COMPUTER ARCHITECTURE

LECTURE 3

BY NDUMISO E. KHUMALO

INPUT/OUTPUT

Objective

- External Devices, I/O Modules.
- Programmed I/O
- Interrupt-driven I/O
- Direct Memory Access
- I/O Channel and Processors

Introduction

- In addition to the processor and a set of memory modules, the third key element of a computer system is a set of I/O modules
- Each module interfaces to the system bus or central switch & controls one or more peripheral devices.
  - An I/O module is not simply a set of mechanical connectors that wire a device into the system bus.
  - Rather, the I/O module contains logic for performing a communication function between the peripheral and the bus.

Introduction

- Why don't peripheral devices connect directly to the system bus?
  - There are a wide variety of peripherals with various methods of operation.
    - It would be impractical to incorporate the necessary logic within the processor to control a range of devices.
  - The data transfer rate of peripherals is often much slower than that of the memory or processor.
    - Thus, it is impractical to use the high-speed system bus to communicate directly with a peripheral.

Introduction

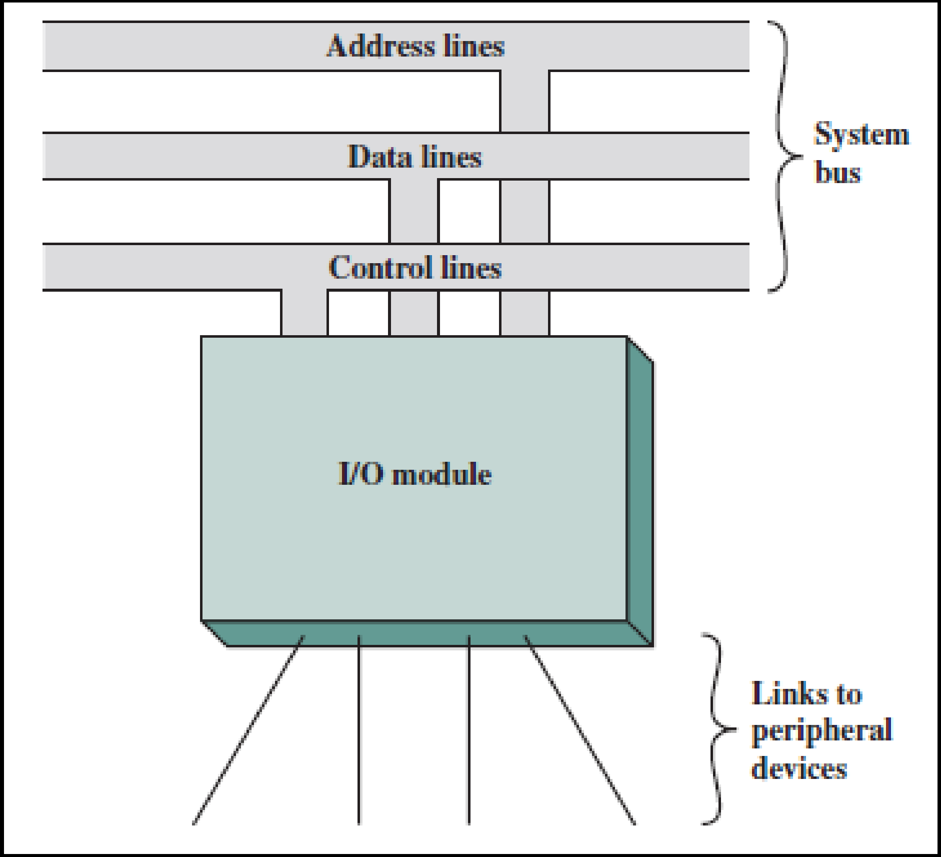
- Why don’t peripheral devices connect directly to the system bus?
  - On the other hand, the data transfer rate of some peripherals is faster than that of the memory or processor.
    - Again, the mismatch would lead to inefficiencies if not managed properly.
  - Peripherals often use different data formats and word lengths than the computer to which they are attached.

Introduction

- Thus, an I/O module is required, with twomajor functions
  - Interface to the processor and memory via the system bus or central switch.
  - Interface to one or more peripheral devices by tailored data links.

Introduction

- Generic Model of an I/O Module



External Devices,

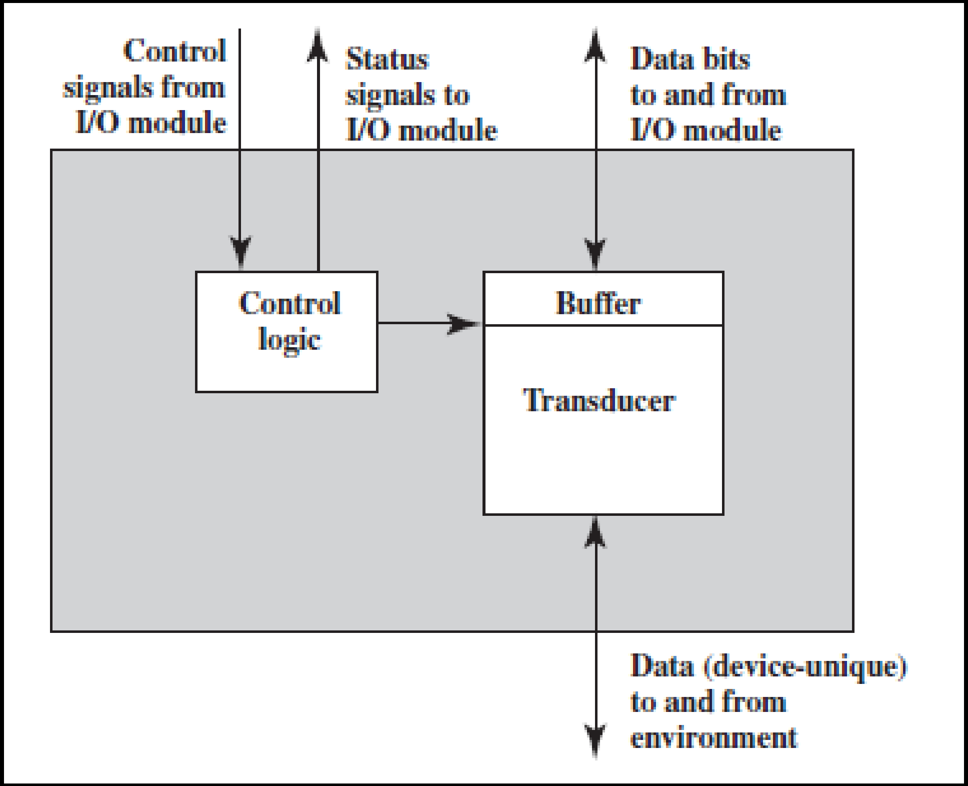
- I/O operations are accomplished through a wide assortment of external devices that provide a means of exchanging data between the external environment and the computer.
  - An external device attaches to the computer by a link to an I/O module
  - The link is used to exchange control, status, and data between the I/O module and the external device.
  - An external device connected to an I/O module is often referred to as a peripheral device or, just a peripheral.

