

I/O Management and Disk Scheduling

Categories of I/O Devices

- Human readable
 - Used to communicate with the user
 - Printers
 - Video display terminals
 - Display
 - Keyboard
 - Mouse

Categories of I/O Devices

- Machine readable
 - Used to communicate with electronic equipment
 - Disk and tape drives
 - Sensors
 - Controllers
 - Actuators

Categories of I/O Devices

- Communication
 - Used to communicate with remote devices
 - Digital line drivers
 - Modems

Differences in I/O Devices

- Data rate
 - May be differences of several orders of magnitude between the data transfer rates

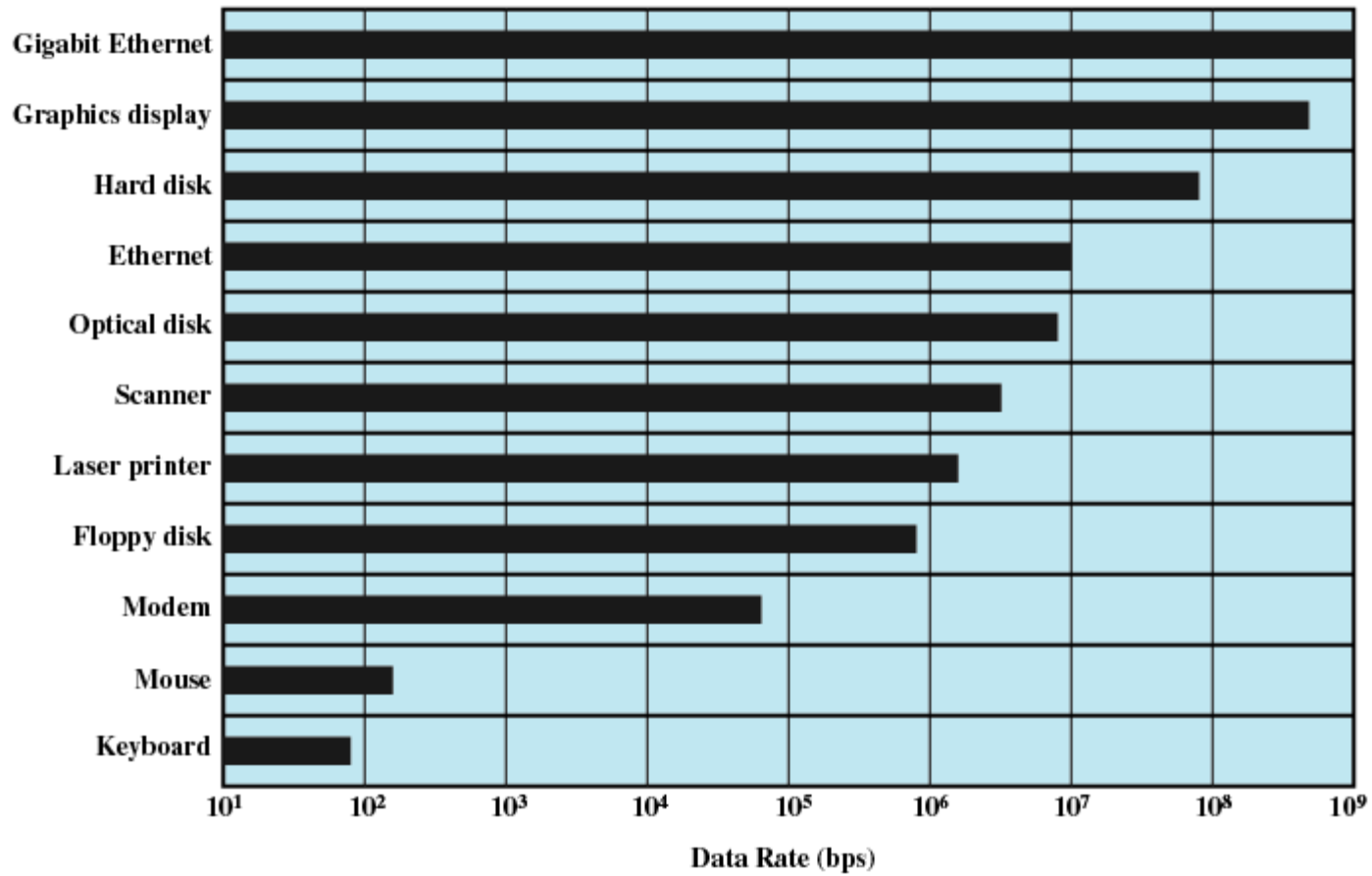


Figure 11.1 Typical I/O Device Data Rates

Figure info is a bit dated, e.g. “what’s a floppy” :-)

