CS 303
Operating Systems Concepts
Semester I – 2019/2020

Chapter 08 – I/O Management & Disk Scheduling

Content

- Categories of I/O Devices
- Differences in I/O Devices
- Organization of the I/O Function
  - I/O Techniques
  - Evaluation of the I/O Functions
  - Direct Memory Access (DMA)
- Operating System Design Issues
  - Design Objectives
  - Hierarchical Design
- I/O Buffering
  - No Buffer
  - Single Buffer
  - Double Buffer
  - Circular Buffer
  - The Utility of Buffering
- Disk Scheduling
  - Disk Performance Parameters
  - Positioning the Read/Write Heads
  - Disk Scheduling Algorithms

Categories of I/O Devices

External devices that engage in I/O with computer systems can be grouped into three categories:

1. **Human Readable**
   - Suitable for communicating with the computer user
   - Examples include printers, terminals, video display, keyboard, and mouse
2. **Machine Readable**
   - Suitable for communicating with electronic equipment
   - Examples are disk drives, USB keys, sensors, controllers, and actuators
3. **Communication**
   - Suitable for communicating with remote devices
   - Examples are digital line drivers and modems

Differences in I/O Devices

Devices differ in a number of areas:

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

Differences in I/O Devices (cntd...)

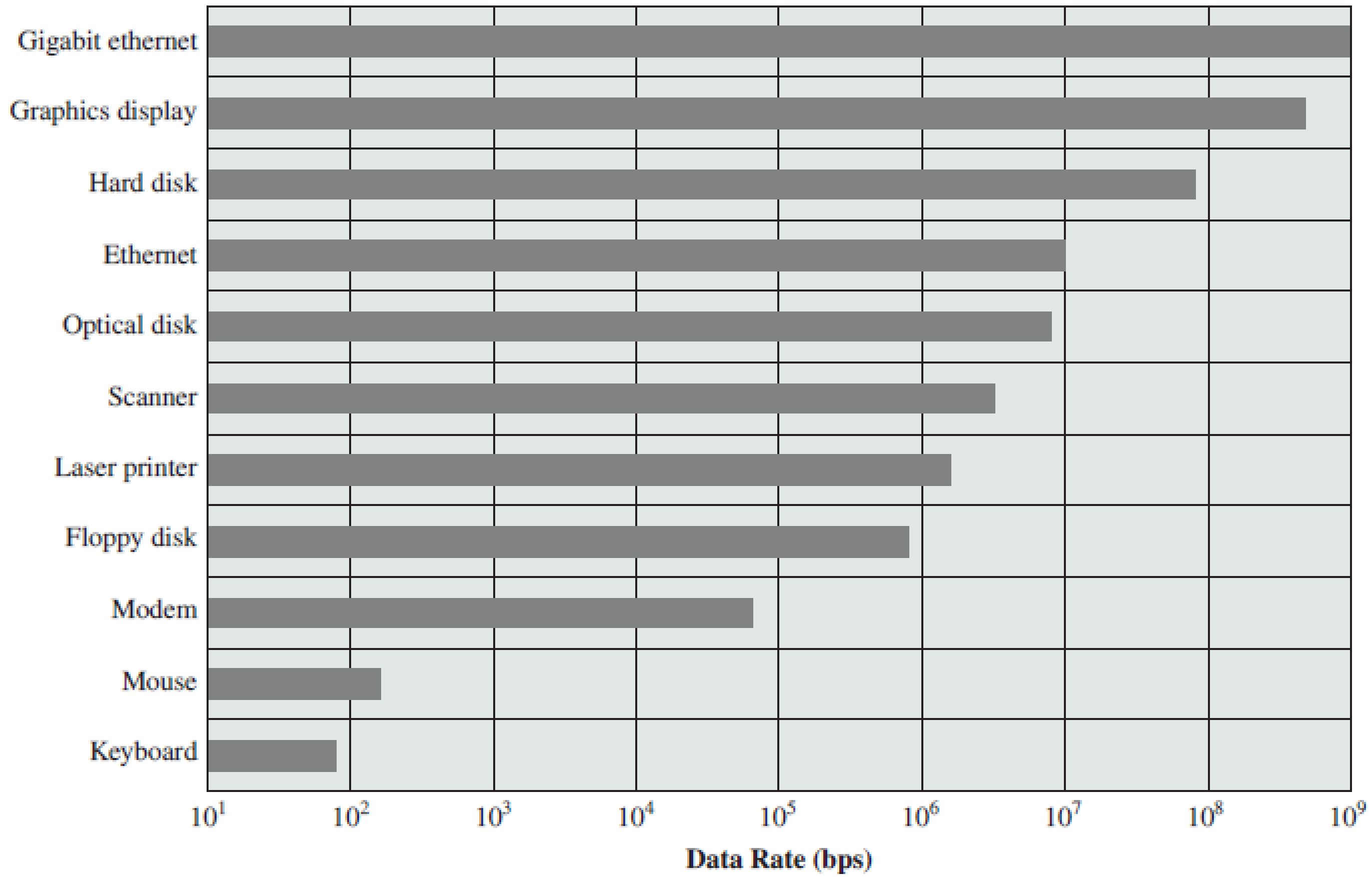


Figure 8.1: Typical I/O Device Data Rates

Organization of the I/O Function

Three techniques for performing I/O are:

1. **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
2. **Interrupt-driven I/O**
   - The processor issues an I/O command on behalf of a process
     - If **non-blocking** – Process continues to execute instructions from the process that issued the I/O command
     - If **blocking** – The next instruction the processor executes is from the OS, which will put the current process in a blocked state and schedule another process
3. **Direct Memory Access (DMA)**
   - A DMA module controls the exchange of data between main memory and an I/O module
   - The processor sends a request for the transfer of a block of data to the DMA module and is interrupted only after the entire block has been transferred

Organization of the I/O Function (cntd...)