External Devices,

- We classify external devices into 3 broad categories:
  - Human readable: Suitable for communicating with the computer user;
    - E.g. Monitors, printers
  - Machine readable: Suitable for communicating with equipment;
    - Eg sensors, magnetic disks, actuators
  - Communication: Suitable for communicating with remote devices.
    - Eg terminal, machine-readable device, other computers

External Devices: The nature of an external device

- Block Diagram of an External Device


Control signals from I/O module
Status signals to I/O module
Data bits to and from I/O module
Control logic
Buffer
Transducer
Data (device-unique) to and from environment

External Devices: 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,
    - perform some control function particular to the device (e.g., position a disk head)

External Devices: The nature of an external device

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

External Devices: The nature of an external device

- 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
  - buffer size of 8 to 16 bits is common for serial devices, whereas block-oriented devices such as disk drive controllers may have much larger buffers.

External Devices: Keyboard/Monitor

- The most common means of computer/user interaction is a keyboard/monitor arrangement.
  - The user provides input through the keyboard, the input is then transmitted to the computer and may also be displayed on the monitor.
  - In addition, the monitor displays data provided by the computer.
- The basic unit of exchange is the character
  - Associated with each character is a code, typically 7 or 8 bits in length.

External Devices: Keyboard/Monitor

- The most commonly used text code is the International Reference Alphabet (IRA)
  - Each character in this code is represented by a unique 7-bit binary code; thus, 128 different characters can be represented
  - Characters are of two types: printable and control.
    - Printable characters are the alphabetic, numeric, special characters that can be printed on paper or displayed on a screen
    - Some control characters have to do with controlling the printing or displaying of characters; an example is carriage return.

External Devices: Keyboard/Monitor

- 1 text code: International Reference Alphabet (IRA)
  - Each character in this code is represented by a unique 7-bit binary code; thus, 128 different characters can be represented
  - Characters are of two types: printable and control.
    - Printable characters alphabetic, numeric, special characters that can be printed on paper or displayed on a screen
    - Some control characters have to do with controlling the printing or displaying of characters; an example is carriage return.
    - Other control characters are concerned with communications procedures

External Devices: Keyboard/Monitor

- For keyboard input:
  - when the user depresses a key, this generates an electronic signal that is interpreted by the transducer in the keyboard and translated into the bit pattern of the corresponding IRA code.
  - This bit pattern is then transmitted to the I/O module in the computer
  - At the computer, the text can be stored in the same IRA code

External Devices: Keyboard/Monitor

- On output
  - IRA code characters are transmitted to an external device from the I/O module
  - The transducer at the device interprets this code & sends the required electronic signals to the output device either to display the indicated character or perform the requested control function

External Devices: Disk Drive

- A disk drive contains electronics for exchanging data, control, and status signals with an I/O module plus the electronics for controlling the disk read/write mechanism.
- In a fixed-head disk, the transducer is capable of converting between the magnetic patterns on the moving disk surface and bits in the device's buffer
- A moving-head disk must also be able to cause the disk arm to move radially in and out across the disk's surface.

I/O Modules.

- The major functions or requirements for an I/O module fall into the following categories:
  - Control and timing
  - Processor communication
  - Device communication
  - Data buffering
  - Error detection

I/O Modules.

- During any period of time, the processor may communicate with one or more external devices in unpredictable patterns, depending on the program's need for I/O
- The internal resources, such as main memory and the system bus, must be shared among a number of activities, including data I/O.
  - Thus, the I/O function includes a control and timing requirement, to coordinate the flow of traffic between internal resources and external devices

I/O Modules.

- 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.

I/O Modules.

- Processor communication involves the following:
  - 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.

I/O Modules.

- Processor communication involves the following:
  - Address recognition: Just as each word of memory has an address, so does each I/O device.
    - Thus, an I/O module must recognize one unique address for each peripheral it controls
- On the other side, the I/O module must be able to perform device communication.
  - This communication involves commands, status information, and data

I/O Modules.

- An essential task of an I/O module is data buffering
  - Whereas the transfer rate into & out of main memory or the CPU is high, the rate is orders of magnitude lower for many peripheral devices & covers a wide range.
    - Data coming from main memory are sent to an I/O module in a rapid burst. 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
    - Thus , the I/O module must be able to operate at both device and memory speeds

I/O Modules.

- An I/O module is often responsible for error detection and for subsequently reporting errors to the
  - 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.

I/O Modules.

- Some form of error-detecting code is often used to detect transmission errors.
- A simple example is the use of a parity bit on each character of data
  - Eg the IRA character code occupies 7 bits of a byte.
  - The eighth bit is set so that the total number of 1s in the byte is even (even parity) or odd (odd parity).
  - When a byte is received, the I/O module checks the parity to determine whether an error has occurred.

