

Basic I/O System Design

Course Teacher:

Md. Obaidur Rahman, Ph.D.

Professor

Department of Computer Science and Engineering (CSE)
Dhaka University of Engineering & Technology (DUET), Gazipur.

Course ID: CSE - 4503

Course Title: Microprocessors and Assembly Language
Department of Computer Science and Engineering (CSE),
Islamic University of Technology (IUT), Gazipur.

Lecture References:

- ▶ Book:

- ▶ *Microprocessor and Microcomputer – Based System Design*,
Author: Mohamed Rafiquzzaman

- ▶ Lecture Materials:

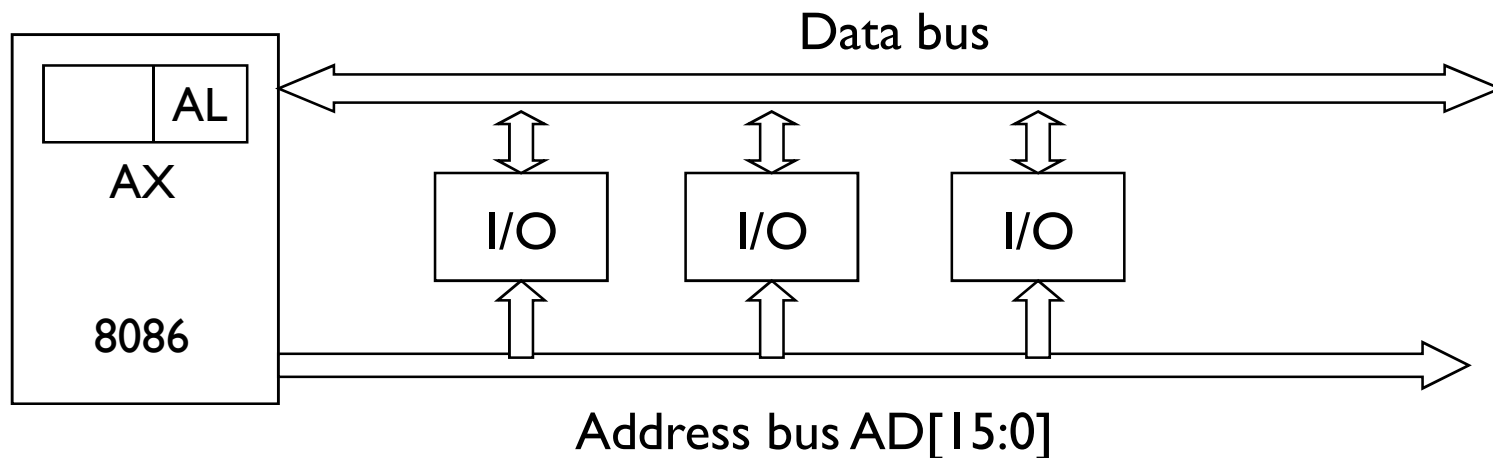
- ▶ *I/O System Design*, Dr. Esam Al_Qaralleh, CE Department, Princess Sumaya University for Technology.
 - ▶ *COMPUTER ORGANISATION & ARCHITECTURE*, Lecture-9 (Input-Output), Dr. Masri Ayob.
 - ▶ <http://home.iae.nl/users/pouwеха/lcd/lcd0.shtml>

Basic I/O System

- ▶ A **Microprocessor** is a great tool for solving problem but is of little or no use if it can't communicate with other devices.
- ▶ **Input-Output devices (or peripherals)** such as Keyboards, Mouse, LEDs, Displays are essential components of the microprocessor-based or microcontroller-based systems.
- ▶ **Input**
 - ▶ Receive data from peripheral (i.e., device)
 - ▶ Send data to computer
- ▶ **Output**
 - ▶ Receive data from computer
 - ▶ Send data to peripheral (i.e., device)

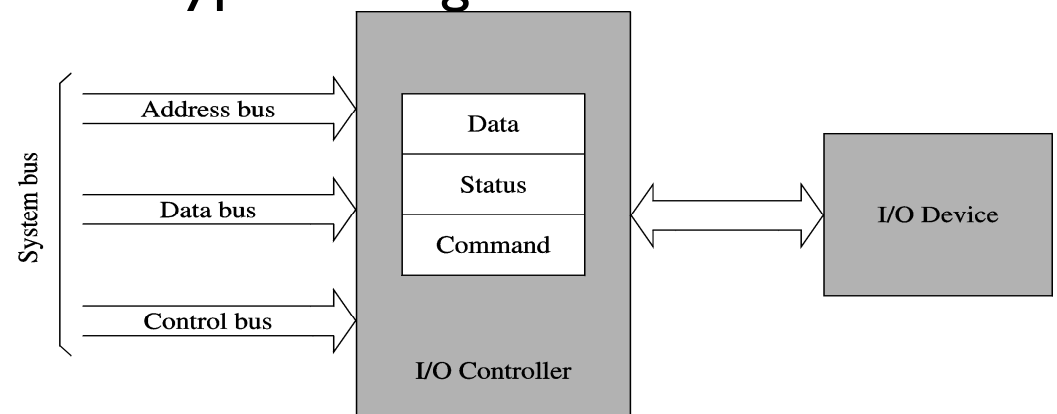
Basic I/O System

- ▶ 8086 processor uses address bus pins AD[0:15] to locate an I/O port
- ▶ 65,536 possible I/O ports
- ▶ Data transfer between ports and the processor occurs over data bus
- ▶ AL (or AX) is the processor register that takes input data (or provide output data)

