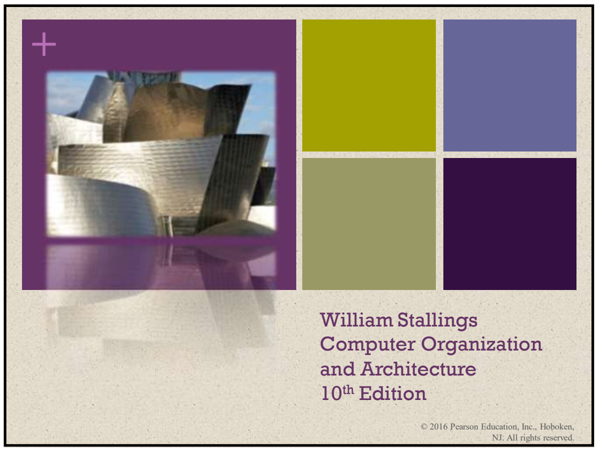
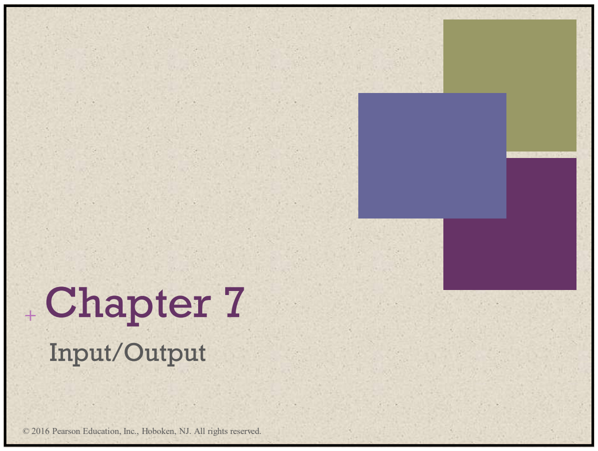


William Stallings
Computer Organization
and Architecture
10th Edition

1



Chapter 7

Input/Output

2

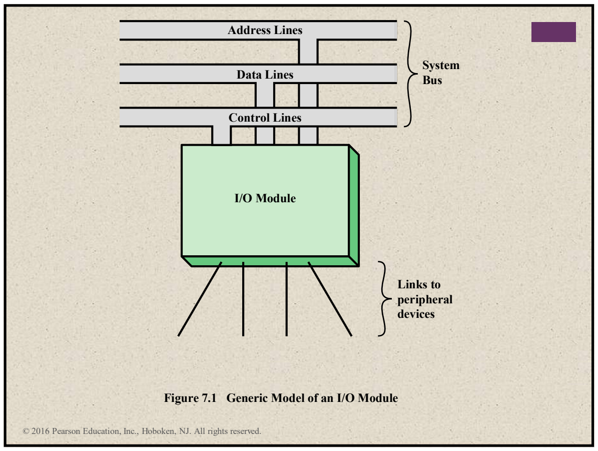
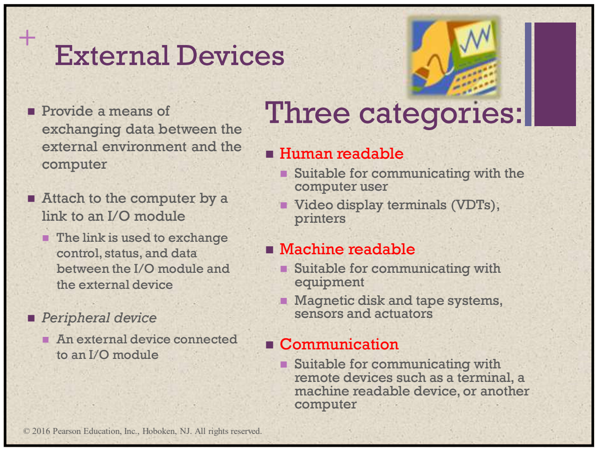


Figure 7.1 Generic Model of an I/O Module

3



External Devices

- Provide a means of exchanging data between the external environment and the computer
- Attach 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
- *Peripheral device*
  - An external device connected to an I/O module

Three categories:

- Human readable
  - Suitable for communicating with the computer user
  - Video display terminals (VDTs), printers
- Machine readable
  - Suitable for communicating with equipment
  - Magnetic disk and tape systems, sensors and actuators
- Communication
  - Suitable for communicating with remote devices such as a terminal, a machine readable device, or another computer

4

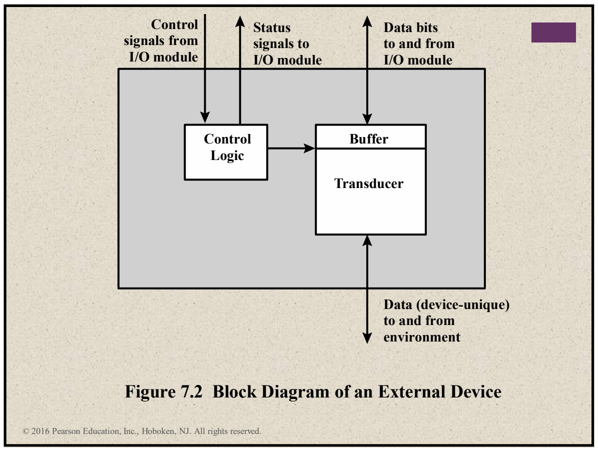
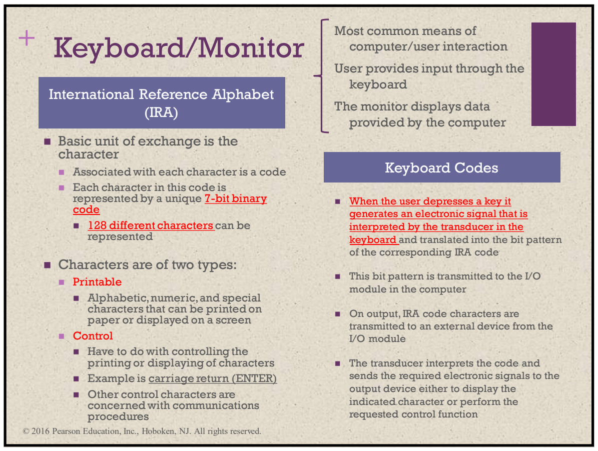

Control signals from I/O module
Status signals to I/O module
Data bits to and from I/O module
Control Logic
Buffer
Transducer
Data (device-unique) to and from environment

Figure 7.2 Block Diagram of an External Device

5



Keyboard/Monitor

- Most common means of computer/user interaction
- User provides input through the keyboard
- The monitor displays data provided by the computer

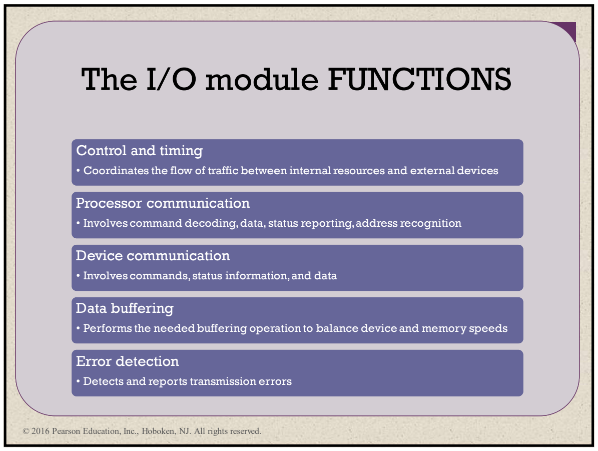
International Reference Alphabet (IRA)

