Operating System

Dr. GuoJun LIU

Harbin Institute of Technology

http://guojunos.hit.edu.cn

Chapter 10

I/O Management and Disk Scheduling

I/O管理和磁盘调度

Learning Objectives

- Summarize key categories of I/O devices on computers
- Discuss the organization of the I/O function
- Explain some of the key issues in the design of OS support for I/O
- Analyze the performance implications of various I/O buffering alternatives
- Understand the performance issues involved in magnetic disk access

Outline

- **I/O Devices**
- **Organization of the I/O Function**
  - The Evolution of the I/O Function
  - Direct Memory Access
- **OS Design Issues**
  - Design Objectives
  - Logical Structure of the I/O Function
- **I/O Buffering**
  - Single Buffer
  - Double Buffer
  - Circular Buffer
  - The Utility of Buffering
- **Disk Scheduling**
  - Disk Performance Parameters
  - Disk Scheduling Policies

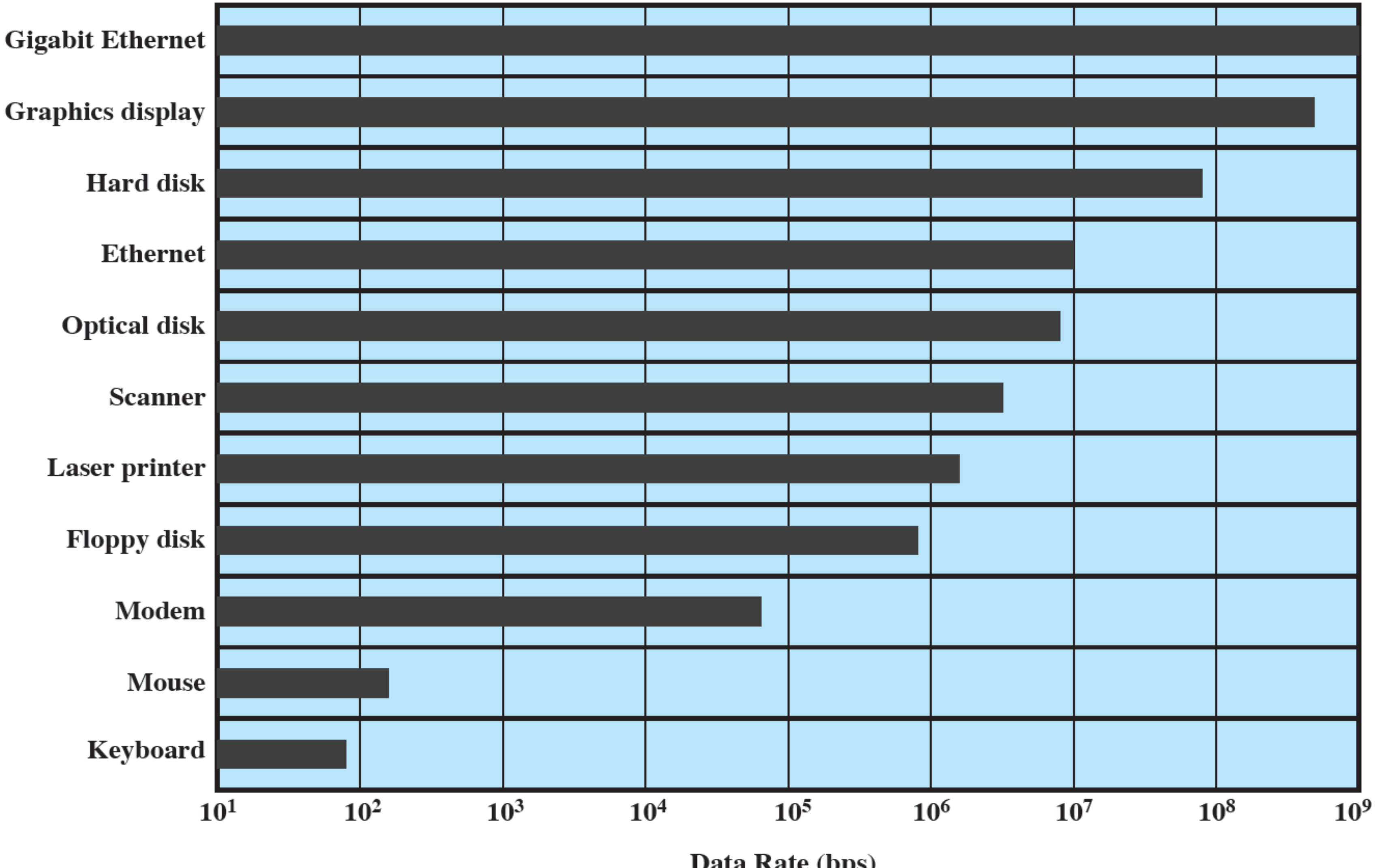
Categories of I/O Devices

- **External devices that engage in I/O with computer systems can be grouped into three categories:**
  - Human readable
    - suitable for communicating with the computer user
    - printers, terminals, video display, keyboard, mouse
  - Machine readable
    - suitable for communicating with electronic equipment
    - disk drives, USB keys, sensors, controllers
  - Communication
    - suitable for communicating with remote devices
    - modems, digital line drivers

