Input/Output System
External Devices

Contents of Lecture:
- ❖ Introduction
- ❖ Generic Model of I/O Module
- ❖ External Devices
- ❖ I/O Module Function
- ❖ I/O Module Structure
- ❖ I/ O Techniques
- ❖ **Summarize number (2)**

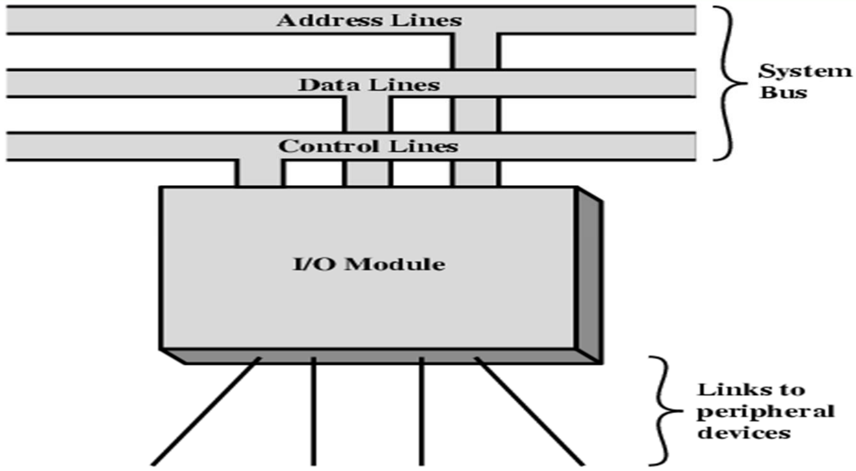
References for This Lecture:
- ❖ William Stallings, Computer Organization and Architecture Designing For Performance, 9th Edition, Chapter 7 : *INPUT/OUTPUT*

Introduction:
- ❖ The computer system's I/O architecture is its interface to the outside world.
- ❖ This architecture provides a systematic means of controlling interaction with the outside world and provides the operating system with the information it needs to manage I/O activity effectively.

- ❖ **Input/output Problems:**
  - ✓ Wide variety of peripherals
    - ∞ Delivering different amounts of data
    - ∞ At different speeds
    - ∞ In different formats
  - ✓ All slower than CPU and RAM
  - ✓ Need I/O modules

Generic Model of I/O Module:
- ❖ The I/O module contains logic for performing a communication **function** between the peripheral and the bus.

- ❖ This module has two major functions (following Figure):
  - ✓ Interface to the processor and memory via the system bus or central switch
  - ✓ Interface to one or more peripheral devices by tailored data links



External Devices:

- External devices that provide a means of exchanging data between the external environment and the computer.

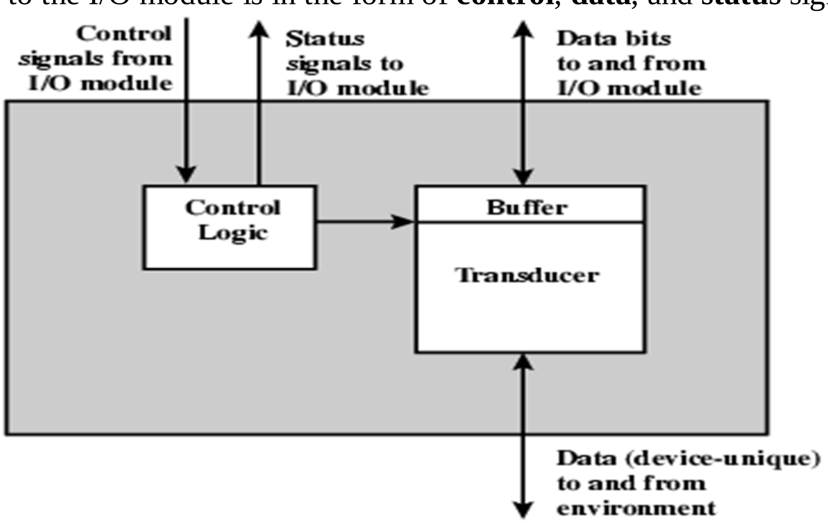
- An external device attaches to the computer by a link to an I/O module (A above Figure). The link is used to exchange **control**, **status**, and **data** between the I/O module and the external device.