- Basic unit of exchange is the character
  - Associated with each character is a code
  - Each character in this code is represented by a unique 7-bit binary code
    - 128 different characters can be represented
- Characters are of two types:
  - Printable
    - Alphabetic, numeric, and special characters that can be printed on paper or displayed on a screen
  - Control
    - Have to do with controlling the printing or displaying of characters
    - Example is carriage return (ENTER)
    - Other control characters are concerned with communications procedures

Keyboard Codes

- When the user depresses a key it 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 transmitted to the I/O module in the computer
- On output, IRA code characters are transmitted to an external device from the I/O module
- The transducer interprets the code and sends the required electronic signals to the output device either to display the indicated character or perform the requested control function

6



The I/O module FUNCTIONS

Control and timing
- Coordinates the flow of traffic between internal resources and external devices

Processor communication
- Involves command decoding, data, status reporting, address recognition

Device communication
- Involves commands, status information, and data

Data buffering
- Performs the needed buffering operation to balance device and memory speeds

Error detection
- Detects and reports transmission errors

7

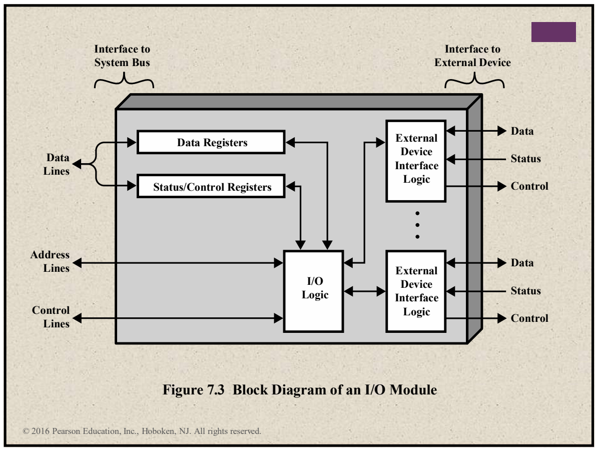
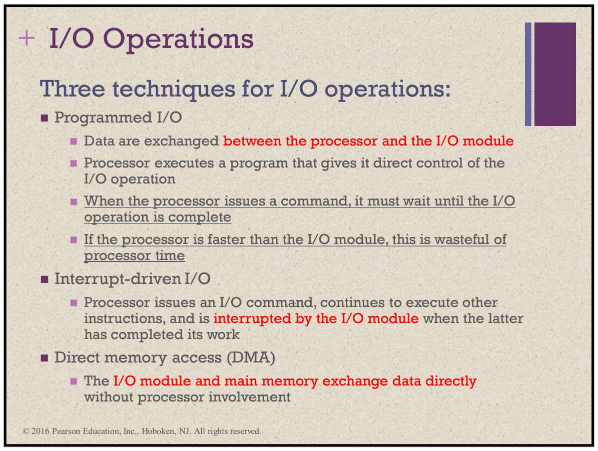


Figure 7.3 Block Diagram of an I/O Module

8

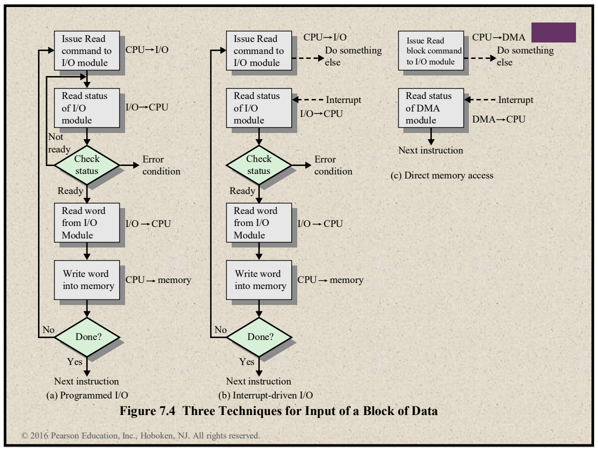


+ I/O Operations

Three techniques for I/O operations:

- Programmed I/O
  - Data are exchanged between the processor and the I/O module
  - Processor executes a program that gives it direct control of the I/O operation
  - When the processor issues a command, 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
- Interrupt-driven I/O
  - 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
- Direct memory access (DMA)
  - The I/O module and main memory exchange data directly without processor involvement

9



(a) Programmed I/O

(b) Interrupt-driven I/O

(c) Direct memory access

Figure 7.4 Three Techniques for Input of a Block of Data

10

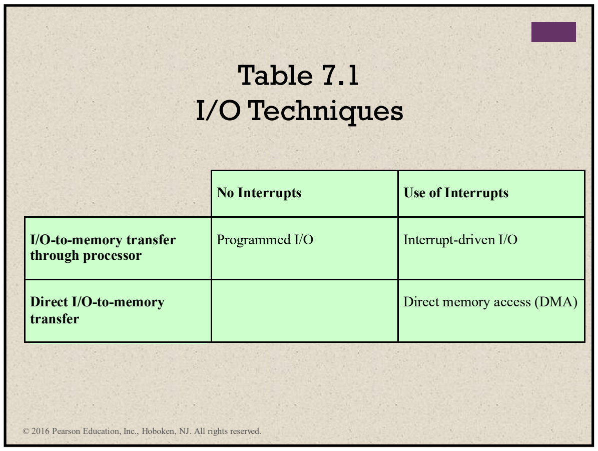
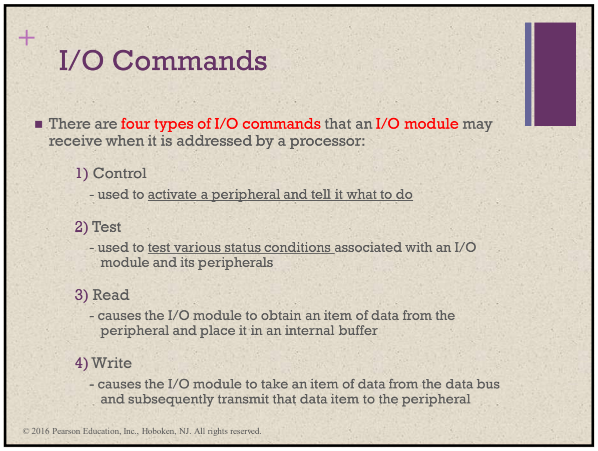


Table 7.1 I/O Techniques

	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)

11



+ I/O Commands

- There are four types of I/O commands that an I/O module may receive when it is addressed by a processor:

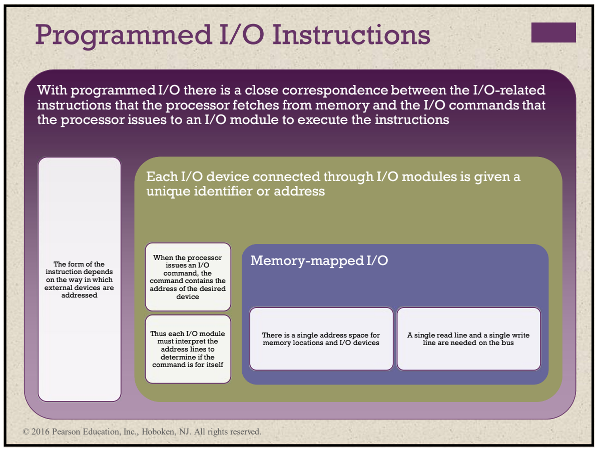
  1) Control
  - used to activate a peripheral and tell it what to do