Differences in I/O Devices

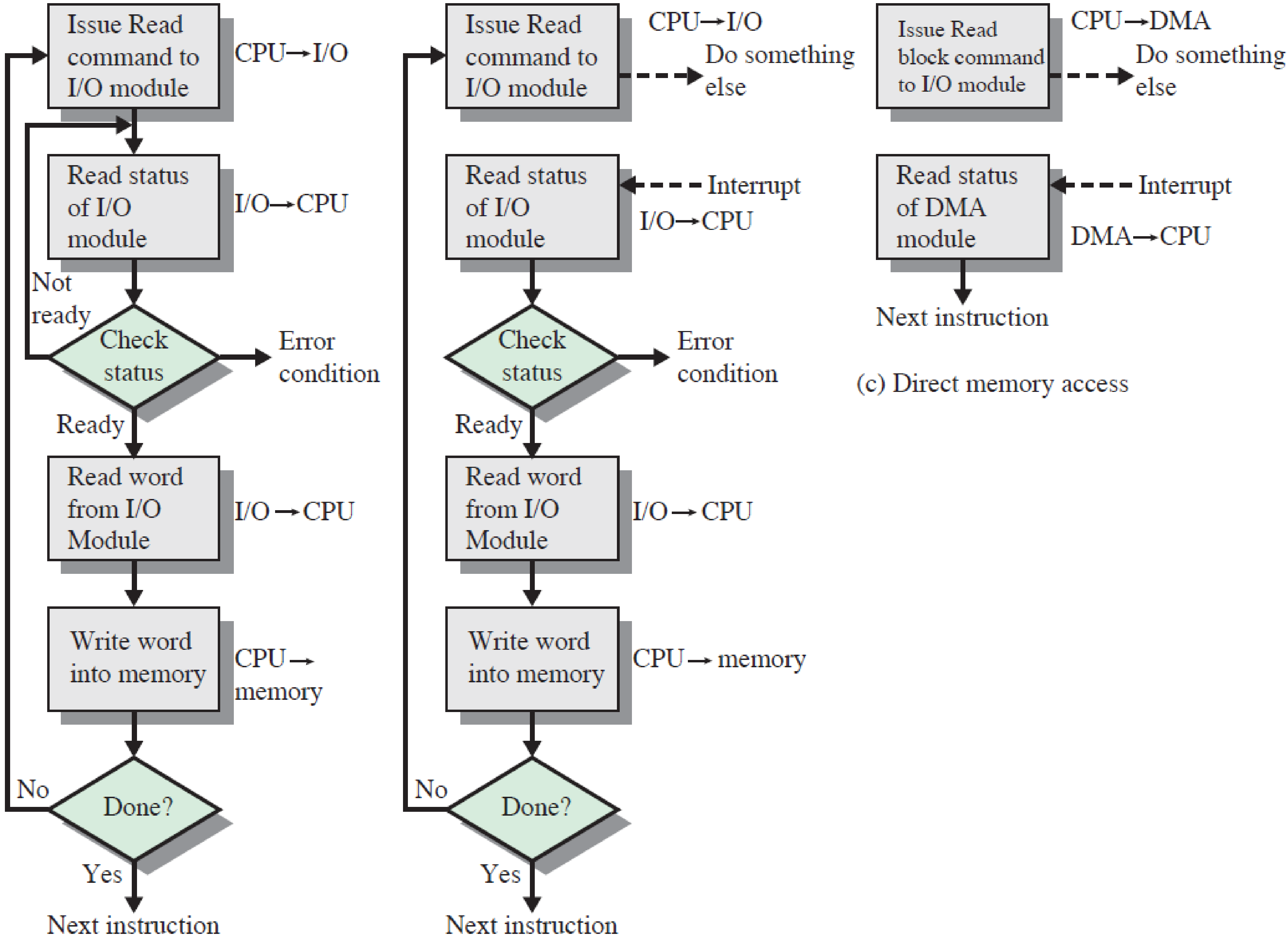
- **Data Rate**
  - there may be differences of magnitude between the data transfer rates
- **Application**
  - the use to which a device is put has an influence on the software
- **Complexity of Control**
  - the effect on the operating system is filtered by the complexity of the I/O module that controls the device
- **Unit of Transfer**
  - data may be transferred as a stream of bytes or characters or in larger blocks
- **Data Representation**
  - different data encoding schemes are used by different devices
- **Error Conditions**
  - the nature of errors, the way in which they are reported, their consequences, and the available range of responses differs from one device to another

Typical I/O Device Data Rates



Organization of the I/O Function

- **Programmed I/O**
  - the processor issues an I/O command on behalf of a process to an I/O module; that process then busy waits for the operation to be completed before proceeding
- **Interrupt-driven I/O**
  - the processor issues an I/O command on behalf of a process
- **Direct Memory Access (DMA)**
  - a DMA module controls the exchange of data **between main memory and an I/O module**



(a) Programmed I/O

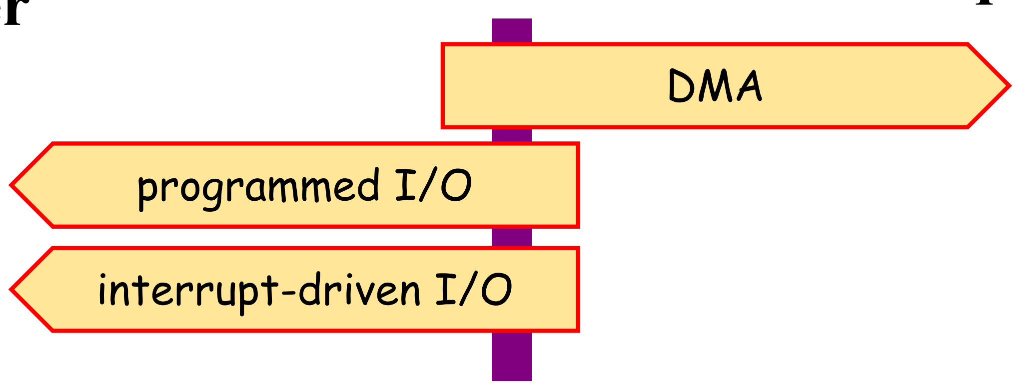
(b) Interrupt-driven I/O

(c) Direct memory access

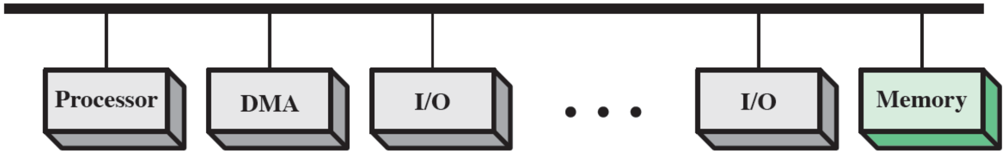
Compare three techniques

- ☒ Transfer rate is **limited** by the speed with which the processor can test and service a device
- ☒ The processor is **tied up** in managing an I/O transfer, a number of instructions must be executed for **each** I/O transfer