Differences in I/O Devices

- Application
 - Disk used to store files requires file management software
 - Disk used to store virtual memory pages needs special hardware and software to support it
 - Terminal used by system administrator may have a higher priority

Differences in I/O Devices

- Complexity of control
- Unit of transfer
 - Data may be transferred as a stream of bytes for a terminal or in larger blocks for a disk
- Data representation
 - Encoding schemes
- Error conditions
 - Devices respond to errors differently

Performing I/O

- Programmed I/O
 - Process is busy-waiting for the operation to complete
- Interrupt-driven I/O
 - I/O command is issued
 - Processor continues executing instructions
 - I/O module sends an interrupt when done

Performing I/O

- Direct Memory Access (DMA)
 - DMA module controls exchange of data between main memory and the I/O device
 - Processor interrupted only after entire block has been transferred

Relationship Among Techniques

Table 11.1 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)

Evolution of the I/O Function

- Processor directly controls a peripheral device
- Controller of I/O module is added
 - Processor uses programmed I/O without interrupts
 - Processor does not need to handle details of external devices

Evolution of the I/O Function

- Controller or I/O module with interrupts
 - Processor does not spend time waiting for an I/O operation to be performed
- Direct Memory Access
 - Blocks of data are moved into memory without involving the processor
 - Processor involved at beginning and end only

Evolution of the I/O Function

- I/O module is a separate processor
 - CPU directs I/O processor to execute program in main memory
- I/O processor
 - I/O module now has its own local memory
 - It's a computer in its own right

Direct Memory Access

- Processor delegates I/O operation to the DMA module
- DMA module transfers data directly to or from memory
- When complete DMA module sends an interrupt signal to the processor

DMA

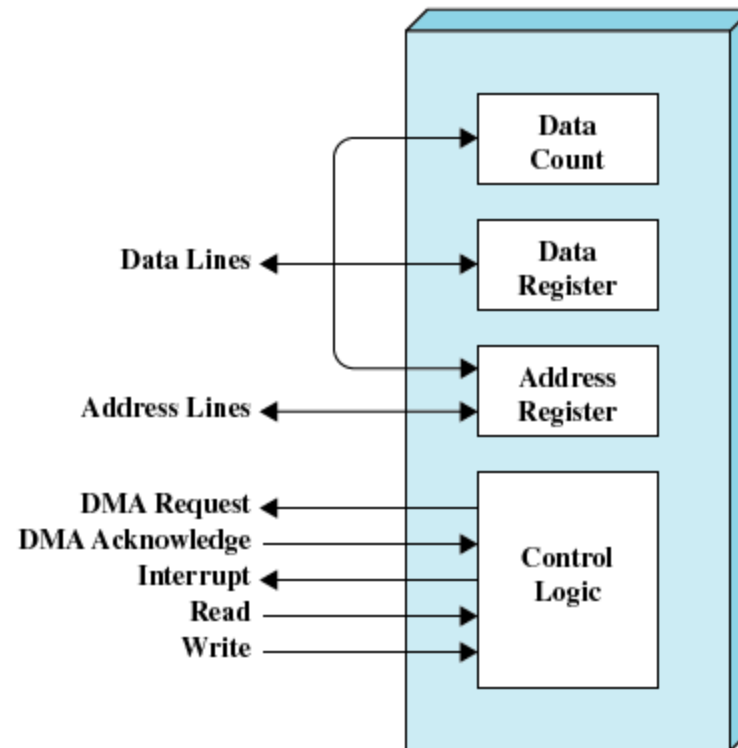
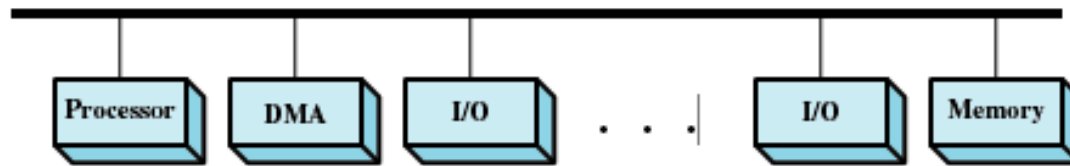
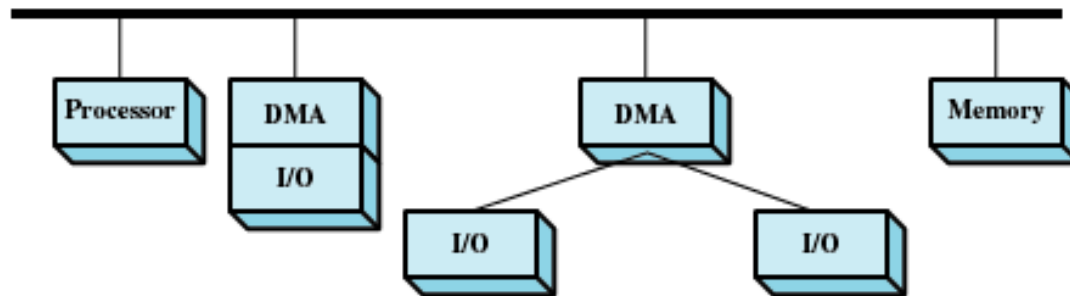