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)

Table 8.1: I/O Techniques

Organization of the I/O Function (cntd…)

Evaluation of the I/O Function:

The evaluation steps can be summarized as follows:

1. The processor directly controls a peripheral device
2. A controller or I/O module is added
3. The same configuration as step 2 is used, 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

Organization of the I/O Function (cntd…)

Direct Memory Access (DMA):

- The DMA unit is capable of mimicking the processor and, indeed, of taking over control of the system bus just like a processor
- It needs to do this to transfer data to and from memory over the system bus

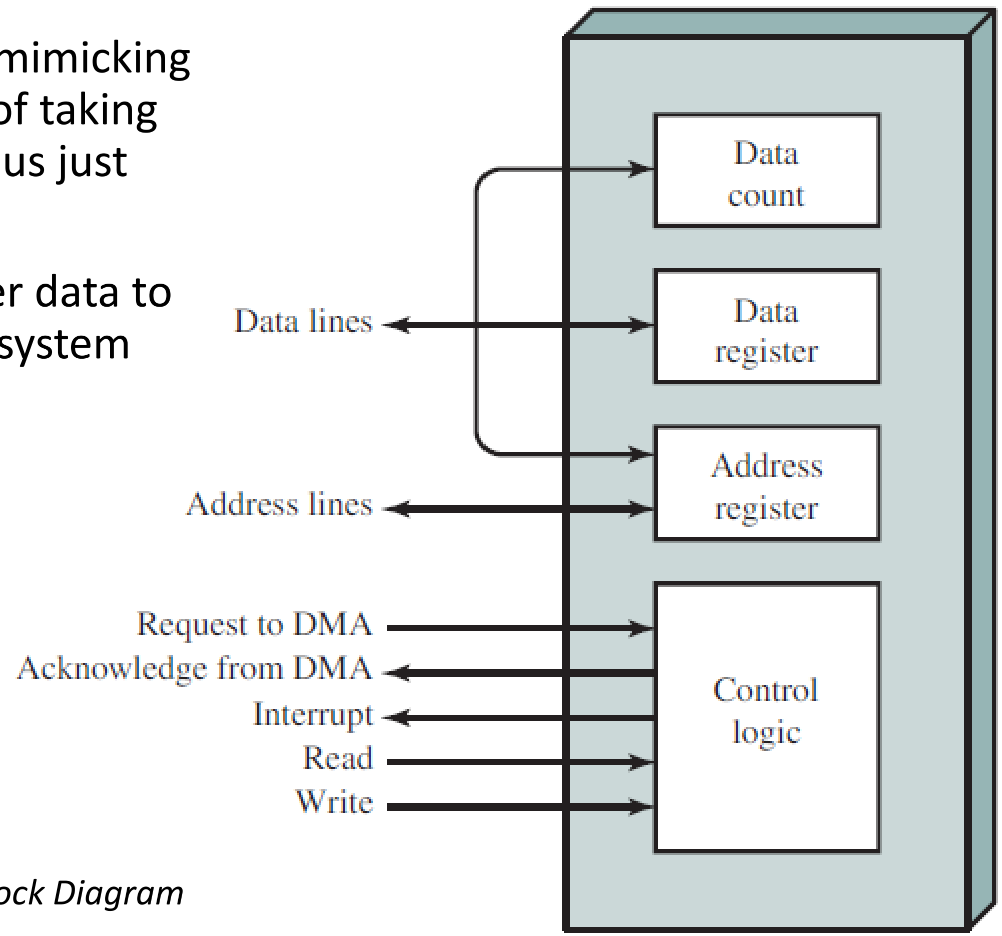


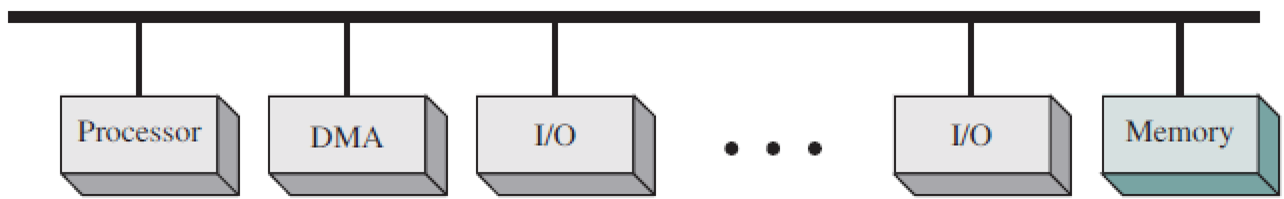
Figure 8.2: Typical DMA Block Diagram

Organization of the I/O Function (cntd…)