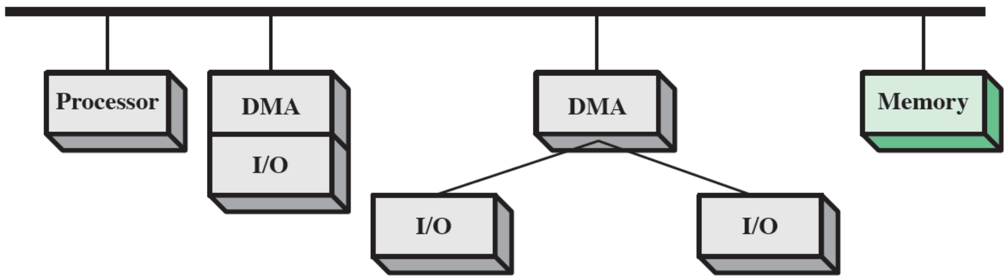
- ☑ processor is involved only at the beginning and end of the transfer
- ☑ More efficient
- ☒ processor executes more slowly during a transfer when processor access to the **bus** is required



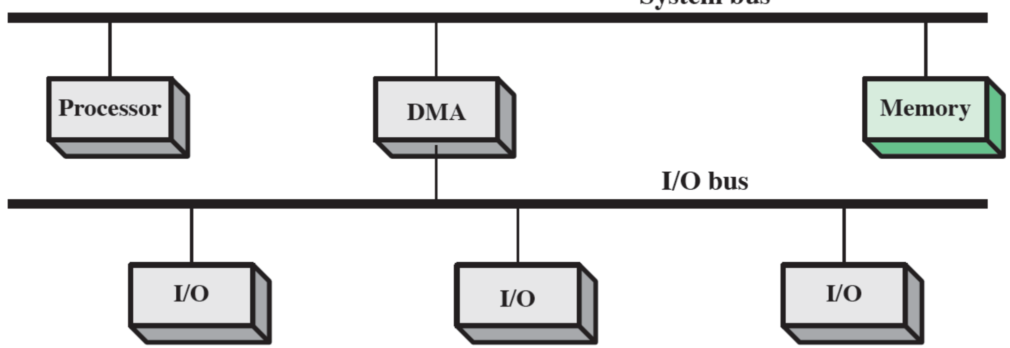
Alternative DMA Configurations



(a) Single-bus, detached DMA



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



(c) I/O bus

Evolution of the I/O Function

1. Processor **directly controls** a peripheral device
2. A **controller** or **I/O module** is added
3. Same configuration as step 2, but now **interrupts** are employed
4. The I/O module is given direct control of memory via **DMA**
5. The I/O module is enhanced to become a **separate processor**, 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

Design Objectives

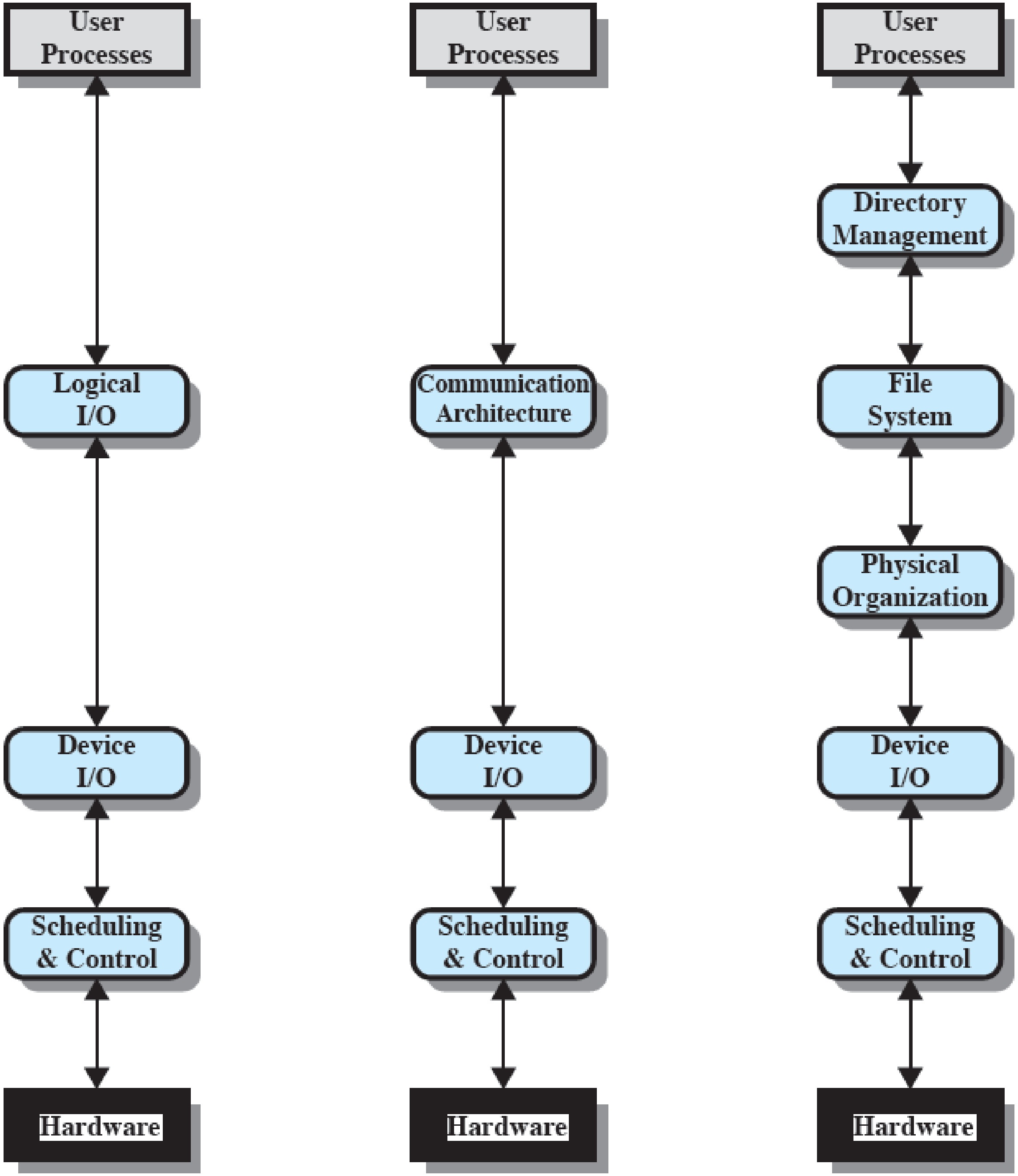
- **Efficiency**
  - Major effort in I/O design
  - Important because I/O operations often form a bottleneck
  - Most I/O devices are extremely slow compared with main memory and the processor
  - The area that has received the most attention is disk I/O