- We can classify external devices into **<u>three categories</u>**:

  - ✓ <u>Human readable</u>: Suitable for communicating with the computer user
    - Screen, printer, keyboard
  - ✓ <u>Machine readable</u>: Suitable for communicating with equipment
    - magnetic disk and tape systems, and sensors and actuators (Monitoring and control)
  - ✓ <u>Communication</u>: Suitable for communicating with remote devices
    - Modem
    - Network Interface Card (NIC)

The nature of an external device:

- The interface to the I/O module is in the form of **control**, **data**, and **status** signals.



- **Control signals** determine the function that the device will perform, such as **send data** to the I/O module (INPUT or READ), **accept data** from the I/O module (OUTPUT or WRITE), report status, or perform some control function particular to the device (e.g., position a disk head).

- **Data** are in the form of a set of bits to be sent to or received from the I/O module.
- **Status signals** indicate the state of the device. Examples are READY/NOT-READY to show whether the device is ready for data transfer.

- **Control logic** associated with the device controls the device's operation in response to direction from the I/O module.

- **The transducer** converts data from electrical to other forms of energy during output and from other forms to electrical during input.

- **A buffer** is associated with the transducer to temporarily hold data being transferred between the I/O module and the external environment.

I/O Module Function:

The major functions or requirements for an I/O module fall into the following categories:

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

1. **Control and timing:** the I/O function includes a control and timing requirement, to coordinate the flow of traffic between internal resources and external devices.

   For example, the control of the transfer of data from an external device to the processor might involve the following sequence of steps:-
   - ✓ The processor interrogates the I/O module to check the 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 the I/O module.
   - ✓ The I/O module obtains a unit of data (e.g., 8 or 16 bits) from the external device.
   - ✓ The data are transferred from the I/O module to the processor.

2. **Processor communication:** the I/O module must communicate with the processor and with the external device. ***Processor communication involves*** the following:
   - ✓ **Command decoding:** The I/O module accepts commands from the processor, typically sent as signals on the control bus. For example, an I/O module for a disk drive might accept the following commands: READ SECTOR, WRITE SECTOR, SEEK track number, and SCAN record ID. The latter two commands each include a parameter that is sent on the data bus.

- ✓ **Data:** Data are exchanged between the processor and the I/O module over the data bus.
- ✓ **Status reporting:** Because peripherals are so slow, it is important to know the status of the I/O module. For example, if an I/O module is asked to send data to the processor (read), it may not be ready to do so because it is still working on the previous I/O command. This fact can be reported with a status signal. Common status signals are BUSY and READY. There may also be signals to report various error conditions.
- ✓ **Address recognition:** an I/O module must recognize one unique address for each peripheral it controls.

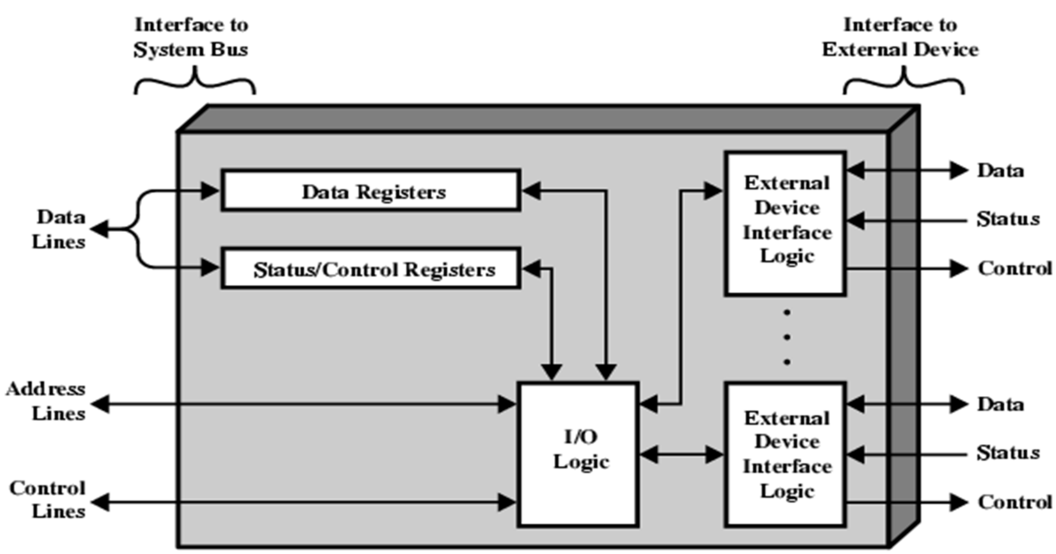
3. **Device communication:** the I/O module must be able to perform device communication. This communication involves commands, status information, and data.