I/O Modules:
I/O Module Structure

- I/O modules vary considerably in complexity and the number of external devices that they control.
- A general description of an I/O modules is as follows:
  - 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

I/O Modules: I/O Module Structure

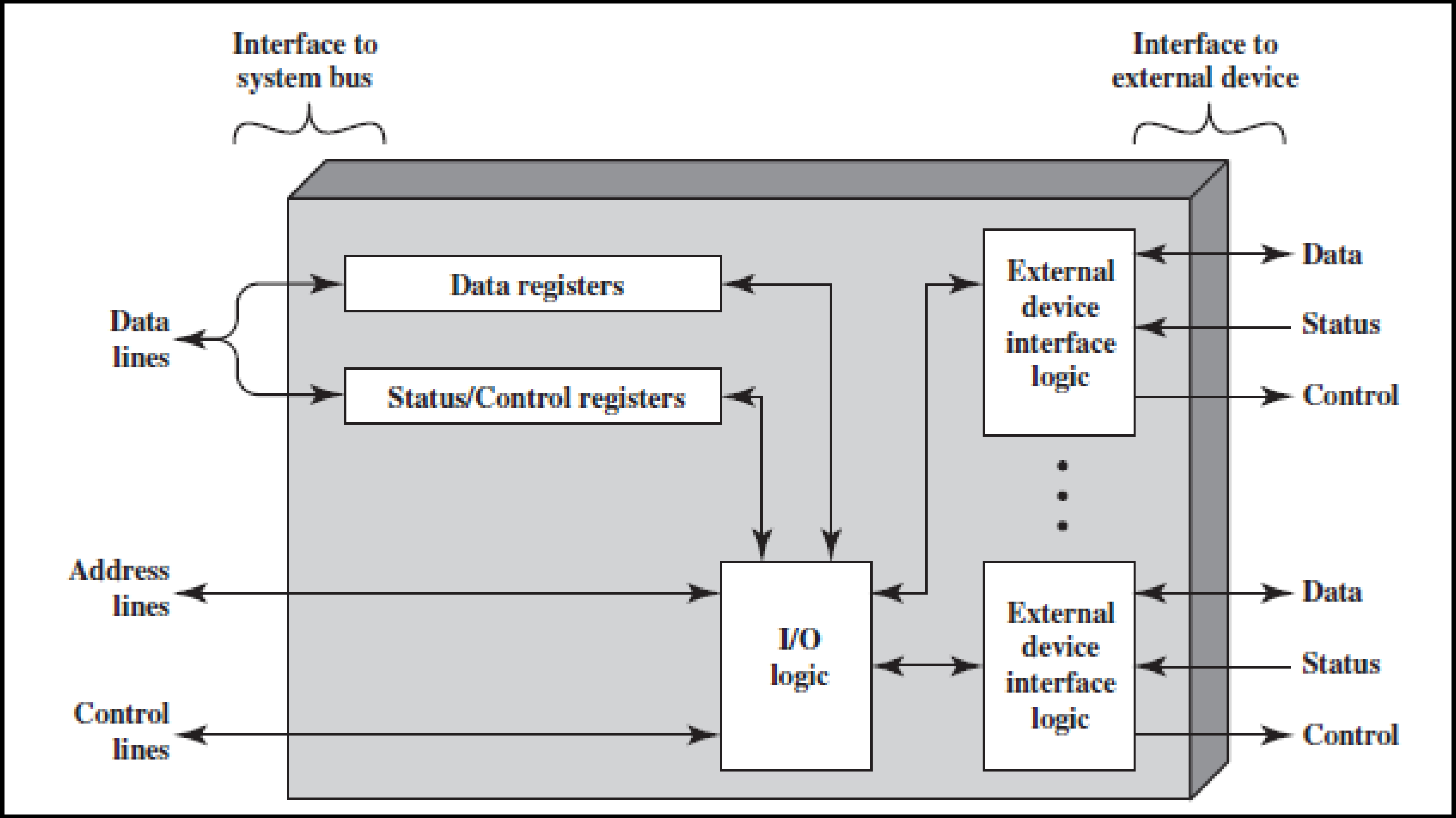
- A general description of an I/O modules is as follows:
  - A status register may also function as a control register, to accept detailed control information from the processor
  - The logic within the module interacts with the processor via a set of control lines.
    - The processor uses the control lines to issue commands to the I/O module

I/O Modules:
I/O Module Structure

- A general description of an I/O modules is as follows:
  - 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 CPU to view a wide range of devices in a simple-minded way, so that the CPU can function with simple read/write, & open/close file commands

I/O Modules:
I/O Module Structure

- Block Diagram of an I/O Module



Programmed I/O,

- Three techniques are possible for I/O operations
  - With 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 waste of processor time

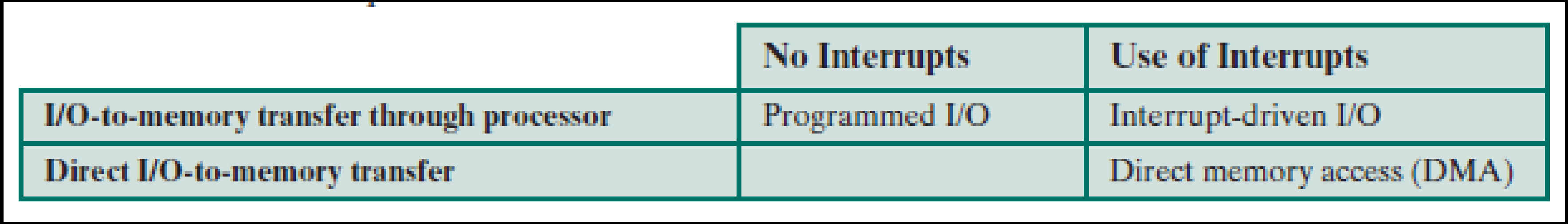
Programmed I/O,

- Three techniques are possible for I/O operations
  - With 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

  - With both programmed and interrupt I/O, the processor is responsible for extracting data from main memory for output and storing data in main memory for input.
    - The alternative is known as direct memory access (DMA).

Programmed I/O

- I/O Techniques



	No Interrupts	Use of Interrupts
I/O-to-memory transfer through processor	Programmed I/O	Interrupt-driven I/O
Direct I/O-to-memory transfer		Direct memory access (DMA)

- When the processor is executing a program and encounters an instruction relating to I/O, it executes that instruction by issuing a command to the appropriate I/O module.

Programmed I/O

- To explain the programmed I/O technique, we view it :
  - first from the point of view of the I/O commands issued by the processor to the I/O module,
  - and then from the point of view of the I/O instructions executed by the processor.

Programmed I/O: I/O Commands

- To execute an I/O-related instruction, the processor issues an address, specifying the particular I/O module and external device, and an I/O command.
- There are 4 types of I/O commands that an I/O module may receive when it is addressed by a CPU:
  - Control: Used to activate a peripheral & tell it what to do.
    - For example, a magnetic-tape unit may be instructed to rewind or to move forward one record.
    - These commands are tailored to the particular type of peripheral device.