- **Generality**
  - Desirable to handle all devices **in a uniform manner**
  - Applies to the way **processes view** I/O devices and the way the **operating system** manages I/O devices and operations
  - **Diversity** of devices makes it difficult to achieve true generality
  - Use a **hierarchical, modular approach** to the design of the I/O function

Hierarchical Design

- Functions of the OS should be separated according to their complexity, their characteristic time scale, and their level of abstraction
- Leads to an organization of the operating system into a series of layers
- Each layer performs a related subset of the functions required of the operating system
- Layers should be defined so that changes in one layer do not require changes in other layers

A Model of I/O Organization



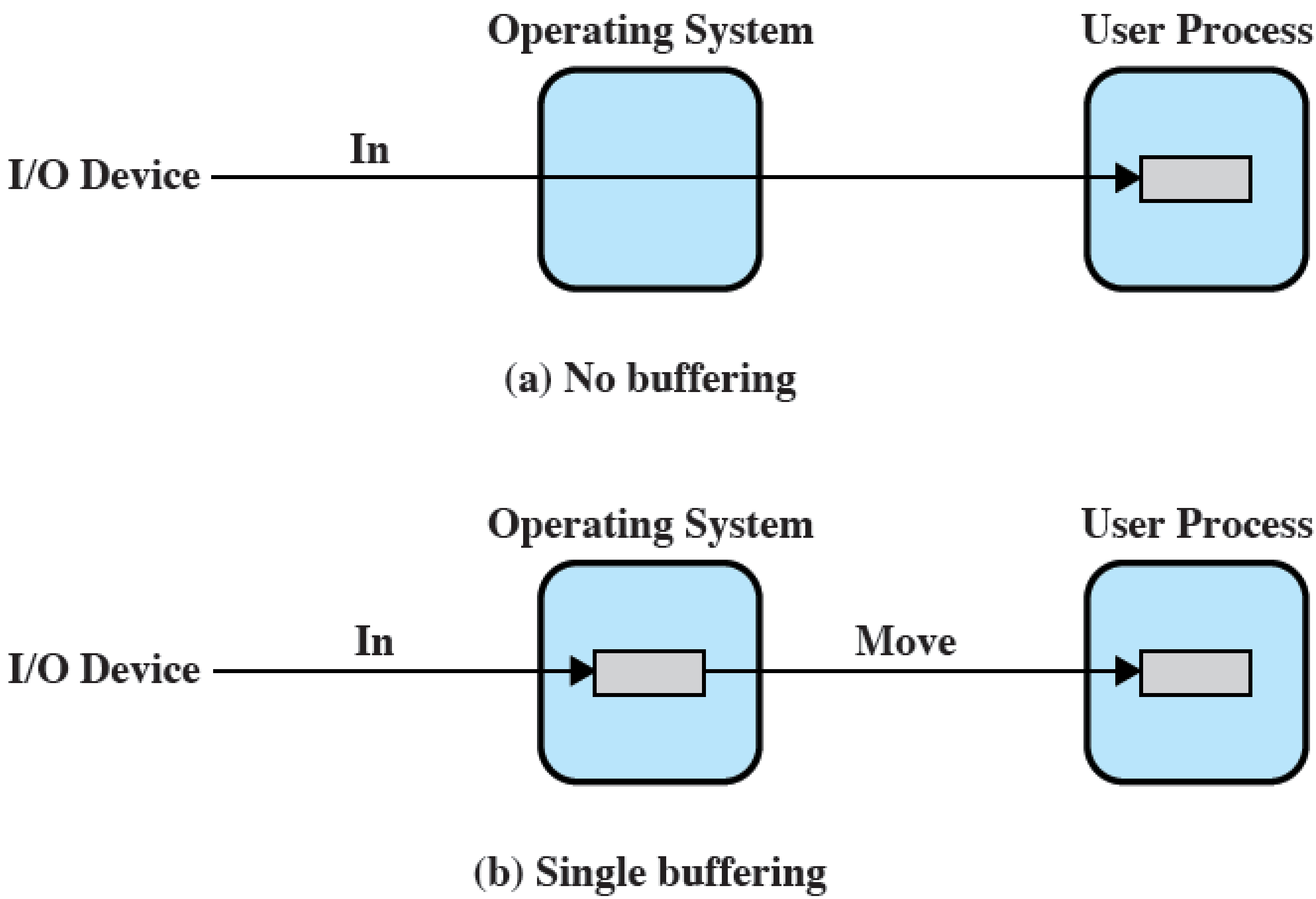
(a) Local peripheral device (b) Communications port (c) File system

Buffering

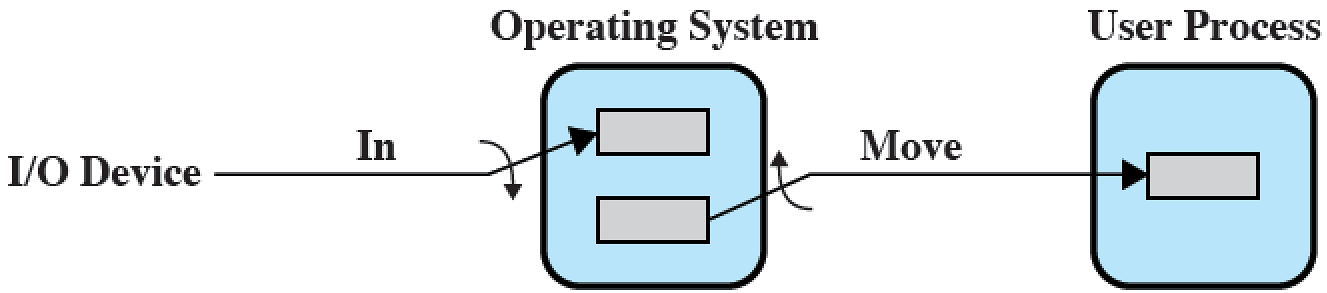
- **Perform input transfers in advance of requests being made and perform output transfers some time after the request is made**