  2) Test
  - used to test various status conditions associated with an I/O module and its peripherals

  3) Read
  - causes the I/O module to obtain an item of data from the peripheral and place it in an internal buffer

  4) Write
  - causes the I/O module to take an item of data from the data bus and subsequently transmit that data item to the peripheral

12



Programmed I/O Instructions

With programmed I/O there is a close 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

The form of the instruction depends on the way in which external devices are addressed

Each I/O device connected through I/O modules is given a unique identifier or address

When the processor issues an I/O command, the command contains the address of the desired device

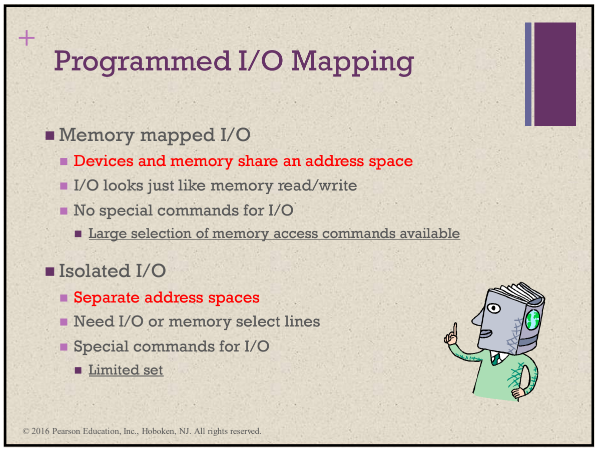
Thus each I/O module must interpret the address lines to determine if the command is for itself

Memory-mapped I/O

There is a single address space for memory locations and I/O devices

A single read line and a single write line are needed on the bus

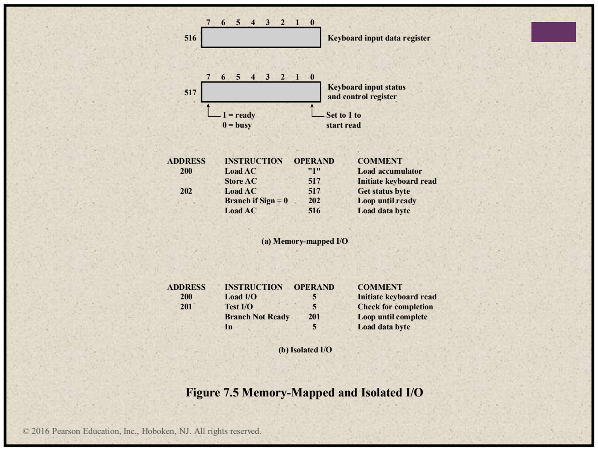
13



Programmed I/O Mapping

- Memory mapped I/O
  - Devices and memory share an address space
  - I/O looks just like memory read/write
  - No special commands for I/O
    - Large selection of memory access commands available
- Isolated I/O
  - Separate address spaces
  - Need I/O or memory select lines
  - Special commands for I/O
    - Limited set

14



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

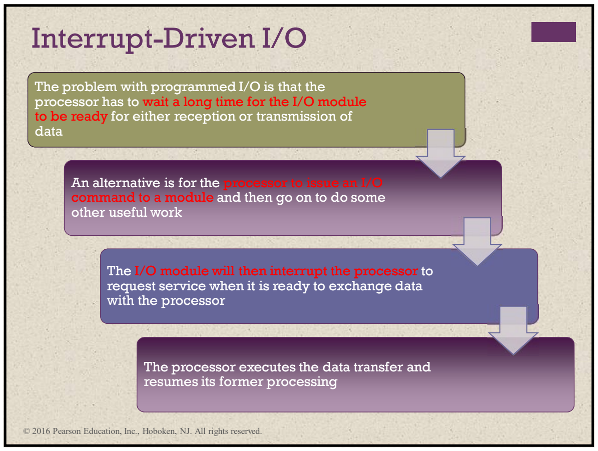
(a) Memory-mapped I/O

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

Figure 7.5 Memory-Mapped and Isolated I/O

15



Interrupt-Driven I/O

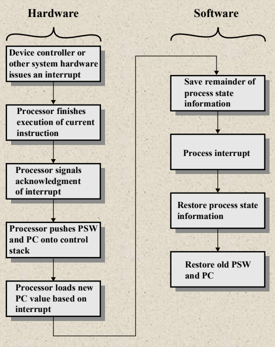
The problem with programmed I/O is that the processor has to wait a long time for the I/O module to be ready for either reception or transmission of data

An alternative is for the processor to issue an I/O command to a module and then go on to do some other useful work

The I/O module will then interrupt the processor to request service when it is ready to exchange data with the processor

The processor executes the data transfer and resumes its former processing

16



Hardware

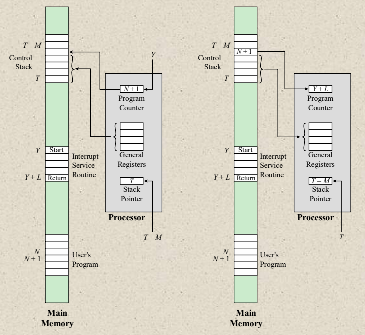
- Device controller or other system hardware issues an interrupt
- Processor finishes execution of current instruction
- Processor signals acknowledgment of interrupt
- Processor pushes PSW and PC onto control stack
- Processor loads new PC value based on interrupt

Software

- Save remainder of process state information
- Process interrupt
- Restore process state information
- Restore old PSW and PC

Figure 7.6 Simple Interrupt Processing

17



(a) Interrupt occurs after instruction at location N

(b) Return from interrupt

Figure 7.7 Changes in Memory and Registers for an Interrupt

18

Interrupt Driven I/O: Design Issues

Two design issues for Interrupt Driven I/O:

- Because there will be multiple I/O modules, how does the processor determine which device issued the interrupt?
- If multiple interrupts have occurred, how does the processor decide which one to process?