Programmed I/O: I/O Commands

- There are 4 types of I/O commands that an I/O module may receive when it is addressed by a CPU:
  - Test: Used to test various status conditions associated with an I/O module and its peripherals.
    - The processor will want to know that the peripheral of interest is powered on and available for use.
    - It will also want to know if the most recent I/O operation is completed and if any errors occurred

Programmed I/O: I/O Commands

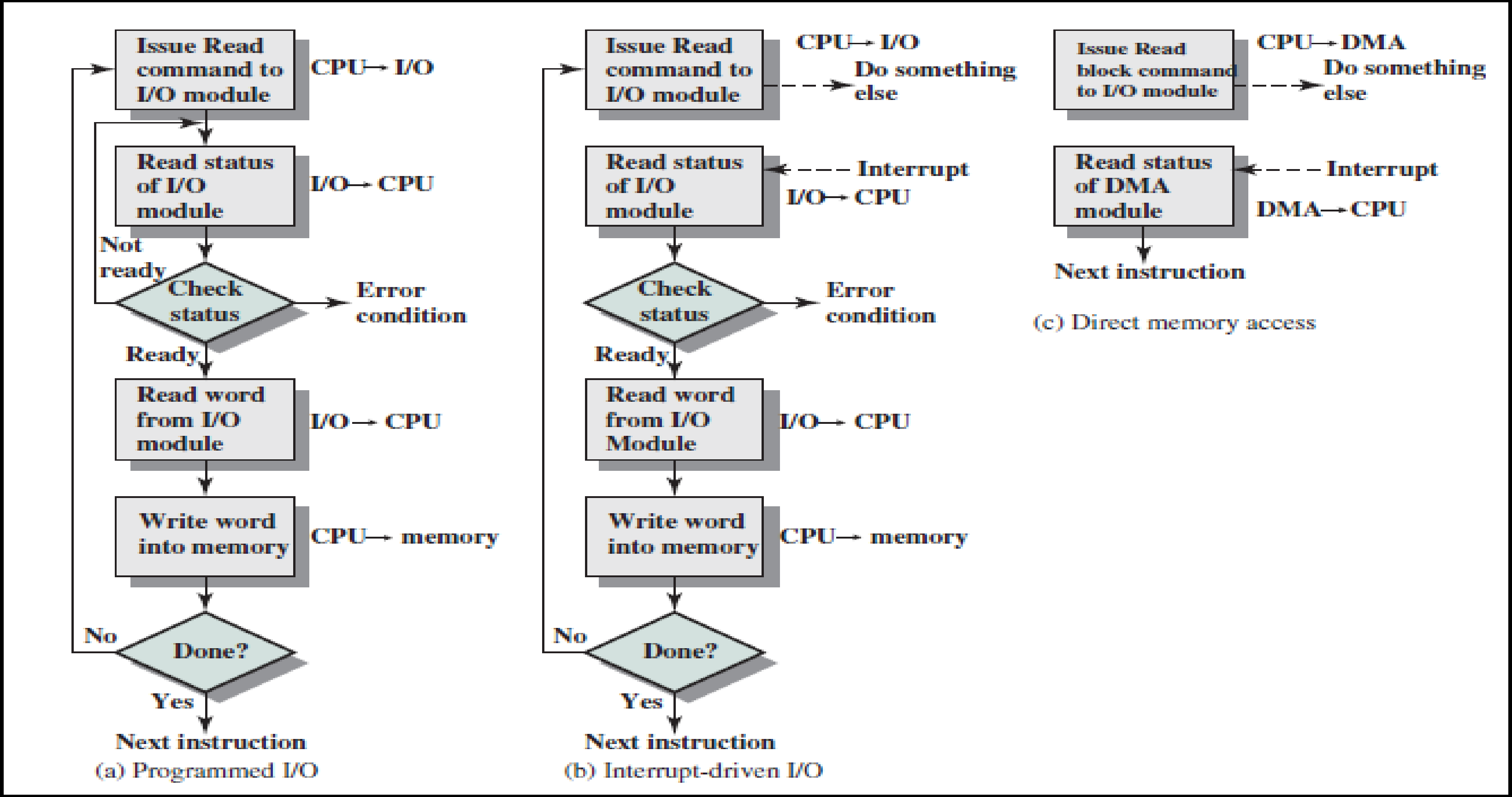
- There are 4 types of I/O commands that an I/O module may receive when it is addressed by a CPU:
  - Read: Causes the I/O module to obtain an item of data from the peripheral and place it in an internal buffer
    - The processor can then obtain the data item by requesting that the I/O module place it on the data bus.
  - Write: Causes the I/O module to take an item of data (byte or word) from the data bus and subsequently transmit that data item to the peripheral.

Programmed I/O: I/O Commands

- 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.
  - The flowchart highlights the main disadvantage of this technique: it is a time- consuming process that keeps the processor busy needlessly.

Programmed I/O: I/O Commands

- Three Techniques for Input of a Block of Data



(a) Programmed I/O (b) Interrupt-driven I/O (c) Direct memory access

Programmed I/O: I/O Instructions

- With programmed I/O, there is a clos correspondence between the I/O-related instructions that the processor fetches from memory and the I/O commands that the processor issues to an I/O module to execute the instructions
  - That is, the instructions are easily mapped into I/O commands, and there is often a simple one-to-one relationship.
  - The form of the instruction depends on the way in which external devices are addressed.

Programmed I/O: I/O Instructions

- Typically, there will be many I/O devices connected through I/O modules to the system.
  - Each device is given a unique identifier or address.
  - When the processor issues an I/O command, the command contains the address of the desired device.
  - Thus, each I/O module must interpret the address lines to determine if the command is for itself.