Block-oriented device	Stream-oriented device
• stores information in blocks that are usually of fixed size • transfers are made one block at a time • possible to reference data by its block number • disks and USB keys are examples	• transfers data in and out as a stream of bytes • no block structure • terminals, printers, communications ports, and most other devices that are not secondary storage are examples

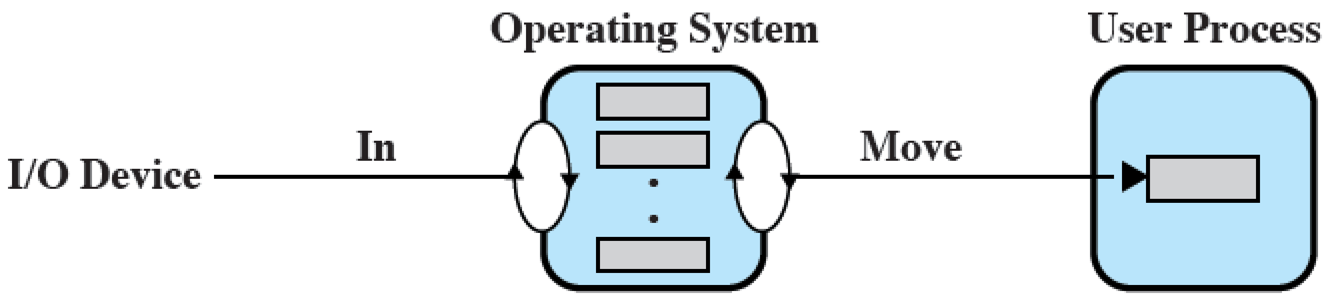
I/O Buffering Schemes (input)



I/O Buffering Schemes (input)