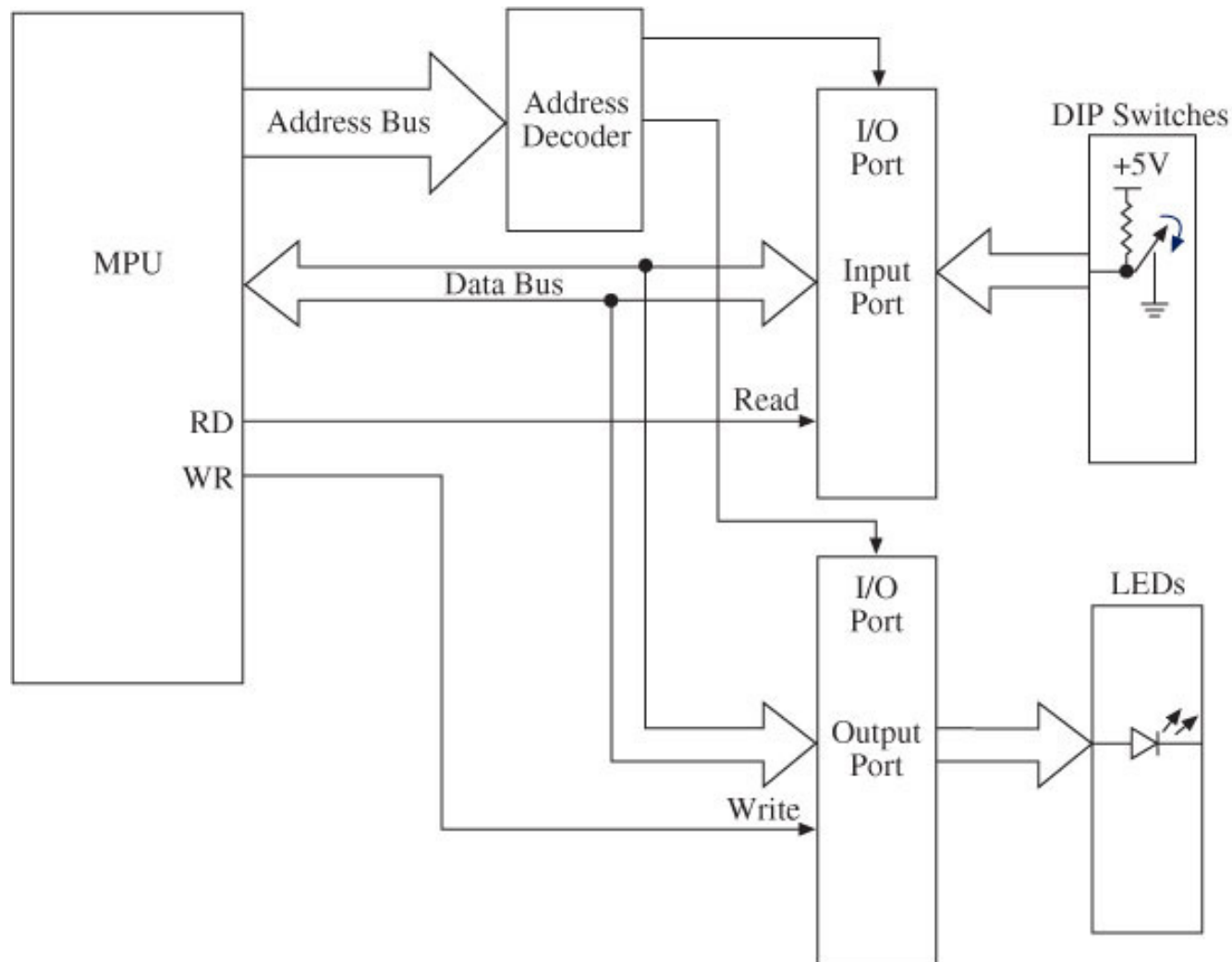


Basic I/O System

- ▶ I/O devices serve two main purposes
 - ▶ To communicate with outside world
 - ▶ To store data
- ▶ **I/O Controller Chip** (e.g., 8255) acts as an interface between the systems bus and I/O device.
 - ▶ Relieves the processor of low-level details
 - ▶ Takes care of electrical interface
- ▶ I/O Controller Chips have three types of registers
 - ▶ Data register
 - ▶ Command register
 - ▶ Status register



Block Diagram of Basic I/O System



Block Diagram of Basic I/O System

- ▶ Access one port at a time is called Serial Interfacing.
- ▶ To **read (receive)** binary data from an input peripheral
 - ▶ **MPU** places the address of an input port on the **address bus**, enables the input port by asserting the **RD signal**, and reads **data** using the **data bus**.
- ▶ To **write (send)** binary data to an output peripheral
 - ▶ MPU places the address of an output port on the **address bus**, places **data** on **data bus**, and asserts the VWR signal to enable the output port.
- ▶ **Remember:**
 - ▶ Writing to the port
 - ▶ When the MPU sends out or transfers data to an output port
 - ▶ Reading from the port
 - ▶ When the MPU receives data from an input port

I/O Instructions

- ▶ **IN** is the instruction that reads information from an I/O device.
- ▶ **OUT** is the instruction that writes/sends data to an I/O device.
- ▶ In reality the data transfer takes place between the microprocessor accumulator (AL or AX) and the I/O device .
- The I/O device may be identified using two methods:
 - **Fixed address** – A byte called **p8** immediately following the **opcode** stores an 8 bit I/O address. This is called fixed because this is stored with the **opcode** in the ROM.
 - **Variable address** – Register DX holds a 16 bit I/O address. Because this can be changed.

I/O Instructions

- ▶ **IN AL, p8** - A byte input from port p8 into AL
- ▶ **IN AX, p8** - A word input from port p8 into AX
- ▶ **IN AL, DX** - A byte input from the port addressed by DX into AL
- ▶ **IN AX, DX** - A word input from the port addressed by DX into AX

I/O Instructions

- ▶ If **p8** is used, only 8 bits are used thus the address or port number appears on lines A0-A7
 - ▶ Hence, the first 256 I/O port addresses (00H – FFH) are accessed by both fixed and variable I/O instructions.
- ▶ If **DX** is used, 16 bits thus the address or port number appears on lines A0-A15
 - ▶ Addresses 0100H-FFFFH are only accessed by variable I/O Address instructions.