Programmed I/O: I/O Instructions

- When the CPU, main memory, & I/O share a common bus, 2 modes of addressing are possible:
  - With memory-mapped I/O:
    - there is a single address space for memory locations and I/O devices.
    - The processor treats the status and data registers of I/O modules as memory locations and uses the same machine instructions to access both memory and I/O devices.
    - So, for example, with 10 address lines, a combined total of 210 = 1024 memory locations and I/O addresses can be supported, in any combination

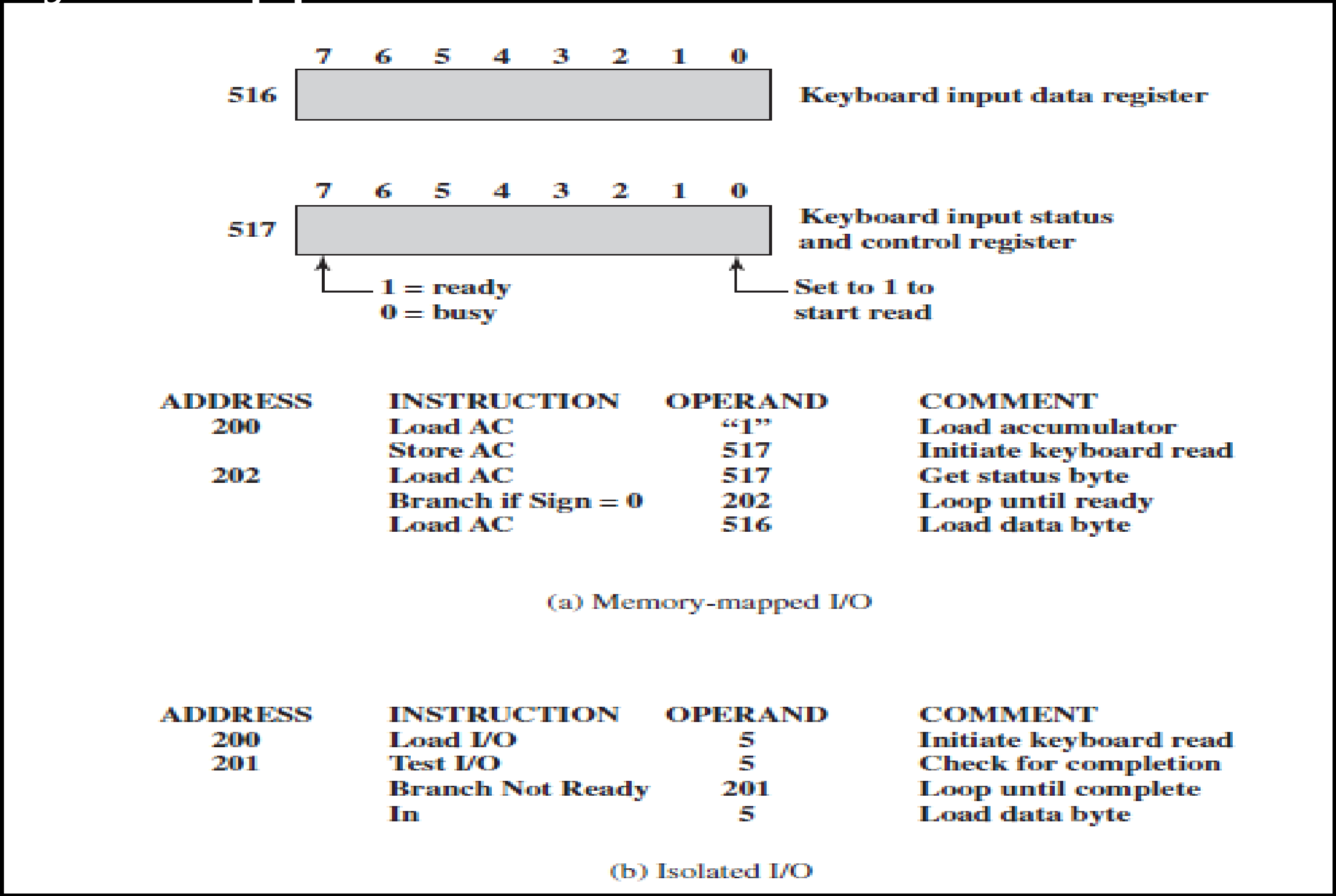
Programmed I/O: I/O Instructions

– isolated I/O:

- With memory-mapped I/O a single read line and a single write line are needed on the bus
- Alternatively, the bus may be equipped with memory read and write plus input and output command lines
- The command line specifies whether the address refers to a memory location or an I/O device.
- The full range of addresses may be available for both.
- Again, with 10 address lines, the system may now support both 1024 memory locations and 1024 I/O addresses.
- Because the address space for I/O is isolated from that for memory, this is referred to as isolated I/O.

Programmed I/O: I/O Instructions

- Memory- Mapped and 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

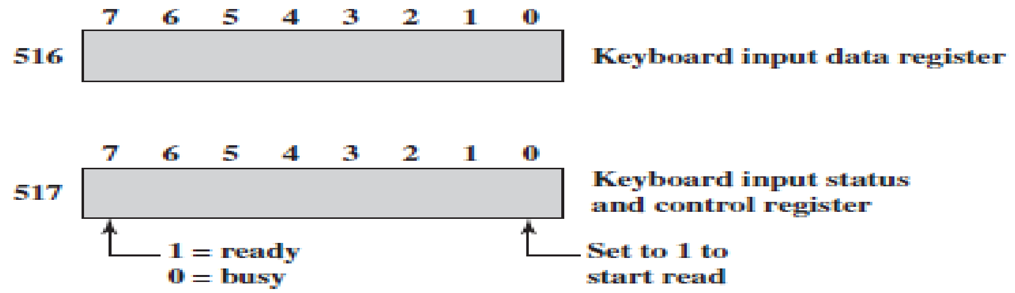
(a) Memory-mapped I/O

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

Programmed I/O:
I/O Instructions

- Memory- Mapped and 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

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

Interrupt driven I/O

- The problem with programmed I/O is that the processor has to wait a long time for the I/O module of concern to be ready for either reception or transmission of data.
  - The CPU, while waiting, must repeatedly interrogate the status of the I/O module.
    - As a result, the level of the performance of the entire system is severely degraded.
- Alternative is for the CPU to send an I/O command to a module then go on to do other useful work

Interrupt driven I/O

- The problem with programmed I/O is that the processor has to wait a long time for the I/O module of concern to be ready for either reception or transmission of data.
  - The CPU, while waiting, must repeatedly interrogate the status of the I/O module.
    - As a result, the level of the performance of the entire system is severely degraded.
- Alternative is for the CPU to send an I/O command to a module then go on to do other useful work

Interrupt driven I/O

- The I/O module will then interrupt the processor to request service when it is ready to exchange data with the processor.
- The processor then executes the data transfer, as before, and then resumes its former processing

Interrupt driven I/O

- from the point of view of the I/O module
  - For input, the I/O module receives a READ command from the processor.
  - The I/O module then proceeds to read data in from an associated peripheral.
  - Once the data are in the module’s data register, the module signals an interrupt to the processor over a control line
    - The module then waits until its data are requested by the processor.