4. **Data buffering:** The data are buffered in the I/O module and then sent to the peripheral device at its data rate. In the opposite direction, data are buffered so as not to tie up the memory in a slow transfer operation. The I/O module must be able to operate at both device and memory speeds.

5. **Error detection:** an I/O module is often responsible for error detection and for subsequently reporting errors to the processor. One class of errors includes mechanical and electrical malfunctions reported by the device (e.g., paper jam, bad disk track).Another class consists of unintentional changes to the bit pattern as it is transmitted from device to I/O module. Some form of error-detecting code is often used to detect transmission errors. When a byte is received, the I/O module checks the parity to determine whether an error has occurred.

<u>I/O Module Structure:</u>

- ❖ Next figure provides a general block diagram of an I/O module.



- The module connects to the rest of the computer through a set of signal lines (e.g., system bus lines).
- Data transferred to and from the module are buffered in one or more data registers; there may also be one or more status registers that provide current status information.
- A status register may also function as a control register, to accept detailed control information from the processor.
- The logic within the module interacts with the processor via a set of control lines.
- The processor uses the control lines to issue commands to the I/O module.
- The module must also be able to recognize and generate addresses associated with the devices it controls. Each I/O module has a unique address or, if it controls more than one external device, a unique set of addresses.
- The I/O module contains logic specific to the interface with each device that it controls.
- An I/O module functions to allow the processor to view a wide range of devices in a simple-minded way.
- The I/O module may hide the details of timing, formats, and the electro mechanics of an external device so that the processor can function in terms of simple read and write commands, and possibly open and close file commands.
- An I/O module that takes on most of the detailed processing burden, presenting a high-level interface to the processor, is usually referred to as an I/O channel or I/O processor. An I/O module that is quite primitive and requires detailed control is usually referred to as an I/O controller or device controller.

I/ O Techniques:

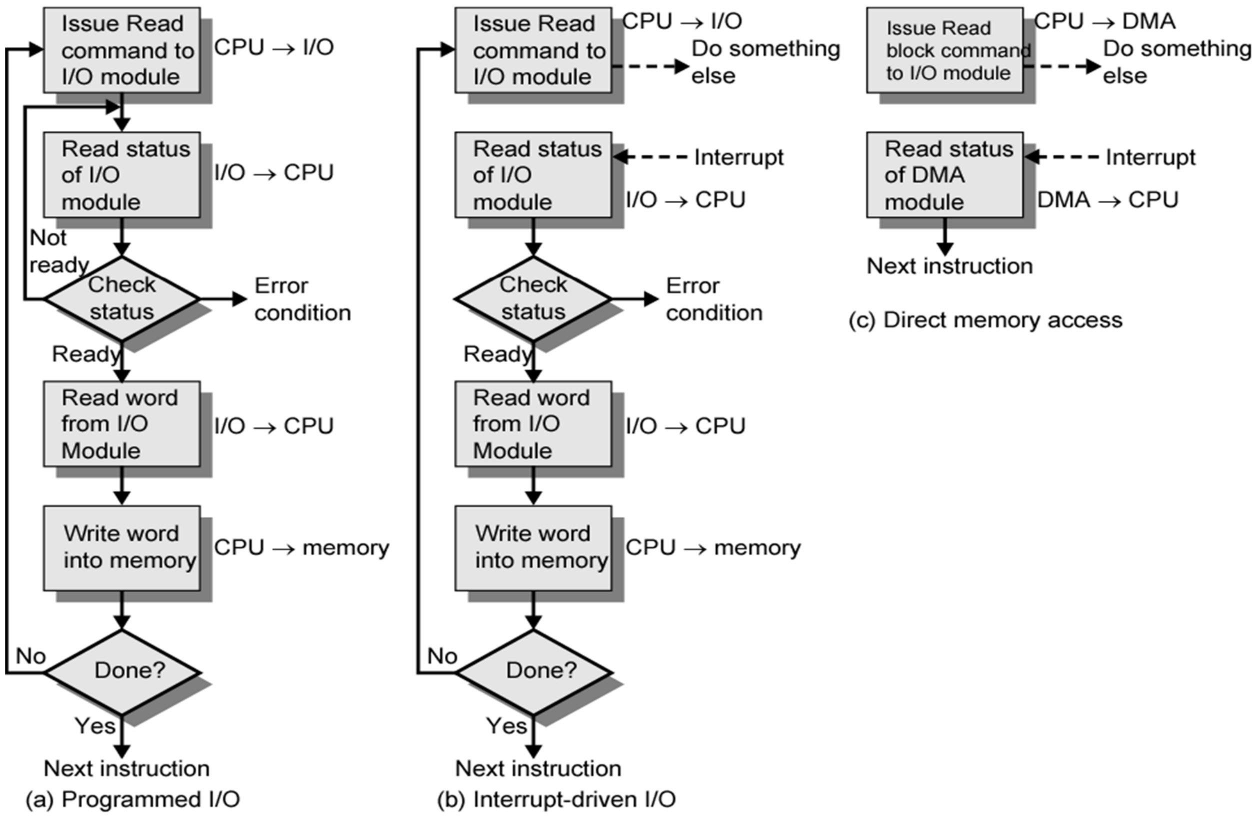
Three techniques are possible for I/O operations:-

