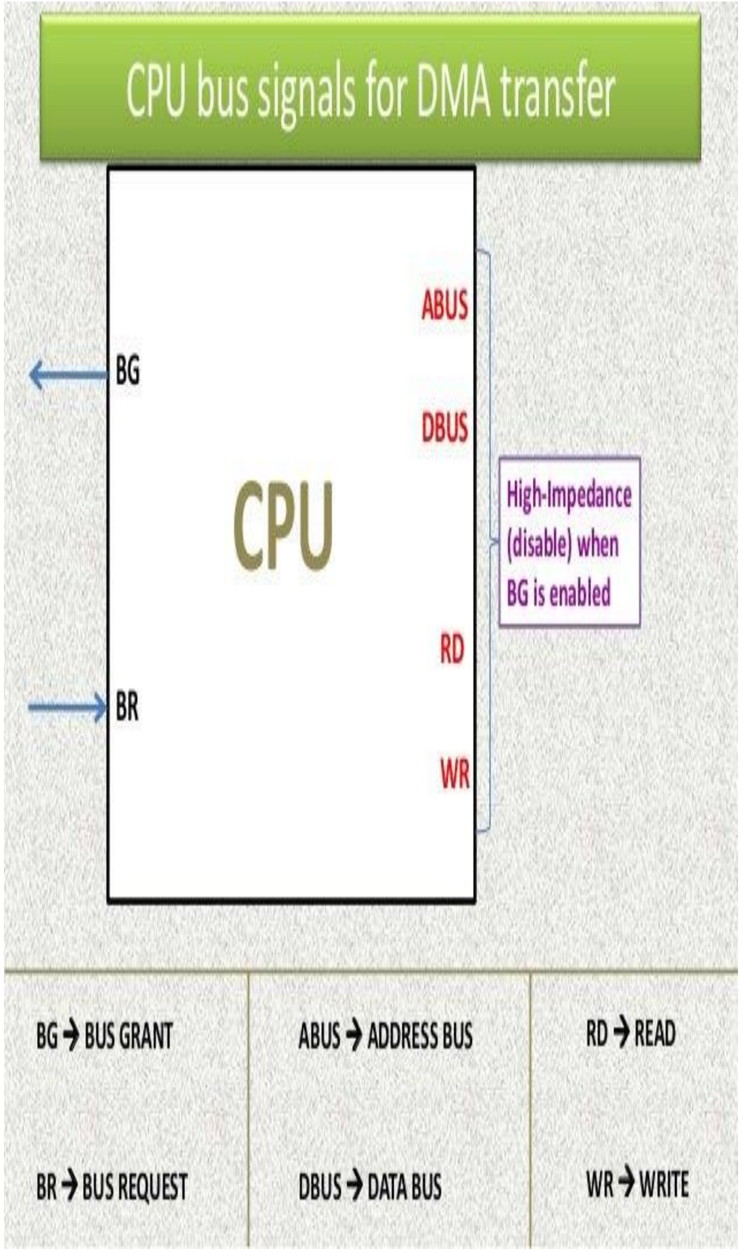
  Employs a priority scheme.