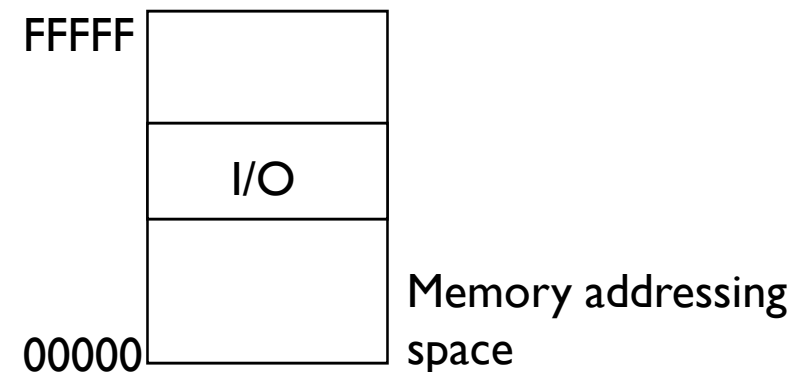
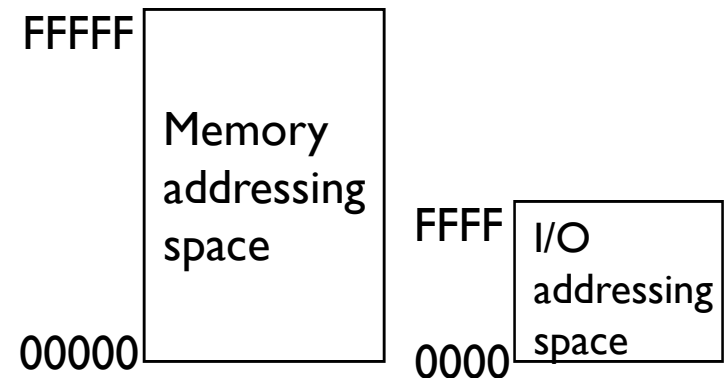
Accessing I/O Devices

- ▶ To communicate with an I/O device, 2 things needed-
 - ▶ **Access to various registers** (data, status,...)
 - ▶ This access depends on I/O address mapping,
 - Two basic ways
 - Memory-mapped I/O
 - Isolated I/O
 - ▶ **A protocol to communicate** (to send data, ...)
 - ▶ Three types
 - Programmed I/O
 - Interrupt-driven I/O
 - Direct memory access (DMA)

I/O Address Mapping

▶ Memory Mapped I/O

- ▶ A device is mapped to a memory location. Sending data to the particular location causes interaction with the device.
- ▶ Reading and writing are similar to memory read/write with same address bus
- ▶ Uses same memory read and write signals
- ▶ Most processors use this I/O mapping



Memory-mapped I/O

I/O Address Mapping

▶ **Memory Mapped I/O**

▶ **Advantages:**

- ▶ Less complication,
- ▶ Less circuitry
- ▶ Less decoding
- ▶ No need to use special signal
- ▶ MOV instruction can be used instead of IN and OUT

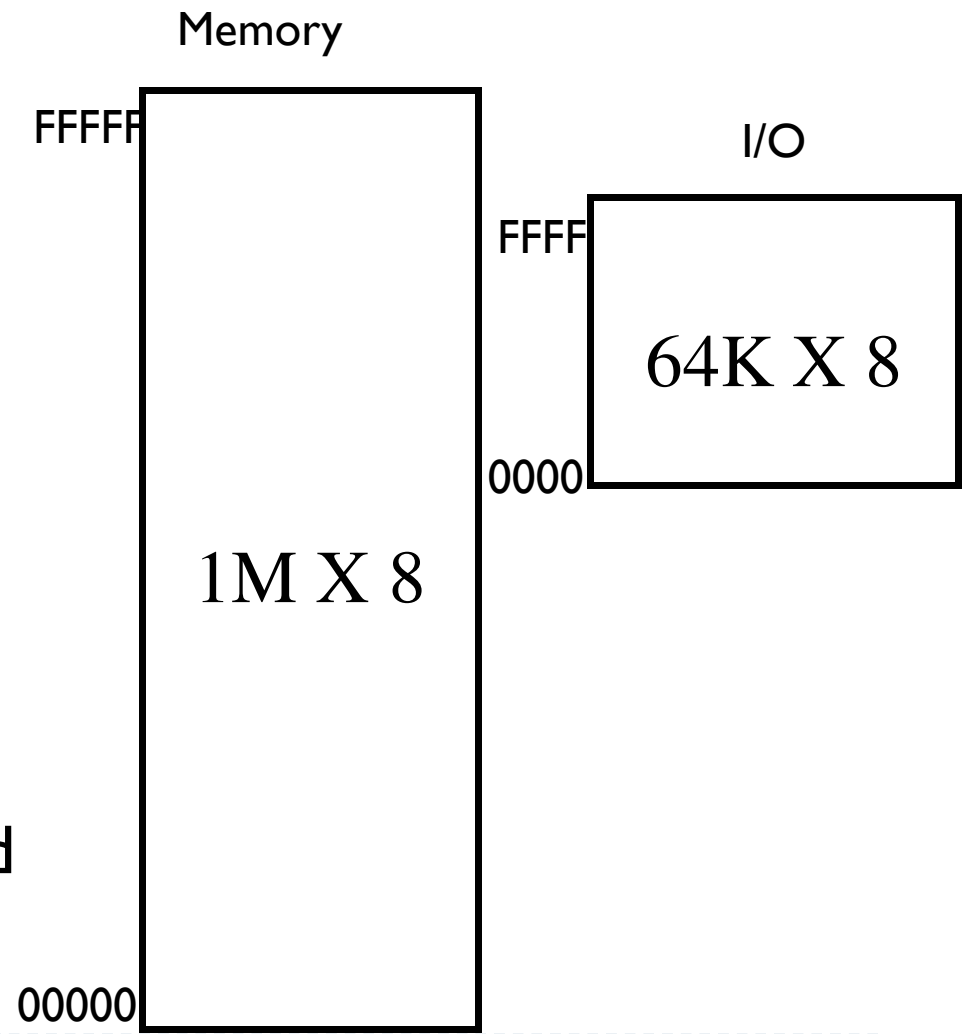
▶ **Disadvantage:**

- ▶ A portion of the memory system is used as the I/O Mapping

I/O Address Mapping

► Isolated I/O

- Separate I/O address space
- Separate I/O read and write signals are needed
 - I/O read
 - I/O write
- This is the most common form of I/O transfer technique used with Intel processors.
- Pentium supports isolated I/O of 64 KB address space.