(c) Double buffering

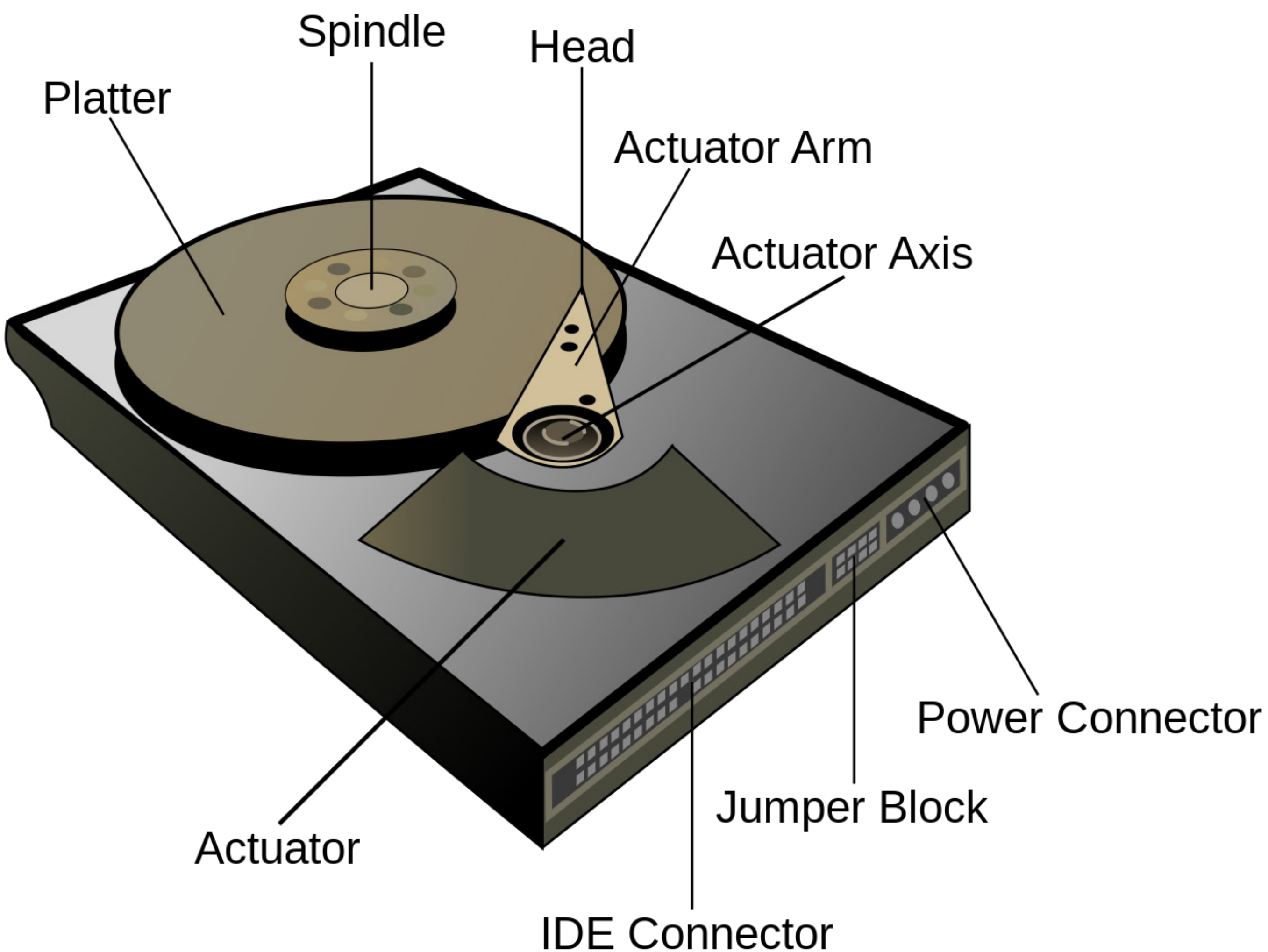


(d) Circular buffering

The Utility of Buffering

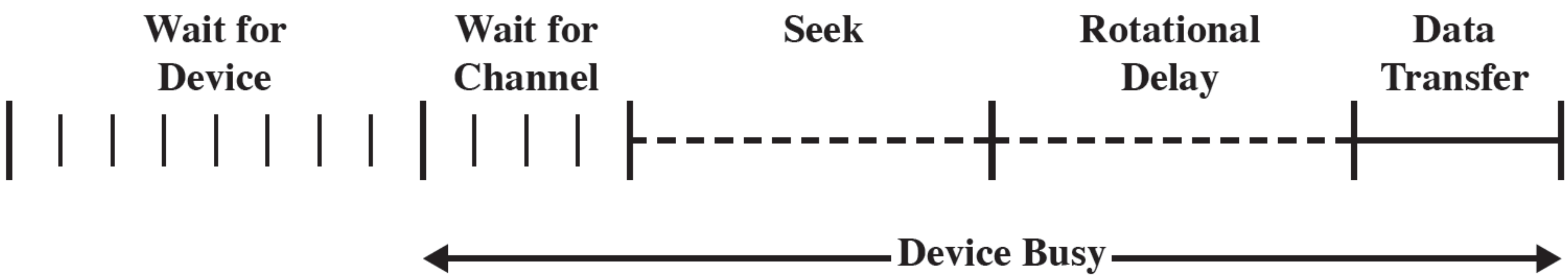
- **Technique that smoothes out peaks in I/O demand**
  - with enough demand eventually all buffers become **full** and their **advantage is lost**
- **When there is a variety of I/O and process activities to service, buffering can**
  - **increase the efficiency** of the OS
  - **increase the performance** of individual processes

Diagram labeling the major components of HDD



Disk Performance Parameters

- **The actual details of disk I/O operation depend on**
  - computer system
  - operating system
  - nature of the I/O channel and disk controller hardware



Timing of a Disk I/O Transfer

Positioning the Read/Write Heads

- When the disk drive is operating, the disk is rotating at constant speed
- To read or write the head must be positioned at the desired track and at the beginning of the desired sector on that track
- Track selection involves moving the head in a movable-head system or electronically selecting one head on a fixed-head system
- On a movable-head system the time it takes to position the head at the track is known as seek time
- The time it takes for the beginning of the sector to reach the head is known as rotational delay
- The sum of the seek time and the rotational delay equals the access time

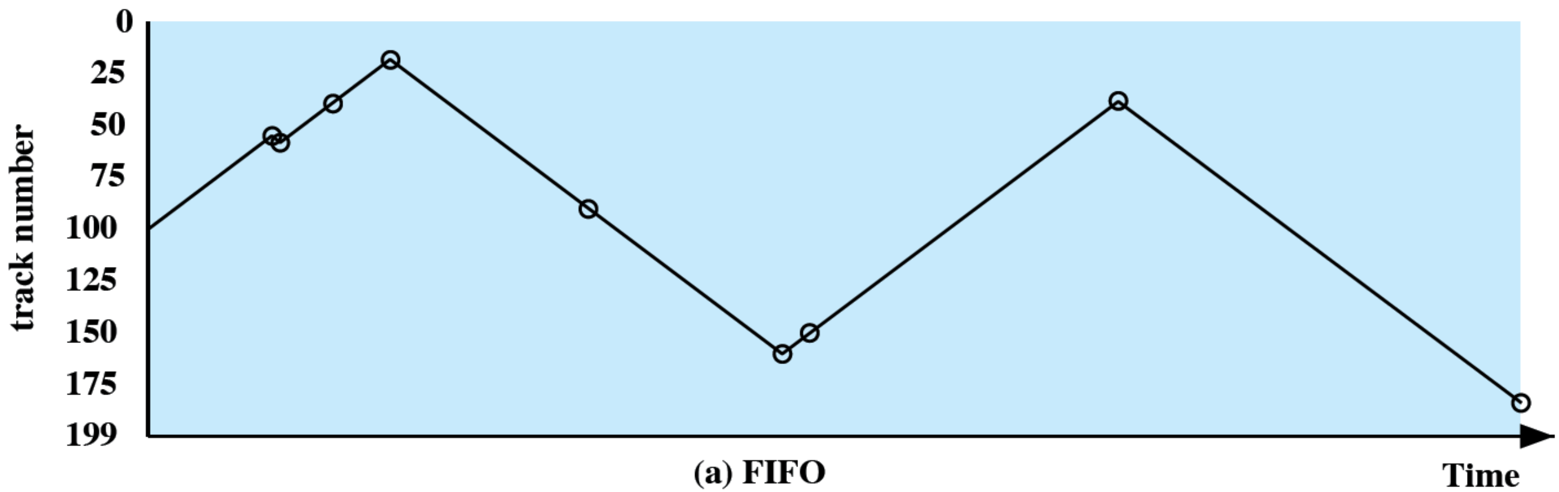
An Example Sequence of I/O Requests

- **we assume that**
  - the disk head is initially located at track 100
  - a disk with 200 tracks
  - the disk request queue has random requests in it

- **the requested tracks, in the order received by the disk scheduler are**
  - **55, 58, 39, 18, 90, 160, 150, 38, 184**

First-In, First-Out (FIFO)

- Processes in sequential order
- Fair to all processes
- Approximates random scheduling in performance if there are many processes competing for the disk



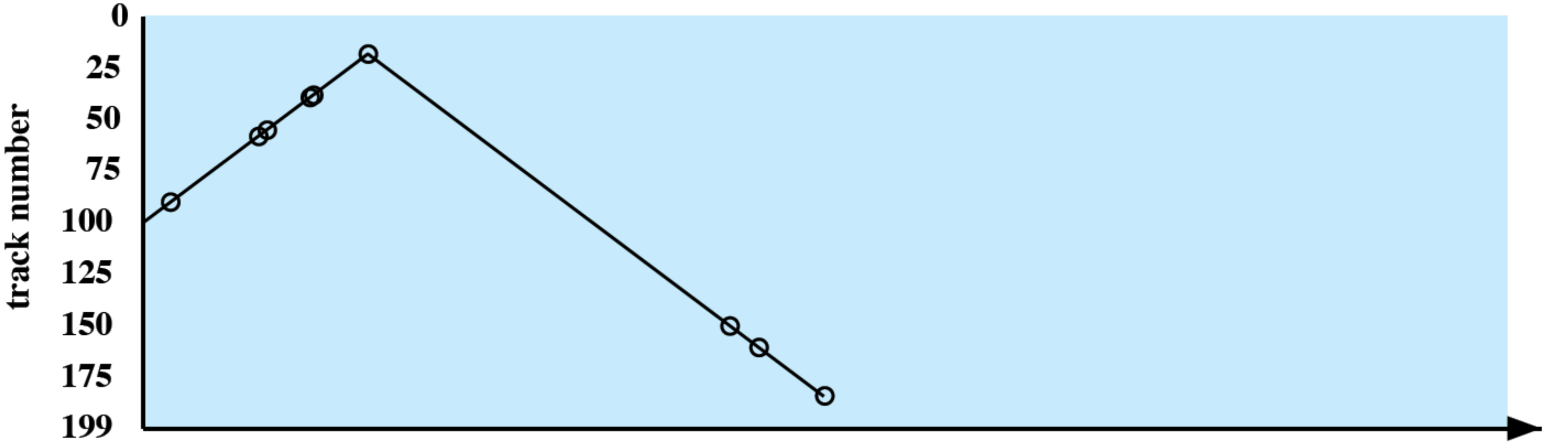
(a) FIFO

Priority (PRI)