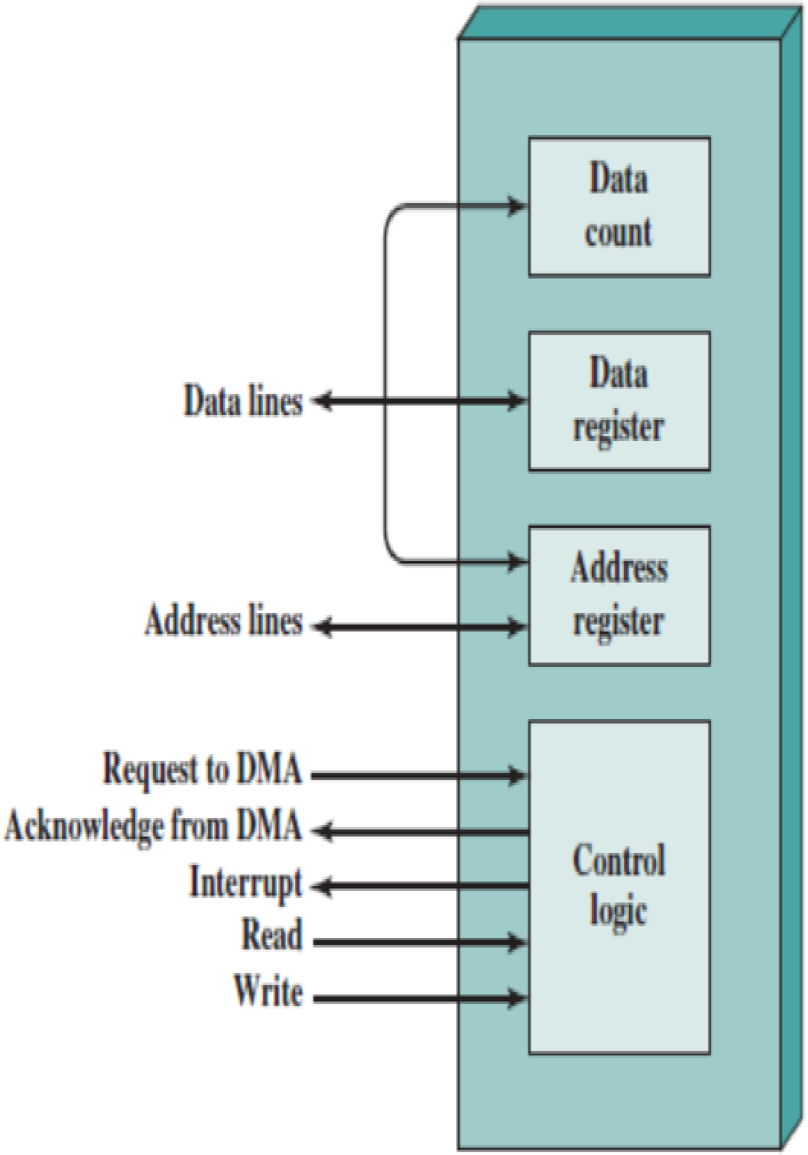
Drawbacks of Programmed and Interrupt – Driven I/O

1. The I/O transfer rate is limited by the speed with which the processor can test and service a device.
2. The processor is tied up in managing an I/O transfer, a number of instruction must be executed for each I/O transfer.

DMA

- Direct Memory Access is a technique for transferring data within main memory and external device without passing it through the CPU.
- DMA is a way to improve processor activity and I/O transfer rate by taking-over the job of transferring data from processor, and letting the processor to do other tasks.
- This technique overcomes the drawbacks of other two I/O techniques which are the time consuming process when issuing command for data transfer and tie-up the processor in data transfer while the data processing is neglected.
- It is more efficient to use DMA method when large volume of data has to be transferred.
- **For DMA to be implemented, processor has to share its' system bus with the DMA module. Therefore, the DMA module must use the bus only when the processor does not need it, or it must force the processor to suspend operation temporarily.**





Typical DMA Block Diagram

I/O Channels & Processors

The Evolution of the I/O Function

1. Processor directly controls peripheral device
2. Addition of a controller or I/O module – programmed I/O
3. Same configuration as 2 – interrupts added
4. I/O module direct access to memory using DMA
5. I/O module enhanced to become processor like – I/O channel
6. I/O module has local memory of its own – computer like – I/O processor

Characteristics of I/O Channels

1) Extension of the **DMA concept**

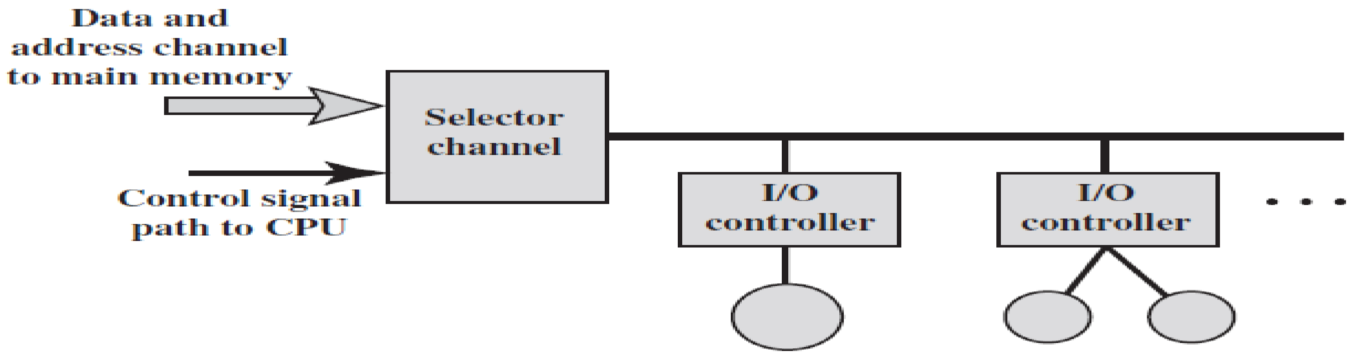
2) Ability to **execute I/O instructions** – complete control over I/O operations

3) Processor does not execute I/O instructions itself (instruction stored in memory to be executed by special purpose processor in I/O channel itself)

4) **Processor initiates I/O transfer** by instructing the I/O channel to execute a program in memory.