19

Device Identification

Four general categories of techniques :

- **Multiple interrupt lines**
  - Between the processor and the I/O modules
  - Most straightforward approach to the problem
  - Consequently even if multiple lines are used, it is likely that each line will have multiple I/O modules attached to it
- **Software poll**
  - When processor detects an interrupt it branches to an interrupt-service routine whose job is to poll each I/O module to determine which module caused the interrupt
  - Time consuming
- **Daisy chain (hardware poll, vectored)**
  - The interrupt acknowledge line is daisy chained through the modules
  - Vector – address of the I/O module or some other unique identifier
  - Vectored interrupt – processor uses the vector as a pointer to the appropriate device-service routine, avoiding the need to execute a general interrupt-service routine first
- **Bus arbitration (İsteğe Bağlı) (vectored)**
  - An I/O module must first gain control of the bus before it can raise the interrupt request line
  - When the processor detects the interrupt it responds on the interrupt acknowledge line
  - Then the requesting module places its vector on the data lines

20

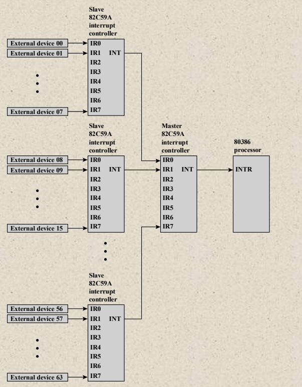


Figure 7.8 Use of the 82C59A Interrupt Controller

21

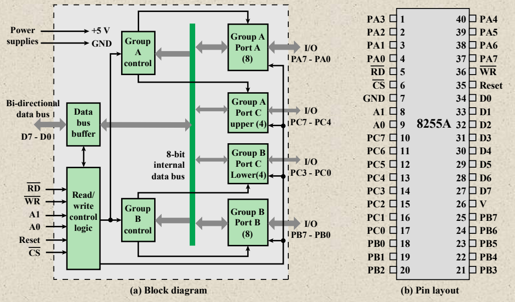
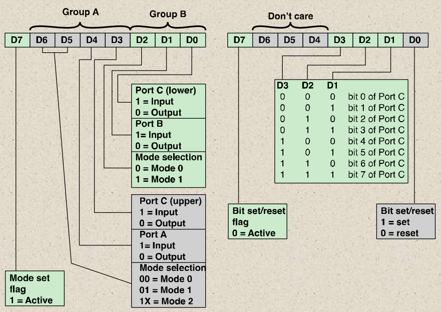


Figure 7.9 The Intel 8255A Programmable Peripheral Interface

22



D3	D2	D1	
0	0	0	bit 0 of Port C
0	0	1	bit 1 of Port C
0	1	0	bit 2 of Port C
0	1	1	bit 3 of Port C
1	0	0	bit 4 of Port C
1	0	1	bit 5 of Port C
1	1	0	bit 6 of Port C
1	1	1	bit 7 of Port C

(a) Mode definition of the 8255 control register to configure the 8255

(b) Bit definitions of the 8255 control register to modify single bits of port C

Figure 7.10 The Intel 8255A Control Word

23

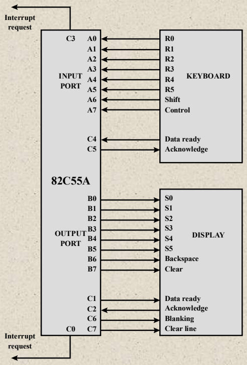
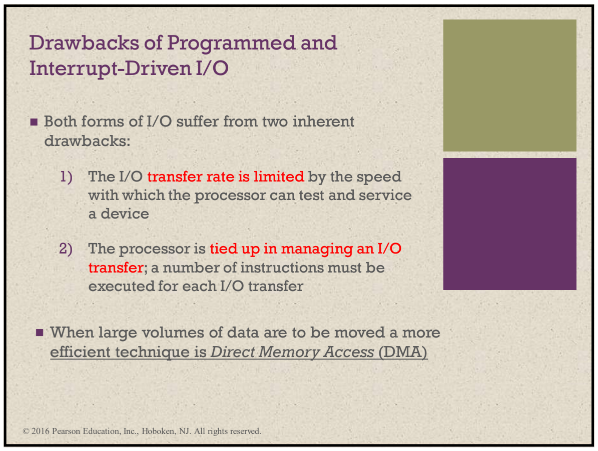


Figure 7.11 Keyboard/Display Interface to 82C55A

24



Drawbacks of Programmed and Interrupt-Driven I/O

- Both forms of I/O suffer from two inherent drawbacks:
  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 instructions must be executed for each I/O transfer
- When large volumes of data are to be moved a more efficient technique is *Direct Memory Access* (DMA)

25

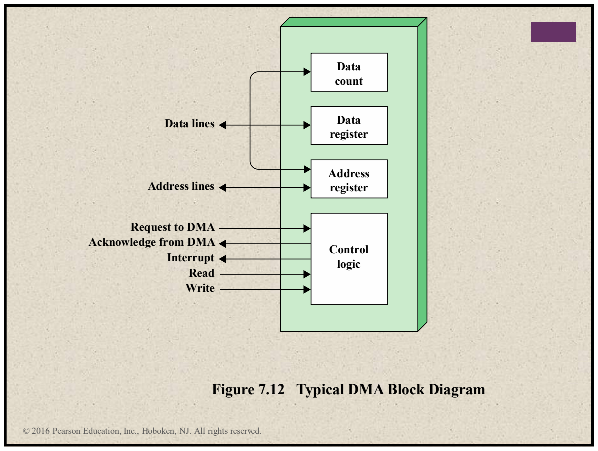


Figure 7.12 Typical DMA Block Diagram

26

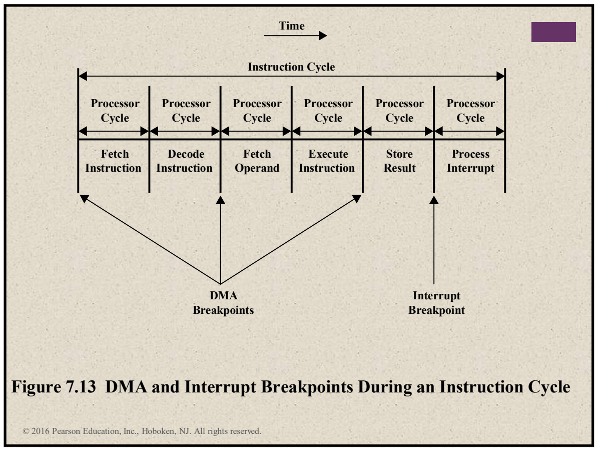


Figure 7.13 DMA and Interrupt Breakpoints During an Instruction Cycle

27

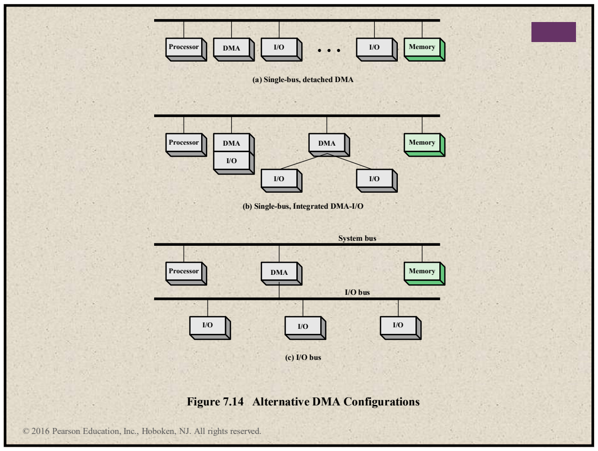
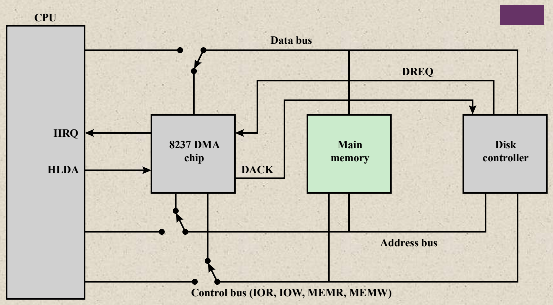


Figure 7.14 Alternative DMA Configurations

28



DACK = DMA acknowledge
DREQ = DMA request
HLDA = HOLD acknowledge
HRQ = HOLD request