I/O Address Mapping

▶ **Isolated I/O**

- ▶ On Personal computers **Isolated I/O** is used.
- ▶ 8 bit port addresses are used to access devices located on system board.
- ▶ 16 bit port address are used to access serial and parallel ports as well as video and disk drive systems.

I/O Address Mapping

▶ **Isolated I/O**

▶ **Advantages:**

- ▶ In this system no memory is wasted for I/O mapping.

▶ **Disadvantage:**

- ▶ Instructions **IN** and **OUT** need to be used to perform data transfer.
- ▶ Separate signals are also needed to interact with the I/O devices which separate it from normal memory access instructions.

Accessing I/O Ports in 8086

- ▶ Register I/O instructions

IN accumulator, port8; direct format

- Useful to access first 256 ports

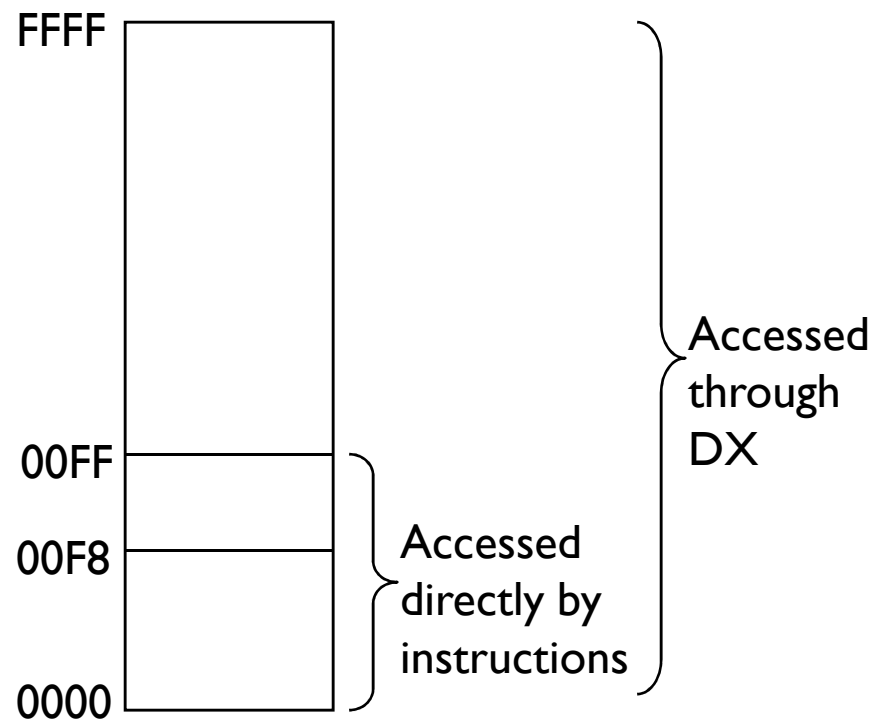
IN accumulator, DX ; indirect format

- DX gives the port address

- ▶ Simple MOV instruction can be used instead of IN and OUT.

Accessing I/O Ports in 8086

► Addressing Space



❑ Accessing directly by instructions

```
IN    AL,    80H
IN    AX,    06H
OUT   3CH,   AL
OUT   A0H,   AX
```