Interrupt driven I/O

- from the point of view of the I/O module
  - When the request is made, the module places its data on the data bus and is then ready for another I/O operation
- From the processor's point of view, the action for input is as follows.
  - The processor issues a READ command.
  - It then goes off and does something else (e.g., the CPU may be working on several different programs at the same time).

Interrupt driven I/O

- From the processor's point of view, the action for input is as follows.
  - At the end of each instruction cycle, the processor checks for interrupts
  - When the interrupt from the I/O module occurs, the processor saves the context (e.g., program counter and processor registers) of the current program and processes the interrupt
  - In this case, the processor reads the word of data from the I/O module and stores it in memory

Interrupt driven I/O

- From the processor's point of view, the action for input is as follows.
  - It then restores the context of the program it was working on (or some other program) and resumes execution.
- Interrupt I/O is more efficient than programmed I/O because it eliminates needless waiting.
  - However, interrupt I/O still consumes a lot of processor time, because every word of data that goes from memory to I/O module or from I/O module to memory must pass through the processor.

Direct Memory Access (DMA)

- Interrupt-driven I/O, though more efficient than simple programmed I/O, still requires the active intervention of the CPU to transfer data between memory & an I/O module, and any data transfer must traverse a path through the CPU.
- Thus, these forms of I/O suffer from two drawbacks:
  - The I/O transfer rate is limited by the speed with which the processor can test and service a device
  - The CPU is tied up in managing an I/O transfer; a # of instructions must be executed for each I/O transfer

Direct Memory Access (DMA)

- Using simple programmed I/O, the processor is dedicated to the task of I/O and can move data at a rather high rate, at the cost of doing nothing else
- Interrupt I/O frees up the processor to some extent at the expense of the I/O transfer rate.
- Nevertheless, both methods have an adverse impact on both processor activity and I/O transfer rate.
- When large volumes of data are to be moved, a more efficient technique is required: direct memory access

Direct Memory Access (DMA) How it works

- It involves an additional module on the system bus.
- The DMA module is capable of mimicking the processor and, indeed, of taking over control of the system from the processor
- It needs to do this to transfer data to and from memory over the system bus.
  - For this purpose, 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
    - Cycle stealing : most common

Direct Memory Access (DMA) How it works

- 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 the following information:
  - Whether a read or write is requested, using the read or write control line between the CPU & the DMA module
  - The address of the I/O device involved, communicated on the data lines.

Direct Memory Access (DMA) How it works

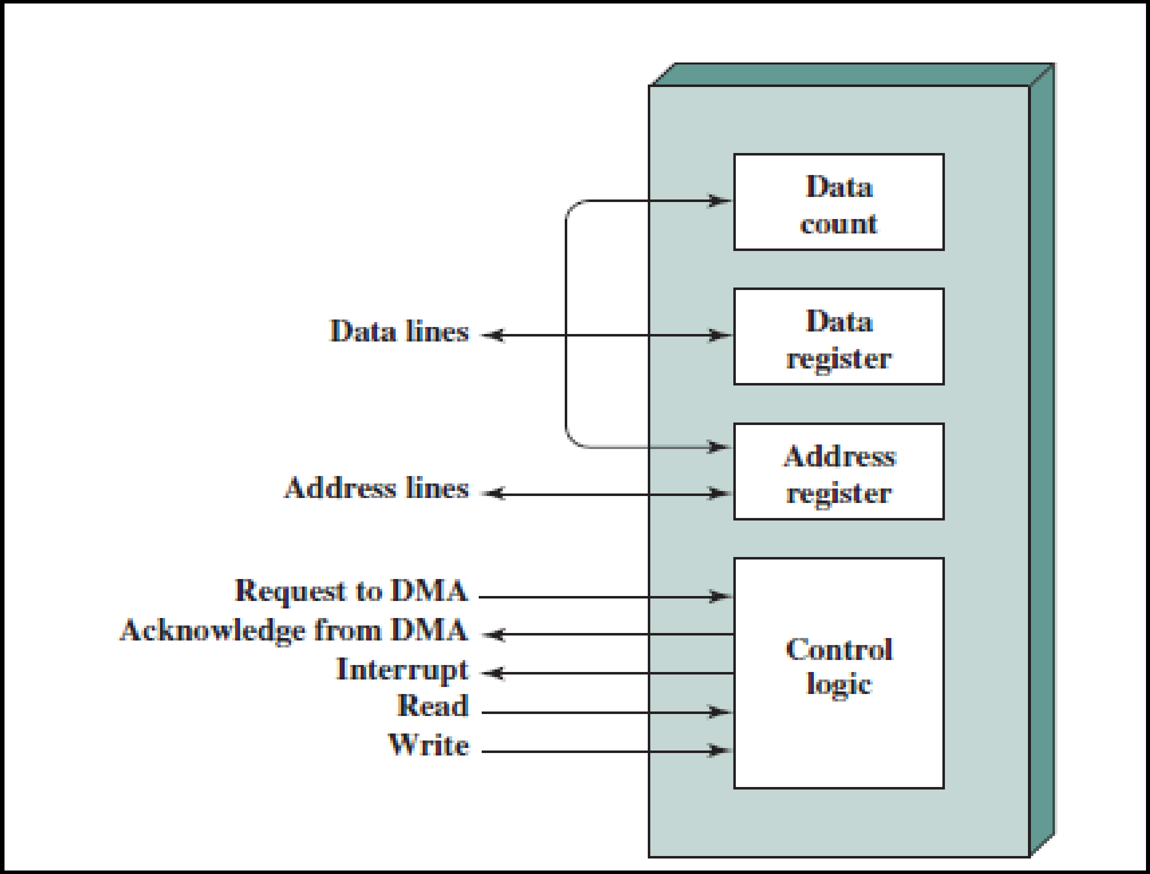
- When the processor wishes to read or write a block of data, it issues a command to the DMA module, by sending to the DMA module the following information:
  - The starting location in memory to read from or write to, communicated on the data lines and stored by the DMA module in its address register.
  - The number of words to be read or written, again communicated via the data lines and stored in the data count register.