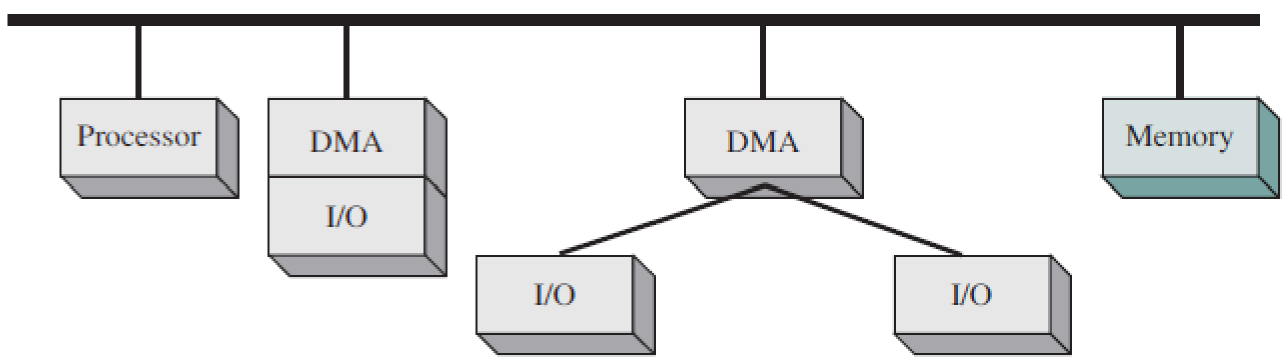
Direct Memory Access (DMA) (cntd…):

The DMA mechanism can be configured in a variety of ways:

a) **Single-bus, detached DMA**



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



Organization of the I/O Function (cntd…)

Direct Memory Access (DMA) (cntd…):

c) I/O bus

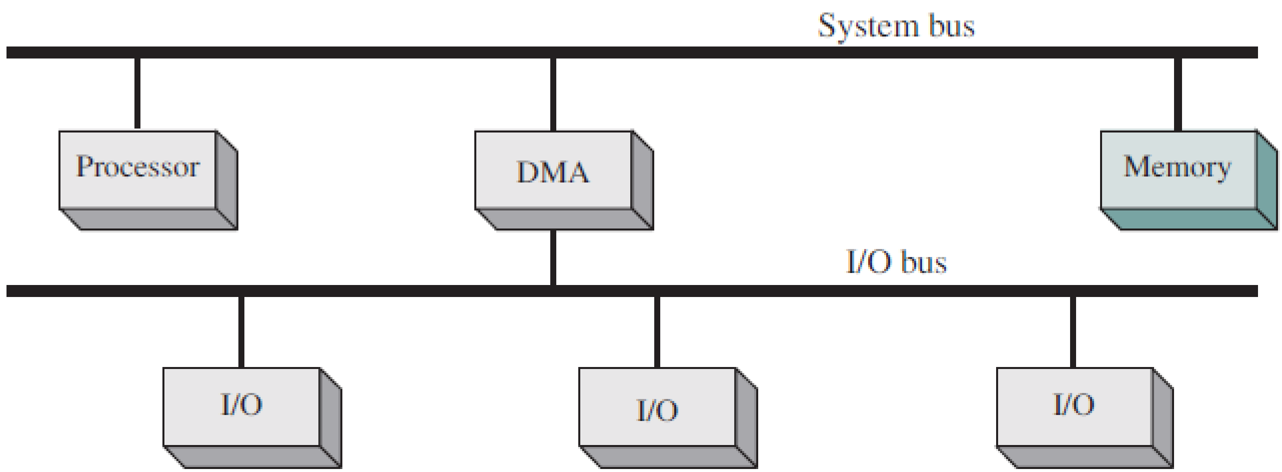


Figure 8.3: Alternative DMA Configurations

Operating System Design Issues

Design Objectives:

- Two objectives are paramount in designing the I/O facility: **Efficiency** and **Generality**

Efficiency	**Generality**
• Major effort in I/O design	• Desirable to handle all devices in a uniform manner
• Important because I/O operations often from a bottleneck	• Applies the way processes view I/O devices and the way the OS manages I/O devices and operations
• Most I/O devices are extremely slow compared with main memory and the processor	• Diversity of devices makes it difficult to achieve true generality
• The area that has received the most attention is disk I/O	• Use a hierarchical, modular approach to the design of the I/O function

Table 8.2: Comparison of Efficiency and Generality

Operating System Design Issues (cntd…)