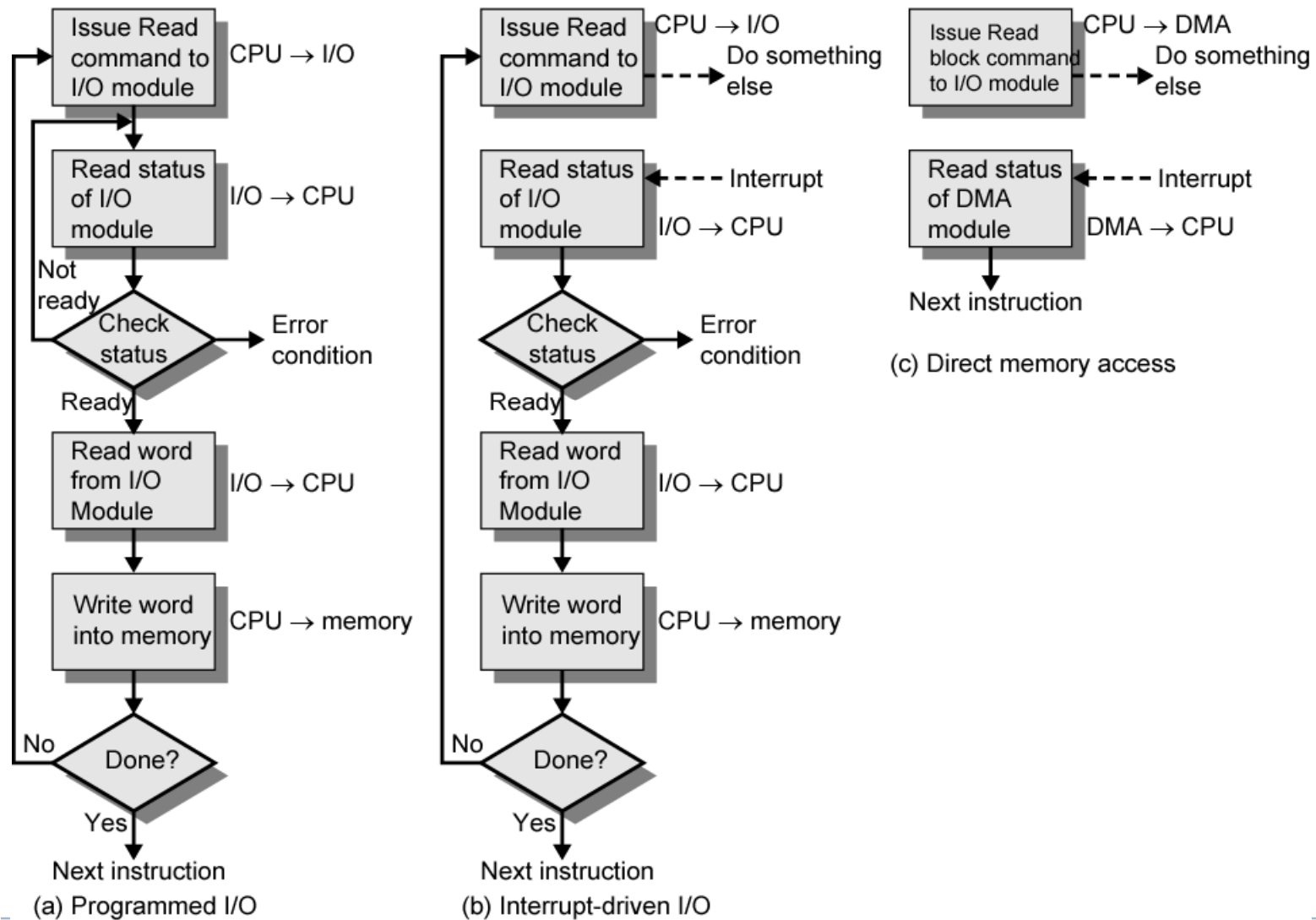
❑ Accessing through DX

```
IN    AL,    DX
IN    AX,    DX
OUT   DX,    AL
OUT   DX,    AX
```

I/O Data Transfer Techniques

- ▶ Data transfer involves three basic techniques
 - ▶ Programmed I/O
 - ▶ DMA
 - ▶ Interrupt-driven I/O

I/O Data Transfer Techniques



Programmed I/O

- ▶ CPU has direct control over I/O
 - ▶ Sensing status
 - ▶ Read/write commands
 - ▶ Transferring data
- ▶ CPU waits for I/O module to complete operation
- ▶ Programmed I/O Wastes CPU time
- ▶ **Addressing I/O Devices**
 - ▶ Under programmed I/O data transfer is very like memory access (CPU viewpoint)
 - ▶ Each device given a unique identifier
 - ▶ CPU commands contain identifier (address)