Figure 11.2 Typical DMA Block Diagram

DMA Configurations



(a) Single-bus, detached DMA



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

DMA Configurations

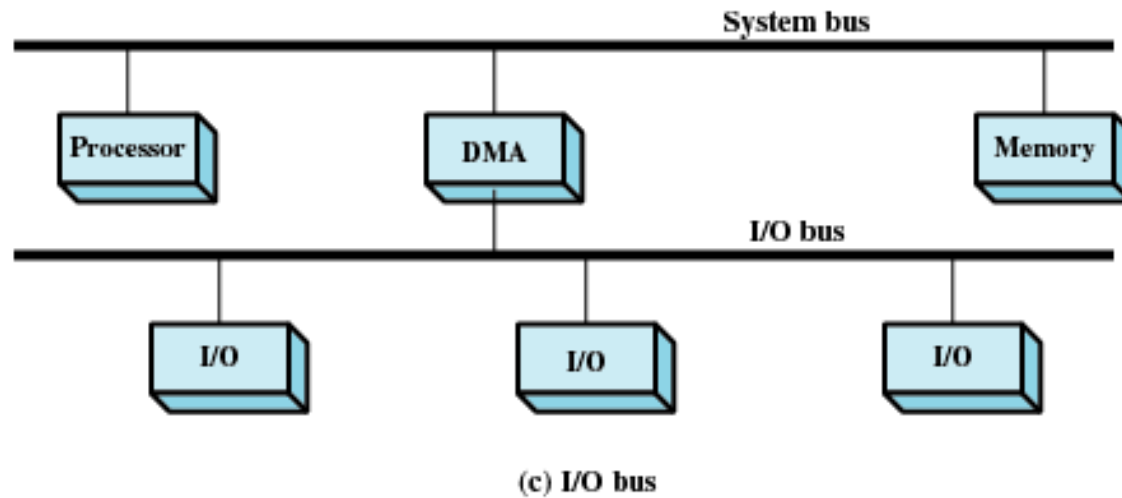


Figure 11.3 Alternative DMA Configurations

Operating System Design Issues

- Efficiency
 - Most I/O devices extremely slow compared to main memory
 - Use of multiprogramming allows for some processes to be waiting on I/O while another process executes
 - I/O cannot keep up with processor speed
 - Swapping is used to bring in additional *Ready Processes*, which is an I/O operation

Operating System Design Issues

- Generality
 - Desirable to handle all I/O devices in a uniform manner
 - Hide most of the details of device I/O in lower-level routines so that processes and upper levels see devices in general terms such as read, write, open, close, lock, unlock