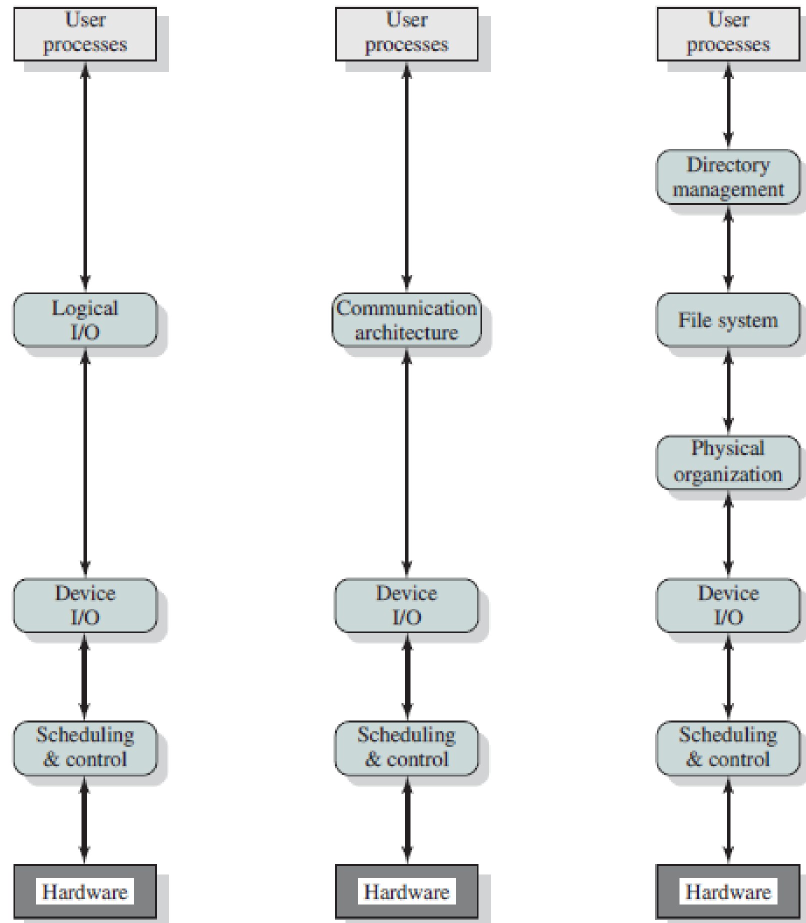
Hierarchical Design:

- 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 OS into a series of layers
- Each layer performs a related subset of the functions required of the OS
- Layers should be defined so that changes in one layer do not require changes in other layers

Operating System Design Issues (cntd...)

Hierarchical Design (cntd...):

Figure 8.4: A Model of I/O Organization



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

I/O Buffering

- Perform input transfers in advance of requests being made and perform output transfers sometime after the request is made
- Various approaches to buffering
  - When discussing them, it is important to make a distinction between two types of I/O devices:

1. **Block-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
2. **Stream-oriented Device**
   - Transfers data in and out as a stream of bytes
   - No block structure
   - Terminals, printers, and most other devices that are not secondary storage are examples

I/O Buffering (cntd…)

No Buffer:

- Without a buffer, the OS directly accesses the device when it needs

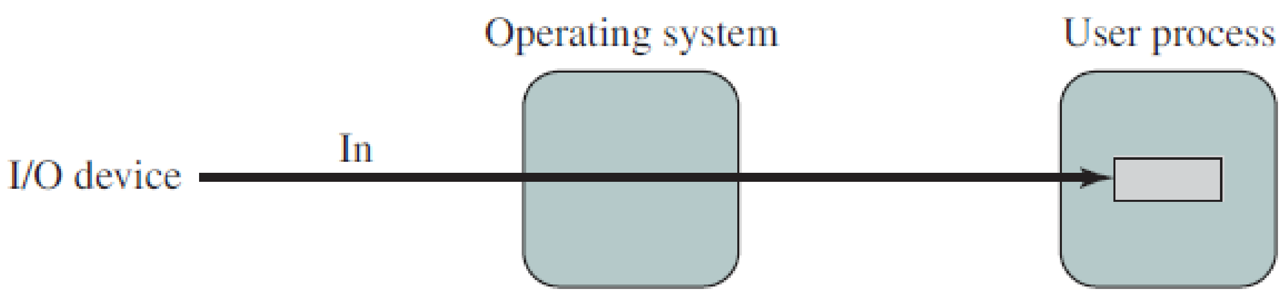


Figure 8.5: The Scheme of No Buffering

I/O Buffering (cntd…)

Single Buffer:

- Operating system assigns a buffer in main memory for an I/O request

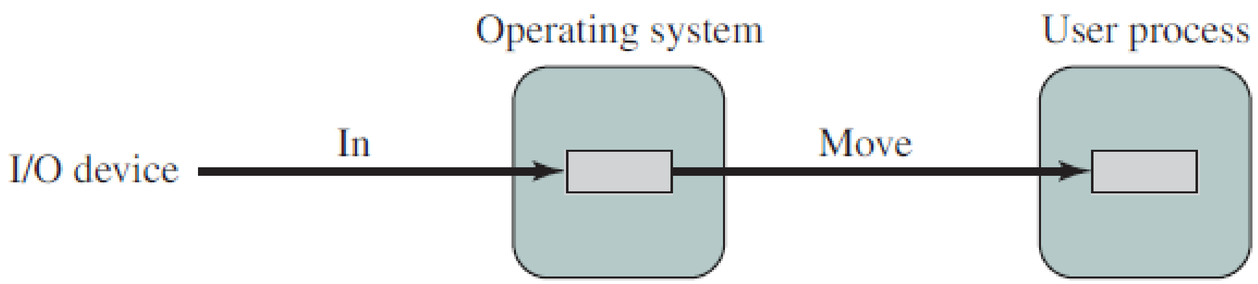


Figure 8.6: The Scheme of Single Buffering

I/O Buffering (cntd…)

Single Buffer (cntd…):