- Control of the scheduling is outside the control of disk management software
- Goal is not to optimize disk utilization but to meet other objectives
- Short batch jobs and interactive jobs are given higher priority
- Provides good interactive response time
- Longer jobs may have to wait an excessively long time
- A poor policy for database systems

Shortest Service Time First (SSTF)

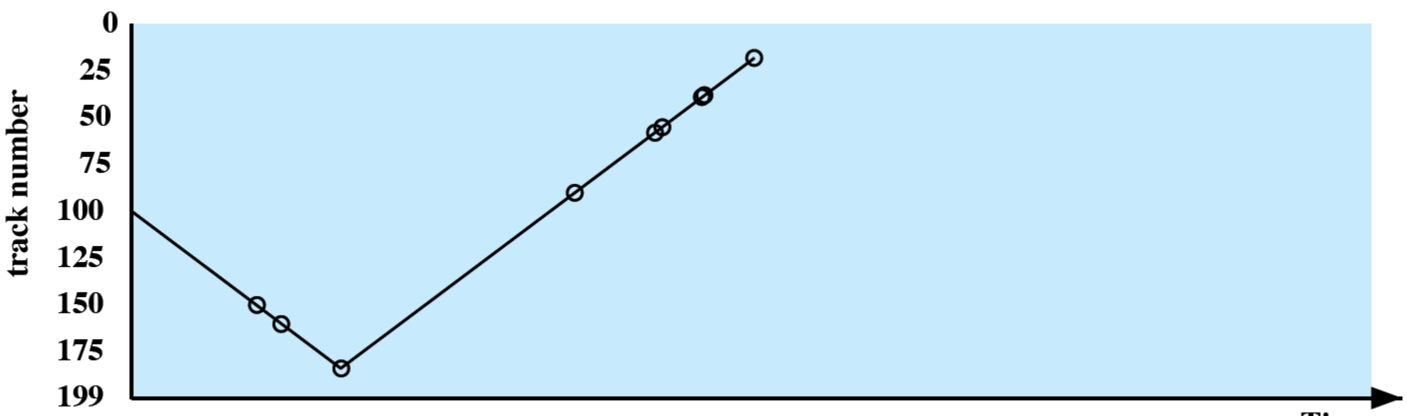
- Select the disk I/O request that requires the least movement of the disk arm from its current position
- Always choose the minimum seek time



(b) SSTF

SCAN

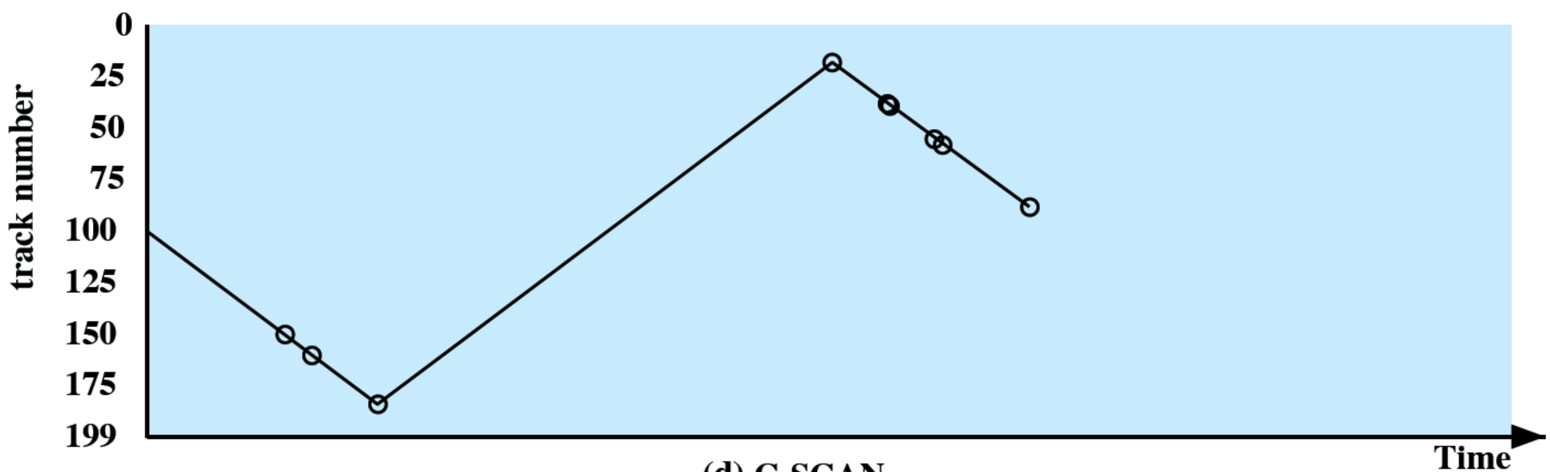
- **Also known as the** elevator algorithm
- **Arm moves in one direction only**
  - satisfies all outstanding requests until it reaches the last track in that direction then the direction is reversed
- **Favors jobs whose requests are for tracks nearest to both innermost and outermost tracks**



(c) SCAN

C-SCAN (Circular SCAN)

- **Restricts scanning to one direction only**
- **When the last track has been visited in one direction, the arm is returned to the opposite end of the disk and the scan begins again**