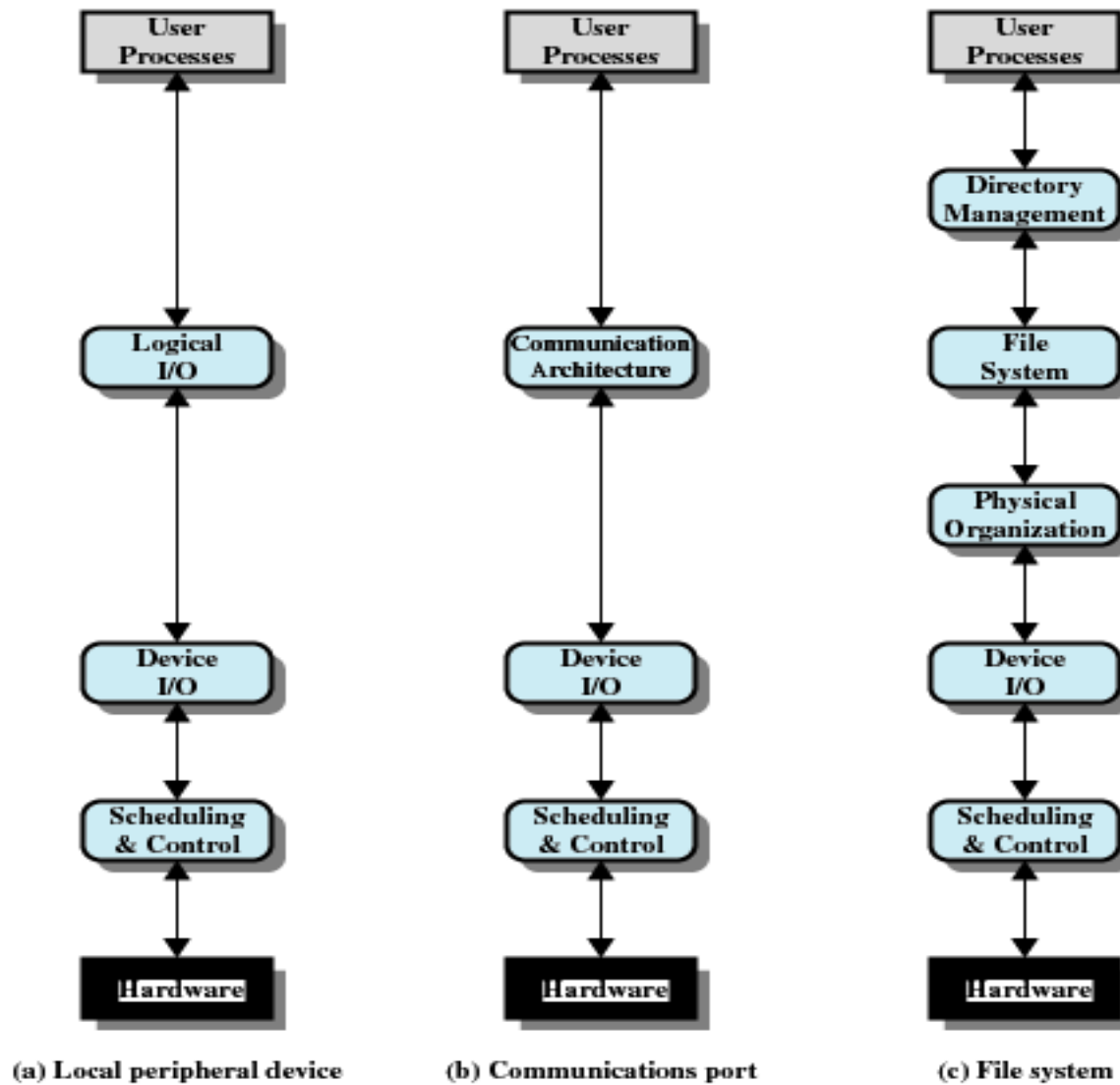


Figure 11.4 A Model of I/O Organization

I/O Buffering

- Reasons for buffering
 - Processes must wait for I/O to complete before proceeding
 - Certain pages must remain in main memory during I/O