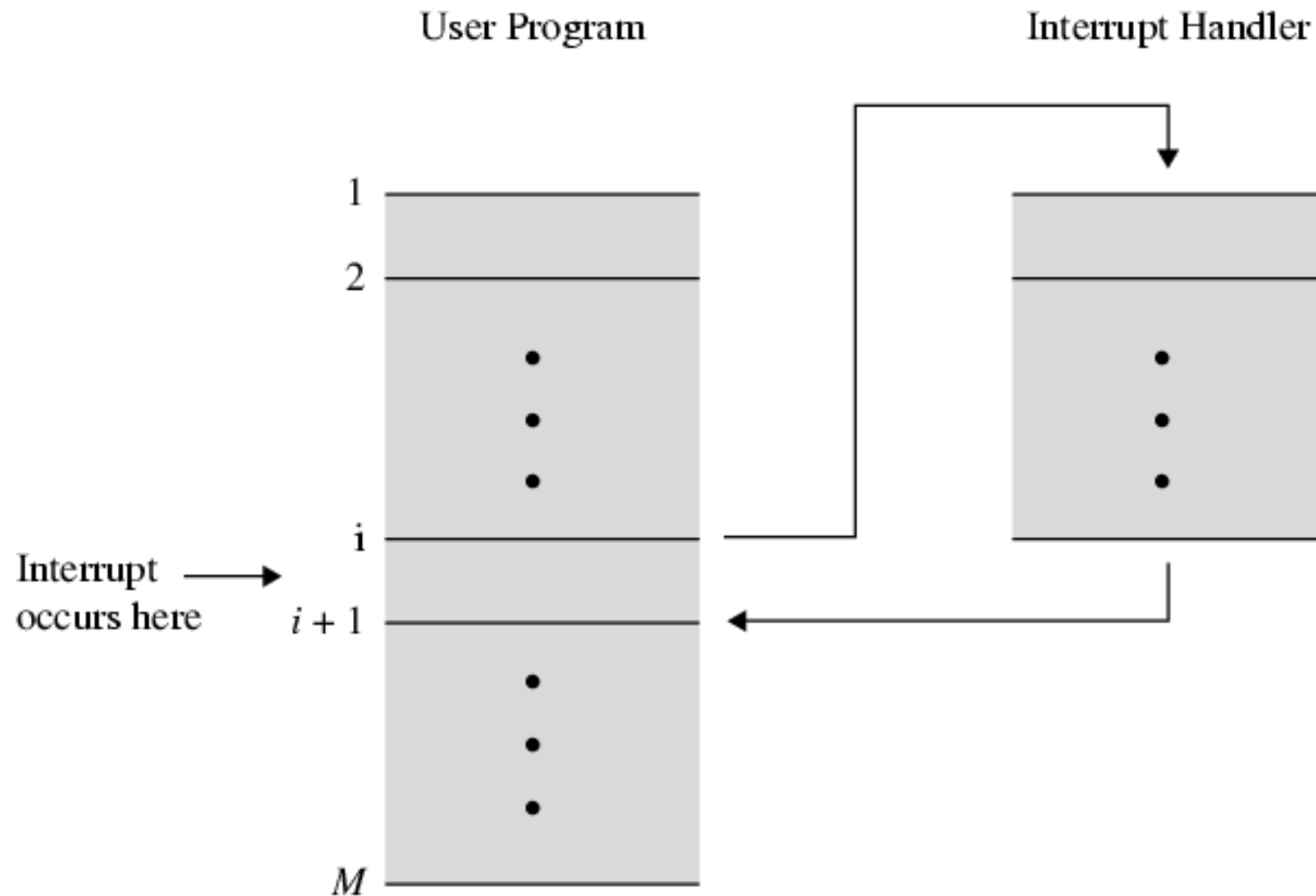
Programmed I/O – Detail Steps

- ▶ CPU requests I/O operation
- ▶ I/O module performs operation
- ▶ I/O module sets status bits
- ▶ CPU checks status bits periodically - **polling**
- ▶ I/O module does not inform CPU directly
- ▶ I/O module does not interrupt CPU
- ▶ CPU may wait or come back later

Interrupt Driven I/O

- ▶ It Overcomes **CPU Waiting Time Problem** of Programmed I/O
- ▶ No repeated CPU checking of device
- ▶ I/O module interrupts when it is **ready**
- ▶ ***Interrupt Handler Chip*** is responsible to make successful data transfer.

Transfer of Control via Interrupts



Multiple Interrupts

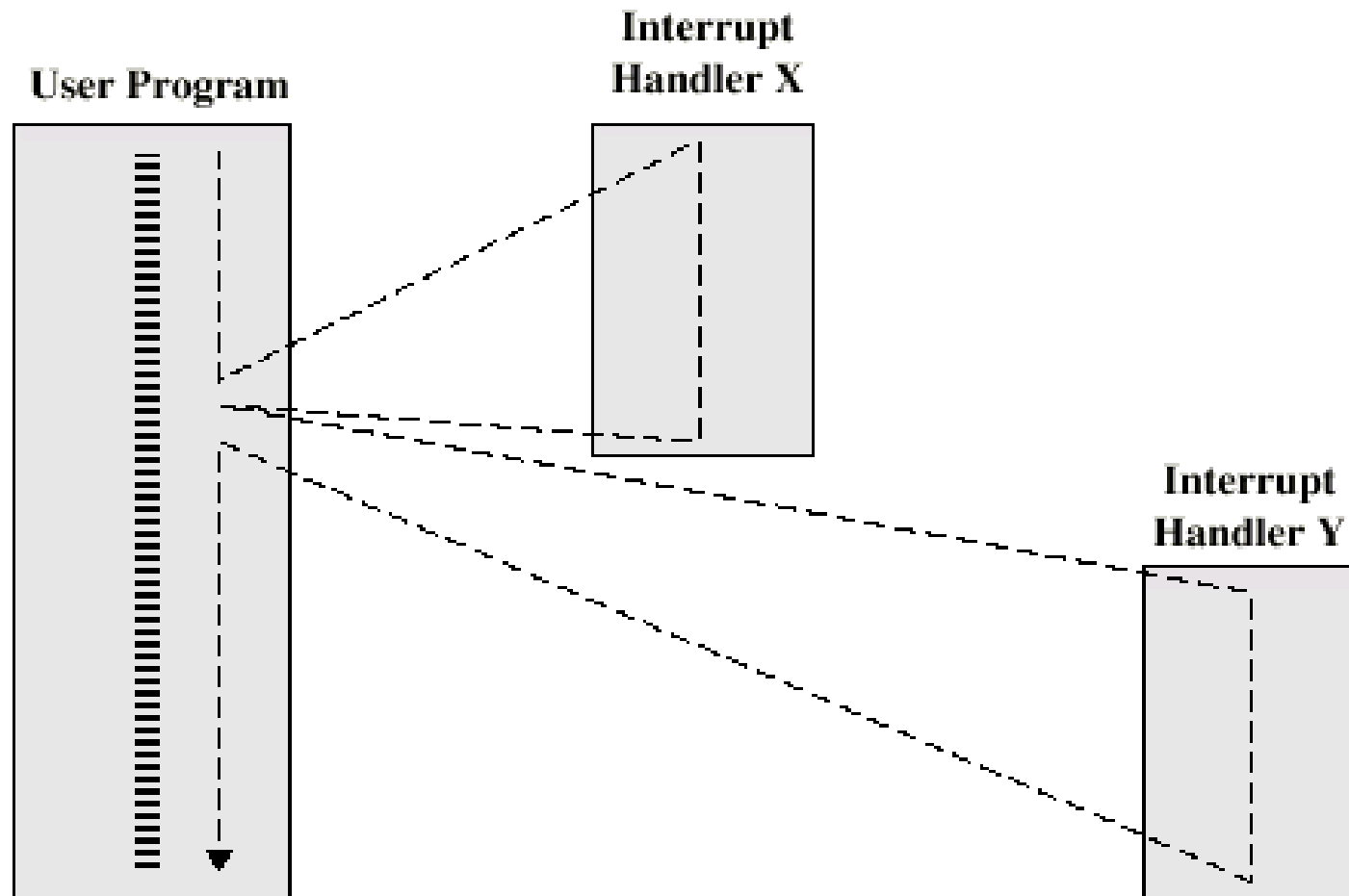
- ▶ **Disable interrupts**

- ▶ Processor will ignore further interrupts whilst processing one interrupt
- ▶ Interrupts remain pending and are checked after first interrupt has been processed
- ▶ Interrupts handled in sequence as they occur