1. Block-Oriented Single Buffer:

- Input transfers are made to the system buffer
- Reading ahead/anticipated input
  - Is done in the expectation that the block will eventually be needed
  - When the transfer is complete, the process moves the block into user space and immediately requests another block
- Generally provides a speedup compared to the lack of system buffering
- Disadvantages:
  - Complicates the logic in the OS
  - Swapping logic is also affected

I/O Buffering (cntd…)

Single Buffer (cntd…):

2. Stream-Oriented Single Buffer:

i. Line-at-a-time Operation

- Appropriate for scroll-mode terminals (dumb terminals)
- User input is one line at a time with a carriage return signalling the end of a line
- Output to the terminal is similarly one line at a time

ii. Byte-at-a-time Operation

- Used on forms-mode terminals
- When each keystroke is significant
- Other peripherals such as sensors and controllers

I/O Buffering (cntd…)

Double Buffer:

- Use two system buffers instead of one
- A process can transfer data to or from one buffer while the OS empties or fills the other buffer
- Also known as **buffer swapping**

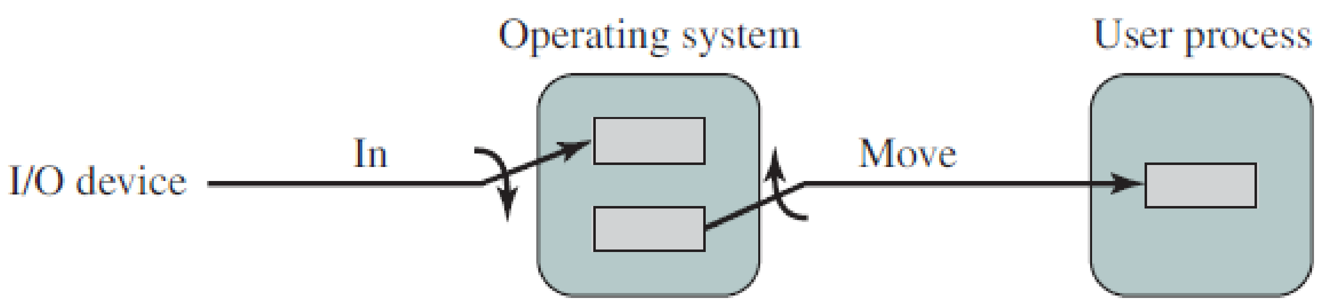


Figure 8.7: The Scheme of Double Buffering

I/O Buffering (cntd…)

Circular Buffer:

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

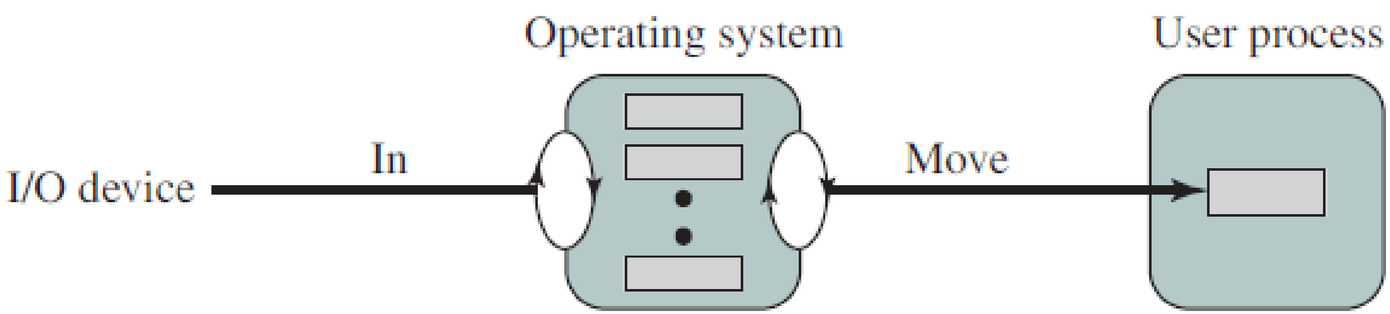


Figure 8.8: The Scheme of Circular Buffering

I/O Buffering (cntd…)

The Utility of Buffering:

- A technique that smoothens 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 and the performance of individual processes

Disk Scheduling

Disk Performance Parameters:

- The actual details of disk I/O operation depend on the:
  - Computer System
  - Operating System
  - Nature of the I/O channel and disk controller hardware

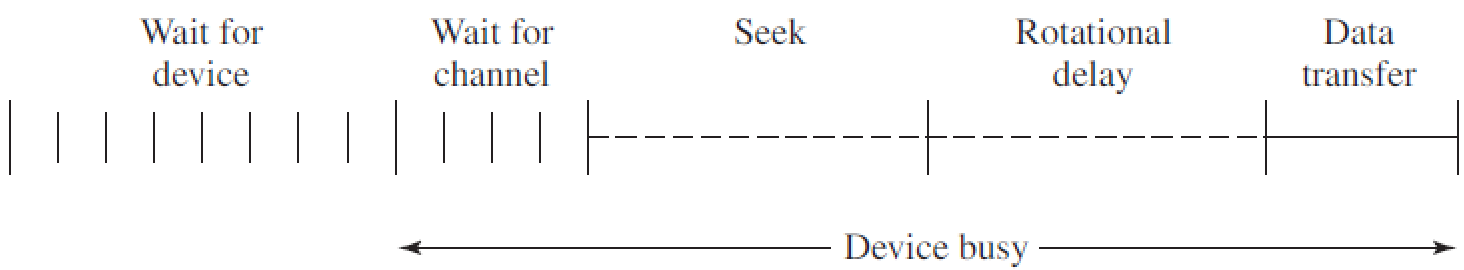


Figure 8.9: Timing of a Disk I/O Transfer

Disk Scheduling (cntd…)