Figure 7.15 8237 DMA Usage of System Bus

29

Fly-By DMA Controller

Data does not pass through and is not stored in DMA chip

- DMA only between I/O port and memory
- Not between two I/O ports or two memory locations

Can do memory to memory via register

8237 contains four DMA channels

- Programmed independently
- Any one active
- Numbered 0, 1, 2, and 3

30

Bit	Command	Status	Mode	Single Mask	All Mask
D0	Memory-to-memory E/D	Channel 0 has reached TC	Channel select	Select channal mask bit	Clear/set channel 0 mask bit
D1	Channel 0 address hold E/D	Channel 1 has reached TC			Clear/set channel 1 mask bit
D2	Controller E/D	Channel 2 has reached TC	Verify/write/read transfer	Clear/set mask bit	Clear/set channel 2 mask bit
D3	Normal/compressed timing	Channel 3 has reached TC		Not used	Clear/set channel 3 mask bit
D4	Fixed/rotating priority	Channel 0 request	Auto-initialization E/D		Not used
D5	Late/extended write selection	Channel 0 request	Address increment/decrement select		
D6	DREQ sense active high/low	Channel 0 request			
D7	DACK sense active high/low	Channel 0 request	Demand/single/block/cascade mode select		

E/D = enable/disable
TC = terminal count

Table 7.2

Intel 8237A Registers

31

Direct Cache Access (DCA)

- DMA is not able to scale to meet the increased demand due to dramatic increases in data rates for network I/O
- Demand is coming primarily from the widespread deployment of 10-Gbps and 100-Gbps Ethernet switches to handle massive amounts of data transfer to and from database servers and other high-performance systems
- Another source of traffic comes from Wi-Fi in the gigabit range
- Network Wi-Fi devices that handle 3.2 Gbps and 6.76 Gbps are becoming widely available and producing demand on enterprise systems

32

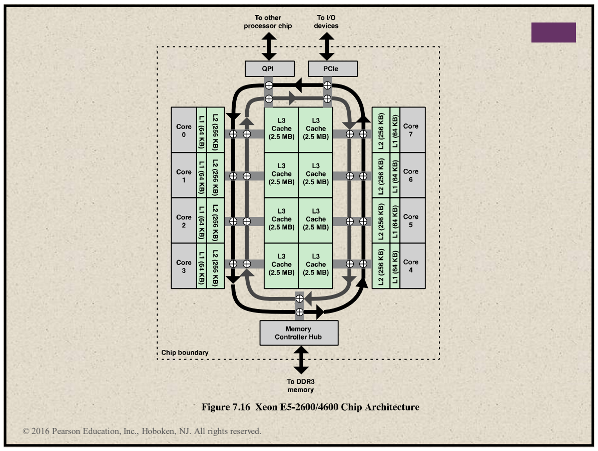
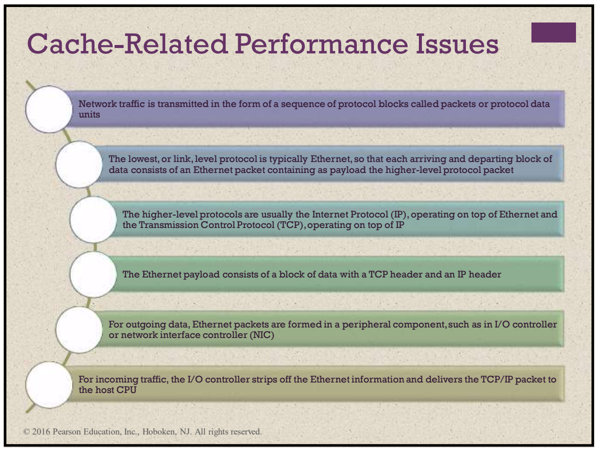


Figure 7.16 Xeon E5-2600/4600 Chip Architecture

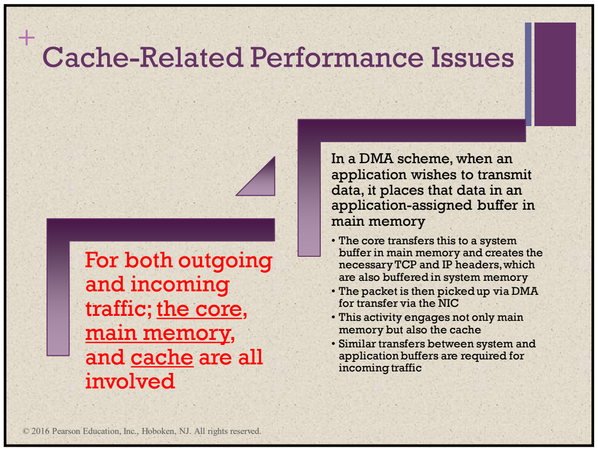
33



Cache-Related Performance Issues

- Network traffic is transmitted in the form of a sequence of protocol blocks called packets or protocol data units
- The lowest, or link, level protocol is typically Ethernet, so that each arriving and departing block of data consists of an Ethernet packet containing as payload the higher-level protocol packet
- The higher-level protocols are usually the Internet Protocol (IP), operating on top of Ethernet and the Transmission Control Protocol (TCP), operating on top of IP
- The Ethernet payload consists of a block of data with a TCP header and an IP header
- For outgoing data, Ethernet packets are formed in a peripheral component, such as in I/O controller or network interface controller (NIC)
- For incoming traffic, the I/O controller strips off the Ethernet information and delivers the TCP/IP packet to the host CPU

34



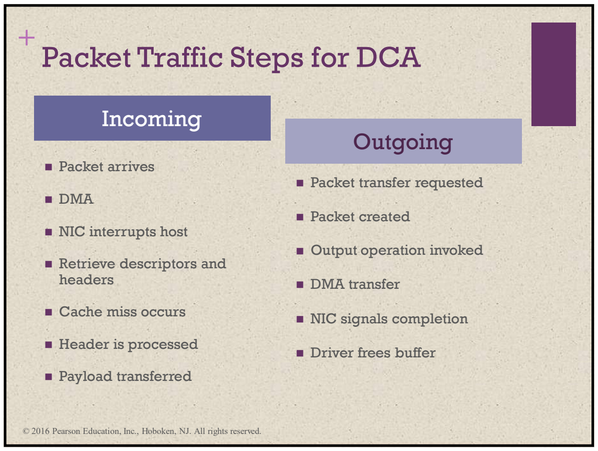
Cache-Related Performance Issues

For both outgoing and incoming traffic; the core, main memory, and cache are all involved

In a DMA scheme, when an application wishes to transmit data, it places that data in an application-assigned buffer in main memory

- The core transfers this to a system buffer in main memory and creates the necessary TCP and IP headers, which are also buffered in system memory
- The packet is then picked up via DMA for transfer via the NIC
- This activity engages not only main memory but also the cache
- Similar transfers between system and application buffers are required for incoming traffic