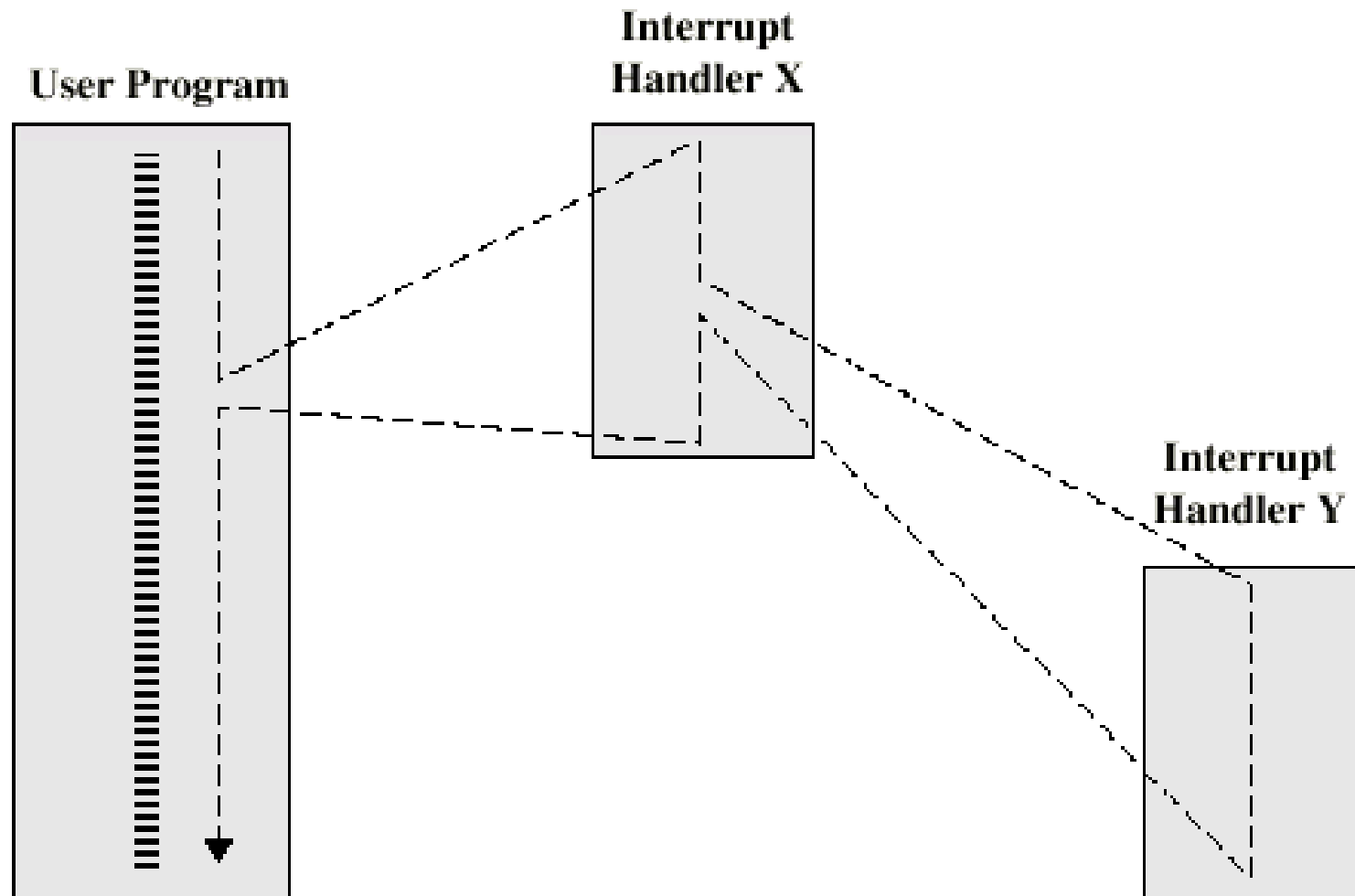
- ▶ **Define priorities**

- ▶ Low priority interrupts can be interrupted by higher priority interrupts
- ▶ When higher priority interrupt has been processed, processor returns to previous interrupt