Disk Performance Parameters (cntd…):

Seek Time:

- The time required to move the disk arm to the required track
- It consists of two key components: the **initial startup time** and the **time taken to traverse the tracks** that have to be crossed once the access arm is up to speed
- Unfortunately, the traversal time is not a linear function of the number of tracks but includes a **settling time**

Rotational Delay:

- The time required for the addressed area of the disk to rotate into a position where it is accessible by the read/write head

Disk Scheduling (cntd…)

Disk Performance Parameters (cntd…):

Transfer Time:

- The transfer time to or from the disk depends on the rotation speed of the disk in the following fashion:

$$\boxed{T = \frac{b}{rN}}$$

Where,

T = Transfer time

b = Number of bytes to be transferred

r = Number of bytes on a track

N = Rotation speed, in revolutions per second

Disk Scheduling (cntd…)

Disk Performance Parameters (cntd…):

Thus, the total average access time can be expressed as

$$\boxed{T_a = T_s + \frac{1}{2r} + \frac{b}{rN}}$$

Where T_s is the average seek time

Disk Scheduling (cntd…)

Positioning the Read/Write Heads:

- When the disk drive is operating, the disk is rotating at a 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**

Disk Scheduling (cntd…)

Disk Scheduling Algorithms:

1. First-In, First-Out (FIFO):

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

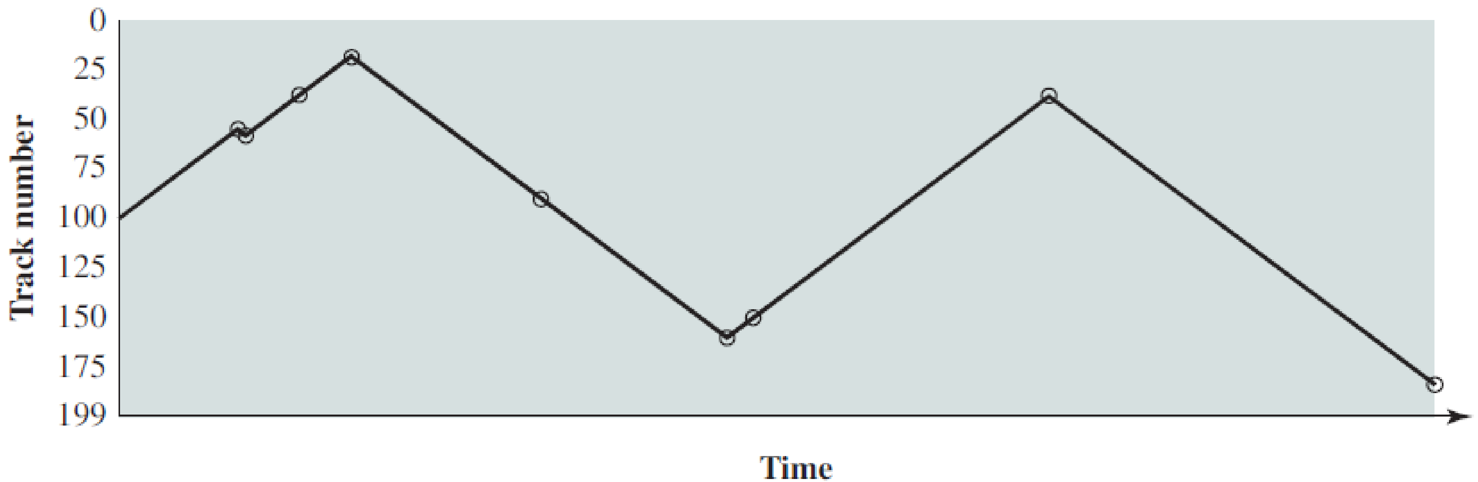


Figure 8.10: The Disk Arm Movement with FIFO

Disk Scheduling (cntd…)

Disk Scheduling Algorithms (cntd…):

2. Priority (PRI):

- Control of the scheduling is outside the control of disk management software
- The 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

Disk Scheduling (cntd…)

Disk Scheduling Algorithms (cntd…):