35



Packet Traffic Steps for DCA

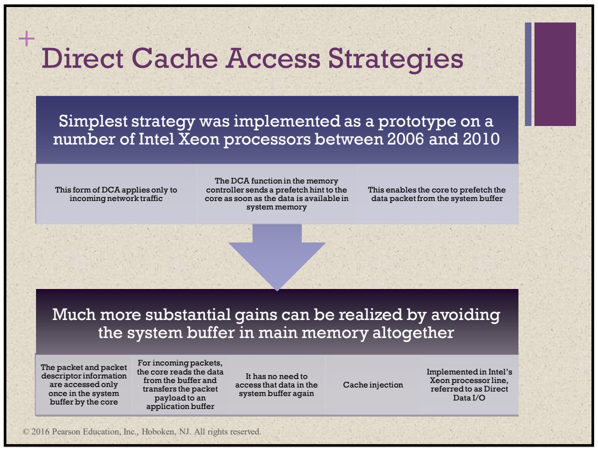
Incoming

- Packet arrives
- DMA
- NIC interrupts host
- Retrieve descriptors and headers
- Cache miss occurs
- Header is processed
- Payload transferred

Outgoing

- Packet transfer requested
- Packet created
- Output operation invoked
- DMA transfer
- NIC signals completion
- Driver frees buffer

36



Direct Cache Access Strategies

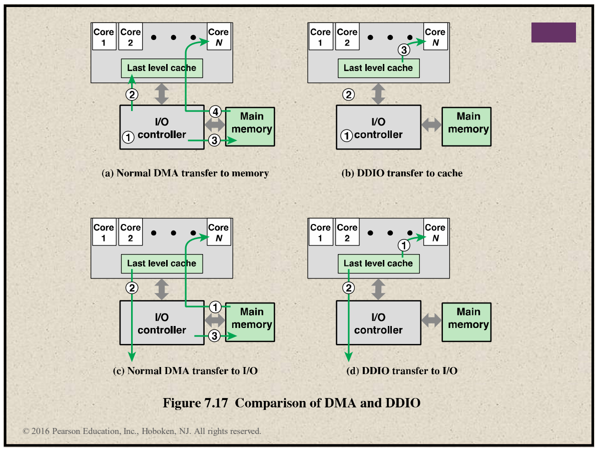
Simplest strategy was implemented as a prototype on a number of Intel Xeon processors between 2006 and 2010

- This form of DCA applies only to incoming network traffic
- The DCA function in the memory controller sends a prefetch hint to the core as soon as the data is available in system memory
- This enables the core to prefetch the data packet from the system buffer

Much more substantial gains can be realized by avoiding the system buffer in main memory altogether

- The packet and packet descriptor information are accessed only once in the system buffer by the core
- For incoming packets, the core reads the data from the buffer and transfers the packet payload to an application buffer
- It has no need to access that data in the system buffer again
- Cache injection
- Implemented in Intel's Xeon processor line, referred to as Direct Data I/O

37



(a) Normal DMA transfer to memory

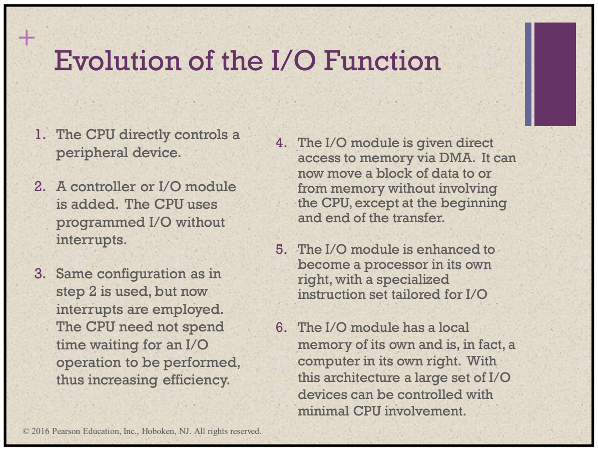
(b) DDIO transfer to cache

(c) Normal DMA transfer to I/O

(d) DDIO transfer to I/O

Figure 7.17 Comparison of DMA and DDIO

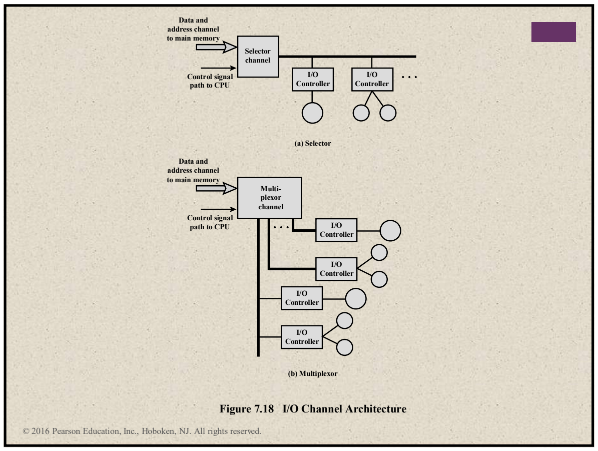
38



Evolution of the I/O Function

1. The CPU directly controls a peripheral device.
2. A controller or I/O module is added. The CPU uses programmed I/O without interrupts.
3. Same configuration as in step 2 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.
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
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.

39

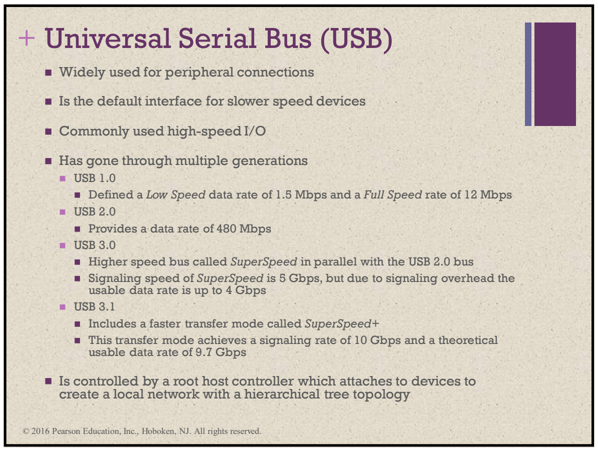

Data and address channel to main memory
Selector channel
Control signal path to CPU
I/O Controller
I/O Controller
. . .
Data and address channel to main memory
Multi-plexor channel
Control signal path to CPU
. . .
I/O Controller
I/O Controller
I/O Controller
I/O Controller

(a) Selector

(b) Multiplexor

Figure 7.18 I/O Channel Architecture

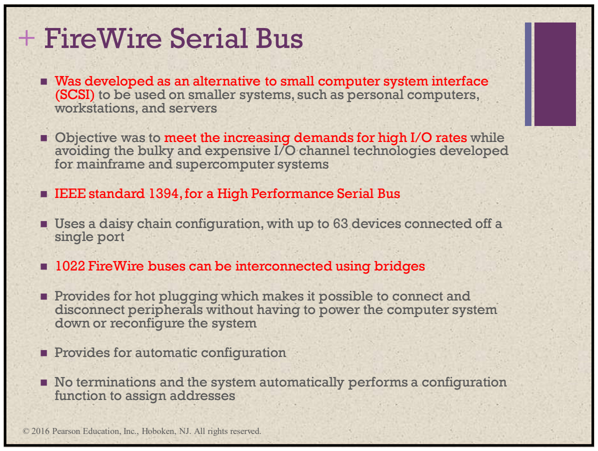
40



+ Universal Serial Bus (USB)