Direct Memory Access (DMA) How it works

- The processor then continues with other work. It has delegated this I/O operation to the DMA module
  - The DMA module transfers the entire block of data, one word at a time, directly to or from memory, without going through the processor.
  - When the transfer is complete, the DMA module sends an interrupt signal to the processor.
  - Thus, the processor is involved only at the beginning and end of the transfer

Direct Memory Access (DMA) How it works

- Typical DMA Block Diagram



Direct Memory Access (DMA) How it works

- The DMA mechanism can be configured in a variety of ways.
  - Single-bus, detached DMA:
    - all modules share the same system bus.
    - The DMA module, acting as a surrogate processor, uses programmed I/O to exchange data between memory and an I/O module through the DMA module.
    - This configuration, while it may be inexpensive, is clearly inefficient.
    - As with processor-controlled programmed I/O, each transfer of a word consumes two bus cycles.

Direct Memory Access (DMA) How it works

- The DMA mechanism can be configured in a variety of ways.
  - Single-bus, integrated DMA-I/O
    - The number of required bus cycles can be cut substantially by integrating the DMA and I/O functions
    - this means that there is a path between the DMA module and one or more I/O modules that does not include the system bus
    - The DMA logic may actually be a part of an I/O module, or it may be a separate module that controls one or more I/O modules

Direct Memory Access (DMA) How it works

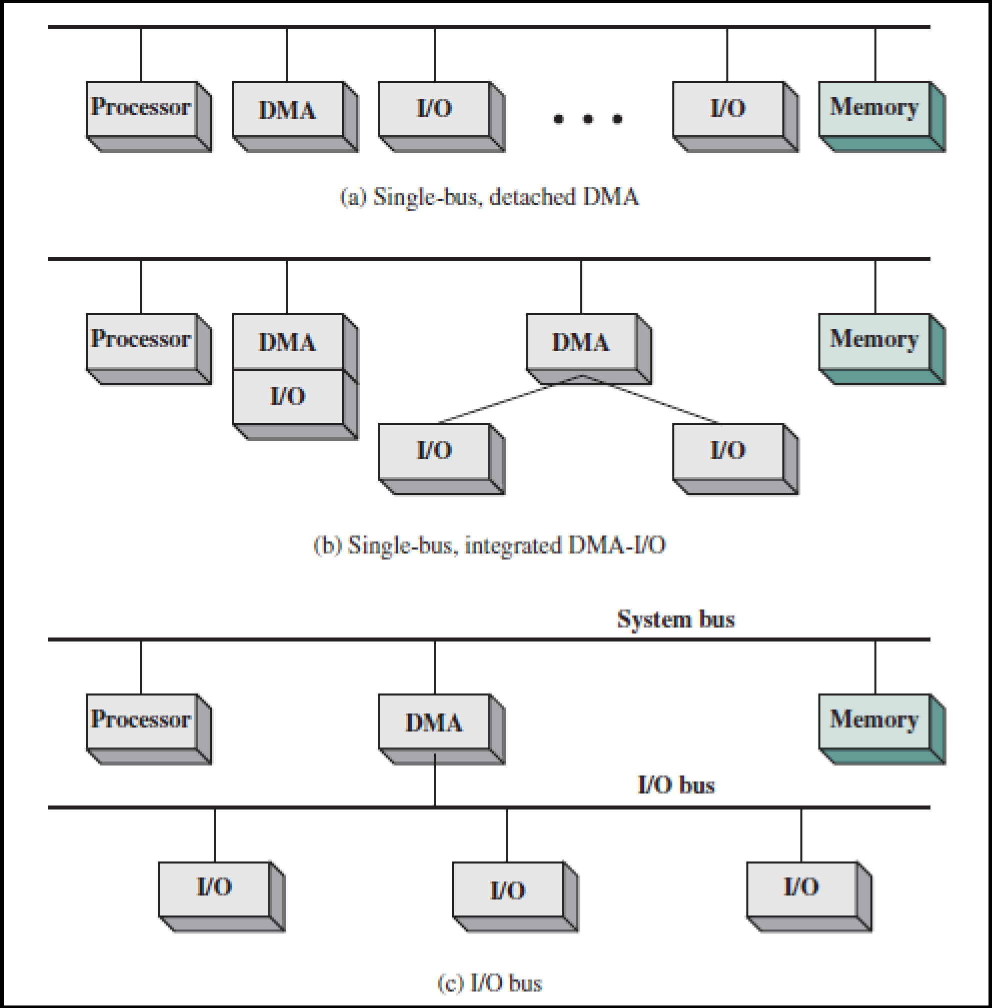
- The DMA mechanism can be configured in a variety of ways.
  - I/O bus
    - This previous concept can be taken one step further by connecting I/O modules to the DMA module using an I/O bus
    - This reduces the number of I/O interfaces in the DMA module to one and provides for an easily expandable configuration

Direct Memory Access (DMA) How it works

- In both of the latter cases, the system bus that the DMA module shares with the processor and memory is used by the DMA module only to exchange data with memory.
  - The exchange of data between the DMA and I/O modules takes place off the system bus.

Direct Memory Access (DMA) How it works

- Alternative DMA Configurations



(a) Single-bus, detached DMA

(b) Single-bus, integrated DMA-I/O

(c) I/O bus

I/O Channel and Processors

- As computer systems have evolved, there has been a pattern of increasing complexity and sophistication of individual components.
- Nowhere is this more evident than in the I/O function.
- The evolutionary steps can be summarized as follows:
  - 1. The CPU directly controls a peripheral device.
    - This is seen in simple microprocessor-controlled devices.

I/O Channel and Processors

- 2. A controller or I/O module is added.
  - The CPU uses programmed I/O without interrupts.
  - With this step, the CPU becomes somewhat divorced from the specific details of external device interfaces.

- 3. The same configuration as in above is used, but now interrupts are employed.
  - The CPU need not spend time waiting for an I/O operation to be performed, thus increasing efficiency.

I/O Channel and Processors

- 4. The I/O module is given direct access to memory via DMA.
  - It can now move a block of data to or from memory without involving the CPU, except at the beginning and end of the transfer.
- 5. The I/O module is enhanced to become a processor in its own right, with a specialized instruction set tailored for I/O
  - The CPU directs the I/O processor to execute an I/O program in memory.
  - The I/O processor fetches and executes these instructions without CPU intervention

I/O Channel and Processors

– 6. The I/O module has a local memory of its own and is, in fact, a computer in its own right.
    - With this architecture, a large set of I/O devices can be controlled, with minimal CPU involvement.
    - A common use for such an architecture has been to control communication with interactive terminals.
    - The I/O processor takes care of most of the tasks involved in controlling the terminals

I/O Channel and Processors

- As one proceeds along this evolutionary path, more and more of the I/O function is performed without CPU involvement.
  - The CPU is increasingly relieved of I/O-related tasks, improving performance

- With steps 5-6, a major change occurs with the introduction of the concept of an I/O module capable of executing a program.