I/O Buffering

- Block-oriented
 - Information is stored in fixed sized blocks
 - Transfers are made a block at a time
 - Used for disks and tapes
- Stream-oriented
 - Transfer information as a stream of bytes
 - Used for terminals, printers, communication ports, mouse and other pointing devices, and most other devices that are not secondary storage

Single Buffer

- Operating system assigns a buffer in main memory for an I/O request
- Block-oriented
 - Input transfers made to buffer
 - Block moved to user space when needed
 - Another block is moved into the buffer
 - “Read ahead”

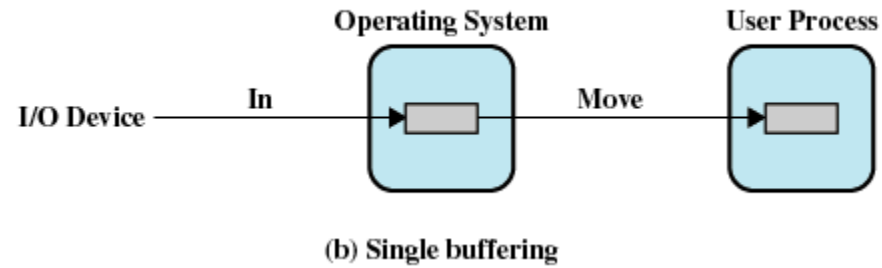
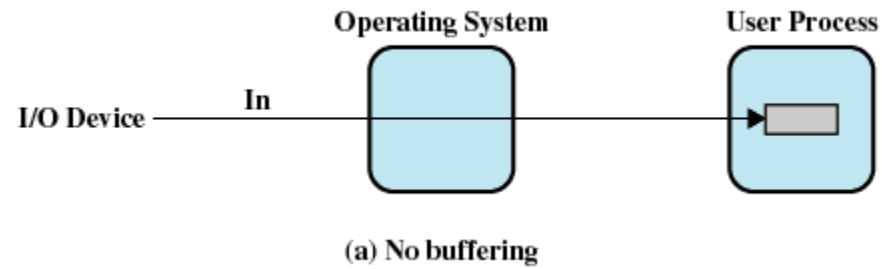
Single Buffer

- Block-oriented (cont.)
 - User process can process one block of data while next block is read in
 - Swapping can occur since input is taking place in system memory, not user memory
 - Operating system keeps track of assignment of system buffers to user processes

Single Buffer

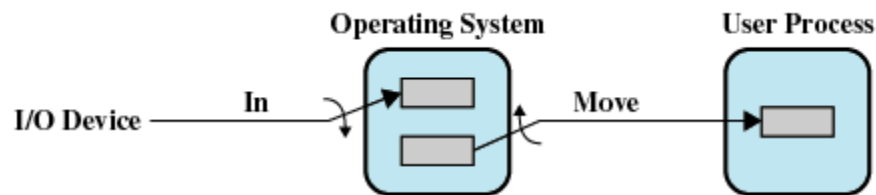
- Stream-oriented
 - e.g. terminal
 - Used a line at time
 - User input from a terminal is one line at a time with carriage return signaling the end of the line
 - Output to the terminal is one line at a time
 - e.g. network I/O
 - NIC (**n**etwork **i**nterface **c**ard)
 - protocol stack

I/O Buffering



Double Buffer

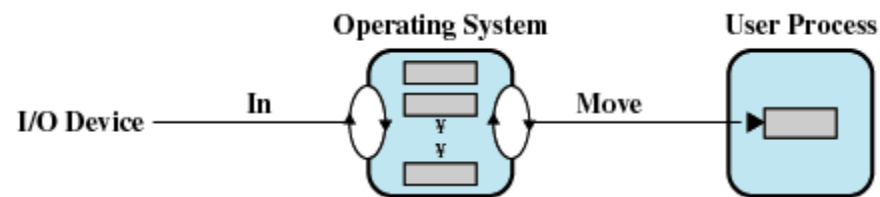
- Use two system buffers instead of one
- A process can transfer data to or from one buffer while the operating system empties or fills the other buffer



(c) Double buffering

Circular Buffer

- More than two buffers are used
- Each individual buffer is one unit in a circular buffer
- Used when I/O operation must keep up with process



(d) Circular buffering