- Widely used for peripheral connections
- Is the default interface for slower speed devices
- Commonly used high-speed I/O
- Has gone through multiple generations
  - USB 1.0
    - Defined a *Low Speed* data rate of 1.5 Mbps and a *Full Speed* rate of 12 Mbps
  - USB 2.0
    - Provides a data rate of 480 Mbps
  - USB 3.0
    - Higher speed bus called *SuperSpeed* in parallel with the USB 2.0 bus
    - Signaling speed of *SuperSpeed* is 5 Gbps, but due to signaling overhead the usable data rate is up to 4 Gbps
  - USB 3.1
    - Includes a faster transfer mode called *SuperSpeed+*
    - This transfer mode achieves a signaling rate of 10 Gbps and a theoretical usable data rate of 9.7 Gbps
- Is controlled by a root host controller which attaches to devices to create a local network with a hierarchical tree topology

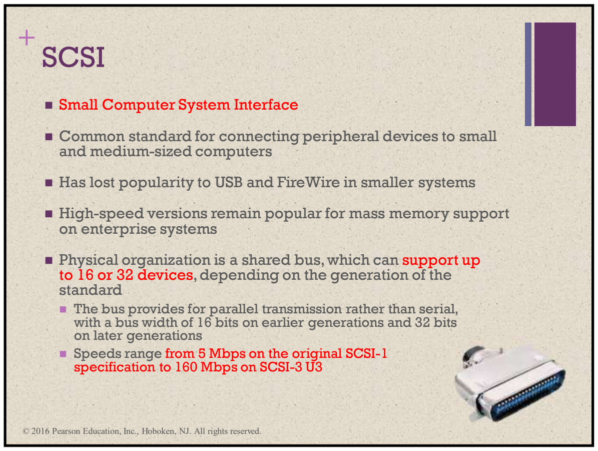
41



+ FireWire Serial Bus

- Was developed as an alternative to small computer system interface (SCSI) to be used on smaller systems, such as personal computers, workstations, and servers
- Objective was to meet the increasing demands for high I/O rates while avoiding the bulky and expensive I/O channel technologies developed for mainframe and supercomputer systems
- IEEE standard 1394, for a High Performance Serial Bus
- Uses a daisy chain configuration, with up to 63 devices connected off a single port
- 1022 FireWire buses can be interconnected using bridges
- Provides for hot plugging which makes it possible to connect and disconnect peripherals without having to power the computer system down or reconfigure the system
- Provides for automatic configuration
- No terminations and the system automatically performs a configuration function to assign addresses

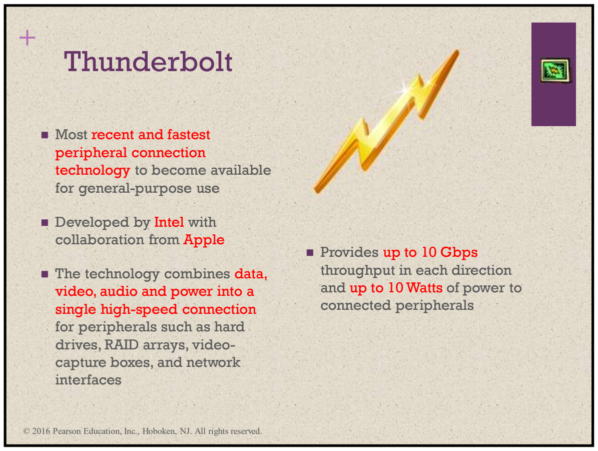
42



+ SCSI

- Small Computer System Interface
- Common standard for connecting peripheral devices to small and medium-sized computers
- Has lost popularity to USB and FireWire in smaller systems
- High-speed versions remain popular for mass memory support on enterprise systems
- Physical organization is a shared bus, which can support up to 16 or 32 devices, depending on the generation of the standard
  - The bus provides for parallel transmission rather than serial, with a bus width of 16 bits on earlier generations and 32 bits on later generations
  - Speeds range from 5 Mbps on the original SCSI-1 specification to 160 Mbps on SCSI-3 U3

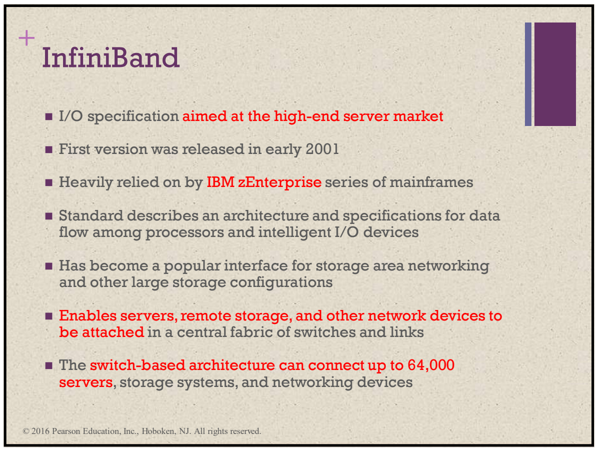
43



+ Thunderbolt

- Most recent and fastest peripheral connection technology to become available for general-purpose use
- Developed by Intel with collaboration from Apple
- The technology combines data, video, audio and power into a single high-speed connection for peripherals such as hard drives, RAID arrays, video-capture boxes, and network interfaces
- Provides up to 10 Gbps throughput in each direction and up to 10 Watts of power to connected peripherals

44

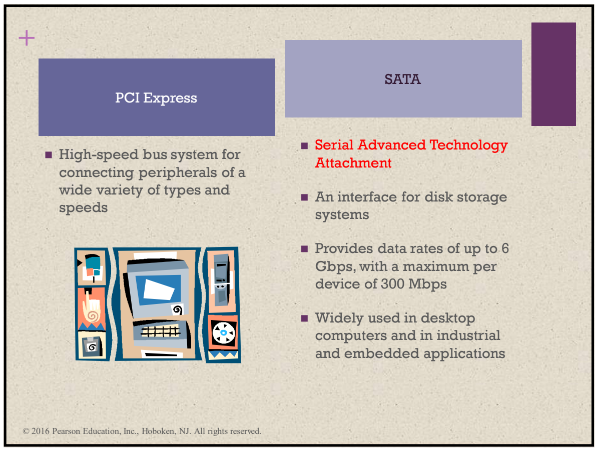


+

InfiniBand

- I/O specification aimed at the high-end server market
- First version was released in early 2001
- Heavily relied on by IBM zEnterprise series of mainframes
- Standard describes an architecture and specifications for data flow among processors and intelligent I/O devices
- Has become a popular interface for storage area networking and other large storage configurations
- Enables servers, remote storage, and other network devices to be attached in a central fabric of switches and links
- The switch-based architecture can connect up to 64,000 servers, storage systems, and networking devices

45



+

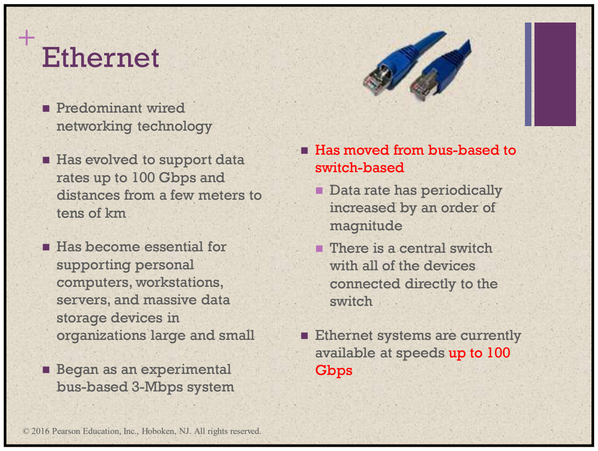
PCI Express

- High-speed bus system for connecting peripherals of a wide variety of types and speeds