I/O Channel and Processors

- For step 5, the I/O module is often referred to as an I/O channel

- For step 6, the term I/O processor is often used

I/O Channel and Processors: Characteristics of I/O Channels

- The I/O channel represents an extension of the DMA concept.
- An I/O channel has the ability to execute I/O instructions, which gives it complete control over I/O operations
  - In a computer system with such devices, the CPU does not execute I/O instructions.
  - Such instructions are stored in main memory to be executed by a special-purpose processor in the I/O channel itself

I/O Channel and Processors: Characteristics of I/O Channels

– Thus, the CPU initiates an I/O transfer by instructing the I/O channel to execute a program in memory.

– The program will specify the device or devices, the area or areas of memory for storage, priority, and actions to be taken for certain error conditions.

– The I/O channel follows these instructions and controls the data transfer.

I/O Channel and Processors: Characteristics of I/O Channels

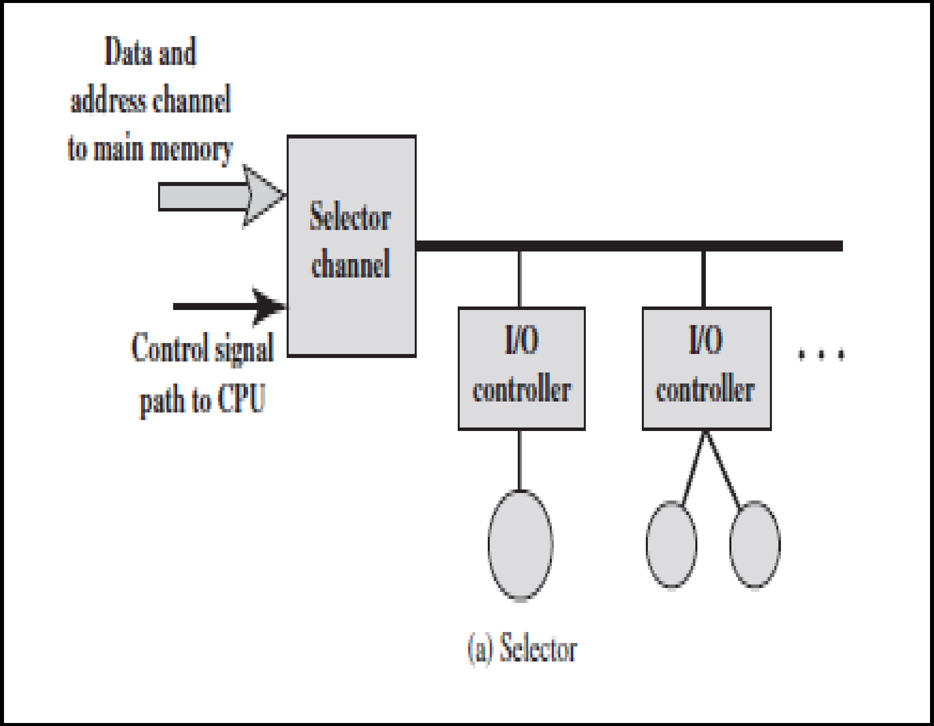
- Two types of I/O channels are common:
  - A selector channel controls multiple high-speed devices and, at any one time, is dedicated to the transfer of data with one of those devices
    - Thus, the I/O channel selects one device and effects the data transfer
    - Each device, or a small set of devices, is handled by a controller, or I/O module

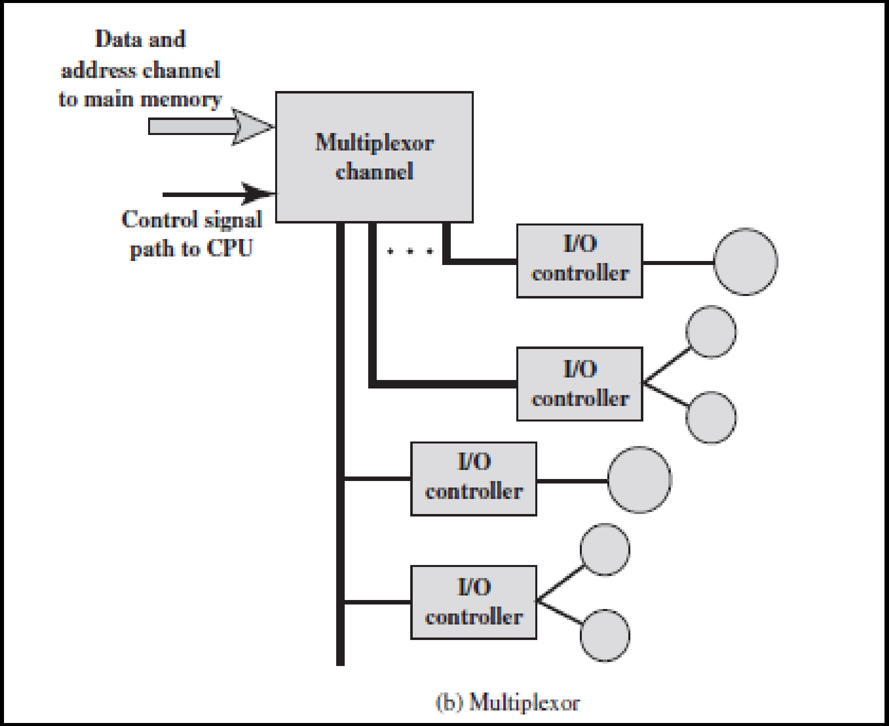
I/O Channel and Processors: Characteristics of I/O Channels

- Two types of I/O channels are common:
  - 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
    - e.g., the resultant character stream from three devices with different rates and individual streams A1A2A3A4….., B1B2B3B4….., and C1C2C3C4……. might be A1B1C1A2C2A3B2C3A4, and so on.
    - For high-speed devices, a block multiplexor interleaves blocks of data from several devices

I/O Channel and Processors: Characteristics of I/O Channels

- I/O Channel Architecture


Data and address channel to main memory
Selector channel
Control signal path to CPU
I/O controller
I/O controller
. . .
(a) Selector



(b) Multiplexor

THE END

- **NEXT TOPIC: COMPUTER ARITHMETIC**