(d) C-SCAN

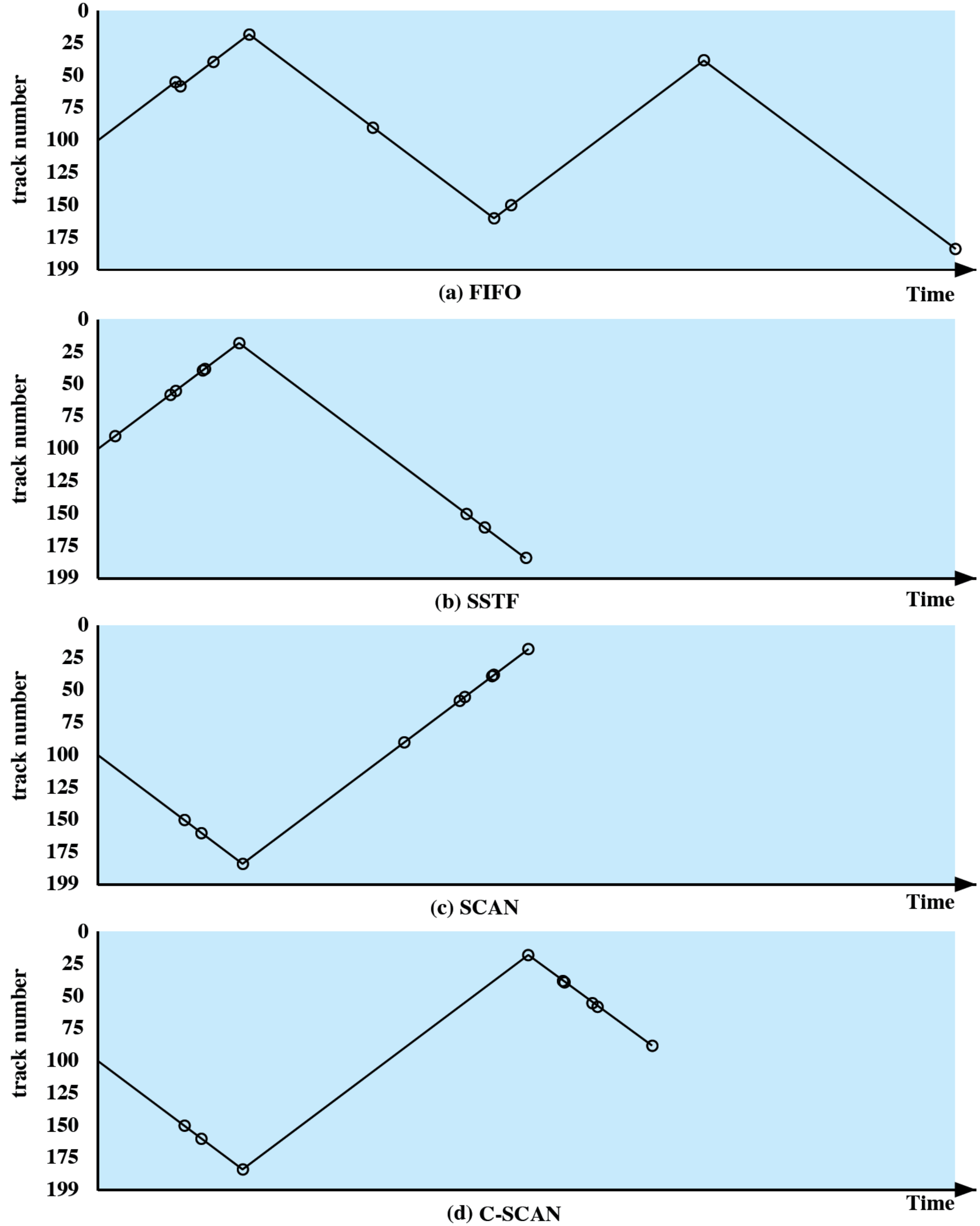
N-Step-SCAN

- Segments the disk request queue into subqueues of length N
- Subqueues are processed one at a time, using SCAN
- While a queue is being processed new requests must be added to some other queue
- If fewer than N requests are available at the end of a scan, all of them are processed with the next scan

FSCAN

- Uses two subqueues
- When a scan begins, all of the requests are in one of the queues, with the other empty
- During scan, all new requests are put into the other queue
- Service of new requests is deferred until all of the old requests have been processed

Comparison of Disk Scheduling Algorithms



Comparison of Disk Scheduling Algorithms

(a) FIFO (starting at track 100)		(b) SSTF (starting at track 100)		(c) SCAN (starting at track 100, in the direction of increasing track number)		(d) C-SCAN (starting at track 100, in the direction of increasing track number)	
Next track accessed	Number of tracks traversed	Next track accessed	Number of tracks traversed	Next track accessed	Number of tracks traversed	Next track accessed	Number of tracks traversed
55	**45**	**90**	**10**	**150**	**50**	**150**	**50**
58	**3**	**58**	**32**	**160**	**10**	**160**	**10**
39	**19**	**55**	**3**	**184**	**24**	**184**	**24**
18	**21**	**39**	**16**	**90**	**94**	**18**	**166**
90	**72**	**38**	**1**	**58**	**32**	**38**	**20**
160	**70**	**18**	**20**	**55**	**3**	**39**	**1**
150	**10**	**150**	**132**	**39**	**16**	**55**	**16**
38	**112**	**160**	**10**	**38**	**1**	**58**	**3**
184	**146**	**184**	**24**	**18**	**20**	**90**	**32**
Average seek length	**55.3**	Average seek length	**27.5**	Average seek length	**27.8**	Average seek length	**35.8**

Disk Scheduling Algorithms

Name	Description	Remarks
	Selection according to requestor	
RSS	Random scheduling	For analysis and simulation
FIFO	First in first out	Fairest of them all
PRI	Priority by process	Control outside of disk queue management
LIFO	Last in first out	Maximize locality and resource utilization
	Selection according to requested item	
SSTF	Shortest service time first	High utilization, small queues
SCAN	Back and forth over disk	Better service distribution
C-SCAN	One way with fast return	Lower service variability
N-step-SCAN	SCAN of N records at a time	Service guarantee
FSCAN	N-step-SCAN with N = queue size at beginning of SCAN cycle	Load sensitive

Summary

- I/O architecture is the computer system's interface to the outside world
- I/O functions are generally broken up into many layers
- A key aspect of I/O is the use of buffers
  - that are controlled by I/O utilities rather than by application processes
- Buffering smoothes out the differences between the speeds
- The use of buffers also decouples the actual I/O transfer from the address space of the application process
- Disk I/O has the greatest impact on overall system performance
- Two of the most widely used approaches are disk scheduling and the disk cache