Two types of I/O channels

1. Selector
2. Multiplexor



(a) Selector

A *selector channel* controls multiple high- Speed devices and, at any one time, is dedicated to the transfer of data with one of those devices. Thus, the I/O channel selects one device and effects the data transfer. Each device, or a small set of devices, is handled by a *controller*

A *multiplexor channel* can handle I/O with multiple devices at the same time. For low- Speed devices, a ***byte multiplexor*** accepts or transmits characters as fast as possible to multiple devices. **A block multiplexor** interleaves block of data from several devices.

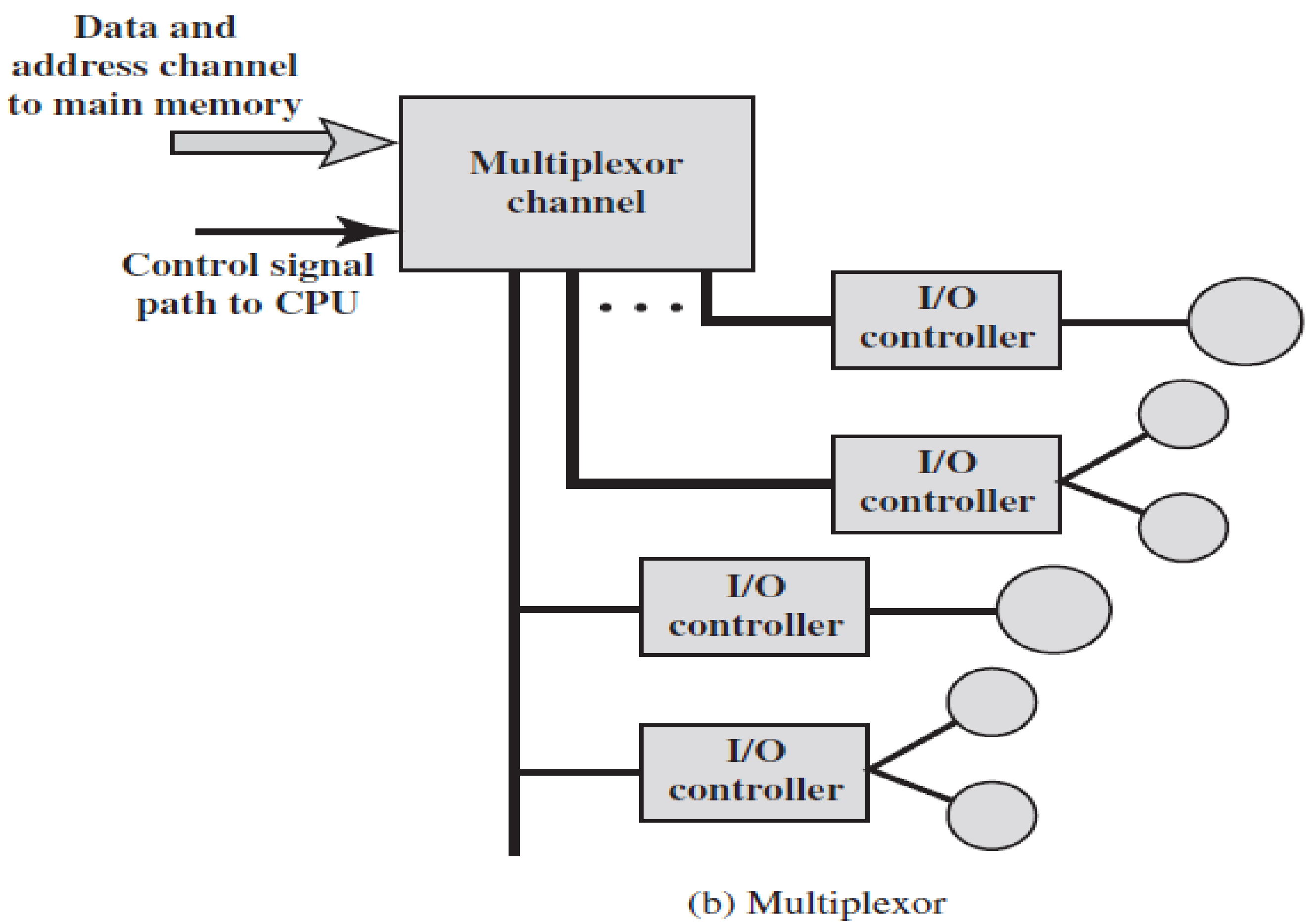


Figure 7.18 I/O Channel Architecture