3. 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

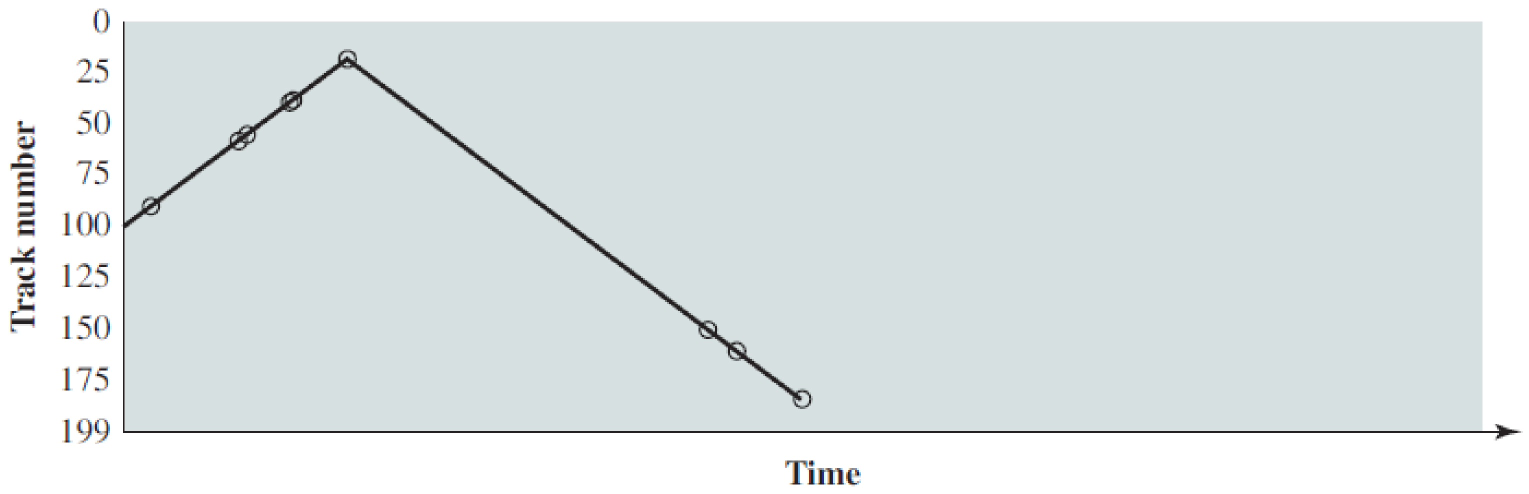


Figure 8.11: The Disk Arm Movement with SSTF

Disk Scheduling (cntd…)

Disk Scheduling Algorithms (cntd…):

4. 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

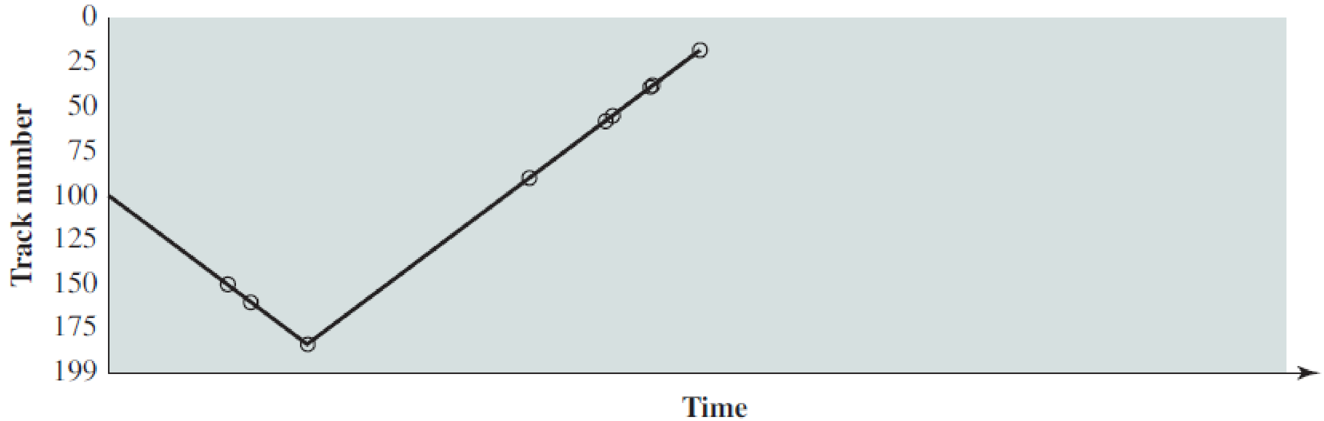


Figure 8.12: The Disk Arm Movement with SCAN

Disk Scheduling (cntd…)

Disk Scheduling Algorithms (cntd…):