SATA

- Serial Advanced Technology Attachment
- An interface for disk storage systems
- Provides data rates of up to 6 Gbps, with a maximum per device of 300 Mbps
- Widely used in desktop computers and in industrial and embedded applications

46

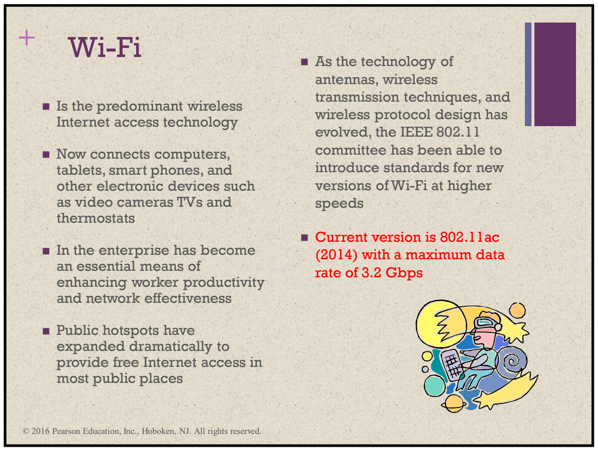


+

Ethernet

- Predominant wired networking technology
- Has evolved to support data rates up to 100 Gbps and distances from a few meters to tens of km
- Has become essential for supporting personal computers, workstations, servers, and massive data storage devices in organizations large and small
- Began as an experimental bus-based 3-Mbps system
- Has moved from bus-based to switch-based
  - Data rate has periodically increased by an order of magnitude
  - There is a central switch with all of the devices connected directly to the switch
- Ethernet systems are currently available at speeds up to 100 Gbps

47



+

Wi-Fi

- Is the predominant wireless Internet access technology
- Now connects computers, tablets, smart phones, and other electronic devices such as video cameras TVs and thermostats
- In the enterprise has become an essential means of enhancing worker productivity and network effectiveness
- Public hotspots have expanded dramatically to provide free Internet access in most public places
- As the technology of antennas, wireless transmission techniques, and wireless protocol design has evolved, the IEEE 802.11 committee has been able to introduce standards for new versions of Wi-Fi at higher speeds
- Current version is 802.11ac (2014) with a maximum data rate of 3.2 Gbps

48

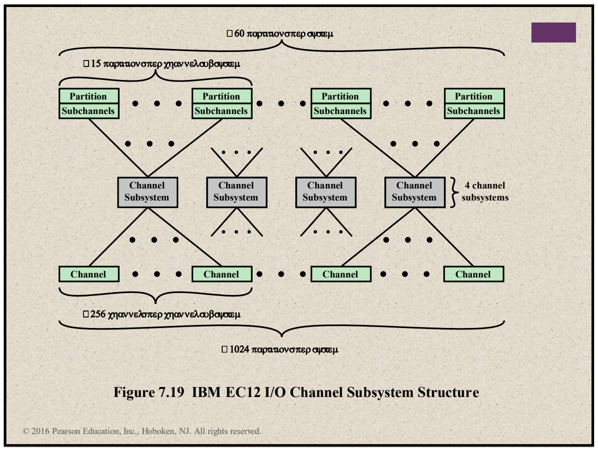
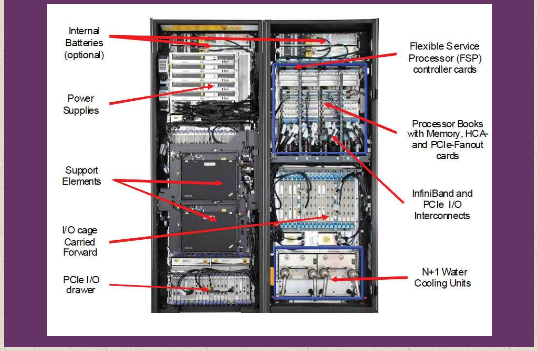


Figure 7.19 IBM EC12 I/O Channel Subsystem Structure

49

Figure 7.20
IBM zEC12 I/O Frames-Front View


Internal Batteries (optional)
Power Supplies
Support Elements
I/O cage Carried Forward
PCIe I/O drawer
Flexible Service Processor (FSP) controller cards
Processor Books with Memory, HCA- and PCIe-Fanout cards
InfiniBand and PCIe I/O Interconnects
N+1 Water Cooling Units

50

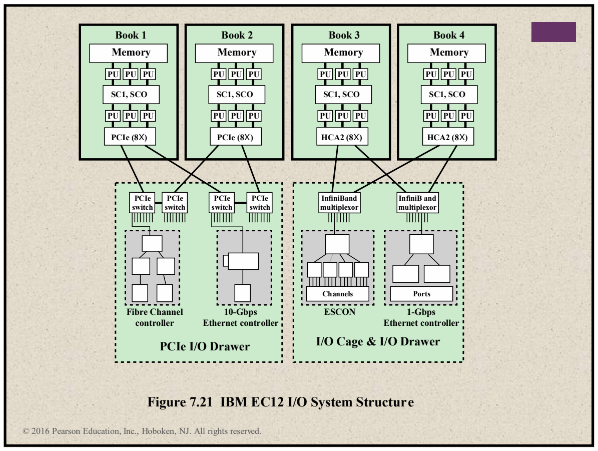
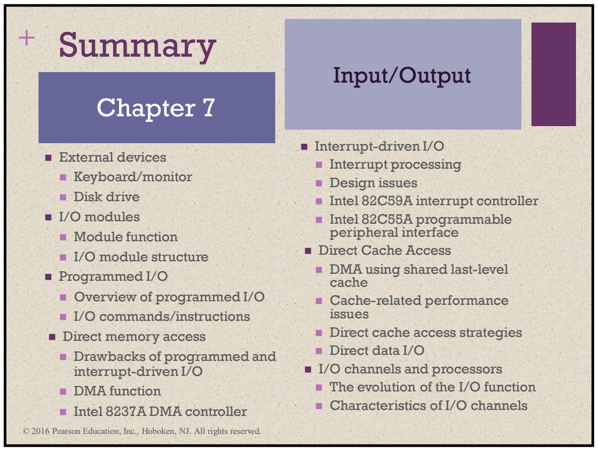


Figure 7.21 IBM EC12 I/O System Structure

51



+

Summary

Chapter 7

Input/Output

- External devices
  - Keyboard/monitor
  - Disk drive
- I/O modules
  - Module function
  - I/O module structure
- Programmed I/O
  - Overview of programmed I/O
  - I/O commands/instructions
- Direct memory access
  - Drawbacks of programmed and interrupt-driven I/O
  - DMA function
  - Intel 8237A DMA controller
- Interrupt-driven I/O
  - Interrupt processing
  - Design issues
  - Intel 82C59A interrupt controller
  - Intel 82C55A programmable peripheral interface
- Direct Cache Access
  - DMA using shared last-level cache
  - Cache-related performance issues
  - Direct cache access strategies
  - Direct data I/O
- I/O channels and processors
  - The evolution of the I/O function
  - Characteristics of I/O channels

52