Multiple Interrupts - Sequential



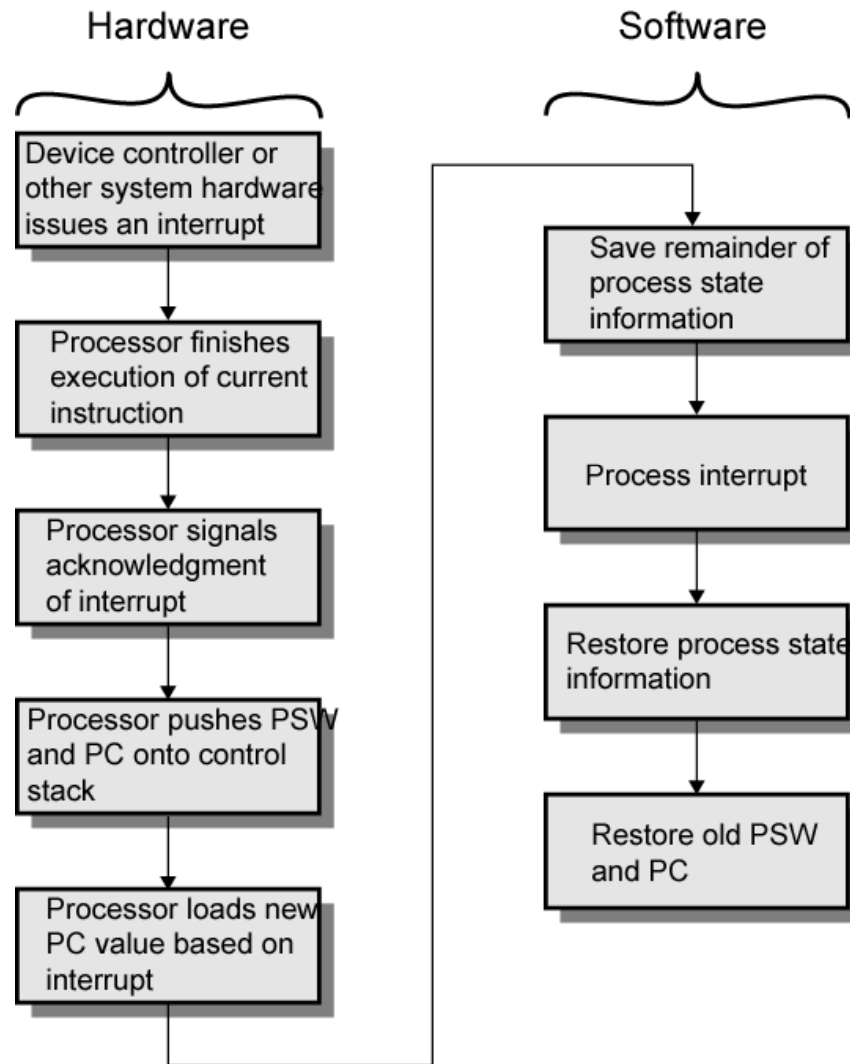
Multiple Interrupts – Nested



Interrupt Driven I/O: Detail Steps

- ▶ CPU issues read command
- ▶ I/O module gets data from peripheral whilst CPU does other work
- ▶ I/O module interrupts CPU
- ▶ CPU requests data
- ▶ I/O module transfers data

Simple Interrupt Processing



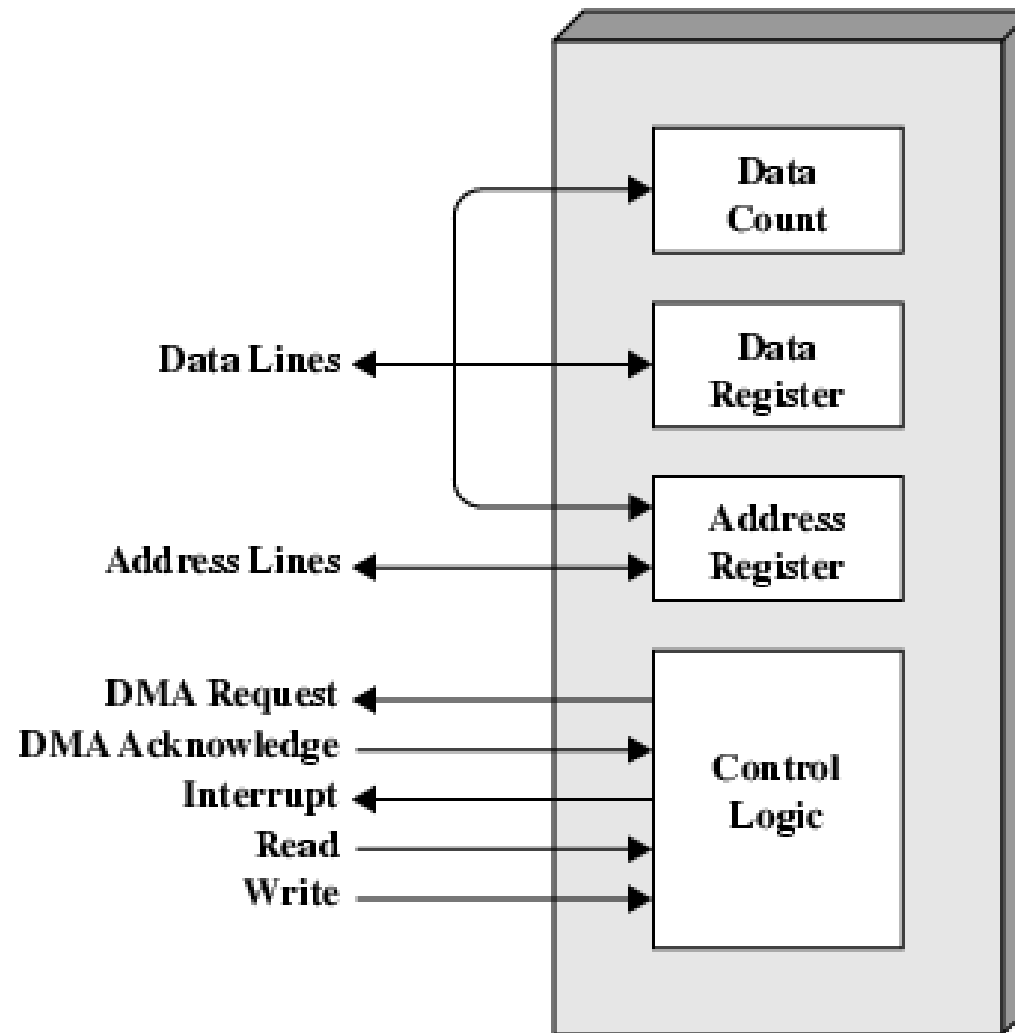
Direct Memory Access (DMA)

- ▶ Interrupt driven and programmed I/O require active CPU intervention
 - ▶ Hence, the data transfer rate is limited
 - ▶ For typical microprocessors 1 (one) byte of data transfer between RAM and I/O take 5 – 10 microseconds.
 - ▶ Hence, CPU is tied up
- ▶ ***DMA is the answer***
 - ▶ DMA is more efficient for transferring large volumes of data.

Direct Memory Access (DMA)

- ▶ DMA doesn't make use of the microprocessor to do the transfer so who is going to do the READ and WRITE operation?
- ▶ A **DMA Controller Chip** is may be built into the microprocessor or may be found external.
- ▶ We can see that the DMA controller has to perform operations that are very similar to operations performed by a microprocessor.
- ▶ They are thus generally designed and built by microprocessor manufacturers.

Typical DMA Module Diagram



DMA Operation – Detail Steps

- ▶ CPU tells DMA controller:-
 - ▶ Read/Write
 - ▶ Device address
 - ▶ Starting address of memory block for data
 - ▶ Amount of data to be transferred
- ▶ CPU carries on with other work
- ▶ DMA controller deals with transfer
- ▶ DMA controller sends interrupt when finished

Thank You !!