5. Circular SCAN (C-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

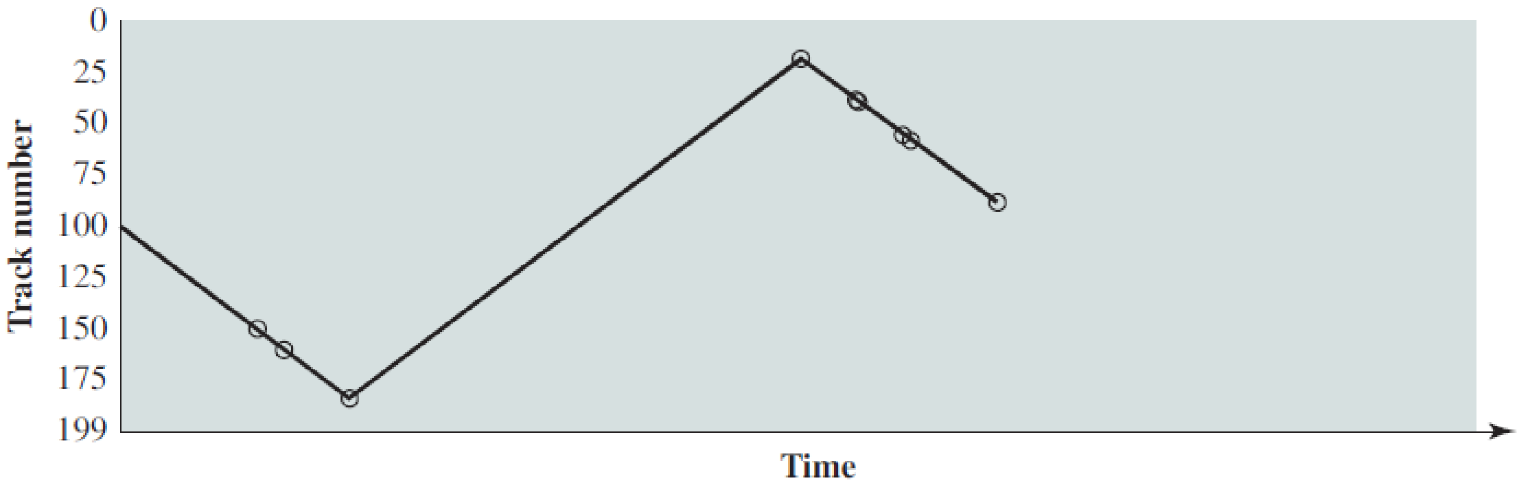


Figure 8.13: The Disk Arm Movement with C-SCAN

Disk Scheduling (cntd…)

Disk Scheduling Algorithms (cntd…):

6. N-Step-SCAN:

- Segments the disk request queue into sub-queues of length ***N***
- Sub-queues 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

Disk Scheduling (cntd…)

Disk Scheduling Algorithms (cntd…):

7. FSCAN:

- Uses two sub-queues
- When a scan begins, all of the requests are in one of the queues, with the other empty
- During the 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

Disk Scheduling (cntd…)

Disk Scheduling Algorithms (cntd…):

Comparison of Disk Scheduling Algorithms:

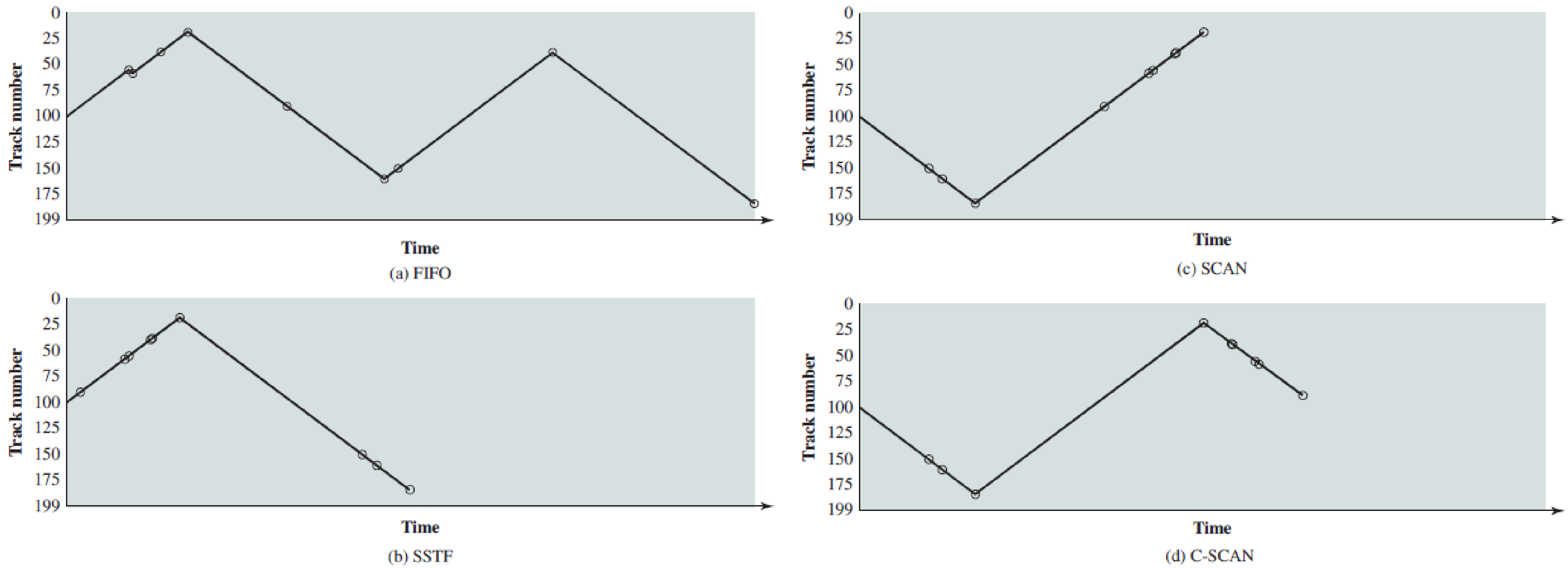


Figure 8.14: Comparison of Disk Arm Movement with Different Disk Scheduling Algorithms

Disk Scheduling (cntd…)

Disk Scheduling Algorithms (cntd…):

Table 8.3: 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 (cntd…)

Disk Scheduling Algorithms (cntd…):

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

Table 8.4: Summary of Disk Scheduling Algorithms

End of Chapter 08