1. **Programmed I/O:** data are exchanged between the processor and the I/O module. The processor executes a program that gives it direct control of the I/O operation, including sensing device status, sending a read or write command, and transferring the data. When the processor issues a command to the I/O module, it must wait until the I/O operation is complete. If the processor is faster than the I/O module, this is wasteful of processor time.

2. **Interrupt-driven I/O:** the processor issues an I/O command, continues to execute other instructions, and is interrupted by the I/O module when the latter has completed its work.

3. **Direct Memory Access (DMA):** With both programmed and interrupt I/O, The processor responsible for extracting data from main memory for output and storing data in main memory for input. With DMA the I/O module and main memory exchange data directly, without processor involvement.

Figure (4): Three Techniques for Input of a Block of Data

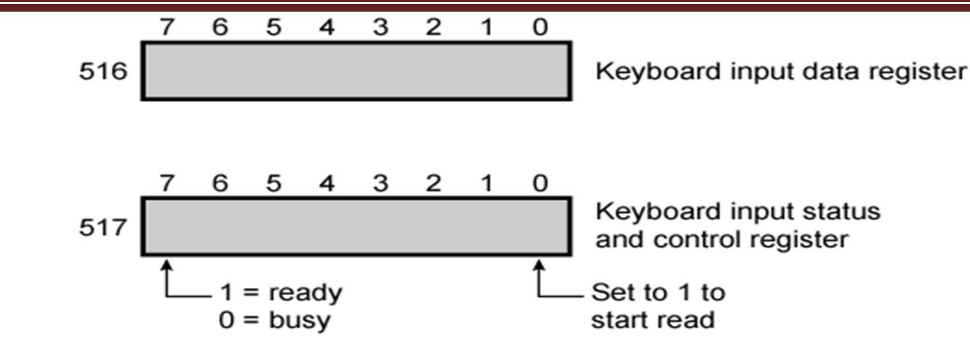


Above figure in *a* part gives an example of the use of programmed I/O to read in a block of data from a peripheral device (e.g., a record from tape) into memory. Data are read in one word (e.g., 16 bits) at a time. For each word that is read in, the processor must remain in a status-checking cycle until it determines that the word is available in the I/O module's data register. This flowchart highlights the main disadvantage of this technique: it is a time-consuming process that keeps the processor busy needlessly**.**

Summarize number(2):

1. Explain how to execute an I/O-related instruction (I/O Commands). By explain the command **Control**, **Test**, **Read** and **Write**.

2. Have the following figure; Discus the **instructions** and **Address** at the **Memory-mapped I/O** and **Isolated I/O**.



ADDRESS	INSTRUCTION	OPERAND	COMMENT
200	Load I/O	5	Initiate keyboard read
201	Test I/O	5	Check for completion
	Branch Not Ready	201	Loop until complete
	In	5	Load data byte

(b) Isolated I/O

ADDRESS	INSTRUCTION	OPERAND	COMMENT
200	Load AC	"1"	Load accumulator
	Store AC	517	Initiate keyboard read
202	Load AC	517	Get status byte
	Branch if Sign = 0	202	Loop until ready
	Load AC	516	Load data byte

(a) Memory-mapped I/O

3. Direct Memory Access Module Function (DMA Module Function) is capable of mimicking the processor and, indeed, of taking over control of the system from the processor. When the processor wishes to read or write a block of data, it issues a command to the DMA module, by sending to the DMA module by using much information. <u>Discus **that information** and the **following diagram**